2 Projektno vodenje

2.1 Upravljanje projektov programske opreme

Upravljanje projektov programske opreme se ukvarja z aktivnostmi, ki zagotavljajo, da je programska oprema dostavljena pravočasno in v skladu z zahtevami.

Kriteriji uspeha:

- dostava opreme **v dogovorjenem času**
- skupni stroški razvoja **v okviru proračuna**
- oprema **ustreza zahtevam stranke**
- vzdrževanje **skladne** in **dobro delujoče razvojne skupine**

2.1.1 Posebnosti upravljanja projektov

- izdelek **ni oprijemljiv** - ga ne moremo prijeti, ni ga mogoče videti - napredek preverimo tako, da preverimo sam izdelek
- številni projekti pri razvoju so **enkratni** - težko predvidimo napake, ki se bodo zgodile pri izvajanju enega projekta
- procesi razvoja so **specifični** in **spremenljivi** glede na posamezno podjetje

2.1.2 Dejavniki, ki vplivajo na vodenje projektov

- **velikost podjetja** - manjša podjetja imajo manjši strošek režiranja
- **stranka** - komunikacija s stranko lahko poteka na neformalnem ali formalnem nivoju, odvisno od tega, kdo je stranka (nekdo ki ga poznamo osebno / vladna organizacija)
- **velikost programske opreme** - večji kot je obseg programske opreme, večjo ekipo potrebujemo
- **vrsta programske opreme** - npr. pri *kritičnih* sistemih je potrebno zabeležiti vse odločitve pri vodenju projetka
- **organizacijska kultura** - pri določenih podjetjih spodbujajo posameznike, spet drugje pa je poudarek na skupinskem delu
- **procesi razvoja** - npr. pri agilnem pristopu poskušamo zmanjšati režijske stroške z upravljanjem

Ti dejavniki povejo, da lahko vodje projektov na različnih projektih delujejo na različne načine.

2.2 Univerzalne aktivnosti upravljanja

- pisanje projektnega predloga
- načrtovanje projekta
- upravljanje s tveganji
- upravljanje z ljudmi
- poročanje

2.2.1 Upravljanje s tveganji

Upravljanje s tveganji se ukvarja s prepoznavanjem tveganj pri razvoju projekta in pripravo načrta za zmanjševanje njihovega vpliva na projekt.

2.2.1.1 Klasifikacija tveganj

Poznamo dva vidika klasificiranja tveganj:

1. **vrsta tveganj** - tehnično ali organizacijsko itd.
2. **na kaj vpliva** tveganje

Na tej podlagi lahko tveganja razdelimo v naslednje skupine:

- **projektno tveganje** - vpliva na časovni načrt ali vire *(npr. iz ekipe odide izkušenj sistemski arhitekt)*
- **produktno tveganje** - vpliva na kakovost ali zmogljivost programske opreme *(npr. kupljena komponenta ne deluje tako, kot bi si želeli)*
- **poslovno tveganje** - vpliva na organizacijo, ki razvija ali dobavlja programsko opremo *(npr. na trg pride konkurenčni izdelek - spremembe na trgu)*

2.2.1.2 Proces upravljanja s tveganji

1. **določitev tveganj** - *opredelijo se projektna, produktna in poslovna tveganja*
2. **analiza tveganj** - *določijo se ocena verjetnosti in posledice tveganj*
3. **načrtovanje tveganj** - *pripravi se načrte za preprečevanje ali zmanjševanje učinkov tveganj*
4. **spremljanje tveganj** - *spremljamo tveganja v celotnem projektu*

2.2.1.3 Določitev tveganj

Seznam pogostih tveganj:

- **tehnološka** tveganja - *uporabljena podatkovna baza ni dovolj zmogljiva za naše zahteve, določene komponente so slabo načrtovane in vsebujejo napake zato jih ni mogoče ponovno uporabiti ...*
- **organizacijska** tveganja - *finančne težave, slabo prestrukturiranje vodstva ...*
- tveganja, povezana z **ljudmi** - *ne najdemo pravih sodelavcev z potrebnimi veščinami, potrebna usposabljanja za zaposlene niso na voljo ...*
- tveganja, povezana z **zahtevami** - *stranka ne razume vpliva spreminjanja zahtev, spremenjene zahteve zahtevajo preveliko predelavo ...*
- tveganja, povezana z **ocenjevanjem** - *podcenjevanje časa razojva, precenjevanje zmožnosti ekipe ...*
- tveganja, povezana z **orodji** - *izvorna koda, ki jo generirajo orodja za izdelavo programske kode je neučinkovita ...*

2.2.1.4 Analiza tveganj

Pri analizi tveganj se oceni **verjetnost**:

- zelo
- majhna
- majhna
- srednja
- visoka
- zelo visoka

Posledice so lahko:

- katastrofalne
- resne
- sprejemljive
- nepomembne

2.2.1.5 Načrtovanje tveganj

Pri načrtovanju moramo razmisliti o:

- strategiji izogibanja - *zmanjšanje verjetnosti*
- strategiji zmanjševanja - *zmanjšanje vpliva (posledic)*

- kriznem načrtu - *opredelitev ukrepov ob nepredvidljivih dogodkih, če se le-ti pojavijo*

2.2.1.6 Spremljanje tveganj

Redno ocenjujemo vsako ugotovljeno tveganje in tako ocenimo ali postaja manj ali bolj verjetno. Prav tako lahko ocenimo, ali so se posledice tveganja spremenile.

2.2.2 Upravljanje z ljudmi

Ljudje so najpomembnejši viri podjetju.

2.2.2.1 Dejavniki upravljanja z ljudmi

- **doslednost** - vse člane ekipe obravnavamo enakopravno, brez favoritiziranja
- **spoštovanje** - člani skupine imajo različne veščine in sposobnosti in to je treba spoštovati
- **vključitev** - vsakega člana skupine vključimo in prisluhnemo vsem mnenjem
- **poštenost** - moramo biti iskreni o tem, kaj se v projektu izvaja dobro in kaj ne

2.2.2.2 Motiviranje ljudi

Motivacija je kompleksen koncept, kjer v splošnem temelji na:

- **osnovnih potrebah** - *hrana, spanje ...*
- **osebnih potrebah** - *spoštovanje, samopodoba ...*
- **socialnih potrebah** - *sprejetost v skupino ...*

2.2.2.3 Hierarhija zadovoljitve potreb

Figure 1: Potrebe

2.2.2.4 Vrste osebnosti

- ljudje usmerjeni k nalogam - motivira jih delo samo
- ljudje usmerjeni k interakciji - motivira jih prisotnost in delo sodelavcev
- samousmerjeni ljudje - motivirani z osebnim uspehom

2.2.2.5 Učinkovitost skupine

Na učinkovitost skupine vplivajo:

- ljudje v skupini
- organizacija skupine
- tehnična in vodstvena komunikacija

2.2.2.6 Komunikacija v skupini

Na učinkovitost komunikacije vplivajo:

- velikost skupine
- struktura skupine
- sestava skupine
- fizično delovno okolje skupine

3 Načrtovanje projekta

3.1 Uvod

3.1.1 Faze načrtovanja

Imamo tri faze življenskega cikla:

- **faza predloga** - *ponudi se pogodba za razvoj ali zagotovitev sistema programske opreme*
- **faza zagona projekta** - *pripravi se načrt z določenimi viri, delitev projekta na inkremente, delitev finančnih sredstev v podjetju...*
- **med projektom** - *načrt se spreminja glede na pridobljene izkušnje*

3.1.1.1 Načrtovanje predloga

Cilj je zagotoviti informacije, ki bodo uporabljene pri določanju cene sistema naročniku.

3.1.1.2 Načrtovanje zagona projekta

Vemo več o sistemskih zahtevah, nimamo še informacij o načrtu ali izvedbi. Načrt za zagon je osnova za dodeljevanje sredstev projekta. Za agilni razvoj je ta korak neizogiben.

3.1.1.2 Razvojno načrtovanje

Projektni načrt je potrebno s časom razvoja **redno spreminjati**.

Načrt, stroške in tveganja je potrebno s časom razvoja **redno pregledovati**.

3.2 Določanje cene programske opreme

Pri določanju cene moramo upoštevati naslednje dejavnike:

- stroški strojne opreme
- stroški programske opreme
- stroški potovanj
- stroški usposablanj

- strošli dela *(plače)*

Ceno lahko **znižamo**, če želimo prodreti na novo tržno območje oz. obrdžati zaposlene za prihodnje priložnosti.

Ceno lahko **zvišamo**, če želi naročnik pogodbo s fiksno ceno in tako prodajalec poveča ceno z upoštevanjem nepričakovanih tveganj.

Zmagovalno ceno dobimo tako, da ocenimo, koliko je naročnik pripravljen plačati za trenutne **funkcionalnosti**.

Če je naročnik pripravljen plačati manj, potem lahko nekaj funkcionalnosti odstranimo in jih dodamo pozneje (*nadrgradnja*).

Dodatne stroške lahko vključimo naknadno, če pride do spremembe zahtev, ki zahtevajo po večjih stroških.

3.3 Načrtno usmerjen razvoj

Pri načrtno usmerjenem razvoju je projekt **podrobno načrtovan** s projektnim načrtom, kjer se beleži delo, ki ga je treba opraviti, kdo bo to naredil, časovni načrt razvoja in seveda tudi izdelki.

Pros & Cons načrtno usmerjenga razvoja:

Pros: zgodnje načrtvoanje omogoča, da se organizacijska vprašanja natančno upoštevajo in da se potencialne grožnje ter težave odkrijejo pred začetkom razvoja projekta.

Cons: številne stvari je treba tekom razvoja revizirati zaradi sprememb v delovnem okolju, kar je lahko zamudno.

3.3.1 Projektni načrt

Določa:

- **vire, ki so nam na voljo**
- **razčlenitev dela**
- **časovni načrt za izvedbo**

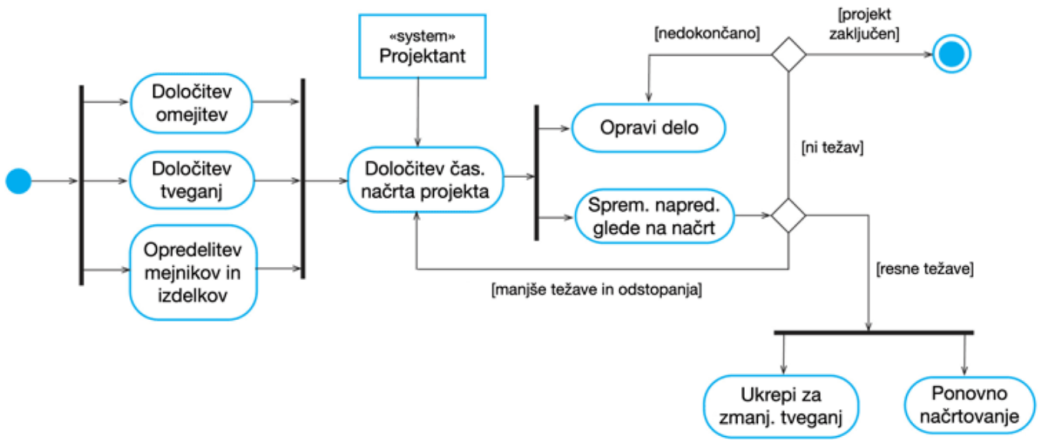
Uporaba principa **W5HH**:

- Zakaj se sistem razvija? (Why)
- Kaj bo narejeno? (What)
- Kdaj bo to narejeno? (When)
- Kdo je odgovoren za posamezno funkcijo? (Who)
- Kje se organizacijsko nahajajo? (Where)
- Kako bo delo opravljeno s tehničnega upravljalskega vidika? (How)
- Koliko sredstev je potrebnih? (How much)

Projektni načrt ponavadi vključuje:

- uvod - *predstavimo cilje in omejitve projekta*
- organizacija projekta - *organiziranje razvojne skupine, vloge oseb v skupini*
- analiza tveganj
- zahteve strojne in programske opreme
- razčlenitev dela - *razčlenitev dela na aktivnosti*
- časovni razpored projekta
- mehanizmi spremljanja in poročanja

Načrtovanje projekta je **ITERATIVNI** proces - začne se med **fazo zagona** projekta.



Slika 4.3: Proces projektnega načrtovanja

Figure 2: Proces projektnega načrtovanja

3.3.1.1 Predpostavke pri načrtovanju

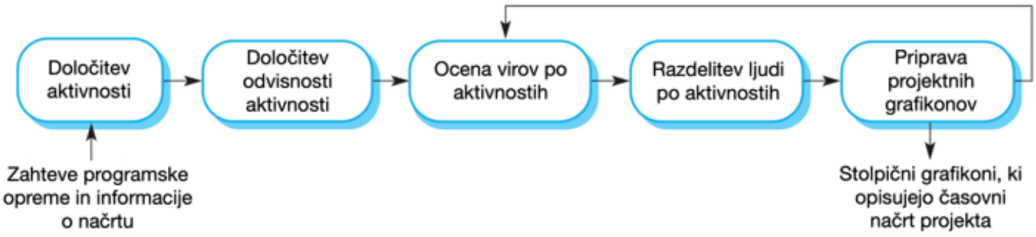
Predvsem moramo biti **realni** in **ne optimistični**.

3.3.1.2 Zmanjševanje tveganja

Če obstajajo resne težave pri načrtovanju projekta, je treba sprožiti ukrepe, da jih omilimo. To lahko povzroči, da je potrebno tudi ponovno načrtovanje projekta in ponovno posvetovanje z delodajalcem oz. stranko.

3.4 Časovno načrtovanje projekta

Pri časovnem načrtovanju moramo upoštevati, da za vsako aktivnost ocenimo, koliko časa bomo potrebovali, da jo dokončamo. Aktivnosti oz. naloge so lahko medseboj odvisne druga od druge, zato je potrebno določiti tudi v kakšnem časovnem zaporedju se bodo naloge izvajale. Oceniti je potrebno tudi vire, ki so potrebni za izvajanje vsake naloge in lahko vplivajo na časovno zahtevnost naloge. Za vsako aktivnost je treba dodeliti ljudi, ki bodo zanjo zadolženi.



Slika 4.4: Proces časovnega načrtovanja projekta

Figure 3: Proces časovnega načrtovanja

Vedno je pametno v časovno načrtovanje vključiti nekaj motečih faktorjev kot npr. nepredvidljivost, precenjena sposobnost skupine ...

3.4.1 Predstavitev časovnega načrta

Časovni načrt ponavadi predstavimo s temu namenjenimi grafikoni, kjer se najpogosteje uporabljajo vizualizacije:

- na podlagi koledarja - *stolpični diagrami, ki prikazujejo razpored aktivnosti glede na čas*
- z omrežjem aktivnosti - *prikazujemo odvisnosti aktivnosti*

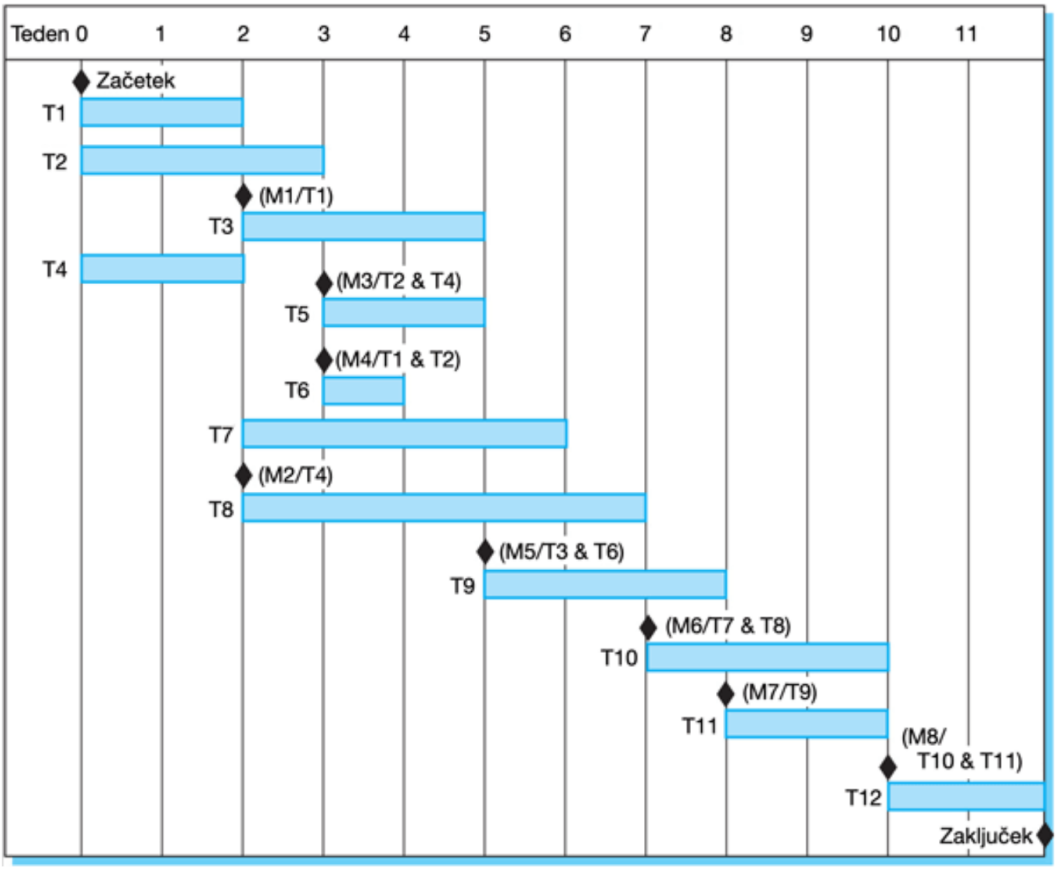
3.4.2 Projektne aktivnosti

Projektne aktivnosti (naloge) so osnovni element načrtovanja. Vsako opredelimo z:

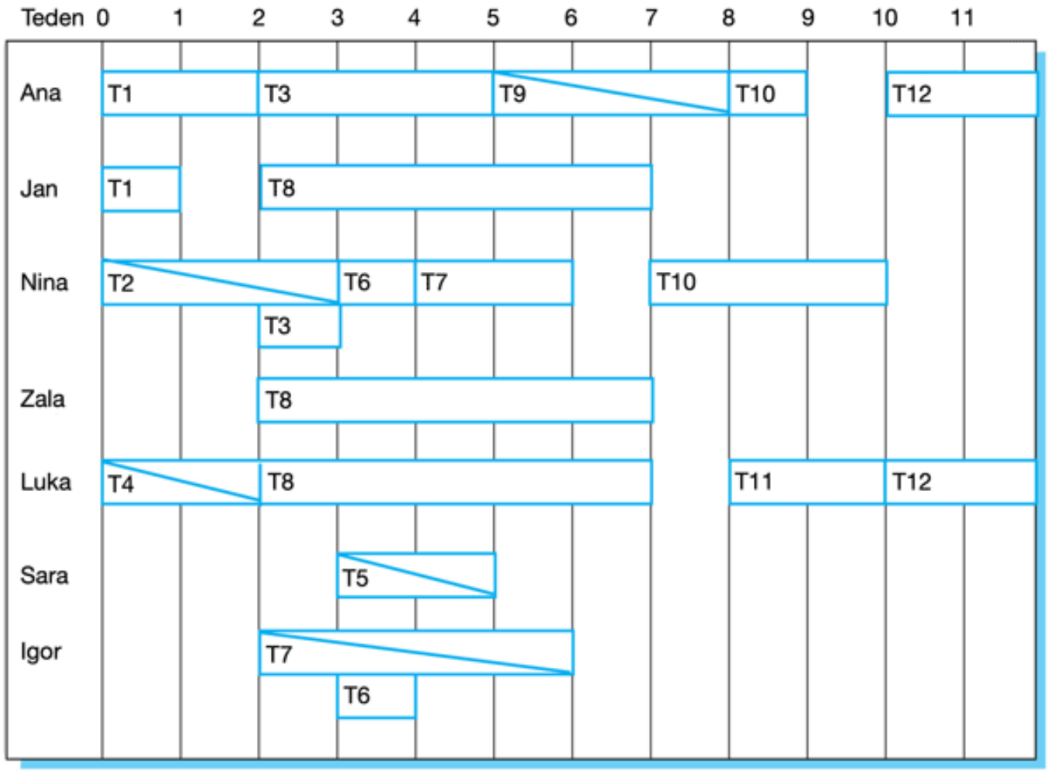
- trajanjem *(v koledarskih dneh ali mesecih)*
- oceno napora *(človek/dni ali človek/mesec)*
- rokom *(kdaj more bit končano)*
- določenim rezultatom *(lahko dokument, lahko sestanek, važen je pregled nad končanim delom) - to so ponavadi delovni izdelki, ki se dostavljajo naročniku, npr. dokument z zajetimi sistemskimi zahtevami*

Tabela 4.8: Naloge, trajanja in odvisnosti

Naloga	Napor (človek/dan)	Trajanje (dnevi)	Odvisnosti
T1	15	10	
T2	18	15	
T3	20	15	T1 (M1)
T4	5	10	
T5	5	10	T2, T4 (M3)
T6	10	5	T1, T2 (M4)
T7	25	20	T1 (M1)
T8	75	25	T4 (M2)
T9	10	15	T3, T6 (M5)
T10	20	15	T7, T8 (M6)
T11	10	10	T9 (M7)
T12	20	10	T10, T11 (M8)



Slika 4.5: Stolpčni diagram aktivnosti



Slika 4.6: Diagram porazdelitve osebja

3.5 Agilno načrtoanje

To je iterativni pristop, kjer je programska oprema razvita in dostavljena naročnikom **v iteracijah**. Kaj vključiti v posamezen **inkrement** je odvisno od napredka in prioritet stranke.

3.5.1 Faze agilnega načrtovanja

1. načrtovanje izdaje - *gledamo več mesecev v naprej in se odločamo o značilnostih, ki jih je potrebno vključiti v* ***izdajo sistema***
2. načrtovanje iteracije - *kratkoročen kontekst, kjer se osredotočamo na načrtovanje posameznega oz.* ***trenutnega inkrementa*** *projekta*

3.5.2 Pristopi agilnega načrtovanja

- **Scrum** - vodejne *preostalih* nalog pri projektu (“to-do board”)
- **igre načrtovanja** - prvotno kot del pristopa Extremnega programiranja (XP) in je odvisna od *uporabniških zgodb* in se uporablja kot merilo napredka projekta

3.5.3 Načrtovanje na podlagi uporabniških zgodb (igra načrtovanja)

Igra načrtovanja temelji na uporabniških zgodah, ki predstavljajo značilnosti, ki jih je treba vključiti v sistem.

Načrtovanje izdaje vključuje:

- zajetje uporabniških zgodb, ki odražajo **značilnosti, ki jih je potrebno uvesti v izdajo sistema**

Slika 4.7: Igra načrtovanja

Figure 4: Igra načrtovanja

- **vrstni red implementacije** značilnosti (zgodb)

3.5.3.1 Dodelitev nalog

V fazi načrtovanja nalog se uporabniške zgodbe razčlenijo v razvojne naloge, kjer velja:

- ena naloga traja približno od 4 do 16 ur
- razvijalci se sami javljajo, katere naloge bodo naredili
- vse naloge, o katerih se pogovarjamo, so referencirane na trenutno iteracijo (morajo biti končane v trenutni iteraciji)

Prednosti pristopa:

- ekipa dobi dober pregled nad delom, ki ga je treba narediti v tej iteraciji
- razvijalec ima občutek lastništva nad nalogo, kar ga dodatno motivira

3.5.3.2 Dostava programske opreme

Časovni urnik inkrementa oz. časovni rok za dostavo **se nikoli ne podaljša** - če v zastavljenem časovnem obdovju ni mogoče narediti vseh nalog, enostavno prilagodimo obseg iteracije tako, da bomo uspeli končati v zastavljenem času.

3.5.3.3 Težave agilnega načrtovanja

- odvisno je od vključenosti in razpoložljivosti stranke
- ne poznajo vse stranke koncepta agilnega načrtovanja, zato jih je težje vključiti v igro načrtovanja

3.5.3.4 Prilagodljivost agilnega načrtovanja

- dobro deluje z majhnimi, stabilnimi razvojnimi skupinami

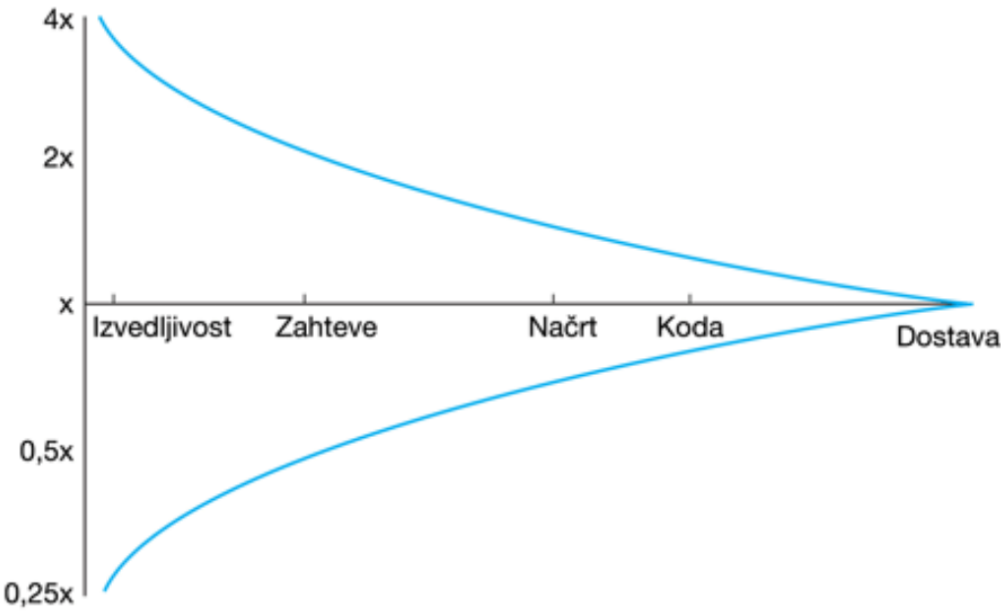
- pri velikih ali geografsko razdeljenih skupinah ali kadar se članstvo v skupini pogosto spreminja, je agilno načrtovanje praktično nemogoče

3.6 Tehnike ocenjevanja

Ko potrebujemo oceniti stroške razvoja, lahko uporabimo različne tehnike, ki temeljijo na:

- **izkušnjah** - *vodja projekta ma izkušnje z vodstvom in približno ve kako se obrača denar*
- **algoritmičnem načrtovanju stroškov** - *uporabijo se pristopi z znanimi formulami in algoritmičnimi postopki*

Pri obeh tehnikah je prisotno **široko območje napake**, če je začetna ocena x potem je lahko končna na primer nekje med $4x$ ali pa $0.25x$. Ocena postaja vse bolj točna ob napredovanju projekta in spremljanju okoliščin.



Slika 4.8: Negotovost ocenjevanja

Figure 5: Negotovost ocenjevanja

3.6.1 Tehnike, ki temeljijo na izkušnjah

Opirajo se na izkušnje vodje iz preteklih projektov, lahko so nenatančne in posledično nezanesljive, saj trenutni projekt morda nima veliko skupnega s prejšnjim.

3.6.2 Algoritmično načrtovanje stroškov

Stroški ocenjeni kot matematična funkcija (to računa vodja projekta):

napor $= A \cdot \text{obseg}^B \cdot M$

- A - *konstantni dejavnik, odvisen od lokalnih organizacijskih praks in vrste razvite programske opreme*
- obseg - ocena obsega izvorne kode
- B - predstavlja kompleksnost, ponavadi v intervalu $[1, 1.5]$
- M - dejavnik, ki upošteva procesne, proizvodne in razvojne lastnosti.

Najpogosteje uporabljena lastnost izdelka za oceno stroškov je **obseg kode**. Večina modelov je med seboj podobnih, a večinoma uoprabljajo različne vrednosti za A, B in M.

B in M sta **subjektivni** oceni ocenjevalca (npr. vodje).

Na točnost ocene vplivajo:

- vključitev ponovno uporabljivih sistemov in komponent
- programski jezik
- distribucija sistema

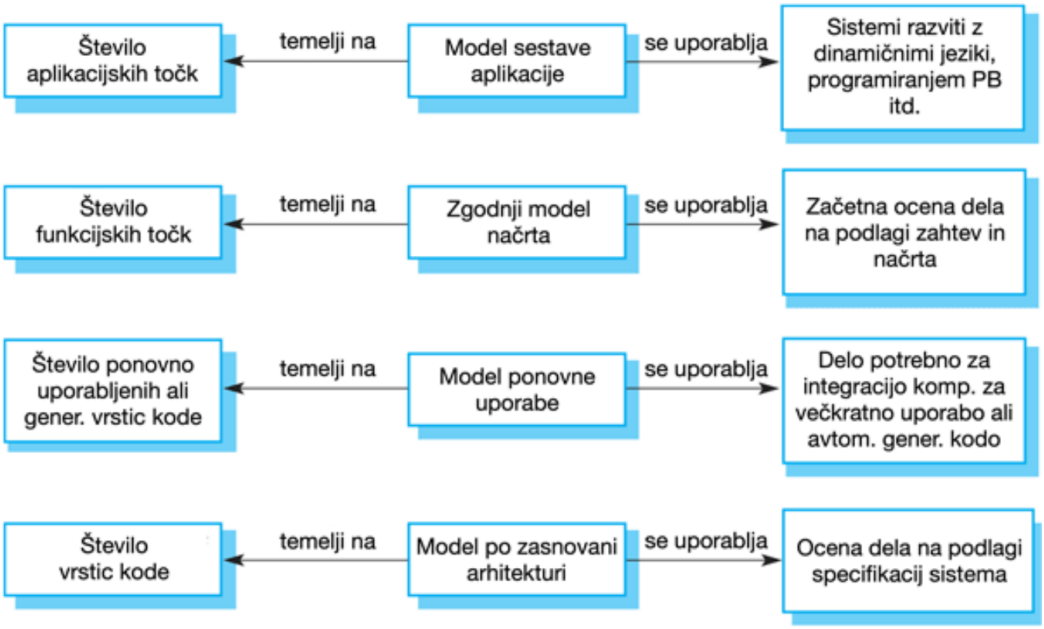
3.7 COCOMO načrtovanje stroškov

Je empirični model za načrtovanje stroškov. Razvil se je že 1981 in ima več modelov. Trenutno je najbolj znan **COCOMO II** model.

3.7.1 COCOMO II

Vključuje modele, ki omogočajo bolj podrobne ocene programske opreme:

- **model sestave aplikacije**
- **zgodnji model načrta**
- **model ponovitve uporabe**
- **model po zasnovani arhitekturi** - *se uporablja, ko je zasnovana sistemska arhitektura in je na voljo več informacij o sistemu*



Slika 4.9: COCOMO modeli ocenjevanja

Figure 6: COCOMO modeli ocenjevanja

3.7.1.1 Model sestave aplikacije

Se uporablja takrat **ko je programska oprema sestavljena iz obstoječih delov** in je v veliki meri prisotna ponovna uporaba.

3.7.1.2 Zgodnji model načrta

Ko so zahteve na voljo ampak **se načrtovanje še ni začelo**.

3.7.1.3 Model ponovne uporabe

Se uporablja za izračun potrebnega dela **pri integraciji komponent za večkratno uporabo**. Upošteva se **izvorna koda**, ki se ponovno uporabi.

Poznamo dva pristopa in sicer:

- ponovna uporaba s pristopom črne škatle - *izvorna koda se ne spreminja in se izračuna ocena potrebnega dela*
- ponovna uporaba s pristopom bele škatle - *izvorna koda se spremeni in se izračuna ocena obsega, ki je enaka številu vrstic nove izvorne kode. Nato se prilagodi ocena nove izvorne kode*

Formula za izračun je

$$\boldsymbol{Napor} = \frac{NAP}{PROD} \times \left(1 - \frac{\%\text{ ponovne uporabe}}{100}\right)$$

kjer velja:

- $\boldsymbol{Napor}$ je ocena potrebnega dela v človek/mesec,
- $\boldsymbol{NAP}$ je skupno število aplikacijskih točk v dostavljenem sistemu,
- $\boldsymbol{\%\text{ ponovne uporabe}}$ je ocena deleža ponovno uporabljene kode pri razvoju,
- $\boldsymbol{PROD}$ je produktivnost aplikacijskih točk, prikazana v tabeli 4.9.

Tabela 4.9: Produktivnost aplikacijskih točk

Izkušnje in sposobnosti razvijalca	Zrelost in sposobnost orodja CASE	$\boldsymbol{PROD}$ ($\boldsymbol{NAP}$ /mesec)
zelo slabe	zelo slaba	4
slabe	slaba	7
ustrezne	ustrezna	13
dobre	dobra	25
zelo dobre	zelo dobra	50

Figure 7: Model sestave

$$\textbf{Napor} = A \times \text{Obseg}^{B} \times M$$

kjer velja:

- $\boldsymbol{A} = 2,94$,
- $\boldsymbol{B}$ je v intervalu od $1,1$ do $1,24$, glede na izvirnost projekta, razvojno fleksibilnost, uporabljene pristope obvladovanja tveganj in zrelost razvojnega procesa,
- **Obseg** predstavlja število vrstic izvorne kode, izraženo v 1.000 (KSLOC[22]),
- $\boldsymbol{M} = PERS \times PREX \times RCPX \times RUSE \times PDIF \times SCED \times FSIL$, kjer množitelji odražajo sposobnost razvijalcev, nefunkcionalne zahteve, poznavanje razvojne platforme itd.
  - $\boldsymbol{PERS}$ - sposobnost razvijalcev,
  - $\boldsymbol{PREX}$ - izkušenost razvijalcev,
  - $\boldsymbol{RCPX}$ - zanesljivost in kompleksnost izdelka,
  - $\boldsymbol{RUSE}$ - potrebna ponovna uporaba,
  - $\boldsymbol{PDIF}$ - zahtevnost platforme,
  - $\boldsymbol{SCED}$ - potreben časovni načrt,
  - $\boldsymbol{FSIL}$ - podporne možnosti ekipe.

Figure 8: Zgodnji model

$$\boldsymbol{Napor} = \frac{ASLOC}{ATPROD} \times \frac{AT}{100}$$

kjer velja:

- $\boldsymbol{ASLOC}$ je število vrstic samodejno generirane izvorne kode,
- $\boldsymbol{AT}$ je delež samodejno generirane izvorne kode,
- $\boldsymbol{ATPROD}$ je produktivnost razvijalca pri integraciji izvorne kode.

Figure 9: Pristop črne škatle

$$\boldsymbol{ESLOC} = \frac{ASLOC}{AAM} \times \frac{1 - AT}{100}$$

kjer velja:

- $\boldsymbol{ASLOC}$ je število vrstic samodejno generirane izvorne kode,
- $\boldsymbol{AT}$ je delež samodejno generirane izvorne kode,
- $\boldsymbol{AAM}$ je prilagoditveni faktor, izračunan iz stroškov spreminjanja ponovno uporabljene kode, stroškov razumevanja, kako vključiti izvorno kodo, in stroškov odločanja o ponovni uporabi.

Nato se potrebno delo izračuna po formuli

$$\mathbf{Napor} = A \times ESLOC^{B} \times M$$

Figure 10: Pristop bele škatle

3.7.1.4 Model po zasnovani arhitekturi

Uporablja enako formulo kot **zgodnji model načrta**, vendar namesto 7 množiteljev za faktor M uporablja 17 povezanih množiteljev.

Obseg je ocenjen kot:

- št. vrstic kode, ki jih je treba razviti
- ocena enakovrednega števila vrstic nove kode
- ocena števila vrstic kode, ki jih je treba spremeniti glede na spremembe zahtev

B je odvisen od 5 dejavnikov:

- arhitektura/odprava tveganj
- prilagodljivost razvoja
- precendenčnost
- zrelost procesa
- povezanost ekipe

Ocene rangirajo od 1 do 5 kjer je 1 zelo visoka in 5 zelo nizka.

B se nato izračuna po formuli:

$$B = (\text{seštevek ocen dejavnikov}/100) + 1.01$$

Množitelji faktorja M:

- atributi izdelka (zahtevane lastnosti izdelka)
- računalniški atributi (omejitve strojne platforme)
- atributi osebja (izkušnje in zmožnosti ljudi)
- atributi projekta (posebne lastnosti projekta)

3.7.1.5 Trajanje projekta in osebje

Koledarski čas se lahko oceni z uporabo formule COCOMO II:

$$\boldsymbol{TDEV} = 3 \times \boldsymbol{Napor}^{0{,}33+0{,}2\times(B-1{,}01)}$$

kjer velja:

- $\boldsymbol{TDEV}$ je nominalni časovni razpored projekta v koledarskih mesecih, kjer se ne upošteva multiplikatorjev, ki so povezani s časovnim načrtom projekta,
- $\boldsymbol{Napor}$ je ocena potrebnega dela iz COCOMO modela,
- $\boldsymbol{B}$ je komponenta kompleksnosti programske opreme, opisana v poglavju 4.2.7.1.4 (pri zgodnjem prototipnem modelu je $B = 1$).

> **Primer**: Če je $B = 1,17$ in $Napor = 60$, potem velja, da je $TDEV = 3 \times 60^{0{,}36} = 13$ mesecev.

Potreben čas je neodvisen od števila ljudi, ki delajo na projektu.