O‘ZBEKISTON RESPUBLIKASI AXBOROT TEXNOLOGIYALARI VA KOMMUNIKATSIYALARINI RIVOJLANTIRISH VAZIRLIGI

MUHAMMAD AL-XORAZMIY NOMIDAGI TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

“Dasturiy injiniring” fakulteti

“Axborot texnologiyalarining dasturiy ta‘minoti” kafedrasi

Boboev L.B., Abduraxmanova N.N.

DASTURIY LOYIHALARNI BOSHQARISH

FANIDAN

O‘QUV Q O‘LLANMA

TOSHKENT 2019

Muallif(lar): Boboev L.B., Abduraxmanova N.N.

O‘quv qo‘llanma. – Toshkent: TATU. 2019. – 194 b.

Jahon amaliyotida ilmiy metodlar asosida loyihalarni ishlab chiqish oqimga aylandi.

Loyiha konsepsiyalari va loyiha boshqaruvi o‘zgarishlarni boshqarish zarurati bilan bog'liq. Loyiha boshqarish turli darajadagi menejerlarning kundalik faoliyatining ajralmas qismi hisoblanadi. Ko‘plab menejerlar yirik loyihalar bilan rasmiylashtirilgan loyihalarni boshqarish usullaridan foydalanish zarurligini shubha ostiga qo‘yadilar. Biroq, ommaviy ishlab chiqarishga qaratilgan tashkilotlarda ham loyihalarni amalga oshirilishi faoliyatning muhim qismidir.

Rasmiylashtirilgan loyihalarni boshqarish usullaridan foydalanish sizga investitsion maqsadlarni yanada aniqroq aniqlash va investitsion faoliyatni optimallashtirishni rejalashtiradi, loyiha tavakkalchiliklarini yanada yaxshiroq hisobga oladi, mavjud resurslardan foydalanishni optimallashtiradi, nizolarni bartaraf etishni, tayyorlangan rejani bajarilishini nazorat qiladi, amaldagi ko‘rsatkichlarni tahlil qiladi va ish vaqtida o‘z vaqtida tuzatishlar kiritadi, kelgusida amalga oshirilayotgan loyihalar tajribasini tahlil qilish va undan foydalanish. Shunday qilib, loyihani boshqarish tizimi butun tashkilotni boshqarish tizimining muhim tarkibiy qismlaridan biri hisoblanadi.

Loyiha boshqaruvi juda yosh ilmdir, ammo aslida uning asosiy tushunchalari XIX asr oxirida shakllantirildi. Ushbu qo‘llanmada zamonaviy boshqaruv nazariyasining ilmiy g'oyalar, ijtimoiy uslublar va asrning biznes yondashuvlaridan qanday ta'sirlanishini o‘rganasiz.

Dizayn texnologiyalaridan foydalanish qadim zamonlarda ham mavjud. Ma'lum ma'noda, etti mo‘'jizaning har birini amalga oshiriladigan ijtimoiy loyihalar deb hisoblash mumkin. Biroq, loyihalar bilan fikr yuritish, muhandislik, biznes va nihoyat ijtimoiy sohada loyiha yondashuvini ongli va keng qo‘llash 20-asrning so‘nggi uchinchi hodisasidir.

Biroq, o‘tgan asrning boshida, ayniqsa, muhandislik sohasidagi kuchli loyiha o‘zgarishlari portlashi qayd etildi.

Loyiha boshqaruvi hozirgi shaklida bir necha o‘n yillar oldin shakllana boshladi.

O‘quv qo‘llanma Muhammad al-Xorazmiy nomidagi TATU Dasturiy injiniring fakulteti talim oluvchi talabalar uchun o‘quv jarayonida foydalanish uchun mo‘ljallangan.

O‘quv qo‘llanma Muhammad al-Xorazmiy nomidagi TATU ilmiy-uslubiy kengashining qarori bilan chop etishga tavsiya etildi (2019 yil “_26_” “ mart ” “9(121)”-sonli bayonnoma).

Taqrizchilar:

Azamov S.S. -Muhammad al-Xorazmiy nomidagi TATU “AT” kafedrasi dosenti, f.m.f.n.

Zaynutdinova M.B. -Toshkent temir yo’l muhandislar instituti “Informatika va kompyuter grafikasi”kafedrasi dosenti, Phd

Muhammad al-Xorazmiy nomidagi TATU, 2019

KIRISH

Mehnat bozorining zamonaviy ehtiyojlariga muvofiq uzluksiz ta‘lim tizimini takomillashtirish va yuqori malakali kadrlar tayyorlash, ta‘lim xizmatlarining ochiqligi va sifati masalalariga davlat tomonidan alohida e’tibor qaratilmoqda.

O‘tgan yillarda ta‘limini rivojlantirish, ta‘lim jarayonini modernizatsiyalash va mazmunini yangilashni ta‘minlash maqsadida uning huquqiy bazasini mustahkamlash bo‘yicha kompleks chora-tadbirlar amalga oshirildi, iqtidorli yoshlarning xalqaro va respublika olimpiadalari hamda tanlovlarda salmoqli natijalarga erishishlarida ularni manzilli qo‘llab-quvvatlash bo‘yicha qator tizimli chora-tadbirlar ko‘rildi.

Dasturiy ta‘minot boshqaruvi ba‘zan dasturiy injiniring va dasturiy injiniringning boshqa dasturlash turlari o‘rtasidagi kalit, bu jarayonni boshqarish deb taklif bildiriladi. Bu bilan biz dastur ta‘minoti taraqqiyotini tashkiliy majburiyatlar, byudjet, jadvallar ketma-ketligiga asoslangan fan va tashkilot kabilar bilan birga o‘rin egallaganligini nazarda tutmoqchimiz. Shuning uchun boshqaruv dasturiy injiniring uchun juda ham muhimdir. Kitobning bu qismida texnikaviy boshqaruv nashridan ko‘ra dastur nashri kabi axoli boshqaruvi yoki ko‘plab tashabbuskorlik tilining strategik boshqaruvini tanishtirib o‘tildi.

1- bob. Dasturiy ta‘minot loyiha boshqaruvi va uning birinchi asosiy qismi tavakkal boshqaruv bilan bog‘liqligini o‘z ichiga oladi. Loyiha rajalash mobaynida boshqaruvchilar nima hatolikka yuz tutishi va bu haqda ular nima qila olishi mukunligi haqida tavakkal boshqaruv, loyiha boshqaruv javobgarligi kaliti hisoblanadi. Bu bob yana axoli boshqaruvi va jamoaviy ishlash bo‘limlarini o‘z ichiga oladi.

2- bob. Loyiha, rejalash va fikirlashni o‘z ichiga oladi. Metal chiziqlarni boshlang‘ich dasgoh rejasi sifatida va rivojlantiruvchi dasturni tushuntirish, muhum taraqqiy etgan usulda qolishni uddaburon usullarining muvaffaqiyat o‘rniga qabul qildim. Yana dasturiy ta‘minot texnikasi va tizimi narxini shu bobda

muhokama qilindi. Algoritmik narx namunasini tasvirlash uchun COCOMO oilasi narx namunasidan foydalndim va uning foyda zararlarni tushuntirib o‘tildi.

3-5- boblar sifat boshqaruvi mavzusiga oid xisoblanadi. Sifat boshqaruvi dasturiy ta‘minot sifatini yaxshilash va kafolatlash texnikasi va jarayonoiga daxldordir va bu mavzu 3- bobda yoritiladi. Sifat boshqaruvi standartlarining muhimligini, jarayon kofolat sifati nazoratini, dasturiy ta‘minot o‘lchovining sifat boshqaruvidagi ro‘lini muhokama qilib o‘tildi.

4- bobda shakl boshqaruvi muhokama qilinadi. Bu jamoalar tomonidan rivojlantirilgan barcha yirik sistemalar uchun muhim qismdir, Garchi talabalar dasturiy ta‘minot taraqqiyoti bilan faqat o‘zlari shaxsan nmashg‘ul bo‘lgan bo‘lsalar ham shakl boshqaruvi har qoim ham ular uchun muhim bo‘lmagan. Shuning uchun bu qismda o‘zgarish boshqaruvi, tizim tuzilishi turli boshqaruvi va shakl rejalashtirish kabi turli hil ko‘rinishdagi mavzular huhokama qilingan.

5- bobda dasturiy ta‘minot jarayoni taraqqiyoti qanday qilib jarayonlar o‘zgarishi ya‘ni maxsulot va jarayon xususiyati barobar rivojlanadi.

Umumiy taraqqiyot bosqichi jarayoni, ya‘ni taraqqiyot o‘lchovi, taxlil jarayoni va almashnuv jarayonini va almashnuv jarayonlarini muxokama qilindi. Tayyor namunalarni qisqacha tushuntirish qobiliyati va SEI‘s jarayon taraqqiyoti qobiliyatini asaslashda davom etildi.

1. LOYIHA BOSHQARUVI

Dasturiy loyiha boshqaruvi dasturiy injiniringini muhim qismi. Loyihalar boshqarila olishi kerak chunki professional dasturiy injiniring har doim majburiy tartib va buyudjet tashkilotlariga bog‘liq. Loyiha boshqaruvchilarining ishi yuqori sifatli dasturiy maxsulot yetkazib berilgan vaqtda bu majburiyatni bartaraf etadi va dasturiy liyihani yuzlashtirishni ta‘minlaydi. Yaxshi boshqaruvlar muvafaqqiyatli loyihalarni kafolatlay olmaydi. Yomon boshqaruvda odatda loyiha muvofaqiyatsiz natija bo‘lsa ham, dasturiy ta‘minot kechiktirib yetkazilar, haqiqiy muloxazadan qimmatroq bo‘lishi mumkin yoki haridorlar ko‘zda tutgan natija chiqmasligi mumkin.

Loyiha boshqaruvi uchun muvaffaqqiyatli yo‘l loyihadan loyihaga aniq o‘zgarishidur ammo ko‘p loyihalar uchun muhim maqsadlar:

1. Dasturiy ta‘minotni mijozga kelishilgan vaqtda yetkazib beradi.
2. Narxlar umumiy byudjetdan ishlatiladi.
3. Mjozning talablariga javob bera oladigan dasturiy ta‘minot yetkazib beradi.
4. Yaxshi deb o‘ylash va yaxshi funksiyani jamoaviy rivojlantirish.

Bu maqsadlar dasturi injiniring uchun yagona emas ammo bu maqsadlar barcha injinirlik loyihalari uchundir. Shu bilan birga, dasturiy injiniring boshqa turlaridan injinerlar bilan farq qiladi. Dasturiy boshqaruv o‘ziga xos qiyinchiliklarni xosil qiladi. Shuningdek dasturiy injiniring boshqa turdagi dasturiy injiniringni dasturiy boshqaruvchi o‘ziga xos qiyinchiliklarini hosil qiladi. Bazi farqlar quyidagilar:

1. Maxsulot abstract bo‘ladi. Masalan kemasozlikda yoki shahar qurilishida siz loyihani qay boschiqda ekanligini ko‘rishingiz mumkun. Belgilangan rejadan kech qolsangiz kutilgan natijaga erishilmaydi. Dasturiy taminot mavhum. Uni ko‘ra olmaysiz ham usholmaysiz ham. Dasturiy ta‘minotni yaratuvchilari yaratilayotgan artifaktlarga qarab mahsulot progresini qaysi jarayonga yetganini

koʻra olmaydi. Har bir jarayonda ishlab chiqilgan maxsulotni sinovdan oʻtkazib koʻrishadi.

2. Dasturiy taʻminotning kata loyihalari koʻp hollarda “bir martalik” boʻladi. Dasturiy taʻminotning kata loyihalari odatda oldingi loyihalardan bazi usullari bilan farq qiladi. Shuning uchun katta tajribaga ega boʻlgan dasturiy taʻminot yaratuvchilari ham kelib chiqishi mumkin boʻlgan muammolarni avvaldan bilishmaydi. Bundan tashqari kompyuter va aloqa texnologiyalaridagi keskin oʻzgarishlar natijasida yaratuvchining bilimi ortda qolgan boʻlishi mumkun. Oldingi loyihalardan olingan bilimlar yangi loyihalarga kerak boʻlmasligi mumkun.

3. Dasturiy taʻminot jarayonlari oʻzgaruvchan va maxsus tashkillangan koʻpriklar va binolar kabi bazi tizimlarning injinerlik jarayonlari yaxshi tushunuladi. Lekin bita tashkilotning dasturiy jarayonlari boshqasinikidan farqlanadi. Standarlashtirish va muvofiqlashtirish sohasida katta tarqqiyotga erishganimizga qaramay biz hozirgacha oldindan ishonch bilan aytolmaymiz dasturiy taʻminotning aniq jarayoni ishlab chiqarishda muammo keltirishni aytolmaymiz. Dasturiy taʻminot loyihasi injinir loyihalashning katta tizimining qismi boʻlganida toʻgʻri boʻladi.

Bu muamolarning natijasida bazi dasturiy loyihalar byudjetdan chiqib ketishi va rejadan kechikisi ajablanarlik emas. Dasturiy tizimlar koʻpincha yangi va texnik tomondan innovatsion boʻladi.

Innovatsion boʻlgan injiner loyihalarda(yangi transport tizimlar kabi) ham grafikka amal qilishda muammolar bor. Bu muammolarni boʻlsa ham koʻplab dasturiy loyihalar vaqtida va byudjetdan chiqib ketmagan holda bajarilishi taxsinga loyiq.

Dasturiy taminot yaratuvchisi uchun ishlashning standar tasnifni yozib berishni ilojisi yoʻq. Tashkilot va yaratilayotgan dasturiy taʻminotga qarab ish tartibi har-xil boʻladi. Lekin koʻp yaratuvchilar faoliyatning quyidagi turlarining baʻzilari yoki hammasi uchun qasidur bosqichida javobgarlikni oʻz boʻyniga olishadi:

1. Loyihani rejalashtirish Loyiha boshqaruvchilari rejalashtirishga, baxolashga loyiha rivojlanish rejasiga va odamlarni vazifalarini bo‘lib berishga javob beradi. Ular qo‘yilgan ishni talabarga mos holda bajarilishini kafolatlashi uchun, jarayon vaqtida bajarilishini va byudjetdan chiqib ketmasligini nazorat qilishadi.

2. Hisobot berish. Loyiha boshqaruvchilari odatda mijoz yoki dasturiy ta‘minot ishlkab chiquvchi kampaniya boshqaruvchilariga loyiha ketishi bo‘yicha ma‘lumot berishga javobgardurlar. Bular har-xil darajada muloqot qilishga qodir bo‘lishi kerak, texnik axborotdan to boshqaruv qarorigachabatafsil tuna olishi kerak. Ular loyiha bo‘yicha batafsil xisobotni muhum axborot kabi qisqa va aniq hujjatlarni tayyorlay olishi kerak. Ular bu malumotlarni qayta ko‘rish jarayon vaqtida taqdim eta olishi shart.

3. Tavakkal boshqaruv. Loyiha boshqaruvchilari loyihaga ta‘sir etishi mumkin bo‘lgan xatarlarni baholashi, ularni monitoringini bajarishi kerak va muammolar paydo bo‘lganda chorasini ko‘rishi kerak.

4. Hodimlarni boshqarish. Loyiha boshqaruvchilari odamlar jamoasini boshqaruvga javobgardir. Ular o‘zining jamoasiga odamlarni tanlashi kerak va jamoa berilgan topshiriqlarni samarali bajarishi uchun ish usullarini belgilashi kerak.

5. Takliflarni yozish. Dasturiy ta‘minot loyihasini birinchi bosqichida ishni bajarish va shartnomani yutish uchun taklif yozilishi mumkun. Taklifda loyiha amaqsadlari va ularning bajarilishi tasnif etiladi. Bunda odatda ish grafik va narxi baxolanadi va shu bilan loyiha shartnomasi shu tashkilot yoki jamoaga berilishi maqsadga muofiligi ko‘rsatiladi. Taklifni to‘gri tuzish muhum vazifalardan biridur chunki ko‘plab dasturiy kompaniyalarning taraqqiy etishi qabul qilingan takliflar va tuzulgan shartnomalar yetrlik miqdorda ekanligiga bog‘liq. Bu masalani yechishda yo‘naltiruvchi tamoyillarni o‘rnatishni ilojisi yo‘q; taklifni yaratish bu amaliyot va tajriba orqali ko‘nikma hosil qilish.

Bu bobda men asosiy etiborni havf-hatarlarni boshqarish va xodimlarni boshqarishga qaratganman. Men loyihalarni rejalashtirish haqidagi ma‘lumotni 2 bobda keltirib o‘tiladi.

1.1. Havf-hatarni boshqarish

Havf-hatarni boshqaruvi loyiha boshqaruvchisi uchun eng muhim ishlardan biri. Havf-hatarni boshqarish loyiha jadvalini yoki ta'sir ko‘rsatishi mumkin bo‘lgan havflarni o‘z ichiga oladi va dasturiy ta'minot sifatli ishlashi uchun bu havf-hatarlalarni oldini olish uchun choralar qilinishi talab etiladi (Hall, 1998; Ould, 1999).

Siz biror bir havf sodir bo‘ladi deyishingiz uchun aniq ishonch hosil qilishingiz kerak. Xatarlar ishlab chiqilgan dasturiy ta‘minotga tahdid qilishi mumkin, yoki tashkilotga. Havf-hatarning uch tegishli toifasi mavjud:

1. Loyihaga jadvaliga yoki resurslariga bo‘ladigan havf-hatarlar. Misol uchun, loyiha havfi tajribali dizayner yo‘qolishi hisoblanadi. Almashtirish natijasida tegishli tajriba va ko‘nikmalar dizayner uzoq vaqt talab qilishi mumkin va binobarin, dasturiy ta'minot dizayni o‘z ishini bajarish uchun ko‘proq vaqt talab etadi.

2. Mahsulotga bo‘ladigan havf-hatarlar, ushbu havf-hatarlar dasturiy ta'minot sifatini yoki ish faoliyatini buzilishiga ta'sir ko‘rsatadigan hatarlardir. Misol bir mahsulot tarkibiy qismi kutilgan sifatdan past holatga tushishi, bu tizimining umumiy ish faoliyatini ta'sir qilishi mumkin, shunday qilib u kutilganidan ancha sekin bo‘ladi.

3. Biznessdagi havf-hatarlar yoki tashkilot dasturiy ta‘minoti ta'sir qiladigan havf-hatarlar. Misol uchun, bir yangi mahsulot joriy qilinganda raqib tomonidan amalga oshiriladigan havf. Raqobatbardosh mahsulot joriy etish, bu mavjud dasturiy maxsulotlar savdosi haqida juda o‘ylangan bo‘lishi mumkin.

Albatta, bu havf turlari haqidadir. Tajribali dasturchi agar ular zudlik bilan almashtirildi bo‘lsa ham, bu loyiha havfi bo‘lishi mumkin va loyihasi jadvaliga

ta'sir qiladi. Bu muqarrar bir yangi loyiha a'zosi uchun vaqt talab etadi, ular zudlik bilan samarali bo‘lishi mumkin emas, shuning uchun, qilingan ishlarni tushunish talab etiladi. Binobarin, tizimi yetkazib jo‘natilmasligi mumkin. Lekin yo‘qotish o‘zgartirish sifatida bo‘lishi mumkin emas, chunki jamoa a'zosi, shuningdek dasturlash xatolarni qilishi mumkin va bu mahsulot havf bo‘lishi mumkin, shuning uchun tajribali dasturchi uchun bir havf bo‘lishi tajribasining yangi marrasini qozonishda muhim bo‘lishi mumkin.

Siz loyiha, mahsulot va biznes uchun havf-hatarlar oqibatlarini belgilash va ularning oqibatini tahlil qilish bilan birga loyiha rejasida bo‘lishi mumkin havf-hatarlarni taxlil qilishigiz va tahlil natijalarini yozib borishingiz kerak. Samarali boshqarish uchun havf-hatarni osonroq muammolari bilan yengish uchun va bu nomaqbul byudjet hosil qilish uchun ishlatiladi.

Loyihaga ta'sir ko‘rsatishi mumkin, muayyan havf-xatar loyiha va dasturiy ta'minot ishlab chiqilgan tashkiliy atrof-muhitga bog'liq. Biroq, u erda ishlab chiqilgan dasturiy ta'minotni turiga bog'liq bo‘lmagan, shuningdek, umumiy havf-xatar bor va bu har qanday loyihada sodir bo‘lishi mumkin. Bu umumiy xatarning ayrimlari 1.1- rasmda ko‘rsatilgan.

Havf-hatarlarni boshqarish dasturiy loyihalari uchun muhim ahamiyatga ega. Buning oldini olish uchun dasturiy ta'minot ishlab chiqish uchun zarur bo‘lgan vaqt va resurslarini baholash va turli individual ko‘nikmalarini o‘qishadi. Siz xatarlarni ta'sirini oldindan tushuna olishingiz kerak; Sizda favqulodda rejalarini tuzishga to‘g‘ri kelib qolishi mumkin, shuning uchun kelib chiqishi mumkin bo‘lgan havf sodir bo‘lsa zudlik bilan ma‘lumotlarni qayta tiklash choralarini ko‘rishingiz mumkin.

Havf-hatar	Ta‘siri	Bayoni
Hodimlar aylanmasi	Loyiha	Tajribali xodimlari undan oldin loyihani tark etadi.
Boshqaruvni o‘zgartirish	Loyiha	Turli ustuvor boshqarish bilan tashkiliy o‘zgarish bo‘ladi.

Uskuna bo‘lmasa	Loyiha	Loyihasi uchun muhim ahamiyatga ega apparat qilinmaydi, jadval bo‘yicha etkazib beriladi.
Talablar o‘zgarishi	Loyiha va mahsulot	O‘zgarishlar soni talablar uchun kutilganidan katta bo‘ladi.
Kechikishlardagi o‘ziga hos hususiyatlar	Loyiha va mahsulot	Muhim interfeyslarni texnik xususiyatlari emas jadvaliga mavjud.
O‘lcham raqami	Loyiha va mahsulot	Tizimining hajmi jiddiy ahamiyatga ega bo‘ladi.
Case vositasidagi uzilishlar	Mahsulot	Kase vositalari, loyihani qo‘llab-quvvatlash, kutilgan sifatda amalga oshirish.
Teknologiya o‘zgarishi	Ish	Asosiy texnologiya qaysi tizim haqida yangi texnologiyalar bilan o‘rnini almashtiradi.
Mahsulot raqobati	Ish	Raqobatbardosh mahsulotni oldin bozoriga taqdim qilishadi kegin tizim yakunlandi.

1.1- rasm. Odatiy loyiha, mahsulot va biznes havf-hatarlarha misollar

Havf-hatarlarni boshqarish jarayonini bir nechta rejasi 1.2- rasmda ko‘rsatilgan. u bir necha bosqichlarini o‘z ichiga oladi:

1. havf-hatarlar aniqlash, siz iloji boricha loyiha, mahsulot va biznes yo‘nalishini aniqlashingiz kerak.

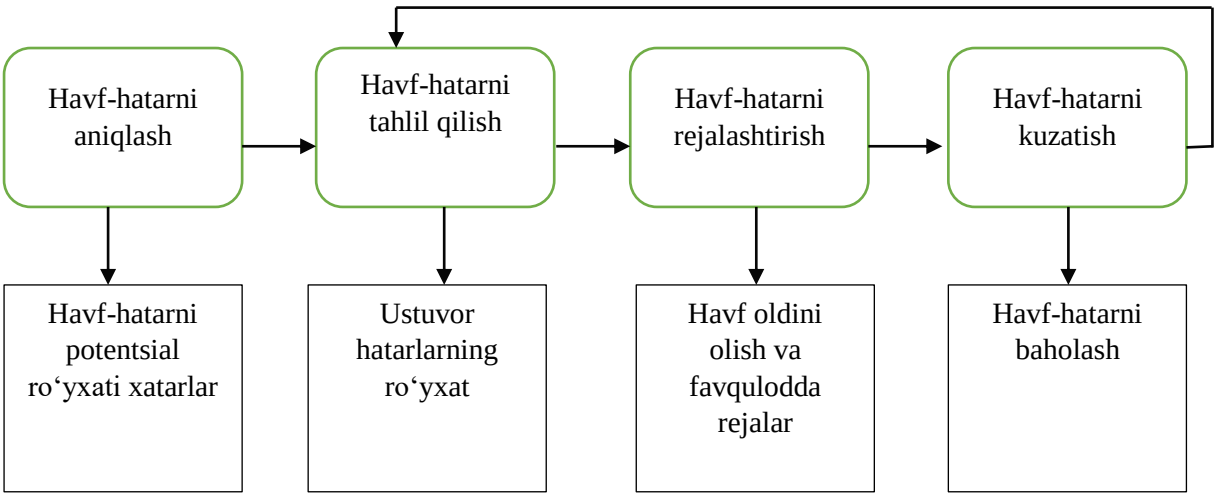
2. Siz hatarlarni ehtimolini tahlil qilishigiz va bu hatar oqibatlarini baholashingiz kerak.

3. havf-hatarni rejalashtirish, siz havf-hatardan qochib havfini bartaraf qilish uchun rejalar qilish kerak yoki loyihasi bo‘yicha o‘z ta'sirini kamaytirish yo‘llarini topshingiz kerak.

4. havf monitoring, siz muntazam ravishda havf va havf uchun rejalarni baholashingiz kerak va havf haqida ko‘proq bilib, bu qayta ko‘rib chiqishingiz kerak.

Siz havf-hatarlarni boshqarish jarayonini natijalarini hujjatlashtirishingiz kerak ya‘ni boshqarish rejasi tuziladi. Ushbu loyihadagi havflar bir muhokama o‘z ichiga olishi kerak, Agar havfi bo‘lsa boshqarish uchun taklif qanday bir bu xatar tahlili va axborot bu muammo bo‘lishi mumkindek ko‘rinadi.

Havf-hatarlarni boshqarish jarayoni davomida davom jarayon loyihani yengillashtiradi. Agar dastlabki havf boshqaruv rejasi tuzilgandan so‘ng, siz kuzatib turgan vaziyatda paydo bo‘lgan havflarni aniqlash uchun havf-hatarlar haqida ko‘proq ma'lumot olish ta'minotining bo‘lishi kerak. Keyin siz havf oldini olish va favqulodda boshqarish uchun rejalarini o‘zgartirishgiz kerak bo‘ladi.



1.2- rasm. Havf-hatarni boshqarish jarayonlari

1.1.1. Havf-hatarni aniqlash

Havf-hatarni aniqlash havf-hatar boshqaruvi jarayonining birinchi qadamidur. Bu ishlab hiqilayotgan dasturiy ta‘minotning yoki taraqqiy etiyotgan tashkilotdagi dasturiy injiniring jarayonlari uchun jiddiy havf tug‘diradigan hatarlarni aniqlash bilan bog‘liq. Havf-hatarni aniqlash jamoa bilan bajariladigan jarayon bo‘lishi mumkun bunda jamoa to‘planib mumkun bo‘lgan havf-haratlarni guruh bo‘lib muhokama etadi. Alternativa sifatida loyiha boshqaruvchisi mumkun bo‘lgan yoki hal qiluvchi hatarlarni aniqlashda o‘z tajribasidan foydalanishi mumkun.

Havf-hatarlarni aniqlashda tayanch nuqta sifatida hatarlarni har-xil turining maxsus ro‘yhati ishlatilishi mumkun. Havf-hatarning maxsus ro‘yhatiga kiritish uchun kamida 6 ta hatar turi mavjud:

1. *Texnologik hatarlar.* Tizimlarni yaratishda foydalaniladigan dasturiy yoki apparat texnologiyalaridan kelib chiquvchi hatarlar.

2. *Odamlar bilan bog‘liq hatarlar.* Yaratuvchilar jamoasidagi odamlar bilan bog‘liq hatarlar.

3. *Tashkiliy hatarlar.* Dasturiy ta‘minot yaratiladigan tashkiliy muhit bilan bog‘liq hatarlar.

4. *Vositalar bilan bog‘liq hatarlar.* Tizimlarni ishlab chiqishda foydalaniladigan dasturiy vosita va qo‘llab quvvatlashni boshqa dasturiy ta‘minot bilan bog‘liq hatarlar.

5. *Talablar bilan bog‘liq hatarlar.* Mijozning talablarini va boshqaruv jarayonlarini o‘zgartirish talablari bilan bog‘liq hatarlar.

6. *Hatarlarni baxolash.* Tizimni qurishga kerak bo‘lgan manbalarni baxolashda paydo bo‘lgan hatarlar.

Har bir turdagi hatarning misollari 1.3- rasmda berilgan. Haratalarni aniqlash jarayoni yakunlangandan keyin siz paydo bo‘lishi mumkun bo‘lgan va maxsulot, jarayon va biznesga tasir o‘tkazishi munkun bo‘lgan hatarlarning uzun ro‘yhatiga

ega bo‘lishingiz kerak. Keyin bu ro‘yhatni o‘zizga qulay holga keltirishingiz kerak. Agar sizda hatarlar juda ko‘p bo‘lsa ularni hammasini kuzatishni ilojisi yo‘q.

1.1.2. Havf-hatarni taxlil qilish

Havf-hatarni taxlil qilish jarayoni mobaynida siz har bir aniqlangan havf-hatarni ko‘rib chiqishingiz va uning extimolligi va jiddiyligi haqida xulosa chiqarishingiz kerak. Bu ishni bajarishni oson usuli yo‘q. Siz oldingi bajargan loyihalar bajarishda paydo bo‘lgan muammolarni tajribasiga tayanishingiz kerak.

Havf-hatar turlari	Yuz berishi mumkun bo‘lgan havf-hatarlar
Texnologiya	Tizimda ma‘lumotlar mazasi nazarda tutilgan bir-xil turdagi ko‘p tranzaksiyalar jarayonidan foydalanaolmaydi.(1) Dasturiy komponentalar qayta foydalanishdagi kmchiliklarni o‘z ichiga oladi, asosan ular rejadan foydalana olmaydi.(2)
Odamlar	U shtatlatga ustalarni imkonsiz jalb qiladi.(3) Shtat kaliti kritik vaqtda foydasiz bo‘ladi.(4) Shtatlar foydasizligi uchun hodisalarga ehtiyoj ezilmaydi.(5)
Tashkilot	Tashkilotlar qayta tuzili turli boshqaruvlar loyihalar uchun hisobot beradi.(6) Tashkilot loyiha byudjeti finansiviy muammolari kuch bilan kamayadi.(7)
Asbob-uskunalar	Kod generatori dasturiy ta‘minot asbob-uskuna kodi davri unimsizdur.(8) Dasturiy asbob-uskunalar bir yo‘lda birga ishlolmaydi.(9)
Ehtiyojlar	O‘zgaruvchilar muhim loyihalarni qayta ishlashni taklif etishda ehtiyoz sezadi.(10) Haridorlar o‘zgarishlarga ehtiyojlariga ta‘siz ko‘rsatishini tushunadi.(11)

Xulosa	Dasturiy ta‘minotni jivojlantirishda vaqtga ehtiyoj borligini chamalash.(12) Kamchilik qiymatini tuzatish chamalanadi.(13) O‘lcho‘v dasturiy ta‘minotni chamalaydi.(14)

1.3- rasm. Turli hildagi havf-hatarlarga misollar

Har bir havf-hatarni extimolligi va jiddiyligini aniq raqamli baxolashni ilojisi yo‘q. Ko‘plab guruhlarni ichidan biriga siz havf-hatarni belgilashingiz kerak:

1. Havf-hatar extimolligi juda past (10%), past (10-25 %), o‘rtacha (25-50%), balant (50-75 %) yoki juda baland deb baholanishi mumkun.

2. Havf-hatar ta‘sirlari katastrafik (loyihani yashashiga havf soladi), jiddiy (asosiy ushlanishlarni chaqiradi), chidasa bo‘ladigan (oldindan bilib bo‘lmaydigan sharoitlar chegarasidagi ushlanishlar) yoki hisobga olinmaydigan deb baxolanishi mumkun.

Havf-hatarlarni jiddiyligigga qarab siz taxlil jarayonini natijalarini jadvallarini ko‘rinishiga keltirishingiz kerak. 1.3- rasmda keltirilgan havf-hatarlarni taxlili 1.4- rasmda ko‘rsatilgan. Ko‘rirnib turibtiki extimollik va jiddiylikni baxolash bu yerda ixtiyoriy. Bu baxolashni amalga oshirish uchun siz loyiha, jarayon, yaratuvchilar jamoasi va tashkilot haqidagi batafsil axborotga ega bo‘lishingiz kerak.

Albatta havf-hatar ta‘sirini extimolligi va baxolanisi o‘zgarishi mumkun chunki havf-hatar haqida ko‘proq axborot ega bo‘lishingiz va havf-hatarni boshqarish rejalari bajarilishi mumkun. Shuning uchun siz bu jadvalni havf-hatar jarayonining har bir qayta ko‘rinishi bilan yangilashingiz kerak.

Havf-hatarlar taxlil qilinib va baxolangandan keyin siz ulardan qay birlari muhumligini aniqlashingiz kerak. Siz qarorni havf-hatar paydo bo‘lish extimolligi va bu havf-hatarni ta‘sirini hisobga olgan holda qarorni qabul qilishingiz kerak. Aslida katastrofik deb paydo bo‘lish extimolligi o‘rtachadan balant bo‘lgan jiddiy havf-hatarlar hisoblanish kerak.

Havf-hatar	Ehtimollik	Natija
Tashkilot loyiha byudjeti finansiviy muammolari kuch bilan kamayadi.(7)	Kuchsiz	Halokat
U shtatlatga ustalarni imkonsiz jalb qiladi.(3)	Kuchli	Halokat
Shtat kaliti kritik vaqtda foydasiz bo‘ladi.(4)	O‘rtacha	Jiddiy
Dasturiy komponentalar qayta foydalanishdagi kmchiliklarni o‘z ichiga oladi, asosan ular rejadan foydalana olmaydi.(2)	O‘rtacha	Jiddiy
O‘zgaruvchilar muhim loyihalarni qayta ishlashni taklif etishda ehtiyoz sezadi.(10)	O‘rtacha	Jiddiy
Tashkilotlar qayta tuzili turli boshqaruvlar loyihalar uchun hisobot beradi.(6)	Kuchli	Jiddiy
Tizimda ma‘lumotlar mazasi nazarda tutilgan bir-xil turdagi ko‘p tranzaksiyalar jarayonidan foydalanaolmaydi.(1)	O‘rtacha	Jiddiy
Dasturiy ta‘minotni jivojlantirishda vaqtga ehtiyij borligini tushuniladi.(12)	Kuchli	Jiddiy
Dasturiy asbob-uskunalar bir yo‘lda birga ishlolmaydi.(9)	Kuchli	Bardoshli
Haridorlar o‘zgarishlarga ehtiyojlariga ta‘siz ko‘rsatishini tushunadi.(11)	O‘rtacha	Bardoshli
Shtatlar foydasizligi uchun hodisalarga ehtiyoj sezilmaydi.(5)	O‘rtacha	Bardoshli
Qiymat kamchilikni tuzatishni tushunadi.(13)	O‘rtacha	Bardoshli
O‘lcho‘v dasturiy ta‘minotni tushunadi.(14)	Kuchli	Bardoshli
Kod generatori dasturiy ta‘minot asbob-uskuna kodi davri unimsizdur.(8)	O‘rtacha	Ahamiyatsiz

1.4- rasm. Havf-hatar turlari va ularga misollar

Boehm (1988) eng balant 10 ta havf-hatarni aniqlash va nazorat qilishni tavsiya etadi lekin men o‘ylaymanki bu raqam ixtiyoriy bo‘lgani yaxshi. Nazorat qilinadigan havf-hatarlarni miqdori loyihaga bog‘liq bo‘lishi kerak. Bu 5 bo‘lishi yoki 15 bo‘lishi munkun. Lekin nazorat qilish uchun tanlangan havf-haratlar miqdori bosjqarilishi kerak. Havf-hatarlarni juda katta miqdori ko‘p axborotni yig‘ishni talab etadi. 1.4- rasmda aniqlangan hatarlarni ichidan katastrafik yoki jiddiy oqibatlarga olib keladigan 8 ta havf-hatarni ko‘rib chiqich yetarli(1.5- rasm).

1.1.3. Havf-hatarni rejalashtirish

Havf –hatarni rejalashtirish jarayoni aniqlangan havf-hatarlarni ichidan muhumlarini ko‘rib chiqadi va bu havf-hatarlarni boshqarish yo‘llarini ichlab chiqaradi. Havf-hatarni har biri uchun u loyihaga ta‘sir etib chiqaradigan muammolar natijasidagi loyiha buzulushlarini kamaytirish uchun qanday chora ko‘rinishini ko‘rib chiqishingiz kerak. Muommolar oldini olish maqsadidda loyihani nazorat qilish uchun siz kerak bo‘lgan axborotni yig‘ishingiz kerak. Oldindan kutulmagan holatlar uchun rejalashtirishni iloji yo‘q. Bunda yana loyiha boshqaruvchisining qarori va tajribasiga suyaniladi.

Havf-hatar	Strategiyalar
Tashkilotni iqtisodiy muammolari	Qisqa dakument yuqori darajadagi biznesni ko‘rsatib bera olishi kerak va byudjet loyihalarni taqdimotini kesa va amalga oshmastli mumkun.
Xodimlar muammosi	Ziyrak mijozlar mumkun bo‘lgan qiyinchiliklar va yuz berishi mumkun bo‘lgan kechikishlar har tomonlama tekshirib sotib oladi.
Xodimlar kasalligi	Qayta tashkil etilgan jamoa ko‘p ketma-ket ish va xodimlarni shuningdek boshqa bir ishlarni tushunadi.
Nuqsonli qismlar	Patensial qayta joylash nuqsonli qismlarni ishonchli bilimi.

Talablarni o‘zgartirish	Talablarni almashtirish ta‘sirini baxolash uchun andoza alingan ma‘iumotni olish; yuqori darajadagi malumotlarni yashirish.
Tashkiliy qayta qurishlar	Qisqa dakument yuqori darajadagi biznesni ko‘rsatib bera olishi kerak.
Ma‘lumotlar bazasida ishlash	Har tomonlama tekshirish yuqori darajadagi ma‘lumotlar bazasini sotib olish imkonini beradi.
Qochirib ishlab chiqilgan vaqt	Qisimlarni sotib olishni har tomonlama tekshirish; dasturiy maxsulotdan har tomonlama foydalanish imkonini beradi.

1.5- rasm. Havf-hatarni boshqaruvchisiga yordam beruvchi strategiyalar

Yana, oddiy jarayon favqulodda rejalashtirish uchun ta'qib qilinishi mumkin, u hukm va loyiha boshqaruvchisi tajribasiga tayanadi. 1.5- rasmda havf boshqarish strategiyasi belgilab berilgan imkoniyatlarni ko‘rsatadi va asosiy hisoblashlari uchun (ya'ni, jiddiy yoki ishonchli), holati 1.4- rasmda ko‘rsatilgan. Bunday strategiyalar uch toifaga bo‘linadi. Bu strategiya quyidagi:

1. Zarur bo‘lgan strategiyasi, ushbu holatda paydo bo‘lish havf-hatar ehtimollik kamayadi. Havf-hatar uchun zarur bo‘lgan strategiya namunasi 1.5-rasmda ko‘rsatilgan, ushbu strategiya nuqsonli komponentlar bilan muomala uchun ishlatiladi.

2. Minimallashtirish strategiyalari degan ma‘noni anglatadi va havf-hatar ta'siri kamayadi. Havf-hatarni minimallashtirish strategiyalariga bir misol 1.5-rasmda xodimlari uchun kasallik bo‘limida keltirilgan.

3. Favqulodda rejalari uchun tayyorlanganlik ma'nosini anglatadi va eng yomon shug'ullanish o‘rniga favqulodda rejalar ishlatiladi, tashkiliy moliyaviy muammolar uchun strategiya 1.5- rasmda ko‘rsatilgan.

Siz uchun muhim tizimlarida ishlatiladigan strategiyalar bilan bu erda aniq holatda ko‘rishingiz mumkin, qachonki siz muvaffaqiyatsizliklarni tark etishingiz

yoki oldini olishingiz kerak. Shubhasiz, strategiyadan foydalanish orqali havf-hatarlarni oldini eng yaxshi samara beradi. Bu bo‘lmasa sizda havf-hatar mavjud bo‘ladi.

Havf-hatarlarni turlari	Potentsial ko‘rsatkichlari
Texnologiya	apparat yoki dasturiy ta'minot qo‘llab-quvvatlash; ko‘p xabar texnologiyasi muammolari.
Hodimlar	xodimlarning yomon tushkunlikka tushishi; jamoa a'zolari orasida kambag'al munosabatlar;
Tashkiliy	Tashkiliy so‘zlashuvlar; yuqori boshqaruv tomonidan harakatlar yo‘qligi
Dastgohlar	CASE vositalaridan foydalanishni jamoasi a'zolari tomonidan istamaslik; haqida shikoyat; ish stantsiyalari uchun kerali dastgohlar talabi yaxshi qo‘llab-quvvatlanadi
Talablar	Ko‘pchilik talablar so‘rovlarini o‘zgartirish; mijoz shikoyatiga ko‘ra
Baholash	Tashkilot etishmovchiligi kelishilgan jadval asosida qondirish uchun ishlatiladi;

1.6- rasm. Havf – hatar indikatorlari

1.1.4. Havf-hatar monitoring

Havf-hatarlarni monitoring, mahsulot haqida taxminlarga tekshirish jarayoni tushuniladi. Siz muntazam ravishda har bir havf-hatarni baholashingiz kerak. Havfi ko‘proq yoki kamroq ko‘rinishga ega bo‘lib kelayotganini ko‘rishigiz kerak yoki yo‘qligini, belgilangan hatarlar haqida qaror qabul qilasiz. Bundan tashqari, xavf ta'siri o‘zgardi yoki yo‘qligini haqida o‘ylashimiz kerak. Buning uchun, boshqa omillar qarash kerak, talablarga soni va uning oqibatlari haqida maslahatlar berisiz, shunda talablarini o‘zgartirishingiz mumkin. Bu omillar xavf turlari bo‘yicha albatta bog'liq. 1.6- rasmda ba'zi omillar misollar berilgan bu havf –hatar turlarini baholashda foydali bo‘lishi mumkin.

Siz INA loyihasi havf-hatarlarning barcha bosqichlari muntazam monitoring qilishingiz kerak. Har boshqaruvi sharh, siz ko‘rib va thekey har bir alohida-alohida xavfi muhokama kerak. Siz qaror qilsangiz havf-hatar kamroq bo‘ladi.

1.2. Hodimlarni boshqarish

Dasturiy taminot ishlab chiqaruvchi tashkilotlarda ishlayotgan hodimlar tashkilotning qimmatli hodimlaridir. Bu yaxshi hodimlarni(mutaxassislarni) ishga jalb qilish va saqlab qolishda ko‘p ahamiyatga ega va tashkilotning sarmoyadan mumkin bo‘lgan eng yaxshi foyda kelishini ta`minlash ham dasturchilarga bog`liq. Mivofaqqiyatli kompaniyalar va xo‘jaliklarda hodimlar tashkilot tomonidan xurmat qilinganda, malakasi va tajribasini ko`rsatadigan javobgarlik belgilanganda erishiladi. Dasturiy loyiha boshqaruvchilari dasturiy ta‘minot rivojlanish ishiga ta‘sir qiluvchi texnik masalalarni tushunishi muhim. Shunga qaramay, baxtga qarshi, yaxshi dasturiy injinerlar kerakli darajada yaxshi shaxs boshqaruvchilari emas.

Dasturiy injinerlarda kuchli texnik malaka bor, lekin loyiha yaratuvchi jamoani umumlashtirish va boshqarish imkoniyatini beruvchi hislat yo‘q. Loyiha boshqaruvchisi sifatida, siz hodimlarni boshqarishning potensial muammolaridan xabardor bo‘lishingiz va boshqaruv malakangizni oshirishga urinish kerak.

Mening fikrimcha, hodimlarni boshqarishda to‘rtta muhim omil bor:

1. Loyiha jamoasidagi bir xil hodimlarga qiyosiy yo‘sinda munosabat bo‘lishi kerak. Hech kim hamma mukofot bir xil bo‘lishini kutmaydi lekin hodimlar ularning tashkilotga hissasi past baholanganini his etmasligi kerak.

2. Turli hodimlarda turli mahorat borligini hurmat qiling va menejerlar bu farqlarni hurmat qilishi lozim. Jamoaning har bir a`zosiga o‘z xissasini qo‘shishi uchun imkon berish kerak. Bazi hollarda, albatta, hodimlarni jamoaga mos emas deb hisoblaysiz va ular davom etolmaydi, lekin loyihaning dastlabki bosqichlarida hulosaga shoshilmaslik kerak.

3. Jalb qilingan hodimlar boshqalar uni tinglayotganini his etganda samarali hissa qo‘shadi va taklifini ham hisobga oladi. Ishchi muhitni rivojlantirish juda muhim qayerdaki, hamma fikrlar, hatoki u eng quyi hodimga tegishli bo`lsa ham, hisobga olingan joyda.

4. Menejer sifatida rostgo‘ylik, siz doim jamoada nima yaxshi v anima yomon ketayotganligi haqida rostgo‘y bo‘lishingiz lozim. Siz texnik bilim darajangiz haqida va ochiq ko`ngillik bilan zarurat bo‘lganda yetarli bilim ta‘minlash uchun ishni kechiktirishda ham rostgo‘y bo‘lishingiz kerak. Agar siz bilimsizlik va muammolarni yopib yuborsangiz, sizdan habar topishadi va jamoani hurmatini yo‘qotasiz.

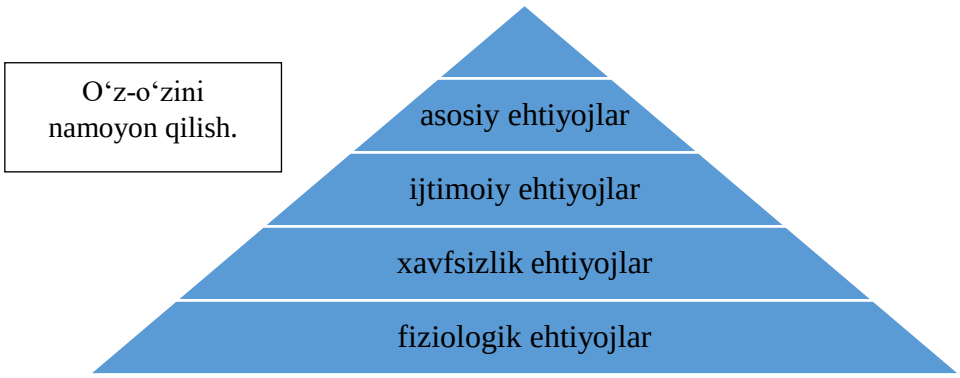
Hodimlarni boshqarish, fikrimcha, tajribaga bog‘liq bo‘lgan narsa, kitobdan o‘rgangandan ko‘ra. Bu bo‘limdan mening maqsadim va keyingi jamoaviy ish haqidagi bo‘lim oddiygina muhim insonlarni va jamoani boshqarishdagi dasturiy loyiha boshqaruviga ta‘sir etuvchi muammolarni tanishtirish. Umid qilamanki, texnik talantli jamoalar bilan kelishishda menejerlar duch keladigan bir qancha muammolarga ta`sirchanligingizni oshirdi.

1.2.1. Hodimlarni undash

Loyiha boshqaruvchisi sifatida, siz o‘ziz bilan ishlovchilarni undashingiz kerak ular o‘zlarining bor imkoniyatlatlarini qo‘shishi uchun. Undash ishni va ishlash muhitini hodimlarni imkon darajasida samarali ishlashi uchun ruhlantiradigan qilib tashkil etishni anglatadi. Agar hodimlar undalmasa, ular qilayotgan ishlariga qiziqmaydi. Ular sekin ishlaydi, hato moyillik ko‘proq bo‘ladi va jamoaning yoki tashkilotning keng maqsadlariga hissasini qo‘shmaydi.

Bu ruhlantirishni ta`minlash uchun siz hodimlarni nima ruhlantirishi haqida ozroq tushunishingiz kerak. Maslow (1954) hodimlarni ehtiyojlarini qondirilshi orqalali ruhlantirishini maslahat beradi. Bu ehtiyojlar bir qancha darajalarda 1.7 grafikda ko`rsatilgandek tuzilgan.

Ierarxiyaning quyi darajalari fundamental ehtiyojlar, uyqu, ozuqa va boshqalar. Ijtimoiy ehtiyojlar ijtimoiy guruhning a‘zosi bo‘lish bilan bog‘liq.



1.7- rasm. Insoniyatning kerakli ierarxiyasi

Izzat-ehtiromga bo‘lgan ehtiyoj boshqalar tomonidan hurmat qilinishni ko`rsatadi va o‘zini anglashga bo`lgan ehtiyoj shaxsiy rivojlanish bilan bog`liq. Hodimlar avval quyi darajadagi ehtiyojlarni qondirishi kerak so`ngra yuqori darajalilarini.

1. Ijtimoiy ehtiyojlarni qondirish uchun, siz hodimlarga hamkorlari bilan uchrashishga vaqt berishingiz va ularni uchrashishi uchun joy bilan ta‘minlashingiz kerak. Bu judayam oddiy barcha jamoa azolari bir joyda ishlaganda, lekin, ko‘pincha, jamoa a‘zolari bir binoda yoki bir shaxar yoki davlatda joylashmaydi. Ular boshqa-boshqa tashkilotlarga ishlashi mumkin yoki uyda turib. Ijtimoiy tarmoq sistemalari va telekonferensialar muloqotni osonlashtirish uchun ishlatilishi mumkin, lekin mening electron sistemalardagai tajribam shuki, hodimlar bir-birini taniganda bular eng samaralilari . siz shuning uchun loyiha boshida bir qancha hodimlar muloqot qila oladigan yuzma-yuz uchrashuvlar tashkil etishingiz kerak. Bu tog‘ridan to‘g‘ri muloqot orqali hodimlar ijtimoiy guruhlarning azosiga aylanadi va guruhning maqsadi va ustunligini qabul qiladi.

2. Izzat-ehtiromga talabni qondirish, siz hodimlarga ular tashkilot tomonidan qadrlanganini ko‘rsatishingiz kerak. Yutuqlarni ommaga tanishtirish haligacha buning oddiygina samarali usuli. Ravshanki, hodimlar mahorat va tajribasi aks etgan holda ish haqi to‘lanayotganini his etishi kerak.

3. Vanihoyat, o‘z ustida ishlashga bo‘lgan ehtiyoj, siz hodimalarga ishi uchun javobgarlik yuklashingiz, zaruriy ishlarni(imkoniyati yetmaydigan ishlarni emas) ularga tayinlashishizgiz va o‘z mahoratlarini oshira oladigan mashg‘ulot dasturlari bilan ta`minlashingiz kerak.

1.8- rasmda menejerlar duch kelishi mumkin bo‘lgan undashga oid mummolarni tushuntirganman. Bu misolda, layoqatli guruh azolari ishga bo‘lgan va guruhga bo‘lgan qiziqishni yo‘qotadi.

Muammoni o‘rganish

Alisa signal sistemalarini yaratuvchi kompaniyada ishlovchi dasturiy loyiha menejeri. Bu kompaniya o‘sib borayotgan qari va nogironlarga mustaqil yashashga yordam beruvchi ko‘makchi texnologiyalar bozoriga kirib borishni xohlaydi. Alisadan signal sistemalari sohasida yangi texnologiyalar yarata oluvchi 6 ta yaratuvchidan iborat jamoani boshqarish so‘raldi.

Alisaning ko‘makchi texnologiya loyihasi yaxshi boshlandi. Yaxshi jamoada ish aloqasi yaratildi va bunyotkor yangi g‘oyalar vujudga keldi. Jamoa kommunikatsiya uchun signal tarmog‘iga ulangan raqmli televideniyadan foydalanib to‘g‘ridan – to‘g‘ri xabar almashish sistemasini yaratishga qaror qildi. Shunga qaramay, loyihada oylar o‘tgach, Alisa apparat dizayner eksperti Dorotyni ishga kech kelishni boshlaganini sezdi, uning ishini sifati yomonlashgan va buning ustiga u jamoaning boshqa azolari bilan aloqa qilmayotgan ko‘rinadi.

Alisa muammo haqida norasmiy tarzda gaplashdi Dorotyning shaxsiy muhiti o`zgarganmi va shu uning ishiga ta‘sir qilayotgan bo‘lishi mumkinmi. Ular hechnimani bilmaydi, shuning uchun Alisa mumammoni tushunish uchun Doroty bilan gaplashishga qaror qildi.

Bir necha dastlabki rad etishlardan so‘ng, muammo ko‘rindi, Doroty ishga bo‘lgan qiziqishini yo‘qotganini tan oldi. U yarata olishni va texnik interfeyslash qobiliyatidan foydalanishni kutgandi. Shunga qaramay loyiha yo‘nalishi tanlangan, unda ozgina imkoniyat bor. Asosan, u boshqa jamoa azolari bilan C dasturchi sifatida ishlayapdi.

Ishi shubha ostidaligiga qaramay, u interfeyslash mahoratini rivojlantirmayotganidan havottda. U qurilma interfeyslashni o‘z ichiga olgan ishlarni topish bu loyihadan keyin qiyin bo‘lishidn xavotirda. Chunki u keying loyihani o‘ylashini bildirish orqali jamoani hafa qilishni xohlamaydi, u ular bilan muhakamani qisqa qilishga qaror qilgan.

1.8- rasm. Yakkalikka undash

Uning ishini sifati pastlaydi va qabul qilib bo‘lmaydigan bo‘ldi. Bu holatga tezda kurashish kerak. Agar siz muammoni hal etmasangiz, boshqa guruh azolari norozi bo‘lishadi va ular adolatsiz ish taqsimotini bajarayotgani his etadi.

Bu misolda, Alisa Dorotyning shaxsiy axhvoli muammo bo‘lishi mumkinligini bilishga urinadi. Shaxsiy qiyinchilik odatda undashga ta‘sir qiladi chunki hodimlar ishga diqqatini jamlay olmaydi. Siz ularga vaqt berishingizga va bu masalalarni yechishda qo‘llab quvvatlashingizga tog‘ri kelishi mumkin, siz ularni ish beruvchiga javobgarligi hali ham borligini ravshantirishingizga to‘g‘ri kelsa ham.

Dorotyning undash muammosi loyiha kutilmagan yo‘lda yaratilganda ko‘p uchrayduganlaridan. Bir turdagi ishni bajarishni kutgan hodimlar butunlay boshqacha ishni bajarishi bilan tugaydi. Azolar mahoratini loyihada qabul qilingan yo‘ldan boshqa yo‘lda rivojlantirishni xohlaganda bu muammo bo‘ladi. Bu ahvolda, siz jamoa a‘zosi jamoni tark etishi va boshqa joydan imkon qidirishiga qaror qilishingiz mumkin. Bu misolda, shunga qaramay, Alisa Dorotyga, uning tajribasini oshirish ijobiy kariera bosqichi ekanligiga ishonch hosil qilishga urinishga qaror qildi. U Drortyga dezayn erkinligini beradi va ayni loyihasi tugaganidan so‘ng unga ko‘proq imkoniyat beradigan dasturiy injiniring mashg‘ulot kurslarini tashkil qiladi.

Maslovning undash modeli chegaralangan darajada foydali, lekin muammo shundaki u motivatsiyaga alohida shaxsiy fikrni oladi. Bu esa hodimlarning tashkilotning, professional guruhning, bir yoki ko‘proq madaniyatning bir qismi degan faktni ishonchli hisobini olmaydi. Bu shunchaki oddiy ijtimoiy ehtiyojdan qoniqish so‘rovi emas—hodimlar umumiy maqsadlarga erishish uchun guruhga yordam bersish orqali ham undalishi mumkin.

Jips bo‘lgan guruhning azosi bo‘lish ham ko‘p hodimlar uchun kuchli undalma. Qoniqarli ishli hodimlar ko‘pincha ishga borishni yoqtiradi chunki ular birga ishlaydiganlari tomonidan undaladi va ular qilgan ishni qiladi. Shuning uchun, shaxsiy undash haqida o‘ylaganda ham tashkilotning maqsadlariga

yetishish qanday qilib to‘liq guruh undalishi haqida o‘lashingizga to‘g‘ri keladi. Guruhni boshqarish masalalarini keyingi bo‘limda muhokama qilaman.

Shaxsning turi ham undashga ta‘sir qiladi. Bass va Dunteman (1963) professionallarni ikki guruhga bo‘lgan:

1. Ish yo‘naltiradigan hodimlar, bajarayotgan ishidan undaladiganlar. Dasturiy injiniringda, bular dastur yaratishdagi intellektual baxs orqali undaladiganlar.

2. Shaxsiy yo‘naltiriladigan hodimlar, asosan shaxsiy muvofaqqiyat va tanilish orqali undaladigan hodimlar. Ular dastur yaratishga o‘zlarini maqsadiga erishish uchun qiziqadi. Bu bunday hudbin va faqat o‘zini manfaatlarini o‘ylaydi degani emas. Aniqrog‘i, ularda uzoq muddatli maqsadlar bor, karierani rivojlantirish kabi, ularni undaydigan va ular maqsadga erishishga yordam uchun ishda omadli bo‘lishni xohlaydilar.

3. Muloqot yo‘naltiradigan hodimlar, aralashish va hamkasblarning harakatidan undaladigan hodimlar. Dastur yaratish markazlashganda, muloqot yo‘naltiradigan hodimlar ko‘proq jalb qilinmoqda.

Muloqot yo‘naltiradigan shaxslar odatda guruhning bir qismi sifatida ishlashni yoqtiradi, ish yo‘naltiradigan hodimlar va shaxsiy yo‘naltiriladigan hodimlar individual faoliyat ko‘rsatishni afzal ko‘radigan joyda. Ular samarali aloqachilar. Guruxda turli xil shaxs turlarini aralalashishini 1.10 figurada muhokama qilaman.

Xar bir shaxsning undovi har bir sinfning elementidan yaraladi, lekin undashni bir turi har doim ustun. Shunga qaramay, shaxslar o‘zgaradi. Masalan, to‘g‘ri taqdirlanmayotganini his etgan texnik hodimlar shaxsiy yo‘naltiriladigan hodimga aylanishi mumkin va ular shaxsiy qiziqishlarini texnik qiziqishlardan ustun qo‘yadi. Agar guruh yaxshi ishlasa, shaxsiy yo‘naltiriladigan hodimlar muloqot yo‘naltiradigan hodimga aylanishi mumkin.

1. 3. Jamoaviy ishlash

Ko‘p professional dasturlar ikkitadan bir necha yuzgacha bo‘lgan o‘lchamida farqiluvchi guruhlar tomonidan yaratilgan. Katta guruhdagi barcha bir muammo ustida ishlashi mumkin emasligi ravshan bo‘lganidek, katta jamoalar bir qancha guruhga ajralishadi. Har bir guruh sistemaning bir qismini bajarish uchun javobgar. Oddiy qoida bo‘lganidek, dasturiy injiniring guruhlarida 10 tadan ortiq azo bo‘lmaydi. Hamma boshqa hammani taniydi va butun guruh loyihani va tuzilayotgan dasturni muhokama qilish uchun stol atrofiga uchrashuvga yig‘ilishi mumkin.

Tog‘ri texnik mahorat, tajriba va shaxslarni balansiga ega bo‘lgan guruhni yig‘ish muhim menejment masalasi. Biroq, omadli jamoalar odiyroq tog‘ri balansli mahoratli shaxslarning jamlanmasi. Yaxshi guruh bir dam va unda jamoa ruhi bor. Jalb qilingan hodimlar jamoa muvofaqqiyatidan shaxsiy maqsad kabi quvonadilar.

Birdam guruhda guruh azolari guruhni guruh azolari bo‘lgan individuallardek muhim deb hisoblaydi. Yaxshi boshqarilgan va birdam guruhning azolari guruhga sodiq. Ular guruhni himoya qilishga urinadilar, borliqdek, huddi tashqi halaqitlardan. Bu guruhni mustahkamlaydi va muammolar va kutilmagan holatlar bilan kelishadigan qiladi.

Guruhlarning foydali tomonlarini birlashtirish:

1. Guruh o‘z sifat standartini rivojlantirishi mumkin chunki bu standartlar bitimlar orqali rivojlanadi, ular kuzatilayotgan bo‘lishi ehtimol tashqi standartlar guruhga yuklangandan ko‘ra.

2. Individuallar bir biridan o‘rganadilar va bir birini qo‘llab quvvatlaydilar. Guruhdagi hodimlar bir biridan o‘rganadilar. E’tiborsizlik orqali keladigan ta‘qiqlar ikki tomonlama o‘rganish qo‘llab quvvatlanganda kamaytiriladi.

3. Bilim baham ko‘riladi. Davomiylik saqlab qolinadi agar biror guruh a‘zosi ketsa. Guruhdagi boshqalar muhim topshiriqlarga egalik qila oladi va dastur haddan tashqari vayron bo‘lmasligi ta‘minlanadi.

4. Qayta omillash va davomiy rivojlanish qo‘llab quvvatlanadi. Guruh azolari jamoaviy ishlashadi yuqori sifatli natijalar keltirish va muammolarni to‘g‘rilash uchun, aslida dizayn yoki dasturni yaratgan shaxslarga qaramasdan.

Case Study: Jamoa ruhi

Alice, tajribali loyiha menejeri, yaratish muhimligini tushunadigan yagona guruh, ular yangi mahsulot ishlab chiqish uchun shartnoma tuzish bilan barcha guruh mahsulot andoza oladi va dizayn jalb qilishni oila a'zolari bilan texnologik muhokama qilishadi. Bundan tashqari, Alice ularni yani guruhning boshqa a'zolarini kutib olish uchun bu oila a'zolarini rag'batlantirishni rivojlantiradi.

Alice ham guruhda har bir kishi uchun oyda bir marta birga tushlik qilish uchun vaqt ajratish. Bu tushlik barcha jamoasi a'zolari uchun bir imkoniyat, norasmiy va shaxsiy masalalarni atrofida gapirish, bir-biriga yaqindan bilish uchun. Tushlikda, Alice tashkiliy yangiliklar, siyosat, hokazolar strategiyasi haqida aytadi va guruhining u haqidagi fikrlarini biladi. Guruh a'zolari umumiy mavzuni qisqacha muhokama qiladi va keyin sarhisobi ko‘riladi, shuningdek yoshi katta keksa yosh vakillaridan ya‘ni qarindoshlaridan yangi mahsulot g'oyalarini so‘rashadi.

Har bir necha oyda Alice guruhi jamoasi uchun qaerdadir “yuz kun” tashkil qiladi va tashkilotni texnologik yangilash haqidagi harakatlarga ikki kun sarflaydi. Har bir guruh a'zosi tadbirda guruh uchun tegishli texnologiyalar asosida yangilanish tayyorlaydi va guruhga uni taqdim etadi. Bu yaxshi bir off-sayt uchrashuv bo‘lib ijtimoiy hamkorlik va dasturlarni muhokama qilish uchun mehmonxonada rejalashtirilgan vaqt ko‘p bo‘ladi.

1.9- rasm. Guruh birdamligi

Yaxshi loyiha rahbarlari har doim guruh butunligini rag'batlantirishga harakat qilish kerak. Ular guruh a'zolari uchun ijtimoiy tadbirlar tashkil mumkin va ularning oila a'zolari, bir butunlikni tashkil qilish uchun harakat qiladi, guruh nomini va shu guruh shaxsini tashkil etish orqali guruh identifikatsiya tuyg'usini va

o‘z o‘rnini egallashlari, guruh bo‘lib turli tadbir va sport musobaqalarida qatnashishlari mumkin.

Guruhni jipslashtirishga targ'ib etishning eng samarali yo‘llaridan biri shu guruh ichida bo‘lishdir. Agar mas'uliyatli va ishonchli bo‘lish uchun guruh a'zolari bilan muomila qilish kerak, degan ma'noni anglatadi va axborotlardan erkin foydalanish mumkin bo‘lgan yo‘llarni tashkil qilish kerak bo‘ladi. Bazida boshqaruvchilar axborotlardan foydalanish uchun barcha hodimlarga ruhsat bera olmaydiganday bo‘lishadi. Bu doimo bir muhitini yaratadi ya‘ni ishonchsizlik muhitini. Oddiy axborot almashishdagi qulaylik hodimlarni qadrli ekanini va ularni guruh bir qismi ekanligini sezdirishning bir yo‘lidir.

Masalan siz 1.9- rasmda bu misolni ko‘rishingiz mumkin. Alice mamnun guruhning boshqa a'zolari uni muntazam norasmiy uchrashuvlar aytadi. U mahsulotni rivojlantirishning ertangi bosqichlari uchun o‘z tajribalaridan olingan yangi g‘oyalari bilan kelib ularga nutq so‘zlaydi. Qisqacha qilib aytganda hodimlarning jipslashtirishning oddiy yo‘llaridan biri hodimlar yangi texnologiyalar tajribalarini almashinish orqali birgalikda hordiq chiqarishdir.

Guruhning samaradorligi ma‘lum miqdorda yaratilayotgan loyiha va tashkilotning hususiyatiga bog‘liq. Agar hodimlarni tashkilotda hotirjam ishlashlari va doimiy bandlikning kafolati ta‘minlanmasa, guruh azolari etiborini dasturiy ta‘minotni yaratishga qaratishlari qiyin bo‘ladi. Shu bilan birga, qo‘shimcha ravishda loyihalashtirish va tashkiliy masalalar operatsiyasi buyrug‘iga ta'sir qiluvchi uch asosiy omillar mavjud:

1. Guruhdagi hodimlar Siz dasturiy ta'minot kabi loyiha guruhdagi hodimlarni to‘g‘ri yo‘naltirishingiz kerak, bundan tashqari mijozlar bilan muzokaralar qilinib turli xil faoliyalarni rivojlantirishni o‘z ichiga oladi.

2. Tashkilot guruhi, Guruh shunday tashkillanishi kerakki, hodimlar loyihaga o‘z qobiliyatiga yarasha hissa qo‘shishi kerak va kutilgan natija olinishi kerak.

3. Texnik va boshqaruv aloqa Guruh a'zolari, dasturiy ta'minot jamoasi va loyihaga oid boshqa shaxslar o‘rtasidagi yaxshi aloqani o‘rnatish katta ahamiyatga ega.

Boshqaruvning boshqa masalalari kabi, yahshi jamoani olish loyiha muvaffaqiyatli yakunlanishini kafolatlamaydi. Boshqa ishlar kutilmagan tomonga o‘zgarishi mumkin, shu qatorda biznes va biznes muhitdagi o‘zgarishlar. Lekin siz guruh tarkibi, tashkillanishi va aloqalariga e’tibor qaratmasangiz, siz kelgusida qator qiyinchiliklarga uchraysiz.

1.3.1. Guruh a‘zolarini tanlash

Guruh boshqaruvchisi yoki liderning vazifasi jipslashgan guruhni yaratish va ular birgalikda samarali ishlashi uchun ishni to‘g‘ri tashkillashdir. Bunda shaxslarni va ularni tehnik ko‘nikmalarini to‘g‘ri moslashtirishni nazarda tutadi. Ba‘zida tashkilotdan tashqari shaxslarni ishga olishadi; lekin ko‘pincha dasturiy ta‘minot yaratish uchun boshqa loyihalarda ishlash tajribasi bor hodimlar qoyiladi. Lekin boshqaruvchilar jamoani tanlashda to‘liq ozod bo‘lmaydi. Ular ko‘pincha tashkilotda bor hodimlarni ishlatishga majbur, ular bu vazifani bajarishga to‘liq mos kelmasa ham.

1.3.1.bo‘limida aytganimdek, dasturiy ta‘minot yaratuvchilarning ko‘pchiligi birinchi navbatda o‘z ishi bilan undaladi. Shuning uchun ko‘pincha dasturiy ta‘minot yaratish guruhlari texnik muammolar qanday echilishi kerakligi haqida o‘z g‘oyaga ega bo;lgan odamlardan iborat bo‘ladi. Buning natijasida interfeys standartlari buzilganligi haqida habar beriladi, tizimlar bor qodi bo;yicha o‘zgartiriladi, keraksiz titizim bezaklari yaratiladi va h.

Qo‘shimcha hodimlari bor guruh faqat tanlangan texnik xodimlari bor guruhdan yahshiroq ishlaydi. O‘z ishiga qiziqqan hodim texnik tomondan ham tayyor bo‘ladi. Vazifa bajarilishi o‘ziga yahshiligini bilgan hodim vazifaning bajarilishini tezlashtiradi. Hamkorlikda ishlashga yo‘naltirilgan hodimlar guruhda ishlashni yengillashtiradi. Men o‘ylaymanki guruh bo‘lib ishlashga mo‘ljallangan

hodimlar, hamkorlikda ishlash salohiyati yuqori bo‘lgan hodimlar bo‘lishi kerak. Ular guruhda kelib chiqishi mumkin bo‘lgan kelishmovchiliklarni oldini olish orqali guruhga katta foyda keltirishadi.

Men 1.10- rasmda Alice boshchiligidagi qo‘shimcha hodimlari bor guruhni ko‘rsatganman. Bu guruhda hodimlar qobiliyatlariga ko‘ra vazifa taqsimlangan va ular hamkorlikda ishlashga yo‘naltirilgan, Lekin men 1.8 rasmda ko‘rsatganimdek, Dorothy muammo keltirib chiqdi, chunki uga o‘zi hohlagan vazifa yuklatilmagan. Fred muammoli sohada yarim stafka ishlashi ham muammo tug‘rishi mumkin. U ko‘proq texnik muammolarni yechishga qodir bo‘lganligi uchun guruhdoshlari bilan yaxshi aloqa qila olmaydi. U o‘zini guruhning bir qismi deb hisoblamaganligi uchun loyiha mazmunini yahshi tushunmaydi.

Tasodiflar tahlili:Guruh tashkillanishi

Alice tizimni rivojlantirish guruhini tashkil qilish uchun, qo‘shimcha qobiliyatga ega bo‘lgan hodimlarni tanlash kerakligi muhimligini biladi. Guruh a‘zolarini tanlashda u so‘rov o‘tkazgan, ularni maqsadga, o‘ziga, yoki hamkorlikda ishlashda yo‘naltirish uchin. U o‘zini-o‘ziga yo‘naltirilganligini his qildi, chunki u hisoblardiki loyiha faqat boshqaruvchi e’tibori ostida rivojlanishi mumkin. Guruhni to‘ldirish uchun u hamkorlikka yo‘naltirilgan ikkita hodim va masalaga yo‘naltirilgan hodimni topishi kerak edi. Uning yakuniy bahosi:

Alice—o‘ziga yo‘naltirilgan

Brian—masalaga yo‘naltirilgan

Bob— masalaga yo‘naltirilgan

Carol—hamkorlikka yo‘naltirilgan

Dorothy— o‘ziga yo‘naltirilgan

Ed— hamkorlikka yo‘naltirilgan

Fred— masalaga yo‘naltirilgan

1.10- rasm. Guruh tashkillanishi

Ba‘zan qo‘shimcha qobiliyatli guruhni tashkillash imkoni bo‘lmaydi. Agar shunday bo‘lsa guruhning tashkilotchisi guruhni shunday tashkillash kerakki,

shaxsiy manfaat guruh manfaatidan ustun bo‘lmasiligi kerak. Bu boshqaruvni bajarish oson bo‘lishi uchun guruh a‘zolari loyihaning barcha bosqichlarida ishtirok etishlari kerak. Shaxsiy tashabbuskorlik guruh a‘zolariga loyihada ularning hissasi haqida tushuntirish berilganda foyda keltirishi mumkin.

Masalan aytish mumkun dasturiy muxandis loyixada kodni yaratish va loyixaga ijobiy o‘zgartirishlar kiritish uchun kerak.Agar u bu o‘zgartirishlar tizimni boshqa qismiga tasir ko‘rsatishi xaqida tasavvurga ega bo‘lmasa , u zarar keltirishi mumkin.Gurux azolari loviga boshidan qatnashsa loyixa qarorlari nima uchun qabul qilinganligini tushunishadi.Ular shu qarorlarga mos o‘zgartirishlar kiritish mumkun.

1.3.2. Guruhning tashkillanishi

Guruh tashkillanishi usuli bu guruh tomonidan qabul qilingan qarorlarga, axborot alamashinish usullarga hamda rivijlanish guruhi va loyihaga qiziqish bildirayotgan odamlar o‘rtasidagi munosabatlarga ta‘sir qilishi mumkin. Loyiha boshqaruvchilari uchun quyidagi tashkiliy savollar muhim bo‘ladi:

1. Loyiha boshqaruvchisi guruhning texnik lideri bo‘lishi shartmi? Arxitektura yoki tizimning texnik lideri dasturiy ta‘minotni ishlab chiqish vatida qabul qilingan kritik, texnik qarorlar uchun javobgare bo‘ladi.

Bazi hollarda loyiha boshqaruvchilari bu vazifani bajarish uchun ko‘nikma va tajribaga ega bo‘ladi. Lekin katta loyihalar uchun texnik liderlikni bo‘yniga olish uchun loyiha arxitekktori etib katta muhandis tayinlanishi maqsadga muofiqdur.

2. Kritik texnik qarorlarni qabul qilishga kimlar tortiladi va ular qanday qabul qilinadi? Qororlar tizimli arxitektor loyiha boshqaruvchisi tomonidan qabul qilinadimi yoki guruh azolarining kengroq diapazonida muhokama qilinadimi?

3. Tashqaridagi qiziqayotgan shaxslar va korhonaning raxbaryati bilan hamkorlik qanday yo‘lga qo‘yiladi? Ko‘p hollarda bu hamkorlik uchun loyiha boshqaruvchisi javobgar bo‘ladi agar tizimli arxitektor bo‘lsa u yordam beradi.

Lekin alternative tashkiliy madelda shunday maxsus o‘rin bo‘lish kerakki unga tashqi aloqalarni bajarishga ko‘nikmasi bo‘lgan odam qo‘yilishi kerak.

4. Bir joyda joylashmagan odamlarni guruhga qanday birlashtirish mumkin? Hozirda yaratilayotgan guruhlarda har hil tashkilotlardan va uyda ishlaydigan odamlar birlashtirilyapti. Bu guruh qarorlarini qabul qilish jarayonlarida etiborga olinishi kerak.

5. Guruhda bilimlar almashinuvini qanday amalga oshirish mumkin? Guruhni tashkillashda axborot tarqatilish usullariga etibor qaratilishi kerak. Axborotni katta hajimda uzatilishi adamlarni charchatadi va bajarayotgan vazifasidan chalg‘iydi.

Oz miqdordagi dasturiy guruh odatda norasmiy tarizda tashkil etiladi. Guruh lideri guruhning boshqa azolari bilan birgalikda dasturiy ta‘minotni yaratish bilan shug‘ullanadi. Norasmiy guruhda bajarilishi kerak bo‘lgan ish butinicha muhokama qilinadi va vazifalar qobilyat va tajribaga asosan bo‘linadi. Guruhning kattalari axitektura dizayni uchun javobgar bo‘lishi mumkin. Lekin batafsil ishlab chiqish va amalga oshirish – alohida vazifa yuklangan guruh azosining javobgarligidur.

Favqulotda dasturiy(FD) guruhlar (Bek, 2000) har doim norasmiy guruh bo‘ladi. FD entuziyasi taqid qilishadiki rasmiy tuzilmada axborot almashuvi taqiqlangan.

FD da boshqaruv qarorlari deb hisoblangan ko‘plab qarorlar (grafik haqida qarorlar kabi) guruh azolariga o‘tkazilgan. Dasturchilar kodni rivojlanishida juft-juft bo‘lib ishlaydilar va tadbiq etilgan dasturlar uchun birgalikda javobgar bo‘ladilar.

Norasmiy guruhlar muvafaqqiyatli bo‘lishi mumkin, ayniqsa guruh azolarining ko‘pchiligi sinovdan o‘tgan va qobilyatli bo‘lsa. Bunday guruh qarorlarni umumiy kelishuv bo‘yicha qabul qiladi, natijada birgalikha ishlash yaxshilanadi. Lekin agar guruh asosan tajribasiz yoki mas’ulyatsiz qatnashchilardan iborat bo‘lsa aniq yo‘naltiruvchi yo‘qligi uchun guruh qatnashchilari orasida koordenatsiya yetishmaydi va loyiha ijobiy yakun topmaydi.

Ierarxik guruhlar – bu ierarxiya yuqorisida liderlar guruhi bor ierarxik tuzulmaga ega bo‘lgan guruhlardur. Unda guruh qatnashchilariga nisbatan ko‘proq rasmiy hokimyat berilgan va u ishni yo‘naltirishi mumkin. Aniq tashkiliy tuzulma mavjud qarorlar ierarxiya yuqorisida qabul qilinadi va ierarxiya ostidayi odamlar tomonidan bajariladi. Kommunikatsiya bu yuqoridagi shtatdan kelgan tavsiyanoma va ierarxiyada ostidagi qatlamlardan yuqori qatlamlargacha kichik “ko‘rariluvchi” kommunikatsiya mavjud.

Bu yondashuv yaxshi ishlashi mumkin qachonki yaxshi tushunilgan muammo bo‘linib ierarxiyaning har-xil qisimlarida yechimi topiladi. Bu holatalarda ierarxiya qisimlari orasida nisbatan kichik kommunikatsiya talab qilinadi. Lekin bunday holatlar dasturlashda quyidagi sabablarga ko‘ra kam uchraydi:

1. Dasturiy ta‘minotidagi o‘zgarishlar ko‘pincha tizimning bir nechta qismini o‘zgarishini talab qiladi va bunda ierarxiyaning hamma qatamlarida muhokama va muzokaralarni talab qiladi.

2. Dasturiy ta‘minot texnologiyalari shunchalar tez o‘zgarmoqdaki, ko‘pincha yoshroq xodimlar tajribali xodimlarga nisbatan ko‘proq bilimlarga ega bo‘ladi. Pastdan yuqoriga bo‘lgan kommunikatsiyada loyiha boshqaruvchisi yangi texnologiyalardan foydalanish imkoniyatlari hahiqa bilmastligi mumkin. Yosh shtatlat yaratishda ishlatilayatgan eskirgan texnalogiyalarni deb ishga bo‘lgan ishtiyoqi pasayadi.

Demakratik va ierarxik tashkillangan guruhlar guruh a‘zolarining texnik qobilyatlari o‘rtasida juda katta farq bo‘lishini tan olmaydilar. Eng yaxshi dasturchilar yomon dasturchilaraga nisbatan 25 marotaba samarali ishlashi mumkin. Eng yaxshi dasturchilarni eng yaxshi sharoitlar bilan ta‘minlab, ulardan imkoni boricha samarali faydalanish kerak. Bunday sharoitni ko‘zda tutgan dastlabki tashkiliy madel dasturchining asosiy jamoasi bo‘lgan.

Yuqori malakali dasturchilardan samarali faydalanish uchun Baker (1972) va boshqalar (Aron 1974: Brooks 1975) jamoa alohida yuqori malakali asosiy dasturchi atrofida yaratilishini taklif qilishdi. Dasturchining asosiy jamoasini

asosiy ta‘moyili – malakaliy va tajribali hodim dasturiy ta‘minotning yaratilishiga to‘liq javobgar bo‘lishi kerak. Ular turli-tuman savollarga chalg‘imastligi kerak va ularning ishi texnik va ma‘muriy tomondan yaxshi ta‘minlanishi kerak. Ular butun etiborini yaratilayotgan dasturiy ta‘minotga qaratishi kerak va vaqtlarini tashqi uchrashuvlarga sanrflamasligi kerak.

Lekin dasturchining asosiy jamoasi madelida – mening fikrimcha asosiy dasturchi va unin yordamchisi hisobiga qolganlar qaram bo‘lib qoladi. Jamoaning yetarli javobgarlik berilmagan qolgan azolari qobilyatlariga yarasha ish berilmaganini xis qilib ishtiyoqi so‘nishi mumkin. Muammo paydo bo‘lganda ularda yetarlicha axborot bo‘lmasligi munkun va qaror qabul qilishda ishtirok etish imkoniyati yo‘q. guruhni bunday tashkillashda o‘ziga yarasha havf-hatarlari mavjud va ular bu tashkillanish turi olib kelishi mumkin bo‘lgan foydadan yo‘qqa chiqarishi mumkin.

1.3.3. Guruh kommunikatsiyasi

Guruh azolari bir-biri bilan va loyihaga ta‘alluqli boshqa shaxslar bilan samarali muloqat qilishi juda muhum ahamiyatga ega. Guruh azolari o‘zlarining ishi holati, bajarilgan loyiha va avvalgi loyiha qarorlaridagi o‘zgarishlar haqidagi axborot bilan almashish kerak. ular paydo bo‘lgan muammolarni boshqa tegishli shaxslar bilan yechimini topishlari kerak va bu shaxslarga tizimda, guruh va yetkazib berish rejasidagi o‘zgarishlar haqida xabar berishlari kerak. Yaxshi kommunikatsiya guruhning hamkorlikda ishlash qobilyatini kuchaytirishga yordam beradi. Guruh azolari guruhdagi boshqa odamlarning maqsadlari, kuchli va zaif tomonlari haqida tushunchaga ega bo‘ladi.

Kommunikatsiyaga samarali va samarasiz ta‘sir etuvchi omillar:

1. Guruh o‘lchami. Guruh kattalashgan sari uning qatnashchilari uchun samarali muloqatta bo‘lish qiyinlashad. Guruh o‘lchami n bo‘lganda bir tomonlama aloqa liniyalar miqdori n*(n-1) ga teng bo‘ladi, demak guruhda 8 ta qatnashchi bo‘lsa kommunikatsiya yo‘llari 56 teng bo‘ladi. Bu degani bazi

odamlar bir-biri bilam kam muloqatta bo‘ladi. Guruhdagi qatnashchilarning maqomidagi farqlar natijasida aloqalar ko‘’pincha bir taraflama bo‘ladi. Boshqaruvchi va tajribali muhandislar tajribasi kamroq muloqatta o‘zini bilimli deb xisoblab, suxbatdan bosh tortishi yoki tanqidiy fikirlar bildirishi mumkin.

2. *Guruh tuzilmasi.* Norasmiy tuzilmali guruhlardagi odamlar rasmiy ierarxik tuzilmaga ega bo‘lgan guruhlardagi odamlarga nisbatan samarali muloqat qilishadi. Iierarxik guruhlarda aloqa ierarxiya bo‘yicha tepaga yoki pastga qarab amalga oshiriladi. Bitta qatlamdagi odamlar bir-birlari bilan gaplashmastligi ham munkun. Bir necha guruhda yaratilayotgan katta loyihada bu muammo tug‘dirishi mumkin. Har-xil osti tizimlarda ishlayotgan odamlar boshqaruvchilar orqali muloqatta bo‘lishsa bu yerda ushlanish ba tushunmovchiliklar paydo bo‘lishi havfi katta.

3. Guruh tarkibi. Birxil shaxs turidagi odamlar (1.2 bo‘limda muhokama qilingan) to‘qanash kelishi mumkin va natijada aloqa uzilishi mumkin. Shuningdek birxil jinslik guruhlarga nisbatan har-xil jinslik guruhlarda muloqat yaxshi bo‘ladi (Marshall va Heslin 1975). Ayollar ko‘pincha erkaklarga nisbatan kirishiliroq bo‘lishadi va o‘’zaro muloqat dispechirlari hamda guruh yordamchilari vazifasini bajarishlari mumkin.

4. *Jismoniy ish muhiti.* Ish joyini tashkillanishi muloqatni yengillashtirish yoki man etishi asosiy omildur. Ko‘proq axborot olish uchun elektron kitobni ko‘ring(http://www.SoftwareEngineering.com/Web/Management/workspace.html)

5. *Ruxsat etilgan aloqa kanallari.* Aloqaning ko‘plab har-xil shakillari mavjud – yuzma – yuz, elektron hatlar, rasmiy hujjatlar, telefon va WEB 2 texnologiyalar (ishtimoiy tarmoq va wikis). Loyiha guruhlari brogan sari tarqalgan bo‘lib jamoa azolari masofadan ishlayotgani uchun siz aloqani yengllashtirish uchun texnalogiyalar diapazonidan foydalanishingiz kerak.

Loyiha boshqaruvchilari odatda tig‘iz vaqtda ishlashadi, shuning uchun ko‘p vaqt talab qilmaydigan aloqa kanallarini ishlatishlari mumkin. Shuning uchun ular axborotni uzatish, xodimlarni to‘plash va mijozlarni toppish uchun uchrashuv va rasmiy hujjatlardan foydalanish mumkin. Bu usul loyiha boshqaruchisi nuqtaiy

nazaridan aloqa o‘rnatishga samaralik yondashuv xisoblansa ham, amalda doim ham samara keltirmaydi. Ko‘p hallarda odamlar jiddiy sabablarga ko‘ra uchrashuvlarga kelmasligi mumkin va taqdimotni ko‘rmaydilar. Katta hajmdagi hujjatlar ko‘pincha o‘qilmaydi chunki o‘quvchilar hujjatlar ahamiyatga ega ekanligini bilishmaydi. Bitta hujjatning bir nechta turlari bo‘lgandan keyin o‘quvchilar undagi o‘zgarishlarini anglashlari qiyin bo‘ladi.

Samaralik kommunikatsiyaga erishildi deb aloqa ikki tomonlama bo‘lganda va undagi odamlar muammo va axborotni muhokama qilishi ham taklif va muammolar o‘zaro mos kelishi o‘rnatilganda aytish mumkin. Bularga uchrashuvlar orqali erishish ham mumkin. Lekin ular ko‘pincha amaldor shaxslar qo‘l ostida bo‘ladi. Oldindan ogoohlantirmasdan uchrashuvlar o‘tkazish naf bermaydi. Borgansari loyiha guruhlari masofadagi qatnashchilarni o‘z ichiga olmoqda, bu ham uchrashuvlarni uyushtirishda muammo tug‘diradi.

Bu muammolarni oldini olish uchun siz axborot almashuvini ta‘minlash uchun wikis va bloglar kabi web texnologiyalardan foydalanishingiz mumkin. Wikis hamkorlikda hujjatlarni yaratish hamda muharirlash uchun imkoniyat yaratadi, bloglarda guruh azolari savol va izohlardan iborat bo‘lgan muhakamalarni olib borishlari mumkin. Wikis va bloglar joylashgan joyidan qatiy nazar loyiha qatnashchilari va unga ta‘luqli shaxslarga axborot almashuvini ta‘minlaydi. Ular axborotni boshqarishga va elektron pochta orqali amalga oshiriladigan muhokama mavzularini ajratishga yordam beradi. Shuningdek siz birorta muammoni hal etishni muhokama qilishga extiyoj sezilganda xabarlar almashuvi yoki telekonferensiyadan foydalanishingiz mumkin.

Asosiy fikirlar:

- Dasturlash loyihalari rejaga asosan va budjetdan chiqmagan holda bajarilishi uchun dasturiy ta‘minot loyihasini yaxshi boshqarish katta ahamiyatga ega.

- Dasturiy ta‘minotni boshqaruvi boshqa texnik boshqaruvdan farq qiladi. Dasturiy ta‘minot mavhum. Loyihalar yabgi yoki innovatsion bo‘lishi mumkin shuning uchun ularni boshqarishda tajribasi bo‘lmasligi mumkin. Dasturiy

ta‘minot jarayonlari ishlab chiqarishning odatiy jarayonlari kabi rivojlangan bolmasligi mumkin.

- Havf-hatar boshqaruvi hozirda loyiha boshqaruvining muhim vazifalaridan biri deb tan olingan.
- Havf-hatar boshqaruvi bosh loyihani havf-hatarlarini identifikatsiya qilish va baxolashni, havf-hatarlar paydo bo‘lish extimoli va paydo bo‘lsa loyihaga ta‘sirini aniqlashni o‘z ichiga oladi. Siz extimoli bor havf-hatarlarni oldini olishni, bartaraf etishni rejalashtirish kerak.
- Odamlarni shaxsiy rivojlanishga boshqaruvchilar va kasbdoshlar undaydi.
- Dasturiy ta‘minot yaratuvchi guruh azolari soni kam bo‘lishi va o‘zaro aloqalari kuchli bo‘lishi lozim. Guruh samaradorligiga ta‘sir etuvchi asosiy omillar – guruhdagi odamlar, guruh tashkillanishi shakli va guruh azolari o‘rtasidagi aloqa.
- Guruh ichidagi aloqalar guruh qatnashchilarining maqomi, guruh o‘lchami, guruhning gender tarkibi, shaxs turlari va ruxsat etilgan aloqa kanallari kabi omillar ta‘sirida o‘rnatiladi.

Nazorat savollari:

1. Tushuntiring nima uchun dasturiy ta‘minot tizimlari mavhumligi dasturiy ta‘minot loyihasini boshqarishda o‘ziga xos muammolar tug‘diradi.

2. Tushuntiring nima uchun eng yaxshi dasturchilar dasturiy ta‘minotni eng yaxshi boshqaruvchilari bo‘la olmaydi. Javob berishda 1.1 bo‘limdan olishingiz mumkin.

3. Adabiyotda berilgan holatlarni ishlatib muvafaqqiyatsizlikka uchragan dasturiy loyihalarda sodir bo‘lgan boshqaruv qiyinchiliklari va hatolarni aytib o‘ting (Fred Brooksning “Odamning Hayoliy Oyi”dan boshlashingizni taklif qilaman).

4. 1.1 rasmda berilgan havf-hatarlarga qo‘shimcha qilib dasturiy ta‘minot loyihalarida paydo bo‘lishi mumkin bo‘lgan 6 ta boshqa havf-hatarni aniqlang.

5. Bajaruvchi tizimni yakunlashda doimiy narx belgilangan shartnomalari loyiha havf-hatarini mijozdan bajaruvchiga o‘tkazish uchun ishlatilishi mumkin. Agar birorta ish notog‘ri ketsa bajaruvchi to‘lishi kerak bo‘ladi. Bunday

shartnomalardan foydalanish maxsulot havf-hatari paydo bo‘lish extimolini ko‘paytirishiga qanday ta;sir qilishini taklif eting.

6. Tushuntiring loyiha rivojlanishi va texnik yechimlar haqida guruhning hamma azolarini habardor qilish guruhning hamkorlikda ishlash qobilyatini nima uchun yaxshilashi mumkin.

7. Sizning fikringizcha ko‘p boshqaruv qarorlari guruh azolari ixtiyoriga berilgan FD jamoalarida qanday muammolar paydo bo‘lishi mumkin?

8. Loyiha jamoasida aloqalar muhimligini ko‘rsatib beruvchi misolni keltiring. Tasvvur qiling jamoaning ba‘zi azolari masofadan ishlaydi va oldindan ogohlantirishsiz bir jamoani bir joyga yog‘ish qiyin.

9. Sizning boshqaruvchingiz dasturiy ta‘minotni rejadan oldin topshirishni so‘rayapti, bunda sizning loyiha guruhingiz qo‘shincha ish vaqti uchun haq olmaydi. Jamoaning hamma azosida yosh bollar mavjud. Siz boshqaruvchingizni bu talabini qabul qilishingiz kerakmi yoki jamoangizni vaqtini oilasiga emas ishga bag‘ishlashiga chaqirishingiz kerak. shuni muhokama qiling. Sizning qaroringizga qaysi omillar ta‘sir qiladi?

10. Dasturchi sifatida sizga loyiha boshqaruvchisi vazifasini ko‘tarilishi sifatida taklif qilishdi, lekin siz tashkilotchi sifatida emas texnik tomondan samaraliroq xissangizni qo‘shishingizni xis qilyapsiz. Taklifni qabul qilishingiz kerakmi? Shuni muhokama qiling.

2. LOYIHANI REJALASHTIRISH

Loyihani rejalashtirish – dasturiy ta‘minot loyiha bosharuvchisining asosiy vazifalaridan biridur. Boshqaruvchi sifatida siz ishni qisimlarga bo‘lishungiz va ularni loyihalashni jamoa azolariga topshirishingiz kerak, paydo bo‘lishi mumkin bo‘lgan muammolarni oldini olishingiz va bartaraf etishga tayyor turishingiz kerak. Loyiha boshida yartilgan loyiha rejasi ish qanday bajarilishi loyiha guruhi va mijozlari bilan muhokama qilish uchun va loyiha qanday rivojlanayotganini baxolash uchun ishlatiladi.

Loyihani rejalashtirishni hayotiy sikli 3 bosqichdan iborat:

1. Taklif bosqichi. Unda siz dasturiy ta‘minot tizimini rivojlantirish yoki yaratish shartnomasi uchun narx taklif qilasiz. Bu bosqichda sizga ishni yakunlash uchun yetrli manbaga ega ekanligingiz va mijozga aytiladigan narxni chamalashingiz uchun reja kerak.

2. Loyihani ishga tushurish bosqichi. Bunda siz loyihani kim ishlashini loyiha qanaqa qismlarga bo‘linishini, manbalar sizning korhonangizda qanday bo‘linishini va boshqalarni rejalashtirasiz. Taklif bosqichiga nisbatan bu bosqichda siz ko‘proq axborotga ega bo‘lasiz va dastlabki baxolashni mukammalshtirishingiz mukin.

3. Liyoha davomida davriy, bunda siz rejangizni olingan tajriba va ish ketishi haqidagi axborot asosida o‘zgartirasiz. Yaratilayotgan tizim va jamoa qobilyati haida ko‘proq ma‘lumotga ega bo‘lasiz. Bu axborot sizga ishni bajarishga qancha vaqt sariflashga aniq baxo beraolasiz. Bundan tashqari daasturiy maxsulotga talablat o‘zgarishi mumkin va odatda buning natijasida ish taqsimoti o‘zgarishi va rejadagi muddat o‘zgarishi mukin. Odatiy ish loyihalari uchun loyihani ishga turushish bosqichida yaratilgan reja o‘zgartiriladi. Lekin tezkor yondashuv ishlatilganda reja qisqaroq atama bo‘lib doimiy ravshda dasturiy ta‘minot rivojlanish mobaynida o‘zgartiriladi. Tezkor rejalashtirish 3.4 bo‘limida muhokama qilingan.

Taklif bosqichidagi rejalashtirish ilojsiz taxminiy bo‘ladi, chunki yaratilayotgan dasturiy ta‘minotga bo‘lgan talablarnng to‘liq kamplektiga ega emassiz. Bu paytda siz dasturiy ta‘minotning talab qilingan funksionalligining yuqori darajasi tasnifiga asoslangan takliflarga javob berishingiz kerak. Ko‘pincha reja taklifning muhum qismidur, shuning uchun siz ishni bajarishni extimollik rejasini tushushingiz kerak. Siz shartnomani tuzganingizdan keyin taklif kiritilgandan so‘ng amalgam oshirilgan o‘zgarishlarni xisobga olgan holda loyihani takroran rejalashtirishingiz kerak bo‘ladi.

Shartnoma uchun narxni belgilayotganda siz mijozga taklif qiladigan dasturiy ta‘minotni yaratish narxini haqida bir qarorga kelishingiz kerak. Bu narxni xisoblash uchun siz loyiha ishini yakunlashga ketadigan sanf-harajatlaringizni baxolashingiz kerak. hat bir faoliyatga qancha manba ketishini baxolab bundan butun faoliyatning umumiy narxi kelib chiqadi. dasturiy ta‘minotni yaratishni narxini aniq taxmin qilish maqsadida dasturiy ta‘minotga bo‘lgan sarflarni doimiy obyektiv tarzda hisoblashingiz kerak.

Sarf harajatlaringizni maqbul baxosini olganingizda siz mijozga aytiladigan narxni hisoblashingiz mumkin. Keyingi bo‘limda ko‘rish mumkin bo‘ladi dastiruy ta‘minot loyihasini baholashga ko‘p omillar ta‘sir etadi – bu shunchaki narx+foyda emas.

Dasturiy ta‘minot yaratish loyihasining sarf harajatlarining xisoblashda siz 3 ta parametirdan foydalanishingiz kerak:

- Ish kuchi sarflari (dasturiy ta‘minot yaratruvchilari va boshqashqaruvchilari oyligi);
- Apparat va dasturiy ta‘minot sarflari, xizmat ko‘rsatish ham shuni ichida;
- Sayohat va o‘qitishga ketadigan sanflar.

Ko‘p loyihalar uchun eng katta narx – ish kuchi narxi. Siz loyiha ishini tugatishga ketadigan umumiy ish vaqtini baxolashingiz kerak (odamni oyi hisobida). Bunday baxolashni bajarishda sizda axborot kamlik qiladi shunday qilib siz eng yaxshi baxolanishni olib unga qo‘shimcha muhim bo‘lgan oldindan bilib bo‘lmaydigan holatni qo‘shishingiz mumkin (qo‘shimcha vaqt va ish kuchi).

Tijorat tizimlari uchun siz odatda nisbatan arzon bo‘lgan apparat vositalaridan foydalanasiz. Lekin siz maxsulotni va dasturiy ta‘minot platformasini litsenziyalashtirishingiz kerak bo‘lganda dasturiy ta‘minot sarflari katta bo‘lishi munkin. Loyiha bir nechta joyda yaratilayotganda ularni hammasiga borib kelish talab qilinadi. Bu sayohatga ketadigan sarflar kam bo‘lsa ham, unga ketadigan vaqt ko‘pincha behuda ketadi va loyihaning ish vaqtini ko‘paytiradi. Yangi texnologiyalar yordamida uyushtirilgan uchrashuvlar ham masofadan hamkorlikni ta‘minlovcha dasturiy ta‘minot bunday sayohatlar miqdorini kamaytirishi mumkin. Tejalgan vaqt unumdor loyiha ishiga saflanishi mumkin.

Tizimni yaratishga shartnomasi tuzilgandan so‘ng loyihani ishga tushurish rejasini yaratish uchun loyiha grafaik rejasini mukammallashtirish kerak(asosiy). Bu bosqishda siz bu tizimga bo‘lgan talablar haqida ko‘proq bilishingiz shatr. Lekin siz yaratishda tezkor yondashuvdan foydalanayotgan bo‘lsangiz siz talablarning to‘liq spetsifikatsiyasiga ega bo‘lmasligingiz ham mumkin. Bu bosqishda sizning maqsadingiz hodimlarni ishlatish va byudjetni tuzish to‘g‘risida qaror qabul qilish uchun foyadalanish mumkin bo‘lgan loyiha rejasini yaratishdur. Siz tashkilot manbayingizni taqsimbal ko‘rishimgiz kerak va yangi hodimlarni ishga olish haqida qaror qilishingiz lozim.

Rejada loyihani nazorat qilish mexanizimlari ham belgilanishi lozim. Siz loyiha rivojlanishini kuzatishingiz, rejadagi va amaldagio‘zgarish hamda sarflarni taqqoslashingiz lozim. Ko‘p tashkilotlarda nazoratning rasmiy prodseduralari mavjud lekin boshqaruvchi loyiha xodimlari bilan norasmiy muhokamalarni amalga oshirib doim rivojlanishning aniq holatini tasniflab berishga tayyor turishi lozim.

Norasmiy nazorat qiyinchiliklar paydo bo‘lganda muammolarni oldini olishga yordam beradi. Masalan loyiha hodimlari bilan har kung muhokalamalar natijasida dasturiy ta‘minotning hatosini aniqlashdagi maxsus muammosini ko‘rsatib berishi mumkin. Bunda boshqaruvchi muammoni hal qilish uchun expetrni yonlashi mumkin yoki uni dastur doirasida hal qilishi mumkin.

Loyiha rejasi yaratish jarayoni mobayni jarayonida o‘zgaradi. Rejalashtirish hodimlarga maqsadlari va ularga erishish yo‘llarini tushuntirish uchun loyiha rejasi foydali hujjat bo‘lishi uchun kerak bo‘ladi. Shuning uchun dasturiy ta‘minot rivojlanishi mobaynida grafik, smeta va havf-hatarlar qaytadan ko‘rilishi lozim.

Agar tezkor usul ishlatilayotgan bo‘lsa loyihani ishga tushirish rejasi doimo kerak bo‘ladi, chunki ishlatilayotgan yondashuvdan qat’iy nazar tashkilot loyihaga sariflanadigan manbalarni rejalashtirishi kerak bo‘ladi. Lekin bu reja batafsil bo‘lishi shart emas va unda bajariladiga ishlar ro‘yhati va loyiha grafigi haqida chegaralangan axborot bo‘lishi lozim. Yaratish vaqtida rejalashtirish jarayoniga butun jamoa jalb qilinadi , norasmiy loyiha rejasi va ishchi kuchi baxolanishi dasturiy ta‘minotni har bir qismi uchun tuziladi.

2.1. Dasturiy maxsulotga ketadigan sarf harajatlar

Tamoyilga ko‘ra, mijoz uchun dasturiy mahsulot narxi yaratuvchi uchun foydaga qo‘shimcha rivojlanish narxi hisoblanadi. Amaliyotda, shunday qilib, mijoz uchun belgilangan narx bilan loyiha narxi o‘rtasidagi bog‘liqlik u qadar oson emas. Narxni hisoblayotganingizda, siz tashkiliy, iqtisodiy, siyosiy va tadbirkorlik mulohazalarini hisobga olishingiz kerak.(2.1- rasmda ko‘rsatilgandek). Siz tashkiliy aloqalar, loyiha bilan birlashgan tavakkalchiliklar va foydalaniladigan shartnoma turlari haqida o‘ylashingiz kerak . Bular narxni yuqoriga yoki pastga o‘zgartirib yuborishi mumkin. Tashkiliy mulohazalar sababli, loyiha narxini qarorlashtirish marketing, savdo xodimi, yuqori menejment va loyiha menejerlarini faolligini o‘z ichiga oluvchi guruh bo‘lishi kerak.

Loyihani narxlash nashrlarining ba‘zilarini tasvirlash uchun qiyidagi ssenariy deb tushuniladi:

Kichik dasturlash kompaniyasi, PharmaSoftda 10 ta dasturlash injenerlari ishlaydi. Hozirgina katta loyiha tugallandi, ammo faqat 5 ta malakali xodimni talab qiluvchi shartnoma bor. Shunday qilib, 2 yil mobaynida natijaning 30 shaxsiy yillarni talab qiladigan yirik farmaseftik kompaniya bilan juda katta

shartnoma narxlanyapti. Loyiha kamida 12 oyda boshlanmaydi, Agar kelishilsa, u kompaniyaning moliyasiga aylantiriladi.

PharmaSoft 6 ta odam va 10 oyda tugatish kerak bo‘lgan loyihani narxlash imkoniyatiga ega. Narx (loyihaning boshidan boshlab) $1.2 million dollarga baholanadi. Shunday qilib, raqobatbardosh o‘rinni yaxshilash uchun PharmaSoft $0.8 million dollarni xaridor uchun narxlashga qaror qiladi. Buning ma‘nosi shundayki,garchand shartnomada pul yo‘qotilsada, bu narsa ko‘proq foydali kelajak loyihalari uchun mutaxassis xodimni ushlab qola oladi.

Faktorlar	Ta‘riflar
Bozor imkoniyati	Rivojlanayotgan tashkilot eng quyi narxni belgilashi mumkin, chunki u dastur bozorining yangi segmentiga ko‘chishni istaydi. Bir loyihadagi eng past foydani qabul qilish keyinchalik yaxshiroq foyda qilish imkoniyatini berishi mumkin.Bu tajriba yangi mahsulotni yuksaltirish uchun yordam beradi.
Ishonchsiz narxni baholash	Agar biror tashkilot narx qo‘yishda ishonchsiz bo‘lsa, bu normadagi foyda ustida va tasodifiy holda narxni oshirishi mumkin.
Shartnoma asosidagi muddat	Xaridor yaratuvchiga asos kodning egaligiga ruxsat berishi va undan boshqa loyihalarda qayta foydalanishi mumki. Narx keyinchalik dasturning asos kodi xaridorning qo‘liga o‘tishiga qaraganda ozroq bo‘lishi mumkin.
O‘zgaruvchan talablar	Agar talablar o‘zgarishi mumkin bo‘lsa, tashkilot shartnomada yutish uchun narxni pasaytirishi mumkin. Shartnoma mukofotlangandan so‘ng, yuqori narx talabga qarab, narxlanadi.
Moliyaviy yordam	Yaratuvchilar moliyaviy qiyinchiliklarda shartnomaga erishish uchun narxni pasaytirishadi. Bu biznesni tark

etishga qaraganda yoki normadagi foydadan kichikrog‘ini bajarish uchun yaxshi!

2.1- rasm. Dasturiy ma‘xsulot sarf-xarajatlariga ta‘sir ko‘rsatuvchi omillar

Loyiha narxi xaridor uchun belgilangan narx bilan erkin bog‘langan bo‘ladi. “Pricing to win (Yutish uchun narxlash)” – bu oddiy qo‘llaniladigan strategiya. Buning ma‘nosi, agar biror kompaniyada xaridor kutadigan narx to‘g‘risida fikr bo‘lsa va u mijozning kutgan narxiga asoslangan shartnoma uchun narx belgilaydi. Bu etikaga zid va ishga aloqador bo‘lmagandek tuyilishi mumkin, ammo bu tizim ta‘minlovchi va mijoz o‘rtasidagi afzalliklariga ega.

Loyiha narxi rejadagi taklifga asoslanib kelishiladi. Muzokaralar mijoz va doimiy xaridor o‘rtasida o‘ziga xos loyihani tasdiqlash uchun o‘tkaziladi. Bu o‘ziga xoslik kelishilgan narx orqali chegaralanadi. Sotuvchi va oluvchi maqbul tizimning qandayligiga qarab kelishishi shart. Ko‘plab loyihalardagi o‘rnatilgan faktorlar loyiha talabi emas,ammo narx. Talab narx ortib ketmaganda o‘zgarishi mumkin.

Misol uchun biror kompaniya (OilSoft) biror yog‘ kompaniyasi uchun yoqilg‘i tarqatish tizimini yaratish uchun shartnoma uchun narx belgilayotgan bo‘lib, Jadvallar xizmat stansiyalari uchun yoqilg‘i yetkazib beradi. Bu yerda tizim uchun talab xujjatlari keltirilmagan, shuning uchun Oilsoft yog‘ kompaniyasi byudjetida raqobatbardosh bo‘lishi uchun narxni $900.000 dollar deb baholaydi. Shartnoma kelishilgandan so‘ng, OilSoft tizim talablari bilan bitim tuzadi, shuning uchun asosiy funksiyalar ta‘minlanadi. Keyin ular boshqa talablar uchun narx baholashadi. Yog‘ kompaniyasi bu yerda hech narsa yo‘qotmaydi, chunki shartnoma mukofotlanadi. Qo‘shimcha talablar kelgusi byudjetdan fondga qo‘yiladi, shuning uchun yog‘ kompaniyasining byudjeti yuqori boshlang‘ich dastur narxi orqali buziladi.

2.2. Rejaga asoslangan rivojlanish

Rejaga asoslangan rivojlanish –bu jarayon rejalashtirilgan yerda dasturiy injiniringga murojaat qilish Bir loyiha rejasi ishni kim bajaradi, yaratish jadvali va ish mahsuloti haqidagi ish hisobotlariga koʻra yaratiladi. Menejerlar loyiha qarorlarini tayyorlash va jarayonni oʻlchash uchun rejadan foydalanishadi. Rejaga asoslangan rivojlanish injiniring loyihasi, menejment texnikalari va katta dasturiy loyihalarni boshqarishning “an’anaviy” yoʻliga asoslangan boʻladi. Bu harakatchan rivojlanish bilan farqlanadi, rivojlanishga taʻsir koʻrsatuvchi qarorlar jarayon davomida taʻminlanadi va keyinroq tayyorlangan boʻladi.

Rejaga asoslangan rivojlanishga qarshi asosiy argument –bu koʻp dastlabki qarorlar qaytadan koʻrib chiqilishi kerak ,dastur rivojlantirilgan va foydalanilgan muhitga oʻzgarishlar sababli. Shu kabi qarorlarni taʻminlash oʻylab qilinadi chunki u keraksiz qayta ishlashdan chetga chiqadi. Rejaga asoslangan rivojlanishning koʻmagidagi argument tashkiliy hisobotlarga (xodimlar qobiliyati, boshqa loyihalar va b q) ruxsat beradiga rejalashtirish va muammo va bogʻliqliklar loyiha boshlanishidan oldin kashf etiladi.

Mening fikrimcha, loyihani rejalash uchun eng yaxshi murojaat tez va rejaga asoslangan yaratishning aralashmasini oʻz ichiga oladi. Narxdagi muvozanat loyiha turiga va odamlarning qobiliyatiga bogʻliq. Benihoya yirik xavfsizlik va xavfsiz-kritik tizimlar yirik analizlarni talab qiladi va foydalanishga topshirilishidan oldin sertifikatlashtirilishi kerak.Bular koʻpincha rejaga asoslangan boʻlishi kerak. Boshqa jihatdan kichkina oʻrta kattalikdagi axborot tizimi raqobatdosh muhitning tez oʻzgarishida qoʻllaniladi. Bir necha kompaniyalar bir yaratish loyihasini oʻziga bogʻlagan yerda, rejaga asoslangan yaqinlashuv har bir ishni muvofiqlashtirish uchun qoʻllaniladi.

2.2.1. Loyiha rejasi

Rejaga asoslangan yaratish loyihasida, loyiha rejasi loyiha uchun yaroqli resurslarni oʻziga toʻplaydi. Reja dastur va loyiha uchun tavakkalchiliklarni

aniqlashi kerak. Shunga qaramay loyiha rejasining mahsus detallari loyiha va tashkilot turiga bog‘liq holda o‘zgaradi. Rejalar quyidagi qismlarni o‘z chiga oladi:

1. *Tanishtiruv*. Bu qisqacha loyiha maqsadini tasvirlaydi va loyihani boshqarishga ta‘sir ko‘rsatadigan omillarni (byudjet, vaqt va hk) ni to‘playdi.

Reja	Ta‘riflar
Sifatli reja	Loyihada ishlatiladigan sifatli protseduralar va standartlarni tasvirlaydi
Qonuniy reja	Tizimni qonuniylashtirish uchun ishlatiladigan jadval,yaqinlashuv va resurslarni tasvirlaydi.
Konfiguratsiyalangan boshqaruv rejasi	Boshqarish protseduralarini va strukturalarini tasvirlaydi
Qo‘llab- quvvatlash rejasi	Qo‘llab-quvvatlovchi talab, narx va natijani tasvirlaydi.
Xodimlarning yaratish rejasi	Loyiha guruhi a‘zolarining qanday tajriba va qobiliyatga ega ekanligini tasvirlaydi.

2.2- rasm. Loyihaning qo‘shimcha rejasi

2. *Loyihalashtirish tashkiloti*. Bu jamoa tashkillanishi, odamalar jalb qilinishi va ularning jamoadagi rollarini tavsiflaydi.

3. *Havf-hatarni taxlil qilish*. Bu bo‘lishi mumkin bo‘lgan loyiha tavakkalchiliklarini, risk ehtimolliklari tasvirlaydi, Va Riskni qisqarish strategiyasi taklif qilinadi. Riskni boshqarish 2-bo‘limda keltirilgan.

4. *Apparat va dasturiy resurslarga talablar*. Bunda Apparat va dasturiy qism yaratishni oxiriga yetkazadi.Agar apparat qism sotilishi kerak bo‘lsa, narxni baholash va ta‘minlash jadvali o‘z ichiga olinishi mumkin.

5. *Ish ro‘yhati*. Bu har bir harakat bilan birlashgan ta‘minlovchi aniqliklar va harakatlarga loyiha ro‘yhatini yig‘adi. Masofalar - loyihadagi pog‘onalar belgisi

,ya‘ni jarayon baholangan; taminovchilar – esa ish mahsulotini xaridorga yetkazib beradiganlar.

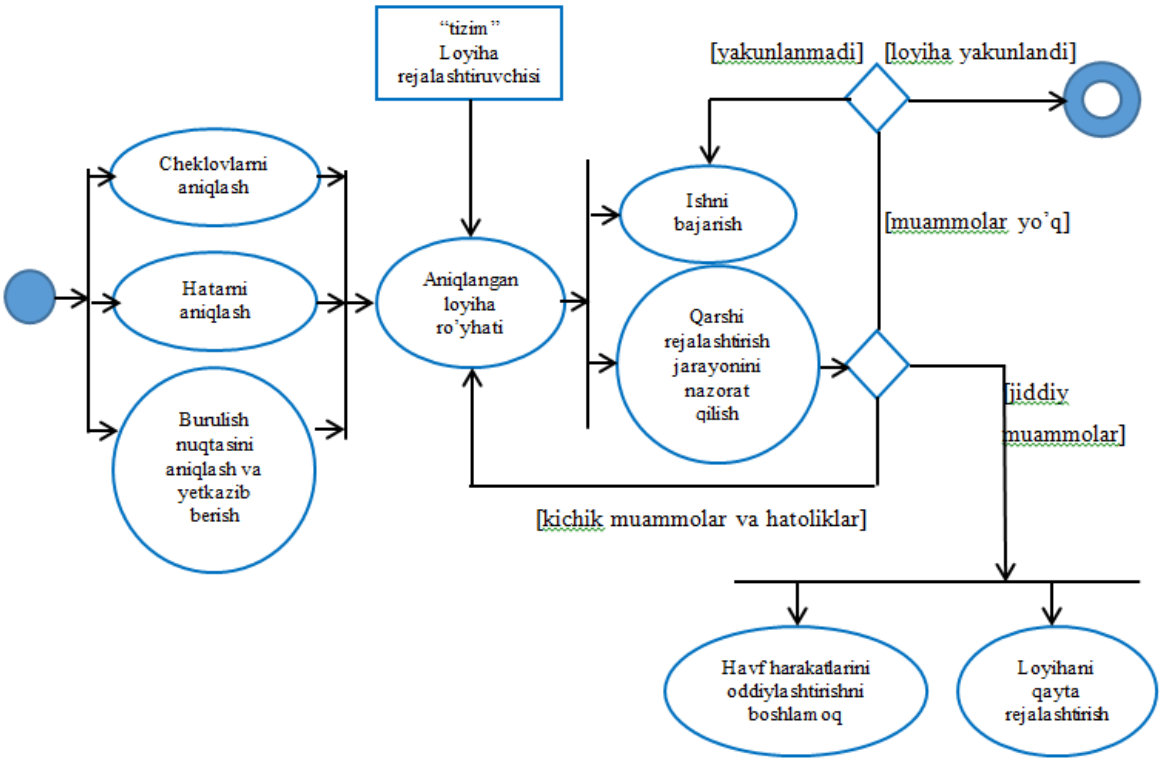
6. *Loyiha jadvali*. Bu Harakatlar o‘rtasidagi bog‘liqliklarni ko‘rsatadi, baholangan vaqt har bir masofaga erishishni talab qiladi. Bu yo‘llar keying bobda tahlil qilinadi.

7. *Monitoring va hisobot berish mexanizmlari*. Bu qachonki loyiha mehanizimi qo‘llanilganda va bular ishlab chiqarilganda , ishlab chiqarilishi kerak bo‘lgan boshqarish hisobotlarini aniqlaydi.

Loyiha rejasi loyiha jadvali va loyiha uchun riskning diqqat markazida bo‘ladi. Siz boshqa jarayon harakatlari shuningdek, testlash va konfiguratsion menejment harakatlarini ta‘minlovchi qo‘shimcha reja ishlab chiqishingiz mumkin. qo‘shimcha rejalar 2.2- rasmda ko‘rsatilgan.

2.2.2. Rejalash jarayoni

Rejalash jarayoni takrorlanuvchi jarayon bo‘lib, u loyiha bosqichi davomida boshlang‘ich loyiha rejasini yaratganingizda boshlanadi. 2.3- rasm – UML loyihani rejalash jarayoni uchun tipik ish oqimini ko‘rsatuvchi harakat diagrammasi.



2.3- rasm. Loyihani rejalashtirish bosqichlari

Reja o‘zgarishlar–o‘zgartirib bo‘lmaydigan. Tizim haqidagi axborotlar va Loyiha jamoasi loyiha davomida foydali bo‘lib qoladi, siz talabni, jadvalni va risk o‘zgarishlarini aks ettiruvchi rejalarni qaytadan ko‘rib chiqishingiz kerak. O‘zgarish biznesi loyiha rejalarida o‘zgarishni boshqarishni maqsad qilib qo‘yadi. Biznesni maqsadiga ko‘ra, bu barcha loyihalarga ta‘sir etadi.

Rejalash jarayonining boshida siz loyihaga ta‘sir etuvchi omillarni tahlil etishingiz kerak. Bu omillar ta‘minlash kuni, yaroqli xodimlar, umumiy byudjet,foydali uskunalar va boshqalarni talab qiladi. Bularni qo‘shganda, siz loyiha masofasini va ta‘minlivchilarni aniqlashingiz kerak. Masofalar siz jarayonni baholashingizga qarshi jadvaldagi tushunchalar, Misol uchun tizimni testlash uchun qo‘ldan qo‘lga o‘tkazish, Yetkazib beradiganlar mijozga yetkazib beriladigan ish mahsulotlari.

Keyin jarayon sirtmoqqa kiradi. Siz loyiha uchun baholangan jadvalni chizasiz va harakatlar jadvalda aniqlanadi. Bir qancha vaqtdan so‘ng (odatda taxminan 2 yoki 3 hafta) siz jarayonni qaytadan ko‘rib chiqishingiz kerak,

rejalangan jadvaldan o‘zaro farqni yozishingiz kerak. Chunki boshlang‘ich loyiha parametrlarini baholash bo‘lishi muqarrar taxmin.

Loyihani rejalayotganingizda u amaliy bo‘lishligi muhimdir, Ba‘zi ta‘riflardagi muammolar loyiha davomida paydo bo‘ladi va bular loyiha kechiktirilishini boshqaradi.Bunda ahamiyatli ehtimolliklar rejangizga qurilishi kerak , shuning uchun har safar loyiha omillari va masofalari qayta bitim tuzilishi kerak emas.

Agar yaratish ishi bilan bog‘liq jiddiy muammolar bo‘lsa, siz loyiha muvaffaqiyatsizligining riskini qisqartirish uchun yengillashtiruvchi risk (tavakkalchilik)qilishingiz kerak. Bu harakatlarga qo‘shimcha ravishda, siz loyihani qayta rejalashga majbursiz.

Bu mijoz bilan loyiha omillari va yetkazib beradiganlar o‘rtasidagi qayta kelishuvni o‘z ichiga olishi mumkin.Ishning yangi jadvali tugatilishi, shuningdek tasdiqlanishi va xaridor bilan ma‘qullanishi kerak.

Agar bu qayta kelishuv muvaffaqiyatsiz va yengillashtiruvchi risk natijasiz bo‘lsa,keyin siz formal loyihaning texnik qayta ko‘rib chiqishni rejalashtirishingiz kerak. Bu qayta ko‘rib chiqishning maqsadi alternative yaqinlashuvni topish bo‘lib, u loyihani davom ettirishga ruxsat beradi va loyiha va xaridor maqsadi va loyiha yaratuvchisi hali ham bir safda ekanini tekshiradi.

Qayta ko‘rib chiqishning natijasi loyihani bekor qilishga qaror bo‘lishi mumkin. Bu texnik natija yoki ma‘muriy muvafaqqiyatsizlik bo‘lishi mumkin, ammo ko‘pincha loyihaga ta‘sir etuvchi tashqi o‘zgarishlar bo‘lishi mumkin. Katta dastur loyihasini yaratish vaqti ko‘pincha bir necha yillar kerka bo‘ladi. Bu vaqt davomida biznes maqsadi va afzalliklari o‘zgaradi. Bu o‘zgarishlar shunday ma‘noni anglatishi mumkinki, dastur uzoq vaqtda talab qilinmagan yoki haqiqiy loyiha talablari mos emas. Keyin menejment loyihani yaratishni to‘xtatishga qaror qiladi yoki tashkiliy maqsadlarda o‘zgarishni aks ettiruvchi loyiha uchun katta o‘zgarishlar amalga oshiradi.

2.3. Loyihani rejalashtirish

Loyihani jadvallash- bu tashkillanadigan loyihada qanday ishlashni qaror qilish jarayoni. Siz har bir vazifani tugatishga kerak bo‘ladigan calendar vaqtni va bu vazifa ustida kim ish olib borishini baholaysiz. Siz shuningdek, har bir vazifani : serverda talab qilinadigan disk fazosi, maxsuslashtirilgan apparat qismida talab qilingan vaqt ,simulatornva qanday sayohat byudjeti bo‘ladi shu kabi vazifalarni tugatish uchun kerak bo‘ladigan resurslarni ham belgilashingiz kerak.Bu jadval keyinchalik yaxshilanadi va o‘zgartiriladi rejalashtirish davomida.

Tez va rejaga asoslangan jarayonlarning ikkalasiga ham boshlang‘ich loyiha jadvallari kerak, shunga qaramay tafsilot bosqichi tez loyiha rejasida ozroq bo‘lishi mumkin. Bu boshlang‘ich jadval odamlar loyihalarga qanday tayinlanadi va shartnoma asosidagi majburiyatiga qarshi loyiha jarayonini qanday tekshirishida qo‘llaniladi. An’anaviy yaratish jarayonlarida jadvalni to‘ldirish dastlab yaratilgan keyin o‘zgartirilgan. Tez jarayonlarda tugatiladigan jadvalning asosiy bosqichini aniqlaydigan umumiy jadval bo‘lishi kerak. Jadvalga tez yaqinlashuv har bosqichni rejalashda ishlatiladi.

Rejaga asoslangan loyihani jadvallash (2.4- rasmda) ajratilgan vazifalarga loyihada umumiy ishni o‘z ichiga olgan hisobotlarni o‘z ichiga oladi. Vazufalar kamida bir xafta va 2 oydan ko‘p bo‘lmagan holda davom etadi. Yaxshi qismlarga ajratishning ma‘nosi ,nomutanosib vaqt loyiha rejasini yangilash va qayta rejalashga sarflanishi shart. Biror bir vazifa uchun maksimum vaqt 8 10 hafta atrofida bo‘lishi kerak. Agar u bundanda uzoqib ketsa, vazifa loyihani rejalash va jadvallash uchun qismlarga ajratiladi.

Ba‘zi vazifalar parallel ravishda, tizimning turli komponentalarida turli odamlarning ishlashi bilan nihoyasiga yetkaziladi. Siz bu parallel vazifalarni muvofiqlashtirishingiz kerak va ish tashkillashingiz kerak, shuning uchun ish majburiyati optimal bo‘ladi va siz vazifalar o‘rtasidagi keraksiz bog‘liqliklarni kiritmaysiz. Bu vaziyatdan chetga chiqish uchun muhim, butun loyiha kechiktiriladi, chunki kritik vazifa tugatilmagan bo‘ladi.

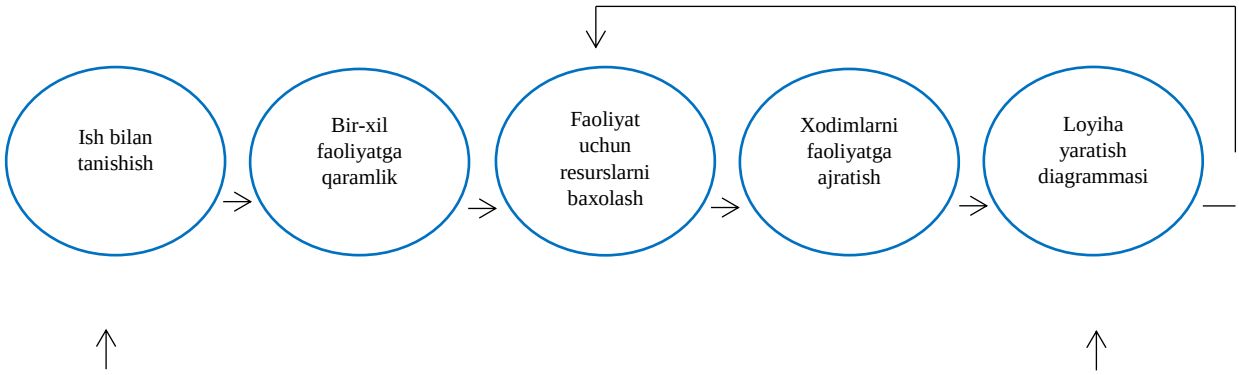
Agar bir loyiha texnik yaratilsa, qachonki barcha xulosalarni ko‘rib chiqishga harakat qilganingizda ,boshlang‘ich baholash deyarli optimistic bo‘ladi. Yangi samolyot,ko‘priklar,mashinaning yangi modellari oldindan bilib bo‘lmaydigan muammolar sababli kechiktiriladi. Jadvallar davomiy uangilanib turilishi shart. Agar loyiha oldingi loyihaga yaqin jadvallansa, oldingi baholash qayta foydalanishi mumkin. Shunday qilib loyiha turli dizayn metodlari va tillarda foydalanilishi mumkin, shuning uchun oldingi loyiha tajribalari yangi loyihani rejalashda yaroqli bo‘lmasligi mumkin.

Men allaqachon taklif qilganimdek, siz jadvalni baholayotganingizda , xato ketuvchi imkoniyatlarni ham hisobga kiritishingiz shart. Odamlarning loyiha ustida ishlashi mavafaqqiyatsizlikka uchrashi mumkin, apparat qism mavafaqqiyatsizlikka uchrashi va dasturiy yoki apparat qism kech ta‘minlanishi mumkin. Agar loyiha yangi va texnik yaratilgan bo‘lsa, uning qismlari qiyinchilikka aylanadi va belgilangan vaqtdan uzoqqa cho‘zilib ketadi.

Yaxshi bosh barmoq qoidasi- agar hech narsa xato ketmaganda baholash, keyin belgilangan muammolarga qarab baholashni oshirish. Belgilanmagan muammolar uchun tasodifiy factor shuningdek baholashga qo‘shiladi. Bu qo‘shimcha tasodifiy factor loyihaning turiga bog‘liq, jarayon parametrlari (so‘nggi muddat, standartlar va h.k) dasturlash injenerlarining sifat va tajribalariga bog‘liq. Tasodifiy baholashlar loyiha uchun talab qilingan vaqtga va natijaga 30%dan 50% gacha qo‘shiladi.

2.3.1. Jadvalni taqdim etish

Loyiha jadvali vazifa bog‘liqligi, kutilgan vaqt, natija va vazifalarni ko‘rsatuvchi katta formatli jadvalda yoki biror jadvalda taqdim etiladi (2.5-rasmda). Shunday qilib taqdim etish uslubi turli harakatlar o‘rtasidagi bog‘liqliklarni ko‘rish uchun qiyin tarzda uni tayyorlaydi.



2.4- rasm. Loyihani rejalashtirish jarayoni

Shu sababga ko‘ra,loyiha jadvalini alternativ grafik taqdim etish o‘qish va tushunishga oson qilib yaratiladi. Taqfim etishning quyidagi ikki turi ishlatiladi:

1. Bar diagrammasi, kalendarga asoslangan, har bir harakat uchun kim javobgar ekani, tugagan vaqt , harakatni boshlanish va tugallanishini ko‘rsatadi. Bu diagrammalar shuningdek kashfiyotchi Genri Ganttning nomiga ‘Gantt charts’ deb ataladi.

2. Faollik tarmoqlari , tarmoq diagrammalari , loyihani tayyorlovchi turli harakatlar o‘rtasidagi bog‘liqliklarni ko‘rsatadi.

Loyihani rejalash uskunasi loyiha jadvali axborotini boshqarish uchun foydalaniladi. Bu uskunalar jadvalga loyixa axborotini kiritishingizni kutadi va loyiha axborotining Ma‘lumotlar Ba‘zasini yaratadi. Bar diagrammalari va faol diagrammalar bu MB dan generatsiyalanadi.

Loyiha harakatlari asosiy rejalash elementi. Har bir harakat:

1. Calendar kunlari yoki oylaridagi davom etish muddati.

2. Natijaviy baho ishni yakunlash uchun person-days or person-months sonini aks ettiradi.

3. Harakat tugallanishidagi so‘nggi muddat.

4. Yaxshilangan nihoya. Bu tugatilayotgan harakatning tushunarli natijasini taqdim etadi.

Rejalanayotgan loyihada siz masofalarni ham aniqlashingiz kerak, ya‘ni ishlab chiqariladigan jarayon tahlili, loyihadagi har bir bosqich. Har bir masofa kichik hisobot tarzda yozib olinadi, tayyorlangan jarayon va bajarilgan ish xulosalanadi.Masofalar bog‘langan harakatlar bilan yoki yakka vazifa bilan birlashgan bo‘ladi. Misol uchun 2.5-shaklda M1 masofa T1 vazifaga birlashgan va M3 T1 va T3 vazifalar juftligiga birlashgan.

Masofaning maxsus turi yetkazib beriladigan loyiha mahsuloti. Yetkazib beriladigan – bu xaridorga yetkazib berilgan mahsulot. Bu loyiha davrining natijasi,shuningdek mahsuslashtirilgan yoki dizaynlashtirilgan. Odatda yetkazib beriluvchilar loyiha shartnomasida mahsuslashtiriladi va mijozning jarayonga bo‘lgan qarashi mana shu yetkazib beriluvchilarga bog‘liq.

Task	Effort (person-days)	Duration (days)	Dependencies
T1	15	10	T1 (M1)
T2	8	15	T2, T4 (M3)
T3	20	15	T1, T2 (M4)
T4	5	10	T1 (M1)
T5	5	10	T4 (M2)
T6	10	5	T3, T6 (M5)
T7	25	20	T7, T8 (M6)
T8	75	25	T9 (M7)
T9	10	15	T10, T11 (M8)
T10	20	15	Dependencies
T11	10	10	T1 (M1)
T12	20	10	T2, T4 (M3)

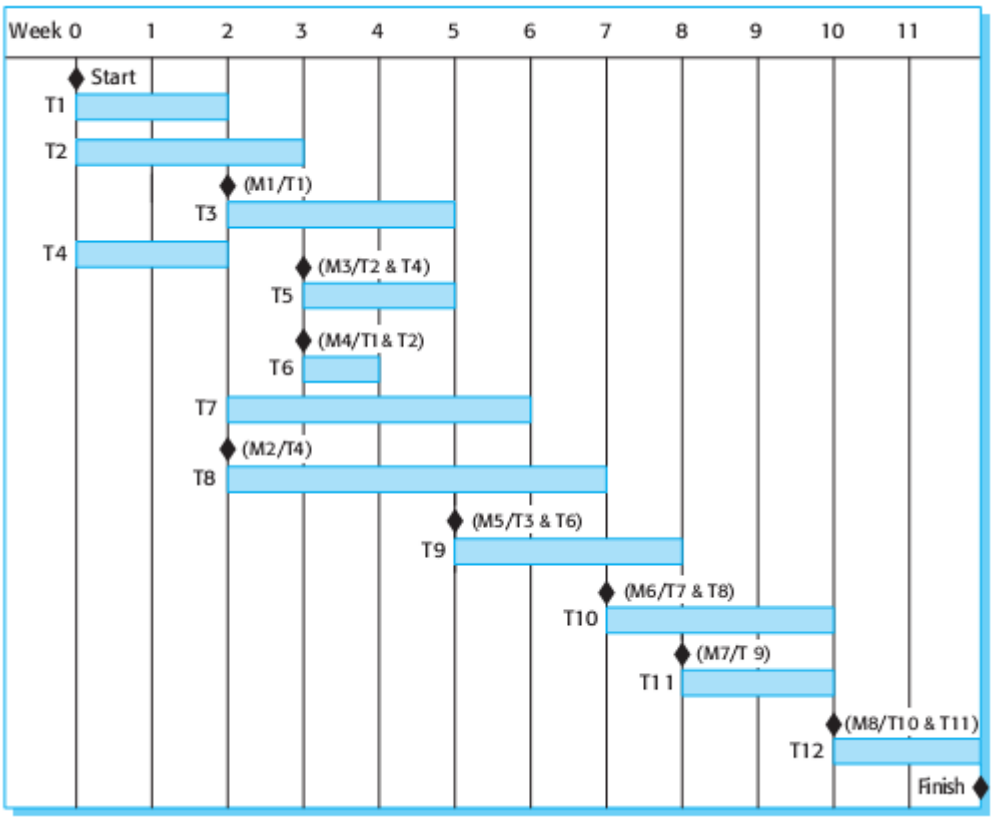
2.5- rasm. Vazifalar, vaqtlar va bog‘liqliklar

Barcha diagrammalari qanday foydalanilganligini tasvirlash uchun , men gipotezaga asoslangan vazifalar to‘plamini yaratganman.(2.5- jadvalda ko‘rsatilgandek). Bu jadval vazifalar, baholangan natija ,davomiylik vaqti va vazifa bog‘liqliklarini ko‘rsatadi. Yuqoridagi jadvaldan ko‘rishingiz mumkinki, T3 vazifa T1 vazifaga bog‘liq,T1 vazifa shuningdek T3 boshlanishidan oldin tugatilgan. Misol uchun T1 komponenta loyihasining tayyorlanishi bo‘lsin,T3 o‘sha loyihani joriy etish. Joriy etish boshlanishidan oldin , loyiha yakunlanishi kerak. Shuni sezish mumkinki, biror vazifa uchunbaholangan davomiylik vaqti talab qilingan natijaga qaraganda ko‘roq va aksincha. Agar natija vaqtga qaraganda ozroq bo‘lsa, buning ma‘nosi to‘liq odamlar to‘liq vaqtda ishlashmayapti. Agar natija davomiylik vaqtidan ortib ketsa,buning ma‘nosi bir necha jamoa a‘zolari bir xil vaqtda vazifa ustida ishlashyapti.

2.6- rasm 2.5- rasmdagi axborotlarni o‘ziga oladi va grafik formatda loyiha jadvali taqdim etiladi. Chapdan o‘ngga o‘qishda, bar(panjaraviy) diagrammasi vazifalar qachon boshlanib tugaganligini aniq ko‘rsatadi. Masofalar (M1,M2 va h.k) diagrammada ko‘rsatiladi. Vazifalar parallel ravishda oxiriga yetkazilgan.(T1,T2 vaT4 vazifalar hammasi loyihaning boshlanishida boshlangan).

Dastur uchun rejalash jadvali ,loyiha menejerlari resurslarni vazifalarga birlashtirishi kerak. Kalit resursi ,albatta, dastur injenerlari va ular loyiha faoliyatlarini belgilashlari kerak. Resursni birlashtirish loyihaga boshqarish uskunalarini kiritish va loyiha ustida ishlayotgan xodimlarni ko‘rsatuvchi diagrammani generatsiyalash(2.7- rasm).

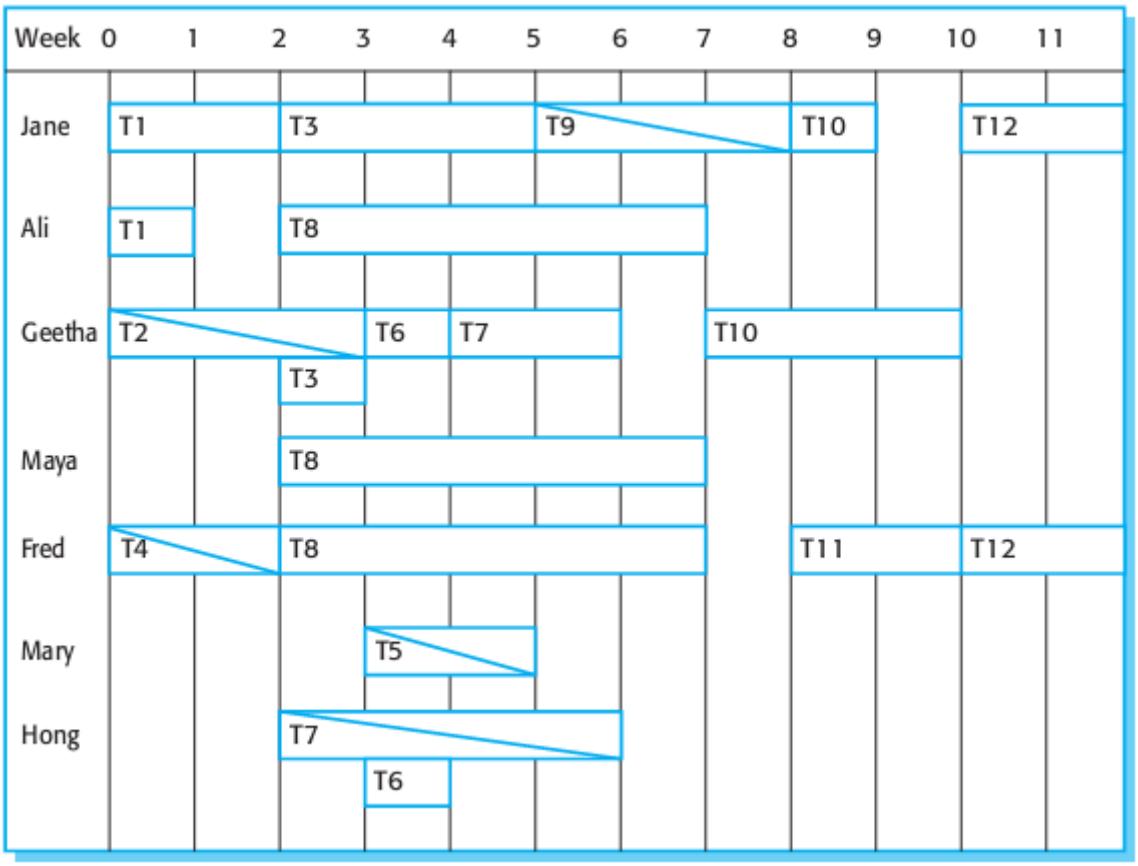
Odamlar bir vaqtda bir vazifaga qaraganda ko‘proq vazifa ustida ishlashlari mumkin va ba‘zan ishlamaydilar. Ular ta‘tilda bo‘lishlari, boshqa loyihada ishlash, mashg‘ulot kurslariga qatnashishlari mumkin.



2.6- rasm. Faoliyat grafigiga ta‘sir etuvchi to‘sizlar

Katta tashkilotlar ko‘p sonli mutaxassislarga ish berishadi. 2.7- rasmda ko‘rishingiz mumkin, Mary mutaxassisu loyihada yakka vazifa ustida ishlayapti. Bu rasm muammolariga sabab bo‘lishi mumkin. Agar loyiha mutaxassis ishlayotganda kechiksa,bu boshqa loyihalarga ta‘sir ko‘rsatadi. Chunki mutaxassis yaroqli emas.

Agar vazifa kechiktirilsa, keying vazifalarga ta‘sir ko‘rsatadi. Ular kechiktirilgan vazifa tugatilmaguncha boshlay olishmaydi. Kechikishlar xodimlar birlashishi bilan bog‘liq jiddiy muammolarga sabab bo‘lishi mumkin,ayniqsa odamlar bir vaqtda bir necha vazifalar ustida ishlayotganda. Agar T vazifa kechiktirilsa, odamlar boshqa W ishni belgilashlari mumkin. Buni yakunlash kechikishga qaraganda ko‘p vaqtni oladi,lekin bir marta belgilanganda , ular haqiqiy vazifa Tga osongina qayta olmaydilar. Bu T da ko‘proq kechiktirishlarga olib kelishi mumkin.



2.7- rasm. Hodimlarni farqlash grafigi

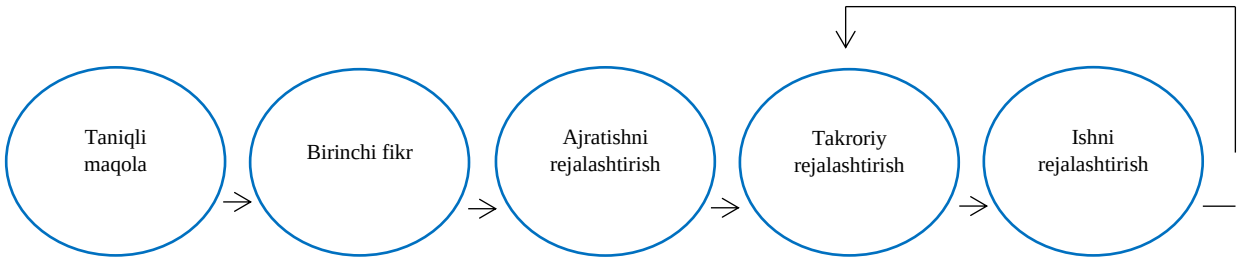
2.4. Takror rejalashtirish

Dastur yaratishning takroriy metodlari deganda ma‘lum kattalashtirishda dastur yaratilganda va xaridorlar ta‘minlanganda takroriy murojaat tushuniladi. Rejaga asoslangan murojaatga o‘xshamagan bu kattalashtirishlarning funksionalligi yaratishda rejalashtirilmagan ammo yaratish davomida qaror qilingan. Bu qarorga binoan kattalashtirish(inkrement) jarayonga hamda xaridorning afzalliklariga bog‘liq. Bu murojaat xaridor afzalligi va o‘zgarish talabi shunung uchun u qulay rejani his qiladi. Cohn’s book {Cohn, 2005 ##1735} –bu tekroriy loyihalashda rejalashning har tomonlama muhokamasi haqida.

Tezkor loyihalash qo‘llaniladi shuningdek Scrum (Schwaber, 2004) va dasturlash (Beck, 2000) rejalash uchun ikkita murojaatga ega.Boshlash vaqtiga muvofiq rejaga asoslangan yaratish va ishlab chiqishni rejalash:

1. Rejalashni ommaga taqdim etish. Tizimni taqdim etishda o‘z ichiga olgan sifalarlarda bir qancha oylarni qidirish.

2. Takrorlanuvchi rejalash. Tizimning keying inkrementini rejalashning markazida turadi. Bu jamoa uchun ishning 2 yoki 4 xaftasi hisoblanadi.



2.8- rasm. Qat’iy dasturlashni rejalashtirish

Men allaqachon 2-bobda Scrum yaqinlashuvi haqida aytib o‘tganman. Men bu yerda eng oxirgi dasturlashda(XP) rejalashni to‘plamoqchiman. Bu “planning game” deb ataladi va u odatda yaratish jamoasini, xaridor vakillarini o‘z ichiga oladi. 2.8- rasmda rejalash o‘yinining bosqichlari tasvirlangan.

XPda gi tizimning o‘ziga xos xususiyatlarini aniqlash tizimda o‘z ichiga olingan xususiyatlarni aks ettiruvchi foydalanuvchi hisobotlariga asoslangan. Loyihani boshlashda jamoa va xaridor hisobotlar to‘plamini aniqlashga harakat qilishadi. Ba‘zi funksionalliklar o‘tkizib yuborilishi aniq ,ammo bu bosqichda muhim emas.

Keyingi bosqich – baholash bosqichi. Loyiha jamoasi hisobotlarni o‘qiydi va muhokama qiladi va vaqtning biror qismi uchun ularga baho beradi. Nisbiy baholash ko‘pincha absolyut baholashdan osonroq. Odamlar ko‘pincha biror narsani bajarish uchun kerak bo‘ladigan vaqtni baholash uchun qiyinchilik topishadi. Bir safar baholash yakunlanadi ,jamoalar tasavvurdagi natija tushunchalarini yig‘ishadi.Murakkab hisobot 8 ta fikrdan, oddiysi 2 ta fir-tushunchaga ega bo‘ladi.

Bir martalik hisobotlar baholangan, nisbiy natija esa “tezlik” tasavvuridan foydalar talab qilinadigan umumiy natijaning birinchi bahosiga o‘giriladi. XP da”tezlik” – bu kuniga jamoa tomonidan belgilangan natijalar miqdori. Tezlikni baxolash taxminan ammo loyihani yaratish davomida yaxshilanadi. Siz tezlikni

baholashga ega bo‘lsangiz , person-days(inson-kun) dagi umumiy natijani hisoblay olasiz. Taqdim etish rejasi hisobotlarni yaxshilash va tanlashni o‘z ichiga olingan bo‘lib , u tizimni taqdim etish uchun ta‘minlangan xususiyatlarni aks ettiradi. Bu jarayonda xaridor hisobga olinishi kerak. Taqdim etish vaqti tanlangan va hisobotlar agar natijaviy baholash vaqt bilan o‘zgarmas bo‘lsa, imtihon qilinadi. Agar unday bo‘lmasa hisobotlar listga qo‘shiladi yoki o‘chirib tashlanadi.

Takrorlashni rejalash takror yaratish jarayonining birinchi bosqichi. Hisobotlar tanlangan takroriylik ,jamoa tezligi va takrorlashni ta‘minlash vaqtini aks ettiruvchi (odatda 2-3 hafta) hisobotlar soni bilan bajariladi. Qachonki takroriylik vaqtiga erishilsa, takrorlash bajariladi, hattoki agar hisobotlarning hammasi bajarilmagan bo‘lsa ham. Jamoa natijaga qo‘shilgan va bajarilgan hisobotlarni ko‘rib chiqadi. Keyin tezlik qayta hisoblanadi va u tizimning keyingi taqdim etish rejalashtirishida qo‘llaniladi. Har bir takrorlashning boshlanishida rejalash bosqichlari ko‘proq bo‘ladi. Yaratish vazifasi 4-16 soatni oladi. Vazifalarning barchasi ro‘yhatlangan qaytarish hisobotlarining barchasini bajarish uchun tugatilishi kerak.Keyin individual yaratuvchilar bajariladigan mahsus vazifalarni belgilaydilar. Har bir yaratuvchi o‘zining tezligini biladi.

Quyida vazifa taqsimotiga yaqinlashishdan ikki muhim foyda ko‘zlangan:

1. Butun jamoa vazifalar tahlilini ko‘rib chiqadi va ularda shuningdek qanday jamoa a‘zolari bajarishi haqidagi tushunchalarga ega bo‘lishadi.

2. Individual yaratuvchilar vazifalarni tanlaydilar;ularda shuningdek bu vazifalarga nisbatan egalik hissi bo‘ladi va bu ehtimol vazifani bajarish uchun ularni motivatsiyalaydi(undaydi).

Jarayon qayta ko‘rib chiqiladi. Bu bosqichda hisobot natijasining yarmi to‘ldirilgan bolishi kerak. Shuningdek, agar takrorlash 24 ta hisobot tushunchalari va 36 ta vazifani o‘z ichiga olsa, 12 ta hisobot va 18 ta vazifa bajariladi. Agar bunday holat bo‘lmasa, xaridorga maslahat beriladi va hisobotlar takrorlashdan o‘chirib yuboriladi.

Rejalashga murojaat bir qancha afzalliklarga ega, ya‘ni dastur rejalashtirilgandek taqdim etiladi. Agar ish ruxsat etilgan vaqtda tugatilmasa, XP

falsafasi –bu kengaytirilgan rasmga qaraganda ko‘proq ish faoliyati doirasini qisqartirish.Shunday qilib,ba‘zi holatlarda inkrement foydali bo‘lish uchun yetarli bo‘lmasligi mumkin.Faoliyatni kamaytirish xaridorlarga qo‘shimcha ish yaratishi mumkin agar ular tizim va boshqasini taqdim etish o‘rtasidagi ish amaliyotlarini o‘zgartirganlarida yoki to‘liq tugatilmagan tizimdan foydalanishga majbur bo‘lishganda.

Takrorlanuvchi rejalashdagi asosiy qiyinchilik bu ya‘ni xaridor zaruriyati va imkoniyatlariga bog‘liq. Amaliyotda, bu rejalashga qiyin bo‘ladi, xaridor ba‘zan boshqa ishga birinchilikni berishi shart. Xaridorlar an’anaviy loyiha rejalariga ko‘proq yaqin bo‘lishi mumkin va tezkor rejalash loyihasiga buyurtma berishi uchun qiyinchiliklarga duch kelishi mumkin.

Mustahkam yaratish jamoasi hisobotlarni birgalikda muhokama qiladi. Qayerda jamoa katta va geografik taqsimlansa , yoki qachon jamoa a‘zoligi tez o‘zgarsa , bu amaliy jihatdan har bir inson uchun imkonsiz bo‘ladi. Natijada katta loyihalar loyihalash menejmentiga an’anaviy murojaatdan foydalanib rejalashtiriladi.

2.5.Texnik usullarni baholash

Loyiha rejasini baholash qiyin. Yuqori darajali foydalanuvchi talabi izohi sifatida boshlang‘ich baholarni berishingiz kerak. Dasturni notanish kompyuterlarda ishlashitishingizga yoki yangi rivojlanish texnologiyalaridan foydalanishingizga to‘g‘ri kelishi mumkin. Loyihaga jalb qilinadigan mutahasislarlar va ularning mahorati balkim mavhum bo‘lar. Loyihaning dastlabki bosqichlarida juda ko‘p loyihaning rivojlanishi qanchaga baholanishini tahmin qilib bo‘lmaydigan noaniqliklar mavjud.

Hattoki baholash va bahoga zo‘r berishga bo‘lgan turli qarashlarni aniqligini belgilashda ham fundamental qiyinchilik bor. Loyiha baholari ko‘pincha bir tomonlama bo‘ladi. Baholash loyiha budjetini ifodalash orqali bajariladi va

loyihada budjet tuzilishi amalga oshirilgani uchun unga moslashadi. Budjetdagi loyiha bunga yaratilayotgan loyiha sifat qiymatlarida erishadi.

Taxmin qilingan baho ekperimentining afzalligini belgilashdan foydalanilmagan joydagi nazorat ostidagi loyihani baholash haqida men bilmayman. Nazorat ostidagi tajriba loyiha menejeriga baho tahmini ko‘rsatmas edi. Haqiqiy baholar keyin taxminiy baholarga taqqoslanar edi. Lekin tashkilotga dasturiy harakat va baholar taxmin qilinishi zarur:

1. Tajribaga asoslangan usullar. Kelgusi harakat talablarini tahmin baholash menejerning o‘tgan loyihalardagi tajribasiga va asosiy sohasiga bog‘liq. Asosiysi, natijaga erishish talablari qanday bo‘lishi haqida menejer chuqur mulohaza yuritadi.

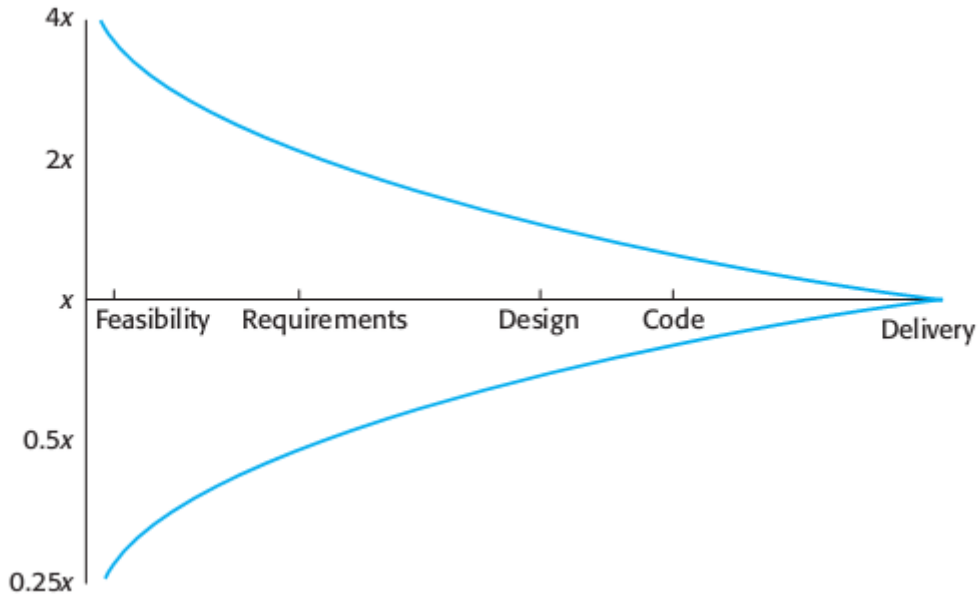
2. Algoritmli baholash modeli. Bunday yondashuvda, mahsulot xususiyatlari, o‘lchami va jarayon xarakteri kabi jalb qilingan mutahasislarni tajribasi loyiha harajatini hisoblashda formulali yondashuvdan foydalaniladi.

Har ikki holatda, tog‘ridan-to‘g‘ri ishni baholash yoki loyiha va mahsulot harakterlarini baholash uchun mulohazangizdan foydalanishingiz kerak. Loyihani boshlang‘ich fazasida, bu baholashlarda hatolarning keng farqi bo‘ladi. Bir qancha loyihalardan yig‘ilgan ma‘lumotga asoslanib, Boehm et al.(1995) shuni aniqladiki, boshlang‘ich baholar ahamiyatli farq qiladi. Agar talab qilingan ishning dastlabki bahosi x oylik ish, tizim yetkazib berilgandagi haqiqiy ishning chegarasi 0.25x dan 4x gacha deb hisoblanishi mumkin. Rivojlantirishni(dasturni) rejalashtirish davomida, baholar yana va yana aniqroq bo‘ladi, loyiha rivojlanishlari kabidir (2.9 rasm).

Tajribaga asoslangan usullar menejerning o‘tgan loyihalardagi tajribasiga suyanadi va haqiqiy ish loyihadagi dasturni rivojlantirishga bog‘liq mashg‘ulotlarga sarflanadi. O‘ziga xos tarzda, alohida ishlab chiqilib eltib beriluvchilarni va rivojlantirilishi kerak bo‘lgan turli dastur komponenentalari va tizimlarni aniqlab berasiz. Bularni varoqqa yozasiz, individual ularni baholaysiz va to‘liq talab qilingan ishni hisoblaysiz. Bu odatda odamlarni ishni baholashga jalb etishda va har bir guruh azosini bahosini tushuntirib berishini so‘rashda yordam

beradi. Bu ko‘pincha boshqalar hisobga olmagan faktorlarni ko‘rsatadi va siz kelishilgan guruh bahosiga uni qo‘shasiz.

Tajribaga asoslangan usulning qiyinchiligi - yangi dastur loyihasi oldingisi bilan ko‘p o‘xshashlikka ega bo‘lmasligi mumkin. Dasturni rivojlantirish juda tez o‘zgaradi va loyiha notanish usullardan, web servis, COTSga asoslangan rivojlantirish yoki AJAX kabi. Agar siz bunday usullar bilan ishlamagan bolsangiz, sizning oldingi tajribangiz talab qilingan ishni baholashda yordam bermasligi mumkin, mahsulotga aniq narx chiqarish ya‘ni aniq baholarni mo‘ljallashni yanada qiyinlashtirib.



2.9- Algoritmli baholash modeli

Algoritmik baholash modeli loyiha o‘lchamlari tahminlari; Rivojlantirilayotgan dastur turi va boshqa jamoa, jarayon va mahsulot faktorlariga asoslangan mahsulot narxlarini oldindan bashorat qilishda matematik formuladan foydalanadi. Algoritmik baholash modeli narxlarni va tugatilgan maxsulot xususiyatlarini va haqiqiy tajribaga eng mos formulani topishni tahlil qilish orqali qurilishi mumkin.

Algoritmik baholash modellaridan asosan dasturni rivojlantirish narxlarini taxminiy qiymatlarini xosil qilishda foydalaniladi. Shunga qaramasdan, Boehm va uning hamkorlari(2000) bu modelning boshqa foydali jihatlarini muhokama qildi.

Bular dasturchi kompaniyalarda investorlarga tahminiy baholarni tayyorlash; hatarlarni aniqlashga ko‘maklashish uchun alternativ startegiyalar; qayta foydalanish haqida ishonchli qarorlar yoki buyurtmalar qabul qilish kabilardir. Dastur loyihasidagi ishni baholash uchun algoritmik model ko‘pincha oddiy fo‘rmulaga asoslanadi:

$$Ish=A*Size(o‘lcham)^B*M$$

A mahalliy tashkiliy tajribalarga bog‘liq o‘zgarmas faktor va rivojlantirilgan dasturning turi. O‘lcham yo dastur kodini o‘lchamini baholash yo funksiya yoki ilovada ifodalangan funksional baho. **B** darajaning qiymati odatda 1 va 1.5 o‘rtasida yotadi. **M** jarayon, mahsulot va rivojlanish xususiyatlari — dastur uchun bog‘liqlik talablari va rivojlantiruvchi jamoaning tajribasi kabilarni moslashtirish orqali olinadigan ko‘paytuvchi.

Yetkazib berilgan tizimdagi manba kodinining qatorlar soni (SLOC) ko‘plab algoritmli baholash modellarida fundamental o‘lchov birlik. O‘lchamni baholash boshqa loyihalarga o‘xshashligi bilan baholash, funksiya yoki ilova xususiyatlarini kod o‘lchamiga o‘zgartirish orqali baholash, tizim komponentalarini o‘lchamlarini kotegogoriyalash orqali baholash va berilgan komponenta havolasidan foydalanib komponenta o‘lchamini baholash yoki bu, oddiy, muhandislik mulohazasidan savol.

Ko‘plab algoritmli baholash modellarida darajali komponenta (yuqoridagi tenglikdagi **B** kabi) mavjud, tizimning o‘lchami va murakkanligiga bog‘liq bo‘lgan. Bu shu haqiqatni ko‘rsatadiki, narxlar loyiha o‘lchamining qatorlari bilan ko‘tarilmaydi. Dasturning o‘lchami va murakkabligi oshganda, qo‘shimcha narxlar kattaroq jamoalar orasida kommunikatsiya, murakkabroq tuzilma boshqaruvi, qiyinroq tizim integratsiyasi va shu kabilar tufayli kelib chiqadi.

Barcha algoritmli modellarda bir hil muammo bor:

1. Loyihada o‘lchamni baholash odatda dastlabki bosqichda qiyin, faqat ishni qanday bo‘lishi aniq bo‘lganda. Funksiya-nuqta va ilova-nuqta baholarini(keyinroq ko‘rasiz) ishlan chiqish oson kod o‘lchamini baholashdan ko‘ra lekin shunga qaramay odatda noaniq.

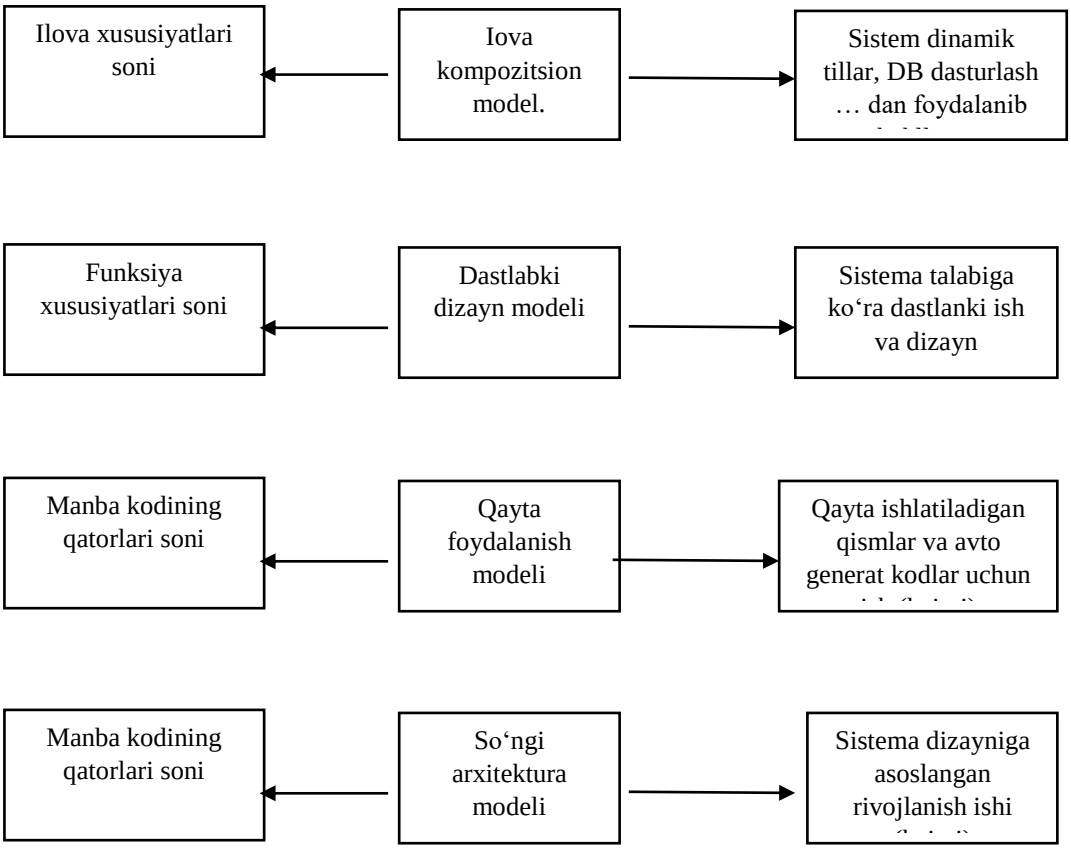
2. **B** va **M** ga qiymat berayotgan omillarning baholari subyektiv. Baholar bir odamniki boshqasidan farq qiladi, ularning kelib chiqishi va rivojlanitirilayotgan tizimga oid tajrobasiga bog‘liq tarzda.

Aniq kod o‘lchamini baholash lohida boshlang‘ich davrda qiyin chunki tugallangan dasturning o‘lchami baholash talab etilganda shakllanmagan dizayn qaroriga bog‘liq. Masalan, yuqori darajali ma‘lumot boshqaruvini talab etgan ilova yo o‘zining ma‘lumotlar boshqaruv tizimini amalga oshiri yoki tijorat malumotlar bazasi tizimidan foydalanishi mumkin. Dastlabki narxni baholashda, bajarilish talablarini yetarlicha yaxshi qondiradigan tijorat ma‘lumot bazasi bormi buni bilishingiz qiyin. Siz shuning uchun tizimgaqancha ma‘lumotni boshqaruvchi kod kiritilishini bilmaysiz.

Tizimni rivojlantirishda ishlatilgan dasturlash tili ham rivojtirilajak kodlar qatorining soniga ta‘sir etadi. Java kabi til C ishlatilgandagidan ko‘ra ko‘proq kodlar qatorini kerakligini toqazo etadi. Shunga qaramay, bu qo‘shimcha kod ko‘proq kompilyatsiya vaqtini tekshirish imkonini beradi shuning uchun yaroqlilik narxlari qisqartirilishi mumki. Qanday bu hisobga olinishi mumkin. Buning ustiga, oldingi loyihalardagi sezilarli miqdordagi kodlar qismidan qayta foydalanish mumkin va buni yodda tutish uchun o‘lcham bahosi moslashtirilishi kerak.

Algoritmli baholash modellari bir tizimni rivojlantirish uchun qancha ish talab etilishini aniqlashni tizimli usuli. Shunga qaramay, bu modellar murakkab va foydalanish qiyin. Juda ko‘p xususiyatlar va ularning qiymatini baholashda ancha noaniqlik chegaralari mavjud. Bu murakkabliklar kuchli foydalanuvchilarni ham gangitib qo‘yadi va shu sababdan algoritmli baholash modelining amaliy tadbiqi oz miqdordagi kompaniyalar bilan chegaralanib qolgan. Yana bir gangitadigan to‘siq algoritmli model o‘lchov (kalibratsiya) uchun zaruratligidir. Modeldan foydalanuvchilar o‘zlarining modellarini va xususiyat qiymatlarini shaxsiy tarixiy loyiharidan foydalanib o‘lchashi (kalibrlashi) lozim, xuddi bu mahalliy amaliyot va tajribani ko‘rsatganidek. Shunga qaramay, juda kam tashkilotlar o‘tgan loyilaridan yetarlicha modelli o‘lchashni qo‘llab quvvatlaydigan shakldagi ma‘lumotlarni yig‘ishgan. Algoritmli modeldan amaliy foydalanish, shuning uchun, model

parametrlari uchun chop etilgan qiymatlar bilan boshlanishi kerak. Bular o‘zlarining tashkilotiga qanday yaqin aloqadorligini bilish modelyer uchun amaliy jihatdan imkonsiz. Agar siz algoritmik narxni baholash modelidan foydalansangiz, yagona bahodan ko‘ra baholar guruhini (eng yomon, kutilgan, eng yaxshi) rivojlantirishingiz kerak va baholash fo‘rmulasini barchasiga qo‘llashingiz kerak.



2.10- rasm. COCOMO II modelini baholash

Baholar siz rivojlantirilayotgan dastur turini yaxshi tushunganingizda, mahalliy ma‘lumotdan foydalanib modelni kalibrlaganingizda yoki dasturlash tili va qurilmalar tanlovi oldindan aniqlanganda aniq bo‘lishga moyil bo‘ladi.

2.5.1. COCOMO II modeli

Dasturiy ta‘minot loyihasiga ketadigan xarakat grafig va sarf-xarajatlarni baxolash uchun bir nechta modellar taklif etilgan edi. Men bu yuerda muhokama

qilayotgan model COCOMO II. Bu dasturiy ta‘minot loyihalarining ko‘p miqdordan olingan ma‘lumotlar asosida yaratilgan empirik model. Bu ma‘lumotlar taxlil qilinib kuzatuvlarga mos keluvchi formulalar topilgan. Bu formulalar tizim o‘lchami va maxsulotni loyiha va jamoa omillarini tizimni rivojlantirishga ketadigan harakatlarga bog‘lab beradi. COCOMO II hujjatlari yaxshi rasmilashtirilgan va nomulkiy baxolash modeli.

COCOMO II modeli birinchi COCOMO narxni baxolashning modeli asosida ishlab chiqilgan, bu model asosan arginal koddan iborat edi (Boehm , 1981, Boehm and Royce, 1989). COCOMO II modeli tezkor rivojlantirish, dinamik tillar, tarkibiy rivojlantirish va ma‘lumotlar bazasiga asoslangan dasturlash kabi zamonaviy yondashuvlarni inobatga oladi. COCOMO II modeli II bo‘limda tasnif etilgan rivojlantirishning spiral modelini qo‘llab-quvvatlaydi va ko‘proq detallashtirilgan baxolashlarni bajaruvchi ost modellardan iborat.

COCOMO II modeli tarkibiga kiruvchi ost modellar (3.10 rasm):

1. *Amaliy tarkibli model.* Bu model ko‘p marotaba ishlatiladigan komponentlardan script yoki ma‘lumotlar bazasini dasturlashdan iborat tizimlarni rivojlantirishga ketadigan harakatlarni baxolash modeli.

Dasturiy ta‘minot o‘lchamini baxolash punktlarini belgilashga asoslanadi va kerak bo‘lgan harakatni baxolash uchun oddiy o‘lcham / unumdorlik formulasi ishlatiladi. Dasturdagi amaliy punklar miqdori – bu ko‘rasatiladigan aloxida ekranlarning miqdorini, taqdim ettilgan xisobotlar miqdorini, Obyektga yo‘naltirilgan dasturlash tillarida (Java kabi) modular miqdori, til skriptini chiziqlari miqdori yoki ma‘lumotlar bazsini dastur kodini baxolashdur.

2. *Dastlabki dizayn modeli.* Bu model tizimning dastlabki bosqichlarida ishlatiladi, tizim dizayini talablarga asosan yaratilgan bo‘ladi. Baxolash men kirish qismida muxokama qilgan baxolashning standart formulasiga 7 ta ko‘paytuvchilar soddalsahtirilgan to‘plamiga asoslanadi. Baxolash funksianallik birliklariga asoslangan bo‘lib ular bunda dastlabki kodning chiziqlariga o‘zgartiriladi. Funksionallik birliklari dasturning ishlash sifatini tekshirishning tilga bog‘lanmagan usuli. Funksionallik birliklarining umumiy birliklarini xisoblash

uchun siz tizimdagi tashqi kirish va chiqish, foydalanuvchilarning o‘zaro muloqatlari, tashqi interfeyslari, ma‘lumotlar bazasining fayl va jadvallarni miqdorini baxolash yoki o‘lchamlarini bilishingiz kerak.

3. *Takroriy foydalanish modeli.* Bu model ko‘p ishlatiladigan component yoki aftomatik ravshda yaratiladigan dasturiy kodlarni birlashtirishga ketadigan harakatni xisoblash uchun ishlatiladi. U odatda post-arxitekturali model bilan ishlatiladi.

4. *Kuzatov arxitektura modeli.* Agar tizim arxitekturasi ishlab chiqilgan bo‘lsa dasturiy ta‘minot o‘lchamining aniq baxolash imkonmi bo‘ladi. Bunda yana yuqorida muhokama qilingan narxni baxolashni standart formulasi ishlatiladi. Lekin bunda kengaytirilgan 17 ta ko‘paytiruvchi to‘plami ishlatiladi: xodimlar qobilyati, maxsulot va loyiha xususiyatlari.

Albatta katta tizimlarda tizimning har-xil qisimlarini rivojlantirish uchun har-xil texnologiya ishlatirlishi mumkun va siz hamma qisimlarni bir xil aniqlikda baxolay olmaysiz. Bunday xolatlarda siz tizimning har bir qismi uchun tegishli ost modelni ishlatishingiz mumkun va natijarni murakkab baxolashga birlashtirishingiz mumkun.

Yaratuvchining tajribasi va qobilyati	**Juda past**	**Past**	**O‘rtacha**	**Yuqori**	**Juda yuqori**
ICASE rivojlanganligi va xususiyatlari	Juda past	Past	O‘rtacha	Yuqori	Juda yuqori
PROD (NAP/oy)	4	7	13	25	50

2.11- rasm. Amaliy qismlar unumdorligi

Amaliy tarkib modeli.

Amaliy tarkib modeli COCOMO II ga prototip loyihalar va mavjud komponentlarni birlashtirib dasturiy ta‘minot yaratish loyihalarida sarflanadigan harakatlarni baxolash uchun kiritilgan. Unda o‘lchangan amaliy punktlarni (bazida obyekt punktlar deb nomlanadi) baxolashga asoslanadi. Ular amaliy punktlar unumdorligi bo‘yicha standartbaxolanadi. Bunda baxolash har bir amaliy g‘oyani

rivojlantirish murakkabligiga ko‘ra moslashtiriladi (Boehm va boshqalar, 2000). Unumdorlik yaratuvchining tajribasi va qobilyatiga bog‘liq shuningdek dasturiy vositalar (ICASE) xususiyatlari odatda rivijlanishda. 2.11- rasmda COCOMO yaratuvchilari tomonidan taklif etilgan amaliy punktlarning unumdorligi darajalari ko‘rsatilgan (Boehm va boshqalar, 1995).

Amliy tarkib odatda dasturiy ta‘minotdan qayta foydalanishni o‘llaydi. Tiizmdagi amaliy punktlarini bazilarida ko‘p marotaba ishlatiladigan komponentlar bo‘lishi shubxasiz. Demak siz baxolashda qayta ishlashni xisobga olishingiz kerak. Shuning uchun tizim prototiplarini yarataishga sarflanadigan harakatni xosoblash uchun yakuniy formula quyidagicha:

PM (NAP (1 %reuse100)) PROD

PM bu odam harakatini oy hisobida. NAP bu yaratilayotgan tizimdagi amaliy punkitlarning umumiy miqdori “% reuse” qayta foydalanilgan kod miqdorini baxolash. PROD 2.11- rasmda ko‘rsatilgandek amaliy punkt umumdorligi. Model taqribiy baxolashni bajaradi, chunki unda qayta ishlatilishga ketgan qo‘shimcha xarakat xispbga olinmaydi.

Dastlabki dizayin modeli

Bu model loyihaning boshlang‘ich bosqichlarida ishlatilishi mumkun, tizimning batafsil arxitektura modeli tayyor bo‘lishidan oldin. Dizayinni dastlabki baxolanishi tanlash uchun foydalidur. Bunda siz foydalanuvchi tomonidan qo‘yilgan talablarni amalga oshirishning xar-xil usullarini taqqoslab ko‘rishingiz kerak bo‘ladi.

Dizayinning dastlabkki modeli shuni nazarda tutadiki, foydalanuvhuning talablari kelishilgan va tizimni loyihalashtirish jarayonining boshlang‘ich bosqichlari to‘liq yo‘lga qo‘yilgan. Bu bosqichda sizning maqsadingiz tez taxminiy smetani bajarishdur. Shuning uchun siz ishni kamaytirish harakatida bo‘lishingiz kerak, masalan ko‘p marotaba ishlatiladigan kodni kiritishga ketadigan harakatni 0 ga teng deb xisoblash kerak.

Bu bosqichda bajarilgan baxolash algoritmik modellar uchun standart formulasiga asoslangan, aynan:

$$Effort(kuch)=A*size(o‘lcham)^{B}*M$$

O‘zidagi ma‘lumotlarning katta to‘plamiga asoslanib Boehm A koyfisenti 2.94 ga teng bo‘lishini taklif qildi. Tizim o‘lchami KSLOC da berilgan, bu dastlabki kod chiziqlarining mingliklarda xisoblanadi. Dasturiy ta‘minotdagi funksional punktlar miqdorini baxolab siz KSLOC ni xisoblab chiqasiz. KSLOC ni tizimning dastlabki smeta o‘lchamini xisoblash uchun siz har-xil dasturlash tillarida yozilgan funksional punktlarni bog‘lovchi dasturiy ta‘minotni bog‘lash uchun standart jadvallardan foydalanasiz.

B exponentasi loyiha o‘lchami kattalashganda harakat kattalashishini aks ettiradi. Loyihaning yangiligi, rivojlanish moslashuvchanligi havf-hatarni aniqlash jarayonlari, rivojlantirish jamoasini birdamligi va tashkilot tajribasiga (2 bobda) binoan uning miqdori 1.1 dan 1,24 gacha o‘zgarishi mumkun. Xisoblashning bu turini afzalliklarini men COCOMO II post arxitektura modelini tasniflashda bu parametrlarni ishlatib ko‘rsatganman.

Xarakatni xisoblash bunda quyidagicha amalga oshiriladi:

$$PM=2.94*size^{(1.1-1.24)}*M$$

Bu yerda

$$M=PERS*RCPX*RUSE*PDIF*PREX*FCIL*SCED$$

M ko‘paytiruvchisi 7 ta loyiha va jarayon belgilariga asoslangan bo‘lib, baxolashni ko‘paytirishi va kamaytirishi mumkun. Dastlabki dizayn modelidagi atrebutlar: maxsulot ishonchliligi va murakkabligi (PCPX), qayta ishlatilish (RUSE), platformaning qiyinchiligi (PDIF), hodimlar qobilyatlari (PREX), grafik (SCED) va qo‘llab-quvvatlash vositalari (FCIL). Bu atrebutlarni men internetdagi kitobda tushuntiraman. Bu atrebutlar ahamiyatini jadvalda ko‘rishingiz mumkun, unda 1 ‘juda past’ va 6 ‘juda yuqorini bildiradi’.

Qayta ishlatish modeli.

2 bobda aytganimdek dasturiy ta‘minotni qayta ishlatish xozir juda keng tarqalgan. Eng katta tizimlar avvalgi ishchi loyihalarda ishlatilgan kodni katta miqdorini o‘z ichiga oladi. Qayta ishlatish modeli ko‘p marotaba ishlatiladigan

kodini birlashtirish yoki yaratishga ketadigan harakatni bgaxolash uchun ishlatiladi.

COCOMO II qayta ishlaydigan kodni ikkita turini ko‘rib chiqadi. ‘Qora quti’ kodi, bu kod uni tushunmagan yoki o‘zgartirilmagan holda ishlatiladi. ‘Qora quti’ kodini rivojlantirishga ketadigan kuch 0 ga teng deb xisoblanadi.

‘Oq quti’ kodi yangi kod yoki boshqa qayta ishlatiladigan komponentlar uchun moslashtirish kerak. Qayta ishlatilishi uchun harakat talab etiladi, chunki kod tizimda to‘g‘ri ishlashi uchun tushunilishi yoki o‘zgartirilishi kerak.

Ko‘p tizimlar 5– bobda aytilganidek tizim modellaridan olingan avtomarik ishlab chiqilgan kodni o‘z ichiga oladi. Model taxlillanadi (ko‘pincha UML da) va modelda belgilangan ob’yektlarni xosil qiluvchi kod yaratiladi. Qayta ishlatish modellarining COCOMO II sida bu yaratilgan kodni birlashtirishga ketadigan kuchni baxolash formulasi ishlatiladi:

PM_{auto}=(ASLOC*AT/100)/ATPROD// yaratilgan kodni baxolash

ASLOC bu qayta ishlatilgan kod chiziqlarining umumiy miqdori, buning ichida avtomatik ishlab chiqilgan kod.

AT bu avtomatik tarzda ishlab chiqilgan qayta ishlatilgan kod foyizi.

ATPROD bu bunday kodni birlashtirishda muhandislar unmdorligi.

Boehm va boshqalar (2000) ATPROD ni 2,400 ga teng deb xisoblashdi. Shuning uchun umumiy holda qayta ishlatilgan dastlabgi kod 20000 chiziq bo‘lsa va uning 30% avtomatik tarizda yaratilgan bo‘lsa yaratilgan koni birlashtirish kuchi quyidagicha xisoblanadi:

(20000*30/100)/2400=2.5 kuch oy xisobida // takroriy kod

Boshqa tizmlardan olingan qayta ishlatiladigan kodi birlashtirishga ketadigan kuchni xisoblash uchun alohida xisob kitob bajariladi. Qayta ishlatish modeli kuchni qayta ishlatilgan komponentlar miqdorini baxolashdan ajralgan holda xisoblamaydi. Qayta ishlatilgan kod chiziqlar miqdoriga asoslangan bu model yangi kod chiziqlarini ekvivalent miqdorini xisoblashni ta‘minlaydi (ESLOC). Bunda o‘zgartirilish kerak bo‘lgan qayta ishlatiladigan kod chiziqlari miqdori asos bo‘lib, bu ko‘paytiruvchi komponentlarni qayta ishlatish uchun siz

bajargan ish hajmini aks ettiradi. ESLOC hisoblash formulasi dasturiy ta‘minotni tushunishga, qayta ishlatiladigan kodga o‘zgartirish kiritishga va bu kodni tizimda birlashtirishga ketadigan kuchni hisobga oladi.

Quyidagi formula dastlabki kodni ekvivalent chiziqlarni miqdorini xisoblash uchun ishlatiladi:

ESLOC=ASLOC*AAM

ESLOC bu yangi kod chiziqlarining ekvivalent miqdori.

ASLOC bu o‘zgartirilishi kerak bo‘lgan komponentlardagi kod chiziqlarining miqdori.

AAM bu adaptatsiyani nazorat qiluvchi ko‘paytiruvchi.

Qayta ishlatish xech qachon bo‘sh bo‘lmaydi va xech qanday qayta ishlatishning ilojisi bo‘lmaganda ham qandaydur harajatlar bo‘ladi. Lekin qayta ishlatish sarflarining kamayishi qayta ishlangan kod miqdorining ko‘payishidur. Kod chiziqlarining katta miqdori uchun o‘rnatilgan baxo tarqatilgan. Adaptatsiyani nazorat qiluvchi ko‘paytiruvchisi (AAM) kodni qayta ishlatishga kerak bo‘lgan qo‘shimcha kuchni nazorat qilish uchun kerak.

Qisqacha qilib aytganda AAM bu 3 ta komponentniing yig‘indisi:

1. Adaptatsiya komponenti (AAF deb nomlanadi) qayta ishlatiladigan kodga o‘zgartirish kiritishga ketadigan sarf-xarajatlarni aks ettiradi. Adaptatsiya komponenti o‘z ichiga sub komponentlarni oladi, ular dizayn kod va itegratsiya o‘zgarishlarini xisobga oladi.

2. Tushunish komponenti (SU deb nomlanadi) qayta ishlatiladigan kodni tushunishga va muhandisni kod bilan do‘stono munosabatiga ketadigan sarf-xarajat. SU notuzilmaviy kod majmuasi uchun 50 dan abyektga yonaltirilgan yaxshi yozilgan kod uchun 10 gacha bo‘ladi.

3. Baxolash (AA deb nomlangan) omili qayta ishlatish haqida qaror qabul qilish uchun ketadigan harakatlar. Shunday qilib, kod yana qayta ishlatilishi haqida qaror qabul qilish uchun taxlil bajarilishi lozim va bu narxga AA deb kiritilgan. Talab qilingan harakatni taxlil miqdoriga qarab AA 1 dan 8 gacha o‘zgaradi.

Agar kodni adaptatsiyasi avtomatik tarizda bajarilishi munkun bo‘lsa bu kerak bo‘lgan harakatni kamaytiradi. Shuning uchun siz avtomatik adaptatsiya qilingan kod % ni (B) va ASLOC ni moslashtirib baxolashni nazorat qilishingiz mumkun. Yakuniy formula quyidagicha bo‘ladi:

ESLOC=ASLOC*(1- AT100)*AAM

ESLOC xisoblanishi bilanoq siz baxolashni standart formulasini qo‘llab talab qilingan to‘liq harakatni hisoblashingiz mumkun bo‘ladi, bu yerda o‘lcham parametri=ESLOC. Keyin buni xisoblab chiqqan avtomatik yaratilgan kodni birlashtirishga ketgan harakatga qo‘shib to‘liq harakatni xisoblaysiz.

Post arxitektura qatlami

Bu model COCOMO II modellari ichidan eng ko‘p to‘xtalgani. U tizim yaroqli bo‘lishi uchun boshlang‘ich arxitekturaviy loyihada qo‘llaniladi. Shuning uchun subtizim strukturasi ma‘lum bo‘ladi. So‘ng tizimning har bir qismi uchun baholash amalga oshirilsa ham bo‘laveradi.

Baholashni tashkil etish post- architecture level da ishlab chiqilgan bo‘lib, u dastlabki loyihalash bosqichida ishlatiladigan formula bilan ayni bir xil:

$$PM=A*size^{B}*M$$

Jarayondagi bu bosqichda loyiha o‘lchami aniqlikda baholanadi. Quyidagi uchta parametrlarsan foydalanish orqali kod o‘lchamini baholashni bajarish mumkin:

1. Keltirilgan yangi kod (SLOC) qatorlarini umumiy miqdorini baholash.

2. ESLOC kod qatorlarining ekvivalent soniga asoslangan reuse qiymatlarini baholash.

3. Tizim talablarini o‘zgarishi sababli o‘zgartirilgan kod qatorlari miqdorini baholash.

KSLOC da umumiy kod miqdorini o‘lchash uchun ushbu parametrlar qiymatlarini qo‘shish mumkin. Baholashdagi oxirgi komponenta – o‘zgartirilgan kod qatorlar miqdori- doim o‘zgaruvchi dastur talablarini aks ettiradi. Bu qo‘shimcha kodni qayta ishlashni boshqaradi.

B eksponentasi loyiha murakkabligi bosqichlariga bog‘langan. Loyihalar qanchalik murakkab bo‘lsa, tizim qiymatining oshishiga ta‘siri shunchalik yuqori bo‘ladi. B eksponenta qiymati shuningdek 5 ta faktorga asoslangan, 2.12- rasmda ko‘rsatilgandek, bu faktorlar 0 dan 5 gacha bo‘lgan olti –o‘rinli jadvalda baholanadi va “0” “extra high” “5” “ very low” . B ni hisoblash uchun baholashlar qo‘shilishi kerak, ularni 100 gacha bo‘lib, natijani 1.01 ga qo‘shish kerak.

Misol uchun Biror bir tashkilot biror sohada loyiha olib boryapti, ya‘ni kichik bir tajrib ustida. Loyiha mijozi jarayonni aniqlamadi. Yangi bir birlashma bu tizimni amalga oshirish uchun birgalikda muhokamaga qo‘yadi. Tashkilot yaqinda dasturni yaxshilash jarayonini o‘rniga qo‘yadi. Eksponentani hisoblashda foydalaniladigan qiymatlar:

1. Precedentedness. Bu tashkilot uchun yangi loyiha.

2. Development flexibility. Jarayonda mijoz zaruriyati bo‘lmaydi,shuning uchun tashqi yuklangan o‘zgarishlar bo‘ladi.

Mezon	Tushuntirish
Precedentedness	Bu turdagi loyiha bilan tashkilotning avvalgi tajribasini aks ettiradi. Juda past bo‘lsa xech qanday tajriba yo‘qligini bildiradi: juda balant bo‘lsa korxona bu soxa bilan to‘liq tanish ekanligini bildiradi.
Rivojlanishning moslashuvchanligi	Rivojlanish jarayonidagi moslashuvchanlik darajasini aks ettiradi. Juda pasti jarayon ishlatilayotganini bildiradi, juda yuqorisi mijoz umumiy maqsadlar qo‘yganini bildiradi.
Arxitektura va havf hatar qarori	Havf-hatarni baxolash taxlilini aks ettiradi. Juda pasti to‘liqmas taxlilni bildiradi, juda yuqorisi havf-hatar darajasini to‘liq taxlilini bildiradi.
Jamoa birdamligi	Jamoa bir-birini tushunish darajasi va birga ishlashini aks ettiradi. Juda past juda og‘ir muloqatlarni bildiradi juda balandi aloqada muammosi yoq

	integratsionallangan va samarador jamoani bildiradi.
Jarayon yetukligi	Tashkilot saviyasini aks ettiradi. Buni xisoblash uchun SMM anketa so‘ro‘vi ishlatiladi, baxoni bilish uchun 5 SMM ni ayrib tashlash kerak.

2.12- rasm. Post-architecture modelida eksponentani hisoblashda foydalaniladiga faktorlar jadvali

3. Architecture/risk resolution (arxitekturaviy qaror). Eng quyi 5 da baholangan keltirilgan. U yerda risk analizlari keltirilmagan.

4. Team cohesion(jamoa mustahkamligi). Nominal 3 da baholangan.

5. Process maturity (Oxiriga yetgan jarayon). Nominal 3 deya baholangan.

Bu qiymatlarning summasi 16. Ulni 100 gacha bo‘lib, natijani 1.01 ga qo‘shgandagi B eksponentaning o‘zgartirilgandagi qiymati 1.17.

Umumiy natijaviy baholash 17 ta mahsulot to‘plamidan foydalanish orqali takomillashtirilgan va tashkiliy attributlar (cost drivers) dastlabki loyiha modelida foydalanilgan 7 ta atributlarga qaraganda ko‘proqdir. Bu attributlar uchun qiymatlarni baholash mumkin.

2.12- rasmda narxni yurgizuvchi attributlar baholash natijasiga qanday ta‘sir ko‘rsatishi ko‘rsatilgan. Dastlabki misolda eksponenta uchun 1.17 qiymat olingan edi va RELY, CPLX, STOR, TOOL, va SCED lar loyihadagi cost driver kalitlardir. Barcha cost driverlar 1 bo‘lgan nominal qiymatga ega, shunung uchun ular natijani hisoblashga ta‘sir ko‘rsatmaydi.

2.13- rasmda cost driverlarning maksimum va minimum qiymati va ular natijani baholashga qanday ta‘sir ko‘rsatishi keltirilgan. Yana bu yerda cost driverlar natijaviy baholashni boshqarishi uchun yuqori qiymatlar ham ko‘rsatilgan ya‘ni u yerda boshlang‘ich baholash uch martaga qaraganda ko‘proq, eng quyi qiymatlar baholash haqiqiysidan taxminan 1-3 ga qisqartirilgan.

Namuna baxosi	**1.17**
Tizim o‘lchami(qayta ishlatish va talablar	128,000 DSI

o‘zgarishi omillari)	
Dastlabki COCOMO bepul	730 odam kuchi oyga nisbatan
Ishonchlilik	Juda yuqori, ko‘paytiruvchi=1.39
Murakkablilik	Juda yuqori, ko‘paytiruvchi=1.3
Xotira cheklovlari	Yuqori, ko‘paytiruvchi=1.21
Foydalanish vositasi	Past, ko‘paytiruvchi=1.12
Jadval	Jadal, ko‘paytiruvchi=1.29
COCOMO ni belgilangan baxolash	2,306 odam kuchi oyga nisbatan
Ishonchlilik	Juda past, ko‘paytiruvchi=0,75
Murakkablilik	Juda past, ko‘paytiruvchi=0,75
Xotira cheklovlari	O‘rtacha, ko‘paytiruvchi=1
Foydalanish vositasi	Juda yuqori, ko‘paytiruvchi=0,72
Jadval	O‘rtacha, ko‘paytiruvchi=1
COCOMO ni belgilangan baxolash	295 odam kuchi oyga nisbatan

2.13– rasm. Narxni yurgizuvchi atrebutlar baxolash natijasiga ta‘siri

2.5.2. Loyiha davom etish muddati va xodimlar

Loyihaning umumiy qiymatini baholashda loyiha menejerlari dasturni rivojlantirish qancha muddatda olib borilishini va loyihada xodimlar qachon ishga muhtoj bo‘lishlarini baholashlari shart. Tashkilotlar qisqaroq jadvallarni talab qilishyapti, shuning uchun ularning mahsulotlari bozorga ularning raqobatdoshlaridan oldin olib chiqilishi kerak.

COCOMO modeli loyihani tugatish uchun talab qilingan calendar vaqtni baholovchi formulani o‘z ichiga oladi:

$$TDEV=3*(PM)^{(0.33+0.2*(B-1.01))}$$

TDEV- bu loyiha uchun nominal jadval,calendar oylarida, loyiha jadvaliga bog‘liq biror ko‘paytiruvchi.

PM- COCOMO modelida hisoblangan natija.

B-murakkab bog‘langan eksponenta, muhokama uchun 2.5.2 seksiyaga qarang.

Agar B = 1.17 va PM = 60 so‘ng

$$TDEV=3*(60)^{0.36}=13 \text{ oy}$$

Shunday qilib,nominal proyekt jadvali COCOMO modeli tomonidan oldindan aytib o‘tilgan va bu jadval har doim ham bir xil bo‘lmaydigan proyekt rejasini talab qilgan. Bu shunday talabki,dasturni ta‘minlash uchun yaratilgan bo‘lishi mumkin. Agar jadval qisqacha bayon qilinsa, bu proyekt uchun talab qilinadigan natijalarni orttiradi. Bu SCED ko‘paytiruvchisi tomonidan hisobga kiritilgan.

Loyiha TDEV ni 13 oy deb baholagan, ammo asosiy jadval 11 oyni talab qilgan. Bu taqdimotda jadvalni qisqartirish (siqish) taxminan 25%. SCED ko‘paytiruvchisi uchun qiymatlardan foydalanish Boehm jamoasi tomonidan chiqarilgan, jadvalni qisqartirishning natijaviy ko‘paytiruvchisi-1.43. Shuning uchun asosiy natija talab qilinadi, agar ushbu tezlashtirilgan jadvalga nominal jadvalga ko‘ra dasturni ta‘minlash uchun talab qilingan natijaga qaraganda 50% ko‘proq duch kelinsa.

Bu yerda biror loyiha ustida ishlayotgan bir qancha odamlar o‘rtasida murakkab bog‘liqlik bor, natija loyihaga bag‘ishlanadi va loyiha yetkazib berish jadvali. Agar 4 ta odam bir loyihani 13 oyda bajarsa (i.e., 52 person-months of effort), birdan ko‘proq shaxsni qo‘shish orqali bu loyihani 11 oyda tugatish mumkin (55 person-months of effort).

Shu sababdan, odamlarni qo‘shish jamoa a‘zolarining mavjudlik mahsuldorligin qisqartirib yuboradi, shuning uchun qo‘shilgan natijani ko‘paytirish

bir shaxsga qaraganda ozroq. Jamoa a‘zolari boshqa odamlar orqali tizim qismlari o‘rtasida interfeysni aniqlash va xabar qilishda ko‘proq vaqt sarflashadi. Xodimlar sonini qo‘shish (misol uchun) shuningdek loyihani ikki baravar qisqartirish vaqt davomiyligini anglatmaydi. Myers (1989) bu jadvalni tezlashish muammolarini muhokama qiladi. U muammolarni hal etadigan loyihalarni taklif qiladi, agar ular ishni tugatish uchun yetarli vaqtsiz dasturni rivojlantirishga harakat qilishsa.

Umumiy natijani bo‘lish orqali loyiha uchun talab qilingan odamlar sonini osonlik bilan baholab bo‘lmaydi. Odatda,kichik sonli odamlar boshlang‘ich loyihani topshirish uchun loyihani boshlash kerak bo‘ladi. Loyihani tez sur’atda qurishda xodim loyiha jadvali bilan o‘zaro munoosabatda bo‘lishi ko‘rsatiladi. Loyiha menejerlari esa loyihani bajarish davomida ishga juda ko‘p xodimlarni qo‘shishdan chetga qochishlari kerak.

Kalit tushunchalar

- Tizimni baholash shunchaki baholangan qiymatlarga bog‘liq emas va foyda rivojlangan kompaniyadan talab qilinadi.Tashkiliy faktorlar shuni anglatadiki,narx (baho) ko‘tarilgan riskni o‘rnini qoplash uchun oshiriladi yoki raqobatbardosh ustunlikni qo‘lga kiritish uchun kamaytiriladi.
- Dastur ko‘pincha shartnomani qo‘lga kiritish uchun baholanadi va amaliy tizim keyinchalik baholangan narxni qanoatlantirishi uchun o‘zgartiriladi.
- Loyihani jadvallash loyiha rejasi qismining turli grafik taqdimotlarini yaratishni taqozo etadi.
- Loyiha ustuni harakatlar to‘plami yoki kutilayotgan natija. Har bir ustunda progressning rasmiy xabari menejmentga taqdim etilishi kerak. Ta‘minlovchi esa ish mahsuloti bo‘lib, u loyiha xaridori tomonidan ta‘minlanadi.
- XP rejalsh o‘yini loyihani rejalashdagi butun jamoani taqozo etadi. Reja incremental rivojlantirilgan va agar muammo vujudga kelsa, u shunday o‘zgartiriladiki, amaliy dastur qisqartiriladi.
- Dastur uchun baholash texnikalari tajribaga asoslangan bo‘lishi mumkin, menejerlar natijaga baho berishadi,yoki algoritmik natija boshqa baholangan loyiha parametrlaridan hisoblanadi.

- COCOMO II qiymatlash modeli o‘sgan algoritmik qiymat modeli bo‘lib, u loyiha,mahsulot,apparat qiemni oladi va xodimlar tayyorlagan narx bahosi hisobiga attributlanadi.

Qo‘shimcha o‘qish uchun

Software Cost Estimation with COCOMO II. Bu COCOMO II modelidagi eng so‘nggi kitob. Unda modelning tugatilgan ta‘riflari ko‘p misollar bilan keltirilgan va bu kitob dasturlarni o‘z ichiga oladi.O‘qish uchun to‘liq tafsilotlar berilgan. (B. Boehm et al., Prentice Hall,2000.).

(B. Boehm et al., Prentice Hall,2000.). Amaliy maqola bo‘lib, unda loyihani baholashning amaliy qiyinchiliklari muhokama qilingan. (P. Armour, Comm.ACM, 45 (11), November 2002.).

Agile Estimating and Planning. Bu kitob keng qamrovli rejalashga asoslangan. Shunday qilib u o‘z ichiga yaxshi, umumiy loyihalashni rejalashtirish maqolalarini oladi.

“Achievements and Challenges in Cocomo-based Software Resource Estimation” u maqola COCOMO II modeli tarixini taqdim etadi, bu modelning variantlarini muhokama qiladi. (B. W. Boehm and R. Valeridi, IEEE Software, 25 (5), September/October 2008.) http://dx.doi.org/10.1109/MS.2008.133.

Nazorat savollari:

1. Qanday holatlarda biror kompaniya dastur narxini baholashga qo‘shimcha maqbul foydaga qaraganda dastur tizimi uchun ko‘p yuqori narx qo‘yishi mumkin?

2. Nega loyihani rejalash jarayoni takrorlanuvchi va nega reje dastur loyihasi davomida davomiy ko‘zdan kechirilishi kerakligini tushuntirib bering!

3. Biror dastur loyihasi rejasidagi har bir qismlarning maqsadini qisqacha turshuntiring!

4. Narxni baholash xatarli, baholashdan qat’i nazar texnika foydalanadi.Qisqara olgan narxni baholashdagi risk(tavakkal)da to‘rtta yo‘lni taklif qiling!

5 \ 2.14-rasm vazifalar sonini,ularning davomiylik vaqtini birlashtiradi. Loyiha jadvalini ko‘rsatuvchi diagramma chizing!

6\ 2.14-rasm loyiha harakatlari uchun vazifalar davomiylik vaqtini ko‘rsatadi. Taxminga ko‘ra, jiddiy muvafaqqiyatsizlik sodir bo‘ladi va 10 kun qo‘yilishini o‘rniga T5 vazifa uchun 40 kun olinadi. Loyiha qanday qayta tashkil qilinganini ko‘rsatuvchi diagramma chizing!

7. XP rejalash o‘yini maqolalarni amalga oshirishdagi rejalash tushunchasiga asoslangan bo‘lib tizim talablarini taqdim etadi. Dastur bajarilganda ushbu yaqinlashish bilan ehtimoldagi muammolarni tushuntiring!

8. Dastur menejeri xavfsiz-kritikal dastur tizimini rivojlantirishda javobgar bo‘lib, bu dastur kasallikdan azob chekayotgan bemorlarni davolash uchun ximiya terapiya mashinasini nazorat qilish uchun loyihalangan bo‘lsin. Bu tizim mashina ichiga o‘rnatilgan va 256 Mbayt xotirali protsessorni ishga tushirish kerak. Bu mashina bemordagi kasallik tafsilotlarini egallash uchun bemor MB(MA‘lumotlar ba‘zasi) bilan aloqa qiladi, davolashdan so‘ng ta‘minlangan radiatsion doza qayd etiladi va MB dagi boshqa davolashlar ko‘rib chiqiladi.

COCOMO metodi bu tizimni rivojlantirish uchun talab qilingan natijalarni baholashda qo‘llaniladi va 26 person-month qiymati hisoblanadi. Barcha narx ko‘paytiruvchilari bu baholashni bajarishdagi 1 ga yig‘iladi.

Nega bu baholash loyihani,xodimlarni va mahsulotlarni tashkiliy faktorlarni hisobga kiritish uchun o‘zgartirilishi kerakligini tushuntiring! Boshlang‘ich COCOMO baholashga ta‘sir ko‘rsatuvchi 4 ta factor taklif qiling. Nega bu faktorlar qo‘shilishini tasdiqlang!

Masala	Doimiylik(kunlarda)	Qo‘shuvchilar
T1	10	
T2	15	T1
T3	10	T1, T2
T4	20	
T5	10	
T6	15	T3, T4
T7	20	T3
T8	35	T7
T9	15	T6
T10	5	T5, T9
T11	10	T9

T12	20	T10
T13	35	T3, T4
T14	10	T8, T9
T15	20	T2, T14
T16	10	T15

2.14- rasm. Masalalarni rejalashtirish

9. Ba'zi dasturiy loyihalarning kod qismlari juda katta ya‘ni millionlab belgilar yozishni o‘z ichiga oladi. Nima uchun COCOMO modelini juda katta tatbiq qilinganidi, yoki yuqori baholashda model qanday harakat qilish tushuntiring.

10. Talablar noaniq ekanligini bilib va ular mijozlar tomonidan talab keyinchalik o‘zgarishlar uchun ko‘proq haq to‘lashlari mumkin, u bizga dasturiy mahsulot uchun past narxda shartnoma tuzishlari uchun ma‘qulmi?

3. SIFAT BOSHQARUVI

Dastur sifati bilan bog‘liq muammolar 1-katta dastur tizimining rivojlanishi bilan 1960-yilda kashf qilingan va 20-asr davomida dastur muhandisligini bezovta qilish davom etgan. Dastlabki dasturlar pardan va ishonchli bo‘lmagan shuningdek ta‘mirlashda ham qiyin bo‘lgan. Bu holatdan qoniqmaslik natijasida, yuqori darajadagi dastur boshqaruvini rasmiy uslublarda foydalanishga sabab bo‘lgan. Mana shu yuqori darajadagi dastur texnikalari yangi dastur texnologiyalari va yuqori dasturlarning umumiy darajada o‘sishiga sabab bo‘ldi.

Dastur tizimlari uchun yuqori darajadagi boshqaruv dastursini 3 ta asosiy munosabati mavjud:

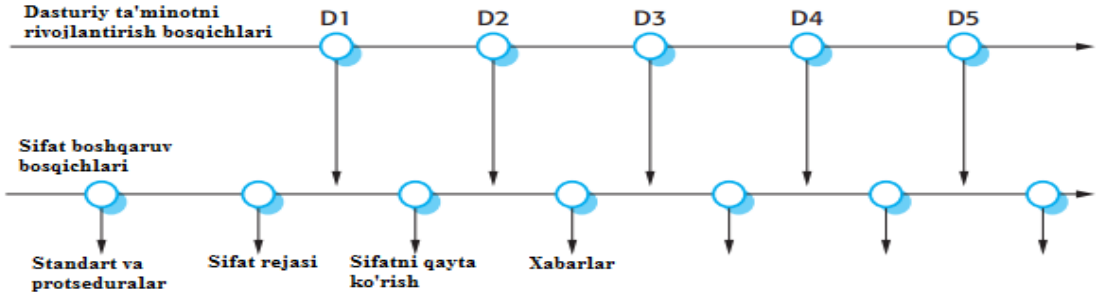
1. Yuqori darajadagi boshqaruv, tashkiliy jarayonlar va standartlarning qobig‘iga asos solinishi bilan aloqador hisoblanadi ya‘niki, yuqori darajadagi dasturga sabab bo‘ladi. Ma‘nosi shuki, yuqori boshqaruv jamoalarni rivojlangan jarayonlar dastursidan foydalanishi aniqlash uchun majburiyatni o‘z zimmasiga olishlari kerak.

2. Yuqori darajadagi boshqaruv asosiy jarayonlar tekshiruvlar arizalarni o‘z ichiga oladi. Bu rejalashtirilgan jarayonlar ketma-ket sodir bo‘ladi va ishonch hosil qiladi. Loyixani ishlab chiqarish hajmi standartlar bilan talabga javob beradi va loyihaga tegishli bo‘ladi.

3. Yuqori darajadagi boshqaruv loyiha uchun yuqori darajadagi rejani bajarish bilan ham aniqlanadi. Yuqori darajadagi reja loyiha uchun yuqori maqsadlarni mo‘ljallashi kerak, shuningdek qaysi jarayonlar va standartlardan foydalanishni aniqlash kerak.

“Yuqori darajadagi ishonch” va “Yuqori darajadagi boshqaruv” ishlab chiqarish sanoatida keng foydalaniladi. “Yuqori darajadagi ishonch” jarayonlar va standartlarning ta‘rifi hisoblanadi shuningdek u ishlab chiqarish jarayonida yuqori darajadagi jarayonlarni tanishtirishda va yuqori darajadagi maxsulotlarga sabab bop’ladi.

Dastur sanoatida turli kompaniyalar va sanoat qurishlari yuqori darajadagi kafolat va boshqaruvlarni turli yo‘llarda taqdim etishadi. Ba‘zan yuqori darajadagi kafolat ishlab chiqarish ta‘rifi jarayonlar va standartlaring soddalashtirilgan shakli hisoblanadi va dasturni yuqori darajali qo‘lga kiritish maqsad qilionadi. Boshqa tomondan yuqori kafolat barcha tasdiq va yaroqliligini o‘z ichiga oladi av maxsulot rivojlangan jamoalar orqali qo‘ldan qo‘lga o‘tgandan keyin murojaat qilinadi. “Yuqori darajadagi boshqaruv” esa sanoat dastursida keng foydalanilmaydi.



3.1- rasm. Sifat boshqaruvi va dasturiy ta‘minot rivojlanishi

Sifatni boshqarish dasturiy ta‘minot yaratish jarayonida xech narsaga bog‘liq bo‘lmagan tekshiruvni ta‘minlaydi. Sifatni boshqarish jarayoni loyiha natijalarini tashkiliy standart va maqsadlarga mos kelishini tekshiradi (3.1-rasm). Sifatni kafolarlash(SK) jamoasi yaratuvchilar jamoasida alohida bo‘lishi kerak, ular dastruriy ta‘minotni obyektiv tekshirish kerak. Bunda ular dasturiy ta‘minotni sifati holati haqida uni yaratish jarayonidan jarayon bosimida bo‘lmagan holda o‘z fikrini bildirin boradi.

Aslida sifatni boshqarish jamoasi birorta yaratuvchi guruhi bilan bog‘liq bo‘lmasligi va butun tashkilot miqyosida sifatni boshqarishga javobgar bo‘lishi kerak. Ular mustaqil bo‘lib loyiha boshqaruvchisidan yuqori bo‘lgan raxbaryatga xisobot berishlari kerak. buning sababi shundaki loyiha boshqaruvchilari loyihani byudjet va grafigiga rioya qilishi shart. Agar muammo paydo bo‘lsa ular jadvalga mos kelishi uchun maxsulot sifatidan kechishlari mumkin. Sifatni boshqarish bo‘yicha mustaqil jamoa sifatning tashkiliy maqsadlari byudjet yoki vaqt

yetmasligi xisobiga putur yetmasligini kafolatlaydi. Kichik tashkilotlarda buning ilojisi yo‘q Sifatni boshqarish va dasturiy ta‘minotni yaratish vazifalarini bir shaxs amalga oshirishi kerak.

Sifatni rejalashtirish bu loyiha uchun sifat rejasini ishlab chiqish jarayonidur. Sifat rejasi dasturiy ta‘minot sifatini va uni baxolash tartibini belgilash kerak. aniq tizim uchun “yuqori sifat”ni belgilaydi. Bunday qilinmasa muhandislar maxsulotning qaysi atrebutlari eng muhim sifat harakteristikalarini aks ettirishi haqaida har-xil va bazida qarama-qarshi takliflar bildirishi mumkin. Sifatni rasmiy rejalashtirilishi rivojlanish jarayonlarini rajalashtirishning ajralmas qismi bo‘ladi. Tezkor usullar sifatni boshqarishda kamroq rasmiy yondashuvdan foydalanadi.

Hamfri (1989) dasturiy ta‘minot boshqaruvi bo‘yicha yozgan kitobida sifatni rejalashtirish uchun tuzilma shaklini ishlatishni taklif qiladi. Buning ichida:

1. Maxsulotni kiritish. Maxsulot uning mo‘ljallangan bozori va maxsulot kutulayotgan sifatini tasnifi.

2. Maxsulotni rejalashtirish. Maxsulotni chiqarishning sanalari va tarqatish va xizmat ko‘rsatishni rejalashtirish.

3. Maxsulotni ishlab chiqarishni boshqarishda ishlatish kerak bo‘lgan ishlab chiqarish, xizmat ko‘rsatish jarayonlarini va standartlarni tasniflari.

4. Sifat maqsadlari. Sifat maqsadlari va maxsulotga bo‘lgan rejalar, buning ichida maxsulot sifatining asosiy atrebutlarini belgilash va asoslab berish.

5. Hav-hatar va havf-hatarlarni boshqarish. Maxsulot sifatiga ta‘sir etishi mumkin bo‘lgan asosiy havf-hatarlar va ularni bartaraf etish bo‘yicha choralar.

Ishlab chiqarilayotgan tizimning o‘lchami va turiga ko‘ra loyihani umumiy rejalashtirish doirasida ishlab chiqilgan sifat rejalari detallarida farqlanadi. Sifat rejalarini yozishda siz ularni qisqaroq yozishingiz kerak. Hujjat ortiqcha uzun bo‘lsa odamlar uni o‘qimaydi va sifat rejasi ko‘zda tutgan maqsadlarga erishilmaydi.

Bazi odamlar o‘ylaydiki dasturiy ta‘minot sifatiga erishish uchun yaratuvchi jamoalar tashkiliy standartlarga asoslangan yo‘naltiruvchi jarayonga rioya qilish yetarlik bo‘ladi. Ular buni shu bilan isbotlashadiki standartlar dasturiy

ta‘minotning amalda yaxshi bajarilishini aks ettiradi va bu amalyot yuqori sifatli maxsulot ishlab chiqarishga olib keladi. Amalda men o‘ylaymanki sifani boshqarish standartlari ko‘p qog‘ozboslikni talab qiladi.

Standart va jarayonlar muhum ahamiyatga ega lekin sifat bo‘yicha boshqaruvchilar shuningdek “sifat madaniyati” ni yaratishga yo‘nalgan bo‘lishlari kerak, bunda dasturiy ta‘minotni ishlab chiqishga javobgar har-bir javobgar shaxs maxsulot sifatining yuqori sifatiga erishishga harakat qilishi kerak. Ular jamoani shunga undash kerakki har bir a‘zo o‘z ishini sifatiga javobgarligini xis qilishi va sifatni yaxshilashga yangicha yondashuvlarni ishlab chiqishi kerak. Standart va protseduralar sifat boshqaruvining asoslari bo‘lsa ham tajribali sifat boshqaruvchilari standartlarda aks ettirilmaydigan dasturiy ta‘minot sifatini nojismoniy aspektlari borligini tan olishadi (elegant, o‘qishga oson va boshqalar). Ular sifatning nojismoniy aspektlarini rivojlantirishga qiziqqan odamlarni qo‘llashi va jamoaning hamma azosining o‘zini fropessional tutishiga chaqirishi kerak.

Sifatni rasmiy boshqarilishi ayniqsa bir necha yil mobaynida katta, uzoq muddat ishlaydigan tizimlatni yaratuvchi jamoalar uchun muhim. Sifat hujjati bu loyihaning har bir ost guruhi uchun yozilgan. Bu yozuvlar muhim vazifalar sedan chiqmaganligi va bitta guruh boshqa guruhlar tomonidan bajarilgan ishlar haqida noto‘g‘ri fikirga bormasligini tekshiradi. Sifat hujjatlari shuningdek tizimning hizmat ko‘rsatish muddati davomida muloqat vostasi bo‘ladi. Tizimni rivojlanishiga javobgar guruh yaratuvchi jamoasi tomonidan bajarilgan sinov va tekshiruvlarni kuzatishga imkon beradi.

Kichik tizimlar uchun sifatni boshqarish muhum bo‘lsa ham kam rasmiy bo‘lgan yondashuv qabul qilinishi mumkin. Ko‘p hujjat talab qilinmaydi chunki yaratuvchilarning katta bo‘lmagan jamoasi norasmiy muhitda muloqat qilishi mumkin. Kichik tizimlar yaratishda sifatning asosiy muammosi sifat madaniyatini yaratish va jamoaning hamma a‘zolari dasturiy ta‘minotning sifatiga bo‘lishli yondashuvda bo‘lishlarini ta‘minlashdir.

3.1. Dasturiy ta‘minot sifati

Sifat boshqaruvining asosiy tamoyillari ishlab chiqarilayotgan maxsulotlar sifatini yaxshilash uchun ishlab chiqarish saboati tomonidan o‘rnatilgan. Shu bilan birga ular “sifat” ta‘rifini rivojlantirishgan, u buyumning batafsil texnik harakteristikasiga (Crosby, 1979) va bardoshlik tushunchasiga asoslangan edi. Bunda asosiy taklif shunda ediki maxsulotlar to‘liq tasniflanib ishlab chiqilgan maxsulotni spetsifikatsiyasiga ko‘ra tekshirish tartiblarini o‘rnatish mumkin edi. Albatta maxsulot har doim ham spetsifikatsiyasiga to‘liq javob bermaydi, shuning uchun qabul qilna veradi. Agar maxsulot “deyarli to‘g‘ri” bo‘lsa u yaxshi deb olinadi.

Dasturiy ta‘minot sifati ishlab chiqarish bilan bevosita taqqaslash mumkin emas. Raqamli tizimlarga nisbatan tolerantlik g‘oyasini qo‘llab bo‘lmaydi va quyidagi sabablarga ko‘ra dasturiy taminot tizimi o‘z spetsifikatsiyasiga mos kelganligi haqida obyektiv xulosaga kelish mumkin emas:

1. Talablarni ishlab chiqish haqida 4 bobda gapirilgan edi, unda aytilganki dasturiy ta‘minotning to‘liq va bir ma‘noliy texnik talablarini yozish qiyin masala. Yaratuvchi va mijozlar dasturiy ta‘minotga qo‘yilgan talablarni har-xil tushunishlari munkin va dasturiy ta‘minot o‘z spetsifikatsiyasiga mos kelishi to‘g‘risida bitta to‘xtamga kelishi qiyin.

2. Texnik talablar odatda qiziqqan shaxslarning bir nechta sinfdan berilgan talablarni o‘z ichiga oladi. Bu talablar ilojsiz murosadir va qiziqqan shaxslarning hamma guruhini talablarini o‘z ichiga olmaydi. Tablari inobatga olinmagan shaxslar tizim kelishilgan talablarni amalga oshirsa ham sifatini past deb xisoblashadi.

3. Bazi sifat xususiyatlarini (masalan tuzatish mumkinligini) bevosita o‘lchash mumkin emas. Shunday qilib ular bir ma‘noli usulda aniqlanmaydi. O‘lchash muammolarini men 3.4 bo‘limda muhokama qilganman.

Shu muammolarni deb dasturiy ta‘minot sifatini baxolash subyektib jarayon, bunda sifat boshqaruv jamoasi kerakli sifat darajasiga erishilganligi haqida o‘z

fikrini bildiradi. Sifatni boshqarish jamoasi dasturiy ta‘minot belgilangan maqsadga mos kelishini ko‘rib chiqadi. bunda tizim xususiyatlari haqida savollarga javob berish kerak bo‘ladi. Masalan:

1. Yaratish jarayonida dasturlash va hujjatlar yuritilishi standartlar asosida bajarilganmi?

2. Dasturiy ta‘minot kwraklilik tarzda tekshirildimi?

3. Dasturiy ta‘minot ishlatilishga berilish uchun ishonchliligi yetarlimi?

4. Dasturiy ta‘minotning ishlashi yaxshimi?

5. Haqiqatan ham dasturiy ta‘minot foydalanishga tayyormi?

6. Dasturiy ta‘minotni tushunish va tuzilishi yaxshimi?

Saqlash	Tushunuvchanlik	Mobillik
Xavfsizlik	Tekshiruvchanlik	Ishlatishga qulayligi
Mustaxkamlik	Mosalshuvchanlik	Ko‘p marotaba ishlatish mumkinligi
Moslashuvchanlik	Modullik	Samaralilik
Chidamlilik	Murakkablik	O‘rganuvchanlik

3.2- rasm. Dasturiy ta‘minot sifatini atrebutlari

Dasturiy ta‘minot sifatini boshqaruvi tizim uning talablari bo‘yicha tekshirishni o‘z zimmasiga oladi. Bu sinovlar natijasiga ko‘ra talab etilgan fuksionallik bajarilishi haqida xulosa chiqariladi. Shuning uchun sifat kafolatlovchi jamoa ishlab chiqilgan testlarni ko‘rib chiqishi va testlar kerakli tarzda bajarilganligini tekshirish uchun test xisobotlarini o‘rganib chiqadi. bazi tashkilotlarda tizimni testdan o‘tkazishga sifat boshqaruvi javobgar lekin bazida bu ishni alohida tizimni testlash guruhi bajaradi.

Dasturiy ta‘minot tizimining subyektiv sifati ko‘p miqdorda uning funksional bo‘lmagan xususiyatlariga asoslanadi. Bu foydalanuvchi amaliy tajribasida aks ettiriladi – dastruriy ta‘minotning funksionalligi kutilgan darajada bo‘lmasa faydalanuvchilar o‘z ishlarini bajarish uchun boshqa yo‘llarni topadilar.

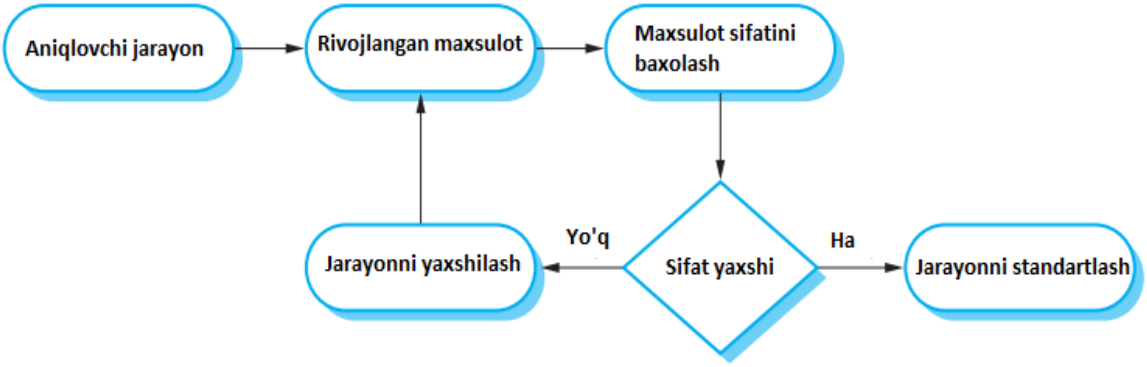
Lekin agar dasturiy ta‘minot ishonchsiz bo‘lsa yoki juda sekin ishlasa, ular o‘z maqsadlariga umuman yetisha olamaydi.

Shuning uchun dasturiy ta‘minot sifati faqat uning funksionallagi to‘g‘ri amalga oshirilganligi emas balki tizimning funksional bo‘lmagan xususiyatlariga ham bog‘liq. Boehm va boshqalar (1978) dasturiy ta‘minot sifatining 15 ta muhim belgisini taklif etti, ular 3.2- rasmda ko‘rsatib o‘tilgan. Bu belgilar dasturiy ta‘minotning ishonchliligi, qulayligi va ishlatish osonligi, samarasi va tuzatish mumkinligiga oiddir. Yuqorida aytganidek ishonchlilik belgilari odatda tizim sifatining eng muhum ko‘rsatkichlaridir. Lekin dasturiy ta‘minotning ishlashi ham juda muhim. Foydalanuvchilar juda sekin ishlaydigan dasturiy ta‘minotni rad etadilar.

Har qanday tizim uchun bu hamma belgilarini aptimallashtirishning ilojisi yo‘q-masalan ishonchlilikni oshirish natijasida ishchanlikni pasaytirish mumkin. Shuning uchun sifatni rejalashtirishda yaratilayotgan dasturiy ta‘minot uchun eng muhim sufat xususiyatlarini belgilash kerak. Samaradorlik muhim bo‘lganda bunga erishish uchun bishqa omillardan vos kechish kerak bo‘ladi. Siz bu haqda sifat rejasida aytib o‘tsangiz muhandislar bunga erishish uchun hamkorli qilishlari kerak. Reajada shuningdek sifatni baxolash jarayoni belgilanishi kerak. Maxsulotda tuzatish mumkinligi yiko ishonchlilik kabi sifat borligini tekshirish bo‘yicha usul kelishuv asosida ishlab chiqilgan bo‘lishi kerak.

Dasturit ta‘minot sifatini boshqarish asosida dasturiy ta‘minot sifati bevosita dasturiy ta‘minotni ishlab chiqarish jarayonining sifati bilan bog‘liq ekanligi haqidagi g‘oya yotadi. Bu yana sanoat tizimlarida kelib chiqadi, ularda maxsulot sifati ishlab chiqarish jarayoni bilan chambar-chas bog‘liq.

Ishlab chiqarish jarayoni ichiga jarayonda qatnashadigan mashinalarni shakillantirish, o‘rnatish va ishga tushirish kiradi. Mashinalar to‘g‘ri ishlasa maxsulot sifati ham yaxshi bo‘ladi. Siz maxsulot sifatini tekshirasiz va sizga kerak bo‘lgan sifat darajasiga yetmaguncha jarayonga o‘zgartirish kiritaverasiz. 24.3-rasmda maxsulot sifatiga yetishishda jarayonga asoslangan yondashuv ko‘rsatilgan.



3.3- rasm. Jarayon bajarilishiga bog‘liq bo‘lgan sifat

Ishlab chiqarishda jarayon va maxsulot sifati o‘rtasida aniq bog‘liqlik bor chunki jarayonni standartlashtirish va nazorat qilish nisbatan oson bo‘ladi. Ishlab chiqarish tizimlari kalibrovka qilingandan keyin yuqori sifatli maxsulotni chiqarish uchun ularni boshqarish mumkin. Lekin dasturiy ta‘minot ishlab chiqilmaydi u yaratiladi. Shuning uchun dasturiy ta‘minot yaratishda jarayon sifati va maxsulot sifati o‘rtasidagi munosabatlar murakkobroqdur. Dasturiy ta‘minotni yaratis-mexanik emas balki ijodiy jarayondir va unda alohida ko‘nikma va tajtibaning ta‘siri muhimdir. Ilovaning yangiligi yoki maxsulotni ertaroq chiqarish niyatida tijorat bosimi kabi tashqi omillar ham ishlatilayotgan jarayonga bog‘liq bo‘lmagan holda maxsulot sifatiga ta‘sir etadi.

Shubxasiz yaratishda foyadalanilgan jarayonlar dastiruy ta‘minot sifatiga katta ta‘sir o‘tkazadi va yaxshi jarayonlar gumonsiz dasturiy ta‘minotning yaxshi sifatini ta‘minlaydi. Jarayonni sifatini boshqarish va mukammallashtirish natijasida yaratulayotgan dasturiy ta‘minotning defektlarini kamaytirishi mumkin. Lekin dasturiy ta‘minotning uzoq vaqt mobaynida foydalanmasdan turib uning tuzatish mumkinligi kabi sifat ko‘rsatkichlarini baxolash qiyin. Bundan kelib chiqadiki jarayon xususiyatlari sifat belgilariga qanday ta‘sir etishini aytish qiyin. Undan tashqari jarayonni standartlashtirish ijodiy patensialga to‘sqinlik qilishi mumkin va natijada dasturiy ta‘minot sifatiga zarar yetishi mumkin.

3.2. Dasturiy ta‘minot standartlari

Dasturiy ta‘minot standartlari dasturiy ta‘minot sifatini boshqarishda muhim ahamiyatga ega. Oldin aytganimdek sifatni kafolatlashning asosiy qismi – dasturiy ta‘,minot yoki dasturiy maxsulotni yaratish jarayoniga oid standartlarni aniqlash yoki tanlashdir. Bunday sifatni kafolatlovchi jarayon qismi sifatida shiningdek bu standartlarini qo‘llab quvvatlovchi vosita va usullar tanlanishu mumkin. Standartlar tanlangandan keyin standartlardan foydalanish va ularni kuzatilishini nazorat qiluvchi jarayonlar belgilanishi kerak.

Dasturiy ta‘minot standartlari 3 sabaga ko‘ra muhimdir:

1. Standartlar tashkilot uchun muhum bo‘lgan bilimdonlikni aks ettiradi. Ular tashkilot uchun eng yaxshi yoki eng mos keluvchi amalyot haqidagi bilimga asoslangan. Bu bilim katta miqdordagi urinish va hatoliklar natijasida paydo bo‘ladi. Standartni yaratilishi tashkilotga bu tajribadan foydalanishni va avvalgi hatoliklarni qaytarmaslikka yordam beradi.

2. Satndartlar “sifat” qismlarni boshqarishda ma‘nosini belgilashga asos bo‘ladi. Oldin aytganimdek dasturiy ta‘minot sifati subyektiv bo‘ladi va standartlardan foydalanib siz sifatning kerak darajasiga erishilganligini asoslab berasiz. Labatta bunda dasturiy ta‘minotning ishonchliligi, ishchanligi faydalanish qulaylugi va soddaligi bo‘yicha foydalanuvchilar talablarni aks ettiruvchi normalarni belgilashga bo‘g‘liq.

3. Bitta odam boshlagan ishni boshqalar uzluksiz davom ettirishida standartlar yordam beradi. Bitta tashkilot doirasidagi hamma muhandislar bir xil usuldan foydalanishini stanadartlar kafolatlaydi. Demak yangi ishni boshlashda o‘qitishga ketadigan harajatlar kamayadi.

Dasturiy ta‘minot sifatini boshqarishda belgilanishi va ishlatilishi mumkin bo‘lgan dasturlashning ikkita bog‘langan:

*1. Maxsulot standartlari.*ular yaratilayotgan dasturiy maxsulotga oiddir. Ular ichiga talablar hujjarlarining tuzulmasi kabi hujjatlar standartlari, obyektlar

sinfining belgilovchi izoxining standart sarlovhasi kabi hujjatlashtirish standartlari va dasturlash tili qanday ishlatilishini belgilovcha kod standartlari kiradi.

2. Jarayon standartlari. Ular dasturiy ta‘minot yaratishda qanday jarayonlar bo‘lishini belgilaydi. Ula o‘z ichiga yaratishning eng yaxshi amalyotini olishi kerak. Jarayon standartlari ichiga spetsifikatsiya, dizayn va ratifikatsiya jarayonlarini,jarayonlarni qo‘llab quvvatlab truvchi vositalarni tasniflari va bu jarayonlar vaqtida yozilishi shart bo‘lgan hujjatlar ta‘sniflari kirishi mumkin.

Maxsulot standartlari	Jarayon standartlari
Loyihani taxlil qilish shakli	Loyiha taxlilini boshqarish
Talablar hujjatining tuzilmasi	Tizimni qurish uchun yangi kodni kiritish
Usul sarlavhasi formati	Varyantlarni chiqarish jarayoni
Java dasturlash uslubi	Loyiha rejasini maqulllash jarayoni
Loyiha rejasining formati	O‘zgartirishni nazorat qilish jarayoni
Maxsulotni o‘zgartirishga so‘ro‘vning shakli	Sinovdan o‘tkazish jarayoni

3.4- rasm. Maxsulot va jarayon standartlari

Standartlar maxsulot sifatini muhimligini shakillantirishi kerak. Vaqt va ketadigan kuch nuqtai nazaridan qimmat bo‘lgan standartlarni aniqlashda ma‘no yo‘q. maxsulot shunday ishlab chiqilishi kerakki, ularni ro‘yhatga olish va ishlatish foyda keltirishi kerak hamda standart jarayonlariga maxsulot yaratishda standartlar qo‘llanishini aniqlovchi jarayonlar bpo‘lishi kerak.

Dasturlashning haqaro standartlarini yaratish uzoq muddatli jarayondur, bunda standartga qiziqqanlar uchrashadilar, izoxlash uchun eskiz tayyorlaydilar va nihoyat standart haqida bir to‘xtamga kelishadi. U.S, DoD, ANSI, BSI, NATO va IEEE kabi milliy va halqaro tashkilotlar standartlarni ishlab chiqish bilan shug‘ullanadilar. Turli loyihalarda foydalanish mumkin bo‘lgan umumiy standartlar. NATO va boshqa himoya tashkilotlari kabi o‘byektlar dasturiy

ta‘minot tashkilotlari bilan tuzilgan shartnomalarda o‘zlarini standartlaridan foydalanishni talab qilishlari mumkin.

Milliy va halqaro standartlar ishlab chiqishida belgilar chizilishi, dasturiy ta‘minot talablarini olish va yozish protseduralari, sifatni kafolatlash protseduralari, dasturiy ta‘minotni tekshirish va jarayonlarni klassifikatsiyalash Java va C++ kabi dasturlash tillari, texnik atamalari, izoxlar ko‘zda tutilgan (IEEE, 2003). IEC 61508 (IEC, 1998) kabi maxsus standartlar himoyasi va havfsizligi muhim bo‘lgan tizimlar uchun yaratilgan.

Tashlikot uchun standartlarni yaratuvchi sifatni boshqaruvi tashkilotning bu standartlarini milliy va halqaro standartlariga asoslash kerak. Xalqaro standartlarni boshlash nuqtasi sifatida ishlatib sifatni kafolatlovchi jamoa standatrlar bo‘yicha yo‘riqnomani tuzish kerak. Unda tashkilotga ketak bo‘lgan standartlar berish kerak. Bunday yo‘riqnomaga kiritilishi mumkin bo‘lgan standartlat misollari 3.4 rasmda berilgan.

Dasturiy ta‘minot yaratuvchilari bazida standartlar da ortiqcha ko‘p ko‘rsatma berilgan deb xisoblaydi va dasturiy ta‘minotning texnik faoliyati uchun ahamyatsiz deb hisoblaydilar. Loyiha standartlari ko‘p hujjat va ish borishini talab etganda shunday bo‘ladi. Odatda muhandislar standartlar ummumiy holda kerakligini tan olishsada ular ko‘pincha loyihaning ularga tegishli qishmida standartlar unchalik kerak bo‘lmaganligiga jiddiy asoslar topib berishadi.

Me’yorlarni o‘rnatuvchi sifat boshqaruvchilari qarshilikni ozaytirish uchun va standartlarga munosabatlarni o‘zgartirish uchun quyidagilarni bajarishlari kerak:

1. *Maxsulot standartlarini tanlashga dasturiy ta‘minot yaratuvchilarini chorlang.* Yaratuvchilar standartlar nimaga tanlanganini tushunsa, shu standartlarga rioya qilishi extimoliko‘payadi. Eng yaxshi varyanti standartlar hujjatida nafaqat standart bayon etiladi balki nega standartlashning bu turi tanlanganligi tushuntirilishi kerak.

2. *Standartlarni har doim ko‘rib chiqish va o‘zgaruvchan texnologiyalarga mos xolda o‘zgartirish kerak.* Standartlar yaratish jarayonida muhim o‘zinga ega va ular odatda tashkilotning standartlar yo‘riqnomasida saqlanadi. Ketadigan sarf

va muhokama qilish kerakligi uchun ularni o‘zgartirishga ko‘pincha hohish bo‘lmaydi. Standartlar yo‘riqnomasi muhim lekin o‘zgarayotgan sharoit va texnologiyalarni aks ettirish uchun ularni o‘zgartirish kerak.

3. *Satandartlari qo‘llovchi dasturiy vositalarini ta‘minlang.* Yaratuvchilar standartklarda nazarda tutilgan zerikarli qo‘l ishini dasturiy vosita yordamida bajarish yo‘llarini topadilar. Vositadan foydalanish mumkin bo‘lsa dasturiy ta‘minot standartni bajarish uchun kam harakat ketadi. Masalan hujjatli standartlari matn protsessori stillari yordamida bajarishi mumkin.

Dasturiy ta‘minotning har-xil turlari yaratishning har-xil jarayonini talab qiladi, shuning uchun standartlar moslashishi kerak bo‘ladi. Ishlash usuli, loyiha yoki loyiha guruhiga mos kelmasa uni tanlashdan fiyda yo‘q. Har bir loyiha boshqaruvchisi jarayon standartlarini o‘zing sharoitiga hos xolda o‘zgartirishga vakolatiga ega bo‘lishi kerak. Lekin bu o‘zgartirishlar maxsulot sifatini pasayishiga olib kelmasligi kerak. Chunki bu tashkilotning mujozlari bilan munosabatiga ta‘sir etishui va loyiha sarflari ko‘payishiga olib kelishi mumkin.

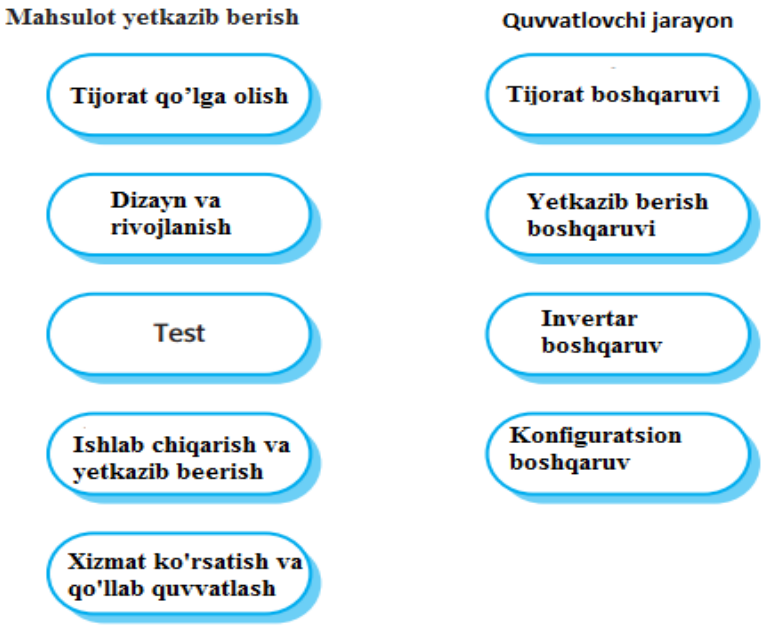
Loyiha boshqaruvchisi va sifat boshqaruvchisi loyihani boshidan sifatni etibor bilan rejalashtirish evaziga mos kelmaydigan standartlar moammosini oldini olishlari mumkin. Ular tashkiliy standartlarni qaysinisi o‘zgartirishsiz ishlatilishi, qaysilari o‘zgartilishi va qaysilari ishlatilmasligi haqida qaror qabul qilishlari kerak. Yangi standartlar mijoz hohishiga yoki loyiha talablariga mos tarzda yaratilishi kerak. Masalan oldingi loyihalarda ishlatilmagan formal talablari uchun standartlar yaratilishi mukin.

3.2.1. ISO 9001 standartlar tuzilishi

Barcha sohalarda boshqaruv tizimlarining sifatini rivojlantirishda foydalanish mumkin bo‘lgan xalqaro standartlar majmui bor bo‘lib,ular ISO9000 deb nomlanadi.ISO9000 ishlab chiqarish tashkilotlaridan tortib xizmat tarmoqlarida bir qator yo‘nalishlarda qo‘llaniladi. ISO 9001,ushbu majmualar ichida umumiysi bo‘lib,dizayn,ishlab chiqarish,mahsulotni saqlash,shuningdek

dasturiy ta‘minotni o‘z ichiga olgan tashkilotlar uchun amal qiladi.ISO9001 dastlab 1987 tilda ishlab chiqilgan ,so‘nggi ko‘rinishi esa 2008 yilda yaratilgan.

ISO 9001 o‘zi dasturiy ta‘minot ishlab chiqish uchun standart emas, lekin dasturiy ta'minot standartlarni ishlab chiqish uchun asos. Bu esa uning umumiy tamoyillarini belgilaydi. Umumiy sifat jarayonlarini tasvirlaydi hamda jarayonlar va standartlar belgilangan bo‘lishi kerak. Bu sifatli hujjatlashtirilishi lozim.



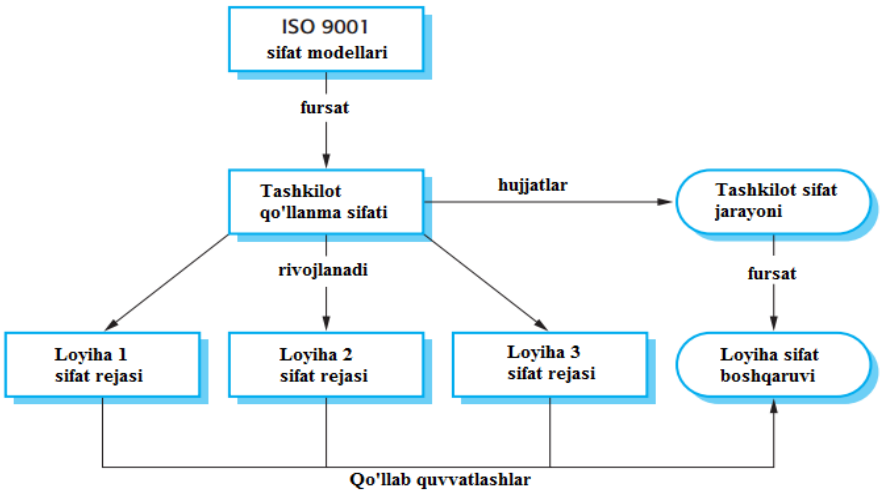
3.5- rasm. ISO 9001 standartining yadro jarayonlari

Agar tashkiliy tuzilish ISO9001 bilan bog‘liq bo‘lsa, uning asosiyhujjat jarayoniga bog‘liq bo‘lishi kerak. Bu shuningdek namoyish etilgan hujjatlarni aniqlashi va saqlab qolishi kerak. Kompaniya sifatli qo‘llanmasi tegishli jarayonlar va mavjud jarayonlarda to‘plangan va muhofaza qilingan ma‘lumotlarni tasvirlashi kerak. ISO9001 kompaniyada ishlatilishi kerak bo‘lgan maxsus jarayonlarni aniqlamaydi. Sifatli standart dasturlar dasturiy ta‘minot uchun mos deb belgilangan.

Qog‘ozbozligi bo‘lmagan kichik kompaniya ISO9001 mos bo‘lishi mumkun. Ba‘zi kompaniyalari batafsil qattiq sifat jarayonlariga ega. boshqalar esa minimal qo‘shimcha hujjatlar bilan , juda kam rasmiy bo‘lishi mumkin. ISO 9001, tashkiliy sifatli qo‘llanmalar va shaxs o‘rtasidagi munosabatlar loyiha sifat rejalari

3.6 shaklda ko‘rsatilgan.Ushbu diagramma Inca tomonidan olingan modelda asosiy sifatli dasturiy ta‘minotda foydalanisih mumkunligi tushuntiriladi.Dielbler va Bamford keyinchalik ISO9001,2000 dasturiy ta‘minot kompaniyalarida qo‘llanilishini tushuntiradi.(3.6 rasm)

Ba‘zi mijozlar ISO9001 sertifikatlangan bo‘lishini talab qilishadi.shundan so‘ng mijozlar dasturiy ta‘minot sifatli boshqaruvga ega ekanligiga amin bo‘lishadi.Ba‘zilar sertifikatlangan ISO 9001 dasturiy ta‘minoti sertifikatlanmaganidan yaxshiroq bo‘ladi deb o‘ylashadi.Bu juda muhum emas.ISO9001 standarti kompaniyada sifat jarayoni boshqaruvi borligiga va uni doimiy boshqarib turishga mo‘ljallangan.ISO9001 kompaniyasi eng yaxshi dasturiy ta‘minot ishlatishiga yoki ularning jarayonlari yuqori sifatli dasturiy ta‘minot boshqarishiga kafolat yo‘q.Misol uchun,kompaniya test standartlarini barcha usullar ni kamida bir martada aniqlab oladi.Afsuski,bu standart sinovi chala dasturiy ta‘minotlarida ko‘rilishi mumkun.



3.6- rasm. ISO 9001 va sifat boshqaruvi

Dasturiy ta‘minotning bazi mijozlari ta‘minotchilar standartlashtirish halqaro tashkiloti ISO 9001 tomonidan tasdiqlangan bo‘lishini talab qilishadi. Mijozlar bunda dasturiy ta‘minot yaratuvchi kompaniyaning sifatni boshqarish tizimi maqullanganligiga ishonch hosil qiladi. Mustaqil akriditatsiya tashkilotlarisifatni boshqarish jarayonlari va rasmiylashtirish jarayonlarini

tekshiradilar va bu jarayonlar ISO 9001 da belgilangan hamma sohalarni qamraganligi haqidaqaror qabul qiladilar. Agar shunday bo‘lsa ular kompanayaning sifat jarayonlari ISO 9001 tomonidan o‘rnatilgan standartga mos kelishini tasdiqlaydilar.

Bazi odamlar o‘ylaydiki ISO 9001 sertifikati berilgan tashkilotlarning dasturiy ta‘minotining sifati bunday sertifikatini olmagan tashkilotlardan yuqori bo‘lishini kaflatlaydi. Bu har doim ham to‘g‘ri emas ISO 9001 standarti tashkilotda sifatni boshqarish protseduralab borligini va u ularga rioya etishini kafolatlaydi holos. ISO 9001 kompaniya dasturiy ta‘minotning yaratishni eng zo‘r usullaridan foydalanishini va ularning jarayonlari yuori sifatli dasturiy ta‘minotga olib kelishini kafolatlamaydi.

Masalan kompaniya hamma usullarni obyekti qabrab oluvchi tekshirish standartini xech bo‘lmasa bir marta e’lon qilishi kerak. Afsuski bu standart usulning har-xil parametrini inobatga olamagan testlovchi dasturga uchrashi mumkin. Tekshiruv protseduralari kuzatilib bajarilgan test natijalari saqlanganda kompaniya ISO 9001 tomonidan sertifikatlangan bo‘lishi mukin. ISO 9001 sifat standartga to‘g‘ri kelishini aniqlaydi va dasturiy ta‘minot sifati foydalanuvchi tomonidan qanday sinalganiga etibor bermaydi.

Rasmiylashtirishga etibor berilmagan va asosiy etibor yaratilayotgan kodga qaratilgan tezkor usullar ISO 9001 dagi rasmiy sifat jarayonlari bilan kam bog‘langan. Bu yondashuvlarni moslashtirish bo‘yicha bir qatir ishlar bajarilgan (Stalhane va Hanssen, 2008) lekin tezkor rivojlantirish tarafdorlari standartga to‘g‘ri kelishlarini tepadagi byukrakratlar tomonidan o‘rganilishiga qarshi edilar. Shuning uchun tezkor usullardan foydalanuvchi tashkilotlar ISO 9001 sertifikatini olishga oshiqmaydilar.

3.3. Sharx va tekshiruvlar

Sharx va tekshiruvlar loyiha sifatini tekshiruvchi SK amallaridur. Bunda dasturiy ta‘minot jarayonlarining rasmiylashtirilishi va xisobotlari hato va

kamchiliklarini aniqlash uchun tekshiriladi va sifat standartlariga rioya qilingani ko‘riladi. 8 va 15 bobda aytilgandek sharx va tekshiruvlar dasturiy ta‘minot tekshirish va ratifikatsiyalash umumiy jarayonining yaqinidagi tekshirish dasturi sifatida ishlatiladi.

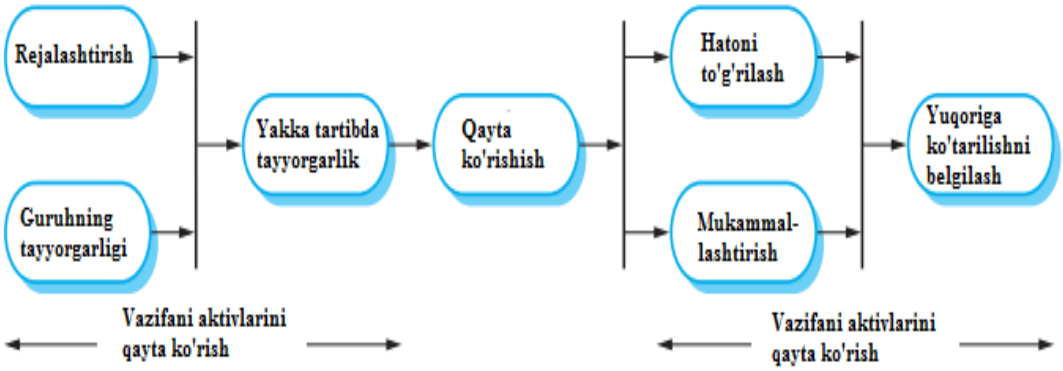
Sharx vaqtida bir guruh odam dasturiy ta‘minot va u bilan bog‘liq hujjatlarni tekshirib muammo va standartga tushmagan joylarni topadi. Sharx jamoasi tizim yoki loyiha sifati darajasi haqida axborot tayyorlaydi. Loyiha boshqaruvchilari bu baxolardan rejalashtirish qarorlarini qabul qilishda foydalanishi mumkin va jarayonni rivojlantirish uchun manba ajratishi mumkin.

Sifatrni sharxlash dasturiy ta‘minot yaratish vaqtida taqdim etilgan hujjatlarga asoslanadi. Shuningdek dasturiy ta‘minotga qo‘yilgan texnik talablar loyiha dizayni yoki kodi jarayon modellari, sinov rejalari, konfiguratsiyani boshqarish protseduralar, jarayon standartlari va foydalanuvchi yo‘riqnomasi ko‘rilishi mumkin. Sharx hujjatlar yoki kodningketma-ketligini va tugaganligini tekshirishi va sifat standartlariga rioya qilinganligiga amin bo‘lishi kerak.

Lekin sharxlar nafaqat standartga to‘g‘ri kelishini tekshiradilar ular shuningdek dasturiy ta‘minot yoki loyiha hujjatlaridagi mummo va kamchiliklarni aniqlash uchun yordam beradi. Sharx xulosalari sifatni boshqarish jarayonini qismi sifatida rasmiylashtirilishi shart. Ammolar aniqlangan bo‘lsa tekshiruvchilar izoxlarini dasturiy ta‘minot muallifi yoki hatolarni bartaraf etuvchilarga berish kerak.

Sharx va tekshiruvlarrning maqsadi jamoa a‘zolarini baxolash uchun emas balki dasturiy ta‘minot sifatini yaxshilashdur. Komponentlarni sinovdan o‘tkazishga nisbatan sharxlash hatolikni aniqlashni umumiyroq jarayonidur. Odamlar tomonidan qilingan hatolar ilojsiz butun dasturlash jamoasiga ko‘rsatiladi. Loyihani boshqaruvchilari jamoa a‘zolarini ko‘rib chiqishda alohida muammolarga etiborli bo‘lishlari kerak. Ular ish madaniyatini shunday rivojlantirishi kerakki hatolar topilganda aybini yuklamasdan uni bartaraf etishga yordam berish afzalroq hisoblanish kerak.

Sifat sharxi raxbaryatga dasturiy ta‘minot rivojlanishi haqida axborot bersa ham, sifat sharxlari rivojlantirishni boshqarish sharxi degani emas. 23 bobda aytganimdek rivojlanish sharxlari dasturiy ta‘minotning rejalashtirilgan jadval bo‘yicha ketishini taqqoslaydi. Ularning asosiy havotiri dasturiy ta‘minot vaqtida va byudjetdan chiqmagan holda yetkazilishidur. Rivojlantirish sharxlari tashqi omillarni inobatga oladi va bunda o‘zgargan sharoitlar yaratilayotgan dasturiy ta‘minot kerak emasligini yoki keskin o‘zgartirilishini bildiradi.



3.7- rasm. Dasturiy ta‘minotni qayta ko‘rish jarayoni

Yuqori sifatli dasturiy ta‘minot yaratilgan loyihalar biznesdagi o‘zgarishlar yoki operatsion muhitni deb bekor qilinishi mumkin.

3.3.1. Qayta ko‘rish jarayoni

Sharxlar detallarida ko‘p o‘zgarishlar bo‘lsa ham, qayta ko‘rish jarayoni (3.7 rasm) odatda 3 fazadan iborat:

1. *Sharxdan oldingi amallar.* Bu sharx samarali bo‘lishi uchun muhim bo‘lgan avvalgi ammalar odatda sharxdan oldingi ammallar rejalashtirishni sharxlash va sharxlashga tayyorgarlikdan iboratdur. Sharxni rejalashtirishga sharxlovchi jamoani, sharx vaqti va joyini belgilash va ko‘riladigan hujjatlarni tarqatilishi kiradi. Sharxga tayyorlanish vaqtida jamoa ko‘riladigan dasturiy ta‘minotni qisqacha sharxni bajarish uchun uchrashish mumkin. Jamoaning alohida

a‘zolari dasturiy ta‘minot yoki hujjatlar va tegishli standartlarni o‘qishadi va tushunishga harakat ilishadi. Ular hat, kanchilik va standartga mos kelmagan joylarni mustaqil tarzda topishadi. Taqrizchilar sharxda qatnasha olmasa dasturi ta‘minotga yozma izohlar qoldirishi mumkin.

2. *Sharxlovchilar uchrashuvi.* Hujjat yoki dastur muallifi hujjtani sharxlavchi jamoa bilan kezib chiqishi kerak. Sharxlovning o‘zi nisbatan qisqa bo‘lishi kerak-keng ko‘p 2 soat. Jamoaning bir a‘zosi sharxlashni boshqazrish kerak, ikkinchisi sharxlovning hamma qarorlari va amallarini rasmiylashtirishi kerak. Sharxlov raisi hamma yozma izoxlar ko‘rib chiqilishi kerak. Sharxlov vaqtida kelishilgan izox va amllar hisobotini sharxlov raisi imzolashi kerak.

3. *Sharxlovdan keyingi amallar.* Sharxlov uchrashuvi tugagandan keyin unda aniqlangan muammolar bartaraf etilishi kerak. Bunda sifat standartlariga mos kelishi uchun dasturiy ta‘minot hatolari tog‘irlanadi yoki hujjatlar qayta yozialadi. Bazi paytlarda sifat sharxida aniqlangan muammolarni yechish uchun manbalarni ko‘paytirish maqsadida boshqaruv sharxini ham o‘tkazish talab qilinadi. O‘zgartirishlar kiritilgandan keyin sharxlov raisi sharxdagi hamma izoxlar hisobga olinganligini tekshirishi mumkin. Bazida keyingi sharxda oldingi sharxning hamma izohlari bo‘yicha o‘zgartirishlar kiritilganligini tekshiriladi.

Sharxlash jamoasida odatda asosiy taqrizchi deb tanlangan 3 yoki 4 odam bo‘ladi. Bitta qatnashchi muhin texnik qarorlarni qabul qilishga javobgarligini o‘ziga oluvchi katta loyihalovchi bo‘lishi kerak. Boz taqrizchi bog‘langan ost tizimlarning loyihalashtiruvchilarini sharxni yaxshilash maqsadida taklif etishi mumkin. Ular butun hujjatni tekshirishga qatnasha olmaydi lekin o‘zlarini ishi bilan bog‘liq joylarni etiborga olishadi. Alternativ sifatida sharxlov jamoasi hujjtani tarqatishi va loyiha qatnashchilarining keng spektridan yozma izox berishlarini so‘rashi mumkin. Loyihani rejasida o‘zgartirish kiritishni talab qiluvchu muammolar kutilmasa sharxlovga loyiha boshqaruvchisi taklif etilmaydi.

Yuqorida ko‘rilgan sharxlov jarayoni jamoa a‘zolari yig‘ilishi oson bo‘lgan holatlarga mo‘ljallangan. Lekin hozir loyiha guruhlari har-xil mamlakat va kontsenetlatga tarqalgan bo‘ladi va ularni bitta joyda to‘planishi amri mahol..

bunday hollarda sharxlov jarayonini bajarish uchun hujjatni muharrirlash vositasidan foydalanish mumkin. Ularning yordamida jamoa a‘zolari dasturiy ta‘minotning hujjat yoki dastlabki kodini izox bilan to‘ldirishlari mumkin. Bu izoxlar jamoaning boshqa a‘zolariga ko‘rinadi va ular bu izoxlarni maullashi yoki rad etishi mumkin. Taqrizchilar orasidafi qarama-qarshiliklarni bartaraf etishalri uchun telefon orqali muloqatlar o‘tkazilishi mumkin.

Dasturiy ta‘minotni tezkor yaratishda sharxlash jarayoni odatda rasmilashtirilmaydi. Masalan dasturiy ta‘minotning har bir iteratsiyasidan keyin sifat moammolari muhokama qilinishi mumkin bo‘lgan sharxlov bor (pring sharxlov). Keyingi bo‘limda muhokama qilganimdek favqulotda dasturlashda dasturlovchi muftlikda kod har-doim jamoaning boshqa a‘zosi tomonidan o‘rganilishi va tekshirilishi kafolatlanadi. Sifatning umumiy muammolari jamoaning har kungi uchrashuvlarida muhokamaqilinadi, lekin FD da odamlar kodni yaxshilash va o‘zgatirishni o‘z zimmasiga oladi. Tezkor yondashuv standartlar bilan boshqarilmaydi shuning uchun standartlarga mos keltirish muammosi tug‘ulmaydi.

Tezkor usullardagi sifatga oid rasmiy protseduralar yetishmovchiligi sifatni boshqarish bunday protseduradagi rivojlantirgan tashkilotlarda tezkor yondashuvlardan faydalanishlarda muammo paydo bo‘lishi mumkin. Sifatni sharxlash dastruriy ta‘minot rivojlanish tempini pasaytirishi mumkin va ular jarayon rivojlanishi boshqarish rejasi doirasida ishlatilishi maqsadga loyiqdur. Reja bo‘yicha boshqaruvchi jarayonda sharxlar rejalashtiriladi va ular bilan bir vaqtda boshqa ish bajarilishi mumkin. Tezkor yondashuvda etibor kodni rivojlantirishga qaratilgani uchun bu foyda keltirmaydi.

3.3.2. Dasturni tekshirish

Dasturni tekshirish-“expert baxolash”, bunda jamoa a‘zolari hamkorlikda yaratilayotgan dasturda hatoliklarni topihadi. 8 bobda aytganimdek tekshiruv dasturiy ta‘minotning tekshiruv va ratifikatsiya qismi bo‘lishi mumkin. Ular dastur

bajarilishini talab qilmagani uchun sinovdan o‘tkazishga qo‘shimcha bo‘lishi mumkin. Budegani tizimning to‘liq bo‘lmagan versiyalari hamda UML modellari kabi taqdimotlari tekshirilishi mumkin. Gilb va Graham (1993) tekshuruvlarning samarali foydalanish usullari sifatida tizim buzilishlarini ko‘rishni taklif etishgan. Tekshiruvlar testlardagi muammolarni aniqlashga va dasturni hatolarini samarasini yaxshilashga yordam beradi.

Dasturlarni tekshirishga jamoaning har-xil a‘zolari taklif etiladi va ular ehtiyotkorlik bilan dasturning dastlabki kodini tekshirib chiqishadi. Ular hato va muammolarni topishadi va tekshiruvchilar uchrashuvida tasniflab beradi. Bu hatolar mantiqiy hato noto‘g‘ri shartni ko‘rsatuvchi anomaliya, kodda qolib ketgan joylar bo‘lishi mumkin. Tekshiruv jamoasi model dizayni yoki dastur kodini tuzatilishi kerak bo‘lgan detal va asosiy anomaliya borligiga tekshiradi.

Tekshiruv paytida hatoliklarni topishni yengillashtirishni topish uchun umumiy dasturiy hatoliklarning nazorat ro‘yhati ishlatiladi. Bu nazorat ro‘yhati kotiblardagi misollarga yoki alohida amaliy sohada keng tarqalgan defektlarhaqidagi bilimlarga soslanishi mumkin. Dasturlashning har-xil tillari uchun siz alohida nazorat ro‘yhatini ishlatasiz, chunki har bir tilda o‘ziga hos hatoliklar bo‘ladi. Xamfri(1989) tekshiruvlarni har tomonlama muhokamasida tekshirishning nazorat ro‘yhatlarining ko‘p misolini keltirgan.

Tekshiruv jarayonida bajarilishi mumkin bo‘lgan tekshirishlar 3.8 rasmda ko‘rsatilgan. Gilb va Graham (1993) har-bir tashkilot o‘zining standart va usullariga asoslanga tekshiruvning nazorat ro‘yhatini ishlab chiqishni aytishgan. Bu nazorat ro‘yhatlari doim yangilanib turishi kerak chunki hatoliklarning yangi turlari paydo bo‘ladi. Nazoarat ro‘yhatidagi punkitlar komplatsiya vaqtida mumkin bo‘lga tekshiruvning har-xil darachasi borligi uchun dasturlash tiliga ko‘ra o‘zgaradi. Masalan Java kompilyatori funksiyalarda parametrlash soni to‘g‘riligini tekshiradi. C kompilyatori bu ishni bajarmaydi.

Tekshiruvlarni amalga oshirgan ko‘plab tashkilotlar ular hatolarni topishda samarali ekanligini bilishdi. Fagan(1986) habar qilishicha dasturdagi 60 % dan ko‘p hatoliklar norasmiy tekshiruvlarda aniqlanishi mumkin. Mills va boshqalar

(1987) to‘g‘rilik argumentlariga asoslangan tekshiruvga rasmiy yondashuv orqali dasturdagi 90% dan ko‘proq hatoliklar topilishi mumkin deb taxmin qilishdi. MCConnell (2004) taqqoslaganda testlashda 25 % hatoliklar topilgan tekshiruvda esa 60%. U shuningdek ko‘p misollar keltirgan bitta misolida expert baxolash kiritilganda unumdorlik 14% oshgan va dasturdagi hatoliklar 90% kamygan.

Tekshiruv yoki expert baxolashlarni samaradorligi haqida ko‘p ijodiy fikirlar bildirilganiga qaramay dasturiy ta‘minot yaratuvchilarining ko‘pi ulardan foydalanishni hohlamaydi. Dasturlashni sinovdat o‘tkazish tajribasiga ega dasturiy ta‘minot yaratuvchilari tekshiruvlar hatoni testlardan ko‘ra yaxshiroq topishini tan olgilari kelmaydi.

Boshqaruvchilar bunga shubha bilan qaraydilar chunki tekshiruvlar loyihalash va yaratishy vaqtida qo‘shimcha sarf harajatni talab qiladi. Ular dasturni sinovda o‘tkazishni ketadigan sarfdan ko‘payib ketishi mumkin deb qo‘rqishadi.

Kamchilik sinfi	Nazorat ro‘yahti
Vaqt kamchiliklari	Ularning qiymatlari barcha dastur o‘zgaruvchilarining boshlang‘ich holdan keyin ishlatiladimi? Barcha constantalar nomidan bachariladimi? O‘lchami massiv yoki o‘lcham-1 bo‘lgan massivlarga teng bo‘lishi kerakmi? Agar belgilar sart ko‘rinishida bo‘lsa, chegara ochiq ko‘rsatiladimi?
Nazorat kamchiliklari	Har bir shartli bayonotida uchun, holati to‘g'ri bo‘ladimi? Har bir halqa bekor qilinishi ma‘lummi? Murakkab bayonotlar to‘g‘ri bo‘la oladimi? Tashkiliy ish jadvalida barcha mumkin bo‘lgan holatlar bormi? Agar holat bayonotlarida keltirilgan har bir holatdan

	keyin, to‘htalish zarur bolsa bu xisobga olinganmidi?
Kiritish/chiqarish kamchiliklari	Barcha kirituvchi o‘zgaruvchilar ishlatilgan bo‘ladimi? Barcha chiquvchi o‘zgaruvchilar chiqishdan oldin qiymatga ega bo‘ladilarmi? Kutilmagan kiritishlar korrupsiyaga olib kelishi mumkinmi?
O‘zaro bog‘lanish kamchiliklari	Barcha funksiya va usullar parametrlarning to‘g‘ri sonini chaqira oladimi? Rasmiy va haqiqiy parametrlar mos turlardami? Parametrlar to‘g‘ri tartibdami? Agar qism hotirasi bo‘lajlarga kirolsa, ular shu xotira model tuzilishida bormi?
Boshqaruv saqlash kamchiliklari	Agar aloqa tuzilishida o‘zgarishlar bo‘lsa, barcha aloqalar to‘g‘ri tayinlanganmi? Agar dinamik hotiradan foydalangan bo‘lsa, hotira to‘g‘ri ajratilganmi? Kerak bo‘lmay qolgan joy bo‘shatiladimi?
Istesno boshqaruv kamchiliklari	Barcha kutilgan hatoliklar xisobga olinganmi?

3.8- rasm. Nazorat ro‘yhati

Tezkor jarayonlar tekshiruv yoki expert baxolashning resmiy jarayonlaridan kam foydalanishadi. Ular ko‘proq bir-birini kodini tekshiga ishonib qolishadi va har bir dasturchi o‘zini kodini tekshirishga ishonadilar. Favqulotda dasturlash amalyotchilari bunday dasturlashni tekshiruvni samarali o‘rnbosari deb bilishadi chunki haqiqatdan ham bunday tekshiruv jarayoni uzluksiz bajariladi. Iikita odam kodning har bir satrini ko‘rib chiqadi va qabul qilishdan oldin tekshiradi.

Juftlikda dasturlashda dastur haqida chuqur bilimga ega bo‘ladi shunki ikkila dasturchi yaratishni davom ettirish uchun bir-birini ishini tushunishi kerak. Boshqa

tekshiruv jarayonlarida bunday bilimga erishish qiyin va dasturchilar uftligi rasmiy tekshiruvda aniqlanishi qiyin bo‘lgan hatolarni topishi mumkin.

Lekin ikkalasi ham bir hil hatolikka yo‘l qo‘ysa o‘zaro kelishmovchilik chiqishi mumkin. Undan tashqari loyiha rivojlanishi to‘xtashini hohlamasdan juftliklar hato axtarishni rad etishi mumkin. Dasturlashni ichidagi odamlar tashqi tekshiruv chamoasi kabi obyektiv bo‘la olmaydi va ularning hatolarini topish qobilyati sihdagi yaqin munosabatlariga putir yetkazishi mumkin.

3.4. Dasturiy ta‘minot o‘lchamlari va metrikalari

Dasturiy ta‘minot o‘lchamlari deganda dasturiy ta‘minot tizimi yoki jarayonning komponentini xususiyati uchun sonli qiymat yoki profilni olish tushiniladi. Bu qiymatlar bir biri bilan va tashkilotda qabul qilingan standartlarga nisbatan taqqoslab, Siz dasturiy ta‘minot sifati xaqida xulosa chiqarishingiz yoki dasturiy ta‘minotning jarayonlari, vositalari va usullarining samaradorligini baxolashingiz mumkin.

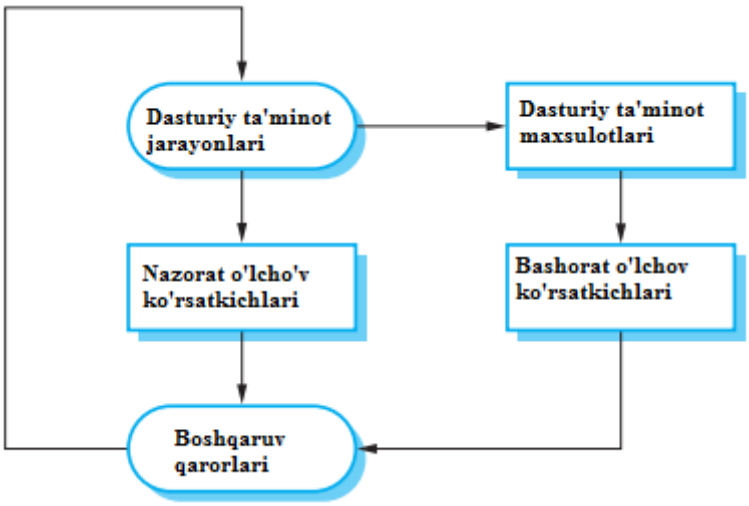
Masalan, aytaylik, tashkilot dasturiy ta‘minotni tekshiruvchi yangi instrumentni qo‘llamoqchi. Instrumentni qo‘llashdan avval Siz dasturiy ta‘minotda xozirgi vaqtda aniqlangan defektlar miqdorini yozib olasiz. Bu instrumentning samarasini tekshirish uchun asos bo‘ladi. Instrumentni ishlatgandan keyin vaqt o‘tib siz bu jarayonni qaytarasiz. Instrument ishlatilgandan buyon ko‘proq defekt topilgan bo‘lsa, bu dasturiy ta‘minotni ratifikatsiyalash jarayoniga yordam berayotgani xaqida aytishingiz mumkin.

Dasturiy ta‘minotni o‘lchashning uzoq muddatli maqsadi shundaki, dasturiy ta‘minot sifati xaqida xulosa chiqarishda sharxlar o‘rniga o‘lchashni qo‘llashdir. Dasturiy ta‘minotni o‘lchash yo‘li bilan tizim baxolanishi ideal bo‘lardi va tizim sifati bo‘yicha metrika va ularning qiymatlari diapazoni kiritilishi mumkin bo‘lardi. Dasturiy ta‘minot yetarli sifat darajasiga ega bo‘lsa, uni sharxsiz ma‘qullash mumkin bo‘ladi. Dasturiy ta‘minot o‘lchamlari birinchi o‘ringa qo‘yilganda bu soxada ancha muvaffaqiyatlarga erishish mumkin bo‘lardi. Lekin

biz xozirda bu ideal xolatdan ancha olismiz va sifatni avtomatik baxolash yaqin kelajakda xaqiqatga aylanishining xech qanday belgisi yo‘q.

Dasturiy ta‘minot metrikasi – ob’ektiv o‘lchanishi mumkin bo‘lgan dasturiy ta‘minot tizimining, tizim xujjatining yoki rivojlantirish jarayonining xususiyatidir. Metrikalar misollari ichiga maxsulotning kodi satrlar o‘lchamida; yozma matnning o‘qilishining qulayligini o‘lchovchi Tuman indeksi (Obstrel, 1962); dasturiy maxsulotdagi topilayotgan xatoliklar miqdori; odamga tizim komponentini yaratishga kerak bo‘lgan kunlar miqdori.

Dasturiy ta‘minot metrikalari nazorat metrikasi yoki bashoratlovchi metrika bo‘lishi mumkin. Nomlar jarayonni boshqarishda yordam bergandek, bashoratlovchi metrikalar dasturiy ta‘minotning xususiyatlarini oldindan aytishga yordam beradi. Nazorat metrikalari odatda dasturiy ta‘minotning jarayonlariga bog‘liq bo‘ladi. Jarayonlar yoki nazorat metrikasini misollari – defektlar xaqida xabar berishga kerak bo‘lgan o‘rtacha vaqt va xarakat. Bashoratlovchi metrikalar bevosita dasturiy ta‘minot bilan bog‘liq bo‘ladi va ba‘zida “maxsulot metrikalari” deb nomlanishadi. Bashoratlovchi metrikalar misollari - cyclomatic modulь murakkabligi (8 bobda muxokama qilingan), dasturdagi identifikatorlarning o‘rtacha uzunligi, va dizaynda ob’ekt sinflari bilan bog‘langan belgi va amallarning miqdori.



3.9- rasm. Bashorat qilish va nazorat qilish o‘lcho‘vlari

3.9- rasmda ko‘rsatilgandek, nazorat va bashoratlovchi metrikalari boshqaruv qarorini qabul qilishga ta‘sir qilishi mumkin. Boshqaruvchilar jarayonga o‘zgartirish kiritish kerakligi xaqida qaror qabul qilishda jarayon o‘lchamlaridan foydalanadi va dasturiy ta‘minotda o‘zgartirish bajarishga ketadigan xarakatni baxolash uchun bashoratlovchi metrikalarni ishlatadi. Bu bobda men asosan bashoratlovchi metrikalarni ko‘rib chiqaman, chunki dasturiy ta‘minot kodini taxlil qilishda ularning o‘rni beqiyosdir. Nazorat metrikalarini va ular jarayonni mukammallashtirishda qanday ishlatilishini 26 bobda ko‘rib chiqamiz.

Dasturiy ta‘minot tizimida o‘lchamlar bajarilishining ikkita yo‘li bor:

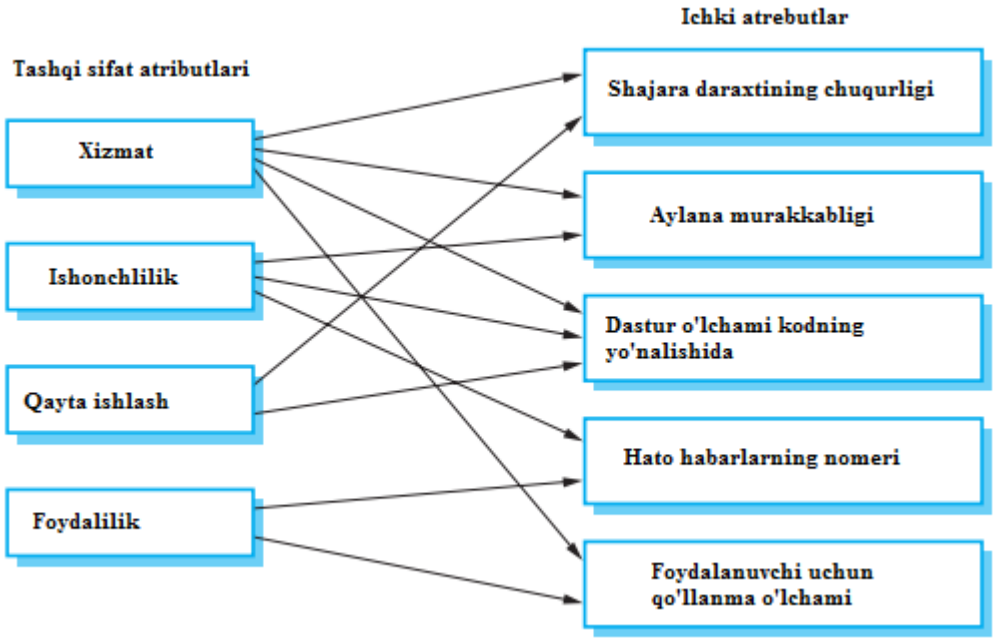
1. Tizim sifatini qiymatini belgilash – tizim komponentlarini xususiyatlarini o‘lchash va bu o‘lchamlarni qo‘shish, Siz tuzatish mumkinligi kabi xususiyatlarni baxolashingiz mumkin bo‘ladi.

2. Sifati standart bo‘lmagan tizim komponentlarini identifikatsiyalash. O‘lchamlar xususiyatlari me’yordan og‘ib turuvchi aloxida komponentlarni identifikatsiyalash mumkin. Masalan, siz komponentlarni o‘lchashiz mumkin va eng murakkabini topishiz mumkin. Ularda xato bo‘lishi mumkin, chunki murakkabligi uchun ularni tushinish qiyinroq.

Afsuski, 3.2- rasmda kursatilgan dasturiy ta‘minotining ko‘p sifat xususiyatlarni to‘g‘ridan to‘g‘ri o‘lchamlarini olish qiyin. Tuzatish mumkinligi, o‘qish osonligi, foydalanishning qulay va osonligi kabi sifat belgilari – yaratuvchi va foydalanuvchilarga tegishli bo‘lgan tashqi belgilardir. Ularda foydalanuvchi tajribasi va ma‘lumoti kabi sub’ektiv omillar ta‘sir etib ular ob’ektiv o‘lchanishining ilojisi yo‘q. Bu belgilar xaqida xulosa chiqarish uchun siz dasturiy ta‘minotning ichki belgilarini (o‘lchami, murakkabligi va x.) o‘lchashingiz va ular sizni xavotirga solayotgan sifat xususiyatlari bilan bog‘likligini ko‘rib chiqishingiz kerak.

3.10- rasmda dasturiy ta‘minotning sifatini ba‘zi tashqi belgilari va ular bilan bog‘liq bo‘lishi mumkin bo‘lgan ichki belgilar ko‘rsatilgan. Diagrammada tashqi va ichki belgilar o‘rtasida munosabatlar bo‘lishi mumkinligi xaqida gapirilgan,

lekin bu belgilar qanday bog‘langanligi aytilmagan.



3.10- rasm. Ichki va tashqi tasturiy ta‘minotlar o‘rtasidagi munosabatlar

Ichki belgining o‘lchami dasturiy ta‘minotning tashqi xususiyatiga ta‘sir qilishi mumkin bo‘lsa, uchta shart bajarilishi kerak (Kitchenham, 1990):

1. Ichki xususiyat aniq o‘lchanishi kerak. Bu xar doim xam to‘g‘ridan to‘g‘ri bajarilishi mumkin emas va bu o‘lchamni bajarish uchun maxsus vositalar kerak bo‘lishi mumkin.

2. O‘lchanishi mumkin bo‘lgan belgi va qiziqtiraetgan tashqi sifat belgisi o‘rtasida bog‘liqlik bo‘lishi kerak. Shunday qilib, sifat xususiyati qiymati o‘lchanishi mumkin bo‘lgan xususiyatning qiymatiga bog‘liq bo‘lishi kerak.

3. Tashqi va ichki xususiyat o‘rtasidagi bu munosabat tushinilishi, tasdiqlanishi va qaysidur ifoda yoki modelь yordamida ko‘rsatilishi kerak. Namunaviy ifodada modelni funktsionali yig‘ilgan ma‘lumotlarni taxliliga asoslanib, modelga kiritilishi kerak bo‘lgan parametrlarni belgilaydi va mavjud ma‘lumotlar asosida bu parametrlarni kalibrovka qiladi.

Komponent murakkabligi kabi dasturiy ta‘minotning ichki xususiyatlari dasturiy ta‘minotning dastlabki kodini taxlil qiluvchi dasturiy vositalar yordamida o‘lchanadi. Bu o‘lchamlarni bajarish uchun mavjud vositalar umumiy

foydalanishdadir. Dasturiy ta‘minot komponentining murakkabligi va ishlatishda kuzatilayotgan rad etishlar miqdori o‘rtasida bog‘liqlik borligi sezilsa xam, buni ob’ektiv namoyish etish qiyin. Bu fikrni isbotlash uchun siz ko‘p komponentlar ishdagi xatolari xaqidagi ma‘lumot va dastlabki kodni taxlil qilishga ruxsatga ega bo‘lishingiz kerak. Juda kam tashkilotlar o‘z dasturiy maxsuloti xaqida ma‘lumotlarni uzoq muddat davomida yig‘ishni o‘z bo‘yniga olgan, shuning uchun rad etishlar xaqidagi ma‘lumotlarni taxlil qilishni ilojisi yo‘q.

Hewlett Packard (Greydi, 1993), AT&T (Barnard i TSena, 1994), va Nokia (Kilpi, 2001) kabi ba‘zi katta tashkilotlar 1990-chi yillarda dasturlarni metrikalarini kiritishgan.

Ular o‘z maxsulot va jarayonlarini o‘lchovlarini o‘tkazishgan va ulardan sifatni boshqarish jarayonlarida foydalanishgan. Dastur metrikalari asosida tekshiruv va ratifikatsiya jarayonlari bajarilgan. Offen va Djeffri (1997) xamda Zal va Fenton (1997) metrikalar dasturini sanoatga kiritishni batafsil muxokama etishgan.

Sanoatda dasturiy ta‘minotni tizimli o‘lchash xaqida ozgina axborot bor. Ko‘p tashkilotlar xaqiqatda xam o‘z dasturiy ta‘minoti xaqida ma‘lumot to‘playdi, bunda o‘zgartirish kiritishga oid talablar va testlashda aniqlangan defektlar miqdori ko‘riladi. Lekin ular bu tizimli o‘lchamlarni nima uchun ishlatishlari noma‘lum: dasturiy maxsulot va jarayonlarni taqqoslash uchunmi yoki o‘zgartirishlarni dasturiy ta‘minot va vositalarga ta‘sirini baxolash uchunmi. Buning murakkabligining bir nechta sababi bor:

1. Tashkilotda metrikalar dasturini ishlatilishi qancha mablag‘ni saqlab qolishi noma‘lumligi. Metrikalardan foydalanmasdan dasturiy ta‘minotni mukammallashtirishda oxirgi yillarda katta o‘zgarishlarga erishildi, shuning uchun dasturiy ta‘minotni o‘lchash va baxolashga sarflashning asoslantirish qiyin.

2. Dasturiy ta‘minot metrikalari uchun xech qanday standart xamda o‘lchash va taxlil qilishning standartlashtirilgan jarayonlar yo‘q. Ko‘p tashkilotdar dasturni o‘lchamlarini standart va vositalarigacha olib borishni rad etishmoqda.

3. Ko‘p tashkilotlarda dasturiy ta‘minot jarayonlari standartlashtirilmagan va

yaxshi aniqlanmagan, yaxshi boshqarilmaydi. SHuningdek bitta tashkilotning ichida jarayon shunchali o‘zgaruvchanki, o‘lchamlar unda ishlatilishi axamiyatsiz bo‘ladi.

4. Dasturiy ta‘minotni o‘lchash va metrikalari soxasidagi tadqiqotlarning katta qismi asosan kod va rejalashtirishni boshqarish jarayonining metrikalariga e’tibor qaratgan. Lekin xozir ERP yoki GOST tizimlarini shakllantiruvchi, yoki tezkor usullaridan foydalangan dasturiy ta‘minot ko‘proq ishlab chiqilyapti. SHuning uchun biz oldingi tadqiqotlar dasturiy ta‘minotni yaratishning bu usullariga qo‘llanishi mumkinmi bilmaymiz.

5. O‘lchamni taqdim qilinishi yuqoridagi jarayonlarga qo‘shimcha ish bo‘ladi. Bu tezkor usullarning maqsadlariga mos kelmaydi, chunki ular dasturni rivojlanishi bilan bevosita bog‘liq bo‘lmagan jarayonlarni rad etadi. Tezkor usullarni ishlatayotgan tashkilotlar metrikalar dasturini qabul qilishi qiyin.

Dasturiy ta‘minotni o‘lchash va metrikalar – empirik dasturlashning asosidar (Endres va Rombach, 2003). Bu tadqiqot soxasida dasturiy ta‘minot tizimlaridagi tajribalar va real loyxalar xaqidagi ma‘lumotlar dasturlash usullari xaqida gipotezalarni shakllantirish va tasdiqlash uchun ishlatiladi. Bu soxada ishlovchi tadqiqotchilar aytishicha, dasturlash usullarining muximligi ular xaqiqatdan xam aytilgan foydani keltirishini isboti bo‘lsagina tasdiqlanadi.

Afsus, ob’ektiv o‘lchamlarni bajarish imkoni bo‘lsa xam va natijalarini ko‘rish mumkin bo‘lsa xam, qaror qabul qiluvchi shaxslar ularga ishonmasligi mumkin.

Qaror qabul qilinishiga ko‘proq yangiligi yoki amaliyotchilarda usullar qiziqish o‘yg‘onganlik darajasi kabi sub’ektiv omillar ta‘sir qiladi. Shuning uchun, men o‘ylashimcha, empirik dasturlashning dasturlash amaliyotiga ta‘sir etishiga xali ancha yillar bor.

3.4.1. Maxsulot o‘lchamlari

Maxsulot o‘lchamlari dastur tizimining ichki xarakterlarni o‘lchashda ishlatiladigan oldindan belgilanuvchi o‘lchamlar xisoblanadi. Maxsulot o‘lchamlari namunasi tizim o‘lchamini, kodlar qatori bilan o‘lchanadigan, yoki xar bir obyekt klassiga bog‘liq bo‘lgan metodlar sonini o‘z ichiga oladi. Baxtga qarshi, bo‘lim boshida tushuntirganimdek, dastur o‘lchami va sikl murakkabligi kabi oson o‘lchanishi mumkin bo‘lgan xarakterlar tushunarlilik va yashovchanlik kabi sifat xarakterlari bilan ishonchli bog‘liqlik yo‘q. Jarayonning rivojlantirilishi, ishlatilgan texnologiya va yaratilayotgan sistema turiga qarab (xarakterlarning bog‘liqligi) bog‘liqlik ham farq qiladi.

Maxsulot o‘lchamlari ikki sinfga bo‘linadi:

1. Dinamik o‘lchamlar, bajarilayotgan dasturdan xosil bo‘layotgan o‘lchashlardan yig‘iladi. Bu o‘lchamlar tizimni tekshirish davomida yig‘ilishi yoki Sistema ishga tushirilgandan keyin yig‘ilishi mumkin. Xatoliklar xabarlari soni yoki xisobni yakunlash uchun ketgan vaq misol bo‘lishi mumkin.

2. Statik o‘lchamlari, dizayn, dastur yoki dokumentatsiya kabi sistema namoyishchilarini o‘lchashlardan yig‘iladi. Statik o‘lchamlarga kodning o‘lchami va ismlarning o‘rtacha uzunligi misol bo‘ladi.

Bu o‘lcham turlari turli xil sifat atributlariga bog‘liq. Dinamik o‘lchamlar samaradorlikni va dasturning ishonchiligini belgilashda yordam beradi. Statik o‘lchamlar esa dastur tizimi va tizim komponentalarining murakkabligi, tushunarligi va yashovchanligini belgilashda yordam beradi.

Odatda dinamik o‘lchamlar va dasturning sifat xarakterlari o‘rtasida aniq bog‘liqlik bor. Aniq funksiya uchun talab etilgan bajarilish vaqtini o‘lchash va Sistema ishga tushishi uchun talab etiladigan vaqtni belgilash ancha oson. Bu to‘g‘ridan to‘g‘ri dastur samaradorligiga bog‘lanadi. Xuddi shundek, dastur xatoliklari soni va yuklanishi va dastur ishonchliligiga to‘g‘ridan to‘g‘ri bog‘liq bo‘lishi mumkin bo‘lgan xatolik turi, 15-bo‘limda muhokama qilingandek.

Muhokama qilganimdek, static o‘lchamlarda, 3.11 da ko‘rsatilgandek, sifat xarakterlari bilan bilvosita bog‘liqlik bor. Turli o‘lchamlarning ko‘plab soni taklif qilindi va ko‘plab tajribalar bu o‘lchamlar va murakkablik va yashovchanlik kabi xarakterlar o‘rtasidagi bog‘liqliqlikni keltirib chiqarishga va ma‘qulllashga xarakat qildi. Bu tajribalarning hech biri yakunlovchi bo‘lmadi lekin dastur o‘lchami va murakkablikni nazorat qilish ishonchlilik, dastur murakkabligi va yashovchanlikning eng ishonchli belgilovchilari bo‘lib tuyuldi.

3.11- rasmdagi o‘lchamlar xar qanday dasturga qo‘llanishi mumkin lekin ko‘proq obyetga yo‘nalganga(OY) xos o‘lchamlar ekanligi ham taklif etilgan. 3.12 Chidamber va Kemererning oltita obyektga yo‘naltirilgan o‘lcham to‘plamini(bazida CK to‘plami deb aytiladi) eslatadi(1994). Aslida bu 1990-yilning boshida taklif etilgan bo‘lsada, ular hali ham OY o‘lchamlari bo‘lib keng ishlatiladi.

Dastur o‘lchami	**Izoh**
Chaqiruvchi/Chaqiriluvchi	Chaqiruvchi boshqa bir funksiya yoki metodni (deylik X) chaqiruvchi funksiyalar sonining o‘lchami. Chaqiriluvchi X funksiya tomonidan chaqiriladigan funksiyalar soni. Chaqiruvchi uchun yuqori qiymat X dizaynning qolgan qismi bilan mustaxkam birlashgani va X ga o‘zgargani sabab ekanini anglatadi.
Kodning uzunligi	Bu dastur hajmining o‘lchami. Odatda Komponenta kodining katta hajmi murakkabroq va xatolikka moyilroq bo‘ladi. Kod uzunligi komponentada xatolikka moyilliklarni aniqlashning eng ishonchli o‘lchami deb ko‘rsatilgan.
Siklsimon murakkablik	Bu dastur murakkabligini nazorat qilishning o‘lchami. Bu murakkablikning nazorati dastur tushunarliliga bog‘liq bo‘lishimumkin. Men

	siklsimon murakkablikni 8-bo‘limda muhikama qilaman.
Ismlarning uzunligi	Ismlarning(o‘zgaruvchi, klass, metod va hoakozolar nomi) o‘rtacha uzunligi o‘lchami. Ismlar qancha uzun bo‘lsa, shuncha ma‘no yaqin beruvchi (izoh kabi) va shuning uchun dasturham tushunarli bo‘ladi.
Holatga moslashish kengligi	Bu dasturdagi shart bayonlarining mos tushish kengligi o‘lchami. Chuqur joylashgan shart bayonlari tushunishga qiyin va ancha xatolikka moyil.
Chalkashish indeksi	Bu hujjatdagi so‘zlar va gaplarning o‘rtacha uzunligi o‘lchami. Hujjatning chalkashlik indeksi qanchalik katta qiymatga ega bo‘lsa, hujjatni tushunish shunchalik qiyin bo‘ladi.

3.11- rasm. Dasturiy ta‘minot maxsulot o‘lchamlari statistikasi

El-Amam(2001), OY o‘lchamlarini yaxshilab qayta o‘rganganda, CK va OY o‘lchamlarini muhokama qiladi va dastur tashqi sifatlariga shu yoki boshqa obyektga yo‘naltirilgan dasturlash o‘lchamlari qanday bog‘liqligini tushunish uchun ishonchli manba yo‘qligin xulosa qiladi. Bu holat chindan ham o‘zgargani yo‘q 2001- yildagi uning tahlillaridan beri.

3.4.2. Dasturning tarkibiy qismi tahlili

Dastur sifatini belgilash jarayonining qismi bo‘lgan o‘lchash jarayoni 3.13 rasmda ko‘rsatilgan. Sistemaning har bir qismi turli o‘lchamlardan foydalanib alohida tahlil qilinishi mumkin. Bu o‘lchamlarning qiymatlari so‘ngra boshqa qismlarga taqqoslanishi mumkin va balkim oldingi loyihalardan yig‘ilgan tarihiy o‘lchov ma‘lumotlari bilan ham.

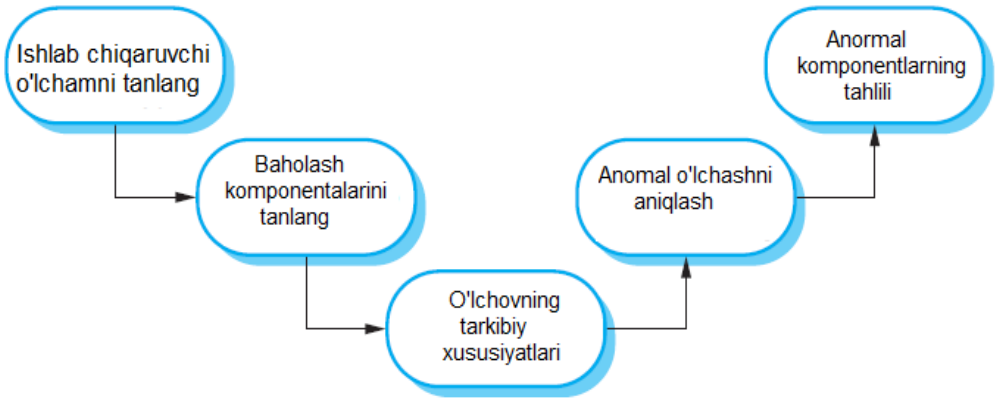
Obetga yo‘naltirilgan o‘lcham	Izoh
Klassga maxsus metodlar (KMM)	Bu klassdagi o‘zining murakkabligiga ko‘ra maxsuslashgan metodlar soni. Shuning uchun, oddiy metod murakkabligi 1 va katta va murakkab metod ancha katta qiymat olishi mumkin. Bu o‘lcham qancha katta qiymat olsa, obyekt klasslari hamshunchalik murakkab. Murakkab obyektlar tushunishga qiyin bo‘lishi mumkin. Ular mantiqan tushunarli bo‘lmasligi mumkin, shuning uchun ular merosho‘rlik daraxtidagi super klasslar kabi samarali qayta foydalanila olmaydi.
Merosxo‘rlik daraxtining kengligi(MDK)	Bu merosxo‘rlik daraxtidagi ostki klasslar super klasslardan xususiyat va vazifalarni(metodlarni) meros olgan joydagi aniq darajalar sonini ko‘rsatadi.
Vorislar soni (VS)	Bu klasdagi bevosita klasslar soning o‘lchami. Bu klass ierarxiyasining enini o‘lchaydi, MDK uning kengligini o‘lchagani kabi. VSning katta qiymati ko‘proq qayta foydalanishni ko‘rsatadi. Bu asosiy klassni ma‘qullashda ko‘proq zo‘r berishni anglatadi ularga bog‘liq bo‘lgan ostki klasslarning soni tufayli.
Obyekt klass bilan birlashish (OKbB)	Klasslar bir klassdagi metod boshqa klassda ifodalangan metodlar va o‘zgaruvchilardan foydalanganda birlashadi. OKbB qanchalik ko‘p birlashish mavjudligining o‘lchovi. OKbBning yuqori qiymati klasslarning yuqori darajada bog‘langanligini anglatadi va shuning uchun dasturdagi bir klassning o‘zgarishi boshqa klasslarga ta‘sir qiladi.
Klasga javob qaytarish (KJQ)	KJQ obyekt klass tomonidan qabul qilingan xabarga javob tarzida mumkin darajada ishga tushadigan metodlar soni o‘lchami. Yana, KJQ murakkablikka ham bog‘liq. KJQning

	qiymati qancha yuqori bo‘lsa, klass shunchalik murakkab va shuning uchun hatoliklarni o‘z ichiga olishga shunchalik moyil bo‘ladi.
Metodlarda bog‘lanish kamligi (MBK)	MBK klassdagi metodlar juftligini hisobga olish orqali topiladi. MBK bir xususiyatni baham ko‘rgan metodlar soni va bir hususiyatni baham ko‘rmagan metodlar soni o‘ryasidagi farq. Bu o‘lchamning qiymati keng muhokama qilib kelingan va bir necha turlarda mavjud. MBK chindan ham boshqa o‘lchamlar berganlarining oxiriga yoki boshiga foydali ma‘lumot, qo‘shimcha qo‘shadimi, bu aniq emas.

3.12- rasm. O‘byektga yo‘nalgan o‘lchamlar formasi

Normadan chetga chiqqan o‘lchashlar, normadan sezilarli darajada boshqa yo‘nalishda bo‘ladi, bu qismlarning sifati bilan bog‘liq xatoliklar borligini nazarda tutishi mumkin. Bu qismlarni o‘lchashda kalit bosqichlar:

1. *Bajariladigan o‘lchashlarni tanlash.* O‘lchash javob berishi maqsad qilingan savollar formulaga solingan bo‘lishi lozim va o‘lchashlar javob berishi talab etilgan savollar tushuntirilgan bo‘lishi lozim. Bu savollarga bog‘liq bo‘lmagan o‘lchashlar yig‘ilishi shart emas. Basilining MSO‘(maqsad-savol-o‘lchash) misoli(Basili va Rombach,1988), 26-bo‘limda muhokama qilingan, qaysi malumot yig‘ilishi kerakligiga qaror qilishda eng yaxshi usul.



3.13- rasm. Mahsulot o‘lchov jarayoni

2. Baholanishi kerak bo‘lgan qismni tanlash. Dastur tizimidagi barcha qismlarning o‘lchamlarini baholashingiz shart bo‘lmasligi mumkin. Bazida, ko‘rinadigan qismlarni tanlashingiz mumkin, Sistemanig sifatini to‘liq baholash uchun. Boshqa safar esa, doimiy foydalaniladigan tub qismlarga e’teibor qaratishni istashingiz mumkin. Bu qismlarning sifati kam foydalaniladigan qismlarning sifatidan ko‘ra muhim.

3. Qism harakterlarini o‘lchash. Tanlangan qismlar o‘lchanadi va hisoblangan o‘lcham qiymatiga bog‘lanadi. Bu odatda avtomatik malumot yig‘ish qurilmasidan foydalanib qism namoyishi(dizayn, kod) jarayonini o‘z ichiga oladi. Bu qurilma maxsus yozilgan yoki ishlatilayotgan dizayn qurilmasining xususiyati bo‘lishi mumkin.

4. Normadan chetga chiqqan o‘lchashlarni aniqlash. Qism o‘lchashlari bajarilganidan keyin, ularni bir biri bilan va o‘lchashlar ma‘lumotlar bazasida yozilgan oldingi o‘lchashlar bilan taqqoslaysiz. Siz har bir o‘lcham uchun noodatiy katta va kichik qiymatlarni qidirishingiz lozim, huddi bular bu qiymatlarni ko‘rsatayotgan qismlar bilan bog‘liq xatoliklar borligini taklif etayotgandek.

5. Normadan chetga chiqqan qismlarni tahlil qilish. Tanlangan o‘lchamlar uchun normadan chetga chiqqan qiymatlar mavjud qismlarni aniqlaganingizda, siz normadan chetga chiqqan qiymatlar bu qismning sifati pasaytirilganini anglatadimi yoki yo‘qmi qaror qilishingiz kerak. Murakkablik uchun normadan chetga chiqqan o‘lcham qiymati (deylik) doim ham past sifatli qismni anglatmaydi. Bu yerda yuqori qiynat uchun boshqa sabab bo‘lishi mumkin, shuning qism sifati muammosi borligini anglatmaydi.

Siz doim yig‘ilgan ma‘lumotni tashkiliy manba sifati ko‘rishingiz kerak va hamma loyihaning oldingi yozuvlarini saqlang hattoki u ayni bir loyihada ishlamagan bo‘lsa ham. Yetarli darajada katta o‘lchashlar ma‘lumotlar bazasi tashkil etilganda, keyin loyihalar o‘rtasida sifat taqqosini amalga oshira olasiz va ichki qism xususiyatlari va sifat xarakterlari o‘rtasidagi aloqani tasdiqlay olasiz.

3.4.3. O‘chash noaniqligi

Dasturiy ta‘minot va dasturiy jarayonlar haqida miqdoriy ma‘lumotlar to‘plash jarayonida siz bu ma‘lumotlarni mohiyatini tushunish uchun ularni tahlil qilishingiz zarur. Ma‘lumotlarni noto‘g‘ri talqin qilish va noto‘g‘ri xulosa chiqarish oson hisoblanadi. Siz shunchaki ma‘lumotning o‘ziga qaray olmaysiz, siz ushbu ma‘lumot olingan konteksni ham hisobga olishingiz zarur.

Yig‘ilgan ma‘lumotlar har xil talqin qilinishiga oid bo‘lgan misolni ko‘rib chiqamiz. Bunda tizim foydalanuvchilari tomonidan qilingan o‘zgartirish bo‘yicha so‘rovlarning miqdori bilan bog‘liq holar ko‘riladi:

Ish boshqaruvchi dasturiy mahsulotning foydalanishga qulayligi va yariqliligi bilan mijozlarning o‘zgartirish bo‘yicha so‘rovlari o‘rtasida bog‘liqlik mavjud deb faraz qilib, ushbu so‘rovlarning miqdorini kuzatishga qaror qiladi. U o‘zgartirish bo‘yicha so‘rovlar qanchalik ko‘p bo‘lsa, dasturiy ta‘minot foydalanuvchilar talabiga shunchalik javob bermaydi deb o‘ylaydi.

So‘rovlarni qayta ishlash va dasturiy ta‘minotni o‘zgartirish qimmatga tushadigan ish hisoblanadi. Shuni hisobga olgan holda tashkilot foydalanuvchilar ehtiyojini qondirsh va o‘zgarishlar narxini pasaytirish maqsadida o‘z ish jarayonini o‘zgartirishga qaror qiladi. Maqsad ushbu ish jarayondagi o‘zgarishlar nisbatan sifatliroq mahsulot va kam miqdordagi o‘zgartirish bo‘yicha so‘rovlarga olib kelishi hisoblanadi.

Jarayon o‘zgarishlari dasturiy ta‘minot ishlab chiqilishida mijoz ishtirokini oshirish maqsadida boshlanadi. Barcha mahsulotlarning beta-testlanishi kiritiladi va mijoz so‘rovi sozlamalari yetkaziladigan dasturga kiritiladi. Ushbu o‘zgartirish natijasida hosil bo‘lgan dasturiy mahsulotning yangi versiyasi egasiga yetkaziladi. Ba‘zi hollarda o‘zgartirish bo‘yicha so‘rovlar miqdori qisqartiriladi. Ba‘zi hollarda esa oshiriladi. Ish boshqaruvchi bunday chalkashlikdan keyin mahsulot sifatiga jarayonni o‘zgartirishning ta‘sirini baholash mumkin emas degan xulosaga keladi.

Ushbu holatdagi kabi ikki xil fikrlilikni nima uchun paydo bo‘lishini tushunish uchun siz foydalanuvchilarda o‘zgartirish bo‘yicha so‘rovlar paydo bo‘lish sabablarini tushunishingiz zarur:

1. Dasturiy mahsulot yetarlicha yaxshi emas va dastur mijoz xohlagan ishlarni bajarishga imkon bermaydi. Shuning uchun mijozlar o‘zlari uchun zarur funksional imkoniyatlar kiritilishini xohlab o‘zgartirish bo‘yicha so‘rov jo‘natadilar.

2. Boshqa tomondan dasturiy ta‘minot juda yaxshi bo‘lishi mumkin va shuning uchun juda keng qamrovli va ishlatish uchun og‘irlik qilishi mumkin. O‘zgartirish bo‘yicha so‘rovlar dasturdan foydalanuvchilarning ko‘pligi va ularda dasturiy mahsulot bo‘yicha yangi g‘oyalarning paydo bo‘lishi sababli ham yuzaga kelishi mumkin.

Shunday qilib, ish jarayonida mijoz ishtirokining o‘sishi, dasturdan norozi mijozlarning o‘zgartirish bo‘yicha so‘rovlari sonini kamaytirishi mumkin. Jaroyondagi o‘zgarishlarning samarali bo‘lganda dasturiy mahsulot qulayroq va foydalanuvchi uchun mos holatga keladi. Mobodo jarayondagi o‘zgarishlar ish bermasa, mijozlar ham muqobil boshqa tizimni qidirishlari mumkin. O‘zgartirishlar bo‘yicha so‘rovlar miqdori dasturiy mahsulot o‘zining bozordagi o‘rnini boy bergan bo‘lsa, ya‘ni undan kam foydalanilayotgan bo‘lsa ham kamayishi mumkin.

Boshqa tomondan ish jarayonidagi o‘zgarishlar ko‘plab mijozlarni hursand qilishi va dastur yaratilish jarayoniga qatnashishni istagan bildirgan yangi mijozlar sonining oshishiga olib kelishi mumkin. Shuning uchun ular o‘zgartirish bo‘yicha so‘rovlarni yuborishadi. Jarayondagi o‘zgarishlar o‘zgartirish so‘rovlarini qayta ishlash natijasida ko‘payishi mumkin. Agar kompaniya mijozlarga ko‘ngilchangroq bo‘lsa, ular o‘zgartirish so‘rovlarini ko‘p jo‘natishlari mumkin, chunki bu so‘rovlar inobatga olishini bilishadi. Ular dasturning keyingi versiyalarida o‘zlarining takliflari hisobga olingan bo‘ladi deb hisoblashadi.

Asosiy fikrlar

- Dastur sifatli boshqarish dastur nuqsonlarini past holatga keltirishni o‘z

ichiga oladi va u dastur faoliyatiga, ishonchliligi, kerakli zarur standartlarga ega bo‘ladi. Bu standartlar bilan to‘g'ri keldi deb ta'minlash uchun jarayonlarini va mahsulotlarni shuningdek jarayonlarni tashkil ishlash uchun standartlar aniqlashni o‘z ichiga oladi.

- Ta'riflar, Dasturiy ta'minot standartlari sifatini ta'minlash uchun zarur bo‘lgan «Eng yaxshi amaliyotlari" mavjud. U dasturiy ta'minot ishlab chiqish, standartlar sifatli dasturiy ta'minot yaratish uchun mustahkam asos vazifasini bajaradi.
- Siz tashkiliy va sifatli qo‘llanmada sifat nazorati protseduralari majmuini hujjatlashtirishingiz kerak. Bu bir sifatli qo‘llanma umumiy modeliga asoslangan bo‘lishi mumkin, u ISO 9001 standarti bo‘lishi mumkin.
- Dasturiy ta'minot jarayoni natijalarini tadqiqotlari sifat standartlari bilan teng bo‘ldi, deb tekshirib hodimlar jamoasi natijani o‘zlariga oladi. Va bu sharx sifatini baholash uchun eng keng tarqalgan usul mavjud.
- Tekshirish dasturi, yoki tengdoshlari bilan ko‘rib, bir kichik guruh muntazam ravishda kodni tekshiradi. Ular batafsil kodni o‘qish va iloji boricha xatolar va kamchiliklar uchun mas'ul. Topilgan muammolar keyin kodni ko‘rib majlisida muhokama qilinadi.
- O‘lchash, Dastur yoki dasturiy ta'minot va dasturiy ta'minot jarayoni haqida miqdoriy ma'lumotlarni to‘plash uchun foydalanish mumkin. Biz mahsulot va jarayon sifati haqida xulosa qilish boryapmiz, dasturiy ta'minot ko‘rsatkichlar qiymatlaridan foydalanish imkoniyatiga ega bo‘lishi mumkin.
- Mahsulot ko‘rsatkichlarida sifati bilan bog'liq muammolar bo‘lishi , bu ayniqsa tabiiy bo‘lmagan komponentlarini ajiratish uchun foydalidir. Ushbu qismlarni keyin batafsil tahlil qilinishi kerak.

Nazorat savollari:

1. Bu jarayon yuqori sifatli dasturiy mahsulotlar olib kelishini va yuqori sifatli dasturiy ta'minot ishlab chiqish nima uchun kerakligini tushuntiring. Bu tizim bilan bog‘lik muommolarini muhokama qiling.

2. Qanday qilib standartlar dasturiy ta‘minot ishlab chiqishning samarali metodlari haqida tashkiliy o‘rnak olish uchun qo‘llanilishi mumkinligini tushuntiring. Tashkiliy standartlardan olish mumkin bo‘lgan to‘rt tipdagi bilimni taklif eting.

3. \ 3.2 rasmda ko‘rsatilgan sifat atributlariga mos ravishda dasturiy ta‘minot sifatini baholashni muxokama qiling. Siz o‘z navbatida har bir atributni ko‘rib chiqishingiz va bu qanday baholangan bo‘lishi mumkinligini tushuntiring.

4. Taqrizchilar izohlarini yozib olish va ