

Software Engineering II (IB)

Wiederverwendung von Software

Prof. Dr. Oliver Braun

Letzte Änderung: 16.05.2017 21:17

Wiederverwendung von Software

- ▶ Strategie bei der Entwicklung auf bereits vorhandene Software zurück zu greifen
- ▶ bereits vor mehr als 40 Jahren als Entwicklungsstrategie vorgeschlagen
- ▶ erst seit 2000 zur Norm für neue Unternehmenssysteme geworden:

Entwicklung mit Wiederverwendung

- ▶ Reaktion auf die Forderung nach
 - ▶ geringeren Produktions- und Wartungskosten
 - ▶ schnellerer Auslieferung
 - ▶ erhöhter Softwarequalität
- ▶ immer mehr Unternehmen sehen ihre Software als wertvollen Aktivposten

Verfügbarkeit wiederverwendbarer Software

- ▶ hat drastisch zugenommen
- ▶ insbesondere auf Grund der Open-Source-Bewegung gibt es eine riesige wiederverwendbare Codebasis
- ▶ Code liegt in Form von Bibliotheken oder ganzen Anwendungen vor
- ▶ viele fachbereichsspezifische Systeme, die angepasst werden können
- ▶ auch große Softwarehäuser bieten wiederverwendbare Komponenten
- ▶ dank Standards, wie z.B. Webservices, können leicht allgemeine Dienste entwickelt und zur Verfügung gestellt werden

Wiederverwendbare Einheiten

- ▶ die wiederverwendbaren Softwareeinheiten fallen von der Größe her sehr unterschiedlich aus
- ▶ Wiederverwendung von Anwendungssystemen
 - ▶ ein ganzes Anwendungssystem wird unverändert oder angepasst verwendet
 - ▶ es gibt auch sog. Anwendungsfamilien, die eine gemeinsame Architektur aufweisen, aber an die Bedürfnisse einzelner Kunden angepasst werden können
- ▶ Wiederverwendung von Komponenten
 - ▶ Größe von Subsystem bis zum einzelnen Objekt
- ▶ Wiederverwendung von Objekten und Funktionen
 - ▶ seit 40 Jahren in Form von Standardbibliotheken üblich
- ▶ ergänzende Form: Wiederverwendung von Konzepten
 - ▶ z.B. Patterns, konfigurierbare Systemprodukte oder Programmgeneratoren

Vorteile der Wiederverwendung

- ▶ Höhere Zuverlässigkeit
 - ▶ Komponenten sind schon in der Praxis getestet
 - ▶ Schwachstellen im Entwurf und in der Implementierung sollten weitestgehend aufgespürt und beseitigt sein
- ▶ geringes Risiko
 - ▶ Kosten sind bekannt
- ▶ effektiver Einsatz von Spezialisten
 - ▶ Anwendungsspezialisten für die Komponenten
- ▶ Übereinstimmung mit Standards
 - ▶ Standardlösungen, wie z.B. Benutzeroberflächen
- ▶ beschleunigte Entwicklung
 - ▶ frühestmögliche Markteinführung

Probleme

- ▶ höhere Wartungskosten
 - ▶ ist Quellcode nicht verfügbar, erhöhen sich möglicherweise die Wartungskosten
- ▶ Mangel an Werkzeugunterstützung
 - ▶ einige Entwicklungswerkzeuge unterstützen die Entwicklung mit Wiederverwendung nicht
- ▶ “Nicht-hier-erfunden”-Syndrom (*Not-invented-here Syndrom*)
 - ▶ einige Softwareentwickler ziehen es vor, Komponenten umzuschreiben
 - ▶ Vertrauen anderen nicht und sehen das Schreiben neuer Software als größere Herausforderung
- ▶ Aufbau Wartung und Verwendung einer Komponentenbibliothek
 - ▶ selbst eine wiederverwendbare Bibliothek zu implementieren ist mitunter sehr anspruchsvoll
- ▶ Suchen, Verstehen und Anpassen wiederverwendbarer Komponenten

Die Wiederverwendungslandschaft

- ▶ Architekturmuster
- ▶ Entwurfsmuster
- ▶ komponentenbasierte Entwicklung
- ▶ Anwendungsframeworks
- ▶ Umhüllen von Altsystemen
- ▶ serviceorientierte Systeme
- ▶ Softwareproduktlinien
- ▶ Wiederverwendung von COTS-Produkten
- ▶ ERP-Systeme
- ▶ konfigurierbare vertikale Anwendungen (generische Systeme)
- ▶ Programmbibliotheken
- ▶ modellgetriebene Entwicklung
- ▶ Programmgeneratoren
- ▶ aspektorientierte Softwareentwicklung

