

Seminar – Winter Semester 08/09

Software Produkt Linien

Vortragsthema:

Wirtschaftliche Aspekte von SPL

Agenda

- 1 Vorteile
 - 1.1 Time to Market
 - 1.2 Qualität
 - 1.3 Produktivität und Kosten
 - 1.4 Skalierbarkeit
- 2 Nachteile
- 3 QCOMPLIMO
 - 3.1 COCOMO 2
 - 3.1.1 Application Composition Model
 - 3.1.2 Early Design Model
 - 3.1.3 Post-Architecture Model
 - 3.2 Relative Cost of Writing for Reuse
 - 3.3 Relative Cost for Reuse
 - 3.4 Gesamtkosten
 - 3.5 Vergleich

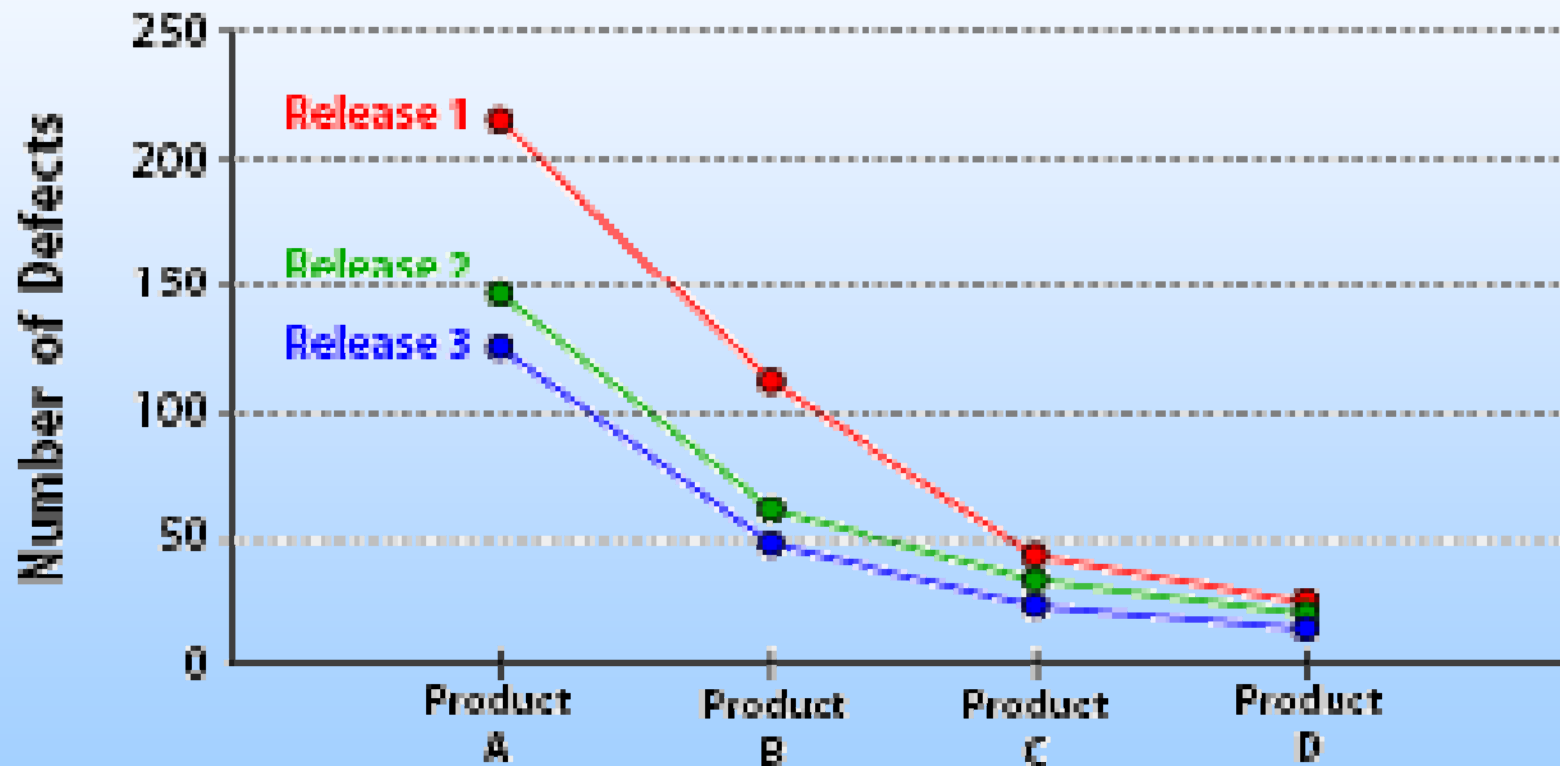
Time-to-Market Vorteil

- Produkte einer SPL sind schneller marktfähig
 - kürzere Zeit für Einnahmen
 - kritische Zeitfenster für Markteintritt
 - auf mehreren Märkten gleichzeitig → mehr Einnahmen
 - Bessere Überlebenschance
- Time-to-Market Verkürzung um Faktor 2 bis 50 (Praxiswerte)
- Time-to-Market Verkürzung durch Delta-Engineering

Qualitätsvorteile von SPL

- Auf zwei Wegen messbar
- Anpassung auf Kundenbedürfnisse → Mass Customization
 - Anzahl der Anpassungsmöglichkeiten = Maß für Qualität
- Festgestellte Fehler in jedem Produkt = Vorteil für ganze SPL
- SPL ist gutes Mittel zur Fehlerreduktion (bis zu 96%)

