COMP6204: Software Project Management and Secure Development

Charlie Britton

January 16, 2024

Contents

0.1 Module Introduction

0.1.1 Overview

Lecture 0: Introduction

2023-10-02T12:00

This module is to prepare students for undertaking large software projects. It introduces the students to the high-level strategies required for managing projects from their genesis to completion. The module also introduces the students to secure engineering of software systems.

We look at models of software projects; cost estimation; contracts, planning and monitoring; and costing and budgeting. We also look at security models, secure computing, threat modelling for designing secure systems, security and trust issues in software system design and popular threat modelling techniques.

The module is split into two parts, with the project management being one and the secure development of software applications being the other.

The module is similar in many aspects to previous modules that are required as part of the MEng course, including COMP2211 (Software Engineering Group Project), COMP2216 (Principles of Cyber Security), COMP3226 (Web and Cloud-Based Security), and tangentially COMP3217 (Security of Cyber Physical Systems).

The lecturer recommended possibly changing the module for a different one if possible as it shares a lot of the content and will likely be quite boring for those that have done similar work before.

0.1.2 Assessment

Labs

This is a slot every Friday from 4-5 p.m. This is for communication when we have group coursework but not for any fixed purpose.

Coursework – 30% Weighting

This is mainly on the project development side of the module. This will involve producing some project planning documents for an IT project. We then review

two other groups and give feedback on theirs. We do not get to choose the groups we are allocated to. This will be allocated in the second week of the semester.

Examination – 70% Weighting

This is a 2 hour, closed book exam with two sections. The first section is for short answer questions. The second section has four questions and we choose 3 out of 4.

The exam will typically be a 50/50 split of the secure development and project management parts.

0.1.3 Textbooks

The lecturer recommended the following textbooks:

- Project Management Essentials by Kalpesh Ashar
- Agile Essentials by Kalpesh Ashar

These are not available outside of official sources as far as I can see and do not exist in the UoS library.

0.2 Definitions

0.2.1 Project

A project is defined as a temporary endeavour undertaken to create a unique product, service or result.

A project might be a new product development, enhancement of an existing product, market research, a feasibility study, developing a software application or constructing a building.

A project might be started for a wide range of things. It could be due to market demand, technology change, a legal or social mandate or quite commonly as an internal organisational need.

0.2.2 Project Management

This is the application of knowledge, skills, tools and techniques to manage a project, which is started with the intention of meeting certain objectives.

The role of project management may fall on a specific department, commonly known as a project management office (PMO), or may be something that a specific person within a team undertakes.

0.2.3 Projects, Portfolios and Programs

A group of projects related to each other is a program. A group of projects grouped together at a higher level might be the portfolio.

Programs are within portfolios, with the portfolio being all the work that has the same strategic objectives. The program could be something like a game which or an application which has different sub-projects such as iOS, Android, MacOS, etc.

When an application is developed, supporting the application after the fact would not be a project, but an ongoing operation.

The difference between an operation and a project is that the project has some set of boundaries whereas an operation doesn't have a finite time limit nor necessarily a fixed scope.

0.2.4 Constraints

Projects will first have a scope, with a given time limit, budget, allocated resources from the budget (e.g., staff, computers, materials). The goal is to achieve the project to a certain set of quality standards. As a final part of the definition of the project, we define the risks to the project and a set of mitigations for these risks.

0.2.5 Stakeholders

Projects will have a wide range of people that either need the project for their own usage or are involved either directly or indirectly in the development efforts.

Projects may have sponsors, who fund the project but have no direct control, a customer, an end user, a project team, the project manager, the organisation performing the work, the government or other statutory bodies, pressure groups (environment, social, political) and society itself.

0.2.6 Organisational Structures

This varies dependent on the organisation, and will differ from a consulting company to a manufacturing company for example. They typically will fall into one of 3 groups:

- **Functional** – The structure is based on the functions or departments. Each section or department will specialise in one functional need.
- **Matrix** – The structure is a derivation of the functional structure, allowing cross-functional projects that would span many departments.
- **Projectised** – Teams are created for the project and the team is the only one to work on the project.

There are many more types of organisational structures.

0.2.7 Product/Service Lifecycle vs. Project Life Cycle

After the main development is done, the product will be done. We then transition from the project life cycle to the product life cycle which contains the support of the product and the eventual end-of-life procedures.

The lifecycle of a project has the following stages:

- **Start-up** – the project has lots of work done on it and the initial sales start to come through. This takes a while as there is a lot of engineering effort to get to the minimum viable product.
- **Rapid Growth** – the project has a few customers and more are starting to come in as new features are added.
- **Maturity** – at some point the product stops gaining customers quickly and the current customers are supported. Product growth is small and features are seldom added.
- **Decline** – at some point, the number of customers begins to decline as other products reach the market and the product is eventually declared end-of-life.

0.2.8 Project Management Phases

A project will have many phases, which may happen concurrently or out of order. We have the initiation of the project, the planning phase, execution, monitoring & control and finally the closure of the project.

Typically, the monitoring and control phase is done at the beginning and is an ongoing part of the project until its conclusion. The execution phase is typically the longest aspect of the project and is putting the plan into motion.

These phases may happen concurrently and there will typically at least be some kind of overlap. The execution phase will use most of the budget for the project as this is where the plan is put into action and all resources besides the planning usage are paid for.

0.2.9 Secure Development

This is the new part of the module, and focuses on interleaving the security aspects of a project with the rest of the development, and incorporating it into the full lifecycle of the project.

The target of attacks has changed from the traditional focus on the OS and the network, to web applications, the browsers themselves, mobile devices and the embedded software they use.

We need to consider the threat landscape of the modern hacker, and if they are funded through a nation state or bad actor, or if they are simply a hobbyist doing it for fun.

Historically, projects didn't have security in mind throughout the project and therefore the organisation had to rely on a reactive policy, which is much more costly than simply building in steps to prevent attacks in the first place.

For project managers, these manifest as additional communications requirements for the project, further linkage between different activities in the lifecycle and the nature in which the product will be used as a relation to the security needs of the product.

Within a project, we have flaws and bugs. The former is through lack or proper planning and is a defect in the design of the software itself. The latter is a defect in the implementation of the planning documents.

0.3 Motivations to Take This Module

Lecture 1: Introduction to Project Management

2023-10-04T11:00

This module is designed to equip us with the skills and techniques that are required to successfully manage projects. Project management is an important part of industry and has substantial economic contributions and benefits.

GDP contributions from project-oriented industries are forecast to reach $34.5 trillion by 2030. Employers will therefore need approximately 25M new individuals working in project-oriented roles by 2030.

As project-based industries grow, so will the need for project managers. In a US-skewed compensation figure, the average compensation would be $124,000 for a project manager.

Project management professionals can take an accreditation course to become a Project Management Professional. Managing your own projects is also a good skill to have.

0.4 Notes Layout

The following notes will be split into the two broad parts of the module. In addition to these notes, there will be an Anki package made available with all the lectures. This can be found at https://chza.me/notes/6204/.

Chapter 1

Project Management

1.1 Motivations to Use Formal Project Management

Using project management can give the company the following advantages:

- Better control of financial, physical and human resources
- Improved customer relations
- Shorter development times
- Lower costs
- Higher quality and increased reliability
- Higher profit margins
- Improved productivity
- Better internal coordination
- Higher worker morale

Without some form of project management in any business with more than a couple of employees, companies would really struggle to make effective use of their staff and keep track of what work is being done within the company.

1.2 Differences Between Projects and Operations

As discussed briefly earlier, a project is a temporary endeavour whilst an operation is an ongoing concern within the business. To give further examples, a project might be a new payroll system for the company, whilst an operation could be the payroll processing that happens once a month.

Further examples include creating new buildings, designing new cars, or developing newer versions of software as examples of projects, whereas maintenance of buildings, producing cars and supporting newly created software would be operations.

Simply remember that a project is temporary, whilst an operation is ongoing.

1.3 Project Attributes

Projects have unique purposes within businesses and must be temporary although it can stretch for a long period of time. A project will drive change and create value for the company or customer. Most projects are developed with progressive elaboration of an idea or in an iterative fashion, adding new features over time. Take for example a software project, where the company that develops it will iteratively add features with every new release.

Projects require resources from the company in different areas (e.g., staff, computing, materials). The project will need to have some way of funding these resources and this can come in the form of a customer or sponsor. A sponsor provides the direction and funding for a project.

As hard as we might try, a project will not be well defined from the outset and over time the scope is refined. Initially, the project will typically be quite uncertain, with only vague guidelines and a vague budget and as the project goes on, the goals will become more clear.

Project managers work with the sponsor, the team and the other people who are involved in the project to define, communicate and ensure that the goals of the project are being met.

1.4 Constraints

A constraint is a restriction on the project in some way or another. Many project managers will focus on the "triple constraint", consisting of:

- **Scope** – what work is to be completed as a part of the project? What unique product, service or result will the customer or sponsor expect from the project?
- **Time** – how long will it take to complete the project? What is the timeline?
- **Cost** – what should it cost to complete the project? What is the budget– the most that we can spend on the project and what resources are needed?

In addition to the "triple constraint", we also have quality, risk, and resources. These were given in the introductory notes in the last session.

The role of the project manager is to optimise against a set of constraints as they run the project.

1.5 Project Management Framework

The PMI institution provides a framework for project management. Within this framework, they provide process groups, knowledge areas and tools and techniques we should follow if we want a successful project.

Stakeholders have already been defined earlier in this document, in addition to most of the knowledge areas. Missing from the earlier definitions are the integration, schedule, communications, procurements and stakeholders.

1.5.1 Knowledge Areas and Tools/Techniques to Help

Project Integration Management

Project integration management is a function that coordinates the work of all other knowledge areas.

Scope Management

The scope management involves working with the stakeholders to define and agree the work that will be undertaken, then manage the work required to complete the project successfully.

Scope management can be helped through the use of a charter, a scope statement, and a comprehensive work breakdown structure (WBS).

Time Management

As part of the knowledge areas, we also have to think about how to manage time, in terms of each aspect of the project and the project overall. We then use our knowledge and historic experience to develop a schedule and make cost-effective use of the resources, including timely completion of the project.

To aid with time management, we can make use of Gantt charts, network diagrams or critical path analyses. These allow us to ensure that all dependency items are finished on the project and that we can work in parallel on features that don't have dependencies on one another.

Cost Management

The cost management aspect is comprised of preparing and managing a budget for the project. To aid in management, we can make use of some techniques introduced in COMP3219 (Engineering Management and Law) to see whether projects are worthwhile in the long run. These include NPV, cost estimates and earned value management. These will be discussed later in this section.

Quality Management

Quality management ensures that the project will satisfy the requirements and needs in which the project was initially given.

Resource Management

We then also have resource management, where we make use of the people and resources given for the project.

Communications Management

Communications management is where we generate, collect, disseminate and store project information.

Risk Management

Risk management is the identification, analysis and response to the risks we may encounter within the project.

Procurement Management

Procurement management is the acquisition or procurement of goods and services which are outside the boundary of the organisation.

Stakeholder Management

Finally, stakeholder management is the identification and understanding of stakeholders and their needs, expectations and engagement throughout the project.

In the introductory lecture, whilst some stakeholders were recognised, we also have the support staff, suppliers and opponents to the project on a more broader sense.

Agile Project Management (Specific)

Agile is a way of working through a software project. In these projects, we create a product roadmap with all the features we need to develop, assigning them to a project and a sprint backlog, then tracking our progress relative to where we think we should be with a burndown chart. We then take time to reflect on the previous sprint with a retrospective.

1.5.2 Defining Project Success

Many projects fail or do not live up to their original goals. An important part of management is defining success and what the customer/sponsor might think is a success in comparison to what the contracted company performs.

There are different ways focusing on a variety of key performance indicators. Did the project provide value? Did it meet scope, time and cost goals? Is the customer or sponsor satisfied with the work? Will the customer recommend your company to others or use your services in the future.

Finally, did the project produce the desired results?

1.6 Project Selection

When doing a project, we use a selection method. We can undertake a project for a range of reasons but qualitatively, we may choose to do this as part of the brand image, to match our complementary products and services, extend the existing portfolio of current products/services, and produce your own product in a competitive landscape (this is likely to be harder if there are already many products in the market).

There may also be quantitative objectives and a project will typically be a mixture of the two.

1.6.1 Discounted Cash Flow

When assigning quantitatively, we estimate cash flows that the product will need and then generate over a given period. We can make use of DCF (discounted cash flow, as discussed in COMP3219 Engineering Management and Law) to see if the project will make the company money over its lifetime.

If the project needs to borrow working capital, and the working capital can be returned with 10% interest, then the cash flows need to generate a greater than 10% return or else the project will run at a loss (and thanks to the wonderful system of capitalism that is in place, we won't fund projects that aren't profitable).

Example. A company has estimated cash flows of:

When	Year 0	Year 1	Year 2	Year 3	Profit
Value Today	-$10M	$2M	$4M	$6M	**$2M**
DCF	-$10M	*$1.818M*	$3.306M	$4.508M	**-$0.368M**

Table 1.1: Cashflow in terms of today's money, DCF and overall profit

The italic value in the table is calculated using the DCF formula:

$$\begin{aligned} \text{DCF} &= \frac{\text{Cash Flow}}{(1+r)^n} \\ &= \frac{2000000}{(1+0.1)^1} \\ &= 1818181 \\ \textit{where } r &= 0.1 \text{ (10\% year on year cost of capital)} \\ n &= 1 \text{ (years since start of operation)} \end{aligned}$$

This is because we need to discount the future values to bring them to a current value which equals the cost of the capital. Using this formula for the remaining values in the table and then calculating the sum yields an NPV of -$0.368M, which would generate a loss.

Using the above method, we can then make use of the net present value (NPV) of the project. The higher the NPV is, the better it is from a quantitative point of view. If NPV < 0, then it will not provide returns to the company.

1.6.2 Internal Rate of Return

We also have IRR (Internal Rate of Return) which gives the return but in terms of the percentage.

If IRR is above the cost of the capital, then the project will make money. If there are two projects, one with IRR of 15% and one of 20%, and cost of capital is 10%, then both are viable but capitalists would pick the 20% project as it provides a higher return.

1.7 Project Charter

Once a company selects the projects to undertake, it has to authorise the projects. The project charter provides an authorisation to carry out the project. The charter is a short (1-2 page) document issued by the sponsor (internal/external), giving a high level overview of the project. It will typically include:

- Project Name and Description
- Deliverables and Constraints
- Business need for the project
- Assumptions
- Justification for the project
- High-level Risks
- High-level Requirements

- The Project Manager and the Stakeholders

1.8 Stakeholder Identification

Stakeholders are an important part of the project and ensuring that we have all of them in a list to enable contact throughout the project is very important.

Following the Project Charter, a list of stakeholders, known as a stakeholder register is prepared, with a strategy for managing them. A list could look similar to the following:

ID	Name	Organisation	Contact Info	Role	Main Expectations	Management Strategy
1	John Doe	ACME Corp.	+44 7123 456789	Customer	...	...

Table 1.2: Sample stakeholder register

1.8.1 Managing Stakeholders with the Power-Interest Grid

In a project there may be many stakeholders, each with varying interests and levels of importance within the project. The grid is as follows:

Power/Influence	Interest	
	Low	High
High	Keep Satisfied	Manage Closely
Low	Least Effort	Keep Informed

Table 1.3: Sample Power/Interest Grid

1.9 Agile

Lecture 2: Introduction To Agile Project Management

2023-10-05T10:00

Agile is a specific form of project management that is widely used in software projects. It has been introduced in several previous modules and was used as the basis for the Software Engineering Group Project.

The Agile Alliance defines Agile as "the ability to create and respond to change". Others define it as "marked by ready ability to move with quick easy grace", or "a term used to describe a mindset of values and principles as set forth in the Agile Manifesto".

Agile is a concept that states how work should be done. The work completed using Agile for project management can be part of a project or part of regular operations. It has several implementations, or methodologies which can be used together or in isolation.

Whilst Agile is a form of project management, it is not recommended for all kinds of work. There are two prerequisites to get the most out of Agile. Work must be uncertain/flexible in scope from the beginning and involve what is known as a 'knowledge worker', or somebody/something with highly skilled resources.

The reason to have a team made of knowledge workers is to allow the project manager to give a lot of authority and decision making scope to the team that they are overseeing.

1.9.1 Why Agile?

Agile is not the only form of project management but has evolved over the years to replace traditional development cycles, and replaces the requirements gathering solely at the beginning of the project, followed by a development plan, then implementation with something more flexible.

You can imagine that by only gathering the requirements at the beginning and working on them sequentially might yield a product that is flawed in design or doesn't have a working version until somewhere near the end.

This type of development lifecycle may be known as 'The Big Bang Approach', or a Waterfall stage of development.

By the time the customer sees what has been developed, it is typically too late to make major changes to the design without incurring huge financial and time setbacks.

In an ideal world, the customer wants to be able to request changes at any part of the development lifecycle, with these changes not being too cumbersome to implement. The customer can only request changes with the product if they can see it in action.

In a traditional approach, the team is not customer-centric and will want to create a plan and then focus on implementing that plan.

1.9.2 Incremental Improvements

With the Agile system, work is divided into workable chunks, which can yield some sort of feature. A feature is comprised of various sub-features, and allows us to build one feature out before moving onto the next.

1.9.3 Iteration

Another of the characteristics of Agile is that it is an iterative flow, where the initial parts of the system are built quickly, allowing the customer to get feedback in to the team before too much time is invested in the system.

The system is built slowly with minimal functionalities to start with and through various iterations, it slowly improves over time until it is in an acceptable state for the customer.

1.9.4 Adaptive

Another feature of the Agile methodology is the ability to be able to change course quite rapidly. The customer way suddenly wish for a feature to not be worked on or may prioritise another feature. The team can then adapt to this.

1.9.5 TDD: Test Driven Development

Unlike traditional software development processes, such as waterfall, the Agile approach works in testing from the outset, allowing each small improvement to the product to be tested against requirements. In a traditional development cycle, the testing happens at the end, once we have done all the design, code and test aspects of the project.

If there is a flaw in the design this becomes costly to correct.

1.9.6 Myths

Being a popular project development tool, Agile has lots of misconceptions. Projects using Agile can still fail and it is not a silver bullet to ensure the success of the project. Further to this it is not a substitute for the correct documentation, a plan of action, or good team discipline within the team.

Despite misconceptions, it also doesn't require a lot of rework of features that have been created already.

1.9.7 The Agile Manifesto

This is comprised of four values, twelve principles, and an unlimited number of practices.

Agile Values

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

Agile Principles

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity–the art of maximizing the amount of work not done–is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

1.10 Agile Methodologies

The values and principles of Agile are implemented through several practices. These are defined as part of several specific Agile methodologies.

1.10.1 SCRUM

This is based on short iteration cycles, called sprints. As part of SCRUM, there is daily collaboration through the form of a daily meeting where people discuss their progress on tasks and anything stopping them. At the end of a sprint cycle (2 weeks, can range from 1-4), there is a retrospective where we incorporate anything learned during the sprint.

At the end of each sprint we should also have a product that is potentially shippable and can be used to show the customer the current product for their feedback.

SCRUM has several pillars:

- **Transparency** – All stakeholders have visibility to what is being done and where we stand.

- **Inspection** – we check the work progress towards achieving goals and correcting deviations early on.
- **Adaptation** – we ensure that any deviations are taken care of to avoid failure.

SCRUM Roles

We have the SCRUM master who is a leader responsible for ensuring that impediments to progress are removed and ensures that the SCRUM events due to take place happen.

They do not manage the team, merely performs the administrative tasks required to keep the team up to date and working to deliver items from the backlog.

The SCRUM master also serves as the point of contact between the product owner and the rest of the team, and communicates the vision, goals and backlog items to the team members.

The SCRUM master is different to a traditional project manager as the SCRUM master doesn't manage the project, just facilitates things on the project for the development team.

The Development Team

This consists of resources with skills that between them are able to deliver all the work. The team is self-organising and performs many functions. Ideal teams work with between 5 and 9 people.

The team should not change during the sprint as this will throw the sprint off but staffing can change between sprints.

Besides the team, there may be other stakeholders within the organisation or external but these are ignored in the SCRUM methodology.

Events

There are several events used in this type of project management:

- **Strategy Meeting** – This is a meeting which is held to discuss strategy, come up with a vision, define epics, user stories, create a backlog and a product roadmap.
- **Release Planning Meeting** – this is held each release to decide prioritisation of requirements and what should be included in the release. This is also where we put high level estimates against items in the backlog. This meeting is managed by the product owner.

- **Sprints** – iterations of between 1-4 weeks which build a potentially releasable product increment. Several of these make a release and several releases make a project.
- **Sprint Planning Meeting** – this is a meeting with the product owner to do estimations of stories in the product backlog and to decide what can be moved into the sprint backlog.

  Decisions on stories to include depend on priority and risks associated with the priority. We try to deliver stories with the most business value first.
- **Daily SCRUM Meeting** – This is a meeting for a maximum of 15 minutes where the development team speaks to answer what was done yesterday, what needs to be done today and any obstacles in the work.
- **Sprint Review Meeting** – This is at the end of the sprint and looks at the work that has been completed. The development team shows the done items to the product owner. This shows acceptance of the work or scope validation. Future plans may also be discussed.
- **Sprint Retrospective Meeting** – This is the last meeting after a sprint review. The SCRUM team discuss what went well and what went wrong. Any findings and learnings are documented to allow processes to improve.

