fhwedel
UNIVERSITY OF APPLIED SCIENCES

Service-orientierte Software-Architekturen

Prof. Dr. Ulrich Hoffmann
FH Wedel

SOA-Metriken

Überblick

Software-Metriken

- Was sind das?
- Verschiedene Arten von Metriken
- Komplexitätsmetriken
- Objekt-orientierte Metriken
- Struktur-Metriken
- SOA?

Software-Metriken

- Was können wir messen?
- Was für Arten von Software-Metriken gibt es?
  - **P**rodukt-Metriken
  - **P**rozess-Metriken und
  - **P**rojekt-Metriken

P P P

Produkt-Metriken

- Charakteristik des Produkts, z.B.
  - Größe
  - Komplexität
  - besondere Produkteigenschaften
  - Geschwindigkeit
  - Qualitäts-/Sicherheits-Ebene

Prozess-Metriken

- Verbesserung des Entwicklungsprozesses, z.B. durch
  - Effektivität der Fehlerbehebung währen der Entwicklung
  - Art der durch Test aufgedeckten Fehler und ihre zeitliche Häufigkeitsverteilung
  - Antwortzeit für Fehlerkorrekturen

Projekt-Metriken

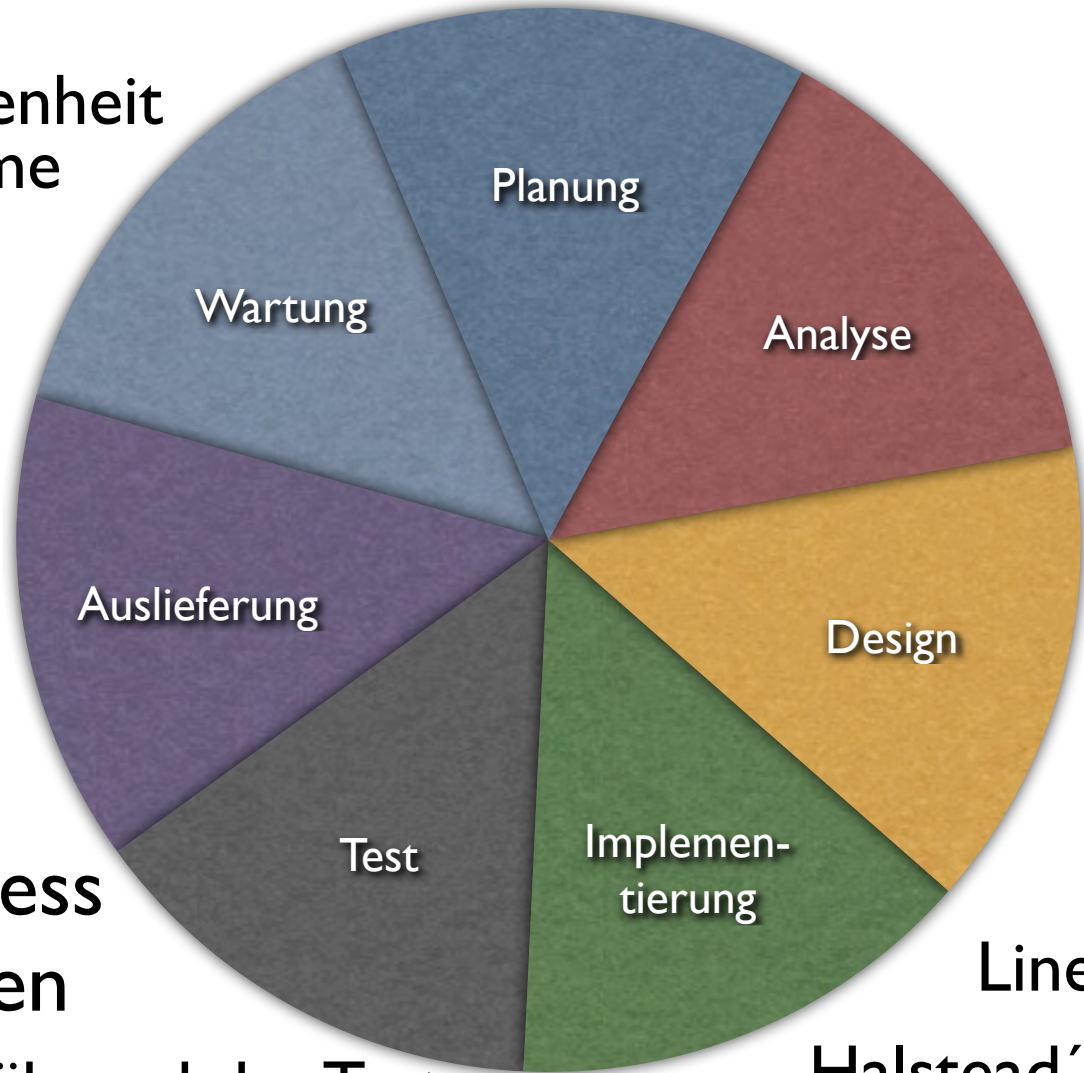
- Merkmale des Projekts, z.B.
  - Anzahl der Software-Entwickler
  - Mitarbeiterentwicklung während des Projekt-Lebenszyklus
  - Projektkosten
  - Terminplan / Termintreue
  - Produktivität

Software-Metriken

Produkt-, **P**rozess- und **P**rojekt-Metriken

Kundenzufriedenheit
Kundenprobleme
Fehlerdichte

End-Produkt-Metriken



Komplexitäts-Metriken

zyklomatische Komplexität

Lines of Code

Halstead´s Software Science

Struktur-Metriken

Im-Prozess Metriken

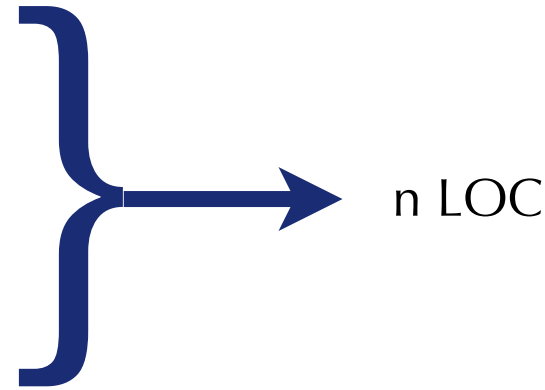
Fehlerdichte während des Tests
Fehlerauftrittsmuster während des Tests
Effektivität der Fehlerbeseitigung

Produktmetrik - Größe der Service-Implementierung

- Lines of Code (LOC)
  - Anzahl der Programmzeilen eines Services
  - Programmiersprachen-abhängig
  - jedoch typisch 0,4 - 2 Fehler je 1000 LOC

```
class orderServiceImpl {
   public String placeOrder(String customerID,
                            OrderList items)
   { ... }
   public OrderInfo getOrderStatus(String orderNr)
   { ... }
   ...
}
```

Service-Implementierung z.B. in Java



Zur Bewertung von Kennzahlen

- **Welchen Stellenwert haben Software-Metrik-Kennzahlen?**

- *Absolut*wert der Kennzahlen ist wenig aussagekräftig.
  - Hohe oder niedrige Werte sind per se nicht schlecht oder gut.
  - Es gibt häufig einen guten Grund, warum Services so sind wie sie sind.

- *Relativ*wert ist bedeutsam
  im Vergleich zu
  - zu öffentlicht bekannten Werten ähnlicher Services
  - ähnlichen Services in der eigenen SOA
  - zu früheren Versionen des gleichen Services

- Starke Abweichungen von Vergleichswerten können auf Probleme hinweisen. Dann muss mit gesundem Menschenverstand untersucht werden.