Fehlerraten in SPL



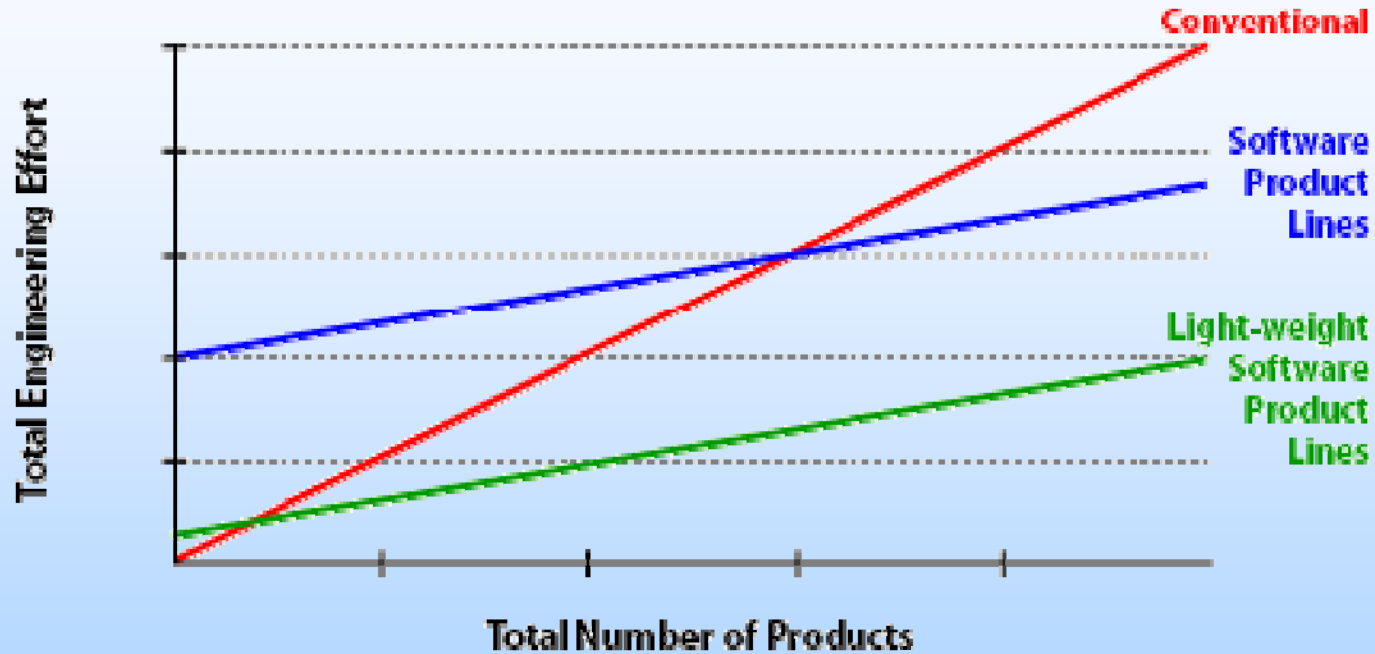
Vorteile guter Qualität

- Zufriedene Kunden = regelmäßige Einnahmen + Empfehlungen
- Weniger Fehler = weniger Overhead im Lebenszyklus
 - weniger Support-Personal
 - kleinere Test-Teams
 - Entwickler können neue Produkte entwickeln
- Qualitätsvorteile durch „gemeinsame“ Fehlerreduktion
→ hochwertige Assets entstehen

Produktivität und Kostenvorteile

- SPL-Techniken erhöhen die Produktivität durch eine Reduzierung des Aufwands
- Weniger Kosten für Entwicklung, Implementierung und Wartung
- Produktivitätssteigerungen ums Doppelte oder Dreifache
- Lohnkosten überwiegen die Entwicklungskosten