Chronology of Events

To kick the project off, we hold a strategy meeting in which we create a backlog for the product. The work is then broken down into releases, with the release planning meeting and finishing with a product demonstration.

Within releases there are sprints which then have a sprint planning meeting to discuss what will occur, then during the sprint we have the daily SCRUM meeting.

After the sprint we have the sprint review meeting with the customer and then the sprint retrospective without the customer.

In Scrum it is not officially defined that there are sprints and releases but this is common practice.

Artefacts

As part of the development process, we create artefacts. These are things that add value to the customer. Besides these the team may decide what else might be needed and create these for the customer.

Artefacts are only created if they are necessary to add value to the customer.

Product Vision

This is a statement created during the strategy meeting to describe the business vision. This allows all involved to get a view about why the work is being undertaken and what can be expected from it.

Roadmap

This is also undefined in SCRUM but is a pictorial representation of features to be delivered in different releases.

Product and Sprint Backlogs

These are units of work that can be completed and are assigned priorities based on the business value compared to the risk. We may also have a release backlog that encompasses the release (collection of several sprints).

Definition of Done

This is a set of completion actions for deliverables within a sprint. It may include peer review, unit testing and checking against acceptance criteria.

Kanban

This is based on a Japanese production system at Toyota. We visualise the workflow, limit what is in progress, manage the flow of work, make process policy explicit and improve work collaboratively.

The Kanban board, such as that in Trello allow us to see the work at a glance, with the requirement on the card. These requirements are placed in a column based on where in the process they are.

We limit the number of work in progress items at each stage to prevent build up of unfinished tasks.

Epics

These are user stories at a high level and demonstrate an action to be completed within the application at a high level.

User Stories

These provide smaller actions that can be put into an index card and could be achieved at a lower level than an epic.

1.11 Systems Approach

Lecture 3: The Project Management Process Groups

2022-10-09T12:00

Projects operate in broad organisational environments. The managers of projects need to take a holistic view of the project within the organisation. To do this, they make use of a systems approach.

The systems approach emerged in the 1950s and presented an analytical approach to management and problem solving. The approach comprises three parts, systems philosophy, systems analysis and systems management.

Systems philosophy is the overall model for thinking of things as systems, the analysis is the problem solving approach to this and the management is what we use to address the business, technological and organisational issues that we may encounter, before we make changes to the system.

1.11.1 Three-Sphere Model

This takes three "spheres" of management. We look at the business side, organisation side and technological side of the system.

Example. When undertaking a project to introduce new tablets into a school environment, the organisation sphere would encompass the following questions:

- Will the project affect all students, or a subset of them
- How will the project affect those that already have laptops/tablets?
- Who will develop applications or books for those tablets?
- Who will provide the training?

We then also have the business sphere, which looks at the costs and impacts:

- What will the project cost the college?
- What will it cost the students?
- What will the support costs be?
- What will the impact on enrolments be?

Finally, we look at the technology sphere:

- What OS should the tablets be based on?
- What applications will be required?
- What will the hardware specifications be?
- How will the tablets affect the network?
- Will we need more power cords in each classroom?

1.12 Product/Project/Software Development Lifecycle

These are different life cycles with different definitions and phases. We have already discussed the difference between the product and project lifecycle. The software development life cycle (also known as the SDLC) is another aspect we can look at.

The product lifecycle is the process of managing the entire lifecycle, from inception, through design and manufacture, through to the service and disposal at the end of the life.

The project lifecycle is a part of the project, and as such is a temporary endeavour. This is where the main part of the engineering effort takes place, and is the application of knowledge, skills, tools and techniques to activities to meet the requirements.

Finally, the SDLC is the process for planning, development, testing and deployment of an information system. It is used, as the name implies, for the development and release of a software project.

Each of the three life cycles has differing phases, with the product life cycle being focused on the initial idea generation, screening, analysis to ensure it makes business sense, the product development cycle, marketing, commercialisation and evaluation and refinement of the end product.

The project management lifecycle focuses on the project aspect of a product, once it has been decided from a list of ideas and stops when the development of the project finishes.

Finally, SDLC focuses similarly to the project management lifecycle but in the application of a software product. We gather the requirements, design, develop, test and implement the system.

The product lifecycle is the longest running aspect, and the product development aspect of the product lifecycle is what we would class as the project life cycle. Within the project life cycle, we have the planning through to the monitoring and control phases, which would wholly encapsulate the SDLC.

We can think of these terms as layers which get progressively more detailed as we go through the chain.

1.12.1 Project Life Cycles

This is a series of phases that a project passes through from its start to its completion. A phase is a collection of logically relates activities that ultimately results in one or more deliverables.

1.13 Continuum of Development Approaches

There are many different ways to approach a development life cycle. We can rigorously define the product upfront in a predictive manner, trying to establish what needs to be done for the project as a whole. We then have iterative development life cycles, where the scope isn't very well defined at the start and we refine the approach through several iterations.

We also have an incremental development approach, where we focus on building on top of previous increments, adding new functionality.

Finally, we have the Agile cycle which is a combination of incremental and iterative approaches to the development lifecycle and is the least fixed of the approaches.

A predictive lifecycle may be known as a waterfall approach as the scope, schedule and cost are defined early on, with and changes carefully managed.

An iterative lifecycle may have a scope defined early on but the time and cost estimates will change as the project goes on and the understanding increases. Iterations provide development through repeated cycles which add functionality.

Iterative product life cycles are good where there is a high degree of change and low frequency of delivery.

Incremental life cycles give deliverables through iterations that add functionality in set time frames or chunks. A deliverable is not complete until after the final iteration. If the product isn't likely to change and is delivered frequently, this is a good approach.

Finally, there is the adaptive life cycle, where we define a detailed scope and produce a useable product at the end of each iteration. This forms the basis for an agile life cycle.

We can also have a hybrid life cycle, where we make use of a combination of the above life cycles, choosing the most appropriate one for each aspect of the product.

1.13.1 Predictive Life Cycle Models

There are many models for predictive development, including the waterfall model, spiral model (where the product is developed using an iterative or spiral approach), a prototyping model where we develop prototypes to clarify user requirements and a rapid application development where the developers work with an evolving prototype.

1.13.2 Project Management Process Groups

These have already been discussed in earlier notes but a process is a series of actions directed toward a particular result. This can be viewed as a number of

related processes.

Process groups involve initiation, planning, execution, monitoring and control and closing processes.

To measure performance of a project, we can track the time spent on each process group and compare it to the average for a project. This allows us to gain insight into how different times in different areas can work with or against other project areas.

For example, it was found that spending another 10% in the planning phase meant that less time was spent in the executing phase. The typical project will allocate the most resources and time to the execution phase, followed by the planning phase. Initiation and closing tasks have the smallest amount of time allocated to them, with the monitoring and control processes falling somewhere in between.

1.13.3 Alpha Project Managers

This describes ideal features in a project manager. They should have a good attitude to the project and belief. They communicate better, spend more time on the planning phase, manage relationship and conflict well, show good leadership, enjoy their work, handle emails better, are open to 2-way feedback and avoid conflict escalation.

1.13.4 Project management Process Group and Knowledge Area Mapping

This is a table with columns for each of the process groups and rows for each of the knowledge areas.

1.13.5 Initiating Process Summary

To initiate a project, it is first approved following a selection process in what is known as a pre-initiating stage.

For this aspect of the project initiation, senior management work together to determine most of the project parameters, such as the scope, time and cost constraints. During their meetings they also identify the project sponsor, select a project manager, develop a business case for the project, review processes/expectations for the project and decide if the project should be further subdivided into different projects.

All of the above processes are completed by senior management in a pre-initiating meeting. Once these are all done, the task is passed to the project manager, who will work to create the project charter and an assumption log, in addition to identification of the stakeholders.

The project manager will then hold a kick off meeting with the project team and the product owner.

1.13.6 Business Case for Projects

A project is unlikely to go ahead without a valid business case. This is a document that identifies the reason for initiating a project, including the value, benefit, and business problem it's designed to solve.

A completed business case for the project should provide structure and organisation throughout the lifecycle of the project. It should be used as a reference routinely.

The project sponsor and project board review and update the business case at key stages to ensure it remains viable and still has a valid business need. These key stages are typically between starting different stages of the project as we want to avoid wasting time and money.

Typical Contents

A business case is a document that will usually contain the following things:

- Introduction and background
- A business objective
- A current situation and a problem/opportunity statement
- Critical assumptions and constraints
- Analysis of options and recommendations
- Preliminary project requirements
- Budget estimate and financial analysis
- Schedule estimate
- Potential risks
- Exhibits

1.13.7 Project Initiation

This stage of the project includes recognition and starting of a new project. We want to ensure that we are taking on the right kinds of projects for the right reasons.

Strategic planning is one of the key things to do at this stage, as this will form the foundation for deciding which projects to pursue. We express the vision, mission, goals, objectives and strategies of the organisation for the project we want to undertake.

As a part of this, we develop the project charter, assumption log, identify stakeholders and hold a kick off meeting.

Initiation Processes and Outputs

As part of the initiation, we will have processes which generate outputs. These are detailed in the table below:

Knowledge Area	Initiating Process	Outputs
Project Integration Management	Develop Project Charter	Project charter, assumption log
Project Stakeholder Management	Identify Stakeholders	Stakeholder register, change requests, project management plan updates, project documents updates

Table 1.4: Initiating Processes and Outputs

1.13.8 Identifying Stakeholders

These are the people who are involved in or affected by the project activities. Internal project stakeholders include the sponsor, project team, support staff and internal customers. We also include top management, functional managers and other project managers internally.

Externally, we also have stakeholders. These can include the customers, competitors, suppliers and other external groups involver or affected by the project.

1.13.9 Stakeholder Management Strategies

As discussed in a previous lecture, stakeholders will have varying levels of interest and influence over a project. We need to derive a strategy for different power/interest levels in the project and make these a part of our management strategy.

An example is listed below:

Name	Interest	Influence	Potential Management Strategy
Joe Fleming	High	High	Joe likes to stay on top of key projects and make money. Have a lot of short, face-to-face meetings and focus on achieving the financial benefits of the project.
Louise Mills	Low	High	Louise has a lot of things on her plate, and she does not seem excited about this project. She may be looking at other job opportunities. Show her how this project will help the company and her resume.

Table 1.5: Potential Project Management Strategies

Categorising Engagement Levels of Stakeholders

Where stakeholders take an interest in the project, they may have different views on the project. A list of different categories is presented below:

- **Unaware** – unaware of the project and potential impacts on them
- **Resistant** – aware of the project and resistant to change
- **Neutral** – aware of the project and neither supportive nor resistant
- **Supportive** – aware of the project and supportive of change
- **Leading** – aware of the project and potential impacts. Engaged in helping it succeed.

1.13.10 The Project Charter

As mentioned previously, this is a document that authorises the project manager to use organisational resources to complete the project. It can sometimes be replaced with a letter of agreement or a formal contract.

Within a project charter there are many sections. One of the most important of these is the sign-off section, where authorisation is given for the project to be undertaken.

It includes the following aspects:

- Title and date of authorisation
- Project manager's name and contact information
- Summary schedule or timeline, with start and end dates.
- A summary milestone schedule if one is available
- A summary of the estimated cost and a budget allocation
- A brief description of objectives, including business need or other justification
- Success criteria or approval requirements
- A summary of the planned approach for managing the project
- Stakeholder needs and expectations
- Project risk
- Assumptions and constraints
- Roles and responsibilities matrix
- A sign-off section for key stakeholders
- A comments section for important stakeholder comments

1.13.11 Differences Between Project Charter and Business Case

Both of these have the same fundamentals. The main difference is scope and the area of focus. The project charter gives a description of the project and deliverables, whilst the business case gives a description of what the company is trying to get from a project in terms of ROI and future opportunity.

The project charter names and authorises the project for the project managers, but the business case justifies a company's decision on why we should take up a project.

1.13.12 Assumption Log

This is a document to record and track assumptions and constraints throughout the project lifecycle. It helps communicate information to key stakeholders, avoiding potential confusion.

Most projects have several assumptions affecting the scope, time, cost, risk and other knowledge areas. It is important to document and validate these assumptions.

A sample log is given in the table below:

ID	Assumption Description	Category	Owner	Due Date	Status	Actions
108	Shipping of materials will only take 2 days	Time	Kristin	Sep. 1	Closed	Require 2 day shipping
122	Employees will take some of the training during non-rowk hours	HR	Lucy	Nov. 1	Open	Meet with dept. heads to discuss

Table 1.6: Sample Assumptions Log

1.13.13 The Kickoff Meeting

This is the first meeting that is done as part of a project and is crucial to get projects off to a good start.

This meeting is held at the beginning of a project so stakeholders can meet, review the goals of the project and discuss future plans. The project champion should speak first and introduce the sponsor and manager. This is a meeting that benefits from a good amount of preparation.

Example. A sample project kick off meeting agenda:

- Introduce each attendee
- Review the project background

- Review project-related documents (business case/project charter)
- Discuss organisational structure
- Discuss project scope, time and cost goals
- Discuss other important topics
- List of action items from meeting
- Date and time of next meeting

1.14 Planning Projects

Lecture 4: Planning Projects

2023-10-11T11:00

T

his was recapped in the next lecture for some reason.

For any project, it is important to develop, refine and follow plans to meet project goals. People are more likely to perform well if they know what they need to do and when.

Project planning is a way to address how to complete a project within a time-frame, with defined stages and designated resources.

After a project is initiated, it moves into the planning stage. This is the part of the project where we set measurable objectives, identify deliverables, schedule the project and plan tasks.

It is often the most difficult and unappreciated process in project management. Many people don't want to take time to plan well, but theory and experience show that planning is important for good execution.

Plans must be realistic and useful in order to best serve their purpose. Planning starts with a definition of the scope. We then plan how to deliver the scope and the time it will take to do this.

Following the scope, we then estimate the detailed cost of the project work, followed by the quality, HR and communication requirements.

The various risks on the project are then identified and managed, and finally we create procurement documents for if we need to purchase products or services from outside sources.

1.14.1 Knowledge Areas Used

As with other areas of the project, it helps to look at the knowledge area, planning process and associated outputs of a project.

Knowledge Ares	Planning Process	Outputs
Project Integration Management	Develop Project Management Plan	Project Management Plan
Plan Scope Management, Collect Requirements, Define Scope, Create WBS	Scope management plan, requirements management plan, requirements documentation, requirements traceability matrix, project scope statement, project documents updates, scope baseline, project documents updates.	

Table 1.7: Knowledge area, planning processes and outputs for project planning

1.14.2 Project Integration Management

This is the process of coordinating all the project management knowledge areas throughout the project lifespan. The main output is a project management plan.

1.14.3 Project Management Plan

This is a document to integrate and coordinate all project planning documents and to help guide a project's execution, monitoring and control, and closure.

Plans in other knowledge areas are subsidiaries to the project management plan and give more detailed information about that knowledge area.

These facilitate communication amongst stakeholders and provide a baseline for progress measurement and project control. A baseline gives a starting point, measurement or observation that can be used for future comparison.

1.14.4 Attributes of Project Management Plans

Lecture 5: Planning Projects Continued

2023-10-12T10:00

These should be dynamic, flexible and receptive to change. These are unique and so are different for each project.

If we just have a small project with a few people, then the project plan won't need to be that big. We can probably just use a charter, team contract, scope statement and a Gantt chart. For a bigger project such as a multistorey scyscraper or a much larger software project, then we will benefit from having many different plans for each knowledge area.

Regardless of the size of the project, all plans will have some common elements. These will include:

- An introduction/overview of the project
- The project organisation
- Management and technical processes
- Work to be performed, i.e., the scope of the project
- Scheduling information
- Budget information
- References to other planning documents

As detailed in the above table, the project scope management knowledge area develops a scope management plan, detailing the scope of the product.

This plan tells us the features and characteristics of the product. This is one of the most critical areas as other aspects of project planning depend on it. If scope is missed or incorrectly defined, then the plan may be incorrect and have to be redone.

This is not good for the business, so we want to ensure we get it right the first time.

Planning the scope might be different in different projects. Some projects are smaller and thus require less planning. The project is given a business case or project charter and the high level requirements, then the detailed scope is planned later.

Other projects are won with bidding and so the scope is produced before the project is given to the contractor.

IT projects usually have an overview of the project given and then expect the solution provider to plan the entire scope.

1.14.5 Requirements Gathering

Once the project charter has been issued and the stakeholders have been identified, we can move on to requirements gathering. This may be done by visiting the customer, over emails, phone or other means.

The requirements gathering will use project resources and eat into the project budget. How the requirements are gathered and the templates for doing this will be given in the project plan.

Requirements may be organised in a spreadsheet with the following headers:

- Requirement ID
- Description

- Type
- Acceptance Criteria
- Requester Information, with names, roles and stakeholder IDs
- A Priority
- Whether it will be included
- Project Phase when to Implement

Once the scope is given in the requirements document, we then finalise the scope in a scope statement. This describes the characteristics and requirements of the project, the user acceptance criteria and project deliverables. Work outside the scope should not be done, and can be explicitly excluded from the scope through the project exclusions section.

The scope statement has a structure as follows:

- The scope description, giving the characteristics of the product of the project
- The product acceptance criteria, giving the characteristics and tests to fulfil or pass to accept the product
- A list of project deliverables
- A list of project exclusions
- Project constraints including the schedule, cost, resources, technology, quality and other expectations
- Project assumptions which are given to the stakeholders so everybody is on the same page.

1.15 Work Breakdown Structure

Once scope is finalised, then it needs to be understood, estimated, allocated and monitored by the project team. This is done by breaking it down into smaller more manageable chunks of work.

Decomposing the scope will create a hierarchical structure called a Work Breakdown Structure.

This is a document that puts the work into deliverables and groups them logically into tasks and sub tasks.

To create a WBS, we need to define the scope and objectives. We record the overarching objective. This could be anything from a new software feature to building a complex product. These details are documented in the project charter.

The WBS is broken down into key phases and deliverables. The project is divided into phases, large deliverables or sub tasks. We keep dividing into smaller and smaller pieces but stop before we list every single task that needs to be done.

Each deliverable is broken down into the tasks and sub tasks that comprise it. Sub tasks are organised into work packages. Work packages are called deliverables and are as low as we go in the breakdown. This is where we define the work, duratio and costs for each task, in addition to the owners of each task.

Each work package should provide assignments that can be completed within a reporting period.

1.15.1 Rules for a Work Breakdown Structure

- The WBS must include 100% of the work to be done.
- Each task and sub task is mutually exclusive
- We focus on deliverables and outcomes rather than actions
- The 8/80 rule – there are several ways to decide when a work package is small enough without being to small. Essentially, it should be no less than 8 hours of effort and no more than 80.
- It is about 3 levels deep
- Every work package should be assigned to a team or individual

Lecture 6: Schedule Management

2023-10-16T12:00

Last week, started with planning. To start this, we divide the project into subtasks and eventually some deliverables. This is called the work breakdown structure (WBS).

The WBS is a key part of the project management process. Usually a project is big and complex and has many aspects. The output of the project may be for example a car, which comprises the paint colour, airbag system etc.

Instead of focusing on one big project, we divide it into aspects, such as safety, security and performance for example. This is then further divided into differing criteria.

Continuing from slide 36 of last lecture.

Reza wants lots of colour as part of the WBS. Also recommends 3 levels, 5 maximum. If we have too much detail, then will be hard to remember and humans can't remember many details.

1.15.2 WBS Best Practices

Don't include verbs, WBSes aren't about actions but deliverables. It is also incorrect to show the activities on the WBS, need to only put the deliverables on. Activities shown on the schedule and not the WBS of the project.

The project manager and the team decide how to organise the work and how many levels to include in the WBS. Good to focus on the top levels of the WBS to avoid distractions from the detail.

The tasks on a WBS are not the specification. A WBS doesn't give a sequential list of steps. Must focus on what work needs to be delivered, not when or how it will be done.

1.15.3 WBS Dictionary

At the bottom level of the WBS, we have the work packages. A project has several work packages which may have hard to remember descriptions. Therefore, we create a one page document called a WBS dictionary, with anything the project manager wants to document.

For example, it may include the work package number, date of update, responsible person, description, acceptance criteria, deliverables, assumptions, resources, duration, planned cost, scheduled start and finish dates.

The time and costs of each package may not be fully estimated at the start of the project, so we may choose to fill in only some aspects of the WBS and leave the others for later in the planning stages.

1.16 Planning Project Schedules

Projects will have schedules, which we derive from estimation techniques. These are manifested as network diagrams or Gantt charts. To derive the order of tasks, we look at the critical path method to determine tasks which need to be completed before others.

We also look at the float of a project and the schedule of the project and how it can be compressed.

We have already looked at what will be delivered, and this is known as scope management. Schedule/time planning is determining when activities/tasks need to be executed, the time each takes, the sequence they are carried out in and how many resources each take.

For planning schedules, we define activities, sequence them, estimate durations and develop the project schedule. We give the schedule management plan, an activity list and attributes, a milestone list, a project schedule network diagram, activity duration estimates, a schedule baseline, a project schedule and a project calendar.

