Management großer Softwareprojekte

Prof. Dr. Holger Schlingloff

Humboldt-Universität zu Berlin,
Institut für Informatik

Fraunhofer Institut für Rechnerarchitektur
und Softwaretechnik FIRST

4.2 Schätzverfahren

1. empirische Schätzverfahren
   - durch die Zielfunktion
   - Expertenschätzung
   - Delphi-Methode
2. algorithmische Schätzverfahren
   - Function-point-Methode
   - CoCoMo, CoCoMo II
3. wissensbasierte Schätzwerkzeuge

Merkregeln Function-Point-Methode

- Setzt frühestens beim Lastenheft ein
- betrachtet das gesamte Produkt
- Sichtweise des Auftraggebers
- Bewertung durch Produktexperten
- Ist-Aufwand muss ermittelt werden
- Unternehmensspezifische Faktoren

Kritik an Function Point Methode

+ Quasistandard, akzeptiert
+ basiert auf Produktanforderungen (nicht LOC)
+ iteratives Verfahren, anpassbar
+ früh einsetzbar (Lastenheft)

- dominiert durch Interessenverband
- wenig objektive Werte, Schätzerabhängig
- umfangreiche empirische Datenbasis
- neigt zur Unterschätzung in frühen Phasen

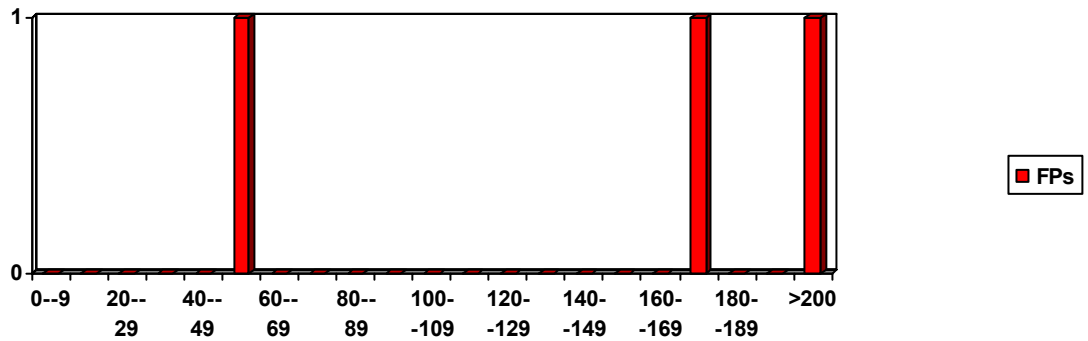
weitere Kritikpunkte

- berücksichtigt nicht OO-Paradigma
- Mischung von Produkt- und Prozesseigenschaften
- mangelnde theoretische Basis

Weiterentwicklung: „Object Points", „Feature Points" usw.

Ergebnisse Hausaufgabe

- Bestimmen Sie die bewerteten Function Points für das Pizza-Beispiel!



Ergebnisse:

(1) zu geringer Rücklauf für eine definitive Aussage; machen Sie die Hausaufgaben!

(2) Zuordnung von FPs folgt gewissen Konventionen (z.B. Maske oder Feld je 1 FP)

LOC, DSI, FP

- LOC (lines of code) und DSI (delivered source instructions) sind fast linear voneinander abhängig
- Produktivität (in LOC/PM) bei höheren Programmiersprachen geringer als bei niederen, Gesamtaufwand trotzdem geringer (Debugging!)

	Größe	Aufwand	Produktivität
Assembler	5000 LOC	7 PM	714 LOC/PM
Java	1500 LOC	5 PM	300 LOC/PM

- FP in LOC mit Faktor umrechenbar (ohne ReUse)
  - 200-300 für low-level, 30-100 für high-level Sprachen

CoCoMo und CoCoMo II

- Grundversion von Boehm, Barry W., „Software Engineering Economics“ (1981)
- Schätzung des Aufwandes in Abhängigkeit von Größe und Komplexität des Projekts sowie sonstiger Rahmenbedingungen

$$a = c * g^{k}$$

g = ***geschätzte Größe***
a = ***berechneter Aufwand***

Annahmen

- Ableitbarkeit des *Umfangs* durch Vergleich mit bereits durchgeführten Projekten
- Wiederverwendeter und generierter Code wird nicht mit gezählt
- Anforderungen bleiben für die Zeit der Entwicklung konstant
- Schätzungen klammern diverse Aufwände aus (z.B. Administration, Training, Umstellung, ...)

