Organisatsiooni- ja arendusprotsessi mallid

1. Sissejuhatus

Mallid on eriti sobivad organisatsioonide ehitamiseks. Mallid moodustavad moodsa kultuurantropoloogia aluse, mis määratleb kultuuri mallide ja suhete terminites. Organisatsiooni mallidest sõltub tugevalt inimeste võime töötada.

Mallid aitavad meil mõista olemasolevaid ja ehitada uusi organisatsioone. Hea komplekt organisatsiooni malle aitab kaudselt kaasa õige protsessi tekkimisele.
Probleeme on antud mallide headuse kontrollimisega, need tulenevad eelkõige sotsiaalsete eksperimentide keerukusest.
Siin esitatud mallid ühendavad empiirilise teadmise põhiprintsiipide seletusega.
Antud mallid on kogutud silmapaistvalt produktiivseid tarkvara valmistavaid organisatsioone analüüsides ja nende alusel koostatud organistasioon oleks tunduvalt erinev tavalistest tänapäeva tarkvara tegevatest organisatsioonidest. Samuti kirjeldavad nad praktikat, mis on erinev tavalistest projektijuhtimise õpetustest.

1.1. Keele kontekst

Siin esitatud generatiivsed mallid on kasutatavad organisatsiooni ehitamiseks ja tarkvara väljatöötamise protsessi juhtimiseks.
Organisatsiooni all mõeldakse mitte ainult asutuse organisatsiooni vaid ühiste huvidega inimeste kooslust, vahel nimetatud ka kasulikuks organisatsiooniks (*instrumental organization*). Organisatsioon vastutab tavaliselt millegi tegemise (*deliverable*) eest. Selle tegemise katset (*endeavor*) nimetatakse projektiks (*project*). Tegevuste malle selles organisatsioonis (ja seega ka projektis) nimetatakse protsessiks (*process*). Organisatsiooni, projekti ja protsessi võib osaliselt vaadelda, kui üksteise erinevaid tahke. Me võime vaadelda ka organisatsioonide hierarhiaid ja võrke. Siin kirjeldatud mallid on rakendatavad eelkõige sisemistele organisatsioonidele ja vähem suurtele organisatsioonide struktuuridele.

Üldiseks kontekstiks on keerukad tarkvaraprojektid, mis koosnevad sadadest tuhandetest või miljonitest koodiridadest (tüüpilised telekommunikatsioonis). Organisatsioonide suuruseks on mõni kuni mõni tosinat inimest. Antud mallid kehtivad rohkem noorte kui vanade organisatsioonide kohta. Vanad organisatsioonid on peaaegu alati tavaliselt bürokraatiad. Noored organisatsioonid läbivad mitu stabiilset olekut alates tihedalt seotud unistajate (*visionaries*) grupist. Organisatsioonide kasvades ilmuvad ja täpsustuvad rollid.

1.2. Keelt juhtivad jõud

See mallide keel toetub AT&T Bell'i Laboratooriumis läbi viidud 3 aastasele uuringule (Pasteur'i protsessi uurimise programm). Keel kirjeldab hästi tuntud, organistasioonide kohta käivaid, folkloori ja praktikaid ning toetud samuti edukate organistasioonide empiirilistel vaatlustel saadud uutele järeldustele.
Pasteur'i analüüsi tehnika toetub visualiseerimisele.

1.3. Nimetu kvaliteet

Mõnedes organisatsioonides on "nimetu kvaliteet": nad täidavad oma eesmärki, andes kasumit osanikele, teenendades hästi kliente ja tagavad rahuldustpakkuva tööpaiga oma töötajatele.

1.4. Keele põhiprintsiip

Keel on inspireeritud Alexander'i arhitektuurilisest keelest. Mitmed mallid on tuletatud Alexander'i mallidest. Antud keel kirjeldab rollide vahelist suhtlemist, mis on kõrgema taseme abstraktsioon, kui indiviidide vaheline suhtlemine.

1.5. Esitus (notatsioon)

Mitmed mallid omavad visuaalseid analooge Pasteur'i analüüsis. Pasteur'i uuringutes kasutati kahte tüüpi pilte.
Esimene on sotsiaalne võrkdiagramm, mida nimetatakse ka külgnevusdiagrammiks (*adjacency diagram*). Iga mudel on rollide ja nende vaheliste suhtlemisteede võrk. Tihedalt seotud rollid on paigutatud lähestiku.
Teine on suhtlemismaatriks, kus ordinaat ja koordinaat esitavad rolle nende organisatsiooni kui tervikuga seotuse tugevuse järjekorras. Maatriksi ruudud on viirutatud seda tumedamaks, mida tihedam on esitatav kahe rolli vaheline suhe.

2. Mallide keel

2.1. Organisatsiooni suurus

Probleem Võtlusvõimelise hinna ja ajakavaga tarkvara väljatöötava organisatsiooni koostamine. Tarkvaratoote esimene väljalase (*release*) sisaldab rohkem kui 25K SLOC. Väljatöötav organisatsioon toimib tavaliselt suurema organisatsiooni koosseisus.

Jõud Kui organisatsioon on liiga suur siis tema tulemused vähenevad kiiresti. Suured organisatsioonid on rasked juhtida.
Liiga väike organisatsioon aga ei pruugi omada vajaliku kriitilist suurust. Üdiselt on väikestel organisatsioonidel väike inerts ja nad võivad muutuda ebastabiilseteks.
Suurus mõjutab tulemust mittelineaarselt. Suhtlemise mahukus kasvab ruudus võrreldes organisatsiooni tööjõudlusega, mis kasvab lineaarselt.
Uued inimesed suhtlevad rohkem teistega kui organisatsioonis kauem olnud.

Lahendus Vaikimisi valida 10 inimest. Kogemus näitab, et hästi valitud ja kasvatatud väikene meeskond võib välja töötada 1,5M SLOC projekti 31 kuuga, 200K SLOC projekti 15 kuuga või 60K SLOC projekti 8 kuuga.
Mitte lisada inimesi väljatöötamise hilistes faasides või püüda saavutada tähtaegu, mis on saadud lõpptähtajast tagurpidi tuletades.

Kontekst See mall tekitab organisatsiooni, kus peaaegu igaüks võb omada "globaalset teadmist". Empiiriliselt on leitud, et suurem osa rolle projektis suudavad suhelda 6 või 7 teise rolliga; 10 inimesega on võimalik saavutada samuti täieliku globaalset suhtlemist.
Teisest küljest on 10 inimest piisav "kriitiline mass".
See mall on tihedalt seotud mallidega, mis toetavad organisatsiooni kasvu.

Põhimõte Hoides organisatsiooni väiksena on igal ühel võimalik teada, kuidas projekt läheb. Hästi minevates projektides on protsessid, mis adapteeruvad ja protsessid adapteeruvad hästi vaid kui nad vastu võetakse ja neisse usutakse. Dialoog, mis on selleks vajalik on võimalik vaid väikestes organisatsioonides. Tom DeMarco on märkinud, et kõik protsessist kasu saavad osapooled peavad olema kaasatud protsessi ja otsuste tegmisse.
10 inimest on alguses ilmselt liiga palju aga see väldib kulutusi inimeste lisamisel hiljem.
See mall on suurte projektide jaoks. Üle 25K SLOC projektid on harva ühe indiviidi teha.
Erinevad juhtimise stiilid (*leadership-based, manager-based*) viivad organisatsiooni tüüpide (*democracy, republic, oligachy*) edule või ebaedule.
Üks meeskond on parem kui alammeeskondade kogum.

2.2. Iseselekteeruv meeskond

(*Self-Selecting Team*)

Probleem Võtlusvõimelise hinna ja ajakavaga tarkvara väljatöötava organisatsiooni koostamine.
Pole olemas veatut kriteeriumit meeskonna liikmete valikuks.

Jõud Volitused (*empowerment*) sõltuvad kompetentsusest ja teadmiste ning võimu jaotumisest.
Halvim meeskonna dünaamika on leitav nimetatud meeskondades. Parim meeskonna dünaamika on iseselekteerunud meeskondades.
Laiad huvid (muusika, luule, ...) näitavad edukale meeskonnaliikmele.

Lahendus Koostada iseselekteeruvaid meeskondi, tehes piiratud välja-sõelumist endiste saavutuste ja laiade huvide järgi.

Kontekst Entusiastlik meeskond, mis on valmis kõigeks, et saavutada projekti eesmärgid.

2.3. Üksik virtuoos (*Solo Virtuoso*)

Probleem Kui suur peab olema organisatsioon? Tarkvaratoote esimene väljalase sisaldab vähem kui 25K SLOC. Peale esimest väljalaset ei ennustata kiiret kasvu.

Jõud Mõned valitud indiviidid on võimelised üksi ehitama terveid projekte.
Suurus mõjutab tulemust mittelineaarselt. Suhtlemise mahukus kasvab ruudus võrreldes organisatsiooni tööjõudlusega, mis kasvab lineaarselt.

Lahendus Teha kogu projekt ühe-kahe inimesega.

Kontekst Tulemuseks on organisatsioon, mis on piiratud vaid väikeste projektidega. Ehkki on vaid üks väljatöötaja roll, võivad teised rollid olla vajalikud, et toetada marketingu, tööriistade haldamist ja teisi funktsioone.
Seda malli ei pea võtma kui "luba häkkida" (*"License to Hack"*). Ei tohi loobuda tehnilistest ülevaatustest ja verifitseerimistest.

Põhimõte On mitmeid näiteid edukaist ühe-mehe projektidest. Nende projektide dünaamika on erinev väikese meeskonna dünaamikast. Ühe väljatöötaja tööviljakus võib olla suurem, kui grupil mitmest effektiivsest indiviidist. On näiteid, kus üks inimene teostas 25K SLOC projekti 4 kuuga ja 2 inimest 135K SLOC projekti 13 kuuga.
Edu sõltub muidugi õige inimese valikust. Boehm märgib, et väljatöötajate effektiivsuse vahe võib olla 20 kordne.

2.4. Ajakava suurus (protsessi mall) (*Size the Schedule*)

Probleem Kui kaua peab projekt kestma eeldusel, et projekt on arusaadav ja projekti suurus hinnatav?

Jõud Kui ajakava on liiga lõtv, siis väljatöötajad muutuvad enesega rahulolevaiks.
Kui ajakava on liiga auahne, siis väljatöötajad põlevad ruttu läbi.
Mõlemail juhtude ei jõua toode turule õigeaegselt, samuti kannatab toote kvaliteet.
Viimasel juhul panevad kompromissid arhitektuurilistes lahendustes aluse hilisematele raskustele hooldusel.
Ilma ajakavata ja tähtaegadeta projektid jätkuvad lõputult, kulutades liiga palju aega tähtsusetute detailide ihumisele.

Lahendus (Au)tasustada väljatöötajaid tähtaegadest kinnipidamisel. Kasutada rahalisi preemiaid ja lisapuhkust.
Tuleb teha kaks komplekti tähtaegu: üks turu jaoks ja üks väljatöötajatele. Väline ajakava on kooskõlastatud tellijaga, sisemine ajakava meeskonnaga. Sisemine ajakava peab keskmise suurusega projekti puhul välimisest 2-3 nädalat lühem olema. Kui mõlemat ajakava ei saa sobitada, tuleb üle vaadata kliendi nõudmised, organisatsiooni ressursid või ajakavad ise.

Kontekst Projekt, mille sisemised tähtajad on paindlikud. DeMarco kirjutab, et kõige tõsisem märk hädas olevast projektist on ajakava, mis on saadud tagurpidi lõpptähtajast alates (1993).

Põhimõte On soovitusi, mille järgi sisemise ja välimise ajakava vaheline lõtk peab olema kuni 2

kuud, sest tavaliselt hilinemine tähendab suuremat möödalaskmist.

2.5. Vorm järeldub funktsioonist (*Form Follows Function or Aggregate Activities into Roles*)

Probleem Projektil puuduvad hästi määratletud rollid aga on teada atomaarsed tegevused.

Jõud Tegevused on liiga väikesed ja nende vahelised suhted dünaamilised, et olla kõlblikud protsessi ehitamiseks. Tegevused koonduvad sageli nedega seotud asjade või teiste valdkonna suhete ümber.

Lahendus Grupeerida tihedalt seotud tegevused (üksteisega seotud realisatsioonis, manipuleerivad samade asjadega, on semantiliselt seotud sama valdkonnaga). Nimetada saadud abstraktsioone ja teha neist rollid. Seotud tegevustest saavad rolli ülesanded (töökirjeldus).

Kontekst Tulemuseks on projekti rollide osaline määratlus.

Põhimõte Kommentaar: "Projektides esineb puuduvaid rolle tänu projektijuhtide kogenematusele.".

2.6. Tegevusala teadmine rollides (*Domain Expertise in Roles*)

Probleem Kuidas jagada inimesi rollidesse, kui on teada atomaarsed protsessi rollid ja väljatöötaja rolli kirjeldus?

Jõud Kõik rollid peavad olema kaetud/täidetud (kvalifitseeritud töötajatega). Jagades oskusi mitme rolli vahel muutuvad suhtlemismallid keerukamateks.
Edukad projektid on komplekteeritud tavaliselt inimestega, kes on juba osalenud edukates projektides.

Lahendus Palgata valdkonna asjatundjaid, kellel on tõestatavad saavutused. Iga inimene võib täita mitut rolli. Mitmetel juhtudel võivad mitu inimest täita ka sama rolli. Valdkonna alased oskused on tähtsamad protsessi alastest oskustest. Kohalikud *gurud* on kasulikud igal alal.

Kontekst See mall on vahendiks, kindlustamaks et rollid saavad edukalt täidetud. Samuti teeb ta rollid autonoomseteks.

Põhimõte Eriti kõrge produktiivsusega projektide meeskonnad on koostatud eriti sügavate teadmistega asjatundjatest.
Kommentaar: "Probleemiks on, et süsteemi-inseneri ja süsteemi-testija rollid. Esimeseks pannakse algajaid ja viimaseks jäävad tavaliselt need, kes mujale ei kõlba.".

2.7. Järk-järguline sissetoomine (*Phasing it In*)

Probleem On vaja palgata inimesi pikaks ajaks lisaks algsetele asjatundjatele, sest projekt vajab lisa tööjõudu.

Jõud On vaja piisavalt inimesi, et saavutada õige kriitiline mass. Töötajad pole tavaliselt vahetatavad.
Algne ekspertide komplekt määrab projekti tooni ja on tähtis palgata võtmeisikud kohe. Samas aga liiga mitme inimesega alustamine paneb liiga suure koormuse meeskonna tuumikule.

Lahendus Palkamise programm tuleb teha järk-järguliseks. Alustada tuleb asjatundjate palkamisest ja projekti kasvades järk-järgult tuua lisaks uusi inimesi.

Kontekst Organisatsiooni saab "täis" palgata vastavalt väljatöötluse koormusele.

Põhimõte Hästi tuntud juhtimispõhimõte, mis lubab saavutada projekti identiteeti varakult ja edukalt kasvada.

2.8. Õpipoiss (*Apprentice*)

Probleem Alati ei ole võimalik palgata vajalike eksperte, kui toimub projekti järk-järguline täis-

palkamine.

Jõud On vaja piisavalt inimesi, et saavutada õige kriitiline mass. Töötajad pole tavaliselt vahetatavad.

Lahendus Tuleb muuta uued töötajad ekspertideks õpipoisi staatuse läbimisega. Iga uus töötaja peab töötama õpipoisina kogenud asjatundja käe all (asjatundja peab olem rohkem kui järelevaataja). Tavaliselt on õpipoisi aeg 6 kuud kuni 1 aasta -- see on aeg mis on vajalik paradigma (musternäidiste kogu) omandamiseks.

Kontekst Tulemuseks on asjatundlikuse säilimine organisatsioonis. See mall lisab samuti organisatsiooni "veoauto arvu" (inimeste arv, kelle puudumine ohustab organisatsiooni, näiteks kui keegi neist peaks veoauto alla jääma `:-)`) jagades teadmisi ja asjatundlikust. Samuti tunnevad "meistrid" endid hinnatuina ja õpipoisid saavad hea keskkonna õppimiseks.

Põhimõte Parem on õpetada inimesi, kui panna nad "tuleproovile". Viimane võib ohustada kogu projekti. See lähenemine teeb võimalikuks valdkonna-spetsiifilised meeskonnad.

2.9. Organisatsioon järgib asukohta (*Organization Follows Location*)

Probleem On vaja jagada tööd ja rollid geograafiliselt hajutatud tööjõule. Toodet töötatakse välja mitmes erinevas koridoris, erinevatel korrustel, erinevates hoonetes või erinevates kohtades.

Jõud Projekti liikmete vahelise suhtlemise mallid järgivad liikmete geaograafilist paigutust. Tarkvara osade vaheline sidestus peab olema toetatud analoogilise sidestusega inimete vahel, kes seda tarkvara haldavad (*maintain*). Inimesed väldivad suhtlemist inimestega, kes töötavad teistes hoonetes, teistes linnades või mere taga. Inimesed töötavad organisatsioonis tavaliselt seotud ülesannete kallal, mis eeldab, et nad suhtlevad üksteisega sagedasti.

Lahendus Arhitektuuriline jaotus peab peegeldama geograafilist jaotust ja vastupidi. Arhitektuurilised ülesanded tuleb jaotada nii, et otsuseid saab kohapeal (geograafilises mõttes) vastu võtta.

Kontekst Tulemuseks on alamorganisatsioonid, mida võib edasi jaotada või organiseerida turu või mõne muu kriteeriumi järgi. On vaja kedagi, kes likvideeriks ummistused, kui ei saavutata üksmeelt.

Põhimõte Thomas Allen on leidnud, et sotsiaalne distants kasvab kiiresti füüsilise eraldumisega. Empiirilised andmed ülemere ühisprojektidest näitavad, et selle malli mittejärgimine võib viia projekti täielikule ebaõnnestumisele.
Steve Berczuk MIT'ist väidab, et "suhtlemine ei pruugi olla väike, kui: 1) projektis osalejate summarne arv on väike; 2) suurem osa suhtlemisest sooritatakse läbi e-posti sarnase vahendi (tagab suurema arvu inimeste osalemise, kui koridorivestlused); 3) osalejad on olnud MINGI aja koos, nii et nad tunnevad teineteist (arvavad tundvat); 4) inimesed ei ole 'mittevajalikest' reisidest väsinud ja on valmis reisima, kui see on vajalik..." (1994).
On juhtumeid, kus turg määrab hajutamise.

2.10. Organisatsioon järgib turgu (*Organization Follows Market or Framework Team*)

Probleem Ei ole selget rolli või organisatsioonilist arvestamist individuaalsete turusegmentidega. Turg koosneb mitmest kliendist, kelle vajadused on sarnased kuid vasturääkivad. Projekt on võtnud omaks tugevad arhitektuurilised põhimõtted ja võib organiseerida tarkvara vastavalt turu nõudmistele.

Jõud Väljatöötav organisatsioon peaks jälgima ja vastama iga kliendi soovidele. Klientide soovid on sarnased ja paljut, mida nad kõik vajavad saab teha ühiselt. Erinevad kliendid ootavad resultaate vastavalt erinevatele ajakavadele.

Lahendus Organisatsioonis, mis soovib teenendada erinevaid turge, on tähtis peegeldada turu

struktuuri väljatöötava organistasiooni struktuuris. Üks sageli unustet võimalus on teadlik "tuumik" organisatsiooni kujundamine, mis toetab ainult kõikidele tutusegmentidele ühist. Ralph Johnson nimetab seda raamistiku meeskonnaks (*framework team*). Esimeseks on tähtis püstitada see organisatsioon.

Kontekst Tulemuseks on organisatsioon, mis võib toetada head arhitektuuri. Selle malli edukusest sõltub ka arhitektuurilise visiooni säilitamise edukus.

