Wirtschaftsinformatik II
Johannes Maria Zaha

Komponentenfindung in monolithischen betrieblichen Anwendungssystemen

Andreas Krammer und Johannes Maria Zaha

Lehrstuhl Wirtschaftsinformatik II
Universität Augsburg
Universitätsstraße 16, 86135 Augsburg
Tel.: +49(821)598-4431, Fax -4432
E-Mail: {andres.krammer|johannes.maria.zaha}@wiwi.uni-augsburg.de
URL: http://wi2.wiwi.uni-augsburg.de

Wirtschaftsinformatik II
Johannes Maria Zaha

Komponentenfindung in monolithischen betrieblichen Anwendungssystemen

Agenda

- Einordnung und Einführung
- Monolithische versus komponentenorientierte Anwendungssysteme
- Transformation
  - Zerlegung in funktionaler Sicht
  - Gegenüberstellung der implementierten Methoden bzw. Prozeduren
  - Identifikation der Komponenten-Frameworks
  - Identifikation von Fachkomponenten
- Ausblick und Diskussion

Einordnung

- Design for Component
  - Neuentwicklung kleiner, funktional leicht definierbarer Komponenten
  - Schnelle und einfache Integration der Software-Komponenten
- Design from Component
  - Komposition von Komponenten zu einem betrieblichen Anwendungssystem
  - Parameterisierung, technische und fachliche Integration
- Design to Component
  - Redesign der Architektur existierender Altanwendungen
  - Schaffung eigenständiger Komponenten durch Identifikation bzw. programmtechnische Umgestaltung

[vgl. HöWe2001, S. 5]

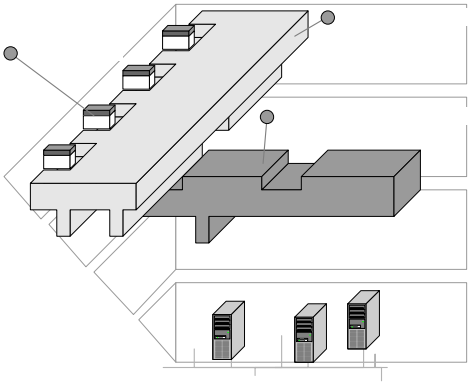
Einführung

- Architektur komponentenorientierter betrieblicher Anwendungssysteme
  - Komponente
    - Black-Box, bestehend aus verschiedenartigen Software-Artefakten
    - Wiederverwendbarkeit, Abgeschlossenheit, Vermarktbarkeit
  - Fachkomponente
    - Bietet Dienste einer betrieblichen Anwendungsdomäne an
  - Komponenten-Anwendungs-Framework
    - Stellt anwendungsdomänenbezogene (Standard-) Dienste bereit
  - Komponenten-SystemFramework
    - Stellt anwendungsinvariante, middelwarenahe Dienste bereit

[Turo2002]

Einführung

- Architektur komponentenorientierter betrieblicher Anwendungssysteme



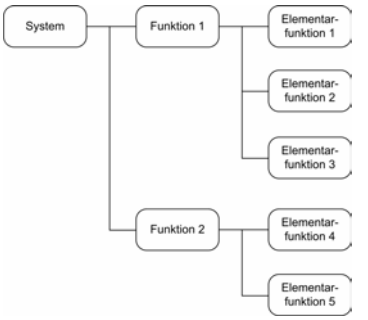
[vgl. Turo2001, S. 36]

Monolithische versus Komponentenorientierte Anwendungssysteme

- Informationstechnische und wirtschaftliche Probleme monolithischer betrieblicher Anwendungssysteme
  - Kostenintensive Wartbarkeit und Erweiterbarkeit
  - Eingeschränkte Vermarktungsmöglichkeiten
  - Eingeschränkte Skalierbarkeit
- Probleme treten bei komponentenorientierten Anwendungssysteme nicht auf
  - Wartungsarbeiten beschränken sich aufgrund der Abgeschlossenheit immer auf wenige Fachkomponenten
  - Weiterentwicklung durch Austausch einzelner Fachkomponenten
  - Verbesserte marktliche Wiederverwendung und Skalierbarkeit

Transformation

- Zerlegung in funktionaler Sicht
  - Hierarchische Zerlegung des betrachteten Anwendungssystems
  - Funktionsdekompositionsdiagramm
    - Gliederung eines Geschäftsprozesses in Funktionen bzw. Elementarfunktionen
    - hier: Zerlegung des Gesamtsystems in Funktionen und Elementarfunktionen

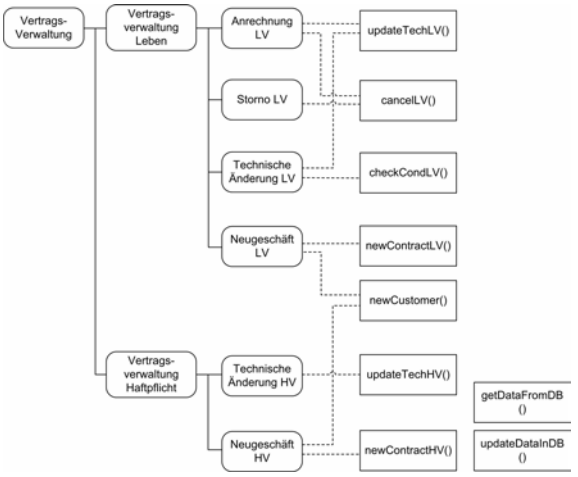


Transformation

- Gegenüberstellung der implementierten Methoden bzw. Prozeduren
  - Bsp.: Vertragsverwaltungssystem für den Versicherungsbereich
  - Identifizierte Funktionen
    - Verwaltung von Lebensversicherungen
    - Verwaltung von Haftpflichtversicherungen
  - Weitere Untergliederung in Elementarfunktionen
  - Zuordnung der Elementarfunktionen zu den implementierten Methoden bzw. Prozeduren
    - Auflistung aus Systemdokumentation bzw. Quellcode
  - Methode bzw. Prozedur kann in mehreren oder keiner Elementarfunktionen genutzt werden

Transformation

- Gegenüberstellung der implementierten Methoden bzw. Prozeduren

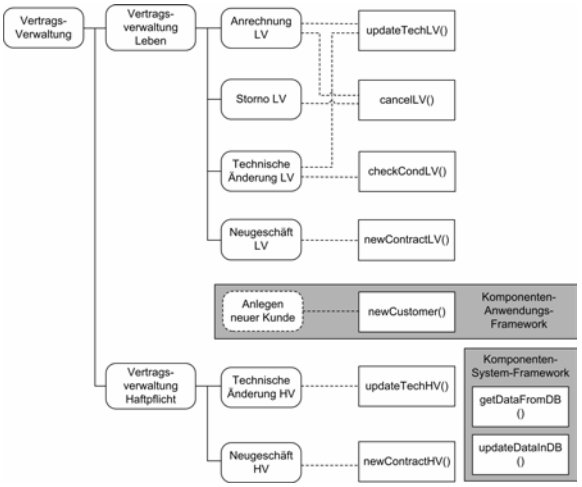


Transformation

- Identifikation der Komponenten-Frameworks
  - Methode bzw. Prozedur ist mit keiner Elementarfunktion verbunden
    - Teil des Komponenten-System-Framework, da kein Bezug zu den fachlichen Aspekten der Anwendung
  - Methode bzw. Prozedur ist mit mehreren Elementarfunktionen verbunden, die unterschiedlichen Funktionen angehören
    - Teil des Komponenten-Anwendungs-Frameworks
    - Integration in mehrere Fachkomponenten ist aufgrund entstehender Redundanz nicht zu empfehlen
  - Komponenten-Anwendungs-Framework stellt anwendungsdomänenbezogene (Standard-) Dienste bereit
    - Methode bzw. Prozedur wird ein neuer Dienst (Elementarfunktion) zugeordnet

Transformation

- Identifikation der Komponenten-Frameworks



Transformation

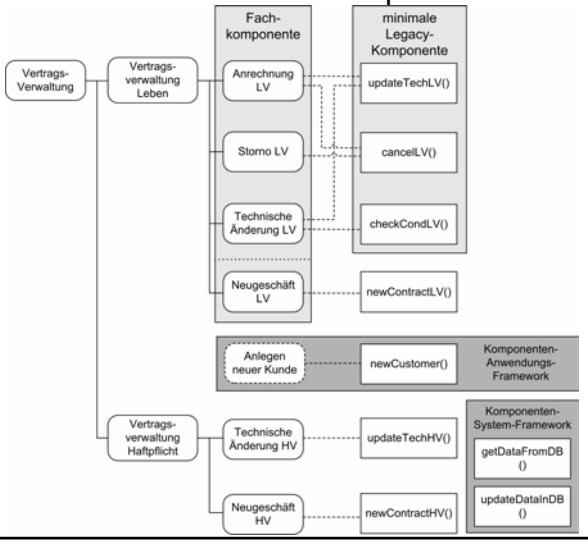
- Identifikation der Fachkomponenten
  - Fachkomponente kann wiederum aus anderen Fachkomponenten bestehen
  - Elementare Fachkomponente
    - es existiert keine andere (kleinere) Fachkomponente, die eine Teilmenge der angebotenen Funktionalität umfasst [vgl. FeTu2000, S. 162]
    - Ist bezüglich der angebotenen Dienste (Elementarfunktionen) disjunkt
  - hier: ausschließlich elementare Fachkomponenten
    - Fachliche Konflikte sind ausgeschlossen
  - Je feingranularer die Fachkomponenten, desto größer auch der Kommunikations- und Koordinationsbedarf
  - Vorteile feingranularer Fachkomponenten
    - Vereinfachung der Wiederverwendung
    - Leichter Erweiterbarkeit um neue (elementare) Fachkomponenten
    - Bessere Vergleichbarkeit der Fachkomponenten

Transformation

- Identifikation der Fachkomponenten
  - Minimale Legacy-Komponente als Untergrenze für die Granularität
    - Minimale Anzahl von Elementarfunktionen (bzw. diesen zugeordneten Methoden/Prozeduren) einer Funktion
    - Eine Methode bzw. Prozedur ist in genau einer minimalen Legacy-Komponente implementiert
  - Methoden bzw. Prozeduren einer Funktion als Obergrenze der in einer Fachkomponente zusammengefassten Dienste
  - Wahl der Größe einer Fachkomponente bietet Entscheidungsspielraum
  - Anhaltspunkt: ausgetauschtes Datenvolumen zwischen Elementen
    - 1. Ebene: Dienste (Elementarfunktionen)
    - 2. Ebene: Methoden bzw. Prozeduren
  - Ziel: Reduktion des Kommunikations- und Koordinationsbedarfs

Transformation

- Identifikation der Fachkomponenten



Ausblick und Diskussion

- Vorgestellter Lösungsalgorithmus
  - Eindeutige Aufteilung vorhandener Implementierungsartefakte auf die Architekturbestandteile komponentenorientierter Systeme
  - Schränkt Gestaltungsspielraum bei der Identifikation von Fachkomponenten ein
  - Übertragbarkeit auf die Transformation auf Client/Server-Systeme ist grundsätzlich gegeben
- Verifikation des Lösungsalgorithmus durch Fallstudie
  - Beschränkung der Methode auf Funktionssicht