SOEN 6841 Software Project Management
Winter 2025
Chapter 3 Exercise
Amanpreet Kaur (40301892)

3.2 Agile projects may have less effort required compared to traditional projects. What factors are responsible for this phenomenon?

Answer:

Agile methodologies significantly reduce the effort required for software development compared to traditional Waterfall approaches. This is achieved through flexibility, continuous feedback, and an iterative approach, which optimize both time and resources. Following are the key factors how Agile minimizes effort while maximizing productivity and software quality:

- **Adaptability Saves Time on Planning & Rework**
  Agile embraces **changing requirements** without excessive upfront planning. Unlike Waterfall, which requires a fixed scope, Agile allows for adjustments **on the go**, reducing rework.
  Example: If a client requests a feature change mid-project, Agile can incorporate it into the next sprint instead of overhauling the entire project plan.

- **Continuous Feedback Prevents Unnecessary Work**
  Agile emphasizes regular stakeholder involvement, ensuring that developers build exactly what users need, reducing wasted effort on unwanted features.
  Example: In a Scrum project, weekly sprint reviews with customers ensure that the product evolves according to real user feedback.

- **Prioritization of High-Value Features**
  Agile teams focus on the most important features first, delivering business-critical functionalities early rather than spending effort on secondary features.
  Example: An e-commerce app prioritizes a smooth checkout experience over less critical features like theme customization.

- **Iterative Development Reduces Waste**
  Agile follows an incremental approach, delivering working software in small iterations. This prevents time-consuming work on unnecessary features and allows teams to pivot early based on feedback.
  Example: A development team releases a minimum viable product (MVP) within weeks, instead of waiting months to deliver a fully completed system that may need changes.

- **Reduced Documentation Overhead**
  Instead of exhaustive documentation, Agile values working software and direct communication. This reduces effort spent writing and updating lengthy documents.
  Example: A team using user stories and Kanban boards to track progress instead of writing detailed requirement documents saves time and effort.

- **Automation & Continuous Testing Reduce Debugging Effort**
  Agile promotes continuous integration (CI) and automated testing, catching defects early and reducing large-scale bug-fixing efforts later.
  Example: A DevOps pipeline automatically runs tests with every code commit, preventing defects from accumulating.

- **Cross-Functional Teams Reduce Dependencies**
  Agile teams are self-organizing and include developers, testers, and designers working together, minimizing delays caused by dependencies on other teams.
  Example: A product team working in Agile can quickly resolve issues without waiting for approvals from separate departments.

- **High Collaboration Prevents Misalignment & Rework**
  Agile encourages daily stand-up meetings and close team collaboration, reducing misunderstandings and the need for time-consuming revisions.
  Example: A UI/UX designer and a developer co-create wireframes and prototypes instead of waiting for finalized design specs.

Therefore, By focusing on adaptability, automation, early feedback, and value-driven development, Agile **minimizes wasted effort** and ensures teams build **only what truly matters**. This makes Agile a more efficient approach compared to traditional project management methods.