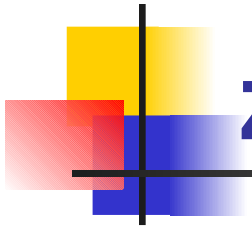


Zagadnienia

- Algorytmiczne metody estymacji kosztów projektu
- Metody CoCoMo oraz CoCoMo 2.0
- Dokładność estymacji metod CoCoMo
- Przykłady estymacji kosztów z wykorzystaniem metod z rodziny CoCoMo

Modele CoCoMo

- CoCoMo (ang. **Co**nstructive **Co**st **Mo**del) - algorytmiczna metoda powstała na podstawie analizy historycznych danych ponad 60 różnych projektów (Boehm 1981)
- Pierwszą wersję nazwano CoCoMo-81; obecnie funkcjonuje wersja oznaczona jako CoCoMo 2.0
- Metody CoCoMo oparte zostały na założeniu, że estymację kosztów projektów można określić w postaci matematyczej funkcji opracowanej na podstawie analizy ukończonych projektów. Opracowany wzór stanowi funkcję atrybutów związanych z produktem oraz wykorzystanym do jego produkcji procesem w ramach projektu (z którym również powiązane mogą być wartości pewnych atrybutów)

Modele CoCoMo-81

Ogólny wzór dla modelu CoCoMo można zapisać w następującej postaci:

$$\text{Estymacja nakładów} = A * S^F * M$$

- Gdzie: **A** – stała zależna od organizacji; **S** – rozmiar systemu (według pewnej miary); **F** – współczynnik określający wpływ rozmiaru projektu na jego realizację; **M** – współczynnik odzwierciedlający atrybuty związane z procesem produkcji, grupą projektową, charakterystyką szacowanego systemu
- Wartości poszczególnych współczynników w metodzie CoCoMo uzależnione są od rodzaju i wielkości projektu oraz poszczególnych atrybutów związanych z produktem i procesem produkcji

Modele CoCoMo-81

- COCOMO-81 wyróżnia estymację w trzech formach:
  - Podstawowej (ang. Basic)
  - Średnio-zawansowanej (ang. Intermediate)
  - Szczegółowej (ang. Detailed)
- Podstawą estymacji w kolejnych formach stanowią estymacje form mniej dokładnych (np.: dla formy średnio-zawansowanej punkt wyjściowy stanowią rezultaty estymacji podstawowej)
- Każda następna forma wprowadza dodatkowe atrybuty wpływające na estymację kosztów przez co wykonane estymacje stają się bardziej dokładne i szczegółowe

Modele CoCoMo-81 – rodzaje projektów (1/2)

- W ramach każdej formy model CoCoMo-81 wyróżnia 3 typy projektów w zależności od stopnia złożoności[1]:
  - **Prosty** (zwany również **Organic** wg Boehm'a) – proste, aplikacje z dobrze znanej dziedziny, tworzone przez małą grupę projektową
  - **Średni** (zwany również **Semi-detached** wg Boehm'a) – bardziej złożone aplikacje, których dziedzina jest słabo rozpoznana przez zespół projektowy, brak doświadczenia w wykorzystaniu narzędzi co może znacząco wpłynąć na koszt realizacji projektu

[1]należy zwrócić uwagę, że podział powstał na początku lat osiemdziesiątych więc nie odzwierciedla istniejącego obecnie podziału oprogramowania (projektów)

Modele CoCoMo-81 – rodzaje projektów (2/2)

- **Zawansowany** (zwany również **Embedded** wg Boehm'a) – aplikacje bardzo złożone (np. osadzone w urządzeniach), wymagające współpracy z wieloma różnymi modułami/urządzeniami, wymagające bardzo dużego doświadczenia i umiejętności programistycznych/analitycznych

Model CoCoMo-81 – estymowane parametry

- **PM** – (ang. Person-Month) liczba miesięcy (nie kalendarzowych !) potrzebnych do realizacji projektu przez jedną osobę
- $\mathbf{T_{DEV}}$ – liczba miesięcy kalendarzowych potrzebnych do realizacji projektu
- **N** – liczba osób, członków grupy projektowej wymagana do realiacji projektu

Model CoCoMo-81 forma Podstawowa (Basic)

