22. Szacowanie kosztów wytwarzania oprogramowania

Ze względu na złożoność realizacji projektów informatycznych (i konieczność uwzględnienia wielu nieplanowanych zdarzeń) szacowanie kosztu wytwarzania oprogramowania jest trudne (podobnie jak ustalanie harmonogramu).

Powody szacowania kosztów wytworzenia oprogramowania:
- ustalenia budżetu projektu
- ustalenia ceny dla klienta

Koszt realizacji projektu informatycznego składa się głównie z:
- kosztów osobowych (wynagrodzenie wykonawców i personelu wspomagającego)
- kosztów sprzętu i oprogramowania usługowego
- kosztów realizacji:
    a) wyjazdy, szkolenia
    b) utrzymanie pomieszczeń, materiały biurowe, itp.

Korzyści z przyjęcia projektu:
- finansowe (zysk gdy środki wydane na powstanie projektu są niższe niż wynagrodzenie otrzymane od klienta)
- wejście w nowy segment rynku, zdobycie doświadczenia
- możliwość zdobycia kolejnego projektu od klienta dla który realizuje się aktualny projekt
- podniesienie kwalifikacji pracowników podczas realizacji projektu

Problemy:
Najtrudniejszym elementem szacowania kosztu realizacji przedsięwzięcia informatycznego jest określenie wydajności pracy (czyli ile zajmie realizacja określonego zadania przez określoną liczbę osób w określonych warunkach).

W celu ułatwienia szacowania kosztu wprowadza się rozmaite metryki wydajności:
- metryki oparte na rozmiarze kodu (np. liczba linijek kodu produkowanego miesięcznie przez pojedynczego programistę)
- metryki oparte na uzyskiwanej funkcjonalności (np. interakcjach z użytkownikiem, interakcjach z urządzeniami we/wy)
Metryki są bardzo szacunkowe i przybliżone, nie uwzględniają rozmaitości i złożoności oprogramowania.

Szacowanie kosztu jest każdorazowo zadaniem złożonym i indywidualnym, można realizując go wspomagać się następującymi technikami:
- modelowaniem algorytmicznym
- ekspertyzami specjalistów
- porównaniami z uprzednio realizowanymi projektami

Modelem algorytmicznego szacowania kosztu realizacji przedsięwzięcia informatycznego (liczby osobogodzin w procesie tworzenia oprogramowania) jest np. model **COCOMO** (ang. constructive cost model) , posiadający dwie wersje I i II.

Model COCOMO I jest modelem empirycznym powstałym z uogólnienia dotychczasowych doświadczeń z wytwarzaniem oprogramowania.

W modelu COCOMO II wyróżnia się cztery podmodele dla różnych faz projektu:
- kompozycyjny (application-composition)
- fazy wstępnej (early design)
- ponownego użycia (reuse)
- post-architektoniczny (post-architecture)

Model COCOMO stosuje proste wzory matematyczne z szeregiem parametrów. W modelu tym koszt wyraża się jako funkcję miary kodu (miarą może być liczba linii lub liczba tzw. punktów funkcyjnych).

Pierwszym krokiem jest oszacowanie z ilu linijek kodu lub punktów funkcyjnych będzie się składać gotowy projekt. Liczba linijek kodu jest przedstawiana w KDSI (1000 (K) delivered source instructions [1 KDSI = 1000 linijek].

Następnie należy ustalić złożoności:
Wybrać do której z trzech poniższych grup pasuje analizowany projekt.
- Łatwy ("organic mode"), to projekt, w którym mały zespół posługuje się znanymi narzędziami pracy. Zna on sprzęt i oprogramowanie, z którymi rozwijany projekt będzie redagować. Presja czasu jest mała. Łatwe projekty są wielkości do max. 50 KDSI.
- Pośredni projekt ("semi-detached"), to projekt, w którym jeden z czynników z projektu prostego nie jest znany, np. zespół nie zna sprzętu, który przyjdzie mu programować itp. Takie projekty są zwykle wielkości do 300 KDSI.
- Trudny ("embedded mode"), to bardzo złożony projekt, wiele czynników jest nieznanych lub należy uwzględnić szczególne procedury,np. w branży bankowej.

Następnie analizowane są dodatkowe cztery atrybuty:

- Produktu (złożoność oprogramowania, stopień ponownego wykorzystania kodu, nowatorstwo projektu, wymagania efektywności i niezawodności oprogramowania, wymagana czytelność stworzonego oprogramowania, wielkość bazy danych)
- Sprzętu (ograniczenia związane z wydajnością, czy tworzony system jest systemem czasu rzeczywistego, ograniczenia pamięci)
- Personelu (analiza możliwości, doświadczenie w tworzeniu oprogramowania, doświadczenie w tworzeniu oprogramowania danego typu, doświadczenie w tworzeniu oprogramowania wykorzystującego dane środowisko, sprzęt czy język programowania, spójność zespołu realizującego projekt)
- Projektu (jakie narzędzia są potrzebne?, jakie metody tworzenia oprogramowania będą wykorzystywane?, jaki jest harmonogram projektu?, jaka jest presja czasu?)

Wzory

	gdzie:
$E = a_b(KDSI)^{bb}$	E - jest nakładem pracy w osobomiesiącach,
$D = c_b(E)^{db}$	D - jest czasem, jaki jest potrzebny do rozwoju projektu,
$P = E/D$	P - oznacza liczbę osób, przy której projekt będzie najefektywniej zrealizowany

Stałe a_b, b_b, c_b i d_b są podane poniżej:

Software project	a_b	b_b	c_b	d_b
Organic	2.4	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32