Comparison of Software Development Models

- SER 416 – Assignment 1
- Waterfall, Spiral, RUP, Agile
- Bhupinder Singh
- Course: SER 416
- October 16th, 2025

Model 1: Waterfall

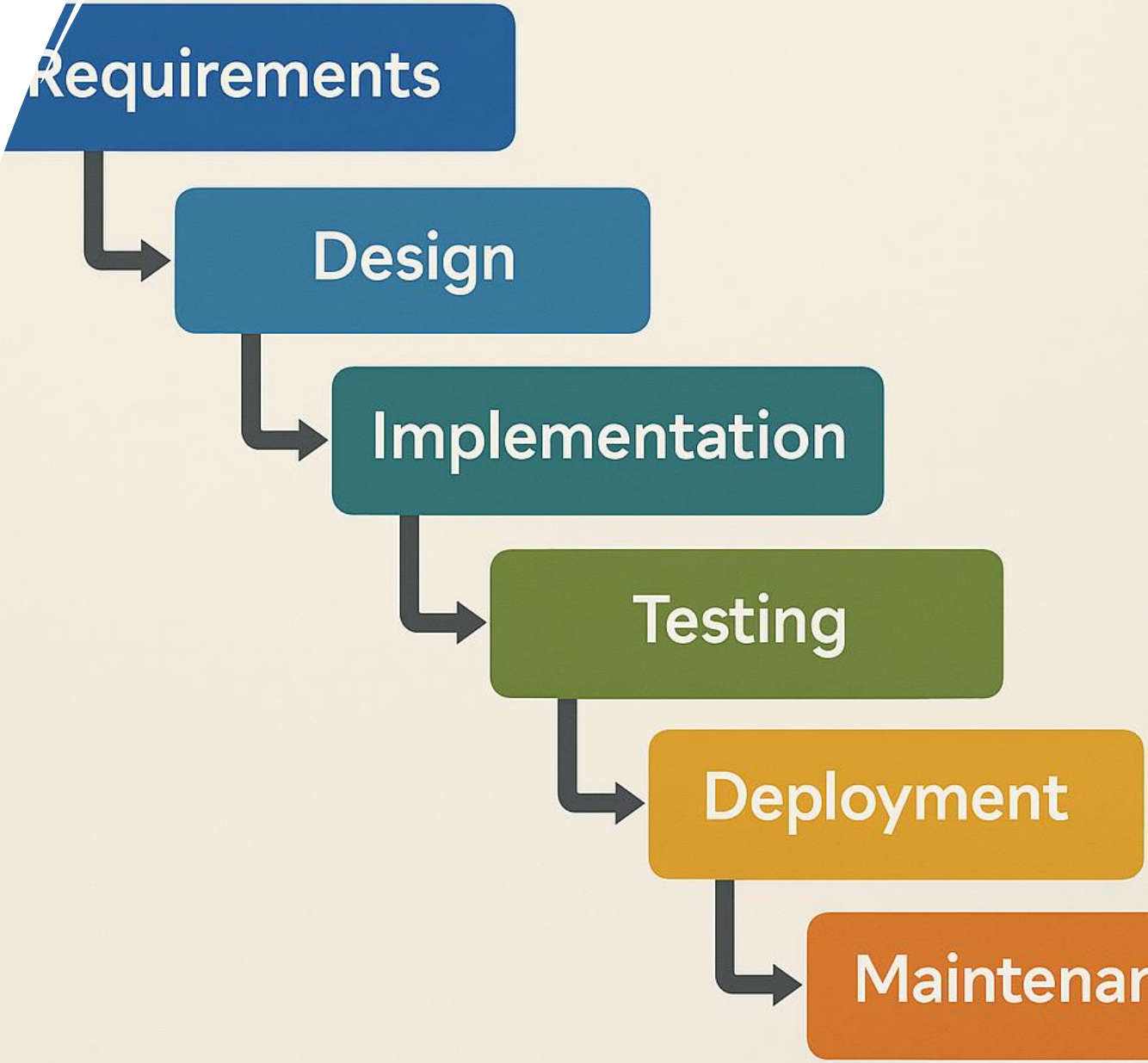
Waterfall – Key Characteristics

- Linear and sequential approach

- Divided into clear phases: Requirements, Design, Implementation, Testing, Deployment, Maintenance