Vorgehen

- Schätzung der Anzahl der im Programm enthaltenen Befehle (in Kilo Delivered Source Instructions, KDSI)
- Bestimmung des Schwierigkeitsgrades des Projektes
  - einfaches Projekt (organic mode)
  - mittelschweres Projekt (semidetached mode)
  - komplexe Projekte (embedded mode)
- Einstufung weiterer Kosteneinflussfaktoren auf einer qualitativen Skala von „sehr gering“ bis „extrem hoch“

Bedeutung der Stufen

einfach: wohlverstandene Anwendung, kleines Entwicklungsteam

mittel: großes Team mit wenig Erfahrung in vergleichbaren Produkten

komplex: eingebettetes System, hohe Sicherheitsanforderungen

Faktoren

- **einfach:** $a = 2.4 * g^{1.05}$
- **mittel:** $a = 3.0 * g^{1.12}$
- **komplex:** $a = 3.6 * g^{1.20}$

  a = Aufwand in PM, g = Größe in KDSI

➔exponentielles Wachstum wegen Kommunikationsoverhead

Entwicklungszeit:

- **einfach:** $t = 2.5 * a^{0.38}$
- **mittel:** $t = 2.5 * a^{0.35}$
- **komplex:** $t = 2.5 * a^{0.32}$

  t = zu erwartende Gesamtzeit

Unzulänglichkeit von CoCoMo81

- neue Vorgehensweisen und –modelle
- Wiederverwendung, Reengineering
- COTS, Middleware
- OO-Paradigma

CoCoMo II (1990 - 2000)

dreistufiges Modell, detailliertere Schätzung:

- **(a) frühe Prototypenstufe:**
  - Schätzung basiert auf Object-Points mit einer einfachen Formel
- **(b) frühe Entwurfsstufe:**
  - Schätzung basiert auf Function-Points
- **(c) Stufe nach dem Architekturentwurf:**
  - Schätzung basiert auf LOC, Wiederverwendung

Informationen: Siehe http://sunset.usc.edu/COCOMOII/Cocomo.html
vergleiche auch Sommerville-Buch

frühe Prototypenstufe (a)

- a = (#OP * (1 - %reuse/100)) / PROD

#OP = Anzahl Object-Points

PROD = OP/PM (standardisierte Produktivität)

Erfahrung	sehr gering	gering	normal	hoch	sehr hoch
Produktivität	4	7	13	25	50

frühe Entwurfsstufe (b)

- nach Erstellung des Pflichtenheftes
- ähnlich zur CoCoMo-81-Methode:

$$a = c * g^k * f + PM_m$$

  - $f = PERS * RCPX * RUSE * PDIF * PREX * FCIL * SCED$
  - c ist anfänglich 2.5
  - g geschätzte Größe in KLOC,
  - k reicht von 1.1 to 1.24, abhängig von Neuheitsgrad, Flexibilität der Anforderungen, Prozess-Reifegrad und Risikomanagement
- PM_m = Aufwand für automatisch erzeugten Code

Kostenfaktoren

RCPX	reliability and complexity (Zuverlässigkeit und Komplexität)
RUSE	reuse (Wiederverwendbarkeit)
PDIF	platform difficulty (Plattformkomplexität)
PREX	personnel experience (Erfahrung der Mitarbeiter)
PERS	personnel capability (Mitarbeiterfähigkeiten)
SCED	required schedule (Terminanforderungen)
FCIL	team support facilities (Infrastruktur)

Bewertung der Kostenfaktoren

- Produkt der Kostenfaktoren in einem „normalen“ Projekt ergibt 1
- Bewertung jedes Faktors als „extrem niedrig“, „sehr niedrig“, „niedrig“, „normal“, „hoch“, „sehr hoch“ oder „extrem hoch“
- Werte laut Tabelle, von 0.5 bis 2.7

Early Design Model		Extra low	Very low	Low	Nominal	High	Very high	Extra high
PCPX	Product Reliability and Complexity	0.49	0.60	0.83	1.00	1.33	1.91	2.72
RUSE	Developed for Reusability			0.95	1.00	1.07	1.15	1.24
PDIF	Platform Difficulty			1.00	1.00	1.00		
PERS	Personnel Capability	2.12	1.62	1.26	1.00	0.83	0.63	0.50
PREX	Personnel Experience	1.59	1.33	1.12	1.00	0.87	0.74	0.62
FCIL	Facilities	1.43	1.30	1.10	1.00	0.87	0.73	0.62
SCED	Required Development Schedule		1.43	1.14	1.00	1.00	1.00	