Schlüsselfaktoren bei der Planung der Wiederverwendung

- ▶ Entwicklungszeitplan für die Software
- ▶ voraussichtliche Lebensdauer der Software
 - ▶ bei langer Lebensdauer, Hauptaugenmerk auf Wartung
- ▶ Fähigkeiten, Hintergrund und Erfahrung des Entwicklerteams
- ▶ Wichtigkeit der Software und ihrer nichtfunktionalen Anforderungen
 - ▶ wenn Software zertifiziert werden muss, kann es problematisch werden, wenn der Quellcode nicht vorhanden ist
- ▶ Anwendungsbereich
- ▶ Plattform auf der das System ausgeführt werden soll

Anwendungsframeworks

- ▶ mögliche Definition
... ein integrierter Satz an Softwareartefakten (wie Klassen, Objekte und Komponenten), die zusammenwirken, um eine wiederverwendbare Architektur für eine Familie von verwandten Anwendungen bereitzustellen.
- ▶ mögliche Klassen
 1. Frameworks für die Systeminfrastruktur
 2. Frameworks zur Integration von Middleware
 - ▶ z.B. .NET oder EJB
 3. Frameworks für Unternehmensanwendungen

Beispiel Webframework

- ▶ Webframeworks bieten i.d.R. folgende Funktionen
- ▶ Sicherheit (Authentifizierung, Zugriffssteuerung)
- ▶ dynamische Webseiten
- ▶ Datenbankunterstützung
- ▶ Sitzungsverwaltung
- ▶ Benutzerinteraktion

Softwareproduktlinien

- ▶ eine der effektivsten Ansätze zur Wiederverwendung
- ▶ Satz von Anwendungen mit
 - ▶ gemeinsam genutzter Architektur
 - ▶ gemeinsam genutzten Komponenten
- ▶ jede Anwendung ist einem speziellen Problem gewidmet
- ▶ das Kernsystem ist so entworfen, dass es an die Bedürfnisse verschiedener Kunden angepasst werden kann

Anwendungsframeworks vs. Softwareproduktlinien

- ▶ Anwendungsframeworks
 - ▶ bleiben unverändert und werden nur genutzt
 - ▶ eher technische Unterstützung
 - ▶ softwareorientiert, selten Unterstützung für Hardwareschnittstellen
- ▶ Softwareproduktlinien
 - ▶ Komponenten werden angepasst und verändert
 - ▶ Domänen- und Plattforminformationen
 - ▶ oft Steuerungsanwendungen für Geräte
 - ▶ Familie verwandter Anwendungen, im Besitz eines Unternehmens

Arten der Spezialisierung eine Softwareproduktlinien

- ▶ Plattformspezialisierung
 - ▶ z.B. Windows, Mac, Unix/Linux
- ▶ Umgebungsspezialisierung
 - ▶ für verschiedene Betriebsumgebungen oder Peripheriegeräte
- ▶ Funktionale Spezialisierung
 - ▶ z.B. Bibliothekssystem für öffentliche Bibliothek, Universitätsbibliothek, Handbibliothek
- ▶ Prozessspezialisierung
 - ▶ an verschiedene Geschäftsprozesse angepasst

Entwicklung von Produktinstanzen

- ▶ Anforderungen der Beteiligten ermitteln
- ▶ System auswählen, das den Anforderungen am Ehesten entspricht
- ▶ evtl. Anforderungen neu verhandeln
- ▶ vorhandenes System anpassen
- ▶ neues Familienmitglied übergeben

Entwicklung

- ▶ Kompromiss zwischen maximaler Wiederverwendung und Erfüllung der genauen Anforderungen
- ▶ je detaillierter die Anforderungen sind, umso unwahrscheinlicher, dass die vorhandenen Komponenten sie erfüllen
- ▶ Softwareproduktlinien berücksichtigen bereits im Entwurf die Rekonfiguration
 - ▶ Komponenten hinzufügen oder entfernen
 - ▶ Parameter und Einschränkungen definieren
 - ▶ Kenntnisse über Geschäftsprozesse einzubeziehen
- ▶ auf verschiedenen Stufen des Entwicklungsprozesses
 - ▶ Konfiguration zur Entwurfszeit
 - ▶ durch Softwarefirma für Kunden
 - ▶ Konfiguration zum Bereitstellungszeitpunkt
 - ▶ durch Kunde selbst oder Berater

