**Factors that affect Effectiveness and Efficiency**

- Cost control
- Quality assurance
- Scope management
- Risk management
- Stakeholder management
- Resource management
- Time-to-market

**Intangible benefits of software projects**

- Improved customer satisfaction
    - Evaluate by surveys
- Increased employee morals
    - Evaluate by surveys
- Enhanced organisational reputation
    - Evaluated by brand awareness surveys

**9 Belbin team roles**

- Plant
    - Creative problem solver
- Monitor evaluator
    - Impartial judge
- Coordinator
    - Dominant positive thinker who focuses on objectives
- Resource investigator
    - Networker
- Implementer/company worker
    - The practical strategist who does jobs others will not
- Completer finisher
    - The team's quality assurer
- Team worker
    - Keeps the team running smoothly
- Shaper
    - Keeps the team focused
- Specialist
    - In-depth knowledge of a key area

**Software Project Managers is more difficult**

- Output is intangible
    - Difficult to measure process or assess quality
    - May require more documentation and testing to meet standards
    - Difficult to estimate costs and timeline accurately
- Processes are less mature
    - Make it difficult to compare different progress or assess their performance

        - Processes and methodologies can quickly become outdated and obsolete
        - Can create challenges in terms of training and education with qualified personnel
- Projects are often novel or innovative
    - Project manager must be creative and adaptable in their approach
    - Project requirements are not well-defined, creating challenges to scope
    - Creating challenges to risk management as it may not be well-understood

**Top-down Methodology**

- Example = waterfall, spiral, RUP, Incremental
- Weakness:
    - Difficulty in obtaining complete, correct and appropriate specifications of what the user actually needs
    - Creating a unique software solution for every project requires a lot of time and effort in development and testing and can also be expensive

**Effective in SEPM**

- Ability to achieve the desired project goals and objectives

**Efficient in SEPM**

- Ability to use resources effectively to achieve desired project goals and objectives

**Exception-based Reports**

- Focuses on reporting only significant exceptions or issues that require attention
- Designed to highlight significant deviations from the project plan or expected outcomes
- Useful in situations where a large amount of data needs to be monitored regularly but most of the data falls within acceptable tolerance levels

**Refactoring**

- Elimination of duplicated codes
- Improvement of structure
- Structuring of data
- Removal of redundant code

**Considering UML and issues**

- What is the purpose of the model?
    - Ensures that the UML is appropriate for the intended purpose
- Who is the intended audience for the UML?
    - Be able to know what level of detail and complexity is appropriate
- What specific modelling techniques and notations will be used?
    - Need to consider the needs of the project and use appropriate model
- What are the potential limitations and drawbacks of using UML?
    - Can be complex and difficult to understand

**Standards**

- A set of guidelines, rules or criteria that defines how software projects should be managed, developed and maintained
    - Provide common framework
- Advantages:
    - Ensure readability and completeness
    - Ensure accuracy and readability
    - Ensure consistency
    - Ensure maintainability
    - Ensure decisions are taken at correct times by the correct person
    - Ensure decision is known
    - Ensure changes are known and correct version is in use

**Functionality, Usability, Reliability, Performance, Supportability (FURPS)**

- By considering these, developers and stakeholders can better understand and prioritise the requirements for the system for its intended purpose

- **F**unctionality - Capability (Size & Generality of Feature Set), Reusability (Compatibility, Interoperability, Portability), Security (Safety & Exploitability)
- **U**sability (UX) - Human Factors, Aesthetics, Consistency, Documentation, Responsiveness
- **R**eliability - Availability (Failure Frequency (Robustness/Durability/Resilience), Failure Extent & Time-Length (Recoverability/Survivability)), Predictability (Stability), Accuracy (Frequency/Severity of Error)
- **P**erformance - Speed, Efficiency, Resource Consumption (power, ram, cache, etc.), Throughput, Capacity, Scalability
- **S**upportability (Serviceability, Maintainability, Sustainability, Repair Speed) - Testability, Flexibility (Modifiability, Configurability, Adaptability, Extensibility, Modularity), Installability, Localizability

- Framework to categorise and prioritise the requirements of a software project
- Functionality
    - Features and capabilities of a system
- Usability
    - How easy the system is to use for the intended userbase
- Reliability
    - System’s ability to perform its functions correctly and consistently
- Performance
    - System’s speed, efficiency and resource usage
- Supportability
    - System’s maintainability, scalability and compatibility with other systems

**Scheduling**

- Translation of a process model into a network of activities
- 4 factors that determine how network is constructed
    - The way the project decomposes into tasks (work breakdown structure, GANTT)
    - The availability of the resources needed to execute the tasks
    - The estimated time completion for each task
    - The interdependencies that dictate pre-requisites for starting a task
- 3 factors that affect the duration of the project
    - Reworking caused by design/software errors
    - Reworking cause by changes in customer requirements
    - Delays caused by risks that materialises as issues

**Checkpoint Reports**

- Covers 3 reporting periods
    - Actions outstanding actions from the previous period
    - Progress made in the current period
    - Results planned for the next period
- Confirms that work package is “within tolerance”
    - On the track to meet its planned objectives, within allowed time and cost constraints
- May be required at a specified frequency (weekly,monthly) or on milestone dates
- Fixed frequency reporting should be long enough to avoid burdening staff but short short enough to keep the report concise and relevant

**Highlight Reports**

- Provides high level summary of project progress to stakeholder of senior management
- Includes current status of each work package and product/artefact
- Should include corrective actions taken and projections for the next reporting period
- Should provide summary of the tolerance situation across the project as a whole
- Should itemise the status of any changes requests and any changes in risk status

**Process model**

- General and incomplete depiction of how a project is controlled and evolves
- Establishes interfaces between actors and workflows and transitions between activities
- The choice of model does not change the project’s requirements or need for code testing before and after integration
- Customer requirements remain the reference point for design regardless of process model chosen

**Artefact**

- Output from the project = product
    - Allows another higher level product to be constructed
- Example = software product
- Should be based on:
    - Project scope definition
    - Cost and schedule definition
    - Risk identification
    - Project feasibility
    - Plans for the project (development) environment

**Work Breakdown Structure**

- Structure that decomposes the project into phases -> activities -> tasks -> sub-tasks
- Example = GANTT chart, product flow diagram

**Managing team performance**

- Identify problem
    - Review team's work and compare
- Communicate the issue
    - Explain standards, feedback and discuss how to improve
- Develop a plan
    - Set specific goals
- Monitor progress
    - Provide ongoing support
- Take action if necessary
    - Reassign member's role, provide additional support

**Transition phase**

- Period of time between completion of construction phase and deployment of the software into production

**Construction phase**

- Software design is put into action and software is developed according to the specifications and requirements outlined in planning and design phase

**Major activities in transition and construction phase**

- Development
- Integration
- User Acceptance Test (UAT)
- Documentation
- Deployment

**Project Administration**

- Provides administrative support to project manager
    - Maintaining documentation, scheduling meetings, coordinating project logistics
- Does not have decision-making authority
    - Provides information and data
- Not a leadership role

**Project Manager**

- Responsible for overall management of the project
    - Planning, execution, control, communication with stakeholders and team members
        - Motivating and resolving team conflicts
        - Making sure of informed decisions and project progress
        - Determine project scope, develop project plan, allocating resources

**Why software projects fail**

- Poor requirements gathering, scope not defined and understood

        - Leads to miscommunication, misunderstandings, failure to meet the needs of end-users
- Inadequate planning and project management
    - Leads to missed deadlines, cost overruns, poor software quality
- Changing requirements
    - Leads to delays, increased costs, failure to meet the needs of end-users
- Leak of stakeholder involvement
    - Failure to understand their needs and requirements
    - Limited opportunities for feedback and validation
- Inadequate testing
    - Leads to defects, errors, poor software quality
- Limited resources such as time, money, personnel
    - Leads to delays, cost overruns, failure to meet the needs of end-users
- Poor communication among team members, stakeholders, project managers
    - Leads to misunderstandings, delays, failure to meet the needs of end-users

**Addressing why projects failed**

- Ensuring that requirements are clearly defined and understood by all stakeholders
- Developing a comprehensive project plan that includes timelines, budget and milestones
- Managing changes to requirements effectively to minimize the impact on the project
- Identifying technical challenges early and developing strategies to address them effectively
- Implementing comprehensive testing strategies to ensure that the software is thoroughly tested and meets the needs of the end-users
- Allocating sufficient resources to the project and managing them effectively to ensure that the project is completed on time and within budget
- Developing effective communication strategies and ensuring that all team members, stakeholders and project managers are informed and updated regularly on the project's progress

**4 components in RUP**

- Workflows
    - Sequence of activities performed by roles which produces artefacts
- Roles
    - Defines responsibilities and skills required for team members
- Artefacts
    - Tangible outputs of the development process
        - E.g. document, codes
- Activities
    - Tasks performed by roles, contributing to the production of artefacts

**Process Modelling's Importance**

- Essential for project success
    - Ensures systematic approach, clarity and structure

- Improved communication
    - Clear understanding of tasks, roles and responsibilities
- Risk management
    - Identifies potential issues early, supports mitigation strategies
- Quality assurance
    - Enables verification, validation and testing throughout development

**Advantages of incremental delivery of software**

- Faster feedback loop
    - Early identification of issues -> Quicker course of correction and improvements
- Early value delivery
    - Clients can use it faster -> benefits it early as compared to waterfall model, spiral model
- Improved risk management
    - Reduces the scope of potential problems -> Easier to identify problems
- Increased customer satisfaction
    - Demonstrates progress to them and see if it meets their needs
- Easier integration and testing
    - Encourages modular design and development -> Easy to facilitate each component
- Better resource allocation
    - Enables better prioritisation and management of development tools
- Enhanced team motivation
    - Visible progress boosts team morale and motivation
        - Encourages collaboration and shared responsibility for project success
- Adaptability to change
    - Supports flexibility and responsiveness to change
        - Easier to accommodate shifting priorities or new requirements

**Disadvantages of incremental delivery of software**

- Increased management overhead
    - More frequent planning, tracking and coordination efforts
- Potential for scope creep, increases project duration or costs
    - Adding additional features or functions of a new product, requirements or work
- Incomplete functionality
    - Clients may still need to wait for critical features to be delivered
- Frequent releases
    - May disrupt client operations, version management problems
- Inefficient use of resources
    - Can lead to delays in other aspects of project
- Increased client involvement
    - Can be time consuming and distract workflow
- Dependency challenges
- Difficulty in estimating project duration

- Due to frequent feedback and requirement changes

**Software Requirement Specifications**

- Introduction
  - Purposes
    - Describes the objective and scope
  - Audience
    - Defines the intended readers (developers, client, stakeholder)
  - Clarifies terminology for clear communication
- Project overview
  - System context
    - Describes how the software fits within the larger system or business environment
  - User roles and responsibilities
    - Outlines user categories and their needs
  - Assumptions and dependencies
    - Lists any external factors or conditions impacting the project
- External features
  - Interface requirements
    - Details user, hardware and software interfaces
  - Data requirements
    - Describes data inputs, outputs and storage needs
  - Compliance
    - Specifies any relevant industry standards or regulations the software must meet
- Supplementary requirements
  - Non-functional requirements
    - Addresses aspects such as reliability, usability and security
  - Quality attributes
    - Defines expected system quality levels, maintainability, scalability
  - Constraints and limitations
    - Outlines any technical or business constraints affecting the project

**Delphi Method disadvantages**

- Time consuming
  - Delays in obtaining and processing export feedback
- Expert availability
  - Difficulty in finding and obtaining suitable experts
- Subjectivity and bias
  - Estimation may be influenced by personal opinions and experiences
- Inaccurate or outdated expertise
  - May not be applicable to specific project
- Lack of transparency
- Limited scope

        - Potential for overreliance on expert input
- Inefficiency in reaching consensus
    - Result in delays and additional resources
- Difficulty in documenting rationale
    - Challenging to provide detailed explanations for estimations

**Delphi Method advantages**

- Anonymity
    - Encourages honest and unbiased input
- Aggregated expertise
    - Uses collective knowledge of experts, increasing accuracy
- Iterative process
    - Allows for identification and resolution of discrepancies
- Controlled feedback
    - Facilitator summarises and shares feedbacks, minimising conflicts
- Adaptability
    - Flexible approach for diverse industries and domains
- Reduced groupthink
    - Encourages diverse opinions and perspectives
- Fosters learning
    - Encourages continuous improvement and knowledge sharing

**Verification**

- Ensures the product is being built according to specified requirements
- Focus
    - Evaluates the process, checks conformity with design documents and specifications
- Example
    - Inspecting code to ensure it follows coding standards and design patterns

**Validation**

- Ensures the built product meets the user's needs and expectations
- Focus
    - Evaluates the product functionality, assess if it solves the intended problem
- Example
    - User acceptance testing
        - Confirms that the software meets user requirements and expectations

**Inception Phase Completed tasks**

- Project scope definition
    - Establishing project boundaries, objectives and high-level requirements
- Stakeholder identification
    - Identifying and documenting key stakeholders and their roles
- Initial risk assessment

        - Identifying risks, evaluating their impact and outlining mitigation strategies
    - Project feasibility analysis
        - Assessing technical, financial and organisational feasibility for the project
    - High-level project schedule
        - Estimating project duration and outlining major milestones
    - Resource planning
        - Identifying necessary personnel, equipment and other resources
    - Project approval
        - Obtaining stakeholder buy-in and approval to proceed to the next phase

**Software reuse**

- Leveraging existing software components, code or designs in new project
    - Reduces development effort, improves consistency, promotes standardisation
- Benefits:
    - Reduced development time
        - Existing components can be integrated faster
    - Lower development costs
    - Improves quality
        - Reduced potential for defects/bugs in new projects
    - Consistency and standardisation
        - Simplifies maintenance and support efforts
    - Easier knowledge transfer
        - Reduces learning curve and onboarding time
    - Increased reliability
        - Reused codes have gone through rigorous testing, enhancing reliability
    - Enhanced maintainability
        - Reused codes are well documented, facilitates easier updates and maintenance

**Measuring successful project reviews**

- Establish review objectives
    - Defines clear goals and align objectives with project quality standard and requirements
- Schedule regular reviews
    - Ensures timely identification and resolution of issues
- Involve relevant stakeholders
    - Encourages diverse perspectives and expertise
- Prepare review materials
    - Compiling of necessary documents, code samples and other artefacts for review
- Conduct structured review sessions
    - Address objectives systematically to encourage constructive feedback and open discussion
- Implement and track improvements
    - Assign responsibilities and deadlines for addressing

- Measuring success
    - Timely issue resolution
        - Monitor progress on action items and measure improvements
    - Quality metrics
        - Track defect rates, customer satisfaction and other relevant indicators
    - Lessons learnt
        - Evaluate review effectiveness and adapt processes for continuous improvement

**Issues that might arise when improving success**

- Resistance to change
    - Team members may resist altering established routines
    - Requires effective change management and communication strategies
- Training and learning curve
    - Team members may need to learn new tools or methods
    - Could impact productivity in the short term
- Resource availability
    - Implementing new processes may require additional resources
    - Could strain the project budget or schedule
- Integration with existing processes
    - Changes may disrupt other connected processes
    - Requires careful planning and coordination
- Effect on project deliverables
    - Process changes could impact project output, timelines or quality
    - Requires careful risk assessment and management
- Organisational culture
    - Company culture may resist change or prefer established ways of working
    - Requires leadership support and cultural adaptability
- Measuring effectiveness
    - Challenging to measure the impact of process changed accurately
    - Requires clear metrics and tracking mechanisms
- Documentation and standardisation
    - New processes must be documented and standardised
    - Requires time and effort to ensure consistency
- Stakeholder buy-in
    - Changes may need approval from stakeholders or upper management
    - Requires effective communication and persuasion skills
- Legacy systems and compatibility
    - New processes may not be compatible with existing systems or technology
    - Requires careful evaluation and potential technology upgrades

**Factors that impact on the maintainability of software**

- Code quality
    - Well-structured, clean and efficient code is easier to maintain

    - Encourage best practices, code reviews and adherence to coding standards
- Documentation
    - Comprehensive and up-to-date documentation simplified maintenance tasks
    - Includes design documents, user manuals and inline code comments
- Modularity and separation of concerns
    - Modular design with clear boundaries between components enhances maintainability
    - Simplifies updates and reduces the risk of introducing new issues
- Consistency and standardisation
    - Consistent coding conventions, design patterns and naming conventions facilitate maintenance
    - Promotes easier understanding and modification of code
- Testability
    - Well-tested software with thorough test coverage is more maintainable
    - Encourage unit testing, integration testing and automated testing
- Technical debt management
    - Proactively addressing technical debt prevents long term maintainability issues
    - Regularly assess and address code quality, design issues and outdated dependencies
- Dependency management
    - Proper management of third-party libraries and dependencies affects maintainability
    - Stay up-to-date with security patches and compatible versions
- Knowledge transfer and team expertise
    - Effective knowledge sharing and training within the team ensures maintainability
    - Cross -training and onboarding processes help maintain expertise across the team

**Risk Register**

- Document used in project management to identify, assess and track potential risks
- Key components
    - Risk descriptions, probability of occurrence, potential impact, risk owner and mitigation strategies
- Purpose
    - Helps in proactive risk management, facilitating timely risk identification, assessment and response planning
    - Aids in communication and understanding of risks among project stakeholders