Produktivitätsvergleich



Rot = konventionelle Produktivitätslinie

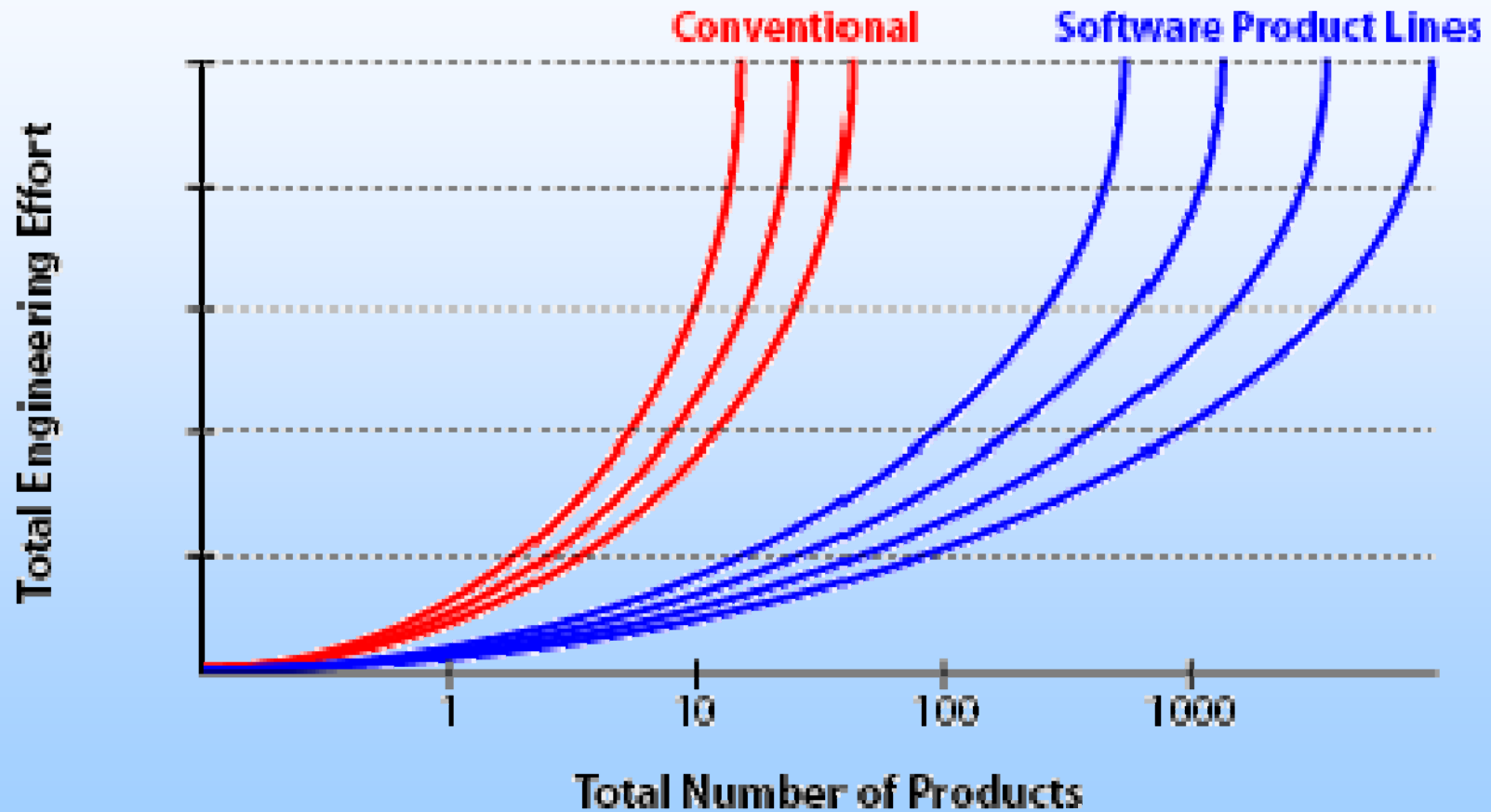
Blau = erste SPL's

Grün = neue SPL's

Produktivität und Kostenvorteile

- Produktivität und Kosteneinsparung durch Wiederverwendung
- Effizientes managen von Unterschieden innerhalb einer SPL
- Vorteile höhere Produktivität:
- Niedrigere Preise → steigende Verkaufszahlen, strategischer Wettbewerbsvorteil
- Kein Outsourcing in „Billiglohnländer“

Skalierbarkeit



Skalierbarkeit

- Skalierbarkeit um optimale Produktmenge anbieten zu können
- Steigt Komplexität („Linksverschiebung“) → Aufwand exponentiell
- Größenordnungen der Produkte besser anpassbar als bei konventionellen Entwicklungstechniken
- Skalierbarkeit = Wettbewerbsvorteile:
 - Einstieg in mehrere Märkte mit zugeschnittenen Produkten
 - Höhere Produktmengen innerhalb eines bestimmten Marktes
 - Größere Produktvielfalt und zielgerichtete Produkte als die Wettbewerber

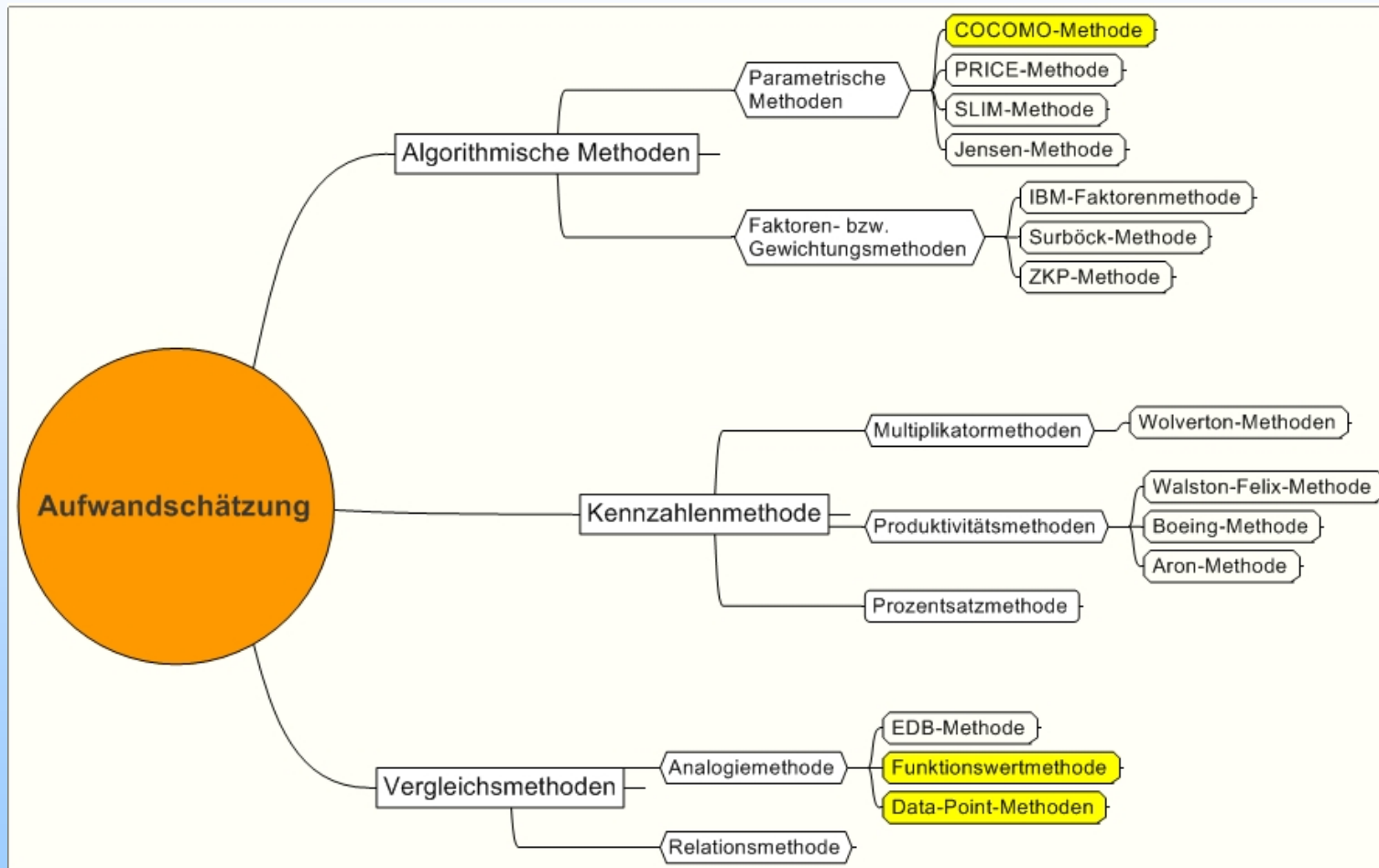
Skalierbarkeit

- Komplexitätsgrenzen zeigen sich durch Fehlerhäufung
- Zeitverlust durch Fehlersuche, statt neue Produkte zu entwickeln (senkt Produktivität)
- Passende Komplexität und Fehlervermeidung durch:
 - Effektivitätsmessung des Einsatzes von Gemeinsamkeiten
 - Effektive Verwaltung von Variationen zwischen den Produkten

