12.–15.09.2010
in Nürnberg

Herbstcampus

Wissenstransfer
par excellence

Wollmilchsäue

Wiederverwendung wider Verwendbarkeit

Jan Leßner

arvato services

Wiederverwendung bei arvato services, DC-SWJ

arvato services
BERTELSMANN

- Ca. 70 Softwareentwickler an 4 Standorten, 50% Externe
- Anforderer überwiegend Fachabteilungen von arvato
- Softwareentwicklung zwischen Fabrikation und Manufaktur
- Einige fachliche Schwerpunkte:
  - Kundenbindungssysteme / Bonusprogramme
  - Abrechnung für Telekommunikation
  - Verwaltung von Riesterrentenanträgen
  - Adress-Management
- Pure Java, klassisches JEE-Umfeld
- Projektgrößen: 50 - 1000 PT / 3-10 Personen / 1 - 12 Monate
- 50% individuelle Anforderungen, 50 % wiederkehrende...

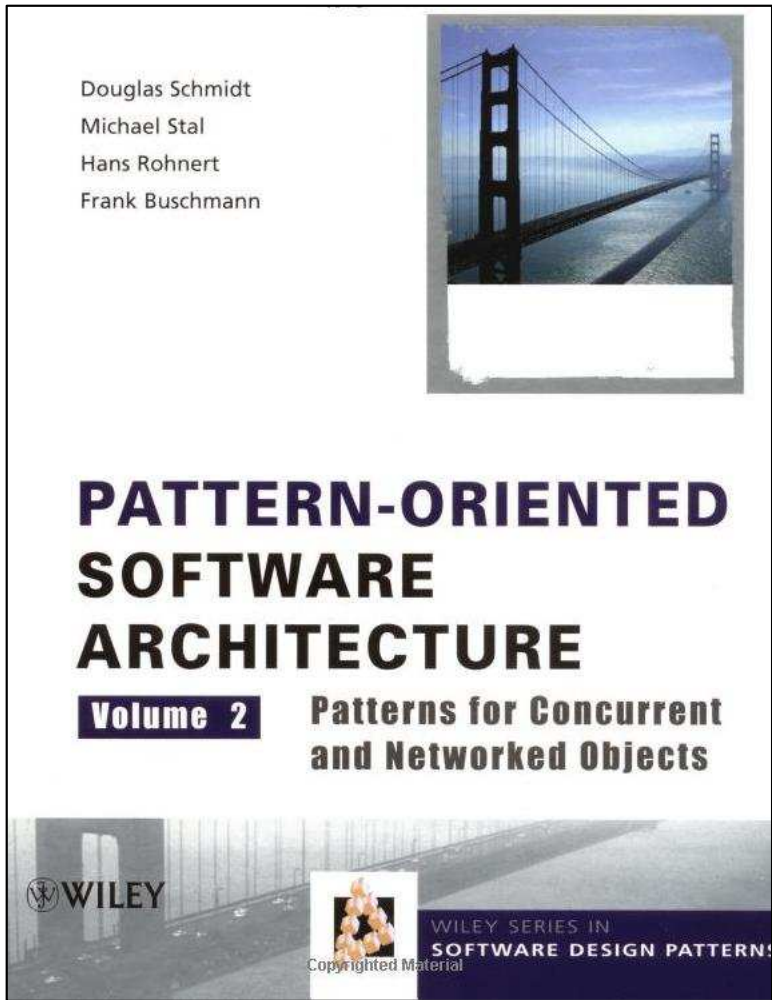
No Future?

arvato services
BERTELSMANN

"Why Software Reuse has Failed..."

Douglas C. Schmidt, Mitautor von
"Pattern-Oriented Software Architecture: Volume 2"

http://www.cs.wustl.edu/~schmidt/reuse-lessons.html



Oder alles ganz berechenbar?

arvato services
BERTELSMANN

- F_{create} = Relative Kosten zur Erzeugung und Verwaltung eines reusable component systems
- $C_{component\text{-}systems}$ = Kosten für die Entwicklung von genug component systems für R Prozent
- $F_{create} >> F_{use}$ $\quad 1 <= F_{create} <= 2.5$
- $C_{family\text{-}saved} = n * C_{saved} - C_{component\text{-}system}$
  $= C_{no\text{-}reuse} * (n * R * (1 - F_{use}) - R$

$$ROI = \frac{C_{family\text{-}saved}}{C_{component\text{-}systems}} = \frac{n * R * (1 - F_{use}) - R * F_{create}}{R * F_{create}}$$

$$= \frac{n * (1 - F_{use}) - F_{create}}{F_{create}}$$

Beispiel: $F_{use} = 0.2$ und $F_{create} = 1.5$

$$ROI = \frac{n * 0.8 - 1.5}{1.5}$$

Break-even mit $n > 2$

Harald Gall, Software Engineering Group
Universität Zürich

www.ifi.uzh.ch/req/ftp/kvse/kap12-reuse.pdf

Es muss wohl heißen...

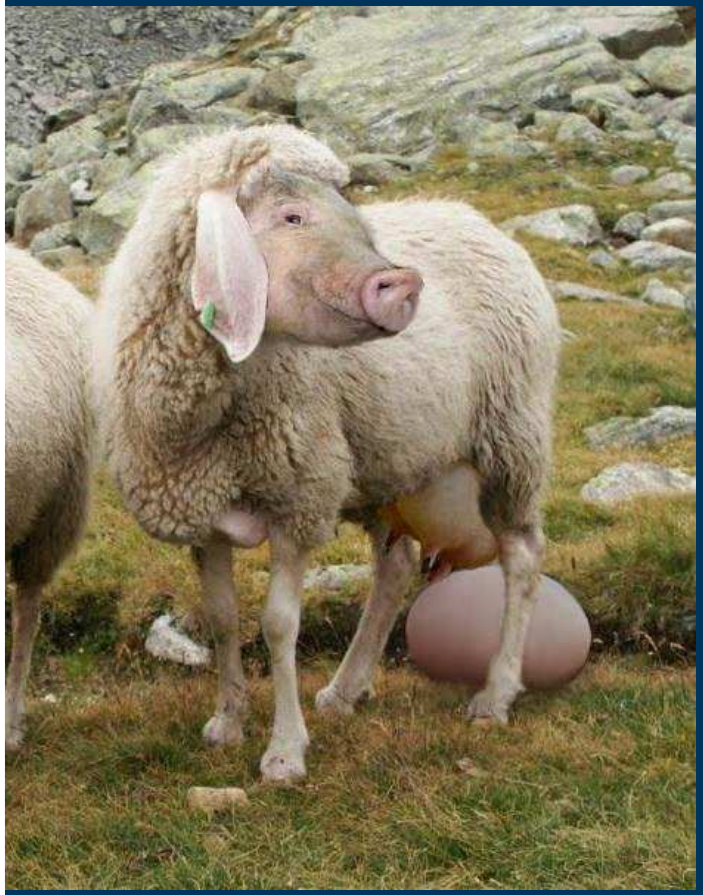
arvato services
BERTELSMANN

"Why Software Reuse has Failed...