Komplexitätsmetriken

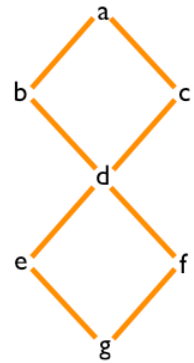
- Halstead Software Science (1977)

- Einfache Metriken:
  - n_1=Anzahl der verschiedenen Operatoren eines Programms
  - n_2=Anzahl der unterschiedlichen Operanden eines Programms
  - N_1=Anzahl der Operatorvorkommen
  - N_2=Anzahl der Operandenvorkommen

- Abgeleitete Metriken:
  - Vokabular (n) = n_1+n_2
  - Länge (N) = N_1+N_2
  - Volumen (V) = $N \log_2(n)$
  - Difficulty (D) = V / V* (V* Volumen einer *kleinsten* Lösung)

Produktmetrik - Komplexität

Wie komplex ist ein Service?



McCabes zyklomatische Komplexität (1976)

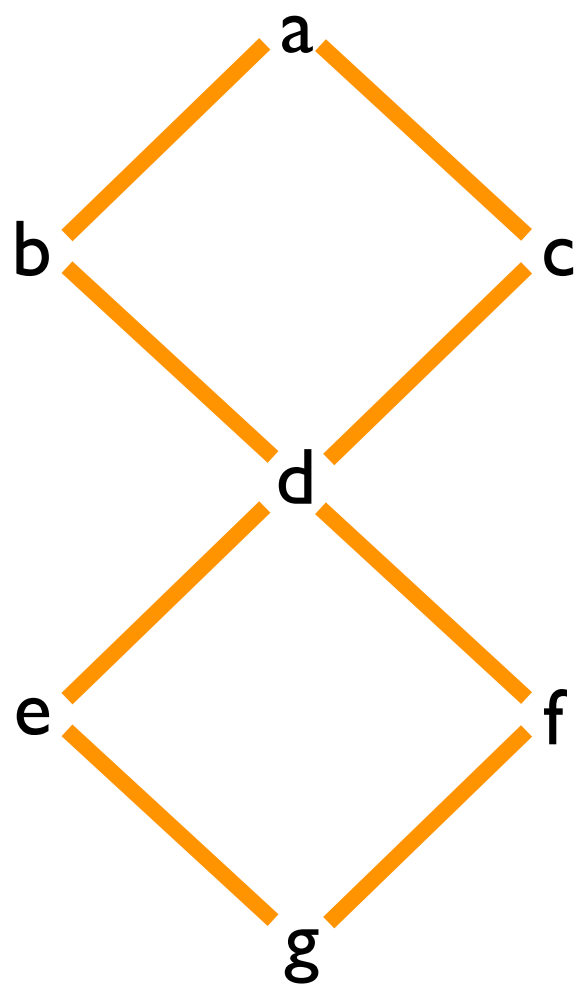
- Service wird als Kontrollflussgraph aufgefasst.
  - Wie groß ist die Anzahl unabhängiger Flüsse durch den Service?
  - M=e-n+2p
    - e: Anzahl der Kanten im Flussgraph
    - n: Anzahl der Knoten im Flussgraph
    - p: Anzahl der unverbundenen Teile des Flussgraphen
- Für eine Funktion: Anzahl binärere Verzweigungen+1

Produktmetrik - Komplexität

Wie komplex ist ein Service?

McCabes zyklomatische Komplexität (1976)

```
IF a THEN

b ELSE c;



IF d THEN

e ELSE f;

g
```