The schedule planning goes below the deliverables in a project as defined in the WBS and gives a list of activities which are actions taken by the team to do a distinct scheduled portion of work.

1.16.1 Sequencing Activities

As part of the project, we will have limited time and resources. We want to ensure that we deliver the project in a minimum possible time. To do this, we have to identify the relationship between different activities and put them into a sequence.

Some activities are dependent on each other and will need to be done serially, others may not be dependent and can be completed in parallel.

The sequencing of a project has a significant impact on the development and management of the project schedule. We try to minimise the amount of context switching that the team does as part of the project, by minimising the total number of parallel tasks on the go at once.

Network Diagram

When sequencing activities, we can create a diagram called a network diagram, which has a start and end box and can split from these where tasks are independent and those that are dependent on previous tasks can flow from the previous task.

Each node shows an activity and each arrow shows a dependency. Nothing from the WBS directly will appear in the network diagram as the network diagram is for activities and the WBS gives deliverables.

Mandatory, Discretionary, External and Internal Dependencies

Mandatory dependencies are inherent in the work that is being done on the project. You cannot hold training classes until the training materials are ready.

Discretionary dependencies are defined by the team and might follow good practices such as holding off on a stage of the project until stakeholders sign off on the previous dependency.

We also have external dependencies, where these are out of our control. Until this dependency is fulfilled, we may not be able to continue with parts of the project that rely on this dependency being fulfilled.

Internal dependencies are within the control of the project team.

1.16.2 Activity Attributes

In addition to activities, we need to document details about them. The description of the activity details is kept in an activity attribute. These contain

the details that need to be documented for better understanding of the activity during the execution phase.

Similar to the WBS dictionary, this includes a set of attributes. These include the name, identifier and a description of the activity. Attributes provide schedule-related information about the activities. Both the activity list and activity attributes should agree with the WBS and WBS dictionary and should also be reviewed by project stakeholders.

1.16.3 Milestones

These are key points in the project where a subset of deliverables or activities are completed. These are for management control of the project. These are combinations of activities and work. There is no additional cost of duration for a milestone. Sample milestones include signoff of key documents, completion of specific products, or completion of important process related work.

Milestones should follow SMART principles.

Lecture 7: Schedule Management Continued

2023-10-18T11:00

When using a network diagram, we can make use of activity on node (AON). This is a technique where boxes represent activities. This is widely used asit can show all dependencies.

There are a set of rules for creating an AON diagram. These are fairly logical and are as follows:

- Networks flow from left to right
- An activity cannot begin until all of its preceding activities are complete
- Arrows indicate precedence and can cross over each other
- Each activity has an incrementing unique identifier
- Looping is not allowed
- Conditional statements are not allowed
- We use start and stop nodes

To summarise the current understanding, we have nodes which represent activities. Activities can run in parallel. If an activity has two or more preceding activities then it is called a merge activity and if it has more than one activity immediately following it then we call it a burst activity. A milestone is a point in time when an activity is started or completed.

1.16.4 Sequencing Activities

There are four types of dependencies in an AON diagram:

- **Finish-to-start** – the successor activity cannot start until the predecessor activity finishes
- **Finish-to-finish** – the successor activity cannot be completed until the predecessor activity has been completed
- **Start-to-start** – the successor activity cannot start until the predecessor activity starts
- **Start-to-finish** – the successor activity cannot be completed until the predecessor activity starts

1.16.5 Lag and Lead

The most common dependency is finish-to-start, i.e., the next activity cannot start until the previous is finished.

In some cases, we may need to wait between the finish of the predecessor and the start of the successor. This is called lag. The lag can be found in all relation types between activities. Sometimes we can also start activities ahead of its predecessor's finish by taking a higher risk in an F-S dependency. This is called the lead and can only be found in F-S relationships. It may also be called negative lag.

Example. A photoshoot takes four days, the editing takes six. Instead of waiting until the end of the photoshoot, we might choose to start editing at the end of day one. This brings the total duration down from ten days to seven by leveraging the lead.

If there is then a fifteen day gap between proofs and printing, then we call this lag.

1.17 Estimating Duration of Activities

Once we have identified activities and their sequences, we need to know the duration to set the calendar dates and deadlines. The activity duration will depend on two things, the effort and the resources required.

The effort is the number of man-days or man-hours which is needed to complete the activity. The effort is the number of work units required to complete the activity.

Once we estimate the effort, the duration can be calculated based on the number of resources working on the activity. For example, an effort of 8-man hours split between 2 people would take half a day.

If we only have one person, then it would take one day. This can be given with the following super complicated formula:

$$\text{Duration} = \text{Effort}/\text{Number of Resources}$$

The key data for effort estimation will come from historical data, experts and the team members performing the activity. There are many different methods to calculate the estimates and the value will be discrete and not a range.

1.17.1 Parametric Estimate

$$\text{Parametric Estimate} = \text{Thumb Rule Per Unit} \times \text{Number of Units}$$

This is generally the most accurate estimate because we do it based on past projects and expert judgement. In a software project we may use the function point (FP) estimation. This is taking the number of functions that the application would require. This could be similar to lines of code or git commits. This is where we calculate the number of function points and a standard productivity chart to gather the final man days estimate of activities.

1.17.2 PERT: Program Evaluation & Review Technique

In parametric estimates, we use previous project data. This is only possible if the work has been done in the past. If the activity is new or significantly different, then it may be hard to produce an accurate figure. PERT has three estimates:

- Optimistic – labelled O, and is the estimate of the earliest possible completion time
- Most Likely – labelled M, and is the estimate of the most probable completion date
- Pessimistic – labelled P, and is the latest possible completion duration of an activity.

To calculate the PERT estimate, we make use of a formula to create a weighted average:

$$\text{PERT Estimate} = (O + 4M + P)/6$$

This gives the most weight to the middle estimate, with a small amount for the optimistic and pessimistic guess.

Example. We have an activity. $O, M, P = 5, 8, 10$. The PERT estimate becomes:

$$\begin{aligned}\text{PERT Estimate} &= (O + 4M + P)/6 \\ &= (5 + (4 \times 8) + 10)/6 \\ &= 47/6 \\ &= 7.833 \text{ days}\end{aligned}$$

We can also estimate something called the risk factor, which is a measurement of the standard deviation of a PERT estimate. This is calculated with the following formula:

$$\text{Risk (stddev)} = (P - O)/6$$

For the above example, this would be:

$$\begin{aligned}\text{Risk (stddev)} &= (10 - 5)/6 \\ &= 5/6 \\ &= 0.833 \text{ days}\end{aligned}$$

1.17.3 Creating the Schedule

We now have activities, dependencies and durations. It is now possible to set calendar dates and finalise the project schedule. We can put this on a diagram, such as that in Figure 1.1.

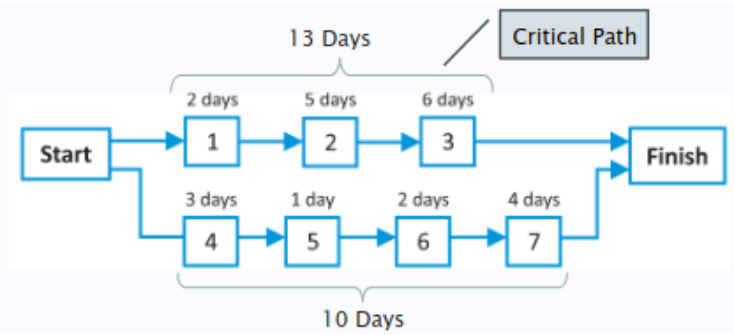

Figure 1.1: Network Diagram with Duration and Critical Path

Based on the start date, we can establish the dates we should be working on each branch and the finish date of the project. We introduce the critical path, which is the longest path from start to finish. This dictates how long the project and dictates where we need to ensure there is adequate monitoring and controlling of the activities on the path.

Any delays on the critical path will affect the whole project. If activities are to be run in parallel and resources are scarce, we can assign more people to the critical path.

Activities on the critical path will not have any "float" or "slack".

1.17.4 Float and Slack

These are measures of time that determines the flexibility or buffer time that an activity has in a project schedule. Activities can be delayed or extended by a

set number of days without affecting the completion date or the start of another activity.

There are subtle differences between the terms. The float is applied to a *path*, whereas the slack is within a single *activity*.

By recognising the float and slack, the critical tasks can be prioritised. We can adjust the non-critical tasks that have high float and slack.

1.17.5 Calculating Float and Slack with a Network Diagram

To do this, we need to perform two passes of a diagram. We take a forward pass through the network, calculating the earliest start and finish dates of tasks, with the backward pass taking the latest start and finish and starts at the end of the project.

The difference between the earliest and latest dates is given as the float or more appropriately the slack of each task.

Lecture 8: Schedule Management Continued the Second

2023-10-19T10:00

With each task in the diagram, we store a set of properties within the space around the task:

Est. Start	Task Name	Est. Finish
Slack/Float	–	–
Late Start	Duration	Late Finish

Table 1.8: Task "cards" giving the ID, start and finish, and the duration.

We can pre populate the cards with the estimated duration and start working forwards. For example, task A, being the first task after the start can be populated as follows:

0	A	*4*
	–	–
	4	

(a) Task A

$\longrightarrow$

4	A	$(4+8)=12$
	–	–
	8	

(b) Task B

Here, the italics show that B inherits its early start from the early finish of task A. We then add the early start and the duration to get the estimated finish.

In the instance we have a merge, then the larger number is taken on the forward pass. This is to allow for the float/slack later.

The reverse pass is similar to the forward pass, but we give the late finish the same time as the early finish and then work backwards. We take the smaller number on the backward pass, until we reach the start again.

The start should always give a late start of 0.

Following this, we can compute the slack/float as:

$$\text{Slack or Float} = \text{Late Start} - \text{Early Start}$$

Although we talk of float per activity, it is obvious that the late start of one activity in a chain will reduce the float of the remainder of the activities.

Example. There is a table below with the activities, durations and the following activities. Create an AON for this.

Activity	A	B	C	D	E	F	G	H	I	J	M	
Duration (Days)	4	5	6	8	3	7	11	4	8	9	4	6
Following Activity	D,E	E	F	H	K	K,G	I,J	M	M	M	M	–

Table 1.10: AON Network Diagram Table

This produces a diagram that looks like such:

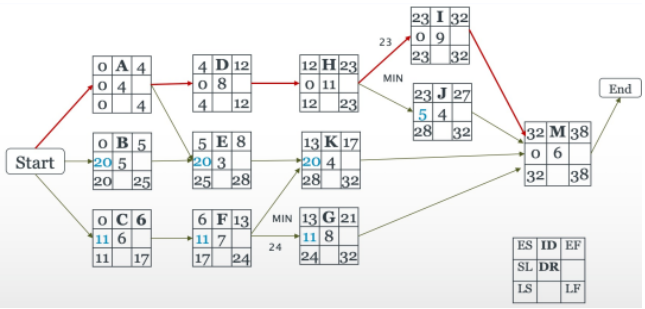

Figure 1.2: AON Diagram Produced from the above table

1.17.6 Near Critical Path

We have already discussed the critical path. Those paths with float close to the critical path are called near critical path. Say the critical path is 7 days long then a near critical path would be 8 or 9 days for example.

The near critical path task chains are the next priority for the project manager as these could become the critical path in the future.

Projects can have multiple critical paths. This increases the total risk as the PM needs to worry about possible delays to the project.

1.17.7 Gantt Charts

These are another popular way to present project activities but instead use long bars for each of the tasks, with longer bars representing longer tasks.

Milestones are given with diamonds and arrows feed from tasks into tasks that follow them.

1.17.8 Development of the Project Schedule

We make use of the results of all the preceding project time management processes to determine the start and end dates of activities and the entire schedule. This is often shown on the Gantt chart for the project.

The goal with development of the schedule is to create a realistic schedule that provides a basis for monitoring project progress from the time dimension.

1.17.9 Schedule Compression

sometimes the schedule doesn't fit within the timelines for the project. We can fix this in 2 ways. The first of these is called crashing and we increase the resources working on the critical path. These can be collected from the non-critical activities or outside hires. This will generally increase the cost of the project.

Another way we can compress the schedule is through fast tracking. This is where we remove some dependencies and do some activities in parallel. We then may have to rework the project because some dependencies have been removed.

1.17.10 Drawbacks of the Critical Path

Any delay on the critical path will delay the project. It assumes that the project has unlimited resources. The chance of completing a project on time are low.

As the duration of the project is based on the critical path, project managers will inflate the duration of activities on a critical path, giving an extra margin for completing the project.

Miscalculations of the critical path affect the schedule for the project. Therefore, instead of the critical path, project managers now make use of a critical chain method.

1.17.11 Critical Chain Scheduling

This is a scheduling method that considers limited resources for a project and adds time buffers to account for these limited resources. The critical chain minimises multitasking.

If we are heavily multitasking, then each task will take longer to complete. For example, 3 tasks of 10 days each would take 10, 20 and 30 days from the start respectively to complete, but if spending 5 on each at a time in a round robin fashion, then will take 20, 25 and 30 days respectively.

1.17.12 Critical Chain Buffers

This is additional time that is allocated to complete a task. As Murphy's Law states that if something can go wrong, it will.

Parkinson's Law states that work expands to fill the time allowed. For traditional estimates, we add a buffer to each activity to use whether it's needed or not.

By using critical chain scheduling, we remove the buffers from each time and instead allocate a project buffer, which is given as additional time before the project's due date.

We then allocate feeing buffers to tasks that are on the critical path.

We also have a resource buffer which is a resource alongside the critical chain to ensure the continuity of resources.

1.18 Project Cost Management

C

reated with ChatGPT

Lecture 9: Project Cost Management

2023-10-23T12:00

- Ensures projects are completed within an approved budget.
- The main tasks for this are Planning cost management, Estimating costs, Determining the budget.
- The deliverables for this are a cost management plan, Cost estimate, Cost performance baseline.

1.19 Planning Cost Management

- Set policies, procedures, and documentation to manage project costs.

- Activities include meetings, expert consultations, and data analysis to create a cost management plan.

1.19.1 Cost Planning

- Detailed project budget is determined after creating the project's schedule.
- Contrast budget with the approximate figure in the Project Charter.
- Adjust the budget or project scope accordingly if discrepancies exist.

1.19.2 Cost Management Plan Contents

- Units of measure, currency, inflation assumptions.
- Precision and accuracy levels for cost estimates.
- Organizational procedure links, Control thresholds for cost performance.
- Rules for performance measurement, Reporting formats and frequency.
- Additional details about cost activities.

1.19.3 Estimating Cost of Activities

- Costs are categorized as fixed or variable.
- Usually, only variable costs are estimated for project activities.
- Fixed costs are equipment or setup costs
- Materials or human resources are variable costs

1.19.4 Estimating Costs

- Refine cost estimates over time.
- Provide supporting details, including assumptions and ground rules.
- These may also be called the basis of estimates
- Significant portion of costs usually are labor costs.

1.19.5 Cost Estimating Techniques

- Analogous Estimates or top down estimates uses costs of past similar projects.
- Bottom-up Estimates Sum of individual activity costs.
- Parametric Modeling Uses project characteristics in mathematical models.
- We make use of detailed steps ground rules and assumptions in conjunction with internal and external experts

1.19.6 Cost Budgeting

- Allocating project cost estimate over time based on work breakdown structure.
- Aim to produce a cost baseline.
- Project budget is the sum of individual estimates plus reserves.

1.19.7 Reserves

- Contingency Reserve For known project risks.
- Management Reserve For unforeseen risks.

1.19.8 Project Budget Components

- Cost estimates + Contingency Reserves = Work Package Estimates.
- Work Package Estimates + Contingency Reserves = Control Accounts.
- Sum of Control Accounts = Cost Baseline.
- Management Reserve + Cost Baseline = Project Budget.

1.19.9 Example Budget

Activity	Fixed Costs (F)	Variable Costs (V)			Total Cost (F+V)
		Resource Cost Per Day/Cost Per Item (A)	Number of Days/ Number of Items (B)	Total Variable Costs (A × B)	
1	$1,000	$100	10	$1,000	$2,000
2	—	$400	15	$6,000	$6,000
3	—	$250	10	$2,500	$2,500
4	$20,000	—	20	—	$20,000
Total Direct Costs					$30,500
Indirect Costs (10% of Direct Costs)					$3,050
Project Costs (Direct + Indirect Costs)					$33,550
Contingency Reserve (10% of Project Costs)					$3,355
Cost Baseline (Project Costs + Contingency Reserve)					**$36,905**
Management Reserve (2% of Cost Baseline)					$3,690
Cost Budget (Cost Baseline + Management Reserve)					**$40,595**

Figure 1.3: Example Budget

1.19.10 Cost Baseline Notes

- Represented as an S-curve.

- Funding requirements may show uneven pattern due to incremental funding.

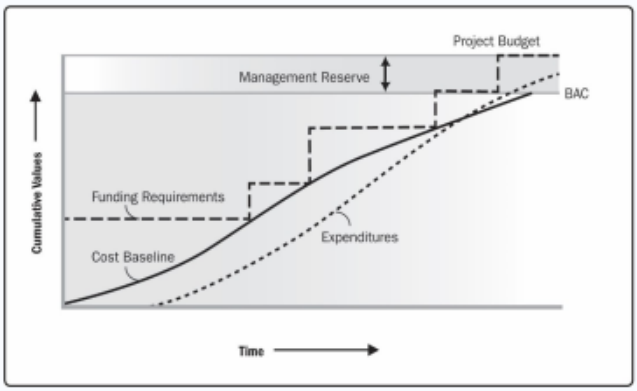

Figure 1.4: Scample S-Curve of the Cost Baseline and Funding Requirements

1.20 Quality

Lecture 10: Quality and Risk Resource and Communications

2023-10-25T11:00

Quality is a term that is misunderstood in projects. It can be equated to adherence to quality standards or providing the best possible features to a product.

Quality actually means conforming to requirements and being fit for use. The product meets written specifications and can be used as initially intended.

1.20.1 Metrics

Metrics are standards of measurements. To achieve quality, we plan quality assurance activities such as reviews, inspections and testing.

Metrics are the outputs of these reviews, inspections and tests and are compared with the acceptable ranges for the project.

Metrics can be used to measure performance in organisational areas and compare them over time or other organisations.

Examples include failure rates of produced products, availability of goods and services, and customer satisfaction ratings.

1.20.2 Benefits of Project Quality Management

Communicating with the customer ensures they are up to date and aware of issues. Incremental feedback also helps to deliver a better final product. Quality

management also leads to better quality products, increased productivity and financial gains for a project.

1.20.3 Quality Planning and Quality Management Plans

Quality planning identifies the quality standards that are relevant to the project and how they can be satisfied. We also then design quality into the products and processes.

The size and complexity of the plan varies with the project needs.

1.20.4 The Quality Plan

Each plan has a desired objective or goal. The goal of quality management needs to be communicated to the stakeholders of the product. After goal identification, measures to ensure the level of standard should be identified.

How will we measure satisfaction, what is the level of quality expected and how do we determine if the quality measures will lead to project success?

1.20.5 Quality Assurance

This is providing evidence to stakeholders to demonstrate that activities relating to quality are being done as defined and promised. The main goal of quality assurance is to evaluate if a project is moving towards delivering quality services.

To achieve this we can do a project audit, process checklist and tests. We make use of qualitative and quantitative measures to do this.

1.20.6 Quality Control

This is different to quality assurance and makes use of operational techniques to ensure quality standards.

If there is a problem with the quality or the execution of a quality plan, then we should take corrective actions. We monitor the project results and delivery to ensure we are meeting the desired results. If we are not meeting these results, then alternative actions should be implemented.

Quality assurance happens prior to an issue, whilst quality control is reactive and happens once we see a problem.

1.20.7 Quality Scope

We use a chart to keep track of metrics, which looks at all aspects of the project. This is sometimes called a project dashboard. We look at the product quality and the process quality to ensure that we are following quality guidelines for the system being created and to ensure we are following good practices.

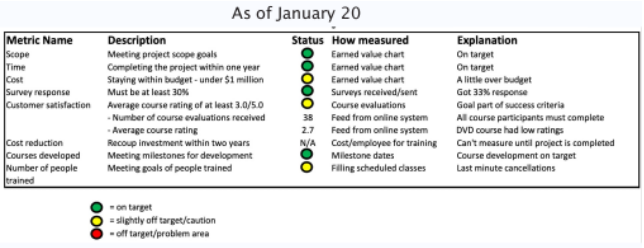

Figure 1.5: Sample Project Dashboard

An example dashboard might look like the following:

We can also have dashboards with more graphical views.

1.20.8 Quality Costs

Good quality should pay for itself in the long run. If we can be preventative instead of reactive to issues, then we spend less time fixing broken things. If a product is made with better quality, then we spend less time inspecting it. If we have a high quality product, there is less rework, defects, bugs etc.

As part of processes, we should build in inspection as part of the development lifecycle instead of at the end where it may be more of an issue.

If a defect is caught early, then it takes less time and money to fix it. We can quantify this with the costs of conformance and the cost of non-conformance.

This gets its own super duper complicated formula:

$$\text{Cost of Quality} = \text{Cost of Conformance} + \text{Cost of Non-Conformance}$$

1.20.9 Optimum Point

This is the point at which the cost of conformance exceeds the cost of non-conformance. If we gain £1 from conformance and this gives less than £1 in savings, then we are not working optimally.

1.20.10 Project Resource Management

Projects need resources. These can vary from people, to machines, to materials.

Human resources are usually the only resources used in services. Projects need to plan the required human resources. This includes the number of resources, their locations and the skills.

Project resource management makes effective use of the people in the project and the physical resources that the project requires.

1.20.11 Project Organisational Chart

This is similar to a company's organisational chart and is a graphical representation of authority and responsibility within a project specifically. This can be done by looking at the reporting relationship–who reports on their progress to who–or skill-based, where we organise based on skills.

Creating a structure to show these relationships is called an organisational breakdown structure (OBS). If we make this based on the skills within the team, then it is called a resource breakdown structure instead.

Example. A project manager forms the top of an organisational breakdown structure. They are reported to by three team leaders, each of which has between 2 and 3 team members. This is then organised into a similar structure as a work breakdown structure:

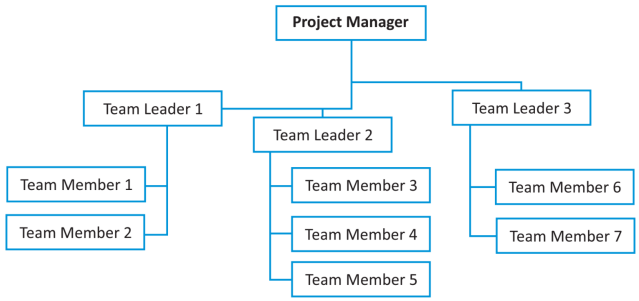

Figure 1.6: Sample Organisational Breakdown Structure

If, however, we choose to split work based on the skills within the organisation, we could end up with a resource breakdown structure such as the following:

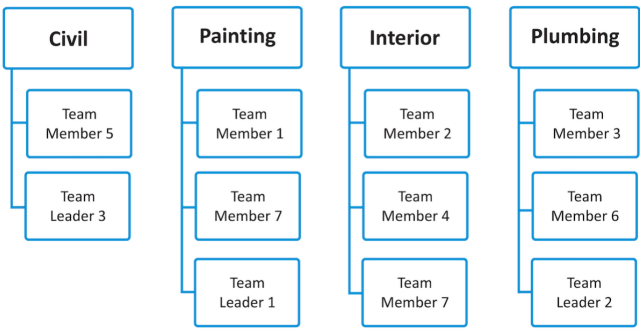

Figure 1.7: Sample Resource Breakdown Structure

These structures can be combined mainly under the OBS by adding the skill of

each member in brackets below their names in a typical OBS. For example, we might have:

Team Leader 1
(Painting)

as part of a combined structure.

1.20.12 Responsibility Assignment Matrices

These are matrices that map the work of the project from the WBS to the people responsible for performing the work.

In a smaller project, we should assign the activities to individuals. For larger projects, we can then assign work to organisational units or teams who will then further subdivide as necessary.

1.20.13 RACI Charts

These assign people to four different aspects of each task. It is another matrix, which gives who is **R**esponsible, **A**ccountable, **C**onsulted, and **I**nformed.

More than one person can be assigned to each of these fields and more than one of these fields can be assigned to an individual person.

We then generate a table with the activities and people, and their RACI assignments:

Activity	John	Mary	Leon	Jack
1	R	A	C	I
2		R, A		C,I
3	R	C	R,A	I
4		C	I	R,A

Table 1.11: Sample RACI Matrix

We can also have an alternative form, where the activities are in each row but the RACI assignments are given in each column with the person responsible in each cell.

1.20.14 Resource Histogram

Once the work has been assigned, we can come up with the staffing requirements in the form of a resource histogram. This gives us the staff requirements over the period of the project, given in a monthly increment:

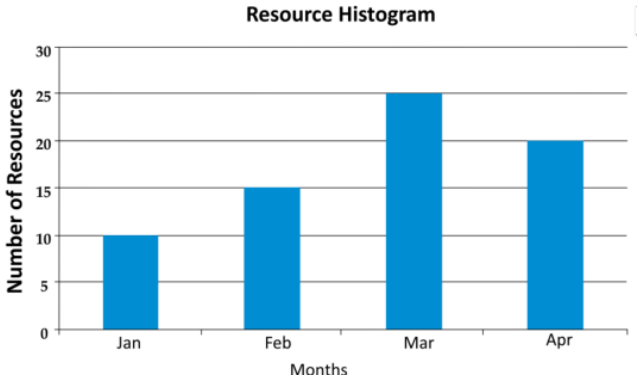

Figure 1.8: Sample Resource Histogram

Lecture 11: Project Resource Planning

2023-10-26T10:00

Resource histograms can be further broken down by the type of resource so we can get a baseline for each different type. This would typically be put into a table of each resource and the requirements each month. We then create a histogram with these different resources stacked to get he overall requirements for the project month by month:

		Month											
Type of Resource	Meaning	1	2	3	4	5	6	7	8	9	10	11	12
PM	Project Manager	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75
HR	Human Resources	1	1	1	1	1	1	1	1	1	1	1	1
SM	Supplier Management	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
IT	Information Technology	0.25	0.5	0.5	0.5	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
Contracting	Contracting	0	0	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
PMO	Project Management Office	0	0.5	0.5	0.5	0.5	0	0	0	0	0	0	0
Miscellaneous	Miscellaneous	0.25	0.25	0.25	0.25	0.25	0.5	0.5	0.5	0.5	0.25	0.25	0.25

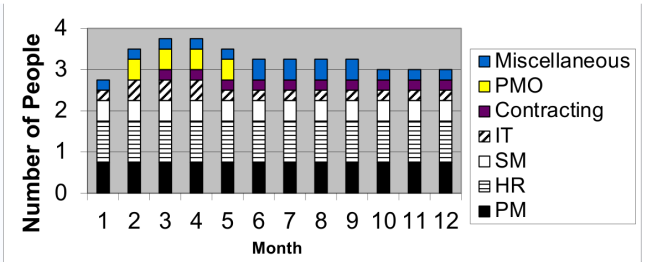

Figure 1.9: Sample Resource Histogram Broken Down by Type of Resource

1.20.15 Staffing Management Plans

These describe when and how people will be added to and removed from a project. We describe the hiring and releasing of resources, with hiring being internally or by outside recruitment of new resources.

We describe the types of people needed to work on the project, the numbers of

each type of person on a monthly basis and how they will be acquired, trained, rewarded and reassigned.

1.20.16 Estimating Activity Resources

This is where we estimate the type, quantity and characteristics of team and physical resources which are needed to complete the project.

This is similar to estimating activity durations and costs. We need to ensure that the estimations are done by people with experience and expertise in similar projects, or we will end up with a bad estimate.

These estimates are updated through the course of the project.

We need to look at various questions when estimating activity resources. We look at the difficulty of performing specific activities, anything unique in the project scope statement that affects the resources, the organisation's history in similar activities and the personnel and whether they have done similar activities before.

We also need to ensure we have the right people, equipment and materials for performing the work. We look at organisational policies that might affect resources, whether the organisation needs to acquire more resources, whether to outsource some work, and assumptions that have been made or need to be made.

1.20.17 Communications Planning

All projects need to communicate with their stakeholders. Details of communication are decided for each project during the planning stage.

A plan will typically include who will communicate the performance reports, which ensures that there's no redundancy or mismatch in the communications.

We also document what will be communicated, and agree upon the report formats and frequencies with the stakeholders.

The project team suggests the details for the reports and gain agreement from the stakeholders. Different stakeholders have different levels of reporting requirements, and some like detailed, whilst others like summaries.

We can then send these reports at the frequencies that the stakeholders require. Most stakeholders like a weekly report and a monthly report. We will gather this when communicating with the stakeholders initially.

Reports may be sent via letter, email or some other form of medium.

1.20.18 Effective Communication

Communication happens between the sender and the receiver. For communication to be effective, the message sent by the sender has to be understood

by the receiver. For verbal communications, the language, pronunciation and speed are important. For written communications, the language, spelling and handwriting are important.

We need to ensure that the sender uses the correct communication medium. For legally binding documents, such as contractual communication, we must use a written form or communication.

1.20.19 Project Risk Management

Risk is an uncertainty that can have a negative or positive effect on meeting project objectives. Risks may be negative or positive. We are typically worried about negative threats to the project as these can send the project off course and lead to a failure.

If we inadequately plan for risks, then we may cause project failures. The main planning processes for this area are identification of risks, performing the risk analysis and planning risk responses.

Identifying Risks

The most important step in risk management planning is to identify all the risks to the project. We determine which risks are likely to affect the project and document the characteristics of these risks. We then output a risk register and a risk report.

The most common ways to identify risks to a project are via brainstorming, historical information and checklists.

Risk Registers

A risk register is a risk management tool to identify potential setbacks within a project. We identify, analyse and solve the risks before they become problems.

Risk registers are usually planned around the projects but can also be useful around project launches and manufacturing.

Risks can include data security threats, communication issues, scheduling delays, unplanned work and theft of materials.

Risk registers identify the risk with an ID, give it a description, category, likelihood, analysis, mitigation, priority, person responsible and the progress of the risk mitigation plan.

We will typically tabulate this as follows:

The category will typically be an area of the project, such as HR, procurement or external. The probability and impact are given in terms of Low, Medium and High. We then have the severity which is computed from the probability and impact. We then assign a priority based on the severity.

Risk ID	Risk Statement	Category	Root Cause	Probability	Impact	Severity	Priority

Table 1.12: Sample Risk Register Table

1.20.20 Responding to Risks

Once risks have been prioritised, we take steps to avoid, mitigate, transfer or accept the risk.

If we avoid the risk, then we take actions to completely remove the risk. Mitigations try and reduce the probability or impact of the risk.

Transfer of the risk takes place when we move the risk to a third party. This can be done through outsourcing and insurance. If we do nothing about the risk or it is low in priority, then we may accept the consequences of the risk.

1.20.21 Procurement Planning

These are planning steps we take to buy things from outside the project. The project in these cases is the buyer and the vendor is the seller. We need to establish what to procure and what to make for a project. Once this is done, we need to establish the type of contract needed based on the type of procurement we need.

When to Make Instead of Buy

If the project has expertise or capability in making the product, has idle resources or free capacity, involves proprietary information, it is cheaper or faster to make or it is not available to buy in the market then we may make the thing.

When to Buy Instead of Make

If the project doesn't have the capability to make it, or it is cheaper or faster to buy, then we may be better off buying the product.

1.20.22 Contract Type Selection

Once we know what will be procured instead of made, then we need to decide on the type of contract to use. The contracts can be based on scope clarity and risk sharing, which are the factors.

There are three types of contract that can be used based on these factors. We can do a cost plus or cost reimbursable contract. This is where the buyer pays the seller at actual cost for the product plus a profit for their service.

We can also do a rate contract, where the buyer will pay a fixed rate for the work of the seller. Finally, the fixed price contract is where the seller agrees to do the work for a fixed sum and is expected to bear the costs if the cost goes over budget.

We will use the cost plus if the risk lies with the buyer. In a rate contract, the risk is shared between the buyer and the seller. In a fixed price contract, the risk lies mostly with the seller.

Lecture 12: Agile Project Planning

2023-10-30T12:00

Unlike traditional project plans, which are created upfront, agile only makes a high level plan at the start of the project, with planning in more detail in the next release or iteration.

Within Agile, we are expected to have to change the specification as time goes on, so replanning is the norm compared to traditional plans which shouldn't need to be changed later on.

Within a traditional software project, we also know most of the requirements upfront but Agile allows us to discover the requirements as we proceed with the project.

Planning in Agile is also done in lumps throughout the project, and adjustments are normal throughout the project.

1.20.23 Minimum Viable Product (MVP)

The MVP is the set of features that allow us to get early customer feedback. Most features of the product are probably not needed for feedback, and the subset that is is what we call the MVP.

The MVP is similar to a mockup or proof of concept.

1.20.24 Minimally Marketable Product (MMP) or Minimally Marketable Feature (MMF)

This is the list of features that form the core functionality of the product. These are the features that are needed to take the product to market.

For an ATM, for example, the ability to withdraw money from it would be an MMP. We validate and learn on the MVP until it is good enough for launch.

1.20.25 Product Backlog and the MVP

The product backlog is the set of features which we define and refine as time goes on. At the top of the backlog, we have the high priority features, which get detailed planning.

We create the minimum viable product based on these features, followed by the minimum marketable product which is a few more features that get detailed planning further along.

After the minimum marketable product, we have a list of optional features. Several of these may be dropped as we get further in the development cycle. The typical rule for this is that about 20% are dropped. We also say that 20% of the features provide 80% of the value we get in the project.

1.20.26 Agile Requirements Gathering

Unlike traditional software development approaches, we gather requirements throughout the project lifecycle. At the start of the project, we know a few requirements so start working on the project based on those. Whilst we work on the project, we establish more requirements, which then get added to the backlog.

We typically gather these requirements in the form of a user story at a high level.

User Stories

These are bite-sized chunks of business functionality. We make use of 3 pieces of information to create these: the role, functionality and business benefit.

> As a *role*, I want *functionality*, so that *business benefit*

We can also document them differently for non-functional requirements. This would be done with:

> Given the *pre-condition*, when an *action is performed* by the user,
> then what is the *action taken by the system*

This could be used to document things such as performance requirements of the system. User stories do not include technical or implementation details.

1.20.27 INVEST (in what? Not Uni it's a ripoff)

An effective user story will have 6 characteristics. These are called the INVEST characteristics.

I **Independent**: The user story should be self-contained, in a way that there is no inherent dependency on another user story.

N **Negotiable**: User stories, up until they are part of an iteration, can always be changed and rewritten.

V **Valuable**: A user story must deliver value to the end user.

E **Estimable**: You must always be able to estimate the size of a user story.

S **Small**: User stories should not be so big as to become impossible to plan/task/prioritize with a certain level of certainty.

T **Testable**: The user story or its related description must provide the necessary information to make test development possible.

1.20.28 Three C's of User Stories

We have card, where stories are written on a card to keep them concise. The details behind the story are revealed in a conversation with the customer before we start the implementation. Confirmation is where the acceptance criteria of the user story is included so the team know when the story is completed.

1.20.29 MoSCoW Prioritisation Method

This is a method to prioritise tasks from the backlog. We have must haves, which are non negotiable needs that are mandatory for the team. We then have should haves, where we have important items that add significant value.

Then we have could haves, for items with small impact if not provided followed finally by will not haves where we assign items that are not a priority.

1.20.30 Agile Work Structure

We use this structure to hide complexity and make the requirements more focusable. We have themes, which are large focus areas spanning the organisation. Then we have initiatives which are a collection of epics driving towards a common goal.

Epics then are large bodies of work that we break down into a number of smaller tasks. These smaller tasks are the user stories that we have been discussing.

As a theme we might have increase revenue. The initiative for this could be penetration of the project management software market. The epic for this would be developing a new module for workflow performance reports. The tasks from this are project management tools research, new features design and new features development.

Requirements squeeze into a hierarchy. The theme will typically take years to accomplish, with initiatives and epics being on the monthly scale. We then also have features, which take weeks, followed by stories which take days and tasks which take hours.

1.20.31 Story Points and Relative Sizing

It is quite difficult to estimate how long work will take. We use a relative estimation between different tasks to estimate tasks in relation to one another. We take a baseline story which is quite small and assign it one story point. We then base the other tasks on the number of story points they get.

This is unrelated to time and simply gives an estimation of the size of the work.

We can also use different estimation methods, by assigning stories as t-shirt sizes, by playing planning poker, using the bucket system or using affinity estimates, where we initially use relative estimates, reorganise the cards and then put items into more specific sizes.

1.20.32 Ideal Time and Real Time Estimation

When user stories are used in an iteration, they are broken down into tasks to make it easier to track. We can either estimate the time for the task by imagining that the entire 8 hour workday is available to perform productive work without any interruptions. In a real time estimation, we ask the team member to consider interruptions to their work.

1.20.33 Integration Planning

Planning for agile projects remains high-level for the long term and then we have more detailed plans for the short term. This is because we expect change in an Agile project and the requirements are subject to change.

Agile teams only write what is necessary and have discussions to make sure everyone understands what is happening. These are typically done in the daily scrum, sprint planning meetings, sprint review and sprint retrospectives.

This allows for necessary discussions and interactions.

1.20.34 Scope Planning for an Agile Project

When we have a predictive project, the scope is defined at the beginning. For an agile project, the scope is not known until the end of the project as the customer can add and remove features from the overall scope in each iteration.

During the backlog refinement, we elaborate and re prioritise work to determine what can be accomplished in that iteration. We can add new features at any time to ensure we are delivering the most value.

1.20.35 Schedule Planning

Instead of a detailed schedule, we focus on the most valuable work to complete within each iteration. This is called timeboxing. The timebox is a period of time when the team works towards the completion of a goal. A sprint is a timebox of 30 days or less.

1.20.36 Dependencies

Ideally, a team can perform all the work in their backlog. If there are dependencies on any of the items, we should ensure that we are working on them

accordingly. If we have many scrum teams, we can have a scrum of scrums, or as I like to call it a scrum^2, where we coordinate efforts and ensure that dependencies are satisfied in an inter team manner.

1.20.37 Kanban

This is a visual board, such as Trello, which is used to keep track of tasks. It means to design, manage and improve flow systems for knowledge work.

Organisations can progressively adapt when using the kanban system as we visualise the flow of work, limit the amount of work in progress tasks and stop starting tasks and start finishing them.

We have cards, which are individual tasks. We then have columns for tasks in different stages of the workflow. The cards go through the workflow until they are completed. We then have work in progress limits to prevent too many tasks in each section.

We can then also have swimlanes to separate activities, teams and classes of service but visaulise them all on one board.

1.20.38 Cost Planning

In an Agile project, there is no total project budget or detailed cost estimate. There is some estimating when using an agile approach but we do these in a relative fashion. Relative estimates are created by comparing work or grouping based on factors such as risk, complexity or required labour.

1.21 Executing Projects

Lecture 13: Executing Projects

2023-11-01T11:00

This aspect of the course discusses the management of the actual project work, management and development of the project team, differing leadership styles and management principles, auditing work, and communications, stakeholders and procurement related actions to be taken.

This all relates to the processes that have been discussed previously, with management of the project work done with the project integration management documents that we prepared. Similarly project knowledge is managed by the integration management, quality is managed by quality management, acquisition of resources is done with project resource management, team development is done through project resource management, team management is done through project resource management.

Communications are managed through the project communication management plan, risk responses through the project risk management plans, procurements

through the procurement management, and stakeholder engagements through the project stakeholder management.

Most project time and cost is spent in the execution phase of the project. We produce deliverables with resources, data on the deliverable is collected including the start and end dates and the cost.

This is then used to measure project variances for tracking the project against its baseline.

Projects may require changes to the original plan as the project goes on. These changes are raised as changed requests and must be approved. The process for handling these changes is defined during the planning phase.

1.21.1 Problems During Execution

Problems may arise during execution. This is more likely to happen if the objectives or scope are unclear, estimates for time and cost goals are unrealistic, technology changes, personnel are incompetent, there are poor conflict management procedures, communications are poor or suppliers are not delivering as promised.

These problems may be avoided by doing a good job with the initiation, planning or monitoring and controlling of the project.

1.21.2 Project Integration Management

The main outputs during the execution phase are deliverables, work performance data, issue logs, change requests, project management plan updates, project documents updates, and organisational process assets updates.

We can then issue a milestone report for the work performance data. This would include the milestone reachec, the date, status, who's responsible and any issues or comments on the milestone.

This might look like:

1.21.3 Issue Logs

These are documents to document, monitor and track issues that need resolving for the project. An issue is a matter under question or a dispute that could impede project success.

Issues can hurt team performance, so we need to take action to resolve them.

Critical issues are anything that prevents progress on scheduled activities. We further call them urgent if they are on the critical path. There is an issue escalation process that ensures issues are resolved.

Milestone	Date	Status	Responsible	Issues/Comments
Researched existing training	8/13	Complete	Jamie (replaced by Abner)	Many basic courses available, but not much advanced/tailored training. (Note: Replaced Jamie with better candidate for project after Jamie completed this task)
Presented supplier management training survey results to steering committee	8/24	Complete	Kristin	Great feedback. Many people stressed the need to have instructor-led training and mentors for soft skills development
Meetings with potential partners	9/21	In progress	Kristin/ Contracting	May need more time for meetings

Table 1.13: Project Milestones

Issues are logged in a tabular format with the number, issue description, impact on the project, date reported, who reported to, who assigned to, their priority, the due date, the status and any comments or follow-ups that need to happen.

1.21.4 Resource Mobilisation and Management

Projects require resources, including human resources and machines. The project plan documents the amount and type of resources required at any given time, and these may be split over multiple locations.

These factors need to be considered whilst mobilising the right number of resources with the right number of skills at the right location.

Holding resources put a cost on the project, so we don't put a hold on them until we need them and will release them when we no longer require them.

1.21.5 Resource Loading

This is in reference to the number of resources and existing schedule requires during specific time periods. It helps project managers to develop an understanding of the demands a project will make on an organisation's resources and individual schedules.

Overallocation is a major issue when it comes to resource loading. This is where we have more resources than are available assigned to tasks.

Project managers use resource histograms to denote the resource loading on

individuals.

1.21.6 Resource Levelling

This is where we accommodate resource constraints by adjusting the dates for a project. This allows us to make realistic deadlines without overworking the team and increasing the total cost of the project. The main purpose of levelling is to create a smoother distribution of resource usage and overallocation.