... and How to Make It Work for You"

Douglas C. Schmidt

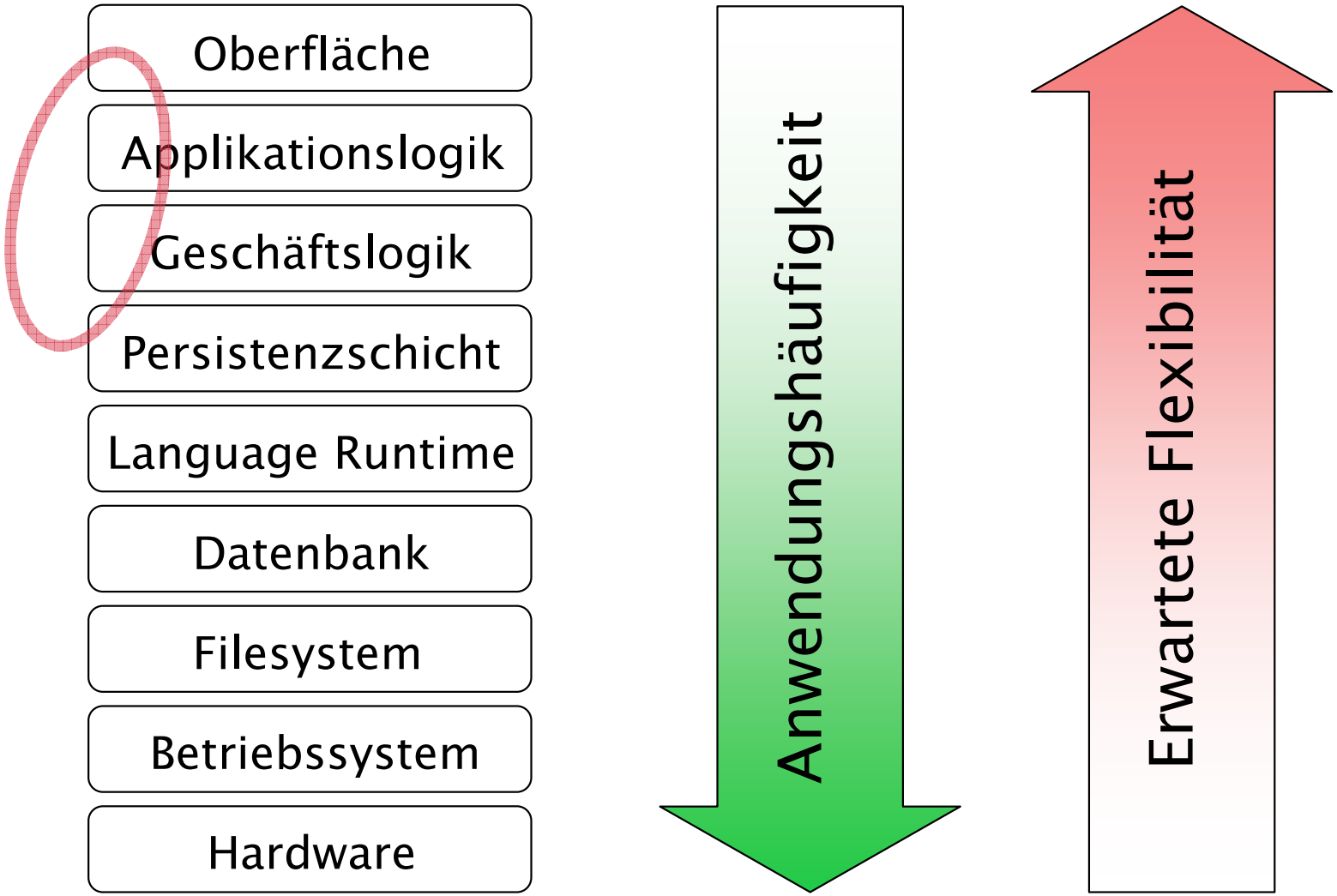
Themenübersicht



- Was soll wiederverwendet werden?
- Das Dilemma
- Beispiel
- Ein paar Auswege

Softwareschichten

arvato services
BERTELSMANN



Das Dilemma

arvato services
BERTELSMANN

- Komplizierte Bausteine möchte man wiederverwenden, um Zeit zu sparen
- Um einen Baustein wiederverwendbar zu machen, müssen Projektspezifika abstrahiert werden
- Durch die Abstraktion wird er noch komplizierter

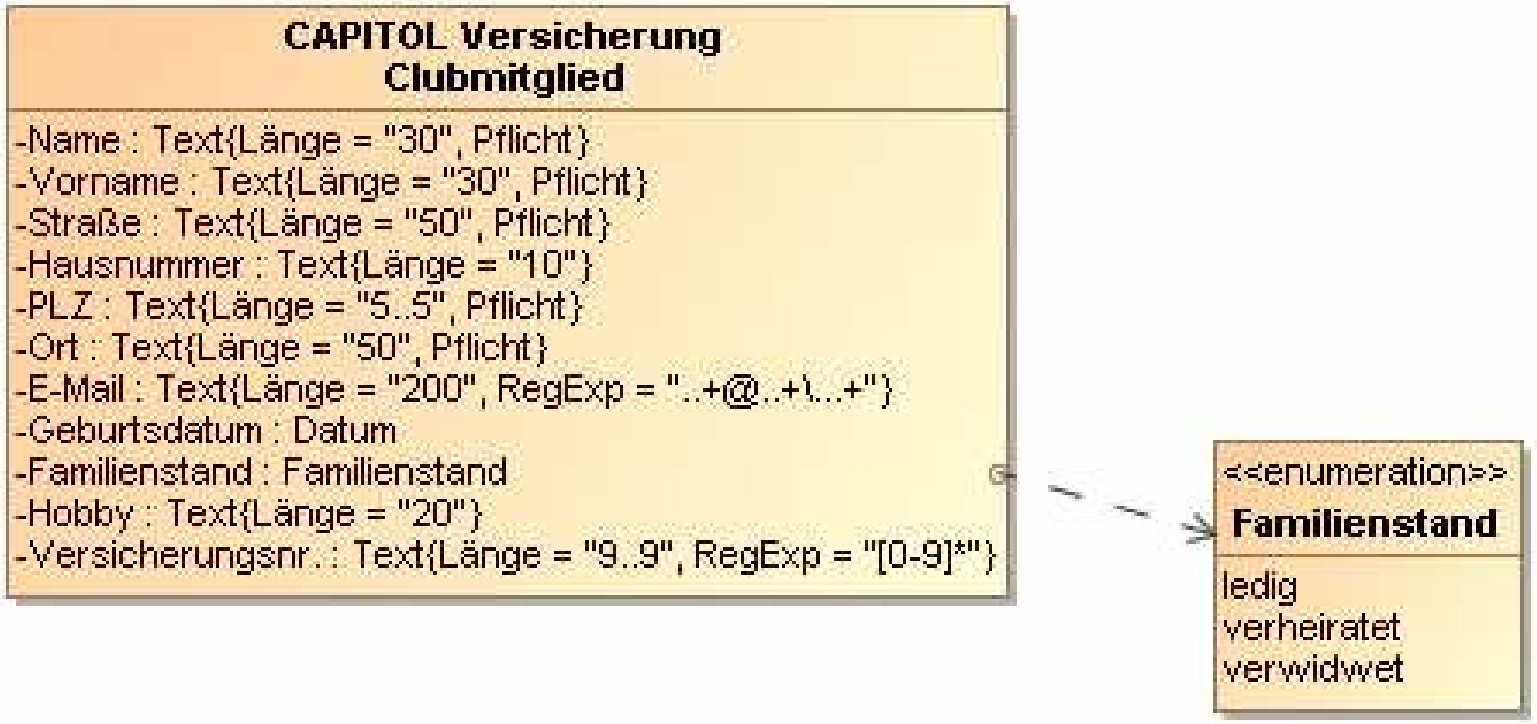
- Besonders komplizierte Bausteine sind tendenziell in Projekten unverstanden und unbeliebt

Noch ein Dilemma (für wahre OO-Experten!)