e = 8 Kanten

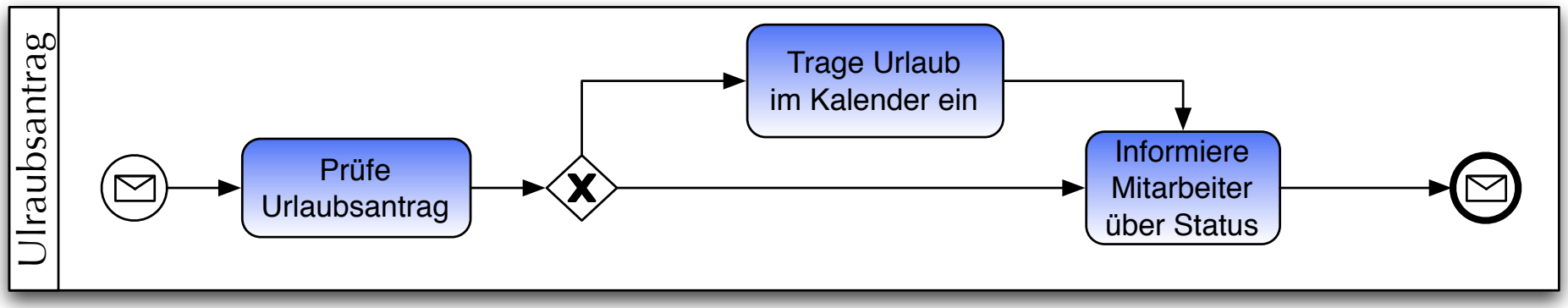
n = 7 Knoten

p = 1 zusammenhängender Graph

M= e-n+2p = 8-7+2*1 = 3

Produktmetrik - Komplexität

Beispiel: BPMN-Geschäftprozessdiagramme



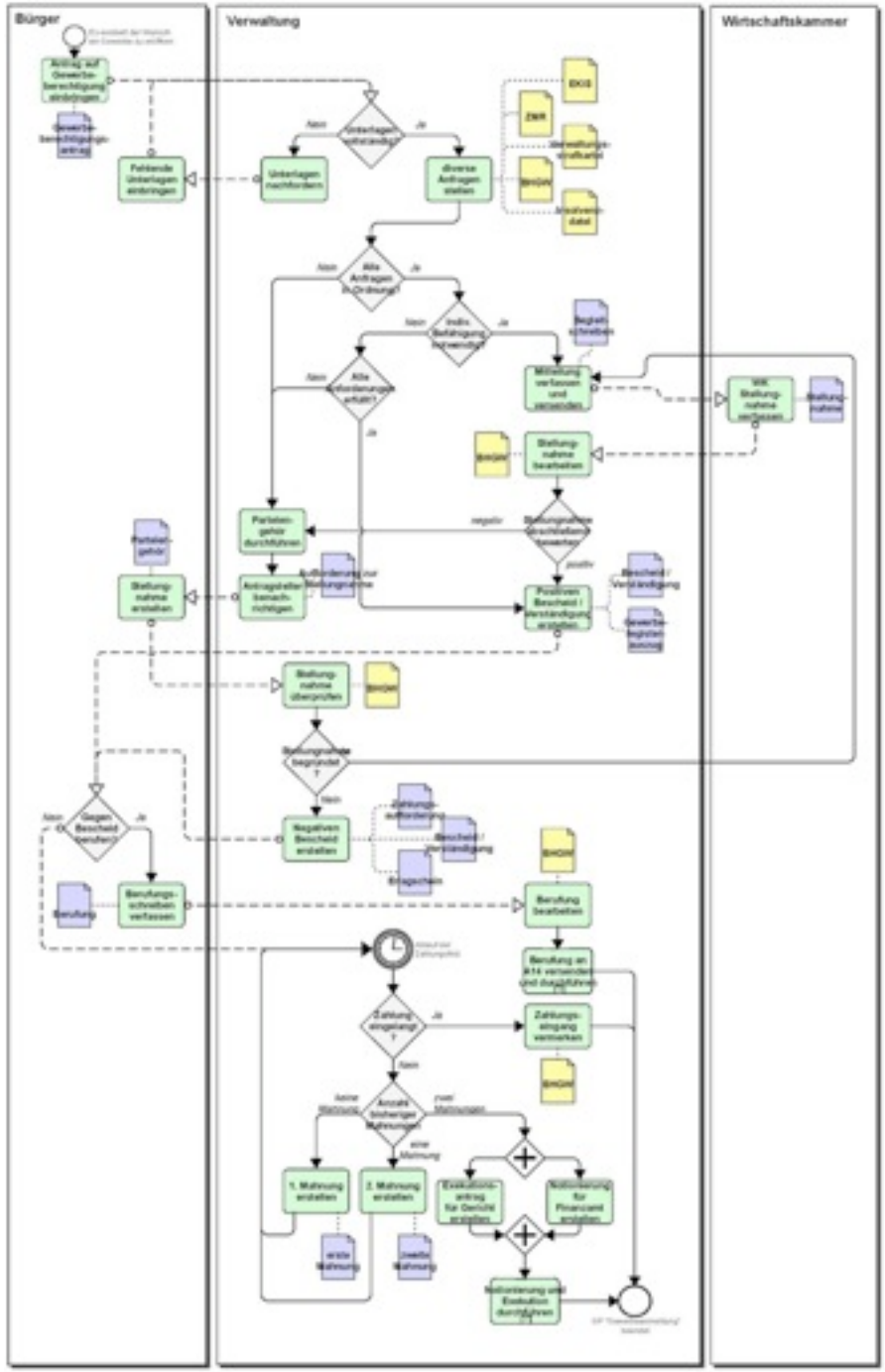
zyklomatische Komplexität

e = 5 Kanten

n = 5 Knoten

p = 1 zusammenhängender Graph

$$M = e - n + 2p = 5 - 5 + 2*1 = 2$$



Grafik nach Prof. Dr. Margrit Falck, „Modellieren mit BPMN“

BPMN-Geschäftsprozess

Gewerbeanmeldung

zyklomatische Komplexität

e = 52 Kanten

n = 34 Knoten

p = 1 zusammenhängender Graph

$$M = e - n + 2p = 52 - 34 + 2*1 = 20$$

Produktmetrik - Service-Kopplung

- Maßstab für die Interaktion zwischen Services
- Ziel: geringe (= lose) Kopplung
- Kopplung zwischen Services findet auf viele Arten statt:

- zu vermeidende Kopplungen
  - **Inhaltskopplung**: (direkte Verwendung der inneren Struktur)
  - **Common-Kopplung** (gemeinsame Daten)
- normale, zu erwartende Kopplung (graduell abzuschwächen)
  - **Kontrollkopplung** (direkte Beeinflussung des Kontrollflusses)
  - **Datenstrukturkopplung** (Austausch gemeinsamer Datenstrukturen)
  - **Datenkopplung** (Ein Service sendet Daten an einen anderen)
  - **Schnittstellenkopplung** (Abhängigkeit von anderen Services)