Example. If we have a project activity on arrow diagram, looking like so:

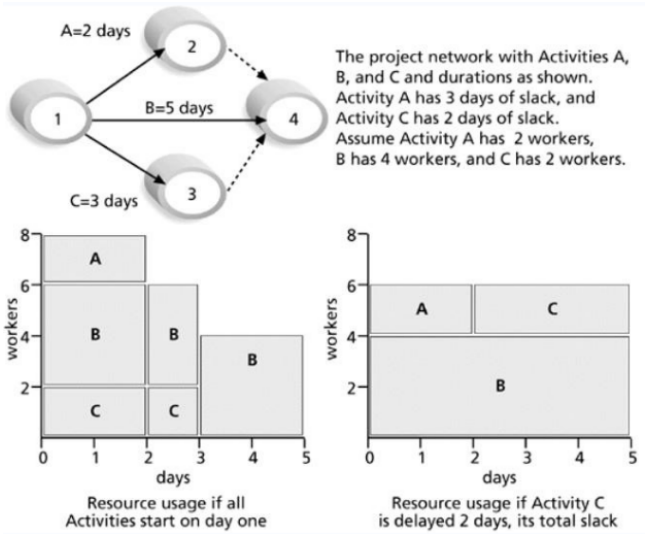

Figure 1.10: Resource Levelling with an AOA diagram, followed by graphs for usage if all activities start on day one versus delaying activity C by two days

1.21.7 Team Management

During the execution phase, the project manager develops a team and manages the members in the team. The manager is responsible for taking actions that improve the effectiveness of the entire team as a unit.

They set the team culture, train the team members, organise team building activities and distribute rewards and recognition to the team members.

1.21.8 Tuckman Model of Team Development

This is the forming, storming, norming, performing, and (because this is a master's module, we get the privilege of) adjourning, which is when we break

up the team when they reach their goals and complete the work. They may also adjourn due to poor performance or cancellation of the project.

1.21.9 Motivation

Project managers must understand motivation theories to effectively execute projects. There are two types of motivation, these are intrinsic where people participate for their own enjoyment, and extrinsic where people do something for a reward or to avoid a penalty.

Maslow the top g he is suggested that people are guided or motivated by a sequence of needs.

1. Physiological Needs
    - Breathing
    - Food
    - Water
    - Shelter
    - Sleep
    - Clothing
    - Reproduction
2. Safety Needs
    - Personal security
    - Employment
    - Resources
    - Health
    - Property
3. Love and Belonging
    - Friendship
    - Intimacy
    - Family
    - Sense of connection
4. Esteem
    - Respect
    - Self-esteem
    - Status

- Recognition
- Strength
- Freedom

5. Self-Actualization
    - Desire to become the most that one can be

This is typically represented as a pyramid the other way round.

1.21.10 Conflict Handling

Conflicts can arise in the workplace and these need to be handled appropriately. We have different conflict handling modes, which we can represent on a graph of cooperativeness versus assertiveness.

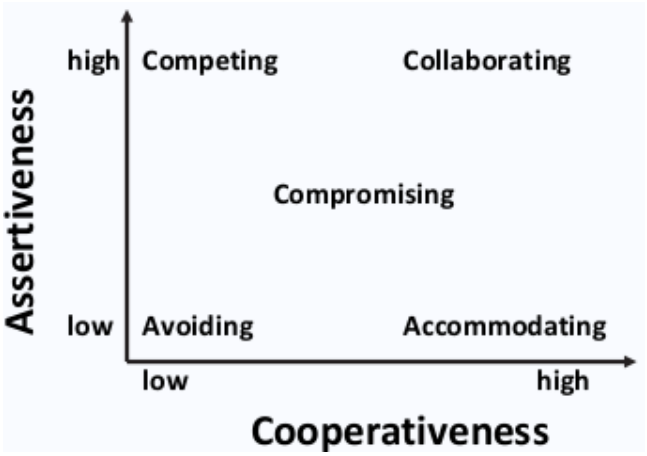

Figure 1.11: Conflict Handling Modes

- **Competing:** This mode is assertive and uncooperative. Individuals pursue their own concerns at the expense of others.
- **Accommodating:** This mode is unassertive and cooperative. Individuals neglect their own concerns to satisfy the concerns of others.
- **Avoiding:** This mode is both unassertive and uncooperative. Individuals do not address the conflict, ignoring both their own concerns and those of others.
- **Collaborating:** This mode is both assertive and cooperative. Individuals work with others to find a solution that fully satisfies everyone's concerns.
- **Compromising:** This mode is moderate in both assertiveness and cooperativeness. The goal is to find a mutually acceptable solution that partially satisfies both parties.

Conflict can be good, as it can produce important results such as new ideas, better alternatives and motivation to work harder.

Project team members may become stagnant or develop groupthink if there are no conflicting ideas in the group.

Task-related conflict can improve team performance and this is where we have differences over team objectives.

We can also have emotional conflict, which stems from personality clashes and misunderstandings. This often depresses team performance.

1.21.11 Leadership Styles

- **Laissez-faire:** A hands-off approach that allows teams to set their own goals and methods.
- **Transactional:** Management by exception, using rewards and punishments to achieve goals or compliance.
- **Servant leader:** Prioritizes relationships and community, with leadership taking a secondary role.
- **Transformational:** Collaborates with others to identify and implement needed changes through inspiration.
- **Charismatic:** Inspires others through enthusiasm and confidence.
- **Interactional:** A blend of transactional, transformational, and charismatic leadership styles.

Daniel Goleman on Situational Leadership

- **Visionary:** Guides people towards a new set of shared dreams when an organization needs a new direction.
- **Coaching:** A one-on-one approach that helps individuals improve their performance.
- **Affiliative:** Focuses on teamwork and creating harmony by connecting people to each other.
- **Democratic:** Leverages people's knowledge and skills to build commitment towards shared goals.
- **Pacesetting:** Sets high performance standards.
- **Commanding:** Also known as autocratic or military style, this is often used and directs people in a clear and firm manner.

1.21.12 Managing Quality

Quality assurance, as already discussed is all of the activities related to satisfying the quality standards for a project. ANother goal of QA is a continual improvement on the quality.

Key outputs include a quality report, test and evaluation documents, change requests, project management plan updates and project documents updates.

1.21.13 Quality Improvement Tools and Techniques

We can make use of a practice called benchmarking by comparing specific practices or characteristics to those of other projects or products within or external to the organisation. We can then see where we are relative to company or industry norms.

We can also do a quality audit, where we review the quality management activities that identify lessons learned, which could improve performance on current or future projects.

We can also make use of process analysis, where we look at how a process operates and determine improvements. We can use a task or kanban board to display work in columns labelled todo, in progress and done.

We can also make cause and effect diagrams which can assist in ensuring and improving quality by finding the root cause of the quality problems.

1.21.14 Project Risk Management

The main executing task as part of project risk management is the implementation of risk responses as defined in the risk management process to plan the responses.

Key outputs include the change requests and project documents updates. This includes the issue log, lessons learned register, project team assignments, risk register and the risk report.

1.21.15 Project Communications Management

This is another key part of the project. We gather information to reate, distribute, store, retrieve and dispose of communications in accordance with the communications management plan.

We then put the documents in a defined place when we dispose of them and this is determined by the document retention regulations that the organisation has.

Important Project Communications Concepts

Project managers should be aware of important aspects related to improving communications. They should know the difference between formal and informal communications, nonverbal communications such as tone of voice and body language, ensuring the correct communications medium, an understanding of individual and group communication needs and the impact of the team size on the project communications.

1.21.16 Project Stakeholder Management

The management of engagement from stakeholders involves working with them to meet their needs and expectations and fostering engagement in decisions and activities.

If this is done well, then the project manager can increase support and minimise resistance from stakeholders, improving the chances of project success.

1.21.17 Managing Stakeholder Engagement

Good teachers use several techniques to engage students, and project managers should do the same. They should ensure that they set the stage early so that stakeholder engagement is expected and welcomed. Many outputs are similar to other knowledge areas, and things usch as change requests and updates to the project management plan should be communicated with stakeholders.

1.21.18 Project Procurement Management

The main executing process is conducting the procurements, by obtaining responses to requests for proposals or bids, selecting sellers and making agreements by rewarding contracts.

Prospective sellers do most of the work by preparing their proposals and bids, usually at no cost to the buyer. The buying organisation is responsible for deciding how to approach sellers and provide the required procurement documents.

Important documents creates as a result of conducting procurements include contracts.

1.22 Monitoring and Controlling Projects

Lecture 14: Monitoring and Controlling Projects

2023-11-02T10:00

During execution of a project, we need to check that the project is going to plan. The project manager spends most of their time in the execution, monitoring and controlling of the project. To do this well, they need to have a good workable plan that can be executed. If the project is not executed as per the plan, then this can lead to issues later on.

Failures in the execution are known as monitoring and controlling failures.

1.22.1 Project Management Process Groups

As discussed earlier, this is a series of actions directed towards a particular result. We have the initiating processes, planning processes, executing processes, monitoring and controlling processes and closing processes. The monitoring and controlling processes form an overarching theme which continues through the whole lifecycle of the project, whereas the rest of these groups are typically somewhat serial.

Monitoring and controlling processes are responsible for the tracking, review and regulation of the progress and performance of the project. Should the project not be performing as expected, this is where we identify and provide corrective actions to the process.

1.22.2 Monitoring

This is the process of regularly measuring progress to ensure that the project is meeting its objectives. As the project is executed, data is collected on the actual scope, actual start and end dates, actual incurred costs, and actual quality measurements. These are then compared with the ones mentioned in the plan. Should there be a variance, then we can do some further analysis to see if the variance is favourable or unfavourable.

1.22.3 Control

If a variance on the project is unfavourable, then we might have to take corrective action to bring the project back on track. This is a part of the control function of the project.

Some unfavourable variance may not warrant a response at all. Sometimes the variance is within limits and does not warrant a response at all, although we may still take a preventative action to prevent issues cropping up in the future.

These corrective and preventative actions may be referred to as CAPA.

1.22.4 Schedule and Cost Tracking

These are the two most important areas to track on a project. During the planning phase, the schedule and cost are baselined, to give an idea of where they should be at different phases in the project. During execution, we then compare the values we collect to the baseline and see where we are at.

We typically track the following three items:

- **Schedule Variance (SV)** – this is the difference between the baseline finish date and the actual finish date of an activity or the project.

- **Cost Variance (CV)** – this is the difference between the baseline and actual cost of an activity or the entire project.
- **Effort Variance (EV) / Work Variance** – this is the difference between the baseline effort and the actual effort spent on an activity or the project.

Parameter	Activity 1	Activity 2	Activity 3
Baseline			
Finish Date	1st August 2012	15th August 2012	5th Sept 2012
Cost	£20,500	£10,250	£60,025
Effort	10 Man Days	3 Man Days	25 Man Days
Actual			
Finish Date	1st August 2012	17th August 2012	4th Sept 2012
Cost	£20,500	£9,000	£62,025
Effort	10 Man Days	3.25 Man Days	23 Man Days
Variances			
SV	-	2 days	-1 day
CV	-	-£1,250	£2,000
EV	-	0.25 Man Days	-2 Man Days

Table 1.14: Variance - An Example

1.22.5 Earned Value Technique

This is a technique to track progress based on cost parameters. We give everything in terms of costs to the project, despite it also calculating schedule and cost variances. We take a baseline and then use the EV technique to determine how well the project meets scope, time and cost goals.

We take baseline information from the scope data, time data and cost data, which are given by the WBS tasks, start and finish estimates for each task, and the cost estimates for each task.

Earned value management is the process of making decisions based on what we know has been achieved, what it cost to achieve the work, if it has cost more or less than planned and if the project is ahead of or behind the planned schedule.

Earned value management can be intimidating to project managers because there are lots of different terminologies associated with it. A list of these is given below:

- **Budget at Completion (BAC)** – this represents the original project budget
- **Planned Value (PV)** – this is the budgeted cost for work scheduled

(BCWS) and varies based on the scope of the work and the point where we're at in the overall schedule.

$$PV = \text{Total Project Cost} \times \text{Percentage of Planned Work}$$

e.g., PV for a 5 month project is £25,000, then PV at 2 months is:

$$\begin{aligned} \text{PV (Complete)} &= £25,000 = \text{BAC} \\ \text{PV (2mo)} &= £25,000 \times \frac{2}{5} \\ &= £10,000 \end{aligned}$$

We can also calculate the PV for a time period, e.g., from month 2 to month 4:

$$\begin{aligned} \text{PV (mos. 2-4)} &= £25,000 \times \frac{3}{5} \\ &= £15,000 \end{aligned}$$

This is inclusive of months 2, 3 and 4, hence 3/5 months in the equation.

- **Actual Costs** – this is the actual cost of the current work performed. If using a project cost management software, then this is easy. We need to remember to include several hidden costs, such as material, resources, hardware, software licences and other overheads. AC can be calculated accumulatively, from the start; or over a specific time period.

With all this data, we can see what we'd planned to do and the budget to execute this plan. There is always some discrepancy from the plan. For example, if we wanted to do 40% of the work in 2 months, but we'd only managed to do 30%, then we need to see what the budgeted cost was for this work.

We have the budgeted cost for work performed (BCWP), which gives the answer. The earned value as such is:

$$\begin{aligned} \text{EV} &= \text{Total Cost} \times \text{Percentage of Actual Work} \\ &= £25,000 \times 30\% \\ &= £7,500 \end{aligned}$$

1.22.6 Earned Value Management System (EVMS)

The earned value management system allows us to plot a graph of the current earned value, planned value and actual cost, giving us the schedule variances and cost variances.

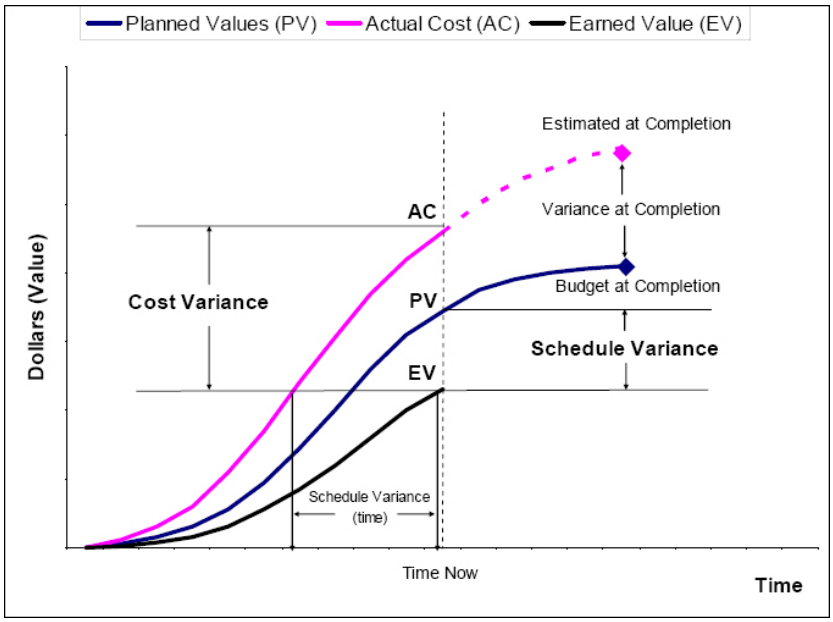

Figure 1.12: Earned Value Management System

This provides a nice graphical representation of the data at differing points in the project, and allows us to see whether we are ahead or behind schedule and above or below the estimated cost.

We can further analyse the data with variance analysis to get the schedule status and cost status, both of which are made up of a variance and a performance index.

1.22.7 Schedule Variance

This is a quantitative indicator of the divergence from the initial planned schedule. If this is negative, then we are behind schedule. If this is positive, then we are ahead of schedule. The variance can be calculated as:

$$\text{SV} = \text{EV} - \text{PV}$$

Example. If we have £7,500 earned value and £10,000 planned value at 2 months into the project, then our schedule variance would be:

$$\begin{aligned}\text{SV} &= \text{EV} - \text{PV} \\ &= 7,500 - 10,000 \\ &= -2,500\end{aligned}$$

The percentage of schedule variance can also be calculated as follows:

$$\text{SV\%} = (\text{SV}/\text{PV}) \times 100$$

In the above example, this would be $(-2,500/10,000) \times 100 = 25\%$ behind schedule.

Schedule variance is understood as a perspective of costs. We get to these costs by looking at the scope of the work planned and completed. Scope, time and cost all come together in earned value management.

1.22.8 Cost Variance

This is a quantitative indicator of the divergence from the initial budget. A negative CV indicates being over budget, with a positive CV indicating under budget. The formulae for CV and CV% are given below:

$$\text{CV} = \text{EV} - \text{AC}$$
$$\text{CV\%} = (\text{CV}/\text{EV}) \times 100$$

Lecture 15: Monitoring and Controlling Projects Continued

2023-11-06

1.22.9 Schedule Performance Index (SPI)

This is another way to measure the project performance. If the index is above one then the project is ahead of schedule, and if less than one it is behind schedule.

We can do this for both the schedule and cost performance indexes, where they are given with the following formulae:

$$\text{SPI} = \text{EV}/\text{PVCPI} \qquad = \text{EV}/\text{AC}$$

For example, the schedule performance index of a project with an earqned value of £7,500 / £10,000 = 0.75, which indicates that we are only at 75% of the original schedule, or 25% behind schedule.

The cost performance index gives us a figure as to how well the project is conforming to the budget. If the CPU is 0.5, then we are at twice the amount of spending that we should be at for this point in the project.

1.22.10 Interpreting EV Numbers

A negative number for cost and schedule variance typically indicate problems in those areas. A negative number means that the project is costing more than planned or taking longer than planned. For indexes less than 100% or 1, then we have a problem.

1.22.11 EV Forecasting

Earned value management gives 4 calculations to give the project manager a forecast into the future of the project. These are:

- Estimate to Complete (ETC)
- Estimate at Completion (EAC)
- Variance at Completion (VAC)
- To Complete Performance Index (TCPI)

These are all extrapolations from previous calculations, which were used to determine the current status of the project.

As we always say, there are two types of people: those that can extrapolate from complete data and

ETC

This represents the expected cost to complete the project. This is only the future budget of the project, not the total that will be required. This can be calculated in two ways: based on project performance:

$$\text{ETC} = (\text{BAC} - \text{EV})/\text{CPI};$$

or based on a new estimate, called management ETC. This is where a new estimate of the remaining tasks in the project are performed.

EAC

This is the full cost that is expected at completion. This can be done on a task-by-task basis or once for the entire project. There are lots of ways to calculate based on how we expect the future performance of the project to be.

If the existing variance was a unique event and the rest should go to plan, then add the remaining budget to the actual cost incurred to date. This is shown by the following formula:

$$\text{EAC} = \text{AC} + (\text{BAC} - \text{EV})$$

We can also base the future cost performance on the past cost performance. If the past is not unusual and is a good indicator of the future, then we use the following formula:

$$\text{EAC} = \text{AC} + ((\text{BAC} - \text{EV}/\text{CPI})$$

This gives the worst-case scenario for the project, given its current trajectory.

The third way we can do this is to think that the future cost performance will be influenced by the past schedule performance. As these are usually related,

there might be a reason to adjust the cost performance based on the schedule performance. If the CPI is low but the SPI is high, then we might think that the final cost performance won't be as bad as the CPI might suggest.

We can use the future cost performance as being influenced by the past schedule performance by calculating the EAC based on an average of the CPI and the SPI:

$$\text{EAC} = \text{AC} + ((\text{BAC} - \text{EV})/(\text{CPI} \times \text{SPI}))$$

We can then vary that formula to use a combination of the SPI and CPI in a weighting, by adjusting variables x and y, such that they sum to 1:

$$\text{EAC} = \text{AC} + ((\text{BAC} - \text{EV})/(x\text{CPI} \times y\text{SPI}))$$

We can also produce a new estimate with the management ETC, which can then be added to the to-date cost (AC) of the project, to determine the final EAC.

$$\text{EAC} = \text{AC} + \text{ETC}$$

The first two methods represent the low and high extremes of costs. The third method estimates somewhere between those extremes. The fourth method represents a new estimate, if none of the other methods can produce the desired result (somewhat suggesting that the fourth is like a magic bullet and should be used immediately if you encounter any issues).

1.22.12 Variance at Completion

This is a forecast of what the variance will be upon completion of the project. This is a good indicator of the expected cost of the overrun or the underrun of the project.

The project manager is usually expected to request additional funding as early as possible or report the potential for an overrun. The VAC represents the size of the request. This is given with the formula:

$$\text{VAC} = \text{BAC} - \text{EAC}$$

1.22.13 To Complete Performance Index

This represents the efficienty level that will make the project finish on time. This is powerful as an indicator as it allows us to ascertain if people will be as productive as the indicator tells you. It is a bigger red flag than other indicators. If it says people need to be twice as efficient as the schedule, then we know that action needs to be taken.