Põhimõte Põhilised mõjuvad jõud on erinevate klientide ajakavad ja organisatsiooni püüd vastata kiirelt klientide nõudmistele. Kaks valdkonna analüüsi tähtsat aspekti on: 1) arhitektuuri laiendamine (töötades baasklasside tasemel) ja 2) arhitektuurilise arengu klientide vajadustele vastavuse tagamine. Üks organisatsioon ei saa hästi järgida mitme kliendi vajadusi ja see mall lubab organisatsiooni eri harudel järgida sõltumatult eri turu segmente.

2.11. Arendaja juhib protsessi (*Developer Controls Process*)

Probleem Milline roll peaks olema projektis suhtlemise keskpunktiks, kui disaini valdkond pole täielikult arusaadav ja iteratsioon on väljatöötluse võtmeks.

Jõud Totalitaarne juhtimine on suuremale osale väljatöötavatest meeskondadest vastuvõtmatu. Õige informatsioon peab läbima õigeid rolle. On vaja, et toetav informatsioon liiguks läbi analüüsi, disaini ja realistasiooni.
Juhid omavad osalist vastutust. Väljatöötajad peavad omama lõpliku vastutust, võimu ja juhtimist toote üle.

Lahendus Paigutada Arendaja (*Developer*) roll omaduse (*feature*) väljatöötamise protsessi keskpunkti. Omadus on süsteemi osa funktsionaalsus, mis on realiseeritud suuremalt osalt tarkvaras, mida saab eraldi pakkuda ja mille eest tarbija on valmis maksma. Arendaja on protsessi informatsiooni arvelduskoda (*clearinghouse*). Arendaja ülesanneteks on nõuetest arusaamine, lahenduse struktuuri ja algortimide ülevaatamine, realisatsiooni ehitamine ja osaline testimine.

Kontekst Tulemuseks on organisatsioon, mis toetab oma põhilist informatsiooni tarbijat. Ehkki Arendaja peaks olema võtmeroll, tuleb hoiduda tema ülekoormamisest.

Põhimõte Pole olemas Disaineri (*Designer*) rolli, kuna disain on kogu ülesanne. Mänedzerid (*Manager*) täidavad toetavat rolli, neid nähakse protsessi juhtimas ainult kriisisituatsioonides. Samas kui Arendaja juhib protsessi, Arhitekt juhib toodet. Arhitekti roll on jagatud Raamistiku Omaniku (*Framework Owner*) ja Arhitektuuri Rühma (*Arhitecture Team*) vahel. See side on eriti tähtis valdkondades, mis pole täielikult arusaadavad, et saaks toimuda iteratsioon kliendiga koos valdkonna uurimiseks.

2.12. Patroon (*Patron*)

Probleem Kuidas saavutada projekti jätkuvus väljatöötavas organisatsioonis, kus määratletakse rolle?

Jõud Tsentraliseeritud juhtimine võib olla piduriks; anarhia võib olla tugevamaks piduriks. Suurem osa sotsiume (*societies*) vajavad kuninga/vanema kuju. Organisatsioon vajab ühte, lõpliku otsusetegijat. Otsuse tegemiseks vajalik aeg peaks olema väiksem, kui otsuse elluviimiseks vajalik aeg.

Lahendus Andke projektile juurdepääs nähtavale, kõrge-tasemelisele mänedzerile, kes on projekti eestvõitlejaks (*champion*). Patroon võib olla projekti otsuste viimane kohtunik; see tekitab organisatsioonile tungi teha otsused kiiresti. Patroon on vastutav projekti tasemel tõkete eemaldamise ja organisatsiooni moraali (heaolu tunde) eest.

Kontekst Patrooni omamine annab organisatsioonile heaolu tunde ja keskpunkti hilisemate protsesside ja organisatsiooniliste muudatuste jaoks.
Mänedzeri roll pole mõeldud totaalselt tsentraliseeritud juhtimiseks vaid eestvõitlejaks.

See tähendab, et mänedzeri mõjupiirkond on suures osas väljaspool toote enda arendustööd ja sisaldab neid, kelle koostöö on vajalik toote eduks (toetavad organisatsioonid, investeerijad, testijad jms.). Selles rollis olev inimene on sageli korporatsiooni unistaja (*visionary*).

Põhimõte Block (1983) kirjeldab nende projekti mõjutavate jõudude tähtsust, mille üle projektil puudub igasugune kontroll.
Arendaja tegemine projekti eest vastutavaks eeldab, et juhtkond võtab endale toetavad rollid. See mall toimib vaid sellises kultuuris, kus mänedzer otsustab olla arendaja teener.

2.13. Arhitekt juhib toodet (*Architect Controls Product or Planning and Architecture*)

Probleem Mitme indiviidi poolt ühiselt disainitud tootel puudub elegants ja loogiline järjekindlus.
Les oeuvres d'un seul architect sont plus belles ... que ceux d'ont plusieurs ont taché de faire.
Pascal, *Pensées*
Väljatöötav organisatsioon vajab tehnilist strateegilist juhtimist.

Jõud Totalitaarne juhtimine on paljude projektorganisatsioonide jaoks viimaseks võimaluseks -- õige informatsioon peaks läbima (vastavaid) õigeid rolle.

Lahendus Tuleb luua Arhitekti roll. Arhitekt peab juhendama ja juhatama Arendajate rolle ning olema nendega tihedates suhetes. Samuti peab Arhitekt olema tihedates suhetes tellijaga.

Kontekst See mall teeb arhitektuurile sama, mis 'Patrooni' mall teeb organisatsioonile: see tähandab annab tehnilise kesk- ja koondamispunkti (fookuse) nii tehnilistele kui ka turuga seotud tööle. Selle ja 'Patrooni' malli vahel on tihe suhe. Totalitaarse arhitekti vastu hakatakse (sala)viha kandma.

Põhimõte Pole olemas Disaineri (*Designer*) rolli, kuna disain on kogu ülesanne. Mänedzerid (*Manager*) täidavad toetavat rolli, neid nähakse protsessi juhtimas ainult kriisisituatsioonides. Samas kui Arendaja juhib protsessi, Arhitekt juhib toodet. Arhitekt on Peaarendaja (*Chief Developer*). Arhitekti ülesannete hulka kuulub nõudmistest aru saamine, süsteemi struktuuri kujundamine ja selle pikaajalise arengu juhtimine.

2.14. Conway seadus (*Conway's Law or Organization Follows Architecture*)

Probleem Kuidas ühtlustada organisatsiooni ja tema arhitektuuri, kui Arhitekt ja väljatöötav meeskond ning arhitektuur on paigas.

Jõud Arhitektuur mõjutab ja vormib suhtlemisteid organisatsioonis. *De facto* organisatsiooni struktuur vormib formaalset organisatsiooni struktuuri ja formaalne organisatsiooni struktuur vormib arhitektuuri. Varajased arhitektuuri visandid on vaid ebastabiilsed lähendused.

Lahendus Kindlustada organisatsiooni sobivus toote arhitektuuriga. Tundub, et on tõenäolisem, et arhitektuur peaks määrama organisatsiooni, kui vastupidi.

Kontekst Organisatsioon ja toote arhitektuur ühtlustuvad.

Põhimõte Kui organisatsioon seotakse toote arhitektuuriga liiga vara, siis arhitektuuri muutumine viib vastuoludele inidiviidide võimkondades.

2.15. Arhitekt realiseerib samuti (αρηιτεκτον)

Probleem Kuidas säilitada arhitektuurilist visiooni läbi kogu realistasiooni?
Väljatöötav organisatsioon vajab tehnilist strateegilist juhtimist.

Jõud Totalitaarne juhtimine on paljude projektorganisatsioonide jaoks viimaseks võimaluseks -- õige informatsioon peaks läbima (vastavaid) õigeid rolle.

Lahendus Peale Arendajatega suhtlemise ja neile nõu andmise, peavad Arhitektid osalema ka realiseerimises.

Kontekst Tulemuseks on arendusoganisatsioon, mis taipab juhtivate Arhitektide ideid ja kasutab otseselt ära arhitektuurilisi oskusi.

Põhimõte Ühes hiljutistest projektidest oli arhitektuuri meeskond hajutatud geograafiliselt nii, et nende vaheline informatsiooni vahetus oli väike, selgus hästi selle malli kasutamise selge esitamine. Olgugi, et üldised arhitektuurilised vastutusalad olid kindlaks määratud ja rollid täidetud, arvestas üks grupp, et Arhitektid realiseerivad ja teine grupp, et Arhitektid ei realiseeri.

On tehtud ettepanekuid, et Arhitektid piirduks vaid üldise infrastruktuuri realiseerimisega ja kõik ülejäänud osad realiseeritaks Arendajate poolt.

2.16. Arhitektuuri ülevaatus (***Review the Architecture***)

Probleem Arhitektuuris ja disainis on "pimedaid" punkte.

Tarkvara tehise (*artifact*) kvaliteeti tuleb hinnata ja parandada.

Jõud Arhitektuurilised otsustused mõjutavad mitmeid inimesi pika aja jooksul. Hoolimata sellest võib üksikutel Arhitektidel ja Disainieritel kujuneda ühekülgne vaatepunkt. Jagatud ja lai arhitektuuriline visioon on vajalik.

Mõju omavad isegi madala-taseme disaini otsustused.

Lahendus Kõik arhitektuurilised lahendused peavad olema üle vaadatud kõikide Arhitektide poolt. Arhitektid peavad üksteise koodi üle vaatama. Sellised ülevaatused peavad olema sagedased -- projekti varajases staadiumis isegi igapäevased. Ülevaatused peavad olema informaalsed, minimaalse paberitööga.

Kontekst See mall kehtestab konteksti mallile Palgaanalüütik (2.23), samuti lahendab ta potentsiaalseid probleeme malliga Koodi omandus (2.17).

Selle malli eesmärgiks on suurendada sidestust nende vahel, kes osalevad arhitektuuris ja realiseerimises. See toimub kaudselt.

Põhimõte Mall põhineb firma AT&T eduka objekt-orienteeritud projekti kogemustel.