Stufe nach dem Architekturentwurf (c)

- selbe Grundformel wie bei (b): $a = f * g^k$
- Exponent hängt von 5 Skalierungsfaktoren ab
  (siehe nächste Folie)
- genauere Größenschätzungen für Einflussfaktor, 17 statt 7 Kostenfaktoren
  - Unbeständigkeit der Anforderungen: Schätzung des Änderungsaufwandes in LOC
  - Wiederverwendung wird berücksichtigt
  - Wertebereich von 0.7 bis 1.7

Skalierungsfaktoren

- **Vorhandensein**
  frühere Erfahrungen mit ähnlichen Projekten
- **Entwicklungsflexibilität**
  Freiheit der Gestaltung des Entwicklungsprozesses
- **Architektur/Risikoauflösung**
  Umfang der durchgeführten Risikoanalyse
- **Teamzusammenhalt**
  Vertrautheitsgrad der Entwickler untereinander
- **Prozessausgereiftheit**
  (5 – CMM)

Berechnung des Exponenten

- jeder der Skalierungsfaktoren wird mit einer Zahl zwischen 5 (sehr gering) und 0 (sehr hoch) bewertet („Minuspunkte“)
- s = Summe der Skalierungsfaktoren ($0 \leq s \leq 25$)
- $k = 1.01 + s/100$

- $a = f * g^k$

Beispiel

neuartiges Projekt ➔ Vorhandensein = 4
unabhängiger Prozess ➔ Entwicklungsflexibilität = 1
keine Risikoanalyse ➔ Risikoauflösung = 5
neues Team ➔ Teamzusammenhalt = 3
CMM = 2 ➔ Prozessausgereiftheit = 3

$\sum = 16$ ➔ $k = 1.17$

17 Kostenfaktoren

- Produktattribute
  - Eigenschaften des zu entwickelnden Produkts
- Computerattribute
  - Beschränkungen durch verwendete Plattformen
- Personalattribute
  - Erfahrungen und Fähigkeiten der Mitarbeiter
- Projektattribute
  - besondere Eigenschaften des Projektes

Product attributes			
RELY	Required system reliability	DATA	Size of database used
CPLX	Complexity of system modules	RUSE	Required percentage of reusable components
DOCU	Extent of documentation required		
Computer attributes			
TIME	Execution time constraints	STOR	Memory constraints
PVOL	Volatility of development platform		
Personnel attributes			
ACAP	Capability of project analysts	PCAP	Programmer capability
PCON	Personnel continuity	AEXP	Analyst experience in project domain
PEXP	Programmer experience in project domain	LTEX	Language and tool experience
Project attributes			
TOOL	Use of software tools	SITE	Extent of multi-site working and quality of site communications
SCED	Development schedule compression		

Produktattribute

- RELY: verlangte Zuverlässigkeit, 0.75 .. 1.4
- CPLX: Produktkomplexität, 0.73 .. 1.74
- DOCU: Dokumentationsbedarf, 0.81 .. 1.23
- DATA: Größe der Datenbank, 0.9 .. 1.28
- RUSE: Wiederverwendungsgrad, 0.95 .. 1.24

Plattformattribute

- TIME: Ausführungszeitkritikalität, 1 .. 1.66
- PVOL: Volatilität der SW/HW-Plattform
- STOR: Speicherbeschränkungen

Personalattribute

- ACAP: Analysten-Fähigkeiten, 1.46 .. 0.71 (!)
- PCON: Kontinuität des Personals, 1.29 .. 0.81
- PEXP: Erfahrung mit der Plattform
- PCAP: Fähigkeiten der Programmierer
- AEXP: Erfahrung im Anwendungsbereich
- LTEX: programmiersprachliche Kompetenz, 0.95 ..1.14