We can calculate the TCPI in two ways, to achieve the original budget, or to achieve the EAC. To achieve the original budget, and the overrun or underrun

of the project hasn't resulted in a change to the schedule or the budget, then we can use the following formula:

$$\text{TCPI} = (\text{BAC} - \text{EV})/(\text{BAC} - \text{AC})$$

To achieve the EAC, where we have made a change to the original budget and the EAC is the new budget, then we use the following formula instead:

$$\text{TCPI} = (\text{BAC} - \text{EV})/(\text{EAC} - \text{AC})$$

We assume that the project budget has not been revised and that the goal is still the original project budget. A high TCPI shows that the team must be much more efficient than they have been to finish on budget.

1.22.14 Performance Reports

Status reports describe the project at a specific point in time. This is a report of all activities from the beginning until today. This is not sent frequently.

Progress reports describe what has been accomplished during a certain period. This is the activities since the last report was sent. The reports are sent more often and are shorter than the previous report.

Forecasts give the predictions of future project status and progress based on past information and trends of the project.

1.22.15 Controlling Scope

The scope cannot be controlled unless first clearly defined and a scope validation process is in place. We need to have a process for changes to the scope and how to request and monitor those changes.

Stakeholders are encouraged to suggest beneficial changes and discouraged from making unnecessary ones.

1.22.16 Integrated Change Control

This is the process of identifying, evaluating and managing changes throughout the project lifecycle. The objectives of this are to:

- Influence the factors that cause changes to ensure those changes are beneficial
- Determine that a change has occured
- Manage changes as they occur

The project team must focus on delivering the work as planned, but can make necessary changes to the initial plan.

1.22.17 Handling Changes

As much as we'd like them not to be, they are a reality in projects. These changes are asked by stakeholders as they add value to them. Too many changes can lead to rework, frustration and increased costs. Changes are difficult to implement towards the later stages of the project. It is best to anticipate changes and then have a change control process in place to handle changes as and when they come.

1.22.18 Change Control Process

A good process should have the following elements:

- Change request template, a standard form that needs to be filled in by the requester
- Change request SPoC, the single point of contact for the person who manages the changes to the project
- Impact analysis team, this is the list of people who analyse change requests and forward them to the approving authority
- Approving authority is the names of the stakeholders who would take the decision on whether the change should be included or not

As part of the process, the change should go through the following processes as part of the request:

- A feasibility study–is it actually possible to implement the change
- Impact analysis–does the change impact the schedule, cost, resources, risk or quality?
- Determining options–how can the change be implemented?
- Getting Approval–presentation of the analysis and recommendations to the approving authority

1.22.19 Project Quality Management

As part of quality control, we get lots of outputs: quality control measurements, verified deliverables, work performance information, change requests, project management plan updates, the project documents updates.

As part of these documents, we have outcomes, such as acceptance decisions, rework and process adjustments. Quality is defined as a product meeting its specification.

Software has differing quality requirements to a traditional project as there is a tension between the customer requirements and developer quality requirements. The customer wants efficiency and reliability, but the developer wants maintainability and reusability.

Some quality requirements can be hard to specify unambiguously. Software specifications are usually incomplete and inconsistent. Therefore, we have a dilemma in that we can't define quality.

In physical products, quality control is able to control the quality of the generated product by controlling the actual manufacturing process. Such control can compensate for weakness in design and materials. Monitoring items produced and adjusting processes to achieve acceptable rates of failure can be done with physical products.

In software development there is no equivalent. We can't resolve a car component that fails too often in service through the use of a higher grade of steel, as there is no higher grade of steel for software...

1.22.20 Quality Compromise

We can't wait for specifications to improve before paying attention to quality management. Quality management procesures must be put into place to improve the quality in spite of an imperfect specification.

1.22.21 Quality Control Tools

Data gathering tools, such as check sheets, statistical sampling, questionnaires and surveys can all be used as QC tools. We can then perform data analyses such as performance reviews and root cause analysis.

We can do inspections, checklists, testing, and product evaluations, and represent these with cause-and-effect diagrams, control charts, histograms, and scatter diagrams.

1.22.22 Seven Basic Tools of Quality (ASQ)

1. Cause-and-effect diagrams–to help find the root cause of quality problems
2. Check sheets–a tool to collect qualitative and quantitative data, giving facts about quality problems
3. Control charts–to illustrate the results of a process over time and show if a process is in control.
4. Histograms–show a bar graph of a distribution of vairables
5. Pareto charts–help to identify and prioritize problem areas
6. Scatter diagrams–show if there is a relationship between two variables ~~and show causation, as correlation is causation~~
7. Stratification–this is a technique used to separate data to see patterns. This can be done as a run chart, giving the history and pattern of variation of a process over time, or as a flow chart, which is a graphical display of

the logic and flow of processes to help analyze how problems occur and processes can be improved.

1.22.23 Check Sheets

These have the defect in column one, then each day in a column, then the total at the end.

1.22.24 Control Charts

Graphical display of data that illustrates the results of a process over time. This has the specification, upper specification limit, control limits and violations of the seven-run rule.

The severn-run rule should have variation about the specification and cross the centreline at most once every seven runs.

Control charts allow us to see whether a process is in control or out of control. If in control, then any variations are part of a random event. Processes in control do not need adjustments. Processes out of control need to have their causes investigated and fixed, as these variations are caused by non-random events.

If seven runs are all below the mean, above the mean, increasing or decreasing, then the process needs examining.

1.22.25 Histograms

Histograms give each bar as an attribute or characteristic of a problem or measurement point, with the height representing its frequency.

1.22.26 Pareto Charts

This is a histogram that can help identify and prioritize problem areas. The variables in the histogram are ordered by frequency of occurence in a chart, with a line chart to show the cumulative percentage on the right.

These help to identify the vital contributors that account for most quality problems in a system. Pareto analysis is referred to as the 80/20 rule, with 80% of the problems often due to 20% of the causes.

Sometimes, different problems don't have the same importance. For example, a life threatening problem should always be considered before the less important problems.

1.22.27 Scatter Diagrams

These show if there is a correlation, or connection between two sets of data. Data points closer to a diagonal line show that the data is related. Graphs may have a positive correlation, negative correlation or no correlation at all.

Correlation does not imply causation.

1.22.28 Stratification

This is a technique to separate data to see patterns. We may use run charts or flow charts in place of this.

A run chart gives the history and a pattern of variation of a process over time. It shows data points plotted in the order in which they occur.

We can use run charts to perform trend analysis to forecast future outcomes based on historical results.

1.22.29 Monitoring and Control in Agile Projects

Within agile, work progress and results are monitored in daily scrums and sprint reviews. If we deviate from the plan, then we can initiate actions, such as rescheduling, and adapt the procedure to learn from it.

We can make use of burn charts to show the current velocity. This measures the productivity rate at which we produce deliverables, validate and accept them.

1.22.30 Burn Charts

Burndown charts show the amount of work remaining compared to the plan. Burnup charts show amount of tasks completed compared to the plan. These can be combined into combined burn charts to show how much work has been completed and how much remains.

1.22.31 Velocity Charts

These are used to measure how much work can be completed in each iteration. This allows creation of accurate and efficient timelines. The velocity is not constant and varies. We use these charts for release planning but not to monitor the team.

A velocity chart is created after the first sprint and then updated after each completed sprint.

1.23 Closing Projects

Lecture 16: Closing Projects

2023

Closing projects is the process of finalizing all activities and bringing the project or phase to an end. We archive all project information, ensuring that the planned work is complete, and free organizational resources.

The end of a project typically includes a final presentation and report. We also reflect on the project to see what can be done to improve future projects.

We want to plan and execute a smooth transition of the project to the normal operations of the company.

Deliverables are finalised and transferred, all documentation is signed off, approved and archived. All work is checked to ensure completion according to the plan and scope. All project management processes should have also been executed. We need to ensure final sign off and approval from all stakeholders. The process also allows us to review and evaluate the performance to ensure future success.

There are 4 ways to close a project. The first is integration, where a project is completed and the products and services are integrated into operations. This is the most common approach.

There is then addition, where the project creates a new product or service resulting in a new organisational unit, such as department, division or company.

We also have extinction, where the project ends because it was successful, or unsuccessful and superseded. This is where the project might end suddenly and would be called termination by murder.

Termination by starvation is where we decrease the budget or cut funding, and this can also be called withdrawal of life support.

1.23.1 Types of Project Closure

Project can go through normal closure, where completed as planned, all deliverables produced and all objectives and goals achieved.

We can have premature closure, where they are closed before they can get off the ground, just before they are about to start. We also have suspension closure, where we put the project on hold. This may be due to external factors or operational constraints.

Finally, we have a failed closure, where the project is unable to achieve its objectives or deliver the desired outcomes. Then it is closed without accomplishing the intended goals.

1.23.2 Importance of Closing a Project

Without a formal process, we risk letting details fall through the cracks, resulting in confusion, a never-ending project, dissatisfied clients and possible liability issues.

Closure helps avoid repetition of mistakes on future projects and objectives, final products or deliverables without support and resources, failure to identify the team or individuals who will own and maintain the solution or creation of liability issues from incomplete payments, contracts or deliverables.

It can also tell us if the goals have been met, clients are happy with the documentation and deliverables, show how on time the project was completed, give any takeaway lessons from bottlenecks or mid-process problems that could be helpful for future teams, and check that senior management are happy with the ourcomes.

We also check to see the expected return on investment, future reference in terms of worth of time, effort, and money, the stakeholders and staff using the software or product are aware of it and trained on its use.

Each of these depend on whether the project is being delivered internally or for a client.

1.23.3 Closing Process Activities

We obtain acceptance of the project deliverables, hand off the operations and support responsibilities, document lessons learned, formalise the closure and obtain a sign-off from the project sponsor and manager.

We can use a checklist to ensure that the project is successfully completed:

1. Formally transfer all the deliverables
2. Confirm the project completion through agreement with stakeholders
3. Document this agreement
4. Review all contracts and documentation to ensure that all parties have been paid for their work and there are no outstanding invoices
5. Release resources from the project, so that they are free to work on other projects.
6. Document the lessons learned over the course of the project, did we stay on budget, was time managed wisely, were there issues or compromises along the way, did the project meet the customers needs?
7. Archive documentation and index in company archives for later reference. It provides a clear paper trail for the processes, decisions and actions taken during the project.
8. Project closure meeting

1.23.4 Project Closure Meeting

We have a meeting with the project sponsor, owner and the team. We answer different questions in the meeting, dependent on if the project was done to completion or if it was cancelled/suspended.

If the work was completed, we check if we accomplished all the agreed upon objectives, follow up work in future projects, how to operationalise the project,

who will take over the support or admin from the project, the lessons learned and that the team knows that their hard work was appreciated.

If the project was suspended, we want to know the reasons for that and if there are any plans to restart the project in the future.

1.23.5 Project Closure Report

Following this meeting, the project manager produces a closure report using the closure report template.

This report measures how closely the project met customer needs, identified what worked well and what needs improvements, any deviation from the original plan and the causes for that. Any methods for improvement, and lessons learnt are also all documented.

When doing the lessons learnt, we could include an action item to prevent recurrence of the issue.

1.23.6 Approvals

Projects are officially completed when the closure report is accepted and approved by project sponsors and other designated stakeholders. The sign off of this document acknowledges that all of the deliverables are complete, reviewed and accepted. This shows agreement amongst stakeholders that the project is complete. This can be viewed as the final management milestone for the project.

1.23.7 Closing Agile Projects

Similar to the predictive approach, Agile and hybrid projects should also have a closure process. Thanks to the scrum events, we constantly have an intentional moment of closure. The daily meeting provides closure for the day before, the sprint reviews provide closure for the sprints, and the sprint retrospectives give us insight on the lessons learnt.

The teams should still hold a close-out meeting regardless and celebrate the end of the project.

1.23.8 Advice on Closure

The projects should be planned for the closing phase. There should be tasks in the WBS and resources allocated to closing the project. An Agile project should include a user story for closing the project.

It is much easity to close the project if the team captures the lessons learnt and other information required for closing as soon as possible.

The project manager should take time to thank the team and other stakeholders on the project.

1.24 PRINCE2

Lecture 17: PRINCE2 2023

PRINCE2 is a project management methodology, with seven principles, themes, processes and roles. PRINCE is an acronym for PRojects IN Controlled Environments (who is in charge of acronyms these days that's just a silly one).

It is a structured PM method that divides the project into manageable phases. It is a planned-ahead methodology and follows a stage-by-stage process to project management.

Unless all seven principles are applied, then the project is not a PRINCE2 project. The seven principles are as follows:

1. Continued business justification
2. Learn from experience
3. Defined roles and responsibilities
4. Manage by stages
5. Manage by exception
6. Focus on products
7. Tailor to the environment

Each principle is given in more detail below.

Continued Business Justification

Essentially, make sure that every project makes business sense, and give a document with the business case, which is reviewed and revised throughout the lifecycle of the project. This is done to ensure the project remains viable.

Learn from Experience

This allows the business to improve their project management abilities. If we avoid repetition of bad things and repeat the good things in projects, then the project is bound to do better.

We attach a lesson log to every project and this acts as a repository for lessons learned in the previous projects.

Roles and Responsibilities are Defined

All decision makers must understand their responsibilities. If this is done, then we can't hand off the responsibility to another person. People can have multiple roles or share a role with others. We have 4 levels that create the structure of a project:

- Corporate
- Project board
- Project manager
- Project team

Manage by Stages

We break down the project into management stages. At each stage, we have a go or no-go decision point. If the project is still worthwhile, then we can proceed. If the project is no longer viable, then we close it.

When transitioning between stages, we check and update the business case, risk, and project plan.

Manage by Exception

This is a management strategy to allow delegation of the day-to-day responsibility of the project. We set a tolerance for each project objective, such as 10 ± 2 days to give a tolerance for the length of the task. If we exceed the tolerances in the plan, then we escalate this 'exception' for a decision.

Focus on Products

The quality requirements of deliverables are of paramount concern.

Tailor to Suit Project Environment

All projects are different. We adapt this methodology to suit the project's needs. The amount of oversight and planning should fit the size of the project, number of people involved, complexity, importance, time, risk, etc. of the project.

1.24.1 PRINCE2 Themes

This method has seven themes which are the project management areas which must be managed throughout the project. These themes are equal to the knowledge areas in the PMBOK Guide.

The seven themes are:

1. Business Case
2. Organisation
3. Quality
4. Plans
5. Risk

6. Change
7. Progress

Once again, these will be listed in more detail below.

Business Case

Document the project's justification. Maintain the justification throughout the project. It should contain a cost-benefit analysis to weigh up the benefit vs. costs, times and risks of the project.

Organisation

The highest level of decision making is the project board. They are not regularly needed to run the project based on the idea of management by exception. Project assurance monitors the performance and products of the project. Change authorities decide on changes. Project support assists the project and team managers.

Quality

The users must get the right products. These products must be delivered to the user specifications.

Plans

Different plans (project, stage, team) are needed by different groups of people. These plans define what, when, who and how much for the project. After these plans are approved, they are managed under change control.

Risk

These are uncertain events which could have negative or positive impacts on the project. These are managed regularly and escalated to senior management as and when necessary.

Change

Change will always happen, either from the outside (e.g., new laws), or inside (e.g., user requirements change). Within PRINCE2, a change authority uses a change budget to pay for changes.

Progress

Progress is tracked with time-driven reports. We take decisions using ad-hoc reports. These reports are sent to the next highest management level. These are compared with what should have happened, and the plan is adjusted to get the project back on track.

1.24.2 Processes

Processes describe what decisions are required, who takes these decisions, what management products support the decision making, and when decisions are taken.

Processes are where principles and themes within the methodology are applied. The processes in this methodology equate to process groups in the PMBOK guide. The PRINCE2 method has 7 processes. These processes give a stepwise progression through the project lifecycle from the start to the closure.

It includes a checklist of activities, products and related responsibilities.

The seven processes are:

1. Starting up a Project
2. Directing a project
3. Initiating a Project
4. Controlling a Stage
5. Managing Product Delivery
6. Managing a Stage Boundary
7. Closing a Project

Starting a Project

The customer submits a request for the new project called the project mandate.

The mandate is evaluated and the company checks whether it can take up the project. Once approved, then a detailed proposal is submitted including the resources actions and other project information.

Directing the Project

The viability of the project is analysed, then it is assigned to the project manager.

Project Initiation

The manager then makes a plan with the cost, time, quality, advantages, risk and scope of the project.

Stage Control

The project is divided into stages and teams are assigned to each stage. The PM then tracks the performance and issues of each stage.

Product Delivery Management

The PM ensures there is no gap between the deliverables and the quality expectations. The board makes a final evaluation and either approves the product or asks for further revisions.

Managing Stage Boundaries

At the closing of each stage, the manager and the board review the outcomes of the stage to ensure the project is on track.

Project Closing

This marks the closing of the outcomes, documentation and reporting for the project.

1.24.3 Roles in PRINCE2

There are seven roles, but three principal roles. The principal roles are the project board, the project manager and the project team. In addition to these roles, there are supplemental roles that ensure requirements and standards are met and that the work runs smoothly. These roles are documented below.

1. Customer – the person or organisation paying for the project to be completed
2. User – someone who will use the deliverables, or will be impacted by the outcome of the project
3. Supplier – the subject matter expert, providing the knowledge to complete the project by designing or building the end result
4. PM – responsible for organising, planning and overseeing work on the project.
5. Project Team – employees who are doing the work and bringing the project vision to life
6. Team Manager – oversee daily work and report to the project manager
7. Administrator – sets meetings, keeps all updated, tracks documentation. This may be absorbed by the PM for smaller projects.

The project board typically includes multiple people: the customer, the end user and the supplier. The board checks for assurance from three perspectives. The customer ensures that the project is still financially viable, through a cost benefit analysis.

The user ensures that the user needs are being met. The supplier checks whether the project is working towards a realistic, practical solution.

Some projects outsource the assurance to an unbiased, third-party team.

1.24.4 Key Variables

There are six key variables that are managed. We identify six aspects or areas that need managing in every project. We make use of KPIs to measure performance goals and project tolerances. We Measure costs, timescales, quality, scope, risk, and benefits.

The scope is the work needed to complete the project. The details of the scope should be explained in the project plan. The costs are detailed in the project plan as well.

Timescales are how long it will take for each phase, in addition to the overall project duration. Risk management is the process of looking after the risks.

We also have the quality requirements of the clients or stakeholders. These need to be met and tested against standards. Benefits are the expected benefits of the project. This could be through a business case and a cost-benefit analysis to explain the purpose and financial or strategic benefits.

1.24.5 6 Types of PRINCE2 Documentation

Business case, risk register giving probability and impacts of risks and opportunities, quality register is a running log of quality checks and ensuring that deliverables meet expectations.

Issues register is a list of problems and concerns from team members. Lessons log is notes on lessons learned to apply to the next stage and future projects. Daily log is a daily diary by the PM that reports on activity and progress.

Chapter 2

Security

Lecture 18: Security by Design

2024

Project security is important as it protects against malice, mistakes and mischance. It helps prevent theft, fraud, destruction and disruption.

Security is all a game of risk management. If we don't spend enough time on securing applications, then they are more susceptible to being hacked and causing the business financial damage. On the other hand, spending too long on security can retract from the main objectives on the project.

We try and take preventative steps to avoid losing time, money, privacy of data, reputation and strategic business advantages. Security is an insurance model kind of business.

From 2005-2007, there were very few data breaches, then 2009-2011 some more, then more and more from 2015-2017. This happens as more and more data is available online and can be extracted by hackers looking to make a profit by selling this data.

Security is of paramount importance these days. Cyber attacks remain a continuing threat and are broadly increasing year on year. The government identify that roughly 40% of companies experience a cyber attack each year, with bigger businesses more likely to report attacks.

Each business loses on average £1,100 per cyber attack. This is typically larger for bigger businesses.

2.0.1 Aspects of Security Practice

This part of the module focuses on the secure application design aspect of security practices, complimenting secure implementation, secure infrastructure

design and deployment as parts that encompass the secure operation of the system as a whole.

2.0.2 Security Design Principles

A principle is a fundamental truth or proposition which serves as the foundation for a given belief or action. A security design principle is a declaration with the intention of guiding security design decisions to meet the goals of the system.

Within systems security, we might follow for example the following 10 principles:

- Least privilege possible
- Responsibility separation
- Trust cautiously
- Simplest solution possible
- Auditing of sensitive events
- Fail securely and use secure defaults
- Never rely on obscurity
- Implement defence in depth
- Never invent security technology
- Find the weakest link

We can ask 4 questions of each principle: why do we do it, what is the principle, what is the tradeoff, and what is an example.

Security by design is the process of designing the system with security in mind from the start, known as a proactive approach instead of a reactive approach.

As part of the SDLC, we introduce extra parts to each phase of the project. We add an initial risk assessment and add security requirements to the requirements phase, then follow secure design principles and follow threat modelling as part of the design phase.

During the implementation, we follow secure coding practice, then test our application with security tests and penetration testing.

Following this, we ensure a secure deployment of the application and setup some sort of vulnerability monitoring.

2.1 Threat Modelling

Lecture 19: Introducing Threat Modelling 2024

This is an important part of security by design as it lists possible issues that we may commonly encounter and good mitigations. These are known as CWEs, or common weakness enumerations.

The CWS is a list of software and harware weaknesses. From 2019-2023, 15 weaknesses have been present with potential mitigations. In addition to the CWE, OWASP (The Open Worldwide Application Security Project) provides a list of the top 10 commonly found risks.

Between 2017 and 2021, broken access control moved up from fifth place to first place. It is the most common thing that is broken, with on average 3% of applications having issues.

