Lernzettel

Anforderungsanalyse und Spezifikation

Universität: Technische Universität Berlin
Kurs/Modul: Softwaretechnik und Programmierparadigmen
Erstellungsdatum: September 19, 2025

Lernzettel: Anforderungsanalyse und Spezifikation

(1) Zielsetzung der Anforderungsanalyse. Die Anforderungsanalyse dient als Grundlage für alle weiteren Phasen der Softwareentwicklung. Sie verwandelt Stakeholder-Bedürfnisse in klare, überprüfbare Anforderungen. Ziel ist es, Missverständnisse zu vermeiden, Nachvollziehbarkeit sicherzustellen und eine klare Validierungskette zu ermöglichen.

(2) Stakeholder-Analyse und Kontext. Identifikation relevanter Stakeholderinnen und Stakeholder (Kunde, Endnutzer, Projektleitung, Wartung, Betrieb). Woraus ergeben sich Anforderungen? aus Zielen, Constraints, Regeln, Sicherheits- und Compliance-Vorgaben. Technische Rahmenbedingungen beeinflussen die Machbarkeit. Methoden: Interviews, Workshops, Fragebögen, Beobachtung.

(3) Anforderungstypen.

- Funktionale Anforderungen: Beschreiben, was das System tun soll (Funktionen, Prozesse, Interaktionen).
- Nicht-funktionale Anforderungen: Qualitätseigenschaften wie Performanz, Sicherheit, Zuverlässigkeit, Benutzbarkeit, Portabilität.
- Domänen- und Randbedingungen: Gesetze, Standards, vorhandene Systeme, Schnittstellen.

(4) Spezifikationstechniken.

- Use Cases und Anwendungsfälle: Schauspieler/ Akteure, Ziele, Schritte, Vorbedingungen, Nachbedingungen.
- User Stories: Kurzformatierte Anforderungen aus Sicht des Nutzers; Akzeptanzkriterien.
- Vor-/Nachbedingungen und Spezifikationsverträge: Klar definierte Garantien.
- Modellierungssprachen: UML-Diagramme (Anwendungsfall, Aktivitätsdiagramm, Sequenzdiagramm) dienen der Visualisierung.

(5) Qualitätsmerkmale und Verifikation.

- Vollständigkeit, Konsistenz, Nicht-Widersprüchlichkeit der Anforderungen.
- Verifizierbarkeit und Validierbarkeit: Kann bestätigt werden, dass Anforderungen erfüllt sind?
- Metriken: Abdeckung, Änderungsaufwand, Nachverfolgbarkeit (Traceability).
- Verifikationstechniken: Hoare-Kalkül, formale Beweise, Tests.

(6) Nicht-funktionale Anforderungen. Beispiele: Leistungsbedarf (Antwortzeit), Skalierbarkeit, Sicherheit, Zuverlässigkeit, Verfügbarkeit, Wartbarkeit, Usability. Hinweis: Nicht-funktionale Anforderungen beeinflussen Entwurf und Architektur maßgeblich.

(7) Use Cases, User Stories Abnahme. Use Cases helfen beim Ermitteln von Interaktionen; User Stories liefern kurze, nutzerorientierte Beschreibungen mit Akzeptanzkriterien. Abnahme erfolgt durch Testfälle, die die Erfüllung der Anforderungen demonstrieren.

(8) Verifikation, Validierung und Nachverfolgbarkeit.

- Verifikation: Wurzel aus formalen Spezifikationen, Hoare-Kalkül, Tests.
- Validierung: Sicherstellen, dass das richtige System gebaut wird (Kundenziele sind erfüllt).
- Nachverfolgbarkeit: Requirement-Traceability von Stakeholdern zu Implementierung und Tests.

(9) Praktische Modellierung und Dokumentation.

- Architektursicht: Zerstäuben in Module, klare Schnittstellen.
- Aktivitäts- und Sequenzdiagramme helfen beim Fluss der Interaktionen.
- Spezifikationsdokumente: klar, konsistent, prüfbar, frei von Mehrdeutigkeiten.