Vorlesung „Softwaretechnik"
Zusatzbemerkungen zum Thema

Wartung und Pflege

Dr. Lutz Prechelt

Interpretation von Wartung und Pflege

- Die scharfe Unterscheidung zwischen Entwicklung und Wartung/Pflege stammt aus dem Wasserfallmodell
- Modernere Sichten:
  - Inkrementelles Modell: Entwicklung einer neuen Produktversion
    - Z.B. Bei eXtreme Programming
  - Allgemein bei agilen Methoden:
    - Versuche so schnell wie möglich in die „Pflegephase" zu kommen, d.h. ein Produkt in den echten Einsatz zu bekommen
    - Stammt aus der Erfahrung, dass man vorher viele Anforderungen einfach nicht sieht oder nicht gut versteht

Interpretation von Wartung und Pflege (2)

- Aus Sicht der Entwurfsphase ist Pflege lediglich ein Entwicklungsprojekt mit besonders hohem Grad an Wiederverwendung
  - Also wünschenswert
  - Das heisst aber, es muss auch geben:
    - Schriftlich festgehaltene Anforderungen
      - Nicht nur die aktuelle Anpassung oder Erweiterung ist die Anforderung, sondern alles!
    - Einen expliziten Entwurf
    - Tests gegen alle Anforderungen
      - Daher kommt z.B. die scharfe Forderung nach automatisierten Tests bei XP
  - Hauptaufgabe bei Pflege ist deshalb Entwurf
    - Verlangt vor allem genaues Verständnis der „wiederverwendbaren Bausteine" (d.h. der existierenden SW)
      - Empirisch belegt: Hauptzeitanteil entfällt auf Programmverstehen

Änderungsmanagement

- Vorgehen ist grundsätzlich das Gleiche wie während der Entwicklung nach Beginn der Implementierungsphase
- Hauptkriterium ist stets: Kosten/Nutzen-Verhältnis

- Anmerkung:
  - Fokussierung auf Nutzen liegt auch der Funktionspunktmethode zur Aufwandsschätzung zugrunde
  - Ebenso dem Planungsspiel bei XP

Anreizsystem für Wartung und Pflege

- Die meisten Entwickler bauen lieber neues als zu warten. Gründe:
  - Freude am Neuen
  - Freude am Erschaffen
  - Faulheit: Wartung ist subjektiv schwieriger
- Anreize zum Maximieren der Wartungsbereitschaft
  Anreize zum Minimieren des Wartungsaufwands
  - Erstentwickler weiss, dass er selbst für Wartung zuständig sein wird
    - Führt zu: Entwurf/Implementierung klarer, Dokumentation nützlicher
  - Entwickler wird am wirtschaftlichen Erfolg beteiligt
    - An geringen Wartungskosten
    - An hohen Verkaufszahlen bzw. Kundenzufriedenheit
    - Führt zusätzlich zu besser hinterfragten Anforderungen

Software-Alterung

- Entsteht, wenn Änderungen nicht passend zum Restentwurf entworfen sind
  - Allmählich: Entwurfsverfall (design decay)
  - Langfristig: „Architekturdrift"
- Gründe für Entwurfsverfall sind
  - Terminlich: Änderung unter Zeitdruck
  - Entwurfsstruktur: Neue Anforderung zu schwer im existierenden Entwurf unterzubringen
  - Organisatorisch: Wartungspersonal kennt Entwurf nicht gut genug
    - Z.B. nie eingewiesen, nicht miterlebt, nicht dokumentiert
- Es kann ökonomisch vernünftig sein, Entwurfsverfall in Kauf zu nehmen
  - Fragt sich allerdings, wie stark, wie schnell und ob in jedem Einzelfall

Software-Sanierung

- Soll eine SW nach starkem Entwurfsverfall weiter gepflegt werden, empfiehlt sich eine Software-Sanierung
  - Forme SW so um, dass sie wieder sauberen Entwurf hat
  - Wähle diesen so, dass künftig erwartete Anforderungen leicht erfüllt werden können
- Sehr schwierige Aufgabe
- Hauptproblem 1: Verstehen der (verfallenen) SW
- Hauptproblem 2: Transformation, ohne Defekte einzuführen

Danke!