Konfiguration zum Bereitstellungszeitpunkt

- ▶ Komponentenauswahl
 - ▶ Module, die die erforderliche Funktionalität bieten
- ▶ Ablauf- und Regeldefinition
 - ▶ Verarbeitung der Daten
 - ▶ Validierungsregeln
- ▶ Parameterdefinition
 - ▶ Werte spezieller Systemparameter spezifizieren

Wiederverwendung von COTS-Produkten

- ▶ COTS-Produkte (*Commercial-Off-The-Shelf*)
- ▶ im Handel erhältliche Produkte
- ▶ die ohne Änderungen am Quellcode an die Bedürfnisse verschiedener Kunden angepasst werden können
- ▶ praktisch alle Desktop-Programme und viele Server-Programme
- ▶ für die Anpassung sind Konfigurationsmechanismen integriert
- ▶ 2 Arten (Details später)
 - ▶ COTS-Lösungen
 - ▶ COTS-Integration

Vorteile bei COTS-Wiederverwendung

- ▶ viel schnellere Bereitstellung
- ▶ Funktionalität bereits bekannt, andere Unternehmen haben bereits Erfahrungen
- ▶ Unternehmen können sich auf Kerngeschäft konzentrieren
- ▶ Technologieupdates i.d.R. einfacher

Probleme bei COTS-Wiederverwendung

- ▶ Anforderungen müssen an Funktionalität und Betriebsweise des COTS-Produktes angepasst werden
 - ▶ evtl. Änderungen der Geschäftsprozesse notwendig
- ▶ COTS-Produkt basiert meist auf Annahmen, die praktisch nicht zu ändern sind
 - ▶ Kunde muss die Annahmen erfüllen
- ▶ Auswahl kann sehr schwierig sein, COTS-Produkte meist nur unzureichend dokumentiert
- ▶ fachliches Wissen vor Ort fehlt
- ▶ Anbieter des COTS-Produktes kann konkurs gehen, von anderem Unternehmen übernommen, Änderungen vornehmen, die für den Kunden mit Schwierigkeiten verbunden sind, ...

- ▶ generische Anwendungssystem, unterstützen
 - ▶ Geschäftstyp,
 - ▶ Geschäftsaktivität oder
 - ▶ komplettes Geschäftsunternehmen
- ▶ z.B. COTS-Lösung für Zahnarzt mit Terminplanung, Patientendatenverwaltung, Abrechnung, ...
- ▶ typischer Vertreter: ERP-System (*enterprise resource planning*)

Generisches ERP-System

- ▶ Module zur Unterstützung verschiedener Geschäftsfunktionen
- ▶ definierter Satz von Geschäftsprozessen zu jedem Modul
- ▶ gemeinsame Datenbank
- ▶ Satz von Geschäftsregeln, die für alle Daten gelten

Konfiguration eines ERP-Systems oder anderem fachspezifischen COTS-Produkt

- ▶ Wahl der erforderlichen Funktionalität (z.B. Wahl der Module)
- ▶ Erstellung eines Datenmodells
- ▶ Definition von Geschäftsregeln
- ▶ Definition der erwarteten Interaktionen mit externen Systemen
- ▶ Entwurf der Eingabeformulare und Ausgabeberichte
- ▶ Entwurf neuer Geschäftsprozesse, die dem zugrunde liegenden Prozessmodell entsprechen, das vom System unterstützt wird
- ▶ setzen von Parametern für die genutzte Plattform

COTS-Integration

- ▶ Anwendungen aus zwei oder mehreren COTS-Produkten, z.T. auch aus Altsystemen
- ▶ COTS-Produkte können über APIs oder Serviceschnittstellen miteinander kommunizieren (falls diese definiert sind)
- ▶ alternativ können auch Ausgaben mit Eingaben verknüpft werden oder eine gemeinsame Datenbank genutzt werden

Entwurfsentscheidungen

- ▶ Welche COTS-Produkte eignen sich von der Funktionalität her am besten?
- ▶ Wie werden die Daten ausgetauscht?
- ▶ Welche Funktionen eines Produktes werden tatsächlich verwendet?

Probleme der Integration von COTS-Systemen

- ▶ Mangel an Kontrolle über Funktionalität und Leistung
- ▶ Probleme mit der Interoperabilität von COTS-Systemen
- ▶ keine Kontrolle über die Evolution des Systems
- ▶ schlechte Unterstützung durch die Anbieter von COTS-Systemen