Projektattribute

- TOOL: Verwendung von Softwarewerkzeugen
- SCED: Entwicklungszeitbeschränkungen
- SITE: Verteiltheit der Entwicklung

konkrete Zahlentabelle

Post Architecture Model		Extra low	Very low	Low	Nominal	High	Very high	Extra high
RELY	Required Software Reliability		0.82	0.92	1.00	1.10	1.26	
DATA	Data Base Size			0.90	1.00	1.14	1.28	
CPLX	Product Complexity		0.73	0.87	1.00	1.17	1.34	1.74
RUSE	Required Reusability			0.95	1.00	1.07	1.15	1.24
DOCU	Documentation match to LC needs		0.81	0.91	1.00	1.11	1.23	
TIME	Execution Time Constraint				1.00	1.11	1.29	1.63
STOR	Main Storage Constraint				1.00	1.05	1.17	1.46
PVOL	Platform Volatility			0.87	1.00	1.15	1.30	
ACAP	Analyst Capability		1.42	1.19	1.00	0.85	0.71	
PCAP	Programmer Capability		1.34	1.15	1.00	0.88	0.76	
PCON	Personal Continuity		1.29	1.12	1.00	0.90	0.81	
APEX	Applications Experience		1.22	1.10	1.00	0.88	0.81	
PEXP	Platform Experience		1.19	1.09	1.00	0.91	0.85	
LTEX	Language and Tool Experience		1.20	1.09	1.00	0.91	0.84	
TOOL	Use of Software Tools		1.17	1.09	1.00	0.90	0.78	
SITE	Multisite Development		1.22	1.09	1.00	0.93	0.86	0.80
SCED	Required Development Schedule		1.43	1.14	1.00	1.00	1.00	

Auswirkung der Faktoren

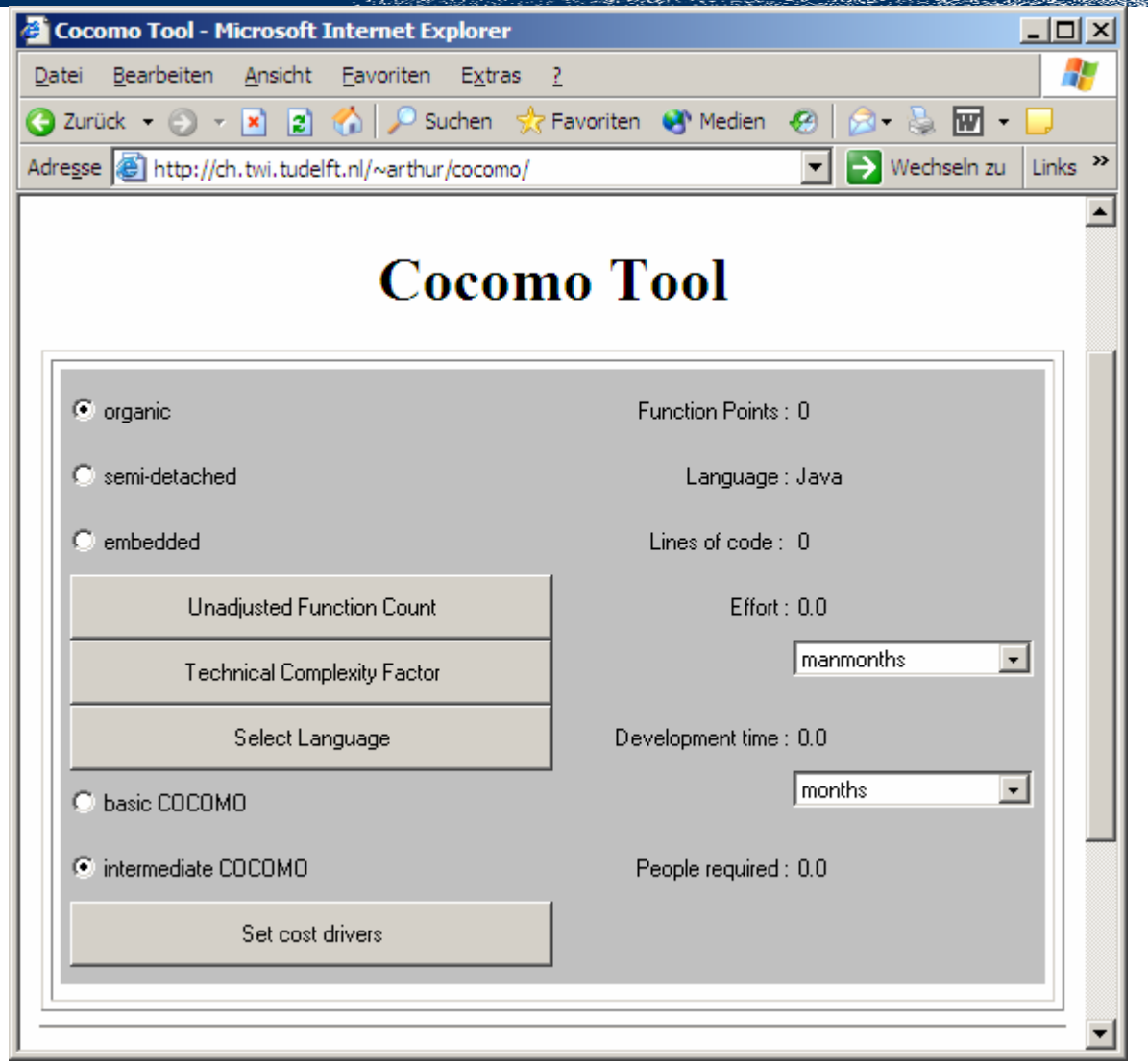
Exponent Systemgröße (einschließlich der Faktoren für ReUse und Unbeständigkeit der Anforderungen) **Erste CoCoMo Schätzung ohne Kostenfaktoren**	1.17 128, 000 DSI **730 Personenmonate**
Zuverlässigkeit Komplexität Speicherbeschränkungen Werkzeugverwendung Zeitplan **Angepasste CoCoMo Schätzung**	Sehr hoch, Faktor = 1.39 Sehr hoch, Faktor = 1.3 Hoch, Faktor = 1.21 Gering, Faktor = 1.12 Beschleunigt, Faktor = 1.29 **2306 Personenmonate**
Zuverlässigkeit Komplexität Speicherbeschränkungen Werkzeugverwendung Zeitplan **Angepasste CoCoMo Schätzung**	Sehr gering, Faktor = 0.75 Sehr gering, Faktor = 0.75 Keine, Faktor = 1 Sehr hoch, Faktor = 0.72 Normal, Faktor = 1 **295 Personenmonate**

Wiederverwendung

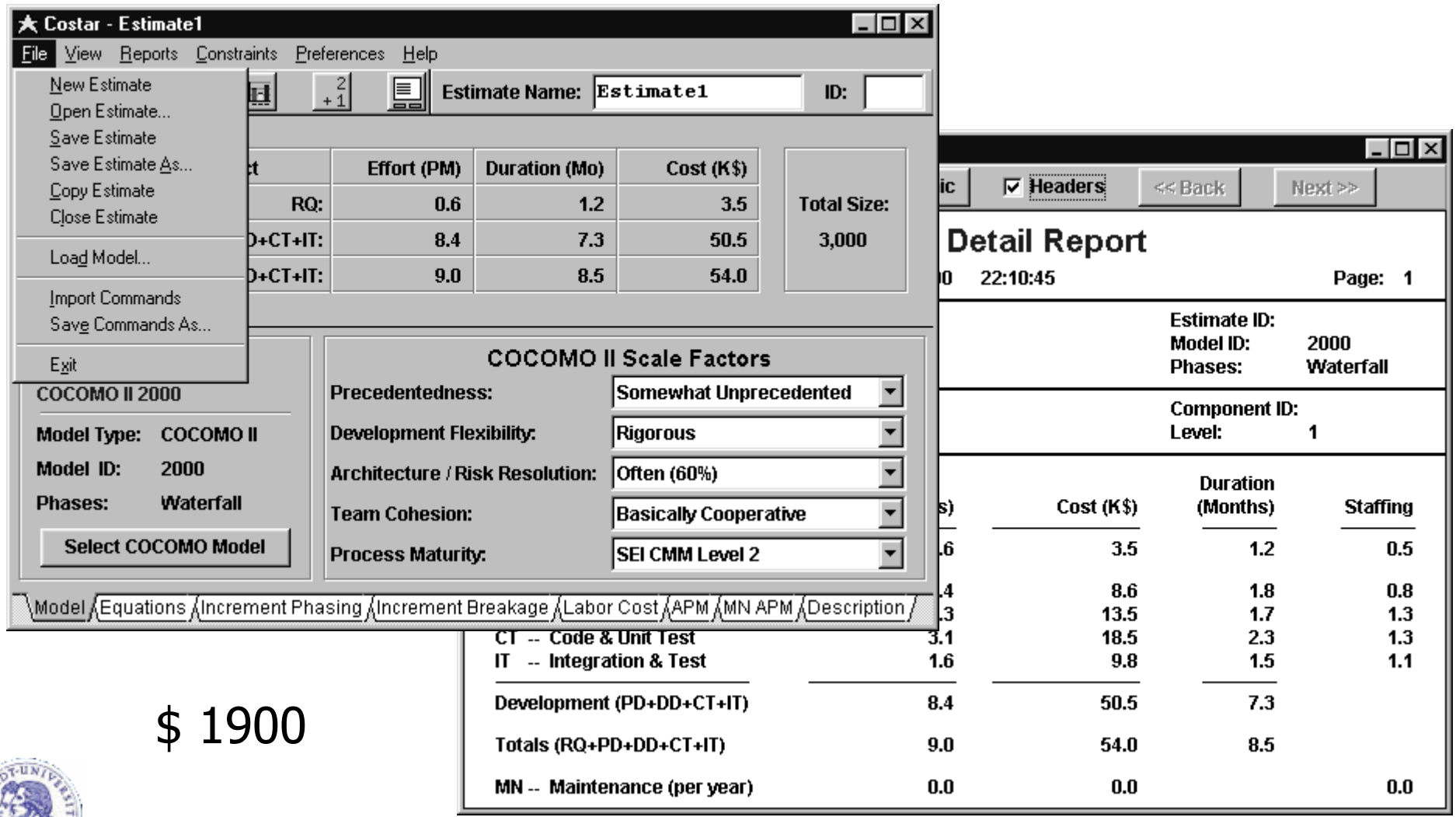
ESLOC = ASLOC * (AA + SU + 0.4*DM + 0.3*CM +0.3*IM)/100

- **ESLOC:** extension SLOC, Anzahl Zeilen neuen Codes
- **ASLOC:** adapted SLOC, zu ändernde wiederverwendete Zeilen
- **AA:** application assessment, Faktor für die Beurteilungskosten für Wiederverwendung
- **SU:** source usability, Faktor für die Kosten der Beherrschung der Software
- **DM:** design modifications, prozentualer Anteil des geänderten Entwurfs
- **CM:** code modifications, prozentualer Anteil geänderten Codes
- **IM:** modifications for integration, prozentualer Anteil des Aufwandes für die Modifikation

Werkzeugunterstützung



kommerzielle Tools ...



$ 1900

Entwicklungszeit

Faustregel von letzter Woche: $t = 2.5 * \sqrt[3]{a}$

CoCoMo:

- **einfach:** $t = 2.5 * a^{0.38}$
- **mittel:** $t = 2.5 * a^{0.35}$
- **komplex:** $t = 2.5 * a^{0.32}$

- CoCoMo 2: $t = 3 * a^{(0.33+0.2*(B-1.01))}$

  $B \sim 1$

Hausaufgabe bis nächste Woche

- Sie sind als Projektleiter des Pizzaservice-Projekts vor die Aufgabe gestellt, eine Kostenschätzung abzuliefern, und wollen dazu ein CoCoMo-Tool einsetzen.

  Vergleichen Sie mindestens drei verfügbare Tools und evaluieren Sie sie bezüglich ihrer Einsetzbarkeit! Begründen Sie Ihre Entscheidung!

Vorteile von COCOMO

- Geeignet für schnelle, grobe Schätzungen der anfallenden Kosten
- Gute Ergebnisse bei kleineren Projekten, die in einer bekannten Entwicklungsumgebung durchgeführt werden (Vergleichbarkeit mit bereits durchgeführten Projekten ist gegeben)
- Abdeckung des Gesamtprojekts angefangen bei der Designphase bis hin zur Testphase (z.T. durch Erfahrungswerte wie 10% Management, 10% Infrastrukturaufwand)

Nachteile von COCOMO

- Vergleichbarkeit mit bereits durchgeführten Projekten nicht immer gegeben
- viele im voraus zu bestimmende Einflussfaktoren
- Erfahrungen zeigen Abweichungen der Schätzungen vom tatsächlichen Aufwand um den Faktor 4

Einsatz von CoCoMo

- CoCoMo81 wurde begeistert aufgenommen
- CoCoMo II vor allem für große Firmen interessant (Bell, Boeing, Motorola, NASA, Rational, Sun, TI, ...)

Marktentwicklung

- weg von „großen“ Basisprojekten, hin zu „kleinen“ Anwendungsprojekten
- Spreadsheets, Skriptsprachen, visuelles Programmieren
- Anwendungsgeneratoren
- Komponententechnologie

Wissensbasierte Schätzung

- Expertensystem zur Vorhersage
- empirisch entwickelte Heuristiken
- regelbasierte Inferenzmethoden
- Parameter-Datenbasis
- großer Markt (NASA-Studie: 20 Anbieter)
- Forschungsthema