**Software Project Management Notes: Table of Contents**

[**25/09/23 Introduction to SPM 5**](#_f1g8u694rloc)

[Characteristics of a Project: 5](#_8233eob5xgli)

[Project Management Cycle: 5](#_6q7m6sa82p06)

[Requirements: 5](#_krxjlmx9230k)

[Design and Code: 6](#_6tor0hit0jly)

[Categorising Projects: 6](#_raezz99ro8fj)

[Stakeholders: 6](#_laipnaa2cm5l)

[Setting Project Objectives: 6](#_rdwy2nvlxw2h)

[Project Management Goals (Q, C, T, P): 6](#_z9m58fagzauc)

[Software Process: 6](#_xjsspayblreo)

[Software Process Issues: 7](#_onb5u1wzl85p)

[Components of Software Project Management: 7](#_mh1czfk8csp3)

[**02-09/10/23 Measurement, Estimation & Data Analysis 1 8**](#_ktu8j1j84m41)

[Role of Measurement, Estimation, and Data Analysis 8](#_actn77umkbas)

[Measurement, Estimation, and Data Analysis (Operational Level - Short Term) 8](#_edpuhu3fjn7t)

[Measurement, Estimation, and Data Analysis (Tactical Level - Medium Term) 8](#_x5eah5j8qc2w)

[Measurement, Estimation, and Data Analysis (Strategic Level - Longer Term) 8](#_gmwmk6dv137s)

[Use as a Management Tool to Answer Questions 8](#_tmxdfqngharl)

[What Is Measurement? 9](#_nxvd3nwyum9b)

[What Entities/Attributes to Measure? 9](#_rq9uninbvzk5)

[Units 9](#_5i5gpj5p95yi)

[Scale Types 9](#_3sg1nedcela)

[Nominal Scales 9](#_rsv4orvbqhqx)

[Ordinal Scales 10](#_18hire6kzfwj)

[Interval Scales 10](#_ri7bbsbio7b1)

[Ratio Scales 10](#_rpzolymsavtk)

[Absolute Scales 10](#_sszegep6jfrz)

[Measurement Validity 10](#_9llqrijkr9h)

[Measurement Accuracy 10](#_xqbba1vnex64)

[Estimation 11](#_15k0pymp0sse)

[Resource Prediction 11](#_yuf0om2ajhbo)

[Modelling Uncertainty 11](#_svn8alp7ihma)

[Confidence Intervals 11](#_hpg23dqfkwe5)

[Ways of improving estimations: 11](#_ze8bcvqurhx4)

[The Delphi method: 11](#_dafq09y2hrkj)

[Estimation by Decomposition 12](#_febnyd2j8e4u)

[Estimation by Analogy 12](#_b9788s9ou4ll)

[Units: 12](#_2dvffyz5zr3t)

[**16-23/10/23 Software Project Planning 1 14**](#_6o6uki3mix0l)

[Motivation 14](#_t17vgstokby6)

[Project Objectives 14](#_fsc5bybwx8sy)

[Step Wise Project Planning Framework 14](#_9jxhkosbux3n)

[Step Wise - an overview 14](#_vh1agxc3gc2e)

[Step 1: Identify Project Objectives 15](#_tyx2t0fspvm5)

[Step 2: Identify Project Infrastructure 15](#_hvh3644hea85)

[Step 3: Analyse Project Characteristics 15](#_o4wszefqvd4z)

[Step 4: Identify Products and Activities 15](#_gur4867nvols)

[Step 5: Estimate Effort for Each Activity 16](#_ifotigtf9lmg)

[Step 6: Identify Activity Risks 16](#_2o412qhlinpr)

[Step 7: Allocate Resources 16](#_v8d3mu46q2uk)

[Step 8: Review / Publicise Plan 16](#_t4zxxjbws7su)

[Steps 9 & 10 16](#_in0pexd6gp9e)

[Prioritising Activities 17](#_tq4ur6sbkx0d)

[Resource Allocation 17](#_vpc9vk2c9cr7)

[**23/10/23 Software Project Planning 2 18**](#_z7weq7o3ixrc)

[Building vs. Buying Software: 18](#_qbhdjp45f07o)

[Advantages of Off-the-Shelf Software: 18](#_xwxy5pnl8mzt)

[Disadvantages of Off-the-Shelf Software: 18](#_stn8eev3226)

[Structured Approach vs. Agile Methods: 18](#_d743o2j3002r)

[Choice of Development Process Models: 18](#_6lig0lb8gkra)

[The Waterfall Model: 19](#_wg3gyr2egth8)

[Waterfall Advantages: 19](#_ocv3xfv5kzeh)

[Waterfall Disadvantages: 19](#_az9x9spab9td)

[Projects Suited for Agile Delivery: 19](#_pxlvtwwe7thk)

[Extreme Programming: 19](#_phdmmcslprej)

[Prototype Terms: 19](#_lthm6uvty1ti)

[Incremental Delivery: 19](#_xl29hn2n7s11)

[Choosing a Model: 20](#_ieg1znf0gomc)

[Steps for Activity Planning and Resource Allocation: 20](#_615n0sx1g42f)

[Prioritising Activities 20](#_xbcc7ibl03xf)

[Resource Allocation 20](#_6gziu0rxs0fm)

[Float is the amount of time a project can be delayed with it affecting the project's overall schedule. 21](#_af75xvnid3w)

[TOTAL FLOAT = LATEST START - EARLIEST START 21](#_3uzdeart2e8o)

[**30/10/23 Project Risk Management 22**](#_poy0s4ds4o27)

[Categories of Risk 22](#_9asx1n6xtloc)

[Categories of Risk (Based on STAT: Structure, Technology, Actors, and Tasks) 22](#_6m1vbsh91saf)

[A Risk Management Framework 22](#_d5q2ohg2knqd)

[Quantitative Risk Analysis 23](#_87umjw7bsoap)

[Qualitative Risk Analysis 23](#_apfzcn6gkx31)

[How to Deal with Risks 24](#_ijwnbuit2xww)

[Overall Goal of Risk Management 25](#_p84p7pl4fq2o)

[PERT (Program Evaluation and Review Technique) 25](#_ml1588ny585z)

[Normal Distribution in PERT 26](#_twjxb2c1on2w)

[**13/11/23 Project Monitoring and Control 27**](#_8qzbm4o8qu5h)

[Project Control Cycle 27](#_p9qjclkamth2)

[Alternative Cycle- Scrum 27](#_db0hpqmhbudh)

[Responsibilities 27](#_a0053xf555ni)

[Alternative View- Scrum 27](#_41tn5sok9wfa)

[Assessing Progress 28](#_g5sqrf9wmn3e)

[Collecting Progress Details 28](#_a7ox7hsx7pc6)

[Red/Amber/Green (RAG) Reporting 28](#_q122u2v9gk1n)

[Cons 28](#_f21b9mm82nk4)

[Alternative to RAG: SCRUM 28](#_7ole73igi1f7)

[Using Gantt Charts 28](#_bqh737tzvlh6)

[Using Slip Charts 28](#_mg4ivkhaevxy)

[- Bent 'Today' Cursor: 28](#_u4rdquetts2t)

[Limitations of Gantt/ Slip Charts 29](#_7e91cwo8cjbn)

[Timeline Representation 29](#_t1jdy6210cx0)

[- Depicts planned and actual time against project activities. 29](#_hbxkm84c35ig)

[Cost Monitoring 29](#_yx4643cs6z60)

[Partly-Completed Tasks 29](#_cwtsdc7t6jcz)

[Exception Planning 30](#_6ohhxrz91e1k)

[- Project manager can change the plan with an exception report. 30](#_y8bjydftp4ec)

[Prioritising Monitoring 30](#_icayn8f7ff7d)

[Getting Back on Track: Options 30](#_y6s5issnuu80)

[**20/11/23 Quality Management 31**](#_77vnqyhdzkgy)

[Software Quality in the ‘Step Wise’ Framework: 31](#_jk9c7sokz31i)

[Product Quality Management (Quality Control): 31](#_ikfibpy1tuwm)

[Process Quality Management (Quality Assurance): 31](#_yk2w5w6qorj5)

[Classification of Quality Characteristics: 31](#_f853spmxhg1h)

[Direct vs. Indirect Measurement: 32](#_12q13mn8o3xa)

[Defining a Quality Management Plan (Quality Planning): 32](#_x9kkgiimwzgw)

[Defining a Quality Characteristic for a Software Product: 32](#_8f7w5ygl4bt)

[Measurement of a Quality Characteristic: 32](#_tqowkyuxm53t)

[Software Quality and the ISO 9126 Standard: 32](#_kepw067izuf7)

[ISO 9126: Functionality: 32](#_dw8tuspco5na)

[ISO 9126: Reliability: 33](#_mqxcj0tc1fh8)

[ISO 9126: Usability: 33](#_m60m5uuvghvc)

[ISO 9126: Efficiency: 33](#_t64touogs02j)

[ISO 9126: Maintainability: 33](#_7c2wlna5sd3h)

[ISO 9126: Portability: 33](#_4hu54wo9ng0r)

[Key Points 33](#_na3nw678t473)

[**27/11- 04/12 Agile Methodologies 34**](#_6w1puyi8yxzn)

[Agile Basics: 34](#_9lwca1n2jjoo)

[Waterfall Model: 34](#_jqww4zxe21nn)

[- Advantages: 34](#_18hsqkqbhjnr)

[- Disadvantages: 34](#_1grstme9tvr6)

[What is Agile Development? 34](#_j0o01nihudix)

[- Projects Suited for Agile: 34](#_u304y24jvuts)

[Agile Benefits: 34](#_tmpfcj683ylp)

[Agile Drawbacks: 34](#_cg1lz29l5wt3)

[Agile Approaches: 35](#_hmmb3xsfxq0h)

[Agile Manifesto Values: 35](#_tamjlrjxment)

[Continuous Improvement: 35](#_xgzvqrii98py)

[Sustainable Pace: 35](#_3hlz8z4zcnw7)

[System of Prioritization: 35](#_s1jpke4k6hh5)

[Scrum Framework: 35](#_nefcm470pht8)

[Three Pillars of Scrum: 35](#_xwbkq8jkgpjg)

[Scrum Artefacts: 35](#_kdy0j91w8rq5)

[Roles in Scrum: 36](#_h8le0twipd25)

[Scrum Events: 36](#_88w60jh7vnu7)

[Product Owner: 36](#_1aaoknnjzz15)

[Scrum Master: 36](#_xz3w1v1kcj8d)

[Scrum Advantages: 36](#_uy1ocwap5e43)

[Scrum Disadvantages: 37](#_8h0xtf7oe08m)

[Kanban Steps: 37](#_l7u8386jhe7s)

[Advantages of Kanban: 37](#_u4ggxoh1zbbe)

[Disadvantages of Kanban: 37](#_xv3fd5kykw3g)

[Scrum vs. Kanban: 37](#_f17k5tlf937p)

# **25/09/23 Introduction to SPM**

Flashcards: <https://revisely.com/flashcards/packs/Y2JEn>

Jobs versus Projects:

- Jobs: Repetition of well-defined, well-understood tasks with little or no uncertainty.

- Exploration: Involves high uncertainty (e.g., finding a cure for a disease) regarding outcome, duration, and budget.

- Projects: Fall in between, combining elements of routine jobs and exploration.

## Characteristics of a Project:

- Non-routine

- Planned

- Aims at a specific target

- Carried out for a customer

- Involves a temporary work group with multiple specialisms

- Comprises several phases

- Constrained by time and resources

- Often large and complex

## Project Management Cycle:

1. Feasibility study: Determines if the project is technically feasible and worthwhile from a business perspective.

2. Planning: Develops a plan if the project is feasible.

3. Implementation/execution, termination, and evaluation: Executes the plan, potentially making adjustments as needed.

## Requirements:

- Requirements elicitation: Identifies client needs.

- Analysis: Converts customer-facing requirements into developer-understandable equivalents.

- Requirements cover functions, quality (non-functional requirements), and resource constraints (costs).

## Design and Code:

- Architecture design: Based on system requirements, defines system components.

- Code and test: Focuses on individual components.

- Integration: Combines components.

- Qualification testing: Tests the entire system.

- Installation: Makes the system operational.

- Acceptance support: Includes maintenance and enhancement.

## Categorising Projects:

- Distinguish between different project types based on factors like voluntariness, information systems vs. embedded systems, and objective-based vs. product-based.

## Stakeholders:

- Stakeholders include users/clients and developers/implementers.

- Different stakeholders may have conflicting objectives, necessitating the definition of common project objectives.

## Setting Project Objectives:

- Objectives should be SMART (Specific, Measurable, Achievable, Relevant, Time-Constrained).

- FYP:

S- designing website   
M- user engagement assessed through testing  
A- uses skills taught in previous university modules  
R- aligns with project purpose   
T- needs to be completed by April 22nd deadline

## Project Management Goals (Q, C, T, P):

- Quality software product

- Cost control

- Timely completion

- Productivity optimisation

## Software Process:

- Involves activities, relationships, dependencies, resources, and constraints used to develop, maintain, and evolve software.

## Software Process Issues:

- Operational, Tactical, and Strategic levels address various aspects, including quality, time, cost, productivity, and long-term organisational goals.

## Components of Software Project Management:

- Planning, organising, staffing, directing, monitoring, controlling, innovating, and representing are essential components of managing a software project.

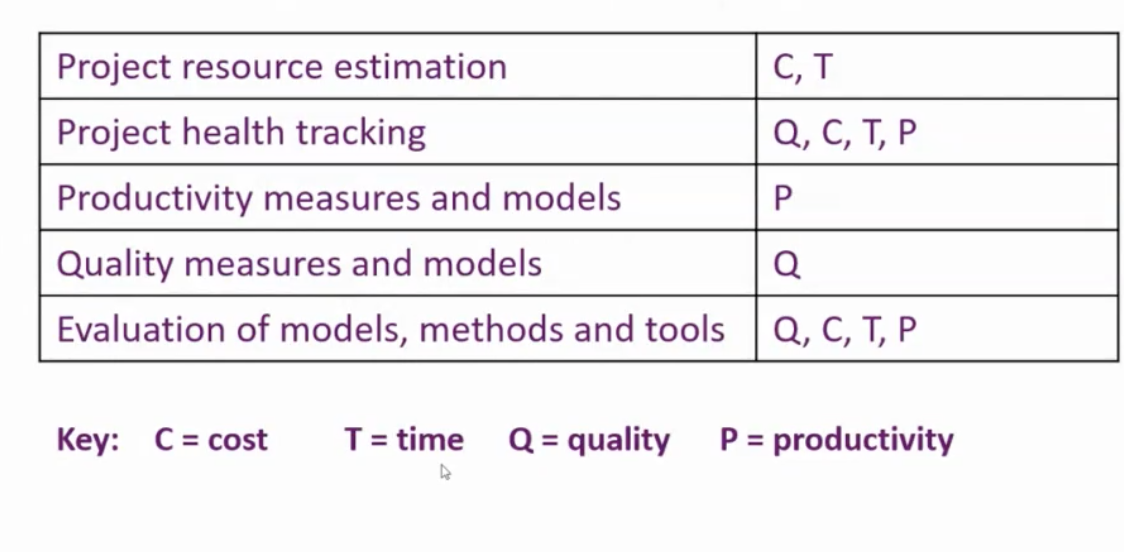
# **02-09/10/23 Measurement, Estimation & Data Analysis 1**

Flashcards: <https://revisely.com/flashcards/packs/ULN6d>

## Role of Measurement, Estimation, and Data Analysis

- Support quantitative comparisons, evaluations, predictions, and decision-making in software projects.

- Examples: Project resource estimation, project health tracking, productivity and quality measures, evaluation of models, methods, and tools.



## Measurement, Estimation, and Data Analysis (Operational Level - Short Term)

- Collect raw data.

- Measure, estimate, predict.

- Analyse data statistically.

- Develop and apply quantitative models.

- Interpret data analysis outcomes.

- Experiment and evaluate.

## Measurement, Estimation, and Data Analysis (Tactical Level - Medium Term)

- Obtain meaningful information for effective project management.

- Improve planning, understanding, and control.

## Measurement, Estimation, and Data Analysis (Strategic Level - Longer Term)

- Contribute to organisation-wide software process improvement.

* Diagnose issues
* Determine changes for improvement
* Monitor the effects of changes

## Use as a Management Tool to Answer Questions

- Assess current performance - how well are we doing?

- Investigate atypical features - can we make positive use of it?

- Identify consistent relationships - do x and y exhibit consistent relationships, why and how can it be useful?

- Compare different aspects - is one better than the other?

- Explain occurrences - why has it happened/ why does it keep happening?

- Predict outcomes of changes - if we change x will y be how we hoped for?

## What Is Measurement?

- Assigning a value to an entity's attribute on a particular scale.

- Requires a measurement instrument and a measurer.

## What Entities/Attributes to Measure?

- Product Attributes: Check specifications, designs, and test plans for clarity and quality.

- Process Attributes: Assess change control efficiency and project scheduling adherence.

- Resource Attributes: Evaluate personnel performance and budget utilisation.

## Units

- Measurement unit: a scalar quantity for comparing values of the same kind (e.g., inches, SLOC) e.g. length can be measured in inches, cm, etc.

Direct measurement = directly getting a number without guesswork or complicated steps.

Indirect measurement = finding a number through calculations or other information, not by directly measuring it.

## Scale Types

- Nominal, ordinal, interval, ratio, and absolute scales, in increasing order of richness.

### Nominal Scales

- Set of category identifiers, not a definer e.g you can identify a red and blue marble by their colours but the colour doesn't define its value, blue can be bigger than red but we don't know that by the nominal scale given (colour in this example)

- Waterfall process model

- Admissible relations: =, ≠.

- No meaningful operations.

### Ordinal Scales

- Linear ordering of values e.g. gold, silver and bronze represents 1st, 2nd and 3rd

- We can say what comes first and what comes after, but we can't measure the exact difference between them (e.g. in a race how far behind 2nd was from 1st)

- Admissible relations: =, ≠, <, ≤, >, ≥.

- Successor and predecessor operations.

### Interval Scales

- Linear order and difference between values,

- Help us measure things where we start at a certain point but doesn't mean "zero" as nothing

- Admissible relations: =, ≠, <, ≤, >, ≥, +, -.

- Successor, predecessor, addition, and subtraction operations.

### Ratio Scales

- Equal interval gaps with an absolute zero.

- Help us compare and share things,

- Additional admissible operations: =, ≠, <, ≤, >, ≥, +, -, \*, /

### Absolute Scales

- A type of ratio scale with only the identity transformation as admissible.

- Used for counting stuff e.g. counting apples in a basket

## Measurement Validity

- Ensuring a measurement is valid by checking if it is accurate, reliable, and meaningful (making sure you're measuring what you're supposed to measure)

## Measurement Accuracy

- = How close your measurement is to the real or true value

- Accuracy varies depending on the attribute.

## Estimation

- Predicting an expected value for an attribute with uncertainty expressed as probabilities e.g. not knowing the volume of a jar but guessing how many sweets inside

## Resource Prediction

- Estimating how many supplies are needed for a future project. It helps you plan and make sure you have enough of what you need.

- Used for determining project feasibility, bidding, planning, and tracking resource expenditure.

## Modelling Uncertainty

- Making decisions while knowing that there are things we don't know for sure e.g. having a backup option to a picnic incase the weather is bad

- Measurements are exact; estimates are uncertain.

- Probability = likelihood of an event happening (measured 0-1)

- Continuous variables = infinite number of possible values within a range. They can take on any real number, measured with precision, and have no gaps between values (e.g., age, weight, time).

- Discrete variables = specific, countable values with no fractions, like the number of students in a class or types of fruits in a basket.

## Confidence Intervals

- When you’re not certain so you give a ballpark estimate e.g someone looks 35-40 years old

- High confidence interval (measured in %) suggests it was predicted with a high level of certainty, doesnt tell if it'll take longer or less than interval

- Interval notation for estimates, expressed as (L, M, U), where M is the central value and L and U are lower and upper bounds. M is the expected value.

## Ways of improving estimations:

### The Delphi method:

- Gather opinions from a group of experts in multiple rounds to reach a consensus / make predictions on a problem. Diverse group, minimises bias

- Delphi Average= (min + (4likely) + max ) /6

### Estimation by Decomposition

- Decompose projects into smaller phases to estimate how long the whole project will take to complete.

- Top-down estimation = starts with an overall project estimate and breaks it into smaller parts, used when project completion time is fixed

- Bottom-up estimation = estimates each small part individually and combines them for the overall project estimate, used for an open-ended or research project

- Cost (£) = Effort (pm) x CpE (£/pm)

### Estimation by Analogy

- Guessing how long something will take based on how long it took to do something similar in the past e.g. walking to the park yesterday took 5 mins so today it should also take ~5mins

- Find projects with similar attributes and adjust based on differences.

- Q = comparable, past project

- P = new project

- Q.E = recorded effort value for the past project

- P.E = recorded effort value for the new project

- RF = ratio factors

- AF = addictive factors

1. Obtain Q.E
2. Identify varying factors between Q and P which could affect how much effort is required for P
3. Adjust Q.E to account for the factors identified

P.E (units pm)= Q.E (RF1 RF2…) + (AF1 + AF2…)

P.E(pm) = Q.E (P.S/Q.S) //this formula used when the only difference between the two projects is the size

P.E(pm) = (minE + maxE) /2 // min/max=lower/upper bounds for the recorded effort when 2 past projects are provided, multiply the +- ~

Estimates Steps:

1. Calculate two effort estimates separately from projects O1 and O2. Estimation by Analogy formula
2. Convert the separate effort estimates to financial costs. Multiply each estimate by unit effort cost rate (£/pm)
3. Resolve any differences between the two financial cost estimates to arrive at a single estimate. Work out average for L, M and U. Answer is M +-

## Units:

* **S size -** example units SLOC (but it could also be a number of modules)
* **EpS** (effort per size) - example of units used: pm/SLOC
* **SpE** (size per effort) called **Productivity** - example of units: SLOC/pm
* **CpS** unit cost - example of units used: £/SLOC
* **CpE** unit cost rate - example of units used: £/pm
* **Effort = S / EpS -** example working units: SLOC / pm/SLOC → pm
* **Effort = S/SpE** working the units: (SLOC) / (SLOC/pm) → pm
* **Cost = S x CpS** (example of units £ but it could also be € (EUROS) - example working the units: SLOC x £/SLOC → £
* **Cost = Effort x CpE -** example working the units: pm x £/pm → £, pm : person- month, but it could be person-week (for example)

# **16-23/10/23 Software Project Planning 1**

## Motivation

- Structured approaches or frameworks for software project planning cater to the struggle of how to begin, they:

- List the main steps involved in project planning.

- Suggest best practices for carrying out these steps.

-Indicate the sequence of steps, including iterations through sequences of steps.

## Project Objectives

- Project objectives encompass Quality, Scope, Time, and Cost.

- Key success factors include:

1. Agreeing on all project goals.

2. Developing clear plans with assigned responsibilities and accountabilities.

3. Managing the project scope effectively.

4. Cultivating constant effective communication.

5. Ensuring management support.

Frameworks: Step Wise & PRINCE2

## Step Wise Project Planning Framework

- Tries to answer the question "What do I do now?"

- Scalable, useful for both small and large projects.

- Applicable to a range of domains.

## Step Wise - an overview

1. Identify project objectives.

2. Identify project infrastructure.

3. Analyse project characteristics.

4. Identify products (things you make) and activities (things you do).

5. Estimate effort for each activity.

6. Identify activity risks.

7. Allocate resources.

8. Review and publicise the plan.

9. Execute the plan.

10. Lower-level planning.

Project Scenario: College Payroll

- The college aims to bring payroll processing in-house by acquiring an off-the-shelf application (blackboard).

- Example: A college currently uses a third-party service for payroll processing, which is expensive and lacks detailed data analysis capabilities.

- The project involves setting up an internal payroll office and developing software add-ons, such as one that combines payroll data with timetable data to calculate staff costs for each course.

- Example: You are hired as the project manager for this scenario.

### Step 1: Identify Project Objectives

1.1 Identify objectives and measures of effectiveness.

- Example: SMART objectives (Specific, Measurable, Achievable, Relevant, Time-constrained).

1.2 Establish a project authority.

1.3 Identify all stakeholders in the project and their interests.

1.4 Modify objectives based on stakeholder analysis.

1.5 Establish methods of communication with all parties.

### Step 2: Identify Project Infrastructure

2.1 Establish a link between the project and any strategic (long-term) plan.

2.2 Identify installation standards and procedures.

2.3 Identify project team organisation.

### Step 3: Analyse Project Characteristics

3.1 Distinguish the project as either objective or product-based.

3.2 Analyze other project characteristics (what’s different about this project).

3.3 Identify high-level project risks (what could go wrong, how can we stop it?).

3.4 Consider user requirements concerning implementation.

3.5 Select development methodology and life-cycle approach.

3.6 Review overall resource estimates.

### Step 4: Identify Products and Activities

4.1 Identify and describe project products.

4.2 Document generic product flows.

4.3 Recognize product instances.

4.4 Produce an ideal activity network.

4.5 Add stages and checkpoints/milestones.

### Step 5: Estimate Effort for Each Activity

5.1 Carry out bottom-up estimates.

5.2 Revise the plan to create controllable activities.

### Step 6: Identify Activity Risks

- To be studied in more detail later in the module.

### Step 7: Allocate Resources

- To be studied in more detail later in the module.

### Step 8: Review / Publicise Plan

- 8.1 Review quality aspects of the project plan.

- 8.2 Document the plan and obtain agreement.

### Steps 9 & 10

- Step 9: Execute the plan.

- Step 10: Lower-level planning.

- Detailed plans are required and feasible for the current activities.

PRINCE2

- **PR**ojects **IN** **C**ontrolled **E**nvironments: is a structured project management framework.

- It's based on seven core principles, including business justification and defined roles.

- PRINCE2 includes seven themes, such as quality and risk management.

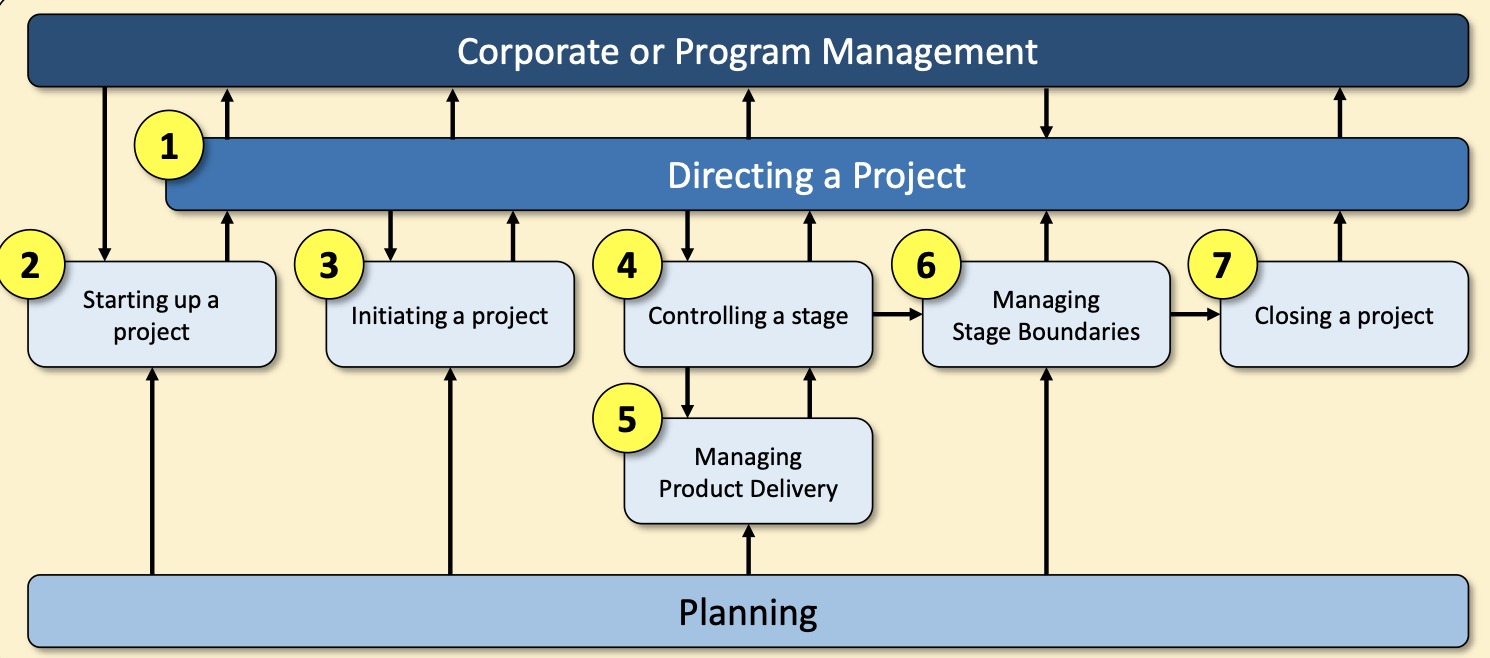
- The framework consists of processes for managing projects, like initiating and closing.

- Well-defined roles and responsibilities are a key aspect.

- PRINCE2 promotes documentation and product-based planning.

- Change control and business justification are central to the approach.

- The framework is highly adaptable and can be tailored to various project types.



## Prioritising Activities

- Prioritising is required when several activities are competing for the same limited resource at the same time.

- Resources are allocated to activities in decreasing priority order, considering total float priority and ordered list priority.

## Resource Allocation

- Identification of the resources required for the project.

- Making the demand for resources more even throughout the lifespan of the project.

- Production of a work plan and resource schedule.

- Key resources include labour, equipment, materials, space, services, time, and money.

- The project manager needs to identify the resources needed for each activity and create a resource requirement list.

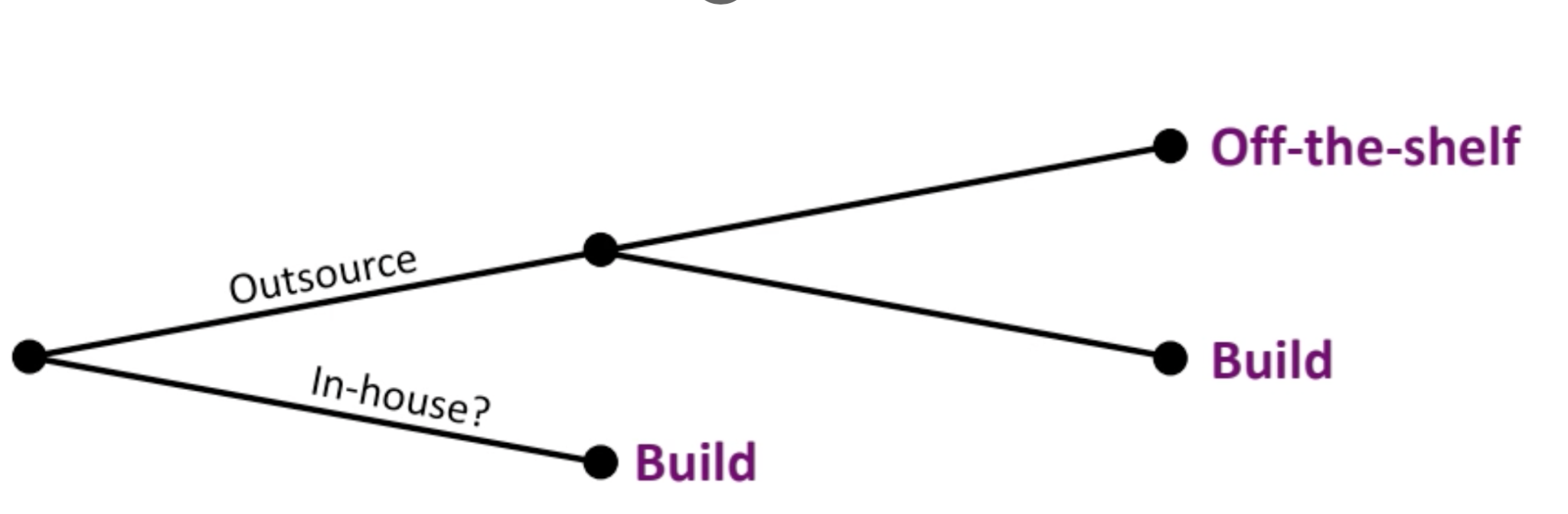
- Resource clashes can be resolved by various strategies, including adjusting start dates, delaying non-critical activities, and adding additional resources.

# **23/10/23 Software Project Planning 2**

## Building vs. Buying Software:

- Software can be built in-house when developers and users are part of the same organisation.

- Software can be outsourced, either purpose-built or off-the-shelf.



### Advantages of Off-the-Shelf Software:

- Cost-effective due to spreading development costs.

- Existing software can be trialled and has no development delays.

- Many defects are likely to be found and eradicated by existing users.

### Disadvantages of Off-the-Shelf Software:

- No competitive advantage since other customers use the same application.

- May require changes in the user's workflow.

- No ownership of the code and limited customization.

- Risk of over-reliance on a single supplier.

### Structured Approach vs. Agile Methods:

- Structured approaches emphasise step-by-step methods with defined steps and intermediate products (e.g., UML).

- Agile methods prioritise speed of delivery and reduced documentation (e.g., Scrum).

## Choice of Development Process Models:

- Various development process models, including the "waterfall," prototyping, incremental delivery, and agile methods like Scrum and extreme programming.

## The Waterfall Model:

- A sequential and predictive approach with clearly defined stages.

- Suitable when end-user requirements are stable and well-understood.

### Waterfall Advantages:

- Simple and easy to manage.

- Well-defined stages and milestones.

- Works well for smaller, well-understood projects.

### Waterfall Disadvantages:

- No working software until late in the project.

- High risk and uncertainty.

- Not suitable for complex, long, or evolving projects.

## Projects Suited for Agile Delivery:

- Agile is suitable for projects with aggressive deadlines, high complexity, and novelty.

## Extreme Programming:

- A type of agile development process with characteristics like short increments, customer feedback, and test-driven development.

- Limitations include reliance on high-quality developers and potential loss of knowledge over time.

## Prototype Terms:

- Prototypes can be throwaway (for unclear design) or evolutionary (for clear design).

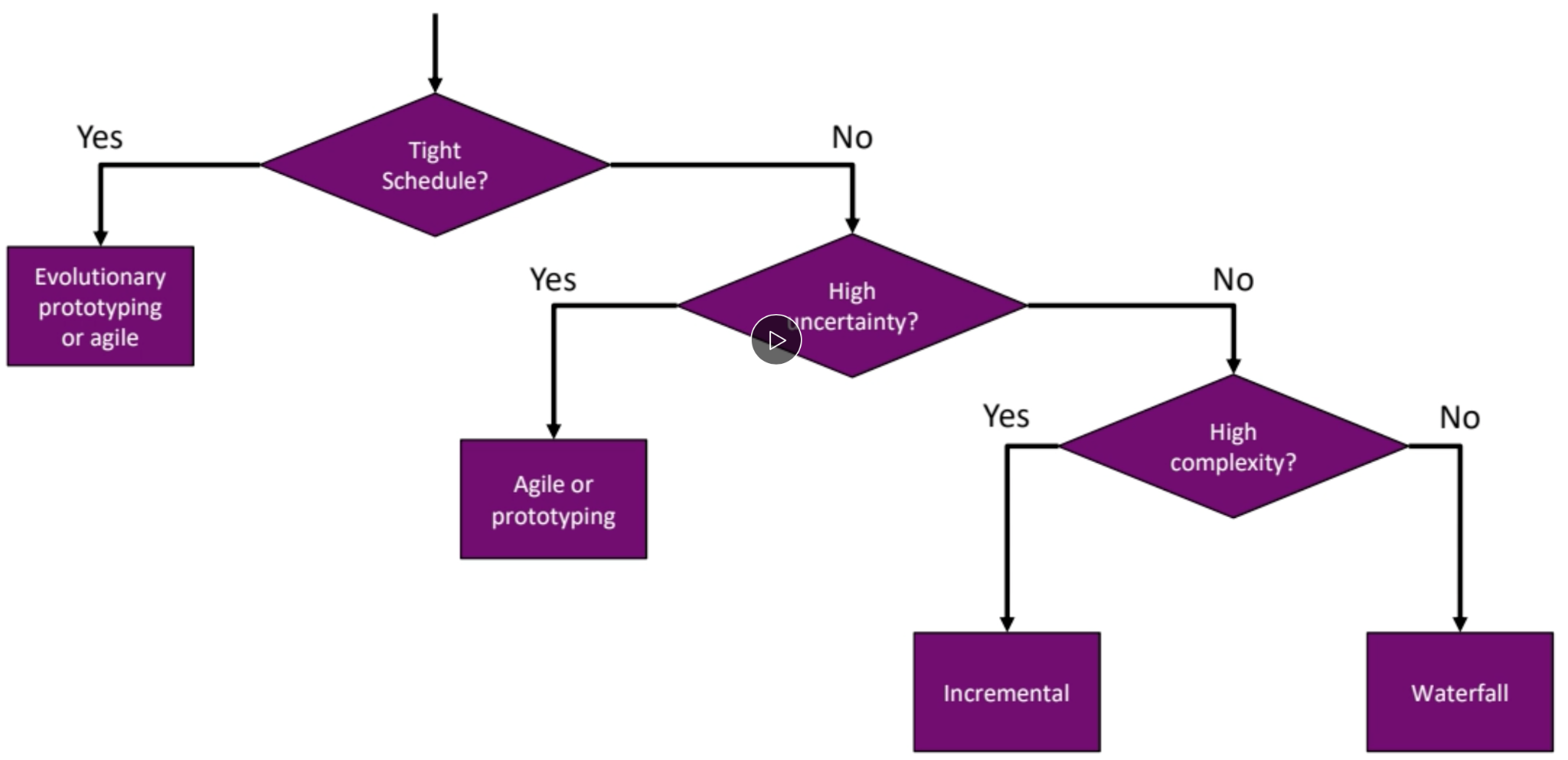
- Prototypes can aid learning, communication, user involvement, and reducing documentation.

## Incremental Delivery:

- The application is divided into components (lots) that are developed and delivered separately.

## Choosing a Model:

- The choice depends on factors like uncertainty, complexity, and schedule.



## Steps for Activity Planning and Resource Allocation:

- Select the project, identify objectives and infrastructure, analyse project characteristics.

- Identify products and activities, estimate effort, identify risks, allocate resources, and review the plan.

## Prioritising Activities

- Prioritising is required when several activities are competing for the same limited resource at the same time.

- Resources are allocated to activities in decreasing priority order, considering total float priority and ordered list priority.

## Resource Allocation

- Identification of the resources required for the project.

- Making the demand for resources more even throughout the lifespan of the project.

- Production of a work plan and resource schedule.

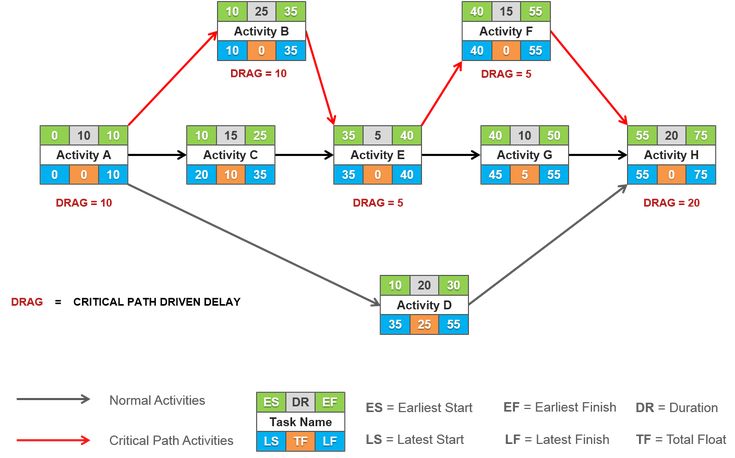
- Key resources include labour, equipment, materials, space, services, time, and money.

- The project manager needs to identify the resources needed for each activity and create a resource requirement list.

- Resource clashes can be resolved by various strategies, including adjusting start dates, delaying non-critical activities, and adding additional resources.

Critical Path Analysis = the sequence of tasks in a project that, if delayed, would result in the project taking longer to complete, as they have zero slack or float.

Example:



## Float is the amount of time a project can be delayed with it affecting the project's overall schedule.

## TOTAL FLOAT = LATEST FINISH - EARLIEST START - DURATION

Fill in: Green left to right, then Blue right to left, then work out Float

# **30/10/23 Project Risk Management**

## Categories of Risk

- Definition of Risk: The chance of exposure to the harmful consequences of future events.

- Negative Risk: Involves potential problems that might impede project success.

- Positive Risk: Can result in good things happening.

- Project Risk Management includes processes for identifying, analysing, and responding to project risk.

## Categories of Risk (Based on STAT: Structure, Technology, Actors, and Tasks)

- *Structure* covers management procedures and communication (e.g. lack of communication).

- *Technology* involves development tools, techniques, and technology embedded in deliverables (e.g. compatibility)

- *Actors* include everyone involved in the project, e.g., developers, users, managers. (e.g low morale / staff leaving etc)

- *Tasks* represent the work to be carried out in the project (e.g. the effort required was underestimated).

## A Risk Management Framework

1. Risk identification: Identify potential risks (e.g. cost/quality and completion date)

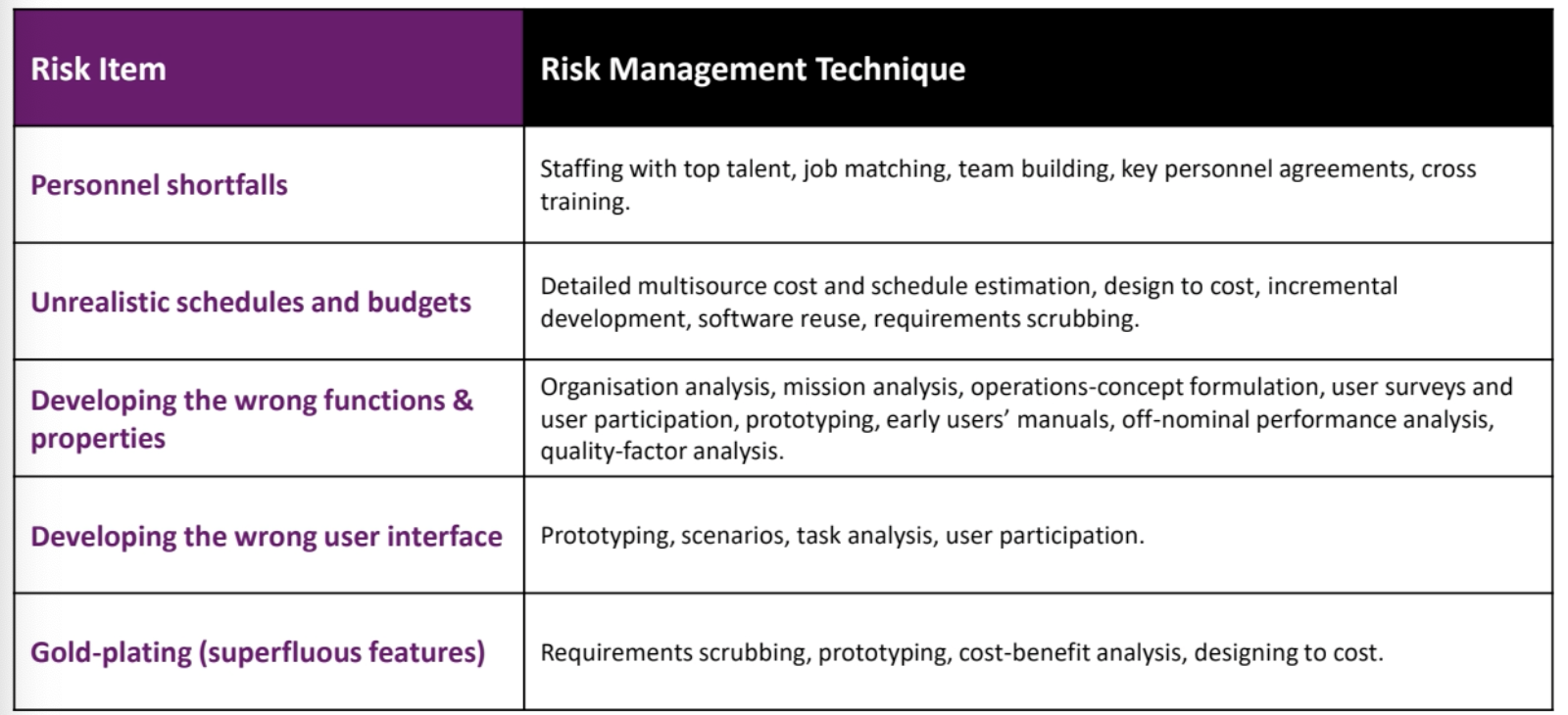
2. Risk analysis and prioritisation: Assess risk likelihood and impact.

- Quantitative risk analysis

- Qualitative risk analysis

3. Risk planning: Determine how to address identified risks.

4. Risk monitoring: Continuously monitor the state of the risks.



## Quantitative Risk Analysis

- Objective: To provide a numerical estimate of the overall risk on project objectives.

- Used for: large and complex projects, projects requiring contingency reserves, and projects with significant go/no-go decisions.

- Techniques:

- Three-Point Estimate: Involves estimating the best-case, worst-case, and most likely outcomes and calculating expected values.

- Expected Monetary Value (EMV) or Risk Exposure: Assigns monetary values to risks and their probabilities to calculate expected cost impact.

- Program Evaluation and Review Technique (PERT): Uses statistical analysis to determine project durations.

- Key Indicators:

- Probability: Ranges from 0 (no chance) to 1 (certain).

- Cost: Measured in terms of money, time, or quality metrics.

- Purpose: To assess the financial impact of identified risks and to make informed decisions based on quantitative data.

Risk Exposure (RE)= probability x cost

## Qualitative Risk Analysis

- Objective: To prioritise risks based on a predefined rating scale considering both probability and impact.

- Use for: Quick assessment of risks, doesn't require specialised tools or software.

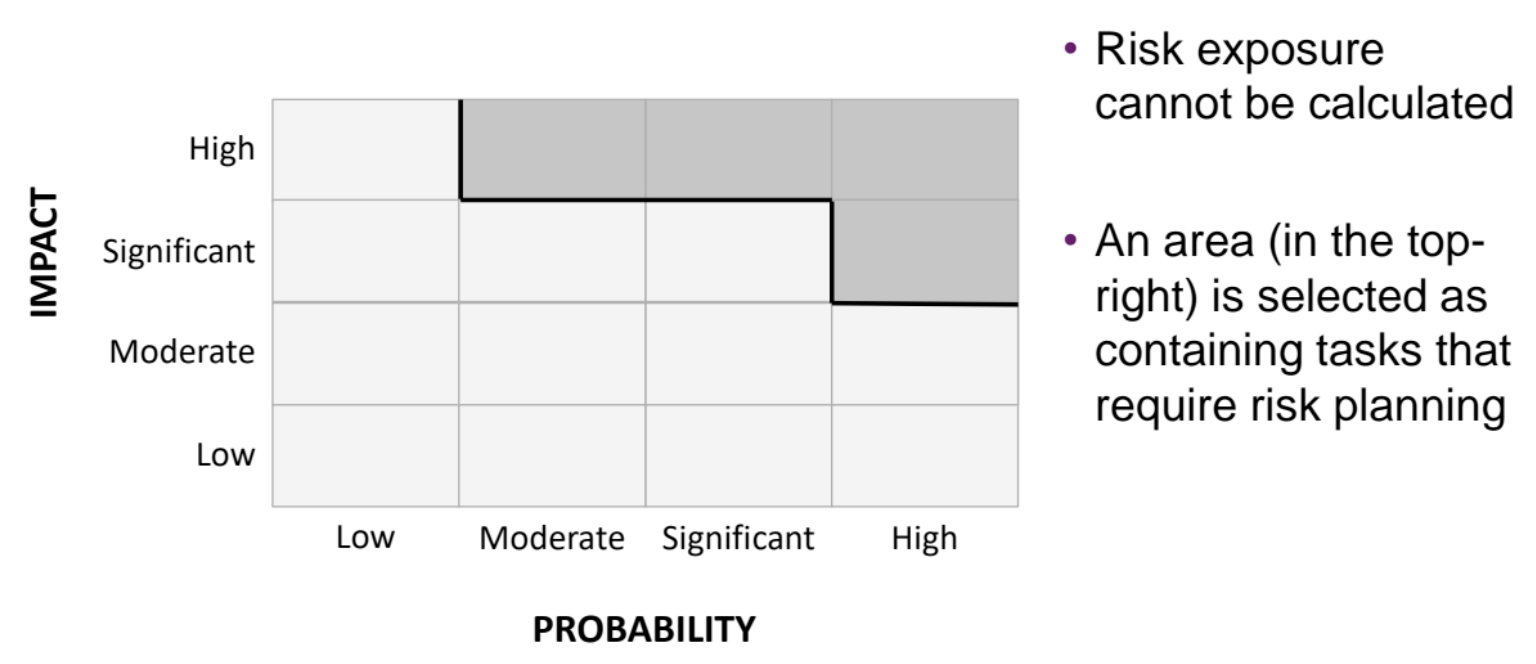
- Process:

- Evaluates probability and impact using a rating scale (rank in respects to other risks).

- Quick and straightforward assessment.

- Purpose: To rank risks according to their potential impact, allowing for easy identification of high-priority risks.

- Probability-Impact Matrix:



- Severity and Likelihood: Factors considered in the assessment.

- Quick and Subjective: Useful for a first-pass risk assessment and prioritisation.

## How to Deal with Risks

1. Risk Acceptance:

- When to Use: When the cost of avoiding the risk is estimated to be greater than the potential damage.

- Example: Accepting a minor risk rather than investing in extensive risk mitigation measures.

2. Risk Avoidance:

- When to Use: To completely avoid the causes associated with the risk.

- Example: Opting for an off-the-shelf software application instead of developing one to avoid the risk of development uncertainties.

3. Risk Reduction:

- When to Use: When you want to take actions to reduce the likelihood of the risk occurring.

- Example: Prototyping to reduce the risk of incorrect requirements or offering bonuses to retain staff and reduce the risk of key members leaving.

4. Risk Transfer:

- When to Use: Transferring the risk to another party or organisation.

- Example: Negotiating a fixed-price contract with an external software supplier to transfer the risk of development estimates to them.

5. Risk Mitigation/Contingency Measures:

- When to Use: To reduce the impact if the risk does occur.

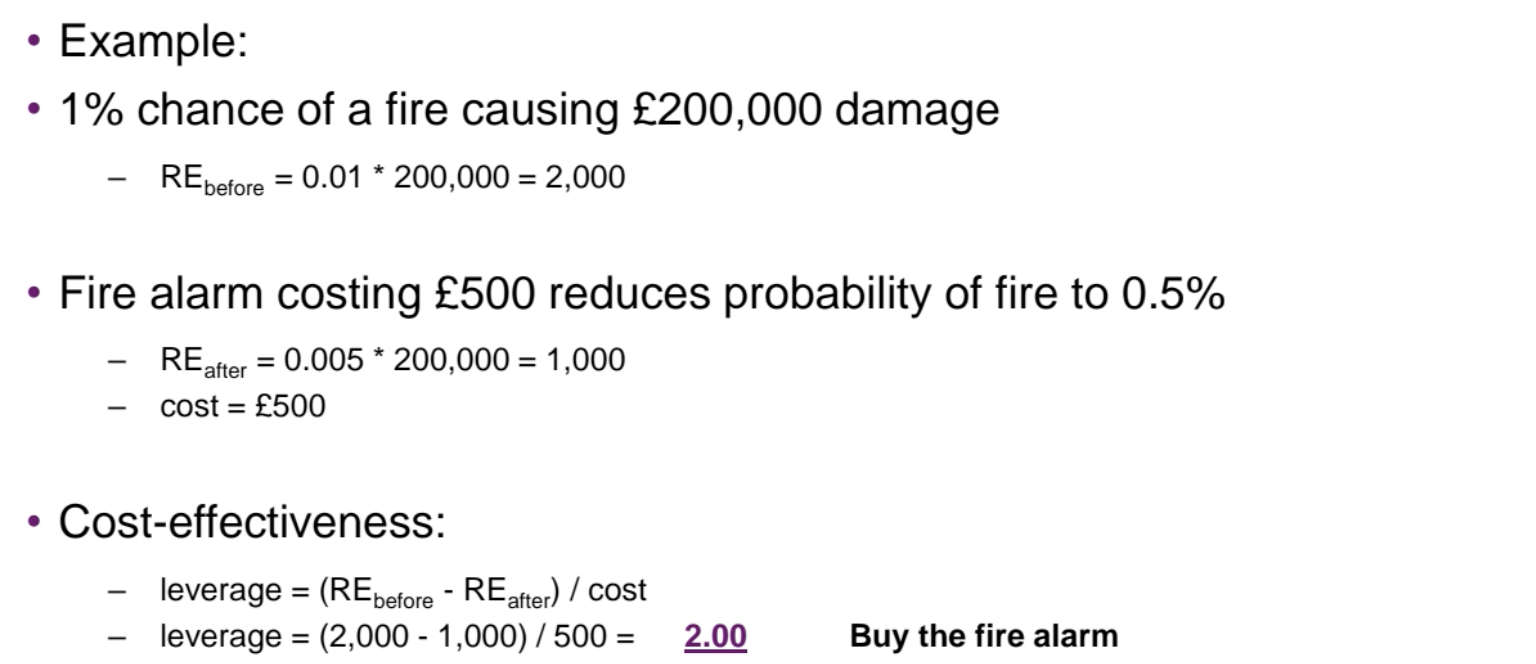
- Example: Regularly monitoring the progress of project activities to identify causes of risk and taking actions such as data backups for rapid recovery in case of data corruption.

6. Cost-Effectiveness of Risk Actions:

- Leverage (probability)- how much have we reduced the risk (should be **>** 1.00, bc problem is bigger than the amount we’re going to spend on it)

Leverage = (REbefore - REafter) / cost

REbefore = risk exposure before we do anything (RE= probability x cost)

REafter = risk exposure after intervention 

## Overall Goal of Risk Management

- Project Risk Management: Focuses on identifying, analysing, and responding to project risks.

- Aims at:

- Maximising the results of positive events (opportunities).

- Minimising the consequences of adverse events (threats).

- Considerations:

- Project size.

- Complexity.

- Speed of implementation.

## PERT (Program Evaluation and Review Technique)

- Used to determine expected time for project activities based on optimistic, pessimistic, and expected durations.

- Helps calculate the probability of project overrun.

- Requires understanding of the critical path analysis.

- Calculates expected duration and variance for each task.

Expected Duration = (min + (4likely) + max ) /6) (use Dephi average)

Variance = ((max-min)/6)^2

Variance is the measure of how much the values vary (small variance means low number)

Z-score = (target date - total expected duration) / square root of total variance

If negative: target date is sooner

If 0: on track

If positive: target date is after

Probability of meeting the Target:

## Normal Distribution in PERT

- Assumes a normal distribution of potential durations.

- The likeliest duration is the middle point, and longer or shorter durations are equally likely within a certain range.

# **13/11/23 Project Monitoring and Control**

## Project Control Cycle

- Define Objectives: Set project goals.

- Collect Data: Gather information relevant to project progress.

- Process Data: Analyse collected data for decision-making.

- Model: Create models to represent project aspects.

- Implement: Execute plans and decisions.

- Make Decisions/Plans: Formulate decisions and plans based on data.

- Check: Regularly check progress against objectives.

- Act: Take actions to address issues and keep the project on track.

## Alternative Cycle- Scrum

Plan > Do > Check > Act > repeat

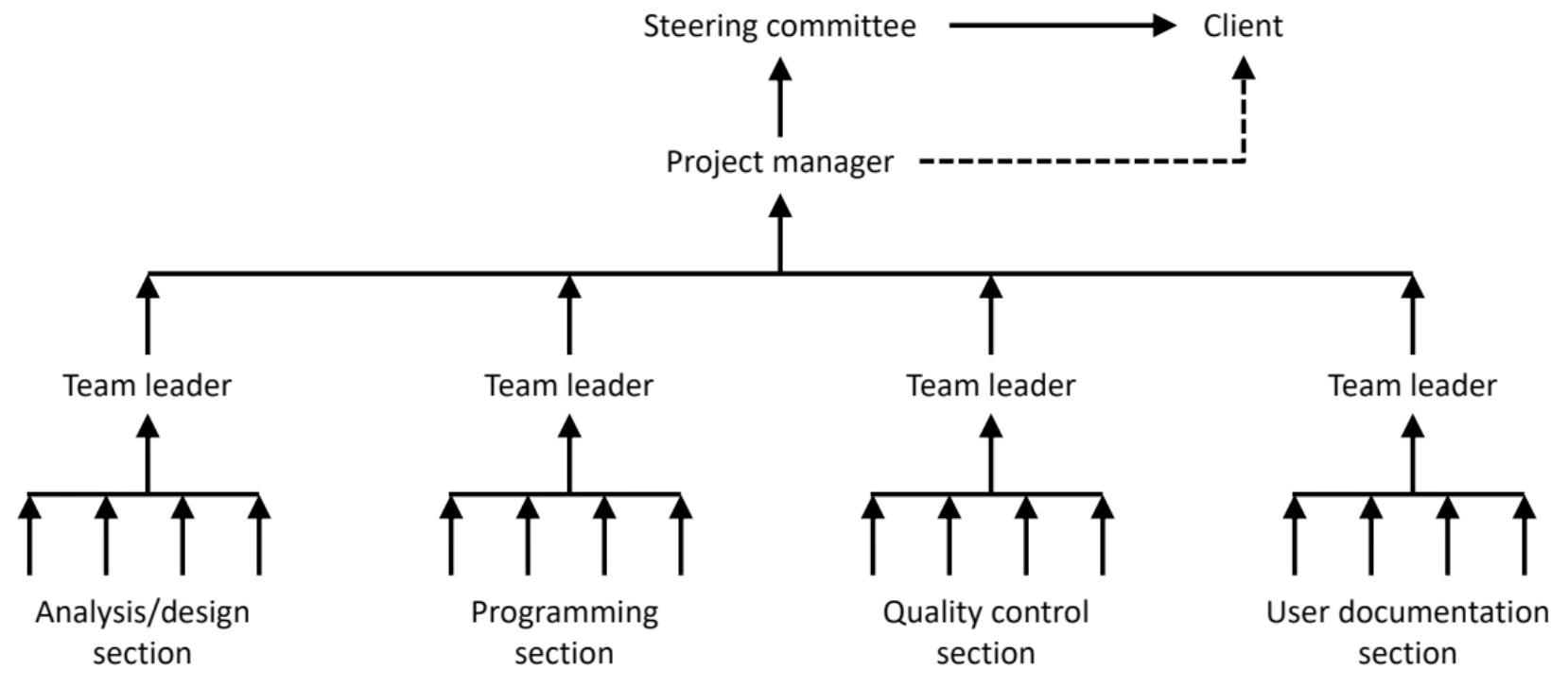
## Responsibilities

- Project Board: Ensures overall project progress, allocated resources

- Project Manager: Day-to-day responsibility for project progress.

- Team Leaders: Responsibilities can be delegated by the Project Manager.

- Example: Reporting hierarchy with a steering committee, project manager, and team leaders.



## Alternative View- Scrum

Product Owner- sets the overall priorities of the product

Scrum Master- responsible for leading team in scrum processes

Development Team- eg developers, ux designers and researchers

## Assessing Progress

- Checkpoints: Event-driven or time-driven evaluations of project progress.

Event-driven e.g. end of project stages in PRINCE2

Time-driven e.g. between sprints in SCRUM

- Example: FYP checkpoints, definition form, and first report submission.

## Collecting Progress Details

- Challenges: Dealing with partial completions and the 99% completion syndrome.

- Example: Subdividing tasks into objective-verified sub-activities to control products, not just activities.

## Red/Amber/Green (RAG) Reporting

- Traffic-light Method categorises tasks into:

- Red: Not on target and recoverable with difficulty.

- Amber: Not on target but recoverable.

- Green: On target.

- Example: Applying RAG reporting to coding and testing module C.

### Cons

- making estimates based on imprecise data

- discrete classifications ergo difficult

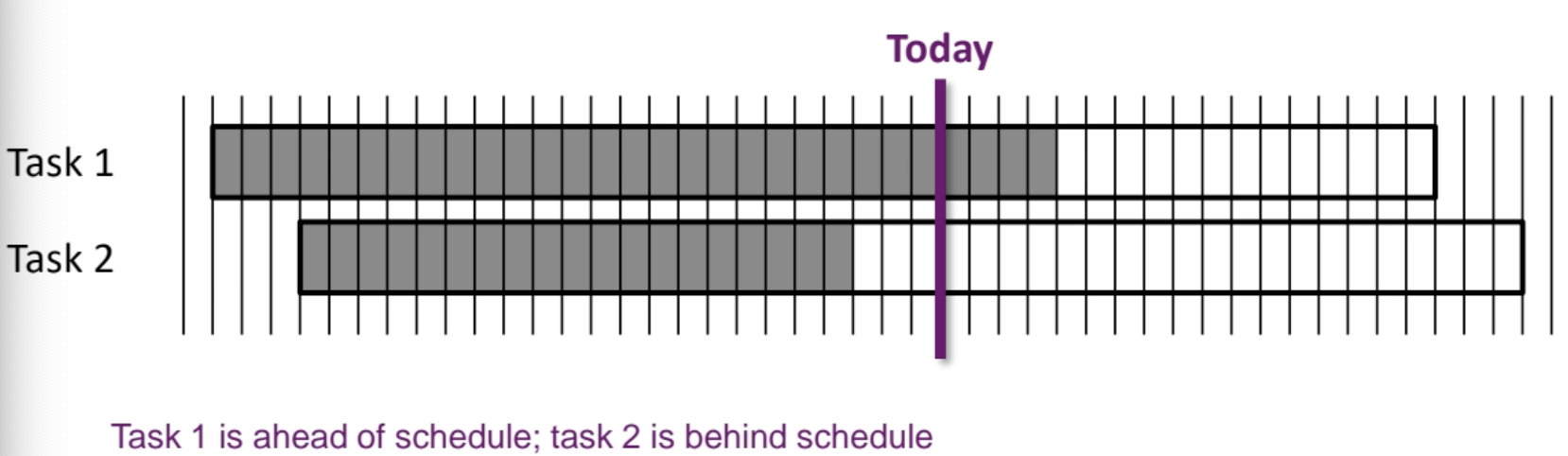
## Alternative to RAG: SCRUM

Product backlog (everything product will do) > to do > doing > done

## Using Gantt Charts

- Track Project Progress:

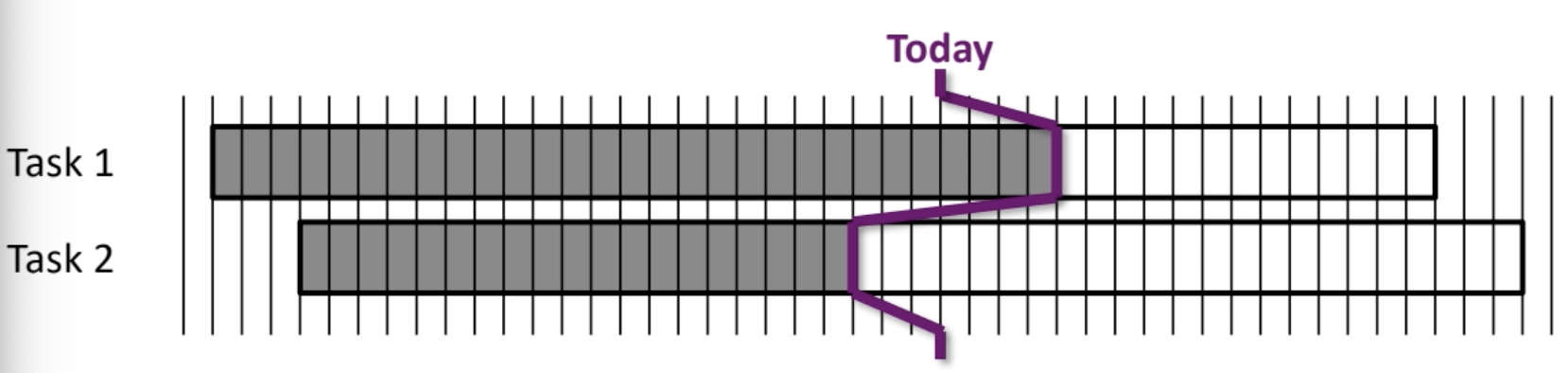
- Shading bars to reflect reported progress.

- Adding a ‘today’ cursor for expected progress.

## Using Slip Charts

## - Bent 'Today' Cursor:

- Indicates activity positions with more jagged lines for inconsistent progress.



### Limitations of Gantt/ Slip Charts

- Do Not Clearly Show:

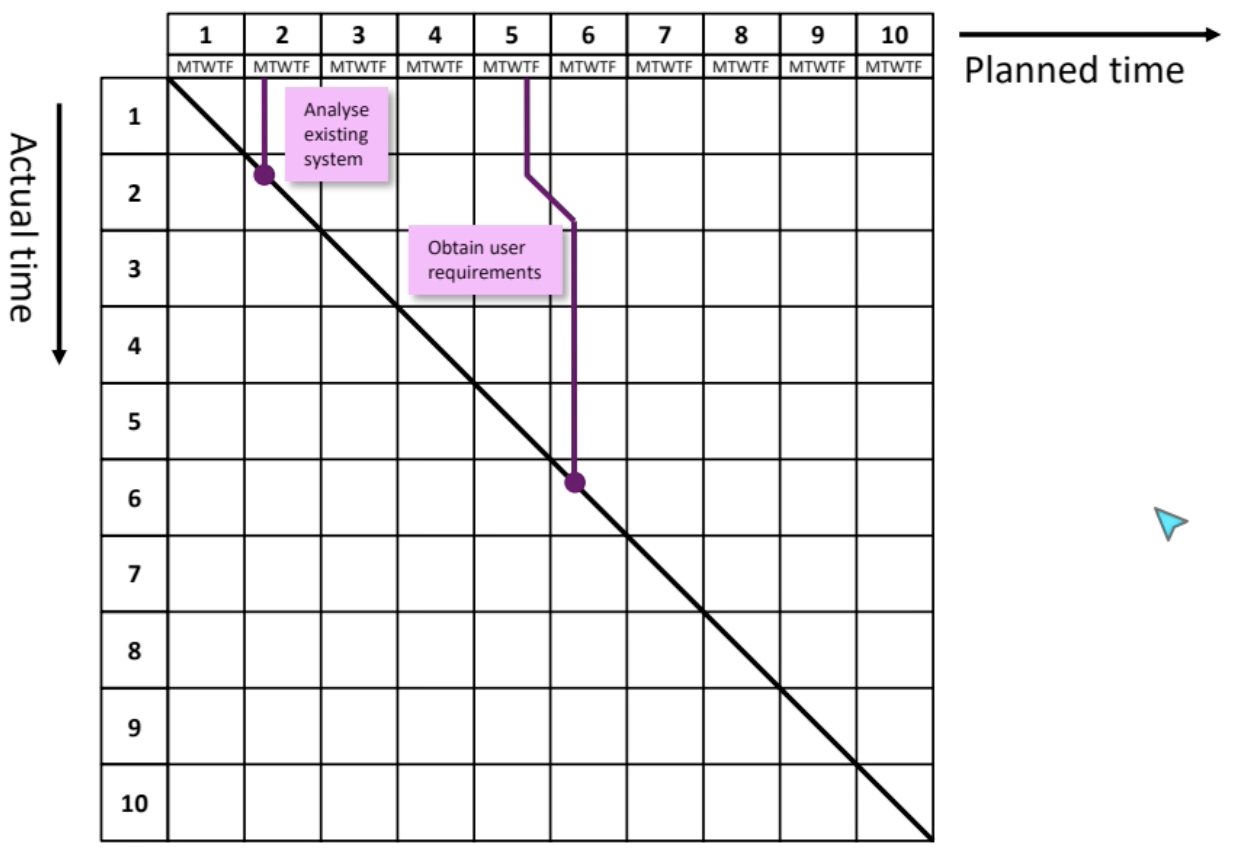
1. Slippage of project completion date over time.

2. Analysis of project trends and productivity (are we consistently going to be late or is it a one off, consistently late affects end date, with a one off we can reallocate resources).

- To Improve: Need a timeline representation to overcome limitations.

## Timeline Representation

## - Depicts planned and actual time against project activities.



## Cost Monitoring

- Earned Value Approach:

- Assigns a value to each task based on original cost estimates.

- Calculates earned value based on the task's state.

## Partly-Completed Tasks

- Techniques like 0/100, 50/50, milestones, and percentages are used.

- 0/100: earned value = 0 until complete, 100 once complete

- 50/50: half of the plan value allocated at the start, the 2nd half on completion

- milestone: each milestone requires a completed product e.g. when design stage completed we should have 10 artefacts

- percentage: estimate of what's been complete

## Exception Planning

## - Project manager can change the plan with an exception report.

- Example: Writing an exception report to justify deviations from the plan.

## Prioritising Monitoring

- Different levels of detail/effort for monitoring based on criticality.

- Example: Focusing more on critical path activities, those with no free float, high-risk activities, etc.

## Getting Back on Track: Options

1. Renegotiate the Deadline: Discussing a new project completion date with stakeholders.

2. Shorten Critical Path: Identifying critical path activities that can be expedited.

3. Reconsider Activity Dependencies: Overlapping activities to reduce dependencies but considering potential impacts on quality.

# **20/11/23 Quality Management**

## Software Quality in the ‘Step Wise’ Framework:

- Steps 1, 2, 3, 4, and 8 are quality-related.

- Importance of software is increasing, and products include third-party components.

Product Quality Management vs. Process Quality Management:

## Product Quality Management (Quality Control):

- Focus: Making sure the final software meets expected quality.

- Goal: Identify and measure needed quality features.

- Example: Checking if the software functions correctly.

- Challenge: Can be hard and expensive.

## Process Quality Management (Quality Assurance):

- Focus: Ensuring the quality of steps used to build the software.

- Goal: Guaranteeing that the process creates a high-quality product.

- Example: Making sure coding and testing processes are effective.

- Challenge: Also difficult and costly.

## Classification of Quality Characteristics:

External vs. Internal Characteristics:

- External: How users see the software (e.g., usability).

- Internal: How developers see the software (e.g., code structure).

## Direct vs. Indirect Measurement:

- Direct: Measuring exactly what's required (e.g., response time).

- Indirect: Using related measurements (e.g., user inquiries) to assess quality.

## Defining a Quality Management Plan (Quality Planning):

What's Included:

- Objectives, quality processes, specified characteristics, metrics, roles, and responsibilities.

- Goal: Make sure everyone understands what needs to be done for quality.

- Example: Outlining how to measure and achieve quality in the software project.

## Defining a Quality Characteristic for a Software Product:

Quality Specification:

- Includes definition, scale, test, minimally acceptable value, and target range.

- Aims: Clearly describe and measure specific quality features.

- Example: Using user testing to check if tasks in the software are completed well.

## Measurement of a Quality Characteristic:

Multiple Measurements:

- Various ways to measure a quality characteristic.

- Examples: Checking availability, failure rate, and mean time between failures for reliability.

## Software Quality and the ISO 9126 Standard:

Introduction:

- ISO 9126, created in 1991, sets standards for software quality.

- Versions cater to those buying software, those making it, and those evaluating it.

### Functionality:

Key Aspects:

- Suitability, accuracy, interoperability, functionality compliance, and security.

- Ensures software does what it should, does it correctly, and follows standards.

### Reliability:

Key Aspects:

- Maturity, fault tolerance, recoverability, reliability compliance.

- Looks at how often software fails, its ability to recover, and adherence to standards.

### Usability:

User Experience:

- Understandability, learnability, operability, attractiveness, usability compliance.

- Concentrates on user-friendly aspects and sticking to standards.

### Efficiency:

Performance Measures:

- Time behaviour, resource utilisation, efficiency compliance.

- Addresses how fast the software responds, its resource usage, and adherence to standards.

### Maintainability:

Ease of Management:

- Analyzability, changeability, stability, testability, maintainability compliance.

- Deals with spotting failures, effort for changes, impact of changes, testing effort, and adherence to standards.

### Portability:

Adaptability and Compatibility:

- Adaptability, installability, coexistence, replaceability, portability compliance.

- Focuses on adaptability, installation ease, sharing resources, replacement ease, and adherence to standards.

## Key Points

- Quality is vital and growing in importance in software.

- Quality cannot be an afterthought but must be quantitatively specified.

- Product quality control is complemented by quality assurance, but it does not guarantee good quality software.

# **27/11- 04/12 Agile Methodologies**

## Agile Basics:

- Agile vs. Waterfall:

- Waterfall: Sequential, rigid, all planned upfront.

- Agile: Embrace change, collaboration, value-driven.

## Waterfall Model:

### - Advantages:

- Simple structure.

- Easy management- each phase has specific deliverables

- Well-documented process (useful for fyp report)- completed one at a time

### - Disadvantages:

- Late software production- produces at the end of lifecycle.

- High risk for changes.

- Not suitable for complex projects where requirements may change.

- Difficult to measure progress

- Makes integration at the end difficult (compatibility issues)

## What is Agile Development?

- Focus:

- Embrace change during the project.

- Prioritise client-developer collaboration.

- Value-driven processes.

### - Projects Suited for Agile:

- Aggressive deadlines.

- High complexity and novelty.

### Agile Benefits:

- Better alignment with business objectives and IT.

- Gains in visibility, adaptability, business value (biggest value, smallest cost) and risk reduction.

- Problems arise faster, and can be solved sooner.

- Control over change costs.

- Longer-lasting software.

### Agile Drawbacks:

- Requires active user involvement.

- Evolving requirements.

- Frequent deliveries.

- System structure degradation.

- Experienced team needed.

### Agile Approaches:

- Extreme Programming (XP): Improves software quality.

- Scrum: Framework for complex problems.

- Kanban: Efficient work management.

### Agile Manifesto Values:

- Individuals and interactions over processes and tools.

- Working software over comprehensive documentation.

- Customer collaboration over contract negotiation.

- Responding to change over following a plan.

### Continuous Improvement:

- Agility provides a framework.

- Adapt practices based on product, team, and environment.

- No agility without quality.

### Sustainable Pace:

- Steady and sustainable work pace.

- Promotes motivation and reduces risk.

### System of Prioritization:

- Focus on visibility.

- High value, low complexity tasks first (report fyp)

- Low value, high complexity tasks last (deliverable fyp).

## Scrum Framework:

- Lightweight, adaptive, and incremental.

- Values: commitment, focus, openness, respect, courage.

### Three Pillars of Scrum:

- Transparency, Inspection, Adaptation.

### Scrum Artefacts:

1. Product Backlog: Features and priorities.

2. Sprint Backlog: Tasks for the current sprint.

3. Product Increment: Output of each sprint.

### Roles in Scrum:

1. Product Owner: Decides features, priorities.

2. Scrum Master: Enforces Scrum values, removes impediments.

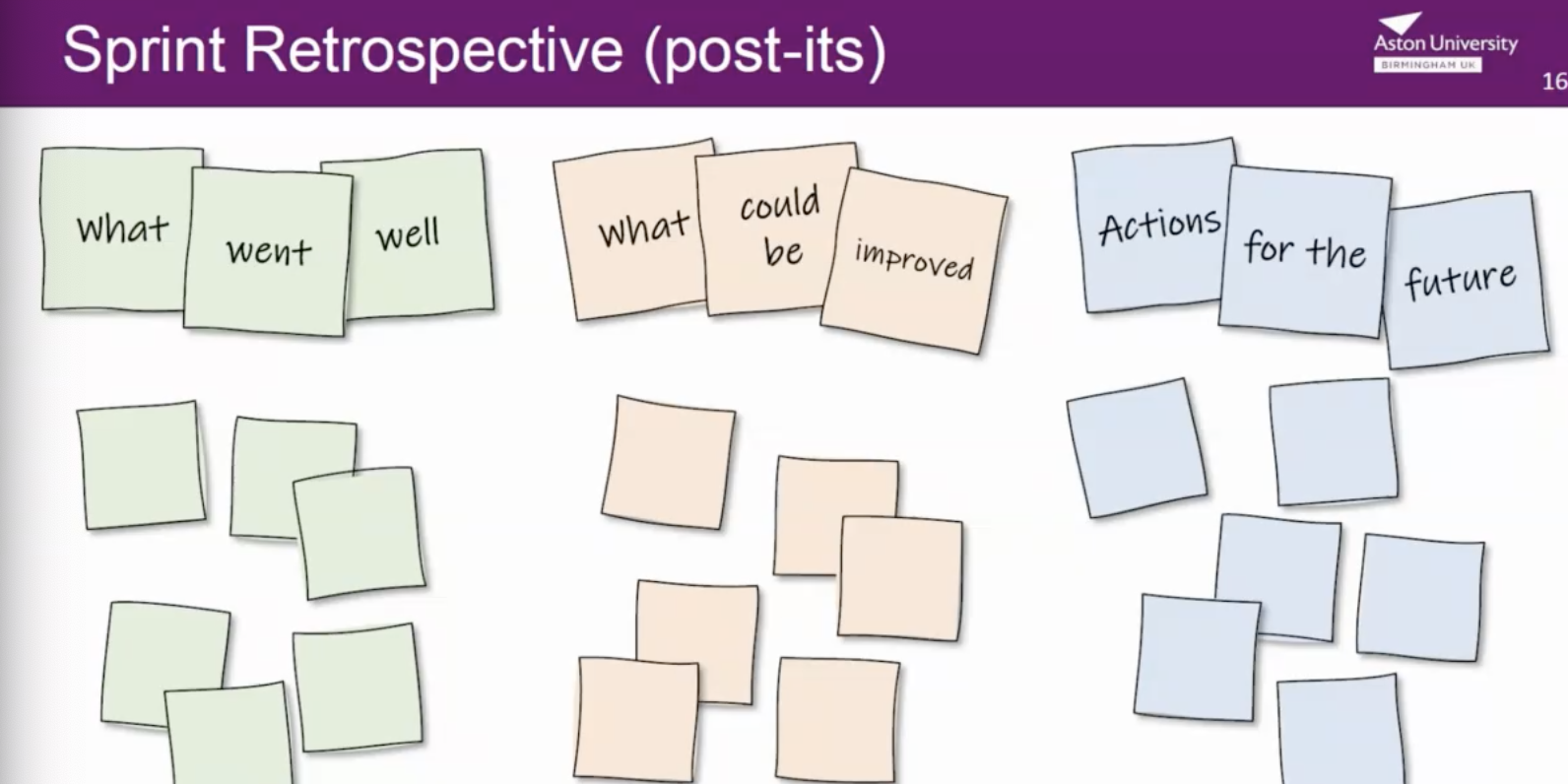
3. Project Team: Self-organising, cross-functional.

### Scrum Events:

1. Sprint Planning: Set goals for the sprint.

2. Daily Stand-Up: Quick progress updates.

3. Sprint Review: Showcase accomplishments (demo).

4. Sprint Retrospective: Reflect and improve, managed by SCRUM master.

## Product Owner:

* Accountable for return on investment (ROI)
* Manages Product Backlog
* Prioritises functionalities
* Clearly defining business needs at different levels of detail.
* Communicates about progress made

## Scrum Master:

* Manages the teams
* Ensures team members collaborate
* Ensures product owner provides necessary information and requirements to the team.

### Scrum Advantages:

* Good for projects where requirements are hard to quantify
* Sprints/ retros makes changes easier to cope with
* Daily stand ups makes it easier to measure individual productivity
* Lower overhead costs, ergo quicker and cheaper results

### Scrum Disadvantages:

* Leads to scope creep- stakeholders will keep demanding new functionality
* Only good for small, fast moving projects
* Requires experiences team members- if they leave mid project, it will have impact the project development

## Kanban Steps:

1. Specify what to implement from backlog

2. Implement it (only pull work when you have capacity)

3. Validate that it works

4. Deliver to customers/ partners

### Advantages of Kanban:

- Easy to use and understand.

- Promotes continuous improvement.

- Highly adaptable.

- Focuses on collaboration.

- Low overheads.

### Disadvantages of Kanban:

- Lack of iteration.

- No defined timeframes.

- Dependency on the board.

## Scrum vs. Kanban:

