Projektijuhtimine objektorienteeritud tarkvara tegemisel

Succeeding with Objects: Decision Frameworks for Project Management,
A. Goldberg, K. S. Rubin,
1995, pp. 542.

Objekt-orienteeritud tehnoloogiale üleminek mõjutab järgnevaid programmeerimispraktikaid:

- Toote protsessi mudeli valik
- Projekti ajakava koostamine ja selle täitmise juhtimine
- Taaskasutuse protsessi mudeli valik
- Projekti meeskonna struktuuri valik
- Tarkvara väljatöötamise keskkonna valik
- Väljaõppe plaani koostamine
- Meetodid protsesside, ressursside ja toote omaduste hindamiseks, juhtimiseks ja ennustamiseks.

Läbi on viidud 39 objekt-orienteeritud tehnoloogiat kasutanud projekti uuring (*case study*).

Raamat pakub kümmet projektijuhtimise otsustusraamistiku (*decision frameworks*), mille jaoks määratakse:

- Põhimõte (*Principle*)
- Siht (*Goal*)
- Sammud (*Steps*)
- Maksiimid (*Maxims*)

Otsustusraamistik on vajalike otsustuste organiseeritud jada.

Tarkvara inseneriteduse ajalugu:

Aasta	
1959	Doug Ross esitas *Western Joint Computer Conference*'il oma ettekande *Gestalt* programmeerimisest, milles ta rääkis käitumusliku (*behavioral*) programmeerimise perspektiividest.
1960	IEEE (end. IRE) Trans. on Human Factors in Electronics avaldas J. Licklider'i artikli "Man-Computer Symbiosis."
1966	C. Bohm'i ja G. Jacopini, CACM'is ilmunud artikkel ilma GOTO'ta programmeerimisest ("Flow diagrams, Turing machines, and languages with only two formation rules.") põhjustas hilisema struktuursete analüüsi ja disainin meetodite arengu.
1968	• OS/360 tuli välja ja Fred Brooks avaldas "Mythical Man Month" • Apollo tarkvara seati valmis järgmise aasta kuundumiseks • AT&T alustas elektrooniliste keskjaamade ESS1 ülespanekut • oli n.ö. *Summer of Love* ja San Francisco happe-rocki kultuuri tõus • kasutuses olid Sketchpad, JOSS ja Grail • polnud veel olemas DEC'i VAX'i, Xerox'i PARC'i ega Intel 4004'ja • tähelepanu oli koondunud B5000, CDC66000 ja PDP-8 • Christopher Alexander oli avaldanud "Notes on the Synthesis of Form", mis manitses disainereid otsima arhitektuuris korduvaid malle (*patterns*) • Mel Conway artikkel *Datamation*'is väitis, et loodav süsteem peegeldab teda loova organisatsiooni struktuuri hoolimata, kas seda soovitakse • Doug Engelbart esitles *Fall Joint Computer Conference*'il oma *Online System*'i (NLS), demonstreerides paari tunni jooksul kuidas multimeedia side ja interaktiivne info-haldamine võivad võimendada inimese intellektuaalseid võimeid • Edsger W. Dijkstra levitas oma tõid operatsioonisüsteemi THE kallal ja avaldas CACM'is kurikuulsa artikli GOTO'de kahjulikusest • Ole Johan-Dahl, Bjorn Myhrhaug ja Kristen Nygaard avaldasid esimese objekt-orienteeritud programmeerimiskeele Simula-67 kirjelduse • Garmisch'is peeti NATO tarkvara inseneriteaduse konverents, millel olid tulipunktis küsimused protsessist -- kuidas efektiivselt luua kvaliteetset tarkvara (ülalt-alla ja alt-üle programmeerimine, tarkvara jaotamine tasemeteks ja mooduliteks, süsteemide graduaalne loomine); Alan Perlis summeeris vestlused: 1. Tarkvarasüsteemi on parim disainida testidega segamini 2. Simulatsioon, mis vastab süsteemile esitavatele nõuetele, organiseerib süsteemi disaini 3. Läbi järgnevate disaini ja testimise tsüklite saab mudelist tarkvarasüsteem

Xerox'i Palo Alto Uurimiskeskuses (PARC) 1970 tehtud objekt-orienteeritud programmeerimise alased tööd lähtusid pedagoogikast -- kuidas õpetada programmeerimiskeele kasutamist inimestele, kuna programmeerimises nähti informatsiooni otsingu ja analüüsi viisi.
Algne objekt-orienteeritud tehnoloogia toetas:
Keerukate informatsiooni mudelite määrangute haldamist
Olemasolevate rakenduste modifitseerimist
Uute graafiliste kasutajaliideste loomist
Smalltalk-80 oli esimene keel ja programmeerimissüsteem, mis kasutas ainult objekte ja teadete saatmist.
Tänapäeval pakub objekt-orienteeritud tehnoloogia:
Kiiret rakenduste realiseerimist, hoolimata hästimääratletud nõudmiste puudumisest
Reaalse-maailma olemite kirjelduste üks-ühele vastavust suhtlevatesse (*interacting*) arvutis esitatavatesse objektidesse
Hallatavaid (*maintainable*) tarkvarasüsteeme

Tarkvaraprojektide kohta käivad maksiimid (Tom Gilb):

- Õnnetused ei juhtu juhuslikult, vaid tänu juhtimisele

- Kui sa ei tea, mida sa teed, ära tee seda suures mastaabis
- Sa võid unustada mõned kriitilised faktorid, kuid nemad ei unusta sind
- Olgu pinge kuitahes suur, ära kunagi anna lubadusi, mida sa ei saa täita

Süsteemi arhitektuuri põhimõtted:

- Süsteemi struktuur peegeldab teda loonud grupi kultuuri ja organisatsiooni
- Kõik projekti-juhtimise otsustused tuleks siduda projekti eesmärkide ja sihtidega
- Kogu informatsioon projekti kohta peaks olema kättesaadav kõigile meeskonna liikmetele

Projektijuhtimise otsustusraamistikud:

- Sihtide seadmine objekt-orienteeritud tehnoloogia kasutamiseks
  - määra projekti sihid ja eesmärgid
  - otsusta, kuidas ja millist kasu saab projekt objekt-orienteeritud tehnoloogia kasutamisest
  - seo ennast objekt-orienteeritud tehnoloogiaga
- Objekt-orienteeritud tehnoloogia kasutamine tarkvara väljatöötamise protsessis

Raamistik	*Põhimõte*	*Siht* *Sammud*
Projekti sihid ja eesmärgid (*Project goals and objectives*)	Et teada, millal oled kohale jõudnud, tuleb teada, kuhu lähed	Määratle sihid ja eesmärgid ja valmista ette otsuste tegemise protsess nii, et kõik otsused oleksid jälgitavad nende sihtide ja eesmärkideni
		• Selgita ärimissioon ja äriprotsessid (äriprotsesside korrastamine -- *Business Process Reengineering*, motod) • Selgita tarkvara väärtused äri- ja meeskonna jaoks -- mida hindate: - tootes - protsessis - ressurssides • Formuleeri soovitud situatsioon (sihid ja eesmärgid toodetele, protsessidele ja ressurssidele) • Hinda olemasolevat olukorda (Tarkvarateaduse instituudi (SEI) välja töötatud *Capability Maturity Model*)
Tulud objekt-orienteeritud tehnoloogiast (*Benefits of object-oriented technology*)	Objekt-orienteeritud tehnoloogia pole eesmärk, vaid vahend eesmärgi saavutamiseks	Ole realistlik ootustes, kuidas objekt-orienteeritud tehnoloogia võib aidata püstitatud eesmärke saavutada
		• Selgita kas ja kuidas objekt-orienteeritud tehnoloogia aitab saavutada seda, mida sa hindad tarkvara tegemises • Määratle projektid, millede sihte on vaja objekt-orienteeritud tehnoloogia abil saavutada
Seo ennast objekt-orienteeritud tehnoloogiaga (*Make an initial commitment to objects*)	Uue tehnollogia edukas sisseviimine vajab hästi ressurssidega varustatud demonstratsiooniprojekti läbi viimist	Vali esimene projekt ja teosta see edukalt, nii et see demonstreeriks objekt-orienteeritud tehnoloogia võimaliku mõju teie organisatsioonis ja oleks aluseks strateegilistele otsustele
		• Saa esmane haridus objekt-orienteeritusest • Saa algne juhtkonna nõusolek kasutada objekt-orienteeritud tehnoloogiat • Vali algne protsessi mudel ja tarkvara väljatöötamise keskkond • Vali ja sea üles pilootprojekt • Hinda pilootprojekti tulemust

Protsessi mudel esitab tegvuste järjekorda ja sisaldab reegleid iga tegevuse sisendite ja väljundite kohta. Protsessi mudelid on tähtsad, kuna nad vastavad küsimustele:

- *Planeerimisest* -- mida peab tegema, et saavutada sihte ja eesmärke
- *Õigustest* -- kuidas me saame mõjutada tulemuse saavutamist
- *Ennustamisest* -- kuhu me liigume
- *Hindamisest* -- kus me protsessis oleme ja miks

- *Jälgitavusest* -- kuidas tulemused saavutati

Strateegia tarkvara tegemiseks kirjeldab ja juhib laias plaanis tegevuste ja protsesside läbiviimist, ilma kasutatavate tehnikate või tehtavate tööde detailidesse tungimata.
Tegevused (*activities*) on sammud, mida võetakse ette mingi eesmärgi saavutamiseks. Protsessi mudelis on nad:

- *Analüüs* -- Probleemi mõistmine tulemusele esitatavate nõudmiste kujul
- *Disain* -- Nõudmisi rahuldava lahenduse arhitektuuri ja detailide väljatöötamine
- *Realiseerimine* -- Disaini väljendamine arvutil täidetaval kujul
- *Dokumenteerimine* -- Analüüsi, disaini ja realisatsiooni iseloomulike joonte kirjeldamine
- *Integreerimine* -- Iseseisvalt loodud osade kombineerimine tervikuks
- *Testimine* -- Realisatsiooni vigade väljaselgitamine
- *Hooldamine* -- Süsteemi parendamine ja muutmine

Meetodid on tegevuste läbiviimise erinevad viisid.
Tööd (*tasks*) on meetodi konkreetsed kasutused antud protsessi mudelis. Iga töö on määratud soovitava väljundi, töö suhte teistesse töödesse ja vajatavate ressursside kirjeldustega.

Protsessi mudelid:

- Kose mudel (*waterfall model*) -- Lineaarselt täidetakse järgnevaid tegevusi
  - Analüüs -- Nõuete iseloomustus
  - Disain -- Disaini kirjeldus
  - Realisatsioon -- Verifitseeritult nõuetele vastav kood
  - Integreerimine -- Verifitseeritult koos töötavad osad

  Kriitiline on nõuete stabiilsus ja õigsus
- Spiraalne mudel (*spiral model*) -- mudel sisaldab prototüüpimist
  Iga tsükkel algab sihtide määratlemisega ja lõpeb tulemuste hindamise ja järgneva tsükkli planeerimisega
  Kriitiline on riskide äratundmine ja haldamine
- Rekursiivne/Paralleelne mudel (*recursive/parallel model*) -- põhimõtteks on probleemi dekomponeerimine alamprobleemideks

Erinevate projektide korral võivad sobivaimaks osutuda erinevad mudelid.

Objekt-orienteeritud tarkvara tegemise strateegiana on sageli sobivaim protsessi iteratiivne rakendamine.
Prototüüpimise eesmärgiks on saada otsustuste tegemiseks piisavat informatsiooni.
Lisastrateegiad objekt-orienteeritud tehnoloogia rakendamisel:

- Dekomponeerimine
- Muutuste integreerimine ja haldamine
- Dokumenteerimine
- Kvaliteedi tagamise strateegiad
  - testid (*test cases*)
  - testidega kaetuse analüüs
  - stsenaariumite salvestus ja mahamängimine
  - koodi lugemine ja ülevaatused

Vigade dokumenteerimine:
Juhtum on soovimatu või ootamatu tulemus, mis on saavutatud testimisel või kasutuses.
Juhtumit atribuudid on:

- Unikaalne identifikaator
- Kirjeldus, kunas ja kuidas juhtus
- Juhtumi registreerija
- Juhtumis osalenud toote identifikatsioon
- Riist ja tarkvara konfiguratsioon
- Lisainformatsioon
- Kuupäev (ja kellaaeg)
- Raskusaste

- Tähtsus
- Lahenduse eest vastutav
- Olek
- Põhjus
- Lahendus
- Lahenduse kuupäev (ja kellaaeg)
- Kommentaarid
- Sarnased juhtumid

Viga on juhtum, mille lahendamine nõuab inseneritagavust.
Vea atribuudid on:

- Unikaalne identifikaator
- Viited juhtumitele, mis paljastasid vea
- Vea registreerija
- Vigase toote (ja toote osa) kirjeldus
- Tegevus (tarkvara väljatöötamisel) kus viga tekkis
- Vea põhjus
- Tähtsus
- Lahenduse eest vastutav
- Olek
- Viide muutuste kirjeldustele, mis parandavad vea
- Lisainformatsioon
- Kommentaarid

Vea kõrvaldamiseks on vajalikud **muutus**ed.
Muutuse atribuudid on:

- Unikaalne identifikaator
- Muutuse kirjeldus
- Muudetava toote (ja toote osa) kirjeldus
- Kommentaar, kas ja kuidas muudeti korduvalt kasutatavat osa
- Muutuse tüüp
- Muutuse kuupäev (ja kellaaeg)
- Muutuse eest vastutav
- Muutuse hind (töötundides)
- Olek
- Viited muudetud komponentidele
- Viited teistele vajalikele muutustele ja muudele muutusest tulenevatele tegevustele
- Kommentaarid
- Viited vigadele, mida muutus kõrvaldab
- Kokkuvõte -- mida tehti ja miks

Süsteemi omadused kirjeldavad, mida süsteem peab tegema. Need võivad olla välised (kasutajale nähtavad) ja sisemised.
Tähtaeg (etapitähis?) (*milestone*) on sobiv punkt ajakavas, millega organisatsioon markeerib edasiliikumist.
Tüüpilised tähtajad:

- Prototüüp lõpetatud
- Prototüübi uuring ja anlüüs lõpetatud
- Toote *alfa* (sisemise) versiooni väljalase
- Toote muutmine *alfa* versiooni vastukajade alusel
- Toote *beeta* (välimise) versiooni väljalase (installatsioon klientide juures)
- Toote muutmine *beeta* versiooni vastukajade alusel
- Toote ametliku versiooni väljalase testimiseks
- Toote ametliku versiooni väljalase installeerimiseks

Töö on väikseim ühik projekti ajakavas, mis aitab kaasa etapitähise saavutamisele.

Töö atribuudud on:

- Unikaal identifikaator (sageli informatiivne nimi)
- Töö sihi kirjeldus
- Tulemuse eesmärgi kirjeldus
- Alamtööde nimistu
- Tulemuste kirjeldus
- Tähtsus temaga seotud tähtaja saavutamisel
- Süsteemi omadused, mida töö loob
- Protsessi mudeli tegevus, mida töö toetab
- Tööga seotud riskide kirjeldus
- Töö tehtavuse hinnang (tõenäosus)
- Planeeritud alguskuupäev
- Eeldatav kestvus
- Vajalikud ressursid (mis eraldatakse)
- Tegelikud alguse ja lõpu kuupäevad
- Tegelikud ressusrsid (mida oli vaja)
- Kõrvalekallete kirjeldused

Planeerimine määramatuses/teadmatuses:

- Formuleeri selgelt, mis on teada ja mis pole teada
- Formuleeri selgelt, mida tehakse teadmatuse kõrvaldamiseks
- Kindlusta, et kõik varased tähtajad täidetakse
- Valmistu ümberplaneerima

Dokumenteerimise plaan (osa U.S. DoD 2167A standardist):

- SRS -- Software Requirements Specification
  Presentation of Data, Function, and Control Flow
- STP -- System Test Plan
- SDD -- Software Design Document
- STD -- System Test Document
- STR -- Software Test Report

Taaskasutamist (korduvkasutamist) (*reuse*) ei saa ennustada, seda saab ainult tõdeda tagantjärgi.

Oma erialal tuleb taaskasutatavad väärtused ise luua, väljaspoole om aeriala jäävad taaskasutatavad väärtused tuleb osta.

Taaskasutatava osa teeki kandmisel STAR (Software Technology for Adaptable, Reliable Systems) projektis nõutakse:

- Osa kirjeldus
- Teeki andja andmed
- Osa komponendid
- Osa ajalugu
- Osa suhted
- Osa atribuudid
- Piirangud
- Õiguste loetelu
- Tarkvara
- Mitmesugused märkused
- Väljalasked
- Meedia kirjeldus
- Meedia

Meeskond on selgelt määratletud eesmärkide nimel koordineeritult koostöötavate inimeste grupp. Meeskonna struktturi all mõistame meeskonna rollide jaotust, mis mõjutab meeskonna liikmete vahelist suhtlemist.
Meeskonna struktuuritüübid:

- Hierarhiline meeskond (lamedates organisatsioonides on otsuste tegemine segatud tegeliku tööga) -- peaprogrammeerija meeskond. Mida suurem on meeskond, seda tõenäolisemalt on ta hierarhiline.
- Egata meeskond -- sihid seatakse konsensusega ja grupi juhtimine on ringlev ülesanne. Inimesed jagavad oma tööd avatult (avalikult). Tõenäoliselt väike meeskond.
- Sõjaväeline meeskond -- alternatiivne hierarhilisele mudelile, eripäraks on inimeste väljaõpetamine tasemeni, kus nad võivad oma volituste piires iseseisvalt otsuseid vastu võtta ja tegutseda. Kombineerib hierarhilise mudeli vastutuse ja volituste täieliku delegeerimisega.
- Komandöri meeskond -- tuletus sõjaväelisest mudelist, kus juht seab üles strateegia ja identifitseerib kriitilised probleemid.

Personali säilitamiseks:

- Tekita neis edukuse ja õnnestumise tunne
- Las neil olla lõbus -- väljatöötajatele tavaliselt meeldib uus tehnoloogia
- Pakku võitlusvõimelist palkka -- ka väljatöötajad peavad sööma

Tarkvara väljatöötamise keskkond (*software development environment*) on tarkvarasüsteemide väljatöötamiseks vajalik meetodite, keelte, teekide, tööriistade ja andmebaasiteenuste hulk.

Analüüs on mõistete, tehnikate ja sammude hulk probleemi mudeli ehitamiseks. Analüüs keskendub sellele, mida süsteem peab tegema, mitte kuidas ta seda peab tegema.
Disain koosneb kahest tegevusest: arhitektuuriline disain ja detaildisain. Arhitektuuriline disain koosneb strateegilistest otsustest, kuidas süsteemi funktsionaalsus on jagatud süsteemi komponentide vahel, kuidas komponendid üksteisesse suhtuvad ja milline on juhtimisvoog. Detaildisain koosneb taktikalistest otsustest, nagu algoritmide ja andmestruktuuride valik vastavalt sihtidele effektiivsuses ja mäluvajaduses.

Tööriista valik:

- Tööriist peab toetama meetodi igat sammu
- Tööriist peab haldama kogu informatsiooni, mida meetod nõuab
- Tööriist peab haldama haldama infomahtu, mis vastab teie probleemi suurusele
- Tööriistas peab sisalduma mehhanism vasturääkivuste leidmiseks kogutud informatsioonis
- Tööriist peab võtma enda peale niipalju kujundusfunktsioone (diagrammide laotus), kui võimalik
- Tööriist peab vastavalt vajadusele toetama mitut samaaegset kasutajat
- Tööriist peab olema võimeline genereerima täidetava algse süsteemi ja dokumentatsiooni

Andmebaasi valik:
Relatsiooniline andmebaas on parem, kui:

- Andmetüübid on lihtsad ja nendevahelised suhted on lihtsad
- Transaktsioonid on lühikesed
- Navigeerimine (*table join*) üle andmete pole vajalik
- Skeemi evolutsioon on aeglane

Objekt-orienteeritud andmebaas on parem, kui:

- Andmetüübid on keerulised ja määratud kasutaja poolt
- Transaktsioonid kestavad kaua
- Eksperimenteerimise toetuseks on vaja andmetest mitut versiooni
- Tuleb kasutada viite järgi navigeerimist

Objekt-Orienteeritud Disain

Strateegilised otsused määravad:

- *Arhitektuuri* -- kuidas eristatakse ja modelleeritakse alamsüsteeme
- *Tarbijaliides* -- kuidas kasutaja suhtleb süsteemiga
- *Informatsiooni haldamine* -- kuidas tegutseda persistentse infoga
- *Side* -- kuidas toimub süsteemi osade ja/või süsteemi väliste osade vaheline side
- *Juhtimine* -- kuidas süsteem otsustab, mida teha (järgmiseks)

Taktikalised otsused määravad:

- Algortimide valiku
- Vahetarkvara (*middleware*) valiku (adapteerimaks ühte objektide komplekti teisega)
- Andmestruktuuride valiku (effektiivsuse ja mälukasutuse seisukohast parimate)

Väljaõppe meetodid:

- Ettevalmistatud ettekanded
  - juhtide seminarid
  - konverentside ettekanded
  - töö klassiruumis
- Juhendamine (*mentoring*)
  - juhendaja näitab kuidas teha, õpilane vaatab
  - õpilane ja juhendaja teevad koos (meister - õpipoiss)
  - õpilane teeb, juhendaja jälgib
- Ise õppimine
  - lugemisklubi (näit. Loetakse kindlat raamatut -- kõik loevad nädala jooksul mingi osa ja seejärel arutatakse see üheskoos läbi)

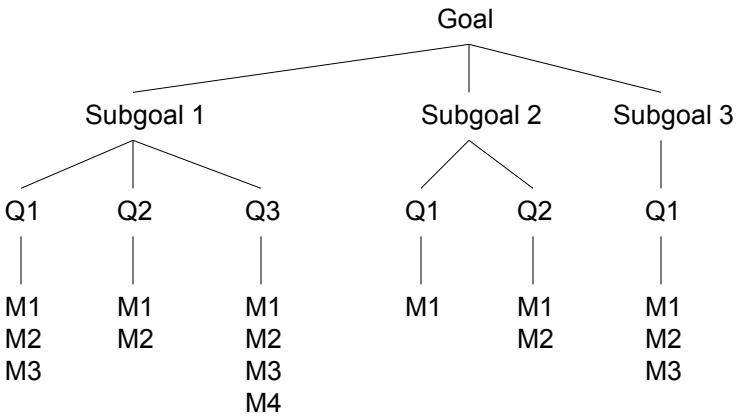
Mõõt (*measure*) seob arvu olemi mingi kindla omadusega (seega, et mõõta, peame otsustama, milline omadus meid huvitab). On olemas kahte liiki omadusi: sisemised ja välimised. Sisemiste omaduste korral mõõdetakse toodet, protsessi või ressursi ja välimiste korral seda kuidas need suhtuvad oma keskkonda.

Toode	*Sisemine omadus*	*Välimine omadus*
Spetsifikatsioon	suurus, taaskasutus (*reuse*), modulaarsus, liiasus (*redundancy*), funktsionaalsus, süntakstiline õigsus	arusaadavus, turvalisus (*safety*), hallatavus (*maintainability*), jälgitavus (*traceability*)
Disain	suurus, taaskasutus, modulaarsus, seostatus (*coupling*), ühtsus (*cohesiveness*), funktsionaalsus	kvaliteet, keerukus, turvalisus, hallatavus, jälgitavus
Kood	suurus, taaskasutus, modulaarsus, seostatus, funktsionaalsus, algoritmiline keerukus, juhtimisvoo struktuursus	töökindlus (*reliability*), kasutatavus (*usability*), turvalisus, hallatavus, jälgitavus
Testandmed	suurus, katvuse tase (*coverage level*)	kvaliteet

Protsess	*Sisemine omadus*	*Välimine omadus*
Spetsifikatsioonide tegemine	aeg, töökulu (*effort*), vajalike muudatuste arv	kvaliteet, maksumuse stabiilsus
Detaildisain	aeg, töökulu, leitud vigade (*faults*) arv	maksumuslik effektiivsus (*cost-effectiveness*), maksumus
Testimine	aeg, töökulu, leitud vigade (*bugs*) arv	maksumuslik effektiivsus, stabiilsus, maksumus

Ressurss	*Sisemine omadus*	*Välimine omadus*
Personal	hind	tööviljakus
Meeskond	suurus, suhtlemise tase, struktuursus	tööviljakus, kvaliteet
Tarkvara	hind, suurus	kasutatavus, töökindlus
Riistvara	hind, kiirus, mälu maht	töökindlus
Kontor	suurus, temperatuur, valgustus	mugavus, kvaliteet

Goal-Question-Metric (GQM) -- viis kasutuskõlblike meetrikate saamiseks. GQM on ülalt alla lähenemine, kus peamine siht dekomponeeritakse lihtsamateks alamsihtideks ja iga alamsihi jaoks määratakse hulk küsimusi ning iga küsimuse jaoks hulk mõõte, mis annavad andmeid, et antud küsimusele vastata.



Otsustusraamistike küsimused:

Raamistik	*Küsimused*
Toote protsessi mudel (*Product Process Model*)	• Milline on süsteemi suurus? • Kui palju on süsteemis vigu (*defects*)? • Millised vead on süsteemis? • Kui keerukas on süsteem? • Milline on keskmine vastuseaeg muudatuspäringule? • Milline osa süsteemist on muutunud uue funktsionaalsus lisamisel? • Milline osa süsteemi funktsionaalsusest ja realisatsioonist on dokumenteeritud? • Milline osa süsteemist on kaetud testidega? • Millist on taasklasutuse viis ja kui palju seda projekti ajal saavutati? • Palju asju (*artifacts*) läbis ja ei läbinud taaskasutuse teste?
Planeerimine ja juhtimine (*Planning and Controlling*)	• Milline on mingi konkreetse töö tegemise ennustatav aeg? • Milline osa ennustatud töömahust on tehtud? • Milline osa ennustatud töömahust jääb teha? • Milline on praegune plaanist kõrvalekalle? • Milline on keskmine kõrvalekalle tööde ennustatud kastvusest? • Palju muutusi tehti tööde sõltuvustes? • Kui palju aega kulus ümberplaneerimisele?
Taaskasutuse protsessi mudel (*Reuse Process Model*)	• Milline on teekides olevate väärtuste (*assets*) võimalik taaskasutuse tase? • Milline osa mingis projektis tehtud asjadest on taaskasutatavad teistes projektides? • Kui tihti on taaskasutatavat väärtust edukalt taaskasutatud? • Kui palju aega on kulunud taaskasutatavate väärtuste sertifitseerimisele, klassifitseerimisele, salvestamisele, leidmisele, aru saamisele ja haldamisele? • Milline osa taaskasutatavaist väärtusist omab piisavat dokumentatsiooni, et väärtust taaskasutavat väljatöötajat aidata? • Milline osa taaskasutatavist väärtusist on testitud vastavalt organisatsiooni testimisreeglitele? • Milline osa taaskasutatavist väärtusist on sertifitseeritud? (sertifitseerimine on protsess, milles määratakse taaskasutatava väärtuse vastavus mingile kvaliteedi tasemele)
Meeskonna struktuur (*Team Structure*)	• Paljude teiste meeskonnaliikmetega peab iga meeskonnaliige oma ülesannete täitmiseks suhtlema? • Palju erinevaid ülesandeid on määratud meeskonna igale rollile? • Milline osa meeskonna liikmetest omab piisavat vilumust aladel, mis on vajalikud nende (meeskonnaliikmete) rollide täitmiseks? • Milline osa töödest on määratud meeskonnaliikmetele, kellel on vastutus nende töö elluviimise eest? • Kui suur on iga väljatöötaja tööviljakus? • Kui suur on projekti meeskonna tööviljakus?
Tarkvara väljatöötamise keskkond (*Software Development Environment*)	• Kui palju tööd on kulutatud (saab kulutatud) tarkvara väljatöötamise keskkonna hindamise peale? • Kui palju maksis (saab maksma) tulemuse portimine teisele platvormile?

	• Milline osa soovitud mõistetest on analüüsi-disaini meetodis? • Milline on valitud keelte, tööriistade, andmebaasi ja meetodite osakaal tehnoloogia turul?
Väljaõpe (*Training*)	• Milline osa väljatöötajatest leidis, et väljaõpe oli effektiivne vajalike oskuste siirdamises? • Kui kaua võtab aega meeskonnaliikmete väljaõpe igal alal igale vilumuste tasemele? • Milline osa kursuste materialidest ja õpetajatest on adekvaatsel tasemel, et saavutada kursuse eesmärgid? • Milline osa meeskonnaliikmetest leiab, et väljaõppe viis ja vorm vastavad nende vajadustele?

Suuruse, tööviljakuse ja töömahu mõõtmine:
Suurus: koodiridade arv, abstraktsioonide arv.
Funktsionaalsuse mõõt: Funktsioonipunktid.

Süsteemi tüübi identifitseerimine
Süsteemi piiride identifitseerimine
Parandamata funktsioonipunktide loendamine
3.1. Andmefunktsioonide tüüpide (liidesed, sisemised failid) loendamine
3.2. Transaktsioonide tüüpide (sisendid, väljundid, päringud) loendamine
Parandusväärtuste määramine
Parandatud funktsioonipunktide arv

Kvaliteedi mõõtmine:
Kvaliteedi mallid. kvaliteedi mõõtmine lähtudes vigadest.
Keerukuse mõõtmine:

- Ühtsus (*cohesion*) -- millisel määral on üksikute süsteemi komponentide poolt tehtavad tööd seotud
- Seostatus (*coupling*) -- süsteemi komponentide üksteisest sõltumatus
- Struktuursus (*structuredness*) -- millisel määral on süsteem organiseeritud äratuntavate mallide järgi

Taaskasutuse mõõtmine:
Põhimõtteline (olemuslik) taaskasutatavus, valdkonnast sõltuv taaskasutatavus, organisatsiooniline taaskasutatavus.

Ebaõnnestumised objektidega:

Ebaõnnestumiste liigid	*Ebaõnnestumisele viinud otsused*
Sihtide ja eesmärkidega seotud ebaõnnestumised	• Oodata tulemusi projektist, mis pole projekti sihiks ega eesmärgiks • Paigutada objekt-orienteeritud tehnoloogia projekti sihiks, mitte vahendiks sihi saavutamisel • Katse võtta kasutusele korraga kõik objekt-orienteerituse aspektid • Puudulik tarkvara väljatöötamise seisukorra hinnang
Toodete ja toote protsessi mudeliga seotud ebaõnnestumised	• Oodata, et üks protsessi mudel töötab kõikide projektide jaoks • Takistada analüüsi mahu kasvu kodeerimise arvel • Vältida võimalust õppida prototüüpidest • Käsitleda prototüüpi tootena • Teha kvaliteedi hindamsest väljatöötaja teine töö
Planeerimisega seotud ebaõnnestumised	• Detailsete ajakavade koostamine, mis ei arvesta puuduva informatsiooniga • Detailsete ajakavade koostamine, milles pole väljatöötajate paralleelsust • Progressi arvestamine kulutatud ressursside põhjal • Progressi arvestamine koodiridade ja klasside arvu põhjal
Taaskasutuse protsessi mudeliga seotud ebaõnnestumised	• Taaskasutuse juhtimise vajaduse ignoreerimine • Taaskasutatavate ressursside ootamine esimesest projektist • Meelitada väljatöötajad taaskasutama
Meeskonna struktuuriga seotud ebaõnnestumised	• Meeskonna struktuur ei vasta projekti protsessi mudelile ja tulemusele • Inimeste juhuslik valik meeskonna rollidesse • Ebapiisavate oskustegainimeste valik meeskonna rollidesse • Asetada projekti juht arendustegevuse kriitiliseks ressursiks • Jagada vastutust ilma volitusteta • Sundida väljatöötajaid kasutama objekt-orienteeritud tehnoloogiat
Tarkvara väljatöötamiskeskkonnaga seotud ebaõnnestumised	• Näha keeltes tehnoloogiat • Toetuda vaid tarkadele inimestele ilma toetavate ressurssideta • Valida tarkvara väljatöötamie keskkond, mis ei luba testida installeeritavas konfiguratsioonis
Väljaõppega seotud ebaõnnestumised	• Eeldada, et inimesed õpivad tegutsedes (väljaõpe puudub) • Õpetada kõike objektdest ühekorraga • Segada ära mõistete teadmist vilumusega
Mõõtmisega seotud ebaõnnestumised	• Mitte koguda mõõtmisandmeid • Koguda andmeid kõige kohta • Koguda andmeid teadmata, mida sellega teha

Meeskonnaliikmete rollid:

- Analüütk prototüüpija -- teeb töötavaid prototüüpe analüüsi faasis, töötab koos analüütikuga
- Disaini prototüüpija -- teeb töötavaid prototüüpe disaini faasis, töötab koos disaineriga
- Objekt-tehnoloogia treener -- üldine ressurss
- Objekt-tehnoloogia ekspert -- annab nõu ja strateegilisi soovitusi
- Raamistiku disainer -- määrab rakenduse ehitamiseks vajalike üldiste osade arhitektuuri
- Taaskasutuse administraator -- vastutab taaskasutatavate väärtuste leidmise ja hankimise eest
- Taaskasutuse hindaja -- hindab, kas tarkvarakomponendid on taaskasutuseks sobivad

- Taaskasutuse insener -- loob ja hooldab taaskasutatavaid väärtusi, ning uuendab väärtuste taaskasutajaid
- Taaskasutuse teegihoidja -- vastutav taaskasutatavate väärtuste sertifitseerimise, klassifitseerimise ja salvestamise eest organisatsiooni taaskasutatavate väärtuste teeki
- Taaskasutuse juht -- planeerib ja juhib taaskasutuse protsessi

Objekt-orienteeritud tehnoloogia tarkvara tegemises kasutamise otsustusraamistikud:

Raamistik	*Põhimõte*	*Siht* *Sammud*
Toote protsessi mudel (*Product process model*)	Inkrementaalne otsustusprotsess, arendus, testimine ja integreerimine annavad effektiivse tulemuse	Vali projekti tegevused, nende järgnevus ja meetodid vastavalt projekti sihtidele ja organisatsiooni juhtnööridele • Lepi kokku maksiimides • Verifitseeri tarkvara arenduse sihid ja eesmärgid • Vali projekti strateegiad, tegevused ja nende järjestus • Vali meetodid igale projekti tegevusele
Projekti plaan ja juhtimine (*Project plan and control*)	Planeerimine ja täideviimine on vahelduvad tegevused, kus plaane tehakse, viiakse läbi ja tulemusi kasutatakse uute plaanide tegemiseks	Tööta välja projekti ajakava (*schedule*) ja viis selle täitmise kontrollimiseks/juhtimiseks, mis suurendab juhtkonna usaldust ja vastab nende ootustele • Määratle vajalikud tähtajad • Määratle süsteemi omadused • Määratle tööd • Hinda iga töö maksumust (COCOMO) *maksumus* = **a** (*suurus*) ^{b} *maksumus*$_{parandet}$ = *maksumus* (Π *faktorid*) • Arvesta maksumust (jaga andmed maksumuse kohta kategooriatesse) • Jälgi ja juhi projekti kulgu
Taaskasutuse protsessi mudel (*Reuse process model*)	Taaskasutatavad väärtused on organisatsiooni strateegilised tulemused/tooted (*products*)	Sea üles struktuur, kuidas planeerida ja hallata taaskasutatavate väärtuste kogumist, jagamist ja ülalpidamist terves organistasioonis • Määratle taaskasutamine - mida taaskasutada - kuidas taaskasutada • Sea üles taaskasutatavate väärtuste teegi täitmise protsess - määratle taaskasutatavate väärtuste kategooriad - hangi, sertifitseeri, klassifitseeri esitle ja salvesta taaskasutatavad väärtused • Sea üles taaskasutatavate väärtuste jagamise protsess • Sea üles taaskasutatavate väärtuste hooldamise protsess
Meeskonna struktuur (*Team structure*)	Meeskonna struktuur peegeldab protsessi mudeli vajadusi ja soovitava tulemuse struktuuri	Identifitseeri eesmärgid (*purposes*), rollid, juhtimisstiilid ja suhtlemiskanalid, mis moodustavad organistasiooni eesmärkide saavutamiseks vajaliku meeskonna

		• Otsusta, milliseid meeskondi on vaja • Identifitseeri rollid igas meeskonnas - rollid on määratud tegevuste poolt - lõppkasutaja peab olema kaasatud • Otsusta iga meeskonna juhtimise stiil - juhid teenendavad meeskondi - vastutus nõuab volitusi - juhid peavad juhtima (nad ei saa olla väljatöötamisel vajalik ressurss) - väljatöötajad otsustavad tähtajad • Otsusta, kuidas toimub suhtlemine - ükski inimene ei peaks suhtlema rohkem kui 8 inimesega - meeskonnad peavad olema jaotatud alammeeskondadeks, kus on 6 - 8 inimest • Leia meeskonnaliikmed - kõige tähtsamad on kogemused
Tarkvara väljatöötamise keskkond (*Software development environment*)	Meeskonna liikmed töötavad koos kõige effektiivsemalt, kui on olemas strateegia meetodite, tööriistade, teekide, keelte, andmebaaside ja üle andmise (*delivery*) koordinatsioon	Sea üles selline väljatöötluskeskkond kõikide meeskonna liikmete jaoks, et meeskond saaks rakendusi luua, ülalpidada (*maintain*) ja üle anda
		• Hinda praegust korduvkasutuse situatsiooni • Mõista väljatöötatavat tarkvara • Määratle kasutajaliidese iseloom • Loo programm disaini hindamiseks
Meeskonna väljaõppe kava (*Team training plan*)	Uue tehnoloogiaga õnnestumine oleneb kõikide meeskonnaliikmete väljaõppest selle tehnoloogia kasutamiseks	Koosta väljaõppe kava, et meeskonna liikmed omandaksid vajalikud oskused, mida nad vajavad uute protsesside ja ressursside soovitavate tulemustega kasutamiseks
		• Määra meeskonnaliikmete oskuste tase • Määratle meeskonnaliikmete soovitavad oskused - keda peaks välja õpetama - mida peaks õpetama - milline oskuste tase on vajalik - milline projekti tulemus sõltub väljaõppe tasemest • Määratle väljaõppe ressursid ja tegevused
Tarkvara mõõtmise kava (*Software measurement program*)	Effektiivne tarkvara mõõtmine on tähtis inseneritegevus	Koosta vajalike andmete saamiseks vajalike mõõtmiste kava, sea üles vahendid nende andmete kogumiseks ja viisid tulemuste kasutamiseks, et saavutada eesmärgid ja sihid

		• Määratle sihid ja eesmärgid • Määra sihtidele ja eesmärkidele vastavad mõõdud • Kasuta mõõtmist hindamiseks, ennustamiseks ja juhtimiseks • Uuenda mõõtmiste plaani