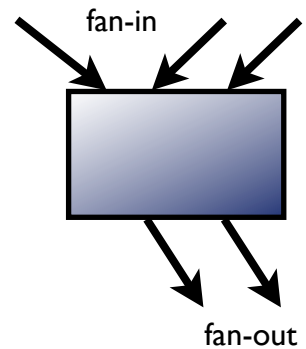
Produktmetrik - Service-Kopplung messen

Kopplung eines Services: **1,0 starke Kopplung**, **0,7 lose Kopplung**

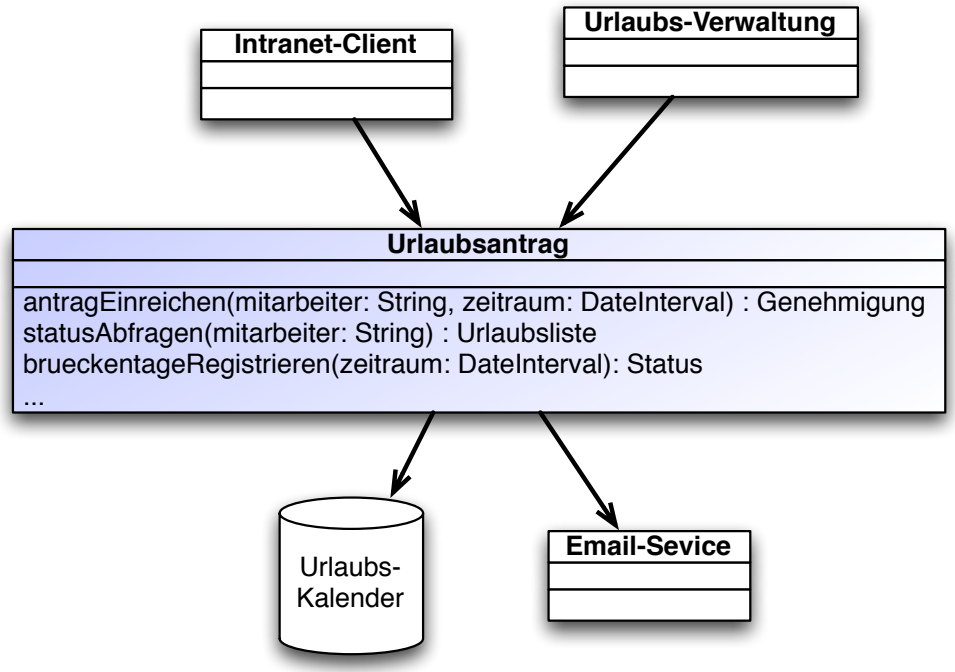
Kopplung $$C = 1 - \frac{1}{d_i + 2 \cdot c_i + d_o + 2 \cdot c_o + g_d + 2 \cdot g_c + w + r}$$

wobei:

- Daten/Kontrollkopplung
  - d_i, c_i Eingangs-Parameter des Services für Daten, bzw. zur Kontrolle
  - d_o, c_o Ausgangs-Parameter des Services für Daten, bzw. zur Kontrolle
- Common-Kopplung
  - g_d, g_c globale Variablen/Ressourcen für Daten, bzw. zur Kontrolle
- Schnittstellenkopplung
  - r die Anzahl der aufrufenden Services (fan-in) und
  - w die Anzahl der aufgerufenen Services (fan-out) ist.



Produktmetrik - Service-Kopplung messen



- Eingangsparameter: $d_i = 4$, $c_i = 0$
- Ausgangsparameter: $d_o = 3$, $c_o = 0$
- Globale Ressourcen: $g_d = 1$, $g_c = 0$
- Schnittstellen: $w = 1$, $r = 2$

$$C = 1 - \frac{1}{4 + 2 \cdot 0 + 3 + 2 \cdot 0 + 1 + 2 \cdot 0 + 1 + 2} = 1 - \frac{1}{11} \approx 0,91$$

SOA-Metriken - wo misst man?

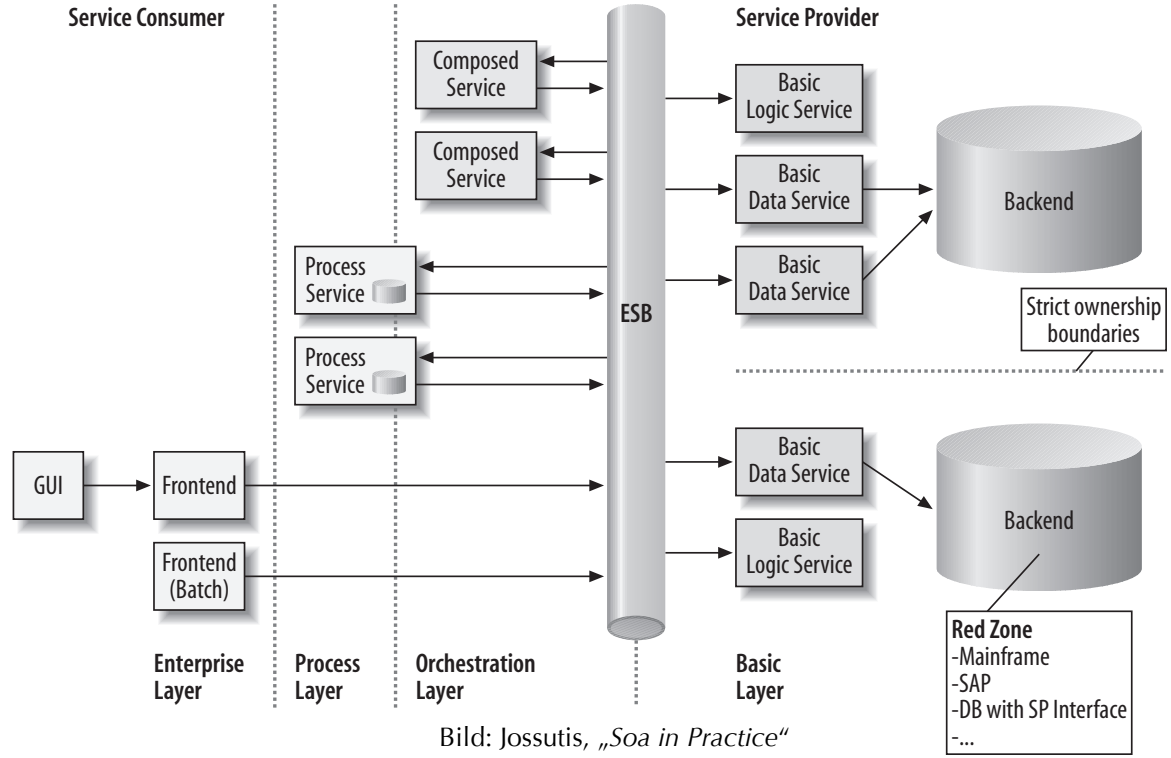


Bild: Jossutis, *„Soa in Practice“*

- statische Programmanalyse
  - Quellcode, Schnittstellenbeschreibung, Deployment-Information
- dynamisches Messen beispielsweise im ESB
  - Welcher Service ruft welchen auf und mit welchen Operationen

Objekt-orientierte Metriken

fhwedel
UNIVERSITY OF APPLIED SCIENCES

Lorenz 1993

- Mittlere Methodengröße (LOC)
- Mittlere Anzahl Methoden pro Klasse
- Mittlere Anzahl von Instanzvariablen pro Klasse
- Tiefe des Vererbungsbaums
- Anzahl der Teilsysteme
- Anzahl der Klassen pro Subsystem
- Verwendung von Instanzvariablen (Gruppen von Methoden)
- Anzahl der Kommentarzeilen pro Methode
- Anzahl der Problem Reports pro Klasse
- Anzahl der Wiederverwendungen pro Klasse
- Anzahl der während der Entwicklung verworfenen Klassen und Methode (inkrementell vs. iterativ)

Objekt-orientierte Metriken

fhwedel
UNIVERSITY OF APPLIED SCIENCES

Chidamber und Kemerer 1994 (CK-Metriken)

- Gewichtete Methoden pro Klasse (Summe der Komplexitäten der Methoden)
- Tiefe des Verbungsbaums
- Anzahl der direkten Unterklassen einer Klasse
- Kopplung zwischen Objekt-Klassen (Objektverwendung)
- Anzahl der Antworten einer Klasse auf eine Nachricht (Polymorphismus)
- Abwesenheit von Kohäsion der Methoden (Anzahl der getrennten Verwendung von Instanzvariablen)

Diskussion

- Was wollen wir bei SOA überhaupt messen?
  - Die Komplexität einer SOA-Landschaft? Oder eines Services?
  - Die Verbesserung im Entwicklungsaufwand gegenüber klassischen Ansätzen? Wann?
  - Die Stabilität oder Änderungsfreundlichkeit des Produkts?
  - Den geringeren Änderungsaufwand bei neuen Anforderungen?
  - Oder?

Fazit

- Metriken des klassischen Software-Engineerings lassen sich auf Services und SOA anpassen.
- Werkzeugunterstützung ist wünschenswert, aber (noch) nicht vorhanden.
- Erfahrung über realistische Kenngrößen für SOA fehlen (noch).