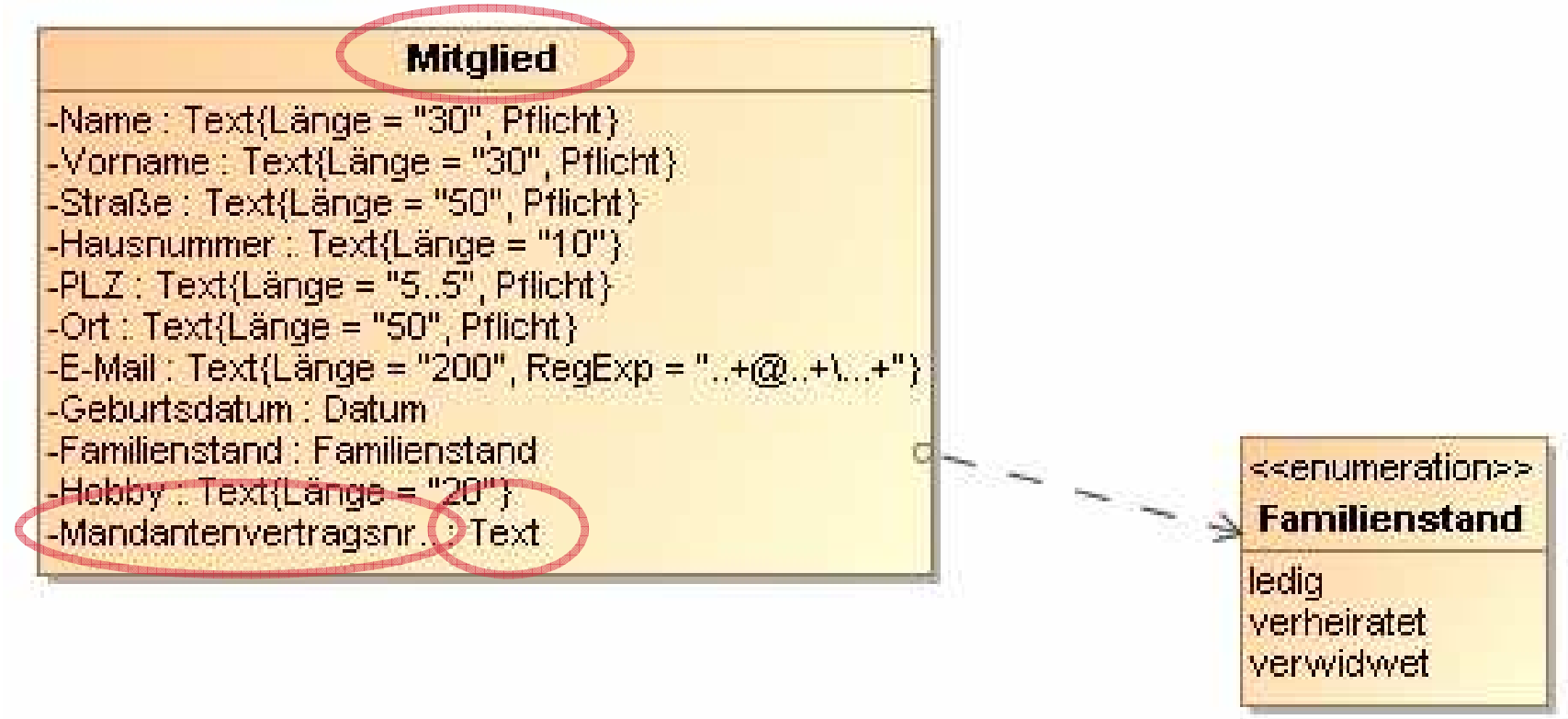
arvato services
BERTELSMANN

- Software wird besonders passend, wenn sie die Sprache der Fachwelt spricht (Grundgedanke des Domain-Driven Design)
- Für eine gemeinsame Sprache gilt:
  - Kein Entwurf auf Halde
  - Keine Abstraktionen über die Begriffswelt der Projektdomäne hinaus
- Das ist ein Widerspruch zum Vorgehen der Abstraktion zur Wiederverwendung