- Dla każdego z trzech typów projektów w formie podstawowej określone zostały następujące funkcje:
  - Prosty: **$PM = 2.4 * (KDSI)^{1.05}$**
  - Średni: **$PM = 3.0 * (KDSI)^{1.12}$**
  - Zawansowany: **$PM = 3.6 * (KDSI)^{1.20}$**

gdzie: **KDSI** – **K**ilo **D**elivered **S**ource **I**nstructions – wartość wyznaczana zazwyczaj na podstawie analizy z wykorzystaniem Punktów Funkcyjnych, określa liczbę linii oprogramowania dostarczonego w wyniku realizacji szacowanego projektu. Według pierwotnej (później modyfikowanej) definicji podanej przez Boehm'a jedna instrukacja DSI oznacza dowolną linię w dostarczonym kodzie oprogramowania (bez komentarzy)

Model CoCoMo-81 forma Podstawowa (Basic)

- Niezależnie od formuły pozostałe szacowane parametry w formie podstawowej wyznacza się według następujących funkcji:
  - Prosty: $T_{DEV} = 2.5 * (PM)^{0.38}$
  - Średni: $T_{DEV} = 2.5 * (PM)^{0.35}$
  - Zawansowany: $T_{DEV} = 2.5 * (PM)^{0.32}$
  - Prosty, Średni, Zawansowany: N = T_{DEV}/T_{REQ}

Model CoCoMo-81 forma Podstawowa (Basic) – Przykład

Zadanie: W systemie zarządzającym prostą książką adresową należy zaimplementować funkcjonalność przypisywania (dodawania) użytkownika do zdefiniowanych w systemie grup kontaktowych. Każda grupa identyfikowana jest poprzez podanie nazwy natomiast użytkownik identyfikowany jest poprzez nazwisko i imię. Interfejs do zarządzania książką składać się powinien z jednego formularza prezentującego aktualnie przypisanych użytkowników do grupy oraz listę zarejestrowanych w systemie użytkowników. Interfejs powinien umożliwiać dodawanie i usuwanie użytkowników do/z grupy oraz dodawanie i usuwanie grup. Infomacja o aktualnym przypisaniu użytkowników do grup może być wykorzystywana przez inne aplikacje w systemie. Wykorzystując formę podstawową metody CoCoMo dla prostego projektu oszacuj nakład pracy potrzebny na implementację opisanej funkcjonalności.

Model CoCoMo-81 forma Podstawowa (Basic) – Przykład

Rozwiązanie:

- Określić KDSI wykorzystując metodę punktów funkcyjnych
- Wyznaczyć PM według formuły podstawowej przy założeniu, że mamy doczynienia z prostym projektem
- Wyznaczyć T_{DEV} według formuły podstawowej przy założeniu, że mamy doczynienia z prostym projektem

Model CoCoMo-81 forma Podstawowa (Basic) – Przykład

Rozwiązanie: Wyznaczenie KDSI

- Krok 1: Określenie liczby punktów funkcyjnych w poszczególnych obszarach:
  - W naszym przykładzie mamy do czynnienia z jednym interaktywnym zapytaniem i jednym zewnętrznym plikiem, w którym przechowywane będą przypisania użytkowników do grup tak więc:
    - Zewnętrzne wejścia i wyjścia danych – 0
    - Liczba zewnętrznych interfejsów danych – 0
    - Interakcja z użytkownikiem (liczba usług) – 1
    - Liczba plików wykorzystywanych przez system – 1

Model CoCoMo-81 forma Podstawowa (Basic) – Przykład

Rozwiązanie: Wyznaczenie KDSI

- Krok 2: Klasyfikacja złożoności poszczególnych punktów funkcyjnych według kategorii: **Low** (niska), **Average** (średnia), **High** (wysoka)
  - Wykorzystujemy tabele opracowane dla metody punktów funkcyjnych

Dla obszaru EO i EQ			
File types	Data Elements		
	0-5	6-19	20+
0-1	Low	Low	Avg
2-3	Low	Avg	High
4+	Avg	High	High

W naszym przypadku mamy doczynienia z plikiem zawierającym 3 elementy -> złożoność niska (Low)

Model CoCoMo-81 forma Podstawowa (Basic) – Przykład

Rozwiązanie: Wyznaczenie KDSI

- Krok 3: Przpisanie wag na podstawie złożoności punktów w określonych obszarach:
  - Wykorzystujemy tabele opracowane dla metody punktów funkcyjnych

Obszar (rodzaj punktu funkcyjnego)	Waga		
	Low	Average	High
ILF	7	10	15
EIF	5	7	10
EI	3	4	6
EO	4	5	7
EQ	3	4	6