Following this, cryptographic failures are the next biggest, with sensitive data exposed due to poor cryptography. Then, we have injection, such as through forms, XSS is now a part of injection.

Insecure design is another category for issues, where the product can't be fixed by a perfect implementation following the definition. We therefore need to redefine the project so that we can have the needed controls to defend against specific attacks.

Following this, we have security misconfiguration. Pretty self explanatory. Then we have vulnerabilities and outdated components. Also pretty self explanatory.

We then have identification and authentication failures, which are kind of like security misconfiguration or broken access control but is sliding down thanks to standardised frameworks.

Software and data integrity failures were new in 2021, with assumptions to software updates, critical data, and CI/CD pipelines being made without verifying.

Security logging and monitoring failures was also a new category, and is hard to test for unless there is a breach. Finally, there is SSRF, where a web application fetches a remote resource without validating the user supplied URL.

2.1.1 Microsoft Security Development Lifecycle

Threat modelling is a core part of this lifecycle and is an engineering technique used to identify threats, attacks, vulnerabilities and countermeasures that could affect the application.

Using threat modelling can help ensure we meet the company security objectives and reduce overall risk.

There are five major steps to threat modelling as part of Microsoft's SDL:

- Defining security requirements

- Creating an application diagram
- Identifying threats
- Mitigating threats
- Validating that threats have been mitigated

This SDL is something that Microsoft have refined over the years and is the standard practice at Microsoft. The SDL defines mandatory security activities, in the order they should occur and grouped by the phases of the SLDC.

Activities in the SDL provide some form of security if implemented alone. Combining these as part of the software process provides even more security than doing each activity alone.

As part of the SDL, we have optional activities that can be added at the discretion of the project team or advisor to get the desired objectives. Focus should be places on the accuracy of the output at each stage. Reports that are fancy but lack the correct information are a waste of time.

2.2 SDLC Concepts and Terminology

Lecture 20: Security by Design Concepts and Secure Development Lifecycle

2024

- **Asset** – anything we need to protect
- **Threat** – anything that could let someone or something obtain, damage or destroy an asset, if we don't protect against it
- **Weakness** – underlying defect that modifies behaviour or functionality
- **Vulnerability** – weakness or gap in the protection efforts
- **Risk** – the potential for loss, damage or destruction of an asset, due to a threat having successfully exploited a vulnerability.

2.3 Secure Development Lifecycle

This is a shift-left initiative, where we integrate security checks as early in the SDLC as possible. We embed security in all stages of the SDLC, through requirements, design, development, testing and implementation.

It is typically much easier to remove the application security vulnerability during the design phase of the application than removing it from production.

At each stage of the lifecycle, we perform reviews of what has been done. We define requirements, then review and define them again in a repeating cycle. We then do the design, review that and redesign. We then develop the idea, test it,

then do a secure code review to see if there are any threats we might have missed. Following successful testing and code reviews, we deploy the application.

Following deployment, we perform penetration testing and maintain the application to make sure that the app meets the latest security standards. We can then go on to update the requirements definition and start the cycle again if something needs to be changed.

2.3.1 Agile Security

Agile, as seen earlier in these notes, is a form of adaptive software development. It is widely used in industry for software engineering products.

We add some definitions to the normal Agile definitions, with a security sprint approach, where we have a sprint which focuses solely on the application security. We only implement security related stories, and perform a secure code review.

We might have an input validation story, a logging story, an authentication story, and an authorization story.

In contrast to the security sprint approach, we might take an every sprint approach, which is similar to the MSDL in that we have requirements that must be completed in each and every sprint. We define requirements that are essential, and stories that are also essential. We can't therefore complete a successful sprint if we don't meet the underlying requirements.

2.3.2 Agile Sprint Comparisons

If we have a dedicated security sprint, then we can get all the security out of the way at the start of the project. We do this sprint first and once done we don't have to retroactively add security to the rest of the application.

This may not be possible at the beginning of the project though, due to possible missing mechanisms (e.g., no middleware setup in place). Good examples of these could be configuration of a bug tracking system, identification of security and privacy experts, creation of a baseline threat model and establishing a security response plan.

In contrast, an every sprint SDL removes the administrative workload at the start, and security will be added as time goes on. For example, updates to the threat model, communications of privacy impacting design changes to the privacy advisor, fixing issues identified by code analysis tools, following input validation and output encoding guidelines to defend against XSS.

2.3.3 Security Requirements

When establishing security requirements, we can look at the security risks that will be in the application. We can look at the information that the application

will be processing, the functionality within the application and by modelling use cases for the application.

We can also consider the group risk and internal audits to avoid conflict in later stages. We define things such as the length of passwords, security schemes and legal and regulatory security requirements as a group.

We can then see if the project is acceptable from an information security perspective and the key requirements that should be deployed.

We can model the architecture of the system to see where services connect and what security needs to be in place. For a typical application a web server might talk to an application server, which would then communicate with an underlying database.

2.3.4 Deployment

When deploying an application, we want to ensure that the minimum required privileges idea is followed, and ensure that for things such as firewalls, only the necessary ports are opened, e.g., 22 for internal only for SSH, 80 and 443 for the world, 8080 for example for connecting from the reverse proxies to the application, then something like 3306 to connect from the application servers to the database.

This doesn't stop lateral movement from within a host but prevents us from being able to go directly from the outside to the database server, for example. This provides a layered model of security as attackers can't brute force the database password, nor run vulnerability checking and penetrations externally without lots of complexity and utilising an exploit in either the web server or the application server, for example.

When connecting a browser to a web server, we need to consider how we will authenticate the users. The data to do this needs to be protected in transit, and impossible for the user to tamper with. We also want to ensure that the session can't be hijacked by an external entity, and prevent replay attacks.

On the web server, we want to ensure that the configuration is secure, handle exceptions properly, authorize the users, validate the inputs again to ensure that any client side validation still holds.

Once this is done, then the user can be presented to the underlying application server, which will again authenticate and authorize the user for specific business functions. We ensure that there is good auditing, monitoring and logging in place.

Finally, the data that the user is submitting or holds as part of authorization needs to be stored in a database. The credentials to connect to this database shouldn't be readily available, and any data that is sensitive should be encrypted or use a one-way hash function such that an attacker doesn't gain much information of the system or the users following a compromise.

2.3.5 Logical Zones

Essentially, compartmentalise different functions, such that people with different access levels can access different things through things such as the firewall. E.g., on the company intranet, maybe allow backend and management access to services, but restrict access through a DMZ from the internet and third parties to public facing aspects of the service.

2.3.6 Building Security into Design

In the design phase of the application, we build in various countermeasures to mitigate threats we might identify. We can identify threats by decomposing the application into its constituent parts and then looking at all the boundaries.

We can then plan the security testing phase based on the countermeasures we design in to ensure these countermeasures work for us. The outputs of this phase would be the security technical specifications and security test plans.

2.3.7 ...and Development

During development, we ensure that developers have taken security awareness programs and that there is some unit testing of security features of the application. They follow secure coding standards, code is checked against automated code review tools and an independent review is held by a third party or IT security to ensure the code is fit for purpose.

2.3.8 Testing

This is done to check security standards are being adhered to and that the implementation is correctly done. We execute the test plans that were made in the design phase, undergo independent pen testing and assessment of infrastructure, before a security release and sign off before pushing to production.

2.3.9 Code Review

Code review helps us to ensure that code is correctly written. Reproducing bugs can be quite complicated and makes delivery of new code slower. The majority of the time is spent getting the bug to reproduce, with much less time actually fixing it.

The cost of poor software quality in the US is at least $2.41 trillion. The accumulated technical debt is $1.52 trillion.

2.4 Essentials

Lecture 21: Security by Design: Essentials

2024

SBD has core properties. The core properties form the foundation on which all other things in security are built. These are:

- **Confidentiality** – guaranteeing access to data to only those with appropriate access rights according to their need to know the information
- **Integrity** – Data or activity can be verified to be authentic and not modified
- **Availability** – authorised actors can have access to the data or functionality whenever they need to.

These three properties support the goal of trustworthiness. In addition to these main properties, we have privacy, which is the right of not having information exposed to unauthorised third parties. This is related to confidentiality.

Safety, or the freedom from physical injury or damage to health, directly or indirectly stems from integrity and availability.

To maintain these core properties, we implement fundamental controls, or functional behaviours and capabilities. These fundamental security controls are:

- **Identification** – the actors in the system must be given a unique identifier within the system
- **Authentication** – allow the users of the system to provide a proof of their identity
- **Authorization** – authenticated actors can be given access rights according to their verified identity.
- **Logging** – provide traceability by recording performed events of the actors
- **Auditing** – records allow us to look back in time and understand the order of events

2.5 Building with Security in Mind

To build an application with security as part of the design, it is helpful to use a security framework. We start with an asset that we want to protect against varying threats.

There may be holes in the protection we provide, called exposure points. Think of a window in a bank vault, for example. To protect these assets, we implement security measures, for example removing the window or adding bars.

2.5.1 Design Patterns

When designing applications, we employ a zero-trust model. We ignore any prior trust relationships we may have and re-verify everything. For example, the web server might handle authentication, but the authentication may need to be passed back to the application. This avoids an exploit in the web server allowing full access to the application server.

We can also design by contract. Similar to how lock-and-key works, we design a server to expect specific input types and formats, and do not allow deviation from that.

We assign least privilege to each operation, providing just enough functionality for the operation to be performed successfully. We also employ defence in depth, such that an exploit in one layer doesn't compromise the whole system.

We keep things simple, avoiding over engineering and complex systems challenges. We don't assume anything secret will remain secret. We separate privilege, such that no one actor holds all access rights.

We consider the human factor. Too much security creates lazy users, who for example, might leave a post-it note with their password around the office.

We implement effective logging. We fail secure, not safe, and we build in the security, privacy and safety of the system from the beginning and not as an afterthought.

2.6 Finding Security Issues

Lecture 22: Intro to Threat Modelling

2024

Security issues may not be immediately obvious to the end user. Therefore, we have many different tools at our disposal to help us catch possible issues. These can include SAST, or static application security testing, where we perform static analysis of code.

We can also do fuzzing or other dynamic testing, and fuzzing implementations are being added to programming languages regularly. We can also perform penetration testing, with a range of different attacks on the application.

Finally, we have the classic bug reports, and these can be publicly published or developers can be informed through private channels, giving them time to fix the issue before an attacker might exploit it.

Further to this, we can also undergo threat modelling, looking at possible security issues and addressing them early on in the program. We understand the requirements of the program better, avoid adding bugs to the code, and improve stakeholder confidence as they will see that we are adding security as part of our goals.

Threat modelling is a combination of systems modelling and threat analysis. We create a representation of the system to see where possible threats might occur.

We can ask ourselves four questions to decide whether our threat modelling efforts will be successful:

1. What are we working on?
2. What can go wrong?
3. What are we going to do about it?
4. Did we do a good job?

Threat modelling is applied between the design and implementation stages of the system.

2.7 Security Impacts

We can use a risk assessment to identify the priority of threats. We can then decide the response to those threats and whether it is a passive or active measure. We define a level of assurance, which is the amount of guarantee that the treat is mitigated, and this will depend on the consequences of the failure.

If the threats are low likelihood, then the project leader and management can be left to decide how they want to handle them. High likelihood threats would require external assessment and a systematic review. They can influence test planning and other security activities on the project.

These activities are added to the requirements to provide assurance. Remember, we cannot test for the absence of a bug in our program, merely see that the bug does not occur on the documented test cases.

2.7.1 System Modelling

This is the process of creating abstractions or representations of a system. It makes use of diagramming to make the system easier to visualize and allow us to make changes and understand the system better.

We can model on a whiteboard, allowing us to make changes easily. We can brainstorm as well, then create formal diagrams once we know what is being done. We can create data flow diagrams, sequence diagrams, and state machines, which are all mathematical representations of code we want to write for the system.

A data flow diagram, as the name implies, will describe the flow of data. Sequence diagrams describe interactions of the system components in an ordered manner. Process flow diagrams, although not common, describe operational flow through actions in the system.

An attack tree provides steps along the path an attacker might try, and fishbone diagrams allow us to see the cause and effect between an outcome and the root causes.

2.7.2 Data Flow Diagrams

These are as follows: They describe the system in multiple diagrams, with

Element	Shape	Definition
Process		Operating unit or task within the system that receives / modifies/redirect input/output, e.g., web service
Container		A unit within the system that contains additional elements & flows (can be source/target).
External Entity		A process or a system involved in the operation of the system but not in scope of analysis e.g., web browser
Data Store		A functional unit where data is held permanently or temporarily e.g., database, file
Data Flow		Data movement between processes, data stores or external entities, e.g., connection strings, payloads
Trust Boundary		Trust-zone changes as data flows through the system. Entities within the same boundary have the same trust level.

Figure 2.1: Data Flow Diagram Elements

different abstraction layers in separate diagrams. There are usually lots of trust boundaries in the system. These are where two or more principals interact.

If we can't draw a trust boundary in the system, then either all parts of the system have the same level of privilege or all communications within the software are in the same boundary. Threats tend to occur around trust boundaries.

Decomposition

We iterate over processes and data stores to see where they need to be broken down. We draw a context diagram, which is high level to start with, and this shows the software and external entities.

We then more to a level one diagram, which is high level with the major business processes. We can then decompose these further in, you guessed it, a level 2 diagram. We can continue decomposition until we cannot decompose further. Et voˆila, we have a bunch of admin that we didn't really want.

We can draw up a table with the external entities, processes and data stores, and assign them a DFD item number. This ensures that we don't miss any parts of the system.

2.8 Threat Modelling Types

Lecture 23: Threat Modelling Exemplifying System Modelling Types

2024

Threat analysis is the process of formally evalujating the degree of threat to an information system or enterprise and describing the nature of the threat.

Within threat modelling, recall that an attack is an attempt to damage or gain access to the system, and an exploit is a successful attack. Threats to the system don't necessarily have to be malicious, they could be accidental, due to natural events, insiders, outsiders.

Threats to a system exist even if there are no vulnerabilities. They also change with the system changes.

We can model threats in many different ways:

- Social threats, where we attack people
- Operational threats, where there is a failure of policy and procedure
- Technological threats, where we have technical issues with the system
- Environmental threats, where we might have natural of physical facility factors (e.g., the maths building basement where some uni infrastructure is flooding on a somewhat routine basis)

An attack vector or threat vector is a way for an attacker to enter a network or system.

2.8.1 STRIDE

This is a model for identifying threats and includes Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, and Elevation of privilege. These are the opposite of the CIA et al., processes.

2.8.2 Phases

We use scenarios initially to understand the security requirements, and the boundary of the security problem. We look at any external dependencies to the program (e.g., the OS, web server, network) and define security assumptions, for example with the database will it encrypt columns, is there a key manager?

We can then move from understanding the security requirements to creation of trust boundaries, where we have the threats that put assets at risk. We look at the attacks that can be used to realise each threat.

We already described threat trees, and these are equal to attack trees. We build a threat tree for our application and an attacker builds an attack tree. We determine the risk for each attack, then prioritize.

We define the conditions for a successful attack, then plan and implement our mitigations.

2.8.3 DFD Keywords

For some reason, we now go back to data flow diagrams... not sure why we couldn't do this in the last lecture.

Data flow diagrams are composed of entities, with some applicable keywords. These are assigned meanings, with any related DFD entities and related threats.

Example. A university is creating a new website to provide online access to patrons and library personnel. We start by determining the requirements. What needs securing? What are the potential threats to the system? What are the limitations on building the system?

We create scenarios. Students can search databases and create holds on items they want. Staff can search the databases and reserve items for up to 15 weeks. Finally, librarians can do anything students or staff can do, show and hide items, and access some account information.

External dependencies include the operating system, off campus accessibility, a MySQL database, on the existing library server, a private network between the web and database server. IT policy dictates that both servers must be behind the firewall and all communications happen over TLS.

Assets include the library users and librarians, user credentials, user PII, a website system, database system, availability of the systems, user code execution on the website, database read access, librarian code execution on the website, database read and write access for librarians, ability to create users and ability to audit system events.

System roles might include an anonymous user, invalid user, student, staff, librarian, site administrator, database administrator, web server user, database read user and database write user.

We can then create a data flow diagram modelling the website and the trust boundaries.

We can then define the threats. For example, a user might evade the authentication system, an anonymous user might gather information from the authentication system, the anonymous user might forcibly browse to pages, librarian might have access to pages on the server, student or staff could modify privilege level, they could also browse to restricted pages, users could tamper with critical data on the client. SQL injection, JS injection, TLS version allows vulnerable algos?

2.8.4 Threat Tree

We look at a goal for a threat, the ways to achieve that goal, and then the mitigations we want to have in place. Tree roots are the goal for the attack, and the leaves are the ways in which to achieve that goal.

2.8.5 Planning Mitigations

We look through the list of threats and ensure that we have sane requirements, e.g., that credentialed users require username and passwords, all pages check authentication, all pages check authorization, no default accounts are left available, we use RBAC, all accesses use least privilege and fail secure.

2.9 System Models

Lecture 24: Threat Modelling: System Models

2024

When threat modelling, we first draw a diagram of the system. We can then identify the threats, provide mitigations and validate that those mitigations will work.

We ask four questions to establish whether the threat modelling efforts will be successful. These are:

1. What are we working on?
2. What can go wrong?
3. What are we going to do about it?
4. Did we do a good job?

When building system models, we look at the major building blocks, e.g., applications, servers, databases or data stores. We then identify the connections to each major block. For example, does the application support an API or a user interface? What talks to the database/data store? Does it only read or write data?

We also ensure the diagram shows all the trust boundaries and that we are able to tell a story without altering the diagram. If we can't explain a story or are lacking detail in a key area, then we break it down further.

2.9.1 Properties of a Good System Model

A good model is accurate, meaningful, representative and living. If the system has not yet been built, then it represents the design intentions or the realised implementation. By living, it means that the model changes with the system, and should reflect the current system design.

2.9.2 Attack Trees

These are formal, methodical ways of describing security of systems, with varying attacks. For example, an attack tree for a physical safe might look as follows:

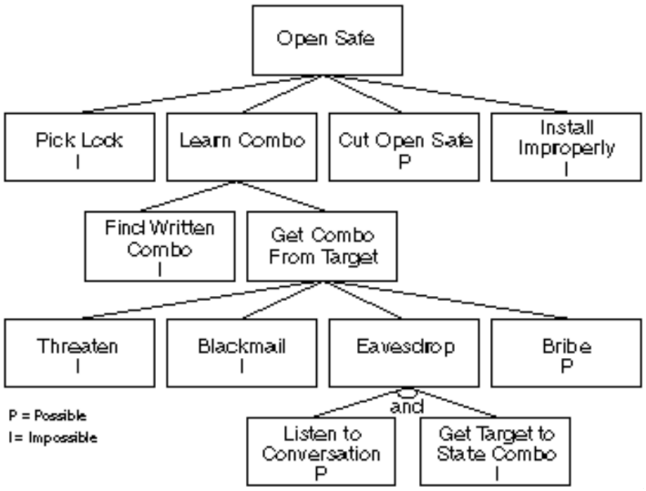

Figure 2.2: Attack Tree for a Physical Safe

You'll notice that the nodes are assigned with either a P for possible, or an I for impossible. This allows us to not spend time on impossible tasks, where we could have more luck doing something that is actually possible.

Another way to look at an attack tree is with special equipment requirements. Something that requires special equipment might have a higher cost to complete than no special equipment, or vice versa, acts as some sort of deterrent for the company we are threat modelling for.

Furthermore, we can assign the cost of an attack, estimating how much each approach will take. Similar to critical path analysis, we can work up the tree to find, say, the cheapest attack that doesn't require special equipment. For nodes that branch to multiple other nodes, we include the node that is cheapest and doesn't require special equipment, then can denote the path up the tree with a dashed line.

2.9.3 Understanding Cyber Security Risks with Attack Trees

Attack trees offer a hierarchy to represent the steps that are needed for a successful attack. We identify the core issue, create the root node for the issue, identify the steps to achieve the core issue. Add steps as nodes beneath the core issue, then rinse and repeat the process for each of the nodes.

2.9.4 Getting the User's Password

This is where the traditional XKCD comes in to save the day... instead of relying on encryption being too strong, we can simply drug and hit the person until they tell us the password.

Unfortunately, that's even worse than cracking their password, which is already illegal in itself. I'd just hit them to be honest.

2.9.5 NCSC and Telecoms

NCSC, or the National Centre for Cyber Security, have created a set of attack trees for telecoms networks. They create four classes of attack: espionage, disruption, pre-positioning, and national dependence.

Espionage would be the process of stealing or corrupting data in the network. Disruption would be stopping service on all or part of a network. Pre-positioning is the process of establishing a foothold in the administrative systems of the network. National dependence is the reliance on a third party for critical parts of the design, procurement, operation, support or incident management of the UK's networks.