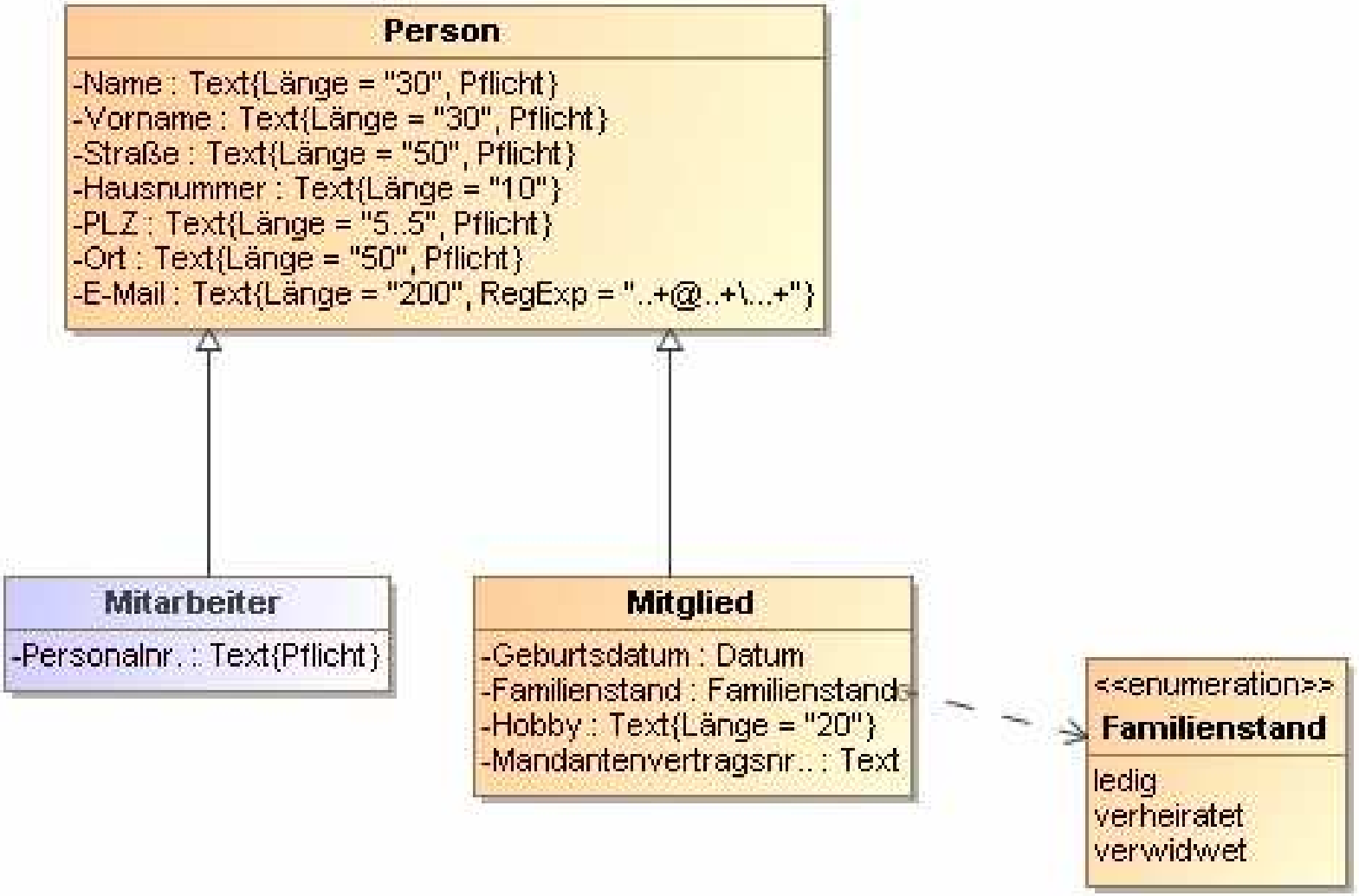
Beispiel: Bonusprogramm



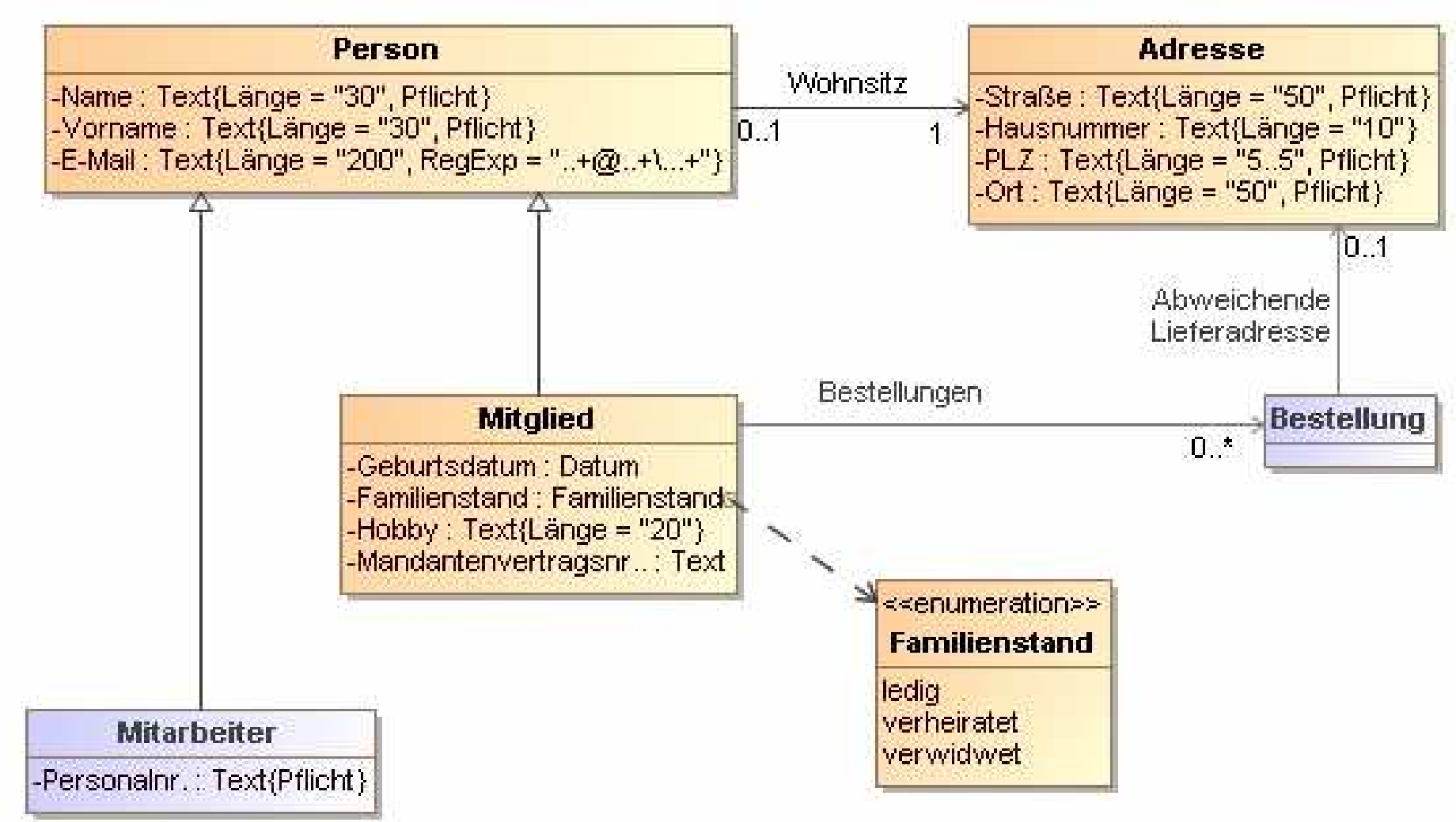
1. Abstraktion: Bezeichner



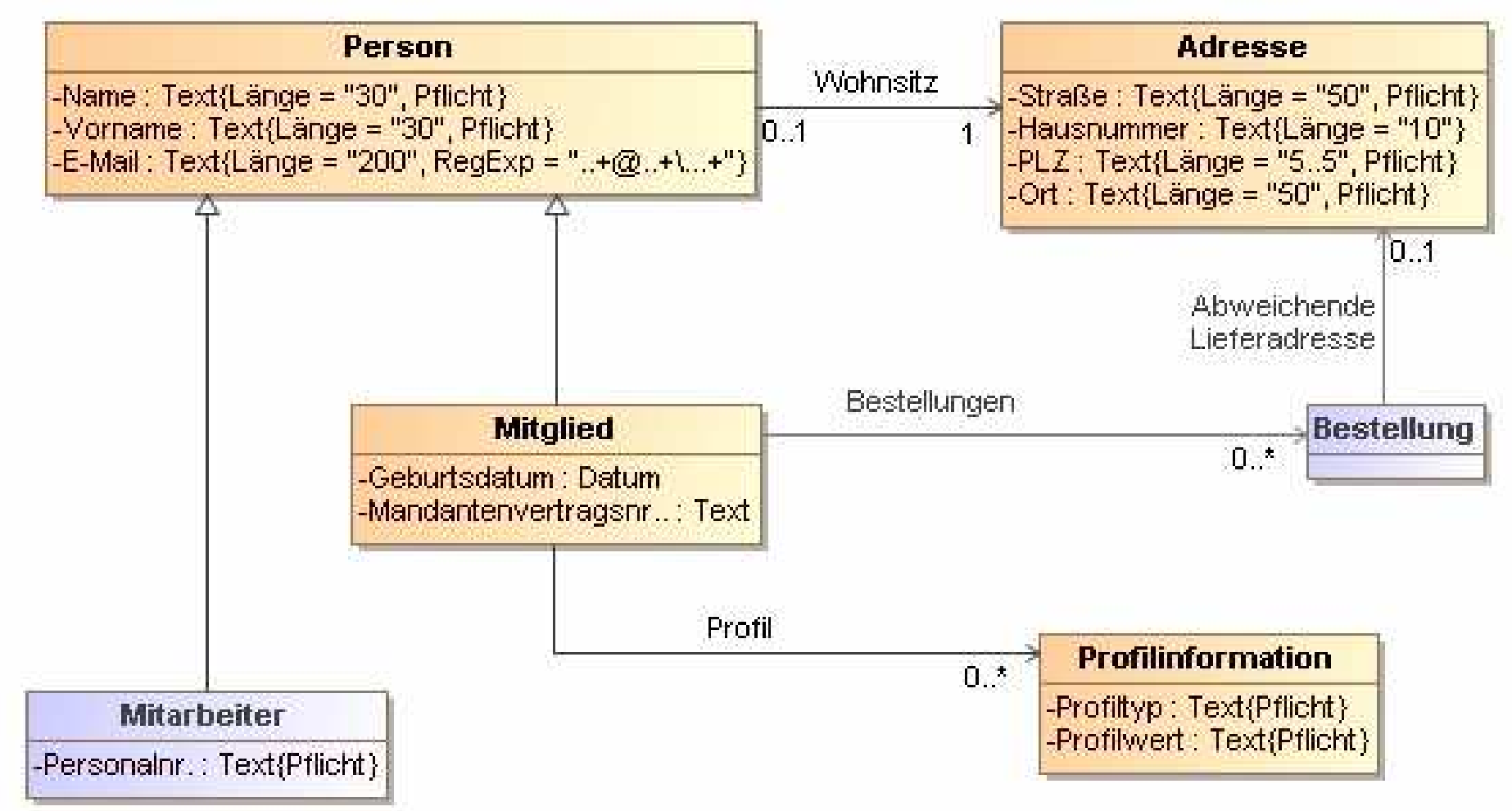
2. Abstraktion: Abspaltung Basisklasse „Person“



3. Abstraktion: Abspaltung „Adresse“

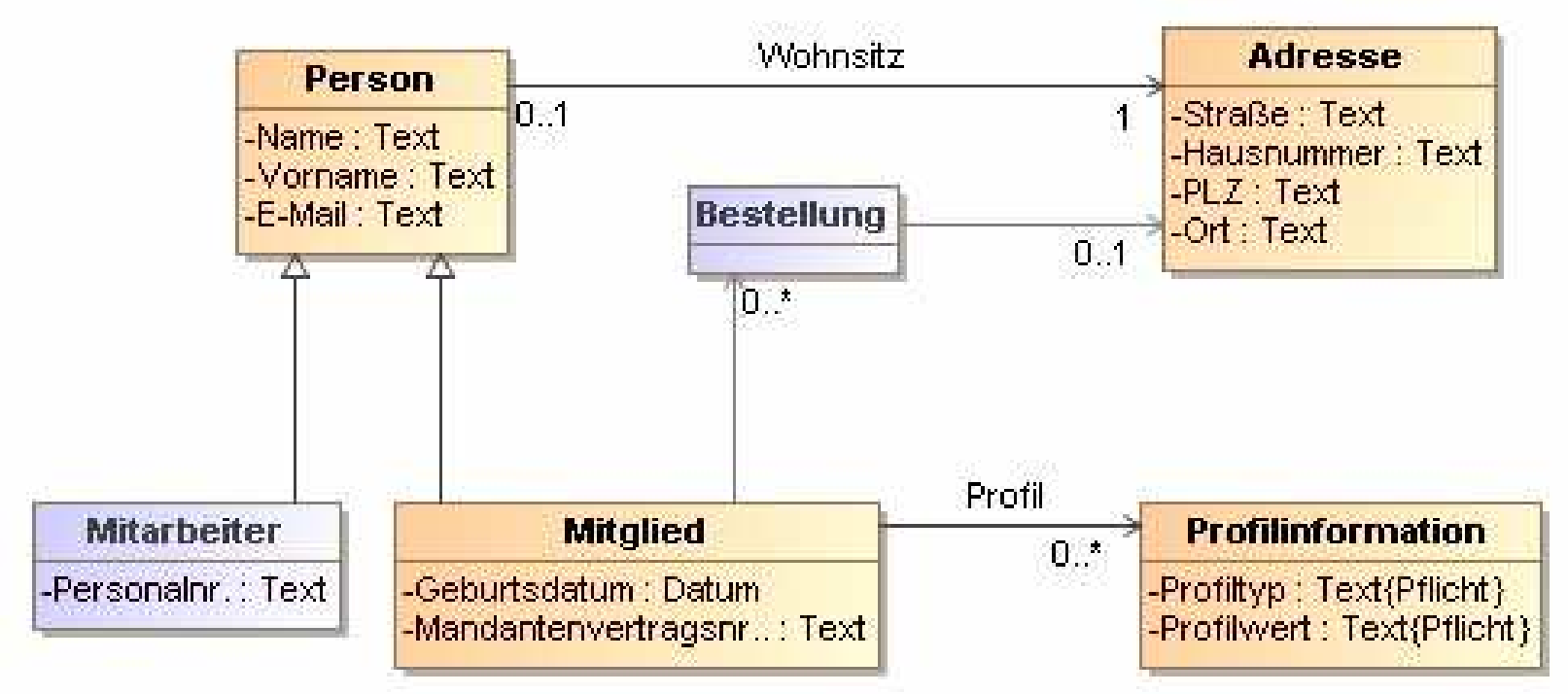


4. Abstraktion: Abspaltung „Profildaten“



Wollmilchsau: Auslagerung Attributconstraints

arvato services
BERTELSMANN



Microsoft Excel - regeln.xls

G9

	A	B	C	D	E	F	G
1	**Klasse**	**Attribut**	**Pflicht**	**Minimal**	**Maximal**	**Regulärer Ausdruck**	
2	Person	Name	x		30		
3	Person	Vorname	x		30		
4	Person	E-Mail			200	..+@..+\...+	
5	Adresse	Straße	x		50		
6	Mitglied	Mandantenvertragsnr.	x	9	9	[0-9]*	
7							

Vergleich

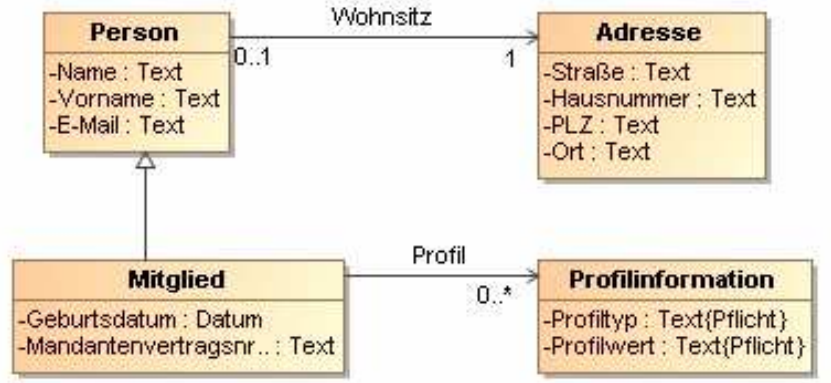
arvato services
BERTELSMANN

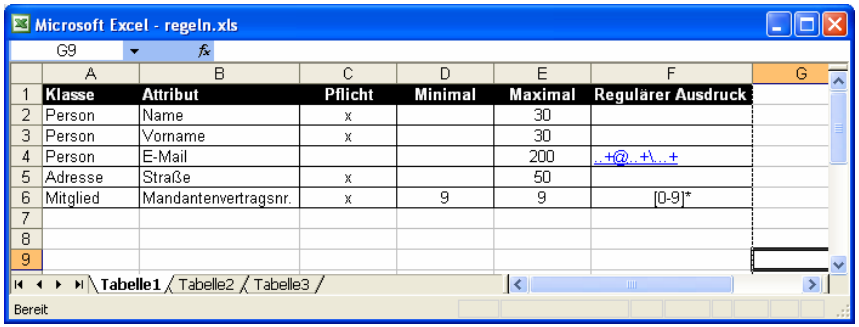
Angefordert

CAPITOL Versicherung
Clubmitglied

-Name : Text{Länge = "30", Pflicht}
-Vorname : Text{Länge = "30", Pflicht}
-Straße : Text{Länge = "50", Pflicht}
-Hausnummer : Text{Länge = "10"}
-PLZ : Text{Länge = "5..5", Pflicht}
-Ort : Text{Länge = "50", Pflicht}
-E-Mail : Text{Länge = "200", RegExp = "..+@..+\...+"}
-Geburtsdatum : Datum
-Familienstand : Familienstand
-Hobby : Text{Länge = "20"}
-Versicherungsnr. : Text{Länge = "9..9", RegExp = "[0-9]*"}

Geliefert

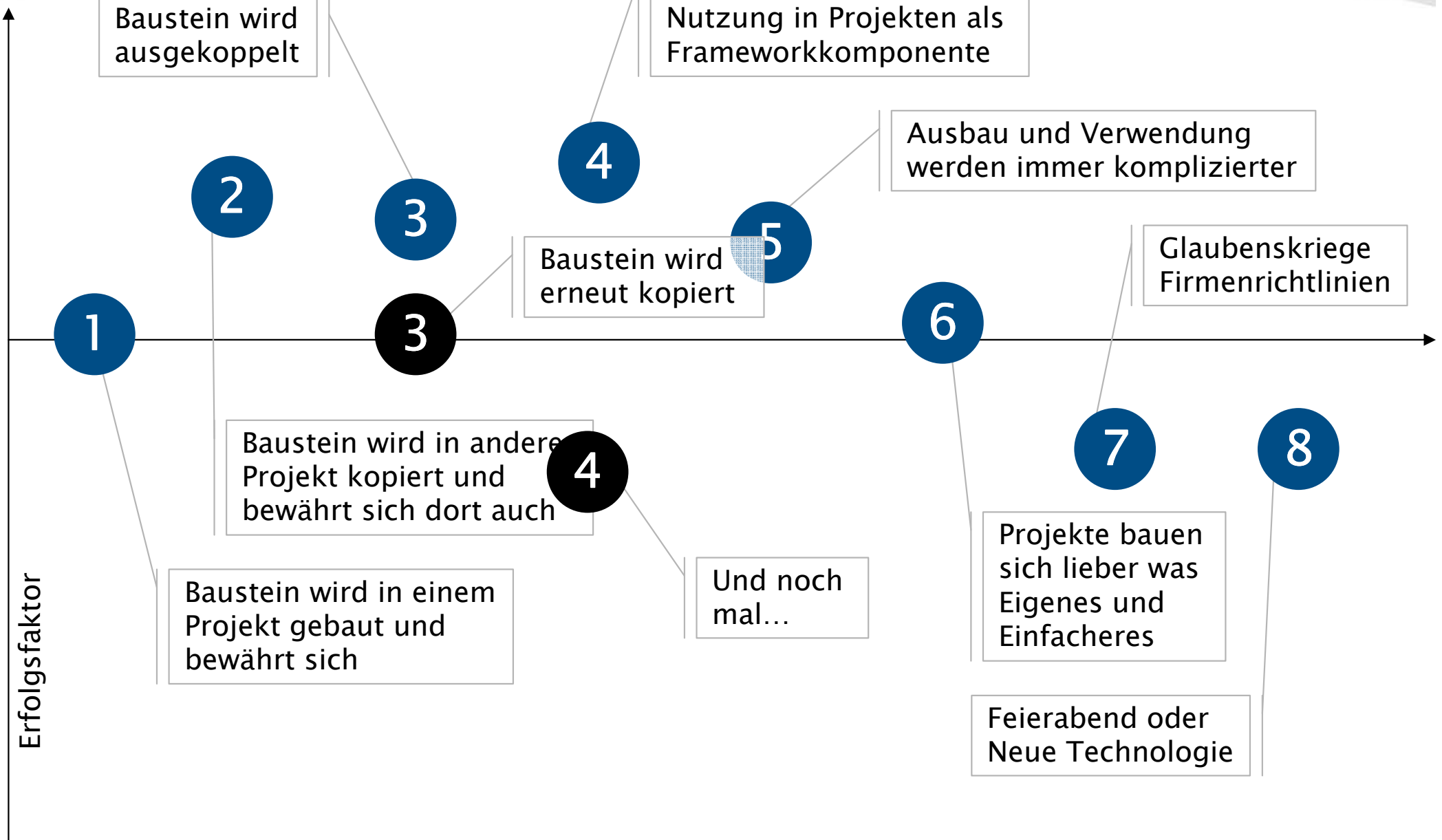




	A	B	C	D	E	F	G
1	Klasse	Attribut	Pflicht	Minimal	Maximal	Regulärer Ausdruck	
2	Person	Name	x		30		
3	Person	Vorname	x		30		
4	Person	E-Mail			200	..+@..+\...+	
5	Adresse	Straße	x		50		
6	Mitglied	Mandantenvertragsnr.	x	9	9	[0-9]*	
7							
8							
9							

Ein typischer Weg der Wiederverwertung

arvato services
BERTELSMANN



Ein paar Auswege

- Zukaufen statt selber machen
- Konzepte statt Code
- Bedingungen für selbst gemachte Wiederverwendung
- Ein alternativer Ansatz

Make or Buy

arvato services
BERTELSMANN

- Wiederverwendbare Bausteine selbst herstellen ist schwierig
- Für bestimmte Bereiche kann man sie zukaufen
- Über Open-Source-Projekte u.U. sehr günstig
- Problem: fremde Bausteine sind tendenziell unpassende Wollmilchsäue (siehe meine Präsentation „Von Mainstreammonstern und Hyperhypes")

Geeignete Bereiche für Zukauf

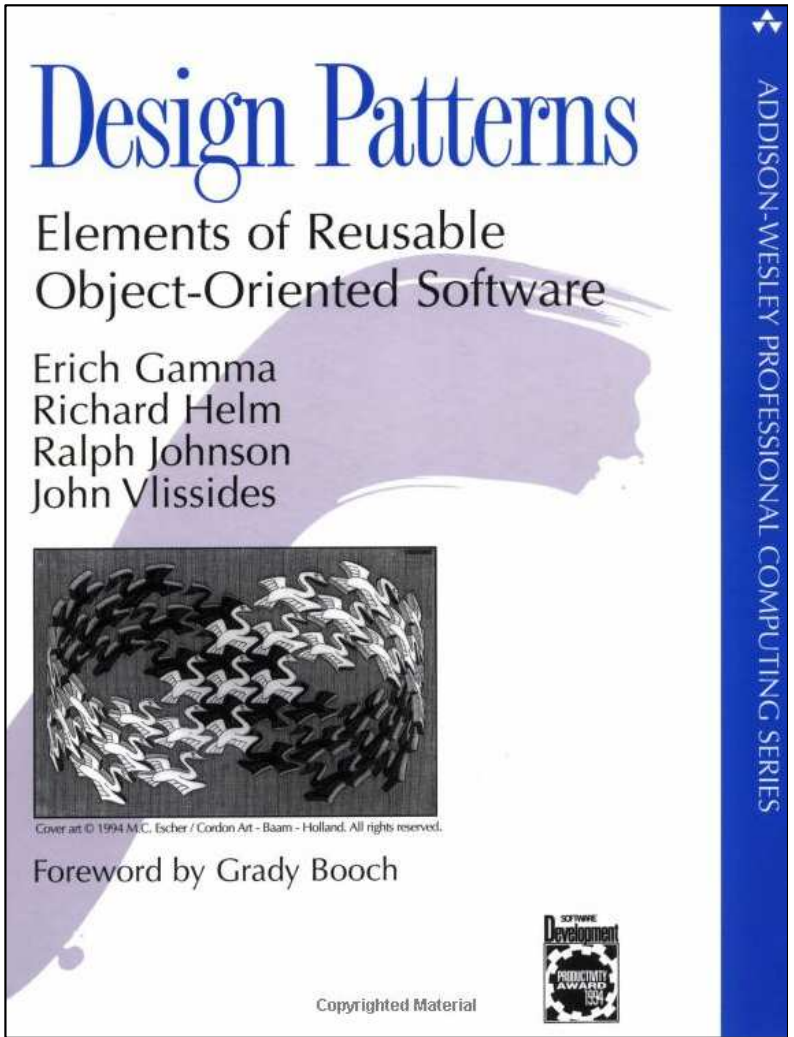
- Kein Bereich eigener Kernkompetenz
- Kein erfolgskritischer Faktor
- Standardisierter Bereiche
- Geringe Änderungshäufigkeit

Konzepte statt Code

arvato services
BERTELSMANN

- Es gibt das GoF-Entwurfmusterbuch...
- Aber keine GoF-Bibliothek

- Manche Ideen sind wertvoller als eine Implementierung



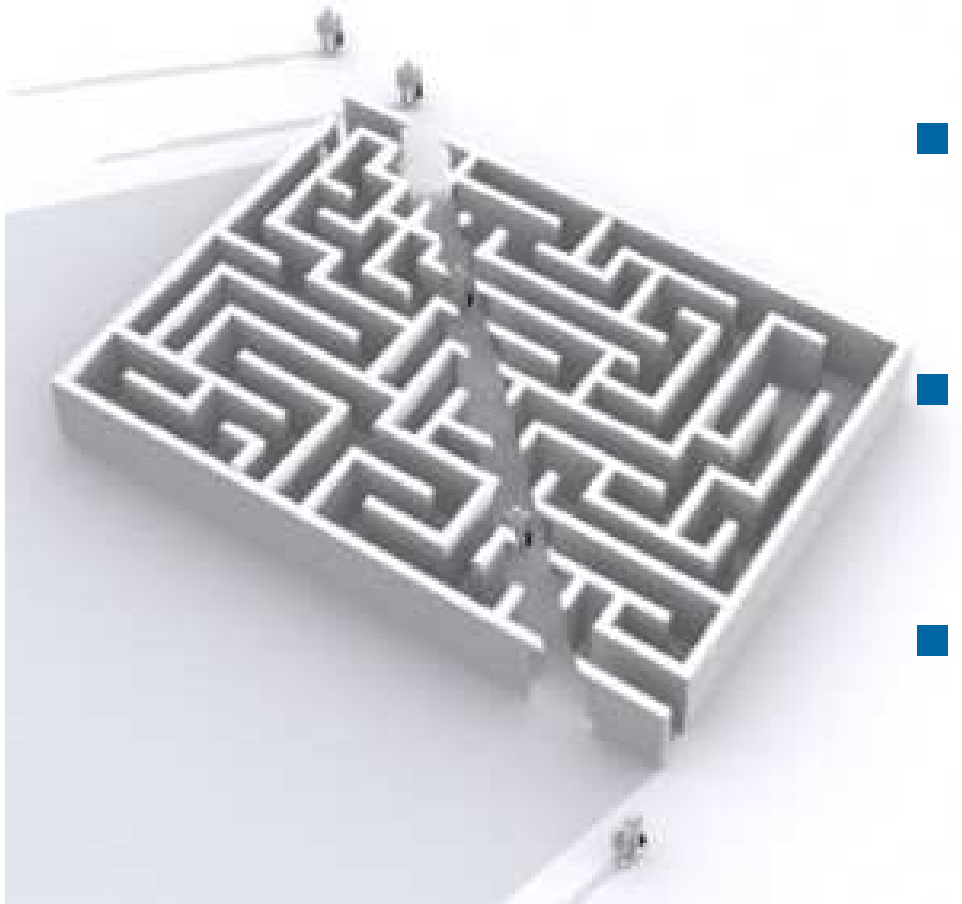
Bedingungen für selbst gemachten Reuse

arvato services
BERTELSMANN

- Mindestkomplexität des Bausteins
- Seeeeehr gute Entwickler und starke Führungspersönlichkeiten
- Entwickler müssen die Designprinzipien verinnerlicht haben
- Kontinuität in der Weiterentwicklung und im Team
- Sukzessiver Ausbau statt großer Wurf auf grüner Wiese
- Auf Basis Pattern-Mining
  (*ein* Projekt liefert oft noch kein Pattern)
- Framework-Entwickler müssen auch
  in Anwendungsprojekten arbeiten

Alternativer Ansatz für fachliche Bausteine

arvato services
BERTELSMANN

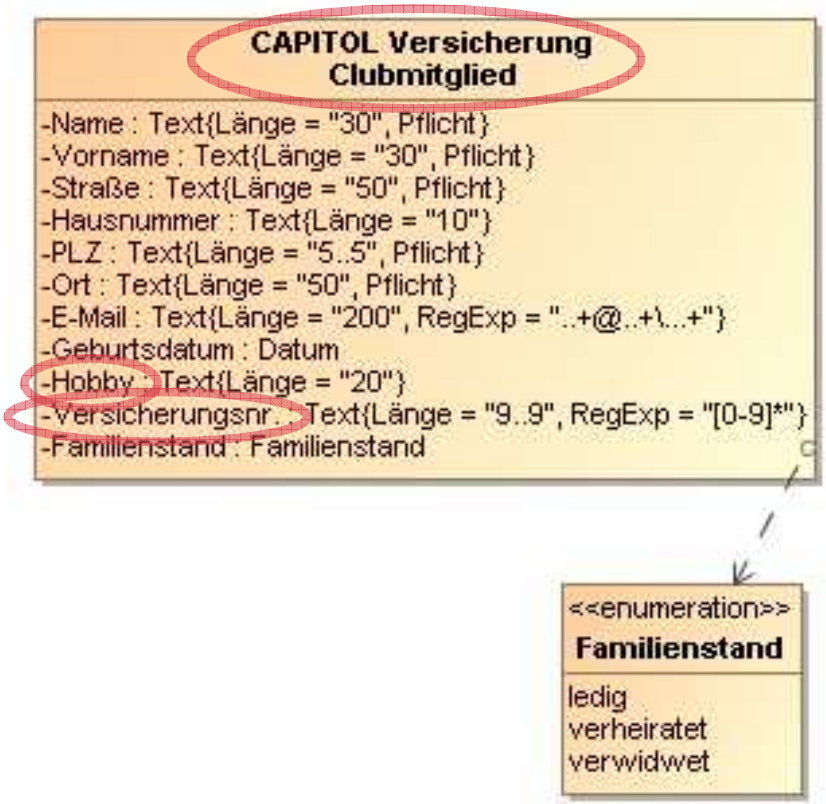


- Verstärkung der technischen Infrastruktur
  - Generatoren basierend auf Modellen
  - Generische Ansätze
  - Regelwerke
- Orientierung am Normalfall statt Maximalfall
- Projektspezifika entfernen statt abstrahieren
- Wiederverwendung als Modell und Sourcecode

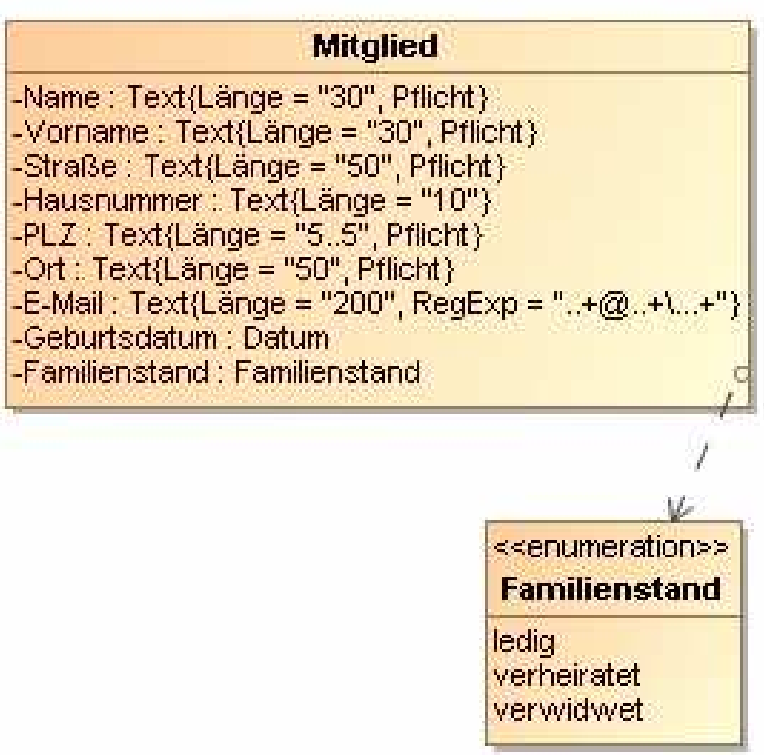
Beispiel: Bonusprogramm

arvato services
BERTELSMANN

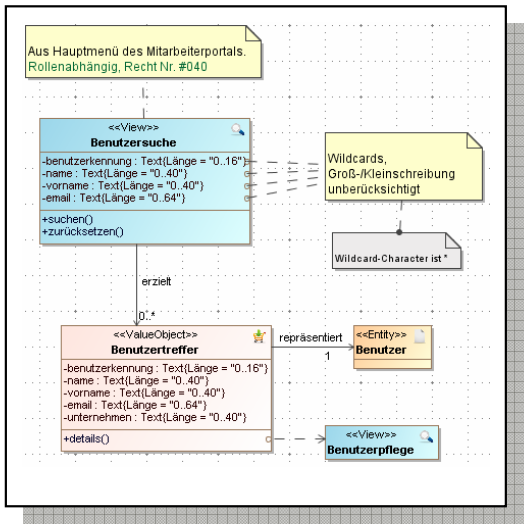
Projektspezifisch



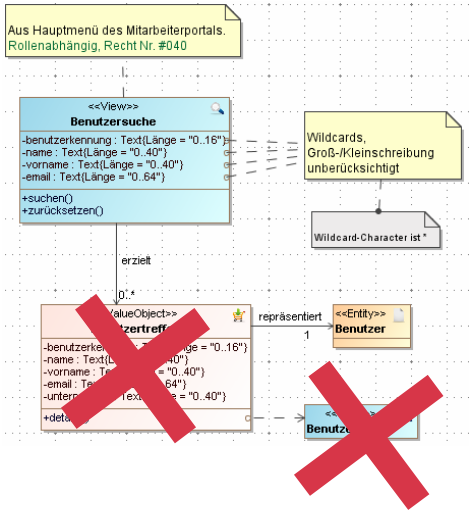
Ausgekoppelt



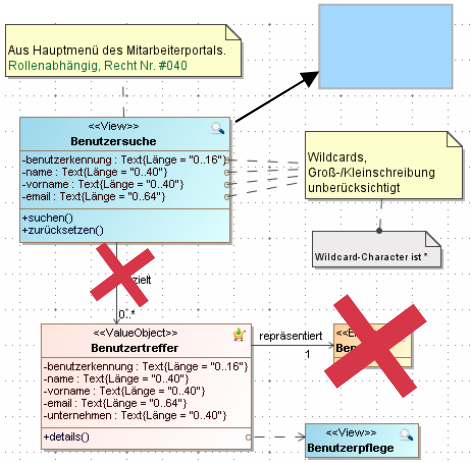
Und noch einen Schritt weiter...



Projekt X



Projekt Y



- Bausteine beispielhaft zu Applikation verdrahtet
- Patterns für lose Kopplung (z.B. Dependency Inversion)
- Framework ist Blaupause inkl. Buildskripte, Tests etc.
- Lose Kopplung erlaubt leichte Anpassung

Ist das nicht Copy-Paste ?!

Ja klar – macht aber nix zwischen unabhängig budgetierten Projekten

Für und Wieder

arvato services
BERTELSMANN

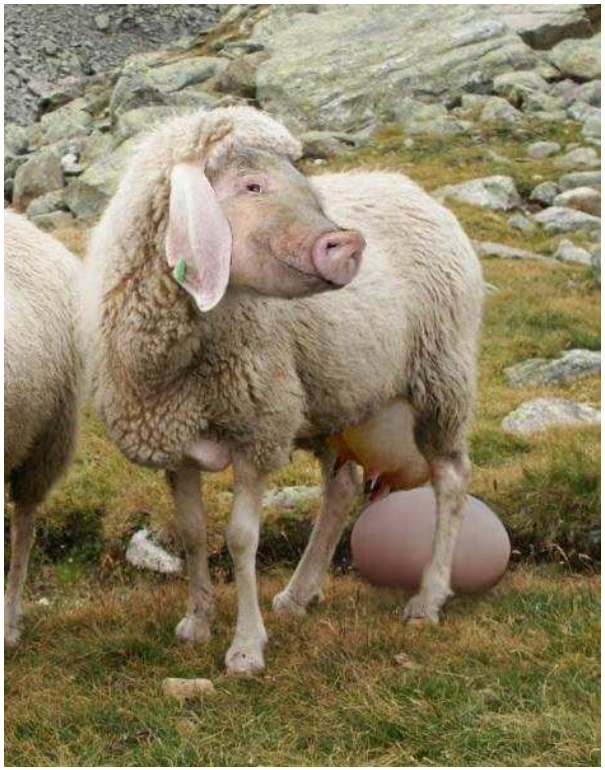
PRO

- Frameworkstrukturen fast wie Projektstrukturen
- Strukturen als Vorbild für Projektarbeit nutzbar
- Es können nicht nur herausragende Entwickler mitarbeiten
- Bausteine direkt als Applikation demonstrierbar
- Beliebige Flexibilität
- Alle Begriffe änderbar auf Projektsprache
  (allerdings trotzdem aufwendig)
- Projekte erhalten auch Buildskripte, Verzeichnisstruktur,
  Unittests usw.

CONTRA

- Anwendungsprojekt ist abgekoppelt von zukünftigen
  Framework-Verbesserungen
- Die Bausteine sind keine eierlegenden Wollmilchsäue

Was ich noch zu sagen hätte...



- Wiederverwendung ist machbar
- Bei Wollmilchsäuen ist die Haltung teurer als der Profit
- Halbherzig geht nicht
- Es lohnt sich zu schauen wo es sich lohnt :-)

... und bei Fragen, fragen!