2.17. Koodi omandus (***Code Ownership***)

Probleem Arendaja ei suuda sammu pidada realisatsiooni koodi muutustega.

On olemas süsteem dokumenteerimise ja tarkvara arhitektuuri jäörgimise mehhanismidega, ja töö toimub organisatsioonis, milles on Arendajad koodi kirjutajateks.

Jõud Kõikide kohustus pole kellegi kohustus.

Vajatakse Arendajate vahelist paralleelsust, et mitu inimest saaksid kodeerida paralleelselt.

Suurem osa disaini teadmistest on kirjutatud koodi. Tundmatus koodis orientreerumine, et disainiga tutvuda, võtab aega. Ajutised muudatused ei tööta.

Igaüks ei saa kõike kogu aeg teada.

Lahendus Iga koodi mooduli omanikuks süsteemis on kindel Arendaja. Välja arvatud eriolukordades, võib koodi muuta vaid koodi omanik.

Kontekst Selle malli rakendamisel peegeldavad arhitektuur ja organisatsioon paremini teineteist. Seotud mallideks on Arhitekt realiseerib samuti (2.15), Organisatsioon järgib turgu (2.10) ja Katkestused takistavad blokeerimist (2.40).

Mall Arhitektuuri ülevaatus (2.16) aitab hoiduda Disainerite ja Arhitektide vaatevälja ahenemisest (kitsast kontsentreerumisest oma ülesandele), mis võib käesoleva malli rakendamisel tekkida.

Põhimõte Koodi omanduse puudumine on tänapäeval põhiline suure tarkvara (*large-scale software*) uurimise töömahukuse tegur. Koodi omandus kuulub kokku arhitektuuriga -- et saaks tekkida koodi omandus, peavad eksisteerima liidesed. Antud mall töötab sarnaselt Conway seadusega (2.14) väiksemas mastaabis.

On olemas palju argumente koodi omanduse vastu, kuid empiirilised vaatlused kinnitavad tema kasulikust. Tüüpiliselt kardetakse vaatevälja ahenemist, riski, et antud koodilõiku tunneb sügavuti vaid üks inimene ja globaalse teadmise kadumist. Neid probleeme võib

lahendada teiste mallidega (Arhitektuuri ülevaatus (2.16), Kaasa kliendid (2.20)).
Tim Born väidab, et eksisteerib seos koodi omanduse ja kapseldumise (*encapsulation*) vahel.
Samuti on väidetud, et koodi omandust peaks rakendama vaid korduvkasutatavale (*reusable*) koodile. Seda kitsendust saaks rakendada, kui keegi suudaks anda kasutatava ja korduvkasutatava koodi erinevuse definitsiooni.

2.18. Rakenduse disain on piiratud testide disainiga (*Application Design is Bounded by Test Design*)

Probleem Kunas disainida ja realiseerida testimise plaane ja test-skripte, kui on olemas dokumenteerimise ja arhitektuuri järgimise mehhanismidega süsteem, Arendajad kirjutavad koodi ja Testija roll on määratud.

Jõud Testide tegemine võtab aega ja seda ei saa alustada alles siis, kui süsteem on valmis (s.t. "kui on teada mida testida"). Stsenaariumid on teada, kui nõudmised süsteemile on teada ja mitmed neist on teada projekti varajases staadiumis.
Testida tegemine (realiseerimine) nõuab detailseid teadmisi liidestest, vormingutest (formaatidest) ja teistest arhitektuuri omadustest (et teha test-skripte ja test-draivereid).
Süsteemi realisatsioon muutub päevadega; pole vaja, et testide disain järgiks lühiajalisi muudatusi tarkvara realisatsioonis.

Lahendus Stsenaariumitel põhinev testide disainimine algab, kui nõuded stsenaariumitele on tellijaga kokku lepitud. Testide disain areneb koos tarkvara disainiga aga ainult tulenevalt stsenaariumite muutumisest: lähtekood on Testijale kättesaamatu. Kui Arendajad otsustavad, et arhitektuurilised liidesed on stabiliseerunud, jätkub madala-taseme testide realiseerimine.

Kontekst Antud mall kuulub kokku mallidega Rakenda kvaliteedikontrolli (2.19) ja Stsenaariumid määratlevad probleemi (2.22).

Põhimõte Tehes tarkvara kättesaadavaks Testija(te)le põhjustatakse nende poolt "Arendaja vaate" omaks võtmine "Tellija vaate" asemel. See viib võimalusele, et Testija testib valesid asju või kontsentreerub valele detailsuse astmele. Tarkvara areng jätkub, kuni arhitektuur stabiliseerub. Kui liidesed on stabiliseerumata, pole mõtet pidevalt muuta testide disaini. Lühidalt -- testide disain algab esimese suurema nõudmiste hulga saabumise lõpul ja lõpeb, kui arhitektuur on stabiilne.

2.19. Rakenda kvaliteedikontrolli (*Engage QA*)

Probleem Kuidas garanteerida toote kvaliteet?
On vaja mingit filtrit arendusorganisatsiooni rollide ja Tellija vahel, et tagada toote kvaliteet.

Jõud Arendajatele tundub, et nad on kõigest õieti aru saanud aga perfektse tarkvara loomine on raske. Kvaliteedis tehakse liiga sageli järeleandmisi, hoolimata et edu sõltub kõrgest kvaliteedist.
Põhimõtteliste kvaliteediprobleemide puhul on tähtis varajane tagasiside.

Lahendus Tehke Kvaliteedijuhist keskne roll. Siduge ta tihedalt Arendajaga kohe kui Arendajal on midagi testida. Arendaja testimisplaan peaks olema paralleelne kodeerimisega, aga Arendajad peaksid enne testimise algust süsteemi testimiseks valmis olevaks deklareerima. Kvaliteedi tagamise organisatsioon peab olema väljaspool projekti konteksti: Kvaliteedijuht ei peaks andma aru arendusorganisatsioonile oma testimisplaanidest.

Kontekst Kui Kvaliteedijuht on kaasatud, on projekt valmis vastu võtma eelinformatsiooni Tellijalt. Kui Kvaliteedijuht ja Tellija on seotud, võib kvaliteedi tagamise protsess alata (kasutusjuhtude kogumine, jne.)

Põhimõte On vähemalt 2 põhjust kvaliteedi tagamise organistasiooni eraldamiseks Arendajate (poolt usaldatavast) organistasioonist: Esiteks, testide tegemine ei peaks olema pimestatud Arendaja perspektiivist. Kui mõlemad nii Arendaja kui ka Kvaliteedijuht teevad

oma teste, saab testimisest "topelt pime" eksperiment (s.t. eksperiment, kus nii kontrollija kui ka kontrollitav ei oma eelnevaid teadmisi). Teiseks, kvaliteedi tagamine peab olema objektiivsuse huvides väljaspool arendava organisatsiooni mõjupiirkonda.

2.20. Kaasa kliendid (*Engage Customers*)

Probleem Kuidas tagada kliendi rahulolu. Samas on vajalik leida tee kvaliteedi tagamiseks vajaliku informatsiooni saamiseks.

Jõud Arendajaid on võrreldud kinnitamata kahuritega laevadekil (nende käitumine on ettearvamatu ja ohtlik).
Nõudmised muutuvad isegi kui disaini ülevaatused on läbi ja kodeerimine alanud.
Puuduvad nõudmised põhjustavad tõsiseid probleeme. Tellijad pole tavaliselt arendusprotsessis osalised, mis teeb raskeks leida ja arvesse võtta nende arvamusi (arusaamu).
Kodeerijate ja juhtkonna vahel peab olema usaldus.

Lahendus Siduda Tellija roll tihedalt Arendaja ja Arhitekti rolliga, mitte ainult Kvaliteedijuhi rolliga.

Kontekst Saavutatud situatsioon toetab nõudmiste avastamist Tellija poolt.

Põhimõte Mõned protsessid ja meetodid toetuvad kliendi kaasamisele (näiteks IBM'i *Joint Application Development*).Teised meetodid juhivad kliendi kaasamisele (nagu Beck'i CRC tehnika). On meetodeid, eriti mitmed CASE süsteemidele põhinevad, mis on ükskõiksed kliendi kaasamise suhtes või seda isegi takistavad.
Malli nimes on kliendid mitmuses, et toetada laiemat vaadet valdkonnale ja mitte lasta ennast pimestada ühest tellijast.
Projektis tuleb väga hoolsalt juhtida klientide ja Arendajate vahelisi suhteid (kasutades malle Stsenaariumid määratlevad probleemi (2.22), Tulemüürid (2.24) ja (37)).
Tähele tuleb panna, et toote kvaliteet pole antud malli juures probleemiks -- tootekvaliteet on vaid üks kliendi rahulolu komponent. Uuringud on näidanud, et kliendid vahetavad teenendavat firmat, kui nad tunnevad, et neid ignoreeritakse (20% juhtudest) või kui tähelepanu mis nad saavad on jäme või mitte-abivalmis (50% juhtudest). Vaid 9% klientidest kellel on probleem mille lahendamine maksab üle $100 ostavad firmalt uuesti kui see ei lahenda probleemi. 82% jätkab sama firma kliendina, kui probleem lahendati kiirelt.
Kuna organisatsioon eksisteerib, et teenendada klienti, peavad muud rollid paigas olema ennem kui Klient on kaasatud.

2.21. Rühmas valideerimine (*Group Validation*)

Probleem Kuidas tagada toote kvaliteeti.
Analüüsi, disaini või realisatsiooni kvaliteet vajab hindamist.

Jõud Kvaliteedijuhi rolli ülesanne on hinnata kvaliteeti. Tavaliselt hindab ta lõpptoote kvaliteeti tehes musta kasti meetodil valideerimist ja verifitseerimist.
Rühm võib taibata üksikust paremini toote probleeme ja võimalusi. Üksikisikud ei pruugi taibata teatud asju (vead süsteemis, jne.) see võib olla objektiivsuse küsimus.