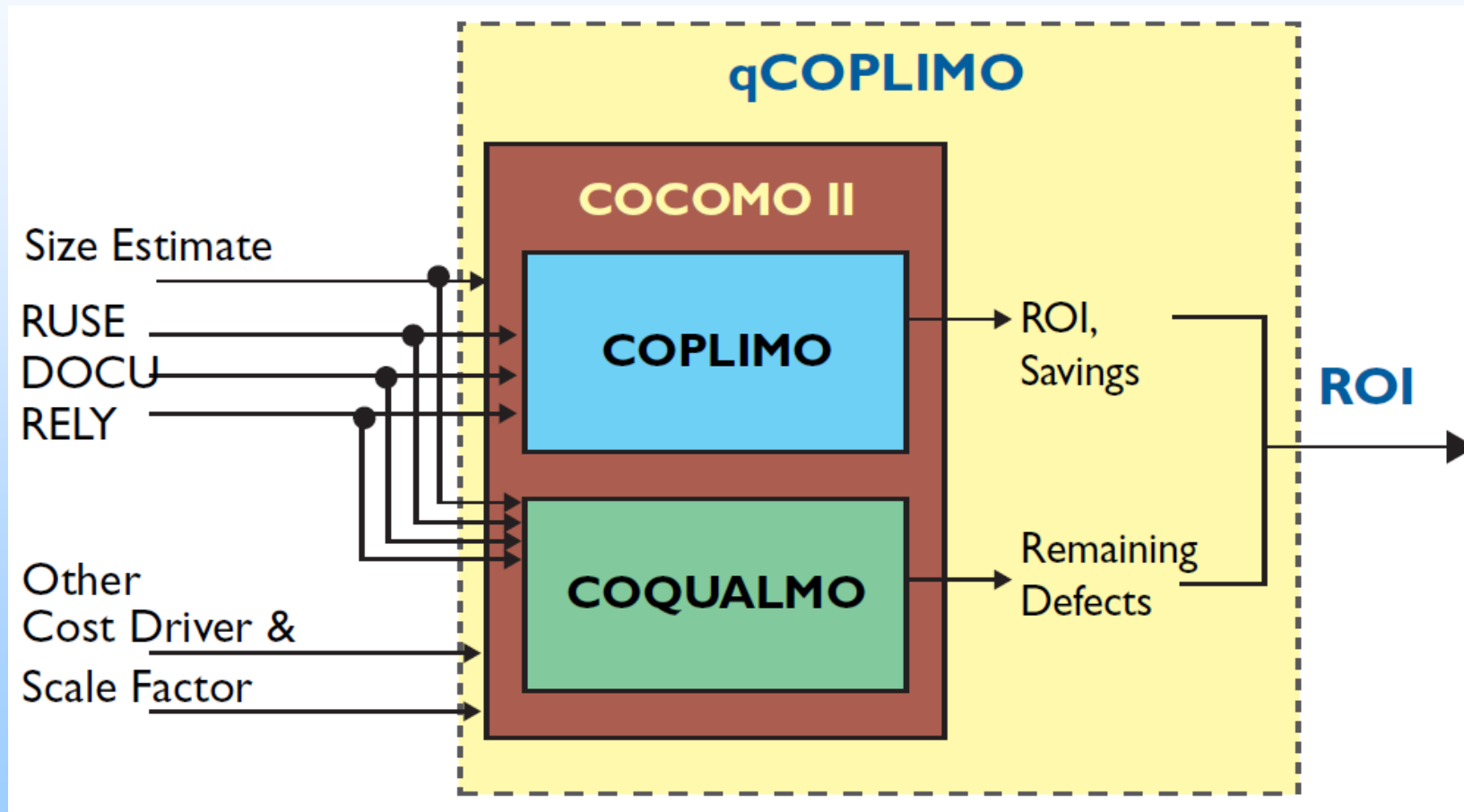
Nachteile/Schwierigkeiten im Überblick

- Gespanntes Verhältnis zwischen Marketing und Entwicklern durch unterschiedliche Sichten
- Hoher Koordinationsaufwand bei Parallelentwicklungen die gleiche Bestandteile nutzen
- Schwierigen und fehlerbehafteten Quellcode auf neue Produkte anzuwenden
- Bei geringer Stückzahl höhere Kosten
- Erhöhter Aufwand als bei „Standartentwicklung“

Übersicht Aufwandsmethoden



QCOPLIMO



COCOMO 2

- Weiterentwicklung des COCOMO 81 Modells
- Aufwandsberechnung
- Zusammengesetzt aus 3 Untermodellen
 1. The Application Composition Model (frühe Prototypenstufe)
 2. The Early Design Model (frühe Entwurfsphase)
 3. The Post-Architecture Model (nach Modellentwurf)

The Application Composition Model

- Verwendete Methode: Objekt Points

| Object Type | Complexity-Weight | | |
|------------------|-------------------|--------|-----------|
| | Simple | Medium | Difficult |
| Bildschirmmakske | 1 | 2 | 3 |
| Bericht | 2 | 5 | 8 |
| 3GL-Modul | 10 | | |

$$OP = \sum (\text{Anzahl der Elemente eines bestimmten Typs}) \cdot (\text{Gewicht})$$

$$NOP = OP \cdot \frac{(100 - \%reuse)}{100}$$

$$\text{Aufwand} = \frac{NOP}{PROD}$$

The Early Design Model

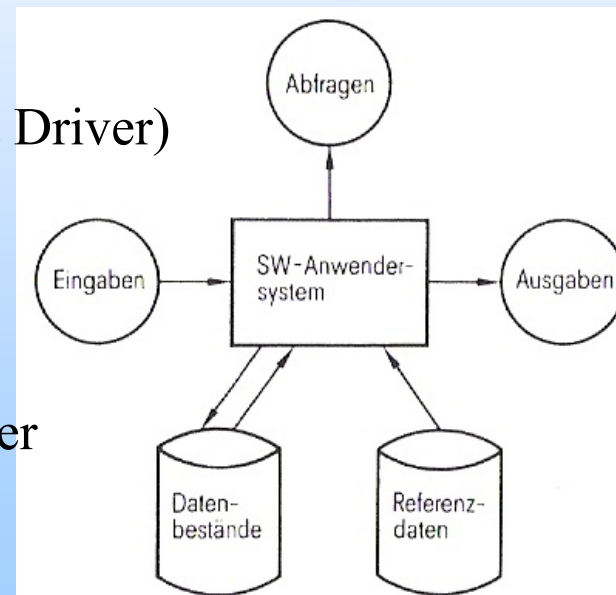
$$Aufwand = 2.45 \cdot E \cdot \left(\left(\text{Größe} \left(1 + \frac{BRAK}{100} \right) \right) \right)^P$$

E = Produkt aus 7 Kostentreibern (Cost Driver)

Größe in Function-Points

Brak = nicht benötigter Code

P = Prozessexponent = $1,01 + \sum \text{Parameter}$



The Post-Architecture Model

$$Aufwand = 2.45 \cdot E \cdot \left(\left(\text{Größe} \left(1 + \frac{BRAK}{100} \right) \right) \right)^P$$

- spezifiziertes Modell
- 17 Cost Driver einbezogen
- Größe [in KDSI] = LOC/1000

QCOPLIMO

- Berücksichtigung von Kosten für die Qualitätssicherung,
- hohe Summe an Wartungs-/Instandhaltungskosten
- Relative Cost of Writing for Reuse (RCWR) für die anfängliche Produktlinienentwicklung
- Relative Cost for Reuse (RCR) für die folgenden Produktentwicklungen

RCWR

- Summierte Kosten für die derartige Implementierung, dass der Quellcode kostengünstig innerhalb der Produktlinie teilweise oder komplett wieder verwendet werden kann.

$$\begin{aligned} C_{RCWR} &= \text{LaborRate} * \text{COPLIMO}_{RCWR} + \text{Software Quality Cost}_{RCWR} \\ &= \text{LaborRate} * [\text{COCOMO baseline (initial software size)} * \text{Effort Adjustment for RCWR}] + \\ &\quad [\text{Cost per Defect} * (1 - \text{Testing Effectiveness}) * \text{COQUALMO (initial software size, EM}_{PL})], \\ &\text{where EM}_{PL} \text{ is the Effort Multiplier of the COCOMO II cost drivers for the product line development} \\ &\text{and COCOMO baseline is calculated as } 2.94 * (\text{software size})^{1.0997} * \Pi(\text{EM}) \end{aligned}$$

RCR

- Min. SPL-Produkt fertig entwickelt
- Kosten der Wiederverwendung in neuer Software in gleicher SPL
- Relation zu einer kompletten Neuentwicklung

$$\begin{aligned} C_{\text{RCR}} &= \text{LaborRate} * \text{COPLIMO}_{\text{RCR}} + \text{Software Quality Cost}_{\text{RCR}} \\ &= \text{LaborRate} * [\text{COCOMO baseline (software size for reuse)}] + \\ &\quad [\text{Cost per Defect} * (1 - \text{Testing Effectiveness}) * \text{COQUALMO (software size for reuse, EM}_{\text{PL}})] \end{aligned}$$

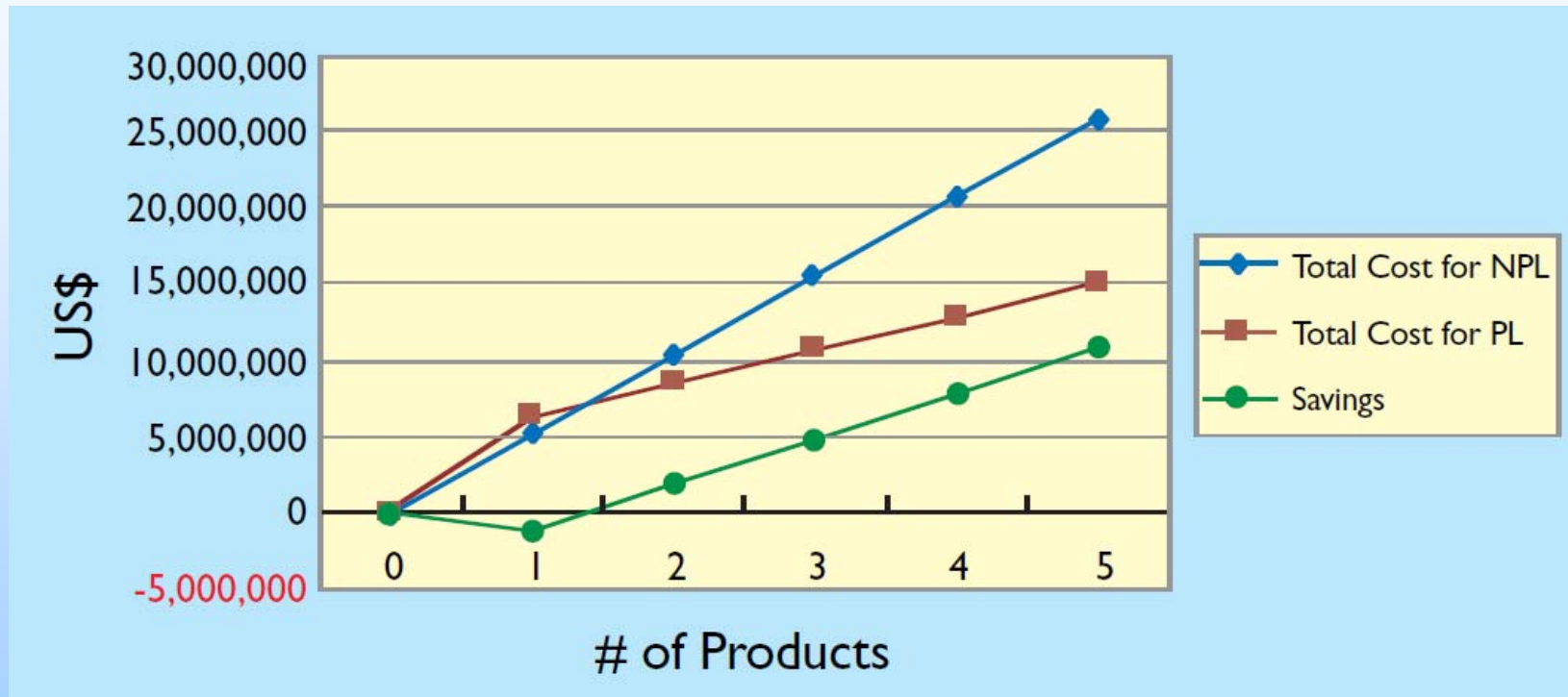
Gesamtkosten

- Kosten für eine SPL in Abhängigkeit der zu entwickelnden Produkte innerhalb der SPL unter Berücksichtigung der Kosten für die Qualitätssicherung

$$C_{PL}(N) = C_{RCWR} + (N - 1) * C_{RCR}$$

where N is the number of products to be developed in SPL

Vergleich



Quellen

- The COCOMO cost estimation model
<http://www.cosy.sbg.ac.at/~pmm/teaching/SS02/se/papers/0020768.pdf>
- <http://www.softwareproductlines.com/index.html>
- Paper: A QUALITY-BASED COST ESTIMATION MODEL FOR THE PRODUCT LINE LIFE CYCLE
<http://sunset.usc.edu/csse/TECHRPTS/2007/usc-csse-2007-721 /usc-csse-2007-721.pdf>
- <http://csse.usc.edu/csse/>
- Burghardt, M. (1997): Projektmanagement: Leitfaden für die Planung, Überwachung und Steuerung von Entwicklungsprojekten. 4. Aufl., Erlangen.
- <http://www.4managers.de/themen/return-on-investment/>
- <http://en.wikipedia.org/wiki/COCOMO>

Danke für die Aufmerksamkeit