- Each phase must be completed before the next begins

- Little customer involvement once requirements are finalized



Waterfall- Strengths

- Simple and easy to manage

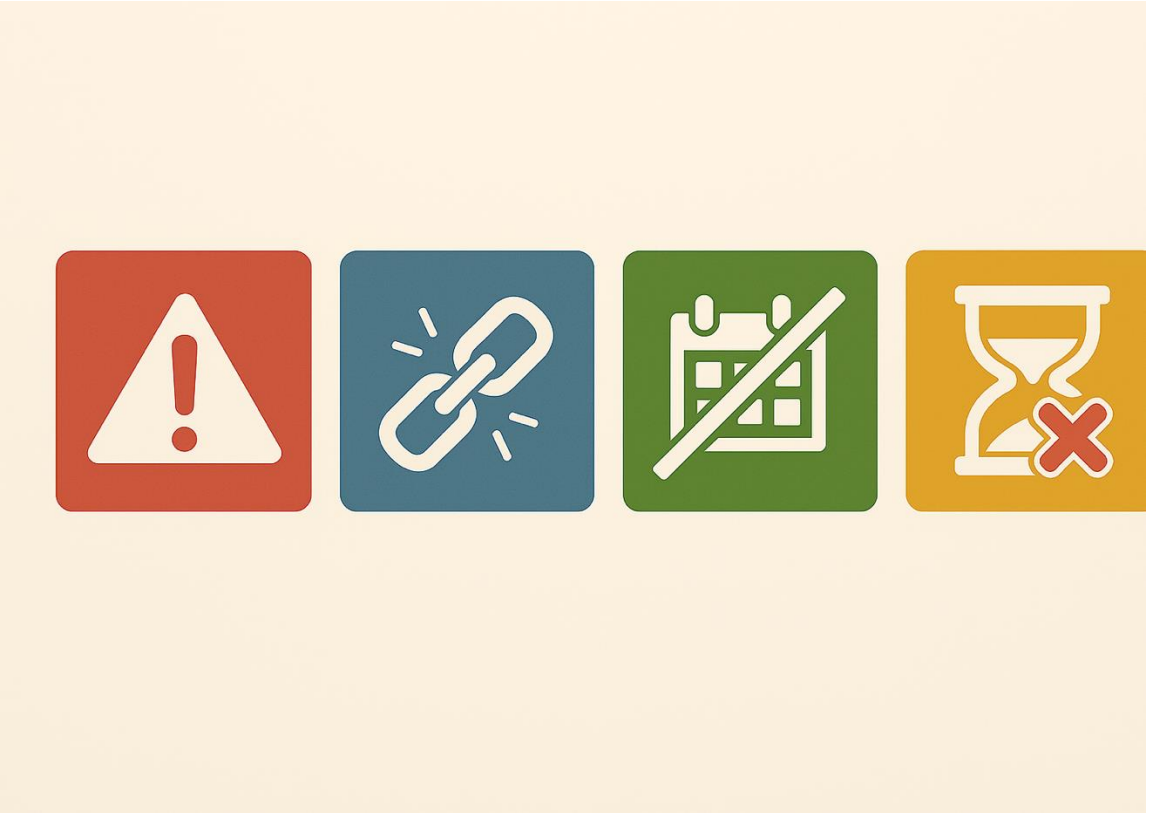
- Clear structure and milestones

- Well-suited for small or well-understood projects

- Thorough documentation ensures clarity

Waterfall – Disadvantages

- Inflexible to changing requirements
- Poor adaptability to changes after the project starts
- High risk of project failure if errors exist in early phases
- No working software until late in the lifecycle



Waterfall – Best Use Case

- Projects with well-defined, stable requirements
- Short-term projects with predictable outcomes
- Government or contract-based projects
- Projects where documentation is more critical than flexibility

Model 2: Spiral

Spiral Model – Key Characteristics

- The Spiral Model combines elements of both design and prototyping in stages, making it especially useful for large, complex, and high-risk projects.
- It emphasizes risk analysis and iterative development.

Spiral Model – Strengths

- Strong focus on risk assessment and mitigation

- Ideal for large and high-risk projects

- Iterative nature allows refinement through multiple cycles

- Stakeholders are involved throughout the process

Spiral Model – Disadvantages

- Can be expensive and complex to manage
- Requires expertise in risk analysis
- Not suitable for small or low-risk projects
- Can lead to extended timelines if not managed properly

Spiral Model – Best Use Case

- The Spiral Model is best used for:

- Large, complex software projects with evolving requirements

- Projects requiring continuous risk assessment

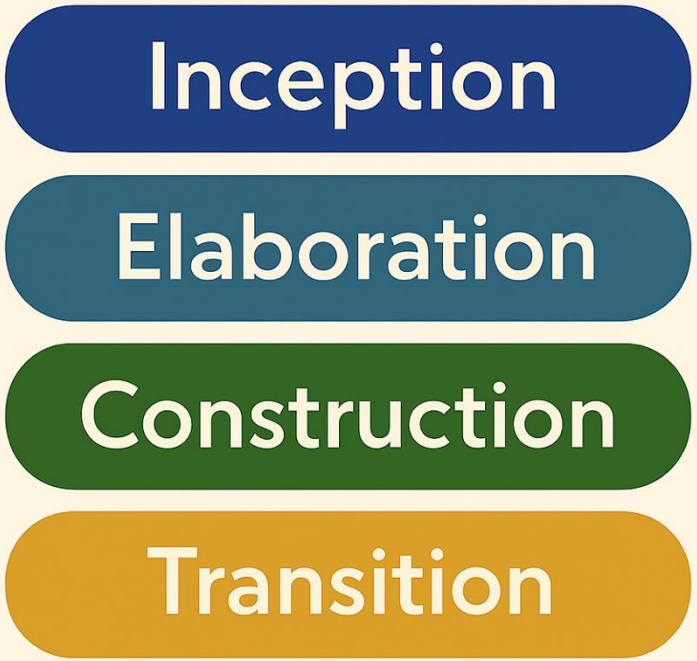
- Projects involving new technologies or unfamiliar domains

- Example: Development of a medical software system with strict regulatory and safety requirements.

Model 3: RUP

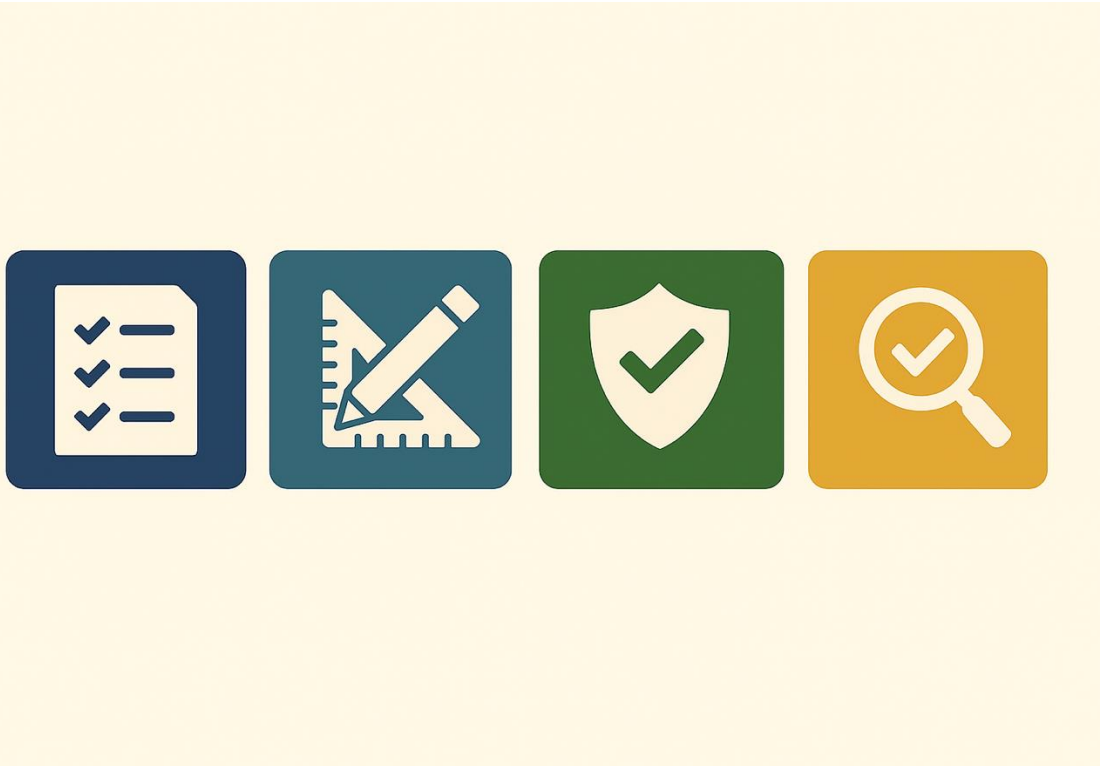
RUP (Rational Unified Process) – Key Characteristics

- Iterative and incremental software development framework
- Divided into four structured phases:
- Inception, Elaboration, Construction, Transition
- Emphasizes disciplined project management and requirements traceability
- Focuses on continuous testing, risk mitigation, and component-based architecture
- Supports role-based workflows and clear deliverables at each phase
- Developed by Rational Software (IBM) to promote software engineering best practices



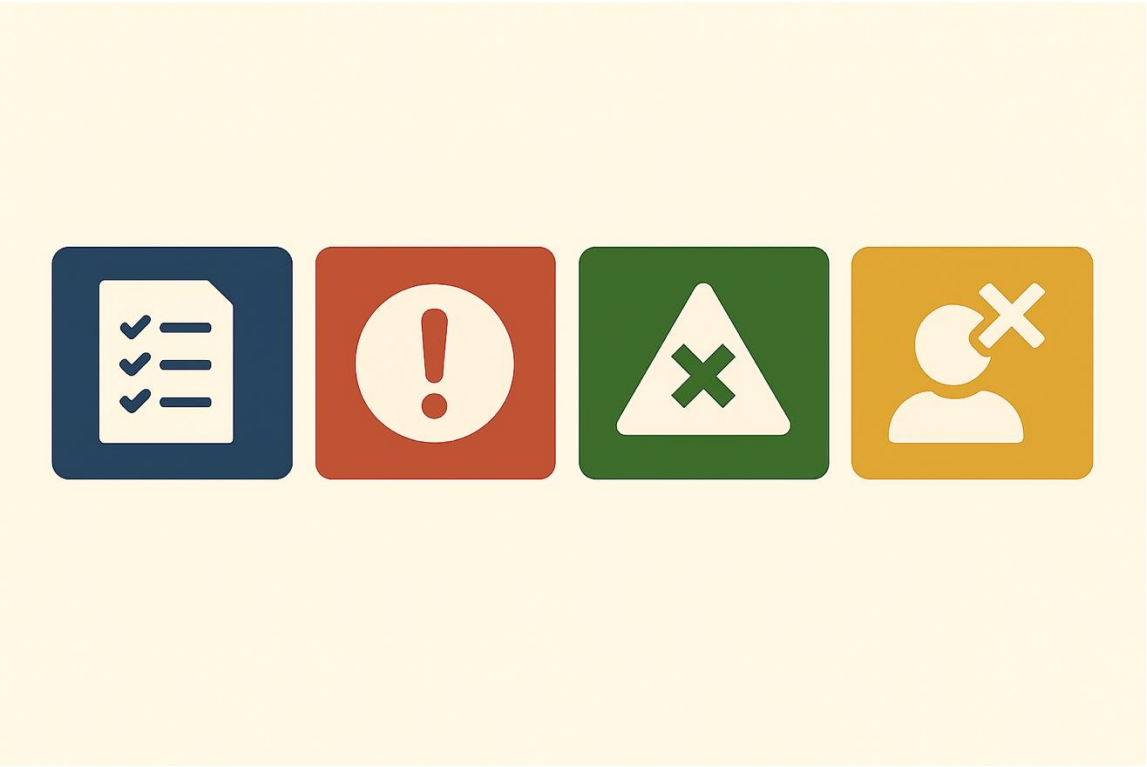
RUP – Strengths

- Iterative and incremental development approach
- Strong documentation and design focus
- Supports risk mitigation early in the process
- Scalable to large and distributed teams
- Emphasizes testing and quality assurance



RUP – Disadvantages

- Can be overly complex and heavyweight
- Requires thorough training and expertise
- Documentation-heavy process can slow down development
- May not be flexible enough for fast-changing projects



RUP – Best Use Case

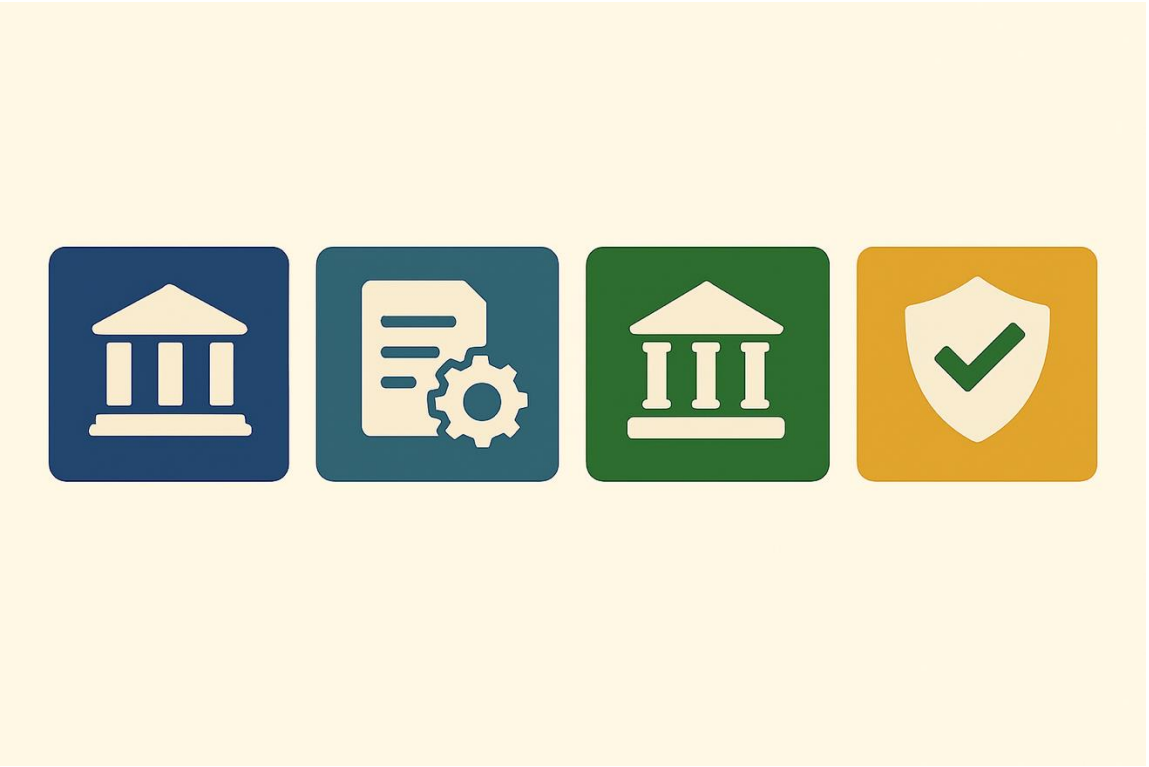
- RUP is best suited for:

- Large enterprise systems

- Projects where requirements are well understood but evolve over time

- Teams needing strong documentation and process control

- Example: Developing a government or banking software system with strict compliance needs.



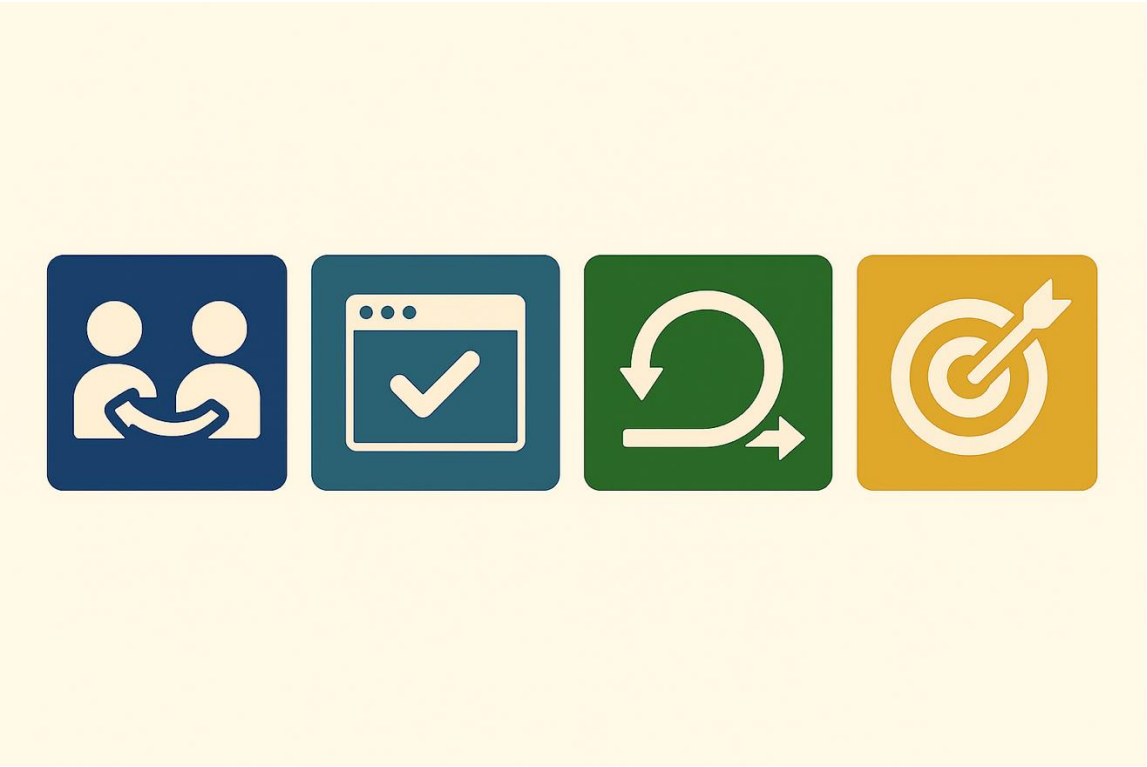
Model 4: Agile

Agile Model – Key Characteristics

- Iterative and incremental approach to software development
- Focused on customer collaboration, early delivery, and continuous improvement
- Delivers working software in short cycles called sprints (typically 1–4 weeks)
- Emphasizes adaptive planning and responding to change
- Encourages cross-functional teams and face-to-face communication
- Based on the Agile Manifesto principles (e.g., working software over documentation)

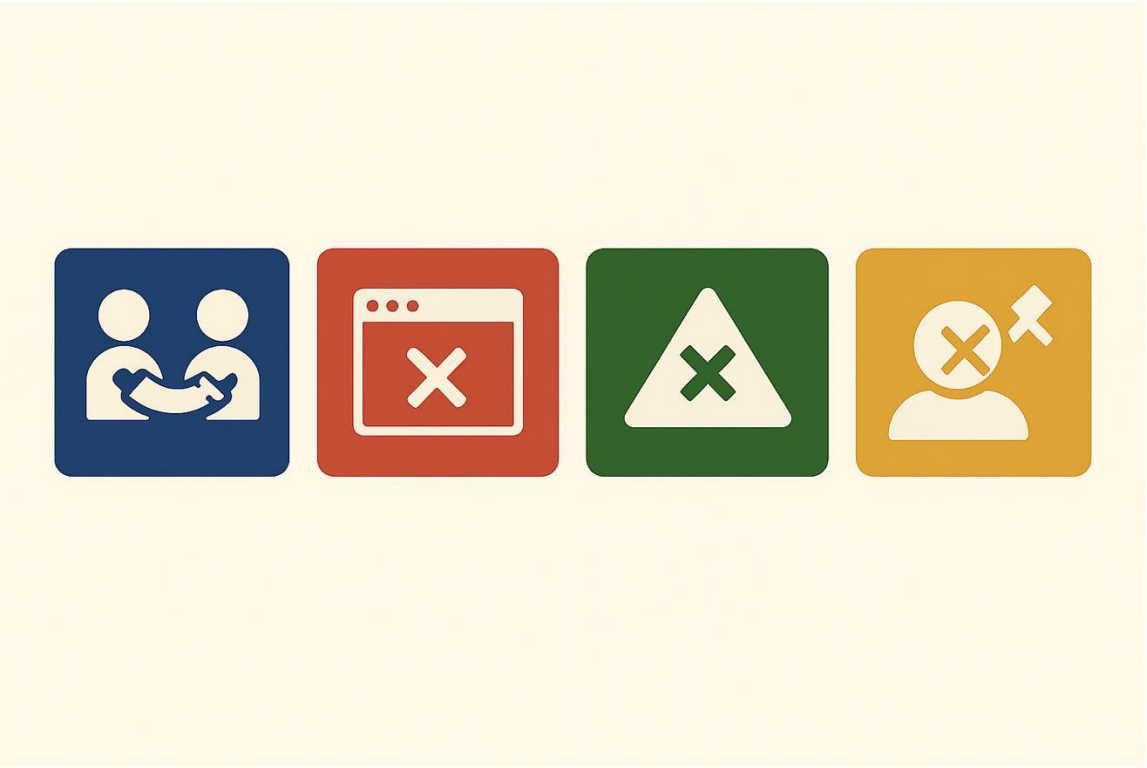
Agile – Strengths

- High customer involvement and satisfaction
- Rapid delivery of working software
- Flexible to changing requirements
- Encourages team collaboration and ownership
- Continuous feedback and improvement



Agile – Disadvantages

- Requires active and consistent customer participation
- Difficult to scale for large, complex projects
- Less emphasis on documentation
- Can lead to scope creep without disciplined management



Agile – Best Use Case

- Agile is best suited for:

- Projects with evolving requirements

- Small to medium-sized teams

- Startups and fast-paced product development

- Example: Building a mobile app where frequent user feedback is crucial.

Project Models – Summary Comparison

Aspect	Waterfall	Spiral	RUP	Agile
Approach	Linear & Sequential	Iterative with Risk Analysis	Iterative and Structured Phases	Iterative & Incremental
Planning Style	Heavy Upfront Planning	Risk-Driven Cycles	Phase-Based with Clear Roles	Flexible, Continuous Planning
Customer Involvement	Low	Medium to High	Medium	Very High
Risk Handling	Poor	Strong (focus on risk assessment)	Moderate	Adaptive, relies on feedback loops
Change Flexibility	Rigid (hard to change mid-way)	Allows for changes between iterations	Moderate – Supports changes at key stages	Very Flexible – Welcomes change
Best For	Well-defined, stable requirements	High-risk, complex, exploratory projects	Large-scale systems with defined workflows	Fast-changing projects, startups, user-focused apps
Documentation	Heavy	Moderate to Heavy	Detailed & Role-Based	Light & Just-in-Time
Team Size	Medium to Large	Large (due to complexity & risk tracking)	Medium to Large	Small to Medium