2.10 Threat Modelling

AWS Lambda is a server-less computing service offered by AWS. We can draw a diagram with trust boundaries as follows:

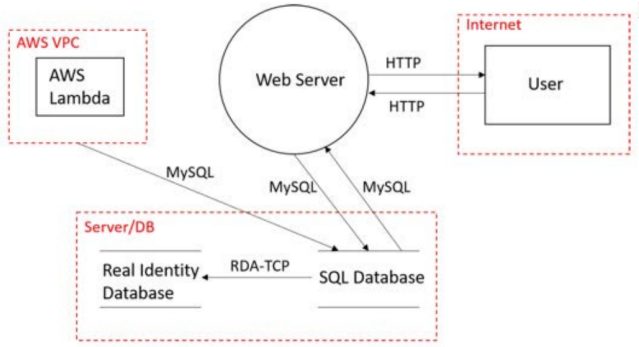

Figure 2.3: AWS Lambda Threat Modelling

We can make a diagram such as the AWS Lambda one with a tool like Microsoft Threat Modelling Tool (TMT). We can also use IrusRisk, which is another commercial offering. We also have OWASP, who provide Threat Dragon.

Another common tool is simply a whiteboard. These are all diagram-based tools. We might also have text-based tools such as OWASP pytm, where users

describe the model in a structured language, instead of drawing a system diagram directly.

Some of these tools include the methodology, e.g., STRIDE, CIA, and LINDDUN. These are covered in the next lecture.

2.11 Methodologies

Lecture 25: Methodologies

2024

As we know, threat modelling is the process of analysing a model of a system to highlight privacy and security concerns we might have. We ask four questions: what are we working on, what can go wrong, what are we going to do about it, and did we do a good enough job?

These govern the process of threat modelling and provide us a way to look at a system and secure it. When applying this to the design of a system, we look at what we are building, still identify the same threats, then instead of what are we going to do about it, we ask what are we doing to defend it against threats. The validation stage is still the same, and ensures that we have done enough mitigations to outweigh the cost of the risks.

2.11.1 Asset-Centric Approach

When we consider what could go wrong, we can focus on assets and how they should be defended. We look at the accessibility of these assets to hackers and how threats can affect the system as a whole.

2.11.2 Attacker-Centric Approach

This is from the point of view of the attacker, and uses attack trees and threat catalogs and lists to look at possible entry points into the system.

2.11.3 System-Centric Approach

This considers the system, and how it decomposes into different components (e.g., software and hardware), and the interaction between the components in the system, the external actors and the interface elements in the system. We have seen these expressed with data flow diagrams in previous lectures.

2.11.4 Attack Libraries

These are collections of knowledge that can be applied to a system and are a structured way to attack a system. These include CAPEC, which provides a hierarchical tree of attacks that can be applied to a system, and the ATT&CK matrix which is a similar thing.

These can also allow us to identify mitigation techniques for a system too, as they are not just used by attackers. These give consistent results when repeated and can be used when we have a thorough understanding of the system.

There is, however, a limitation to the libraries in that we don't consider future attacks as these aren't documented in past attacks.

2.11.5 STRIDE

This is a mnemonic that can be used in threat modelling. It is:

- Spoofing – can the attacker gain access using a false identity
- Tampering – can an attacker modify data as it flows through the application
- Repudiation – if an attacker denies doing something, can we prove he did it
- Information Disclosure – can an attacker gain access to private or potentially injurious data?
- Denial of Service – can an attacker crash or reduce the availability of the system?
- Elevation of Privilege – can an attacker assume the identity of a privileged user?

STRIDE provides a workflow, which is a flowchart of the way we build a threat model with this methodology. It is an iterative process. The process is outlined in the following diagram:

To model a system with STRIDE, we first model it using data flow diagrams, then find threats by looking at defined threats for that part of the name. We can possibly use a checklist of this step, or an attack tree.

We can combine this with a risk assessment model, such as DREAD to prioritise threats. We can then identify mitigations to these, validate and repeat the process.

2.11.6 Ranking Threats

Threats can be ranked from the perspective of risk factors. These can be used to rank threats as high, medium or low risk, where we inform the risk based on the impact, possibility of risk, and the ease of exploitation. We can model risk with the DREAD, which is a mnemonic for different parts of the threats, with each ranked 1 to 10, then these are summed and divided by 5 to create a DREAD score. Those with higher DREAD scores are more of an issue than those with a low score.

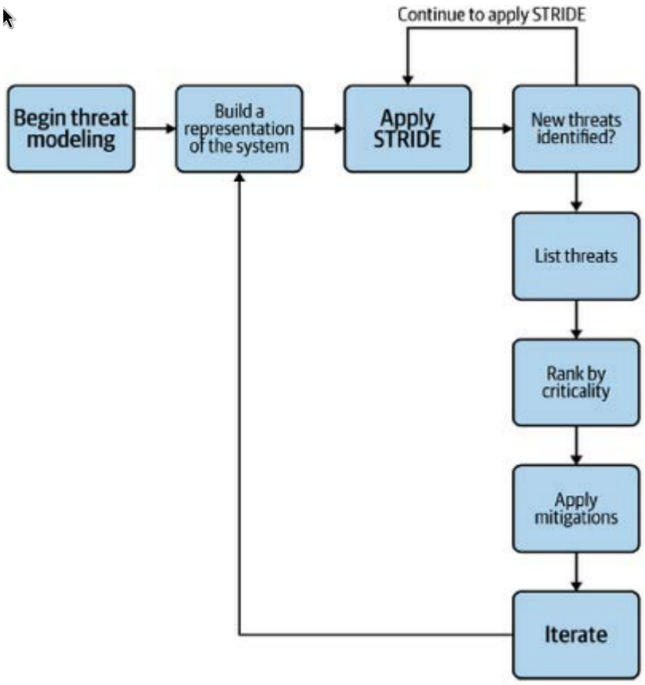

Figure 2.4: STRIDE Workflow

This is a Microsoft score. Each of the following create the mnemonic for DREAD:

- Damage – how big would the damage be if the attack succeeded?
- Reproducibility – how easy is it to reproduce an attack?
- Exploitability - how much time, effort, and expertise is needed to exploit the threat?
- Affected Users – if a threat were exploited, what percentage of users would be affected?
- Discoverability – how easy is it for an attacker to discover this threat?

For STRIDE, we list the property that is violated as well, to ensure that we cover CIA et al. We can map the STRIDE properties to a data flow diagram element, e.g., we can spoof an external entity or process, tamper with a process, data store or data flow, etc. The following matrix shows the STRIDE per DFD element:

STRIDE also lists a set of mitigations, where we can prevent things such as spoofing identity with appropriate authentication, protection of secret data, not storing secrets.

External Entity	Y		Y		
Process	Y	Y	Y	Y	Y
Data Store		Y	?	Y	Y
Data Flow		Y		Y	Y

Table 2.1: STRIDE Mapping to DFD Elements

We can prevent data tampering with appropriate authorisation, hashes, message authentication codes, digital signatures and tamper resistant protocols. For repudiation, we can make use of digital signatures, timestamps and audit trails.

We always use hashing for one way encryption, such as storage of passwords.

For information disclosure, we can ensure we have authorisation, privacy enhanced protocols, encryption, protection of secrets, not storing secrets. For denial of service, we can ensure appropriate authentication and authorisation. We can filter, throttle or apply quality of service measures. For elevation of privilege, we also run applications with the least privilege possible.

2.12 Threat Profile

Once threats and countermeasures have been identified, we can derive a threat profile, including non-mitigated threats, partially mitigated threats, and fully mitigated threats.

STRIDE is the most mature of these threat models, and has been successfully applied to cyber only and cyber-physical systems. It helps us to identify mitigation techniques, and is easy to adopt. Unlike other modelling, it puts no constraints on the source of the threats, and allows us to generate threats freely with no biases, based on experience and research.

As the system increases in complexity, the number of threats increases rapidly.

Lecture 26: Threat Modelling Methodologies 2024

Recall that last lecture, we discussed STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Escalation of Privilege)

This threat model can be applied to different elements of a data flow diagram. For instance, we might have a simple message encryption system (why would we, as we should never invent our own security)

This is shown below:

When applying STRIDE, we could show external entities such as Alice and Bob as spoofable, in addition to the key server.

For tampering, the key server, key repository and key requests would all be at risk. Repudiation could include Alice, Bob and the key server. The list

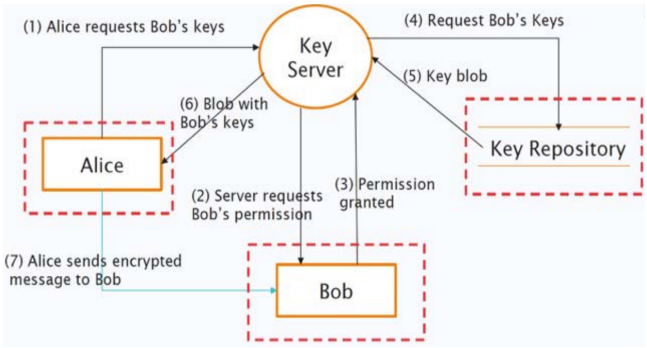

Figure 2.5: DFD for Message Encryption Service

continues and we can see that there are lots of issues with our process.

For example, because we can spoof Alice, we should create a user authentication system. For tampering, data may be attacked in between the key server and repository. If we use TLS for securing the connection between these, then that data is guaranteed to be free from tampering.

For repudiation, we can ensure that we make a log on a separate system. For information disclosure, we may want to mask the identifiers that the system uses for communications. For denial of service, we might want to implement flow rate limitation at the session and network levels.

Finally for elevation of privilege, we should harden the database and reduce the privileges it has when running and validating the inputs.

2.13 Qualitative Risk Model

A generic risk model considers risk a calculation. The risk is the likelihood and the impact of an attack. Risk is the probability that the threat occurs x the cost to the organisation.

2.14 Application Security Risks

Attackers can take different paths through the application to do harm to the business or organisation. These paths represent risks that may be serious enough to warrant attention. These may be trivial to find and exploit or extremely difficult.

2.14.1 Likelihood Rating and Risk Ranking

When modelling application security risks, we can employ the use of a table. The table has different columns for different things, such as threat agents, ex-

ploitability, weakness prevalence, weakness detectability, technical impacts and business impacts.

These follow the path through the application. Threat agents are application specific, the next 4 sections are ranked from 1-3, with 3 being the worst in terms of the attack. Finally, the business impacts are specific to the business that is being targeted so these do not get a rating.

The application security risk scores and meanings are given in the table below:

Threat Agents	Exploitability	Weakness Prevalence	Weakness Detectability	Technical Impacts	Business Impacts
Application Specific	Easy – 3	Widespread – 3	Easy – 3	Severe – 3	Business Specific
	Average – 2	Common – 2	Average – 2	Moderate – 2	
	Difficult – 1	Uncommon – 1	Difficult – 1	Minor – 1	

Table 2.2: Application Security Risks Table

The likelihood rating is the average of the exploitability, prevalence, and detectability. We then multiply this by the technical impact score to get the risk rating.

As part of qualitative risk modelling, we can combine the likelihood and risk ratings together based on the description of the problem. Sometimes the exploitability risk may be known as the vector. We can then add threat IDs, descriptions and the vector, prevalence, detectability and impact scores to a table, then compute the total risk as avg(vector, prevalence, detectability) $\times$ impact.

2.15 LINDDUN

We ask four questions as part of our threat modelling. As part of the threat identification, we can use LINDDUN, which is an anagram:

- L inkability threats against unlinkability
- I dentifiability threats against anonymity and pseudonymity.
- N nonrepudiation threats against plausible deniability
- D etectability threats against undetectability and unobservability
- D isclosure of information threats against confidentiality

U nawareness threats against content awareness

N oncompliance threats against policy and consent compliance

It is a methodology focused on privacy. It consists of 6 steps and is a systematic approach to a privacy assessment. The 6 steps are as follows:

1. Define data flow diagram
2. Map privacy threats to DFD elements
3. Identify threat scenarios
4. Prioritize threats
5. Elicit mitigation strategies
6. Select corresponding PETs

PETs = Privacy enhancing technologies.

The first 3 of these are in the problem space, and the remaining are in the solution space. LINDDUN is similar to STRIDE in that it is a risk assessment but it places emphasis on the privacy implications of the system.

LINDDUN, whilst a useful tool can be very time consuming to implement.

Lecture 27: LINDDUN

2024

As discussed in the last lecture, LINDDUN is a privacy focused risk assessment modelling tool. Here, we cover each of the threats in more detail:

Linkability

Learning more data about the individual by matching related data items together

Identifiability

This is where the identity of the data subject can be learned through leaks, deducted or inferred. Identifying is learning who the individual is.

Nonrepudiation

This is where we have outcomes where an individual is not able to deny certain claims about their involvement in the system. Nonrepudiation is being able to attribute a claim to an individual.

Detectability

This is becoming aware of data subject involvement, membership or participation to the system.

Disclosure Threats

These are where disclosure of personal data might be considered problematic. Excessive collection, storage, processing and sharing of personal data is data disclosure.

Unawareness

This involves assessing the privacy harm to the subject by insufficient informing, involving or empowering of the data subject in its role and relation to the system.

Non-compliance Threats

This is lack of adherence to legislation, regulation, standards and best practices, leading to incomplete management of risk.

2.15.1 Threat Trees

These are similar to attack trees but are LINDDUN specific. These are found in the catalog and are not to be memorised.

2.15.2 Application to Threat Models

We first make the context diagram, which gives a high level overview of the system. We decompose this diagram into a L1 diagram, giving the system in more detail. Where we have a process, we document this as PN, where N is a monotonically increasing number.

We model data flows similarly, with DFN, for each dataflow. We can then build a powerset in a way of the 'Scr.' or source, data flow, destination and location for the threat to occur. The source, dataflow and destination columns will be repeated for each row, where the location of the threat is.

We then work through the acronym and the threat trees, and assign a column of characteristics and a description of the threat.

Following the threat description for each source, dataflow, destination and location, we can generate a threat description.

The threat description includes the threat title, a summary, the involved DFD elements, the threat type and the nodes of the LINDDUN threat tree. We also note the assets that might be involved and the priority of ensuring that the threat is modelled correctly.

2.15.3 Comparison of Security Threat Modelling to Privacy Threat Modelling

Security threat modelling focuses on the protection of company assets and malicious actors that may want to cause deliberate harm. LINDDUN focuses on

the privacy mindset (e.g., medical data) and protects personal data and data subject rights. We view threat as people causing unintended harms to privacy.

Both allow us to create a system model, identify threats systematically, provide mitigations and reflect and evaluate.

Lecture 28: More STRIDE 2024

As per usual, we know the four questions. What are we working on? What can go wrong? What are we going to do about it? Did we do a good job?

The building question can be answered with a data flow diagram. Building these is to understand the system. Analysing them is understanding the threats to the system.

STRIDE helps us find threats, and makes use of attack trees and risk assessment methods such as DREAD. We then identify mitigation strategies and repeat.

STRIDE is an interplay of attacks, mitigations and requirements. Requirements drive threats. Threats expose requirements. Threats need mitigations. Mitigations can be bypassed and finally unmitigatable threats drive requirements.

We plan for mitigation, and can address threats in one of four ways: do nothing, remove the feature, accept the vulnerability, or counter the threats with technology.

S from stride is spoofing, and the security property is authentication. We authenticate principals with things such as Basic, digest, cookies, PKI, digitally signed. To authenticate code or data, we use digital signatures, MACs, and hashes.

To prevent tampering, we implement ACLs, signatures, MACs and tamper resistant protocols. For repudiation, we include strong authentication, secure logging, digital signatures, secure time stamps and trusted third parties.

Information disclosure is prevented with confidentiality. We can use encryption, ACLs, privacy enhanced protocols, protect secrets, and don't store secrets.

Denial of service can be prevented with ACLs, filtering, quotas, throttling, authorisation and HA designs. For elevation of privilege, assign ACLs, group or role membership, privilege ownership, permissions, input validation and run with least privilege.

2.15.4 Custom Mitigations

These are used where standard technologies don't work. It is easy to get these wrong and it's expensive and hard to test. For validation, we check diagram matches final design of the system, we've enumerated the threats and mitigated them. QA is where we check the quality of threats and mitigations, checking we cover all threats and assumptions well.

2.15.5 LINDDUN Mitigations

LINDDUN not only has threat trees but a catalog of privacy mitigation strategies and privacy enhancing technologies it employs.

Most DFD elements can be applied to LINDDN, with the exception of content unawareness, which is specific to the entities. Entities are only able to be linkability and identifiability and content unawareness, not the others.

We can then map the privacy objectives with privacy enhancing techniques, again from the catalog and again not needing memorisation.

2.15.6 Core Properties

Recall CIA triad. These are known as the core properties. These support the key goal of trustworthiness. We also have privacy, which is the right to not have information exposed, which is related to confidentiality; and safety, which is the freedom from physical injury or damage to health whether directly or indirectly. Privacy and safety are related to CIA, but have different focuses.

CIA modelling is much simpler. We might model a webserver and want to model whether confidentiality might be broken, integrity of the data or the availability of the server.

We write the types of breaches as part of a tree, then look at the users or actors that might be able to attack or maliciously affect the system. We then look at the attacks that could happen and the goal is to identify a log source for that attack. We can use these log sources to detect where an attack might have taken place, and risks that might be on the system.

2.15.7 Why not threat model?

Takes time, needs lots of knowledge and a whiteboard for critical thinking. To solve these issues, we can automate with tools but these depend on the quality of the system model.

Lecture 29: Methodologies Please be Done 2024

2.16 Review: Fundamental Security Controls

We have identification, authentication, authorisation, logging and auditing. We have basic design patterns for systems: zero trust, design by contract, least privilege, defense in depth, keep things simple, no secret sauce, separation of privilege, consider the human factor, effective logging, fail secure, build in not bolt on.

2.17 PASTA

Process for attack simulation and threat analysis. Risk centric. Employs attacker-centric perspective to produce asset-centric output in threat enumeration and scoring. Quantifies the risks that impact a business or system. 7 stage process:

1. Define Objectives
2. Define Technical Scope
3. Application Decomposition
4. Threat Analysis
5. Vulnerability and Weaknesses Analysis
6. Attack Modelling
7. Risk and Impact Analysis

Offers views from many angles inc probable attacks and attack vectors, mitigation and risk acceptance. Encourages collaboration. Not lightweight and benefits from knowing design and implementation details beforehand.

2.18 Trike

Framework for security auditing from risk management perspective. Focuses on point of defender. Uses data flow diagram and maps to actors and assets. IDentify threats as elevations of privilege or DoS. Create attack tree from each discovered threat.

Helps to identify mitigation techniques and contributes to risk management. Available tool with automated components. Has poorly documented methodology.

2.19 OCTAVE

Operationally critical threat, asset and vulnerability evaluation. Risk based assessment and planning technique for security. Targets organisational risks not technological. Identifies mitigations, risk management and threat prioritization. Is scalable, but time consuming and has vague documentation.

2.20 VAST Modelling

Based on ThreatModeler, which is automated threat modelling platform. VAST creates two types of models: application threat models with process flow diagrams and then operational threat models which is the attacker POV based on DFDs. Helps identify mitigation techniques. Contributes to risk management,

auto prioritization of threat mitigations. Consistent, automated and scalable but has little public documentation.

Visual, agile and simple threat is what it stands for.

2.21 Security Cards

Brainstorming technique, not formal method. Uses deck of cards to answer questions about attack. Who, why, what assets and how implemented are asks. Encourages collaboration, and out of ordinary threats. Increases knowledge but leads to many false positives.

Lecture 30: Methodologies Last One

2024

This lecture essentially walks through the threat modelling methodologies that have been discussed, and examples of them from OWASP etc.

Anything of note is added below.

A concerted threat is a threat with capabilities of multiple threats all at once.

Microsoft SDL is mandatory security activities, in the order they should occur and as phases of the traditional SDLC.

Continuous threat modelling is needed where we employ a system like Agile. We have guiding principles for this. Teams know the system better than anyone. Teams cannot stop to engage in threat modelling. Quality of threat analysis grows with experience. State of threat model reflects current state of system. Today's model needs to be better than yesterday's and the findings need to match the system.

These make it accessible and agile, educational and unconstrained, representative, scalable and educational, and finally useful.

We ask 4 questions. What are we working on? What can go wrong? What are we going to do about it? Did we do a good job? These make the threat modelling effort successful.

We track threats and assumptions in a table, organising threats with IDs. Assumptions are where we ignore threats for example.

Threats can be tracked by the order of discovery, and given a bug ID. We can fix, mitigate, accept or transfer risk for each threat. For assumptions, we check them and if they are wrong, then we might have to reconsider what goes wrong.

LINDDUN is privacy by design instead of security by design. It is a key concept in GDPR.

It is proactive and preventative, with privacy as the default setting. Privacy is embedded into the design. Full functionality is kept the whole time, and no

tradeoffs to functionality are made for privacy. We implement E2E security, keep the system open and respect user privacy.

LINDDUN refers to the threats. Each threat has one or two privacy properties. LINDDUN has 6 steps, first 3 in problem space, last 3 in solution space. Employs privacy enhancing techniques. This is a taxonomy of mitigation strategies.

STRIDE is threat modelling from a security and organisational security perspective. Doesn't worry too much about privacy. Attacker doesn't respect protocol or use our tools to access our servers. Attacker spends more time than us looking at flaws. Attacker may have large number of machines available for performing complex calculations.

CIA threat modelling is the most basic of threat modelling. Look at each principle, in terms of availability, then integrity, then confidentiality.

DREAD is the subjective risk mopdel. We assign a DREAD score 1-3 for each of the risks. Calculate average of all scores to get the risk.

Attack trees decompose high level goals into individual attacks.