Model CoCoMo-81 forma Podstawowa (Basic) – Przykład

Rozwiązanie: Wyznaczenie KDSI

- Krok 4: Wyznaczamy wartość UFC Unadjusted Function-point Count) według wzoru:

$$UFC = \Sigma \text{ (liczba elementów) * waga}$$

W naszym przykładzie UFC = 1*3 + 1*5 = **8**

- Krok 5: Wyznaczamy wartość KDSI według wzoru

$$KDSI = UFC * KLOC/FP$$

W naszym przykładzie zakładamy, że 1 FP można zaimplementować w 20 liniach kodu w języku Java:

$$KDSI = 8 * 20/1000 = \mathbf{0,16}$$

Uwaga! Powszechnie przyjmuje się, że metoda CoCoMo nie powinna być stosowana w przypadku projektów gdzie KDSI < 2

Model CoCoMo-81 forma Podstawowa (Basic) – Przykład

Rozwiązanie: Wyznaczenie PM

- Krok 1: Wyznaczamy wartość PM (ang. Person-Month) według formuły podstawowej przy założeniu, że mamy doczynienia z prostym projektem :

$$PM = 2.4 * (KDSI)^{1.05}$$

W naszym przykładzie

$$PM = 2.4\ (0,16)^{1.05} = 0,35$$

Model CoCoMo-81 forma Podstawowa (Basic) – Przykład

Rozwiązanie: Wyznaczenie T_{DEV}

- Krok 1: Wyznaczamy wartość T_{DEV} liczby miesięcy kalendarzowych potrzebnych do realizacji projektu:

$$T_{DEV} = 2.5 * (PM)^{0.38}$$

W naszym przykładzie

$$T_{DEV} = 2.5 * (0,35)^{0.38} = \mathbf{1,68}$$

Model CoCoMo-81 forma Średnio-zawansowana (Intermediate)

- Stanowi rozwinięcie formy podstawowej dzięki uwzględnieniu dodatkowych czynników (tzw. Cost Drivers, Effor-Multipliers) wpływających na estymację kosztów
- W metodzie CoCoMo-81 wyróżnionych zostało 15 czynników stanowiących atrybuty związane z produktem (np.: stopień niezawodności, wydajność, rozmiar bazy danych itp.), procesem wytwórczym i projektem (np.: narzędzia programistyczne, harmonogram, procedury gwarancji jakości itp.), sprzętem (np.: ograniczenia związane z pamięcią, wydajnością procesorów, urządzeniami graficznymi itp.) oraz osobami biorącymi udział w realizacji projektu (np.: doświadczenie zespołu w dziedzinie projektu, doświadczenie analityczne, programistyczne itp.)

Model CoCoMo-81 forma Średnio-zawansowana (Intermediate)

- Dla każdego z trzech typów projektów w formie średnio-zawansowanej określone zostały następujące zależności:
  - Prosty, Średni, Zawansowany:

$$PM_{inter} = PM_{basic} * EAF$$

$$EAF = \Pi_{i=1-15} EM_i$$

gdzie: **EM** (ang. Effort-Multiplier) – modyfikator kosztów (jeden z atrybutów)

  - Prosty: $T_{DEV} = 2.5 * (PM_{inter})^{0.38}$
  - Średni: $T_{DEV} = 2.5 * (PM_{inter})^{0.35}$
  - Zawansowany: $T_{DEV} = 2.5 * (PM_{inter})^{0.32}$

Model CoCoMo-81 forma Średnio-zaawansowana – Przykład

Zadanie:

Wykorzystując formę średnio-zaawansowaną wyznaczyć dla poprzedniego zadania ponownie nakłada pracy oraz czas realizacji i wymaganą liczbę developerów przy założeniu, że nie ma żadnych specyficznych wymagań co do sprzętu, złożoność wprowadzanych zmian jest niska, do implementacji wykorzystywane zostaną narzędzia typu RAD a umiejętności w programistyczne, znajmość języka programowania i środowiska produkcyjnego są wysokie.

Model CoCoMo-81 forma Średnio-zaawansowana – Przykład

Rozwiązanie:

- Określić modyfikatory kosztów oraz wyznaczyć wartość EAF na podstawie wzoru:

$$\mathbf{EAF = \Pi_{i=1\text{-}15} EM_i}$$

Zakładamy, że wartość modyfikatorów, których nie uwzględniamy jest nominalna (ang. Nominal)

- Wyznaczyć nominalny nakład pracy wyrażony jako liczba osobo-miesięcy według wzoru:

$$\mathbf{PM_{inter} = PM_{basic} * EAF}$$

- Wyznaczyć czas T_{DEV} według formuły średnio-zaawansowanej przy założeniu, że mamy doczynienia z prostym projektem

Model CoCoMo-81 forma Średnio-zawansowana (Intermediate) – modyfikatory kosztów

Cost Drivers	Poziom					
	Very Low	Low	Nominal	High	Very High	Extra High
Atrybuty Produktu						
Niezawodność (RELY)	0.75	0.88	1.00	1.15	1.40	
Rozmiar bazy danych (DATA)		0.94	1.00	1.08	1.15	
Złożoność (CPLX)	0.70	0.85	1.00	1.15	1.30	1.65
Atrybuty Sprzętu						
Ograniczenia czasu wykonania (TIME)			1.00	1.11	1.30	1.66
Ograniczenia pamięci (STOR)			1.00	1.05	1.21	1.56
Zmiany środowiska produkcyjnego (VIRT)		0.87	1.00	1.15	1.30	
Czas oczekiwania na wyniki (TURN)		0.87	1.00	1.07	1.15	
Atrybuty Personelu						
Umiejętności analityczne (ACAP)	1.46	1.19	1.00	0.85	0.71	
Doświadczenie w tworzeniu aplikacji (AEXP)	1.29	1.13	1.00	0.91	0.82	
Umiejętności programistyczne (PCAP)	1.42	1.17	1.00	0.88	0.70	
Znajomość środowiska produkcyjnego (VEXP)	1.21	1.10	1.00	0.90		
Doświadczenie w języku programowania (LEXP)	1.14	1.07	1.00	0.95		
Atrybuty Projektu						
Wykorzystanie nowoczesnych praktyk programowania (MODP)	1.24	1.10	1.00	0.91	0.82	
Wykorzystanie narzędzi (TOOL)	1.24	1.10	1.00	0.91	0.83	
Wymagany plan projektu (SCED)	1.23	1.08	1.00	1.04	1.10	

Model CoCoMo-81 forma Średnio-zaawnsowana – Przykład

Rozwiązanie: Wyznaczenie EAF i PM_{inter}

- Krok 1: Wyznaczamy wartość EAF korzystając z tabeli wartości modyfikatorów na podstawie wymagań zawartych w zadaniu dotyczących poszczególnych obszarów i atrybutów:

$$EAF = 0.70(CPLX) * 0.82(AEXP) * 0.70(PCAP) * 0.90(VEXP) * 0.95(LEXP) * 0.83(TOOL) = 0.29$$

- Krok 2: Wyznaczamy wartość PM_{inter}:

$$PM_{inter} = 0.35 * 0.29 = 0.10$$

Model CoCoMo-81 forma Średnio-zaawnsowana – Przykład

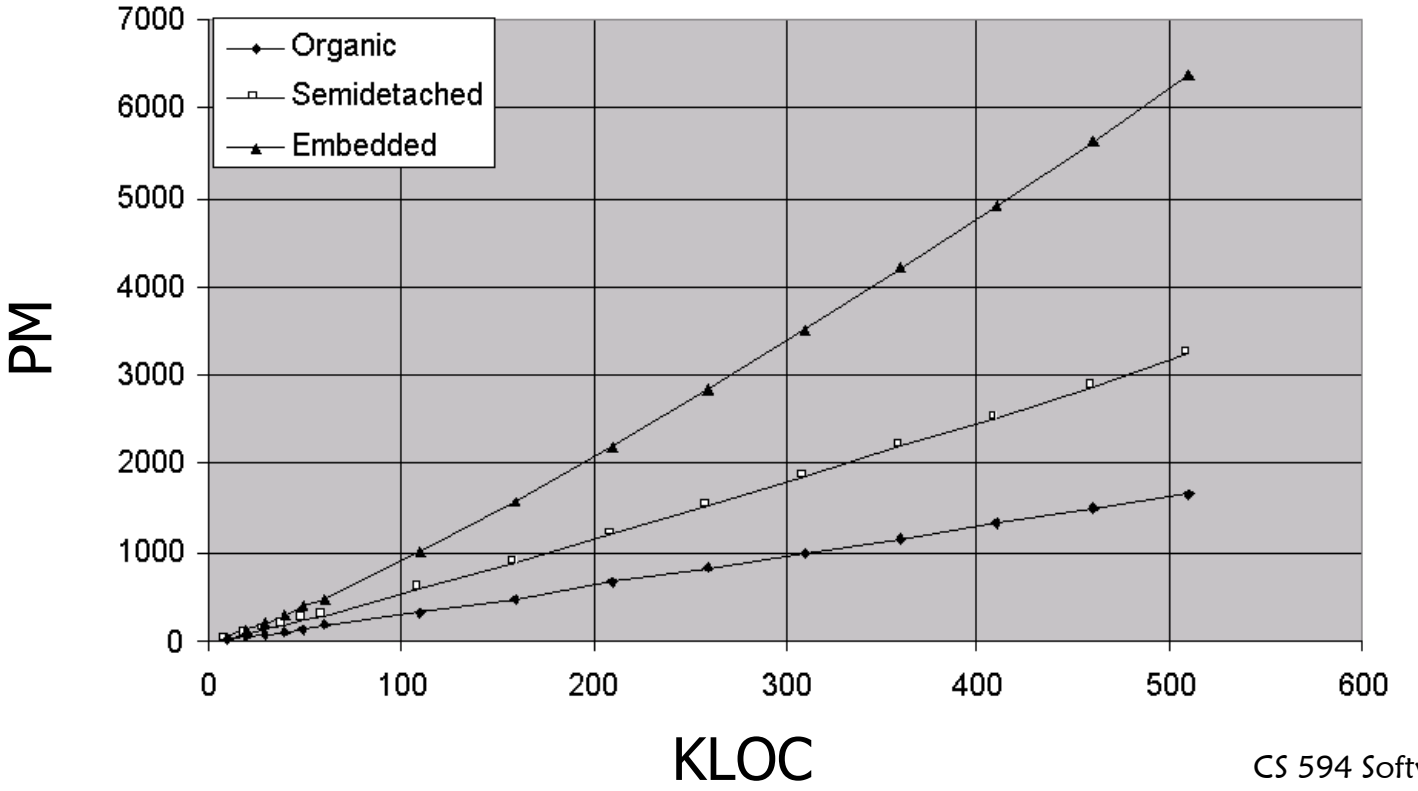
Rozwiązanie: Wyznaczenie T_{DEV}

- Krok 1: Wyznaczamy wartość

$$T_{DEV} = 2.5 * (0.10)^{0.38} = \mathbf{1,04}$$

- Dzięki uwzględnieniu dodatkowych informacji dotyczących projektu szacowany czas realizacji został skrócony o 0,64 miesiąca kalendarzowego (1.68 - 1.04) co daje około 19 dni różnicy

Model CoCoMo-81 porównanie form



CS 594 Software Engineering
dr Thomas E. Potok

Model CoCoMo-81 ograniczenia (1/2)

- Konieczność dostosowania do lokalnych warunków ogranizacji/środowiska produkcyjnego
- Estymacja bardzo wrażliwa na małe zmiany poszczególnych parametrów (dla projektów, których rozmiar jest rzędu 20 KLOC rozbieżność estymacji przy niewielkich zmianach parametrów może sięgać kilkunastu osobo-miesięcy)
- Brak uwzględnienia zastosowania gotowych bibliotek, rozwiązań (ang. **C**ommercial **O**f-**T**he-**S**helf **COTS**)
- Brak wsparcia dla języków 4-GL oraz generatorów aplikacji

Model CoCoMo-81 ograniczenia (2/2)

- Brak uwzględnienia stopnia znajmości wymagań, architektury systemu w miarę rozwoju i stopnia realizacji systemu
- Bazowanie na modelu kaskadowym, brak wsparcia dla podejścia iteracyjnego
- Brak uwzględnienia zarządzania ryzykiem i jego wpływu na koszt i czas realizacji projektów
- Brak uwzględnienia rozproszenia grupy projektowej i realizacji projektów podzielonych na wiele modułów implementowanych przez różne zespoły projektowe

Model CoCoMo 2

- Stanowi rozwinięcie modelu CoCoMo 81 oraz jego adaptację do zmieniających się warunków realizacji projektów informatycznych
- Rozwijany w połowie lat 90-tych głównie przez grupę z University of Southern California
- W stosunku do CoCoMo 81 stanowi rozwinięcie z uwzględnienim następujących zagadnień:
  - Uwzględniono estymację w poszczególnych etapach realizacji projektów zgodnie z modelami spiralnym i iteracyjnym poprzez wprowadzenie trzech faz estymacji:

Model CoCoMo 2

- Faza 1 zwana **Application Composition**
- Faza 2 zwana **Early Design**
- Faza 3 zwana **Post-Architecture**

- Stałe współczynniki ekspotencjalne w poszczególnych formach CoCoMo 81 zastąpione zostały tzw. współczynnikami skalowania (ang. **Scale Factors**)
- Poszerzono wpływ reużywalności oprogramowania a przede wszystkim jej nieliniowy wpływ na realizację projektów (na koszt i czas)
- Uwzględniono podejście obiektowe i komponentowe do projektowania/programowania systemów informatycznych

Model CoCoMo 2

- W znacznie szerszym zakresie uwzględniono wpływ ograniczeń czasowych projektu oraz wymaganej niezawodności produktu na estymację
- Dodano nowe modyfikatory kosztów jak również usunięto te, których wpływ na koszt realizacji projektów przestał odgrywać istotną rolę (np. TURN)
- Wartości większości pozostały współczynników zostały zmodyfikowane na podstawie analizy danych z nowych projektów

Model CoCoMo 81 vs CoCoMo 2

	CoCoMo 81	CoCoMo 2
Struktura	Pojedynczy model estymacji zakładający, że estymacja prowadzona jest przy nie zmieniających się w czasie wymaganiach znanych w pełni przed rozpoczęciem realizacji projektu	Trzy modele odzwierciedlające kolejne fazy modelu iteracyjnego uwzględniającego wzrost rozumienia wymagań i architektury oraz zarządzanie ryzykiem w projekcie
Formuła matematyczna	Effort = $A(c_i)$ $(Size)^{Exp}$	Effort = $A(c_i)$ $(Size)^{Exp}$
Estymacja rozmiaru	KDSI	Object points, function points lub linie kodu źródłowego (SLOC)
Cost drivers	15 współczynników dla których muszą zostać określone poziomy: **RELY, DATA, CPLX,TIME,STOR, VIRT,TURN,ACAP, PCAP,AEXP,VEXP, LEXP, TOOL,MODP, SCED**	17 współczynników dla których muszą zostać określone poziomy: **RELY, DATA,CPLX,RUSE, DOCU,TIME,STOR, PVOL,ACAP,PCAP,AEXP, PEXP,LTEX,PCON, TOOL,SITE,SCED**
Reużywalność	Liniowa zależność wpływu reużywalności	Nieliniowa zależność wpływu reużywalności
Zarządzanie zmianami wymagań	Brak wpływu zmian wymagań na estymację	Uwzględnienie zmienności wymagań w estymacji kosztów i czasu

Model CoCoMo 2 – faza Application Composition

- Application Composition ma na celu umożliwienie realizacji wstępnego szacowania na etapie prototypowania i rozwiązywania zagadnień związanych ze zidentyfikowanym ryzykiem i zagrożeniami
- Dotyczy szacowania obszarów związanych z:
  - Interfejsem użytkownika
  - Interakcją systemu z innymi, zewnętrznymi systemami/modułami
  - Wydajnością systemu
  - Technologią jaka zostanie wykorzystana do realizacji systemu

Model CoCoMo 2 – faza Application Composition

- Szacowanie w fazie Application Composition polega na określeniu nakładu pracy (PM) z wykorzystaniem metody **Object Points** według następującego algorytmu:
  - Krok 1 – wyznaczenie liczby obiektów z jakich powinien składać się system aby realizował założoną funkcjonalność w następujących grupach:
    - Ekrany
    - Raporty
    - Gotowe komponenty (np. ze środowiska RAD)
  - Krok 2 – Klasyfikacja wyznaczonych obiektów pod względem stopnia skomplikowania według następującego schematu:

Model CoCoMo 2 – faza Application Composition

Ekrany			
Liczba zawartych formatek	Liczba tabel i źródło wykorzystanych danych		
	< 4 (<2 serwer, <3 klient)	< 8 (<2-3 serwer, <3-5 klient)	> 8 (>3 serwer, >5 klient)
< 3	Simple	Simple	Medium
3 – 7	Simple	Medium	Difficult
> 8	Medium	Difficult	Difficult

Raporty			
Liczba zawartych sekcji	Liczba tabel i źródło wykorzystanych danych		
	< 4 (<2 serwer, <3 klient)	< 8 (<2-3 serwer, <3-5 klient)	> 8 (>3 serwer, >5 klient)
0 – 1	Simple	Simple	Medium
2 – 3	Simple	Medium	Difficult
> 4	Medium	Difficult	Difficult

Kauffman, Kumar, "Modeling Estimation Expertise in Object Based ICASE Environments,"

Model CoCoMo 2 – faza Application Composition

- Krok 3 – każdemu obiektowi przypisujemy na podstawie określonego poziomu skomplikowania odpowiednią wagę według poniżeszego schematu:

Typ obiektu	Poziom złożoności/waga		
	Simple	Medium	Difficult
Ekran	1	2	3
Raport	2	5	8
Komponent			10

- Krok 4 – określenie liczby punktów Object Points poprzez sumowanie wszystkich wyznaczonych wag (**OP**)
- Krok 5 – oszacowanie procentowe stopnia reużywalności oprogramowania jakie może zostać wykorzystane w projekcie (%Reuse)

Model CoCoMo 2 – faza Application Composition

- Krok 6 – Wyznaczenie liczby punktów funkcyjnych z uwzględnienim stopnia reużywalności oprogramowania według zależności:

$$NOP = OP * (100 - \%Reuse)/100$$

- Krok 7 – określenie nakładów wyrażonych jako PM według zależności:

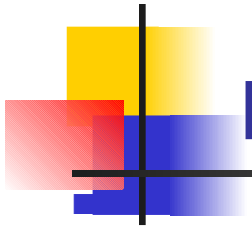
$$PM = NOP/PROD$$

gdzie PROD oznacza produktywność wyznaczoną na podstawie następującego schematu:

Doświadczenie i umiejętności deweloperów oraz dojrzałość i możliwości narzędzi wspomagających proces produkcji oprogramowania (RAD)	Very Low	Low	Normal	High	Very High
PROD	4	7	13	25	50

Model CoCoMo 2 – faza Early Design

- Early Design ma na celu umożliwienie estymacji we wczesnych fazach realizacji projektu, kiedy wstępne wymagania zostały już określone a realizacja projektu jest na etapie tworzenia architektury systemu
- Podstawę oszacowania rozmiaru produktu stanowi metoda punktów funkcyjnych, przeliczanych na SLOC
- Wprowadzono zależność współczynnika ekspotencjalnego (w granicach 1.01 - 1.26) od:
  - Stopnia znajomości danego zagadnienia/domeny/technologii
  - Elastyczności procesu produkcji i wymagań



Model CoCoMo 2 – faza Early Design

- Spójności i łatwości komunikacji członków zespołu projektowego
- Dojrzałości procesu produkcji zgodnie z wymaganiami określonymi przez CMM KPA (ang. Process maturity–SEI)

Model CoCoMo 2 – faza Early Design

- Wprowadzono 7 modyfikatorów kosztów (Cost Drivers) w następujących obszarach:
  - Produkt (RCPX, RUSE)
  - Platforma (złożoność platformy PDIF)
  - Personel (doświadczenie i umiejętności PERS, PREX)
  - Projekt (SCED, FCLI)
- Wprowadzono nieliniową, zależność estymacji kosztu od stopnia wykorzystania istniejących komponentów/bibliotek (rozróżniono wykorzystanie kodu bez modyfikacji i z możliwością dostosowania do potrzeb projektu)

Model CoCoMo 2 – faza Post-Architecture

- Szczegółowe szacowanie nakładów wymaganych na realizację systemu i jego późniejsze utrzymanie
- Rozszerza szacowanie fazy Early Design
- Główne zmiany w stosunku do CoCoMo 81 i fazy Early Design:
  - Wprowadzenie ekwiwalentu linii kodu dla oprogramowania wykorzystywanego w produkcji nowego systemu (ESLOC)
  - Wprowadzenie modyfikatora odpowiedzialnego za zarządzanie zmianami wymagań

Model CoCoMo 2 – faza Post-Architecture

- 17 modyfikatorów nakładu w miejsce 15 z CoCoMo 81 i 7 z fazy Early Design
- Zależność czynnika ekspotencjalnego od 5 współczynników związanych z produktem, projektem i grupą projektową, którym przypisywane są odpowiednie wagi

Model CoCoMo 2 – faza Post-Architecture

Zaobserwowany i wprowadzony do CoCoMo 2 nieliniowy charakter wpływu reużywalności oprogramowania przy produkcji nowych systemów związany jest z trzema czynnikami:

- Kosztem potrzebnym do wyboru, ewaluacji i zaznajomienia się zespołu projektowego z reużywanym oprogramowaniem
- Wykorzystaniem oprogramowania bez modyfikacji
- Wykorzystaniem oprogramowania z jednoczesnym wprowadzeniem własnych zmian (koszt poznania oprogramowania, koszt wprowadzenia samych zmian, koszt testów związanych ze zmianami i ich wpływem na niezmienione części/moduły)

Model CoCoMo 2 – faza Post-Architecture

W wyniku prowadzonych analiz dla fazy Post-Architecture modelu CoCoMo 2 wprowadzono pojęcie ekwiwalentu liczby linii kodu źródłowego (ESLOC) dla reużywanego oprogramowania, który wliczany do rozmiaru szacowanego systemu:

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

gdzie: ASLOC – liczba linii kodu do zaadoptowania

AA – koszt wyboru i poznania adoptowanego oprog.

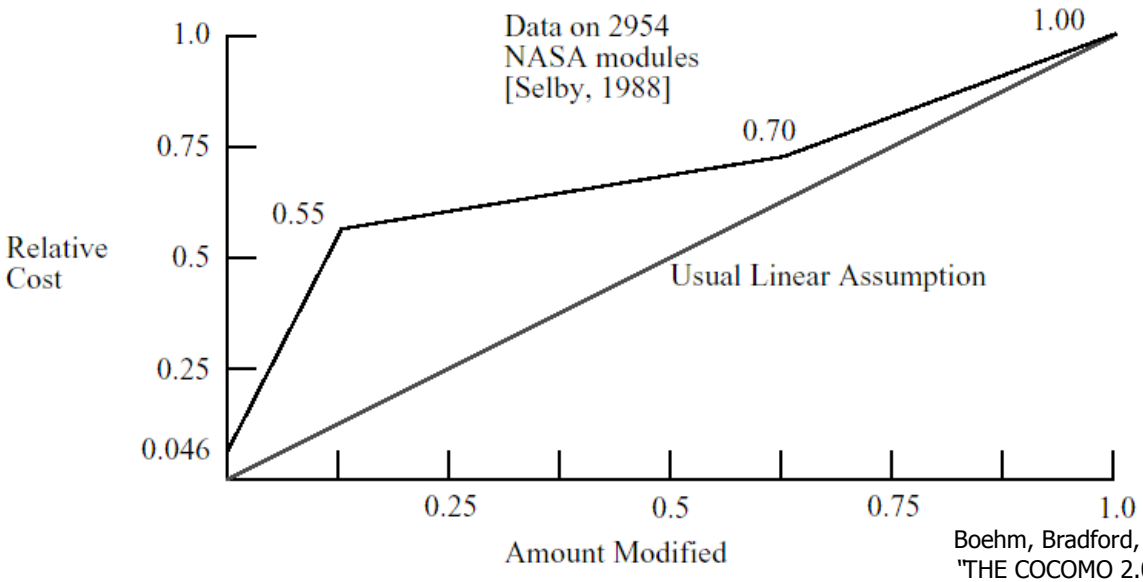
SU – zrozumienie adoptowanego oprog.

DM – wymagany procent modyfikacji architektury

CM – wymagany procent modyfikacji kodu

Model CoCoMo 2 – faza Post-Architecture

IM – wymagany procent integracji zmodyfikowanego oprogramowania w stosunku do zastosowania oprogramowania bez modyfikacji



Boehm, Bradford, Horowitz, Westland, Madachy, Selby, "THE COCOMO 2.0 SOFTWARE COST ESTIMATION MODEL"

Model CoCoMo 2 – faza Post-Architecture

- W fazie Post-Architecture zmodyfikowano również eksponetcjalny współczynnik (B) w zależności określającej szacowany nakład:

$$PM = \Pi_{i=1\text{-}17} EM_i * A * (Rozmiar)^B$$

uwzględniając zaobserwowane nieliniowe zmiany wraz ze zmianami rozmiaru systemów. W rezultacie otrzymano następującą zależność:

$$B = 1.01 + 0.01 * \Sigma W_i$$

gdzie: W_i oznacza wagę współczynnika skalowania w obszarach takich samych jak w przypadku fazy Early Design.

Model CoCoMo 2 – faza Post-Architecture

Poszczególne wagi mogą przyjmować wartości w zakresie [0-5] co daje wpływ zmiany jednej wagi o jeden punkt na poziomie 4,7% całkowitego oszacowania.

Model CoCoMo 2

- Metoda CoCoMo 2 pozwala również na określenie oszacowania nakładów w przypadku optymistycznym oraz pestymistycznym. Zakładając, że E oznacza oszacowanie wyznaczone zgodnie z dowolną fazą modelu wówczas omawiane warianty oszacowania można wyznaczyć zgodnie ze schematem przedstawionym w poniższej tabeli

Model/Faza	Estymacja optymistyczna	Estymacja pesymistyczna
Application Composition	0.50 E	2.0 E
Early Design	0.67 E	1.5 E
Post-Architecture	0.80 E	1.25 E