Zeitplanung

Dipl. Inf. (FH) Andreas Holitschke

19. Oktober 2025

1 Zeitplan

Der Zeitplan ist ein wichtiges Werkzeug, um den Fortschritt zu verfolgen und sicherzustellen, dass das Projekt im Zeitrahmen bleibt. Ein gut strukturierter Zeitplan hilft, die Effizienz des Teams zu steigern und die Wahrscheinlichkeit zu erhöhen, dass das Projekt erfolgreich und termingerecht abgeschlossen wird.

1.1 Schritte zur Erstellung eines Zeitplans

1. **Projektziele definieren**
   Klare Ziele und Anforderungen des Projekts festlegen.
2. **Aufgaben identifizieren**
   Alle notwendigen Aufgaben und Aktivitäten auflisten, die zur Erreichung der Projektziele erforderlich sind.
3. **Aufgaben priorisieren**
   Die Aufgaben nach ihrer Wichtigkeit und Abhängigkeiten priorisieren.
4. **Ressourcen zuweisen**
   Bestimmen, welche Teammitglieder oder Ressourcen für jede Aufgabe benötigt werden.
5. **Zeitaufwand schätzen**
   Für jede Aufgabe den geschätzten Zeitaufwand festlegen.
6. **Meilensteine festlegen**
   Wichtige Meilensteine definieren, die den Fortschritt des Projekts markieren.
7. **Zeitplan erstellen**
   Die Aufgaben, Zeitaufwände und Meilensteine in einem Zeitplan zusammenfassen, z.B. in Form eines Gantt-Diagramms.
8. **Überwachung und Anpassung**
   Den Zeitplan regelmäßig überprüfen und bei Bedarf anpassen, um auf Änderungen oder Verzögerungen zu reagieren.

1.2 Inhalte eines Zeitplans

1. **Projektübersicht**
   Kurze Beschreibung des Projekts, Ziele und Umfang.
2. **Aufgabenliste**
   Detaillierte Liste aller Aufgaben, die im Rahmen des Projekts durchgeführt werden müssen.
3. **Zeitaufwand**
   Geschätzte Dauer für jede Aufgabe (z.B. in Stunden oder Tagen).
4. **Start- und Enddaten**
   Geplante Start- und Enddaten für jede Aufgabe.
5. **Meilensteine**
   Wichtige Meilensteine mit entsprechenden Daten.

6. **Ressourcenzuweisung**
   Zuweisung von Teammitgliedern oder Ressourcen zu den jeweiligen Aufgaben.

7. **Abhängigkeiten**
   Informationen über Abhängigkeiten zwischen Aufgaben (z.B. welche Aufgaben vor anderen abgeschlossen sein müssen).

8. **Risiken und Pufferzeiten**
   Identifikation potenzieller Risiken und Einplanung von Pufferzeiten für unvorhergesehene Verzögerungen.

9. **Überwachungs- und Berichtsmethoden**
   Festlegung, wie der Fortschritt überwacht und berichtet wird (z.B. wöchentliche Meetings, Statusberichte).

Hier ist ein Beispiel für einen Zeitplan für ein Softwareentwicklungsprojekt, das die Phasen der Anforderungsanalyse, des Klassenentwurfs, der Implementierung und des Testens umfasst. Der Zeitrahmen ist auf einen Zeitraum von 12 Wochen ausgelegt, kann jedoch je nach Projektgröße und -komplexität angepasst werden.

1.3 Beispiel-Zeitplan (12 Wochen)

Woche	Aktivität	Beschreibung
1	Projektinitiierung	Kick-off-Meeting, Stakeholder-Identifikation, Projektziele festlegen
2	Anforderungsanalyse	Interviews, Workshops und Umfragen zur Erfassung der Anforderungen
3	Anforderungsdokumentation	Erstellung eines Anforderungsdokuments, das funktionale und nicht-funktionale Anforderungen umfasst
4	Anwendungsfälle definieren	Identifikation und Dokumentation der Anwendungsfälle
5	Klassenmodell entwerfen	Erstellung des Klassenmodells basierend auf den Anwendungsfällen
6	Architekturentwurf	Entwurf der Systemarchitektur und der technischen Infrastruktur
7	Implementierung (Phase 1)	Entwicklung der Kernfunktionen und Klassen gemäß dem Klassenmodell
8	Implementierung (Phase 2)	Fortsetzung der Entwicklung, Integration von Benutzeroberfläche und Backend
9	Testen der Anwendungsfälle	Durchführung von Tests basierend auf den definierten Anwendungsfällen
10	Fehlerbehebung und Optimierung	Behebung von identifizierten Fehlern und Optimierung der Implementierung
11	Benutzerakzeptanztest (UAT)	Durchführung von Tests mit Stakeholdern zur Validierung der Anforderungen
12	Projektabschlus	Abschlussbericht, Dokumentation, Übergabe an den Kunden, Feedback einholen

- **Flexibilität**: Der Zeitplan sollte flexibel sein, um unerwartete Herausforderungen oder Änderungen in den Anforderungen zu berücksichtigen. - **Ressourcenzuweisung**: Stelle sicher, dass die Teammitglieder entsprechend den Aktivitäten und Phasen zugewiesen sind. - **Regelmäßige Meetings**: Plane regelmäßige Teammeetings (z. B. wöchentliche Standups), um den Fortschritt zu überprüfen und Probleme frühzeitig zu identifizieren. - **Dokumentation**: Halte alle Schritte und Ergebnisse gut dokumentiert, um die Nachverfolgbarkeit und Transparenz im Projekt zu gewährleisten.

Dieser Zeitplan dient als allgemeines Beispiel und kann je nach spezifischen Anforderungen und Gegebenheiten des Projekts angepasst werden.

Hier ist ein noch detaillierterer Zeitplan für ein Softwareentwicklungsprojekt über 12 Wochen, der spezifische Aufgaben, Verantwortlichkeiten, Meilensteine, benötigte Ressourcen und mögliche Risiken umfasst.

1.3.1 Detaillierter Beispiel-Zeitplan für ein Softwareentwicklungsprojekt (12 Wochen)

Woche	Aktivität	Aufgaben	Verantwortliche	Meilenstein	Benötigte Ressourcen	Mögliche Risiken
1	Projektinitiierung	• Kick-off-Meeting organisieren • Stakeholder identifizieren • Projektziele und -umfang festlegen • Projektteam zusammenstellen	Projektmanager	Projektziele definiert	Projektteam, Meeting-Raum	Unklare Ziele, fehlende Stakeholder
2	Anforderungsanalyse	• Interviews mit Stakeholdern durchführen • Workshops zur Anforderungserhebung • Umfragen zur Benutzerbedürfnisse • Erstellung eines Anforderungsplans	Business Analyst	Anforderungen gesammelt	Umfragetools, Meeting-Raum	Unzureichende Anforderungen
3	Anforderungsdokumentation	• Erstellung eines Anforderungsdokuments (funktionale und nicht-funktionale Anforderungen) • Review • Meeting zur Validierung der Anforderungen • Anpassungen basierend auf Feedback	Business Analyst	Anforderungsdokument genehmigt	Textverarbeitungssoftware	Missverständnisse in Anforderungen
4	Anwendungsfälle definieren	• Identifikation der Anwendungsfälle • Erstellung von Use-Case-Diagrammen • Dokumentation der Anwendungsfälle • Validierung der Anwendungsfälle mit Stakeholdern	Business Analyst	Anwendungsfälle dokumentiert	UML-Tools, Meeting-Raum	Fehlende Anwendungsfälle
5	Klassenmodell entwerfen	• Analyse der Anwendungsfälle zur Identifikation von Klassen • Erstellung eines Klassendiagramms • Definition von Attributen und Methoden • Review des Modells mit dem Team	Software-Architekt	Klassenmodell erstellt	UML-Tools, Design-Software	Unzureichende Modellierung
6	Architekturentwurf	• Entwurf der Systemarchitektur (z. B. Schichten, Module) • Auswahl der Technologien (z. B. Programmiersprachen, Frameworks) • Erstellung eines Architekturdiagramms	Software-Architekt	Architekturdesign genehmigt	Architektur-Tools, Dokumentation	Technologiewahl nicht optimal
7	Implementierung (Phase 1)	• Entwicklung der Kernfunktionen (z. B. Datenbankanbindung, Authentifizierung) • Implementierung der Hauptklassen • Code • Reviews durchführen • Dokumentation des Codes	Entwicklerteam	Kernfunktionen implementiert	IDEs, Versionskontrollsystem	Verzögerungen durch technische Probleme
8	Implementierung (Phase 2)	• Fortsetzung der Entwicklung (z. B. Benutzeroberfläche, API • Integration) • Implementierung von zusätzlichen Funktionen • Durchführung von Unit • Tests • Dokumentation der neuen Funktionen	Entwicklerteam	Alle Funktionen implementiert	IDEs, Testframeworks	Unzureichende Tests

Woche	Aktivität	Aufgaben	Verantwortliche	Meilenstein	Benötigte Ressourcen	Mögliche Risiken
9	Testen der Anwendungsfälle	• Erstellung von Testfällen basierend auf den Anwendungsfällen • Durchführung von Integrationstests • Dokumentation der Testergebnisse • Feedback-Runde mit dem Team	QA-Team	Anwendungsfälle erfolgreich getestet	Testmanagement Tools	Fehler in der Implementierung
10	Fehlerbehebung und Optimierung	• Analyse der Testergebnisse und Identifikation von Fehlern • Behebung von Bugs und Optimierung des Codes • Durchführung von Regressionstests • Dokumentation der Änderungen	Entwicklerteam	Fehler behoben, System stabil	IDEs, Testframeworks	Zeitüberschreitung bei der Behebung
11	Benutzerakzeptanztest (UAT)	• Planung und Durchführung von UAT-Tests mit Stakeholdern • Sammlung von Feedback und Verbesserungsvorschlägen • Anpassungen basierend auf dem Feedback • Abschlussbericht für UAT erstellen	QA-Team, Stakeholder	UAT abgeschlossen, Feedback gesammelt	Testumgebung, Feedback-Tools	Stakeholder sind unzufrieden
12	Projektabschluss	• Erstellung eines Abschlussberichts • Dokumentation aller Ergebnisse und Learnings • Übergabe des Projekts an den Kunden • Abschlussmeeting zur Reflexion	Projektmanager	Projekt offiziell abgeschlossen	Dokumentationssoftware	