Lahendus Enne Kvaliteedijuhi kaasamist võib Arendusmeekond (koos kliendi poolse osalusega) valideerida disaini. Sellised tehnikad, nagu CRC kaardid ja rühmas silumine aitavad disaini küsimusi teadvustada hulgale inimestele ja lahendada kõik probleemid.
Valideerimisrühma liikme võivad töötada koos kvaliteedi tagajatega, et parandada üldiste tarkvaravigade klassi põhjused.

Kontekst Tulemuseks on protsess, milles süsteemi kvaliteet on pidevalt kogu meeskonna tähelepanu keskpunktis. Selletõttu lahendatatkse probleemid varem. Selle malli rakendamise hinnaks on aeg, mis läheb rühmatööle.

Põhimõte CRC disaini tehnika on tõdetud heaks meeskonna ehitamise vahendiks ja ideaalseks viisiks teadvustada disaini inimestele. AT&T's tehtud uuringud näitavad, et rühmas silumine on eriti tööviljakas. Tuues ka klient nendesse sessioonidesse võib olla (eriti)

suureks abiks. Samas tuleb hoolikalt juhtida klientide ja Arendajate vahelist suhtlust. Empiirilised uuringud on näidanud, et rühmas silumine aitab kaasa õppimisele ja tõstab rühma effektiivsust.

2.22. Stsenaariumid määratlevad probleemi (*Scenarios Define Problem*)

Probleem Disaini dokumendid on sageli väheeffektiivsed vahendamaks Kliendi visiooni, kuidas süsteem peab töötama.
On vaja kaasata Klient ja on vaja mehhanismi, mis toetaks kliendi ja Arendajate vahelist suhtlust.

Jõud Klientide ja Arendajate vahel on olemas loomulik distants ja usaldamatus. Samas on Arendajate ja Klientide vaheline suhtlemine süsteemi eduks määrava tähtsusega.

Lahendus Nõuded süsteemi funktsionaalsusele tuleb esitada kasutusjuhtudena (*use cases*).

Kontekst Probleem on määratud ja tõsiselt võib jätkata arhitektuuriga.

Põhimõte Jacobson, Curtis, samuti Rubin ja Goldberg (viimane toob stsenaariumi isegi ettepoole disainist).

2.23. Palgaanalüütik (*Mercenary Analyst*)

Probleem Disaini notatsiooni ja projektdokumentatsiooni hooldamine ja haldamine on liiga igav ja tüütav töö inimestele, kes otseselt loovad asju (*artifacts*).
Toimub organisatsiooni rollide koostamine. Organisatsioon eksisteerib kontekstis, kus välised ülevaatajad, Kliendid ja Arendajad peavad kasutama projekti dokumentatsiooni, et mõista süsteemi arhitektuuri ja selle sisemisi mehhanisme. Tarbija dokumentatsiooni (näit. kasutusjuhend) vaadeldakse eraldi.

Jõud Kui ARendajad teevad oma dokumentatsiooni ise takistab see neid tegemast oma "tegeliku" tööd. Dokumentatsioon on sageli loetamatu (*write-only*). Inseneridel pole sageli head väljendus- ja suhtlemisoskust.
Arhitektid võivad jääda sageli omaenda jooniste elegantsi ohvriks.

Lahendus Palgata tehniline kirjutaja, kes on asjatundlik vajalikes valdkondades, aga kelle huvid pole kaalul disainis.
See inimene esitab disaini kasutades sobivat notatsiooni ning vormistab ja publitseerib disaini ülevaatuste ja organisatsiooni enda poolt kasutamiseks.

Kontekst Selle malli edu sõltub vajalike oskustega indiviidi leidmisest. Kui mall õnnestub, saavutatakse projekt, mille progressi on võimalik üle vaadata ja jälgida väliste asjatundjate poolt.

Põhimõte Laenatud klassikalisest arhitektuurist: on olemas oht, et ilusad joonised võivad saada omaette eesmärgiks. Arhitektuurilise joonise ülesandeks on erinevate osade vaheliste suhete illustreerimine.

2.24. Tulemüürid (*Fire Walls*)

Probleem Realiseerijaid segavad sageli kõrvalseisjad, kes tunnevad vajadust pakkuda nõuandeid ja kriitikat.
Arendajate organisatsiooni, mis on väljakujunenud korporatiivses või sotsiaalses kontekstis uurivad ülemused, kliendid ja muud kõrvalseisjad.

Jõud Isolatsioon ei toimi -- informatsiooni voolamine on tähtis.
Suhtlemisele kuluv töömaht tõuseb ebalineaarselt koos väliste kaastööliste lisandumisega. Paljud ettepanekud on müra.
Küpsus ja areng on rohkem seotud juhtimisega kui juhitav olemisega.

Lahendus Luua Mänedzeri roll, et varjata Arendajaid väliste teguritegaa suhtlemise eest. Mänedzeri ülesandeks on "nuhtlused eemale hoida".

Kontekst Uus organisatsioon eraldab Arendajad liigsetest välistest katkestustest. et vältida

isolatsiooni peab seda malli kombineerima teistega (Kaasa kliendid(2.20) ja Väravavaht(2.25)).

Põhimõte Sun Tzu (Hiina kindral 25 sajandit tagasi): "Võidab see, kellel on vajalik sõjaline võimsus ja keda ei sega valitseja".

2.25. Väravavaht (*Gatekeeper*)

Probleem Kuidas edendada suhtlemist tüüpiliselt introvertsete inseneridega. Arendajate organisatsioon on formeerunud korporatiivses ja sotsiaalses kontekstis ja uurimise (vaatluse) all ülemuste, Klientide ja teiste kõrvalseisjate poolt.

Jõud Isolatsioon ei toimi -- informatsiooni vool on tähtis.
Suhtlemisele kuluv töömaht tõuseb ebalineaarselt koos väliste kaastööliste lisandumisega. Paljud ettepanekud on müra.
Küpsus ja areng on rohkem seotud juhtimisega kui juhitav olemisega.

Lahendus Üks projekti liige, kelle personaalsuse tüüp on sobiv, tõstetakse väravavahi (uksehoidja) rolli. See isik levitab uusimat- ja raaminformatsiooni väljastpoolt projekti liikmetele, "tõlkides" seda projekti terminitesse.
Väravavaht võib samuti väljastada projekti informatsiooni asjasse puutuvatele kõrvalseisjatele. See roll võib samuti siduda Arendajaid turunduse ja ettevõtte juhtstruktuuridega.

Kontekst See mall tasakaalustab malli Tulemüürid (2.24) ja täiendab malli Kaasa kliendid (2.20), kusjuures kliente käsitatatkse kui kõrvalseisjaid.
Tulemüürid ja Väravavaht üksi on vähe, et kaitsta Arendajaid organistasioonis, mille kultuur lubab turundusega tegelejaid mõjutada arendustöö ajakavasid.

Põhimõte Väravavaht võimaldab kasuliku informatsiooni effektiivset voolamist. Tulemüür takistab segava informatsiooni voolamist. Tavaliselt on insenerid halvad suhtlejad, seega on tähtis ära kasutada häid suhtlejaid. Sama inimene võib sageli täita mänedzeri ja Väravavahi rolli.

2.26. Kujunda valitsusalad (*Shaping Circulation Realms*)

Probleem Suhtlemismallid organisatsioonis pole need, mis nad peaksid olema vastavalt teistele mallidele.
See mall on ehituskiviks teistele siinkirjeldatud mallidele, nagu Organisatsioon järgib turgu (2.10), Arendaja juhib protsessi (2.11), Arhitekt realiseerib samuti (2.15), Rakenda kvaliteedikontrolli (2.19), Kaasa kliendid (2.20), Buffalo mägi (2.28), jne.
Seda malli võib samuti rakendada projektivälistele valitsusaladele nagu mallis Tulemüürid (2.24), jne.

Jõud Õiged suhtlemisstruktuurid rollide vahel on organisatsiooni edu võtmeks. Suhtlemist ei saa juhtida ühest rollist, hõivatud peab olema vähemalt kaks rolli. Suhtlemismalle ei saa dikteerida; peab eksisteerima mingi teisene jõud, et neid tekitada. Suhtlemine jälgib vastutusalade vahelist semantilist sidestust.

Lahendus Inimestele tuleb anda tiitlid, mis loovad hierarhia või "nokkimisjärjekorra" struktuuriga, mis peegeldab vajaliku jaotust. Samuti tuleb inimestele anda ülesanded, milles peegeldub sobiv rollide vaheline suhtlus (mall Nihuta vastutust (2.27)).
Füüsiliselt tuleb inimesed paigutada selliselt, et need, kelle vahel soovitakse tihedamat suhtlust oleksid paigutatud lähestiku (see on malli Organisatsioon järgib asukohta (2.9) täiend).
Inimestele tuleb ütelda, mida teha ja kellega suhelda, tavaliselt püüavad inimesed teie soove järgida, kui neilt midagi mõistliku paluda, mille tegemiseks neil võimalused on.

Kontekst Tulemuseks on tugevamalt sidestatud organisatsioon, mille alamosad on sisemiselt ühtsed (*cohesive* -- nidusad) ja väliselt lahtisidestatud (*decoupled*).

Põhimõte Antud mall järgib Alexander'i malli, millel on sama nimi (§98) ja omab tugevat analoogiat Kodude kobaraga (§37). Tuleb täheldada, et mall Nihuta vastutust (2.27) on antud malliga tihedalt seotud, samuti Väravavaht (2.25).

2.27. Nihuta vastutust (***Move Responsibilities or Load Balancing or Chief Programmer Team***)

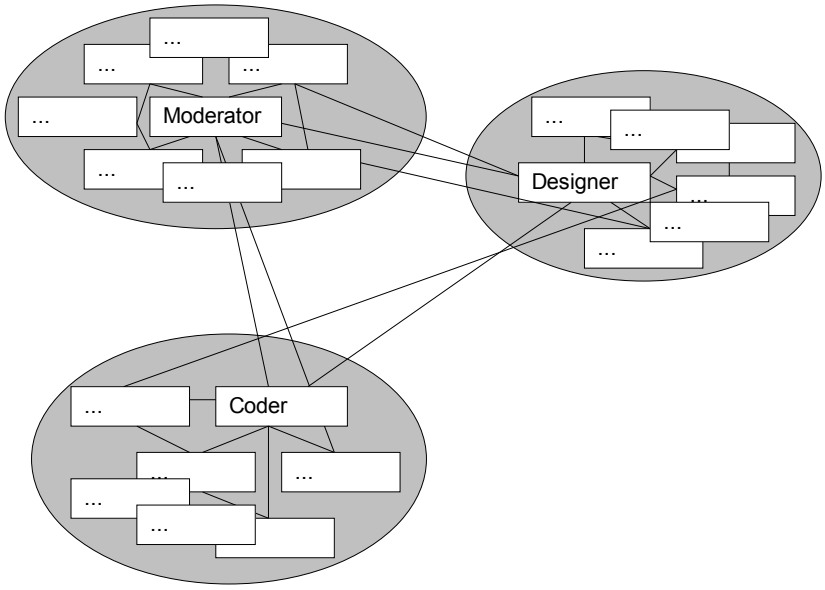
Probleem Halvastijälgitavad suhted rollide vahel võivad viia ebasoovitavate sidestusteni organisatsiooni tasandil.
Mall kehtib iga organisatsiooni või protsessi kohta.

Jõud Eksisteerib soov saada nidusaid (*cohesive*) rolle ja nidusat organisatsiooni.
Lahtisidestatud (*decoupled*) organisatsioon on tähtsam, kui nidusad rollid. nidususe ja lahtisidestuse vahel tuleb teha põhimõttelisi kompromisse.
Kogu rolli nihutamine ühest protsessist või organisatsioonist teise ei vähenda üldist sidestust, vaid nihutab selle tekitajat.

Lahendus Tuleb nihutada vastutus (ülesanded) rollist, mis loob kõige ebasoovitavama sidestuse rollidesse, mis on temaga seotud. Lihtsalt väljendudes tähendab see koormuse tasakaalustamist. Vastustust ei tohiks siirdada suvaliselt; üheks heaks vahendiks selle malli elluviimisel (Arendaja rolli kontekstis) on peaprogrammeerija meeskond (*chief programmer team*).

Kontekst Tulemuseks on protsess, milles ilmneb rohkem tugevalt lahtisidestatud gruppe. Tähtis on tasakaalustada grupi ühtsust lahtisidestusega, seega tuleb antud malli rakendada ettevaatlikult. Näiteks on Arendaja roll sageli suure osa projekti ülesannete kohaks, seega on roll ülekoormatud. Siirdades suvaliselt Arendaja ülesandeid teistele rollidele võib suurendada suhtlemiseks minevat töömahtu. Peaprogrammeerija meeskond võib neid jõude tasakaalustada aidata.
Alternatiivseks töökoormuse tasakaalustamise malliks on Buffalo mägi (2.28).

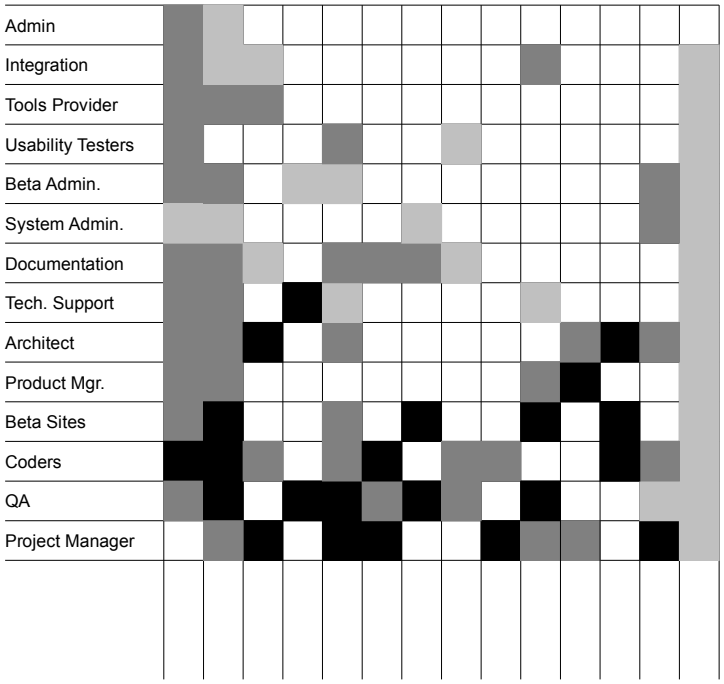
Põhimõte Mall tuleneb suures osas jõududest endist. See on samasugune, kui Mackenzie mudel, milles ülesannete vahelised suhted koos ülesannete ressursside vaheliste suhetega ja nende iseloomuga määravad projekti rollid.



2.28. Buffalo mägi (visuaalselt meenutab suhtlemismaatriks antud mäe kuju) (***Buffalo Mountain***)

Probleem Eksisteerib soov optimiseerida sidet suures tarkvara väljatöötavas organisatsioonis, mille liikmed töötavad ühise projekti kallal.

	Arendav organisatsioon jaotub mitmesse piirkonda; effektiivne osakondade vaheline side on projekti õnnestumise võtmeks.Organisatsioonile on juba moodustunud mingi tuumik.
Jõud	Suhtlemisele kulutatud töömaht kasvab töötajate arvu kasvamisel ebalineaarselt. Suhtlemise kitsaskohaks osutuvatel rollidel ei jää aega teha tööd. Samuti põhjustavad sellised kitsaskohad järjekordi organisatsiooni töökulus ja takistavad teisete inimeste tööd. Täielikult hajutatud juhtimine viib juhtimise kadumisele.
Lahendus	***Alammall 1:*** Projekti iga märkimisväärse suhtlusakti jaoks peaks kahe koostööd tegeva rolli kaugus organisatsiooni keskmest olema väiksem, kui lühim tee läbi kogu organisatsiooni. ***Alammall 2:*** Tuleb vältda naabrite (iga roll, mis on samal kaugusel protsessi keskpunktist, kui antud roll -- s.t. iga roll, mis on sama tugevalt sidestatud protsessi kui tervikuga) vahelist tihedat sidestust. ***Alammall 3:*** Iga koostöö intensiivsus on pöördvõrdeline suhtlevate rollide kauguste summaga protsessi keskpunktist. ***Vahendid A:*** Vaheta rollide ülesandeid nii et koostööpartnerid kujuneksid vastavalt antud mallile (vaata Nihuta vastutust (2.27)). ***Vahendid B:*** Paiguta füüsiliselt inimesed ümber, et parandada nende suhtlemisvõimalusi (vaat Töö voolab sissepoole (2.29)). ***Vahendid C:*** Suurenda rolli juhtimisulatust projektis (sarnane mitme rolli ühendamisele). Ilmselt on parim ühendada samalaadsete ülesannetega rolle, või veelgi parem ühendada rolle, millel on samalaadsed partnerid.
Kontekst	Uues organisatsioonis on rollide vaheline suhtlemine rohkem tasakaalustatud.
Põhimõte	Suurem osa neist mallidest on empiirilised. Alammall 1 on empiiriline. See on põhiliseks alammalliks, see mall viib sisse "tsentraalse tendentsiga hajutatud juhtimise", mis aitab hoida üldist suunda ja organisatsiooni nidusust. Teine viis alammalli 1 sõnastamiseks on, et suurem osa suhtlemismaatriksi ruute asetseb telgede lähedal. Alammall 2 väldib killustunud rühmi. Samuti aitab ta vältida sündmuste lineaarset järjestust protsessi kaugetes osades. Lineaarne sündmuste järjestus (konveieri kasutamine) põhjustab suhtlemismaatriksi ruutuda paiknemise allpool diagonaali. Mall, mis väldib diagonaalil asetsevaid punkte, toetab paralleelsust ja iseseisvust. Alammall 3 leevendab alammalli 1, lubades punkte, mis on diagonaalist kaugemal aga mitte liiga kaugel telgede alguspunktist. See lubab tihedat "tuuma" protsessi keskpunktis. See alammall aitab tasakaalustada koormust rollide vahel. Paljud organisatsioonid on kahereziimilised: tuuma rollid suhtlevad tihedalt aga välismised rollid praktiliselt üldse mitte. Alammall 3 tasakaalustab kõikide rollide koormust. Kõikide nende mallide üldiseks (ja raskelt selgitatavaks) omaduseks on, et nad parandavad toote kvaliteeti ja vähendavad toote turule toomise aega. Nad korreleeruvad juhtimise suure ulatusega, mis omakorda vähendab projekti läbiviimiseks vajalike inimeste arvu, vähendades omakorda suhtlemisele minevat töömahtu, parandades nidusust ja põhjustades antud malli rekursiivselt parendavat iseloomu.



2.29. Töö voolab sissepoole (*Work Flows Inward*)

Probleem Töö, mis lisab tootele väärtust, peab olema tehtud autoritaarsete rollide poolt.
Tegemist on organistasiooniga, kus informatsiooni voogusid on võimalik analüüsida.
Organisatsioon väljendab juhtkonna "nokkimisjärjekorda".

Jõud Mingi keskne juhtimine ja suunamine on vajalikud.
Tarkvara tegemise ajal peaks töö kitsaskohad olema süsteemi keskuses (külgnemisdiagrammil (1.5)). Kui organisatsiooni suhtlemiskeskus genereerib rohkem tööd, kui ta teeb, kujuneb organisatsiooni tööviljakus juhuslikuks ja ennustamatuks.
Arendajatele on juba turu vajadusi tutvustatud mallide Tulemüürid (2.24) ja Väravavaht (2.25) kaudu.

Lahendus Töid peavad genereerima kliendid, seda tuleb filtreerida toetavate rollide poolt ja läbi viia realiseerimise ekspertide poolt keskel. Juhte ei tohi paigutada suhtlemismaatriksi keskele, nad koormatakse üle ja teevad vähem läbimõeldud ning protsessi dünaamikat mittearvestavaid otsuseid.

Kontekst Tulemuseks on organisatsioon, mille suhtlemismaatriksil on rohkem punkte allpool diagonaali, kui selle peal.

Põhimõte Katz'i ja Kahn'i organisatsioonide analüüs näitab, et töö peab olema koondatud protsessi keskmesse ja protsessi kekpunkti peavad olema koondunud tegevused, mis lisavad väärtust (Arendaja juhib protsessi (2.11)).
Organisatsioonid, mida juhivad professionaalsed juhid omavad esindusliku äriprotsessi aga nad ei tundu saavutavat samat toolikuse taset, kui inseneride poolt juhitavad organisatsioonid. Programmeerija-kesksetes organisatsioonides on väärtust lisavad rollid protsessi keskmes (Arendaja juhib protsessi (2.11), Arhitekt realiseerib samuti (2.15)).
Juht (mänedzer) peaks toetama neid rolle ja nende tööd ning hoidma "kahjurid" eemale (mallid Patroon (2.12) ja Tulemüürid (2.24)).

2.30. Kolm kuni seitse abilist rolli kohta

Probleem Organisatsioonis on suhtlemine jaotunud ebaühtlaselt.
Organisatsiooni põhiline sotsiaalne võrk on ehitatud.

Jõud Iga antud rolli liigne sidestus koormab selle üle.

Liiga väike sidestus põhjustab rolli informatsioonilist nälgimist ja alakasutust.

Lahendus Tuleb tagada, et igal rollil oleks kolm kuni seitse abilist.

Kontekst Tulemuseks on tasakaalustatum organisatsioon, parem koormuse jaotumine ja väiksem arv isoleeritud rolle.

Põhimõte Empiirilised tulemused Pasteur'i projektist näitavad, et iga antud roll suudab alal hoida kõige enam seitse pikaajalist suhet (välja arvatud eriti tootlikes organisatsioonides, kus see arv võib ulatuda üheksani).
Rollide vaheline suhtlemine on täielik, kui iga roll suhtleb iga teise rolliga. Organisatsiooni suhtlemisküllastumust võib kirjeldada suhtlemisteede arvu ja võimalike suhtlemisteede koguarvu suhtega. Antud projekti suuruse juures on see suhtarv kõrge tootlikusega organisatsioonides suurem, kui teveliselt.

2.31. Stabiilne baas (*Named Stable Bases*)

Probleem Kui sageli tuleb integreerida?
Ajakava on paika pandud.

Jõud Pideva integreerimise korral, peavad Arendajad püüdma liikuvat märki. Liiga pikkade integreerimisvaheaegade puhul aga blokeeritakse Arendajad.
Stabiilsus on hea.
Peab toimuma areng ja see peab olema tajutav.

Lahendus Süsteemi liidesed -- arhitektuur tuleks stabiliseerida mitte sagedamini kui kord nädalas. Muu tarkvara võib muutuda (ja seda võib integreerida) veelgi sagedamini.

Kontekst Projektil on olemas sihid, see mõjutab Kliendi vaadet protsessile ja omab tugevat mõju samuti Arhitektile.

Põhimõte Malli põhiline mõte seisneb selles, et projekt peab planeerima uusi muutusi nii, et nende mõju võib aimata. Vähemtähtis on muudatuse sisu publitseerimine (see jääb suure muudatuste mahu puhul niikui nii lugemata), kui see et projektorganisatsioon mõistaks, et muudatus toimub.
Abiks on samaegselt mitme erineva stabiilsuse tasemega bassversiooni eksisteerimine. Näiteks üks AT&T projekt omas igaöist versiooni, mis garanteeritult vaid kompileerus; iganädalast versiooni, mis garanteeritult läbis algsed süsteemitestid; ja umbes kahenädast versiooni, mida loeti piisavalt stabiilseks kvaliteeditagamise testideks.

2.32. Jaga ja valitse

Probleem Organisatsioon suureneb punktini, kus ta ei suuda enam lihtsalt iseendaga toime tulla (s.t. organisatsiooni otsusetegmise mehhanism ei tööta).
Protsessi ja organisatsiooni jaoks on rollid määratud ja mõistetakse nende vahelist suhtlemist.

Jõud Kui organisatsioon on liiga suur, ei ole teda võimalik juhtida. Mittenidusad organisatsioonid on segadusseajavad ja hajutavad fookust.
Probleemide eristamine on hea. Hoolimata sellest on kasulik omada organisatsiooni piire, mis on piisavalt "kerged".

Lahendus Tuleb leida rollide kogumikud, mis on tugevalt üksteisega, aga lõdvalt teiste rollidega seotud. Selliste rollide ümber tuleb luua eraldiseisvad organisatsioonid ja protsessid.

Kontekst Iga uus alamorganisatsioon on laialdaselt iseseisev olem, millele võib antud mallide keelde kuuluvaid malle omakorda rakendada.

Põhimõte Tuleb tähele panna, et igale alamorganisatsioonile, mis tekib peale selle malli rakendamist võib rakendada suuremat osa antud rollidest, kuna alamsüsteem on omaette süsteem.

2.33. Iseseisvad faasid (*Decouple Stages*)

Probleem Kuidas tarkvara arendusprotsessi erinevad faasid (arhitektuuri disain, disain,

kodeerimine) lahti sidestada?
Tingimusel, et arendusprotsess toimub hästi tuntud valdkonnas.

Jõud Tarkvara arendusprotsessi faasid peaksid olema sõltumatud, et vähendada sidestust. Sõltumatus takistab informatsiooni voolamist, kuid loob võimaluse paralleelsuseks.

Lahendus Tuntud valdkonna (valdkondade) jaoks tuleb tarkvara arendusprotsessi faasid järjestada. Töö üleandmine faaside vahelpeab toimuma hästi määratletud liideste kaudu. See võimaldab automatiseerida ühte või mitut faasi või luua töökorraldus, kus kogemusteta tööjõud suudab faasi läbi teha.

Kontekst Uus organisatsioon (tulemuseks on organisatsioon), mis lubab spetsialiseerumist tarkvara arendusprotsessi osade järgi, selle asemel, et spetsialiseeruda kliendi probleemi järgi. See lähenemine on "ohutu" vaid hästi tuntud valdkondades, kus üleminek vajadustelt realisatsioonile on otsene (sirgjooneline).
Valdkonnad, mis on hästi tuntud, on head kandidaadid automatiseerimiseks (mehhaniseerimiseks). Vähem tuntud valdkondade puhul peab arendusprotsess toetuma kaasatud inimeste loovusele ja peab olema rohkem paralleelsust ning seotust.
See mall juhatab sisse järgmise.

Põhimõte Vaata konteksti.

2.34. Rumm, kodar ja pöid (rehv) (*Hub, Spoke, and Rim*)

Probleem Kuidas tarkvara arendusprotsessi erinevad faasid (arhitektuuri disain, disain, kodeerimine) lahti sidestada järjestatud (järjestikustatud -- *serialised*) arendusprotsessis, säilitades paindlikust (tundlikust -- *responsiveness*).
Tingimusel, et arendusprotsess toimub hästi tuntud valdkonnas ja saavutatud on esteetiline struktuur (vaata järgmist malli).

Jõud Tarkvara arendusprotsessi faasid peaksid olema sõltumatud, et vähendada sidestust. Sõltumatus takistab informatsiooni voolamist, kuid loob võimaluse paralleelsuseks.

Lahendus Seo iga roll mingi keskse rolliga, mis juhatab protsessi tegevusi (*orchestrates process activities*).
Paralleelsuse saab uuesti sisse tuua, kui keskne roll organiseerib tegevuste konveieri.

Kontekst Protsess on korrastatum ja tõenäolisemalt korratav kui vaid malli Esteetiline struktuur (2.35) rakendamisel. Protsessi disainer peab vältima keskse rolli kujunemist pudelikaelaks.

Põhimõte See mall toetub empiirilistele uuringutele ühest AT&T suurest projektist ja konveieri teooriale.

2.35. Esteetiline struktuur (*Aesthetic Pattern*)

Probleem Organisatsioonil on ebaregulaarne struktuur.
Eeldusel, et organisatsioon on ehitatud kasutades eelnevaid malle, projekt on varajases staadiumis ja organisatsioon peab planeerima, kuidas edasi areneda toote esimesest väljalaskest kuni toote toetuse ja hoolduse faasini.

Jõud Vastutuse ühtlane jaotumine on hea, kuna see jaotab töökoormuse. Regulaarseid struktuure (nagu hierarhiad) saab lihtsalt kasvatada, lisades rohkem inimesi, ilma et algne struktuur häviks. Regulaarne hierarhiline struktuur ei jaota vastutust ühtlaselt.

Lahendus Organisatsioonis peavad olema selgelt eristuvad osad, milledest võivad välja kasvada iseseisvad osakonnad, kui projekt edeneb ja laieneb, et turgu teenida.

Kontekst Organisatsioonil on alus, millelt lähtudes kasvada.

Põhimõte See mall peegeldab empiirilisi vaatlusi kõrge tööviljakusega organisatsioonidest. Selle punktini on antud mallide komplekt aidanud luua funktsionaalset, kõrge tööviljakusega organisatsiooni. Aga lisatööd tuleb teha, et lubada organisatsioonil elegantselt kasvada. Seda saab saavutada määrtaledes alamorganisatsioonide juured antud organisatsiooni sees. Kui selliseid ei leita, on võimalik et organisatsioon pole

võimeline kasvama. Näiteks on raske kasvatada Peaprogrammeerija meeskonda.

2.36. Sidestamine vähendab varjatust (passiivsust) (*Coupling Decreases Latency*)

Probleem Tarkvara arendusprotsess pole piisavalt tundlik (reaktsioonivõimeline -- *responsive*); arenduse intervallid on liiga pikad; toode ei jõua õigeks ajaks turule ("turuaknasse"). Probleem tekib teenendusprotsessis ja erijuhtudel väikestes disaini/realistasiooni protsessides, kus kasutatakse iteratiivset või inkrementaalset lähenemist.

Jõud Tarkvara arendusprotsessi faasid peaksid olema iseseisvad, et vähendada sidestust. Sõltumatus parandab võimalusi paralleliseerimiseks, kuid takistab informatsiooni voolu.

Lahendus Tuleb avada suhtlemiskanalid rollide vahel, et suurendada üldist rollide sidestatuse taset. See on eriti tähtis suhtlemisel protsessi kesksete rollidega.
Rollide vaheline suhtlemine võib olla kujundatud mallide Töö voolab sisse (2.29) ja Nihuta vastutust (2.27).

Kontekst Sidestus suurendab rollide sõltuvust üksteisest, mis ei pruugi alati soovitav (hea) olla.
Informatsiooni üleandmine käest-kätte suurendab varjatust (latentsust).
Rollide vaheline hüpete arv peaks olema iga antud probleemi jaoks minimaalne.

Põhimõte Antud mall põhineb tarkvara inseneriteadusele ja sotsiomeetrilistele põhimõtetele.

2.37. Prototüüp

Probleem Nõudmiseid mida kogutakse protsessi varajases staadiumis on raske valideerida ilma testimata.
Organisatsioon üritab koguda nõudmisi, mis on vajalikud testide planeerimiseks ja arhitektuuri disainiks. Alternatiivselt tuleb arendustöö algul välja kõrge riskiga või tundmatu ala (valdkond). Seda malli võib rakendada ka arendustöö hilisemates staadiumites aga mida varem seda rakendatakse, seda parem.

Jõud Nõudmised on pidevas muutumises. Kirjutatud nõudmised on tavaliselt liiga mitmetähenduslikud (ebaselged).
Projekt vajab nõudmistes tehtud muudatusi nii kiiresti kui võimalik.
Disainerid ja realiseerijad peavad nõudmisi vahetult mõistma.

Lahendus Ehitada prototüüp, mis aitab mõista nõudmisi. Prototüübid on eriti kasulikud välisliideste jaoks. Visata prototüüp ära, kui selle kasutamine on lõppenud.

Kontekst Tulemuseks on nõudmiste parem määratlemine, täienduseks kasutusnäidetele (*use cases*). Antud mall sobib kasutamiseks koos mallidega Kaasa kliendid (2.20) ja Stsenaariumid määratlevad probleemi (2.22).

Põhimõte Tarkvara tegijad võivad ka siin õppida klassikalisest arhitektuurist.

2.38. Ei väikestele libastumistele (*Take No Small Slips*)

Probleem Kui kaua peaks projekt kestma? Toote tegemine on pooleli ja protsessi tuleb jälgida.

Jõud Kui projekt kestab liiga kaua, muutuvad Arendajad enesega rahulolevaiks ja sobiv aeg turule tulekuks ("turuaken") võib mööda minna.
Kui projekt on liiga kiire (ajagraafik on liiga pingeline) võivad Arendajad läbi põleda ja samuti ei mahu toode "turuaknasse".
Projektid, kus puudub ajakava kestavad lõputult, Arendajad viidavad liiga palju aega siludes detaile, mis on asjasse puutumatud või ei teeni Tellija huva (vajadusi).

Lahendus Paul Chisholm selgitab lahendust: "Hea viis on elada "ilma väikeste libastumisteta" -- Müütiline inimkuu (Brooks) : Iga nädal tuleb mõõta, kui lähedal on kriitiline tee. Kui ollakse kolm päeva maas projekti ajakavast tuleb kasutusele võtta 3 päevane "pettuse" kordaja. Kui "pettuse" kordaja, on muutunud liiga suureks (naeruväärseks), tuleb ajakava

järele anda. See väldib ajakavaga loksutamist.

Kontekst Tulemuseks on paindliku tähtajaga projekt. Kuupäevi on alati raske ennustada. DeMarco märgib, et üks kõige tõsisematest märkidest raskustesse sattunud projektist on ajakava koostamine tagurpidi -- lähtudes lõpukuupäevast.

Põhimõte Antud mall toetub MIT'i projektijuhtimise simulatsioonidele ja Brooks'i raamatule "Müütiline inimkuu". Suurem osa mõistlike projekte jälgib seda malli.

2.39. Arendustöö paarides

Probleem Inimesed kardavad üksi probleeme lahendada. eelduseks on, et koodi omandus on kindlaks määratud (on rakendatud malli Koodi omandus (2.17)) ja arendustöö käib.

Jõud Inimesed tunnevad mõnikord, et nad saavad probleemi lahendada siis kui neid abistatakse. Mõned probleemid on suuremad kui mistahes indiviid.
Liiga palju inimesi ühe ja sama klaviatuuri ja ekraani taga vähendab nii tööriista kui teda kasutatavate inimeste effektiivsust.

Lahendus Tuleb panna kokkusobivaid Arendajaid paaridesse koostööd tegema. Nad võivad koos toota rohkem, kui eraldi. Samuti pole inimeste paar nii nõrk (nõrkade külgedega), kui üksikisik.

Kontekst Effektiivsem arendusprotsess.

Põhimõte Samasugune, kui mallil Rühmas valideerimine (2.21).

2.40. Katkestused takistavad blokeerimist (*Interrupts Unjam Blocking*)

Probleem Sündmused ja tööd protsessis on liiga keerukad, et kavandada arendustöö tegevusi lineaarse jadana.
Seda malli tuleks kasutada kõrge tööviljakusega disaini- või realiseerimisprotsesside või vähe (varjatud -- latentsete?) teenendavate protsesside korral. Ajakava kavandamise probleeme puudutatakse väikeses ulatuses. See tähendab, et antud malli ei tuleks kasutada osakondade töö ajakava loomiseks vaid koostööd tegevate indiviidide korral.

Jõud Ajakavast täieliku ülevaate (arusaamise) saamine on võimatu.
Programmeerijad, kelle arendustöö ajakava on pikim saavad kasu, mida rohkem teiste programmeerijate koodist on lõpetatud, ennem kui nad proovivad integreerida või testida oma loodud koodi, eeldusel, et nende töötsüklit mingil muul moel ei õnnestu lühendada.

Lahendus Kui roll tahab blokeerida kriitilist ressurssi, tuleb antud ressurssi pakkuva rolli töö katkestada, et nood ei blokeeriks teiste tööd. Kui lisakoormuson küllalt suur, ei mõjuta see läbilaskevõimet. Mall parandab alati lokaalset peidetust (latentsust?).

Kontekst Protsessil peab olema suurem läbilaskevõime -- selle eest tuleb maksta siirema sidestusega. Sidestus võib olla tulenenud varasematest mallidest (nagu Töö voolab sisse (2.29), Nihuta vastutust (2.27), Buffalo mägi (2.28) ja Sidestamine vähendab varjatust (2.36)).

Põhimõte Eesmärk on, et see mall kehtib (on kasutatav) kõige sagedamini ühistööd tegevate ja ühe projekti kallal töötavate Arendajate korral.
Võib olla kasulik määrata katkestustele prioriteedid ja teenendada neid, mis optimiseerivad organisatsiooni kui terviku tootlikust. S.t., et on parem blokeeringust vabastada 4 inimest, kui vabastada üks "kriuksuv ratas". Otsuse tegemise protsess olgu kiire, suurema osa ajast on see hajutatud. Kui läheb vaja arbitreerimist, tuleb rakendada malli Patroon (2.12). Patroon ja Mänedzer võivad meeskonna läbivaatusel projekti edasi aidata. Maranzano ütleb: "Ärge pange liiga palju kriitilisi ülesandeid ühele inimesele".

2.41. Ära katkesta katkestust (*Don't Interrupt an Interrupt*)

Probleem Mall Katkestused takistavad blokeerimist (2.40) põhjustab inimeste saalimist.
Rakendatakse malli Katkestused takistavad blokeerimist (2.40).

Jõud Üks töötaja on vältimatult blokeeritud teie tõttu -- te ei saa teha kahte asja ühe korraga.
Täieliku, kõiketeadva ennustamise ja planeerimise ootused on mõtetud.

Lahendus Kui kellegi tööd põhjustas katkestus nagu mallis 40, peab ta tagasi lükkama teised katkestused, kuni antud töö on tehtud.

Kontekst Tulemuseks samalaadne organisatsioon, kui malli Katkestused takistavad blokeerimist (2.40) puhul.

Põhimõte On lihtne kuid meelevaldne reegel -- planeerimisest ei tohiks teha laiulatusliku tseremooniat.

2.42. Tasusta edu (*Compensate Success*)

Probleem Kuidas paremini pakkuda ressurssidele sobivat motivatsiooni?
Organisatsioon koosneb arendajate grupist, kes peavad kinni pingelisest ajakavast.

Jõud Ajakava on halb motivaator, lai valik ajakavasid on tavaliselt samaväärsed antud ülesande jaoks.
Altruism ja väheisekad meeskonnad on viktoriaanlik igand.
Firmadel on sageli "tee või viska nurka" tüüpi projektid, nende juhtimine peaks olema teistsugune.
Põhjendamatult erinevad autasud motiveerivad saajaid aga võivad häirida nendega võrdseid.

Lahendus Tuleb sisse seada rohked autasud isikutele, kes aitasid kaasa edukale "tee või viska nurka" projektile. Kogu meeskond (sotsiaalne ühik) peaks saama võrreldavaid autasusid, et vältida nende indiviidide demoraliseerimist, kes hindavad ennast võrdluses teiste palgaga.
Pidutsemine on väga effektiivne autasu.

Kontekst Tulemuseks on organisatsioon, mis keskendub vähem ajakavale ja rohkem Kliendi rahuldamisele ning süsteemi alastele saavutustele.
Kõrged autasud võivad põhjustada indiviidide ülepingutust, mis viib stressile ja on potentsiaalseks riskiks projektile.

Põhimõte On olemas tugev korrelatsioon edukate tarkvaraprojektide ja eriti tulusate tasustamissüsteemide vahel.
Suurem osa Ameerika autasumehhanisme on suunatud probleemide väljajuurimisele, mitte lahenduste väljapakkumisele.
Heaks töömudeliks on arstide ja juristide grupid, kus juhid saavad vähem palka, kui töötajad.

Kirjandus

James O. Coplien, **A Development Process Generative Pattern Language**, AT&T Bell Laboratories, Proceedings of PLoP/94, Monticello, Il., August 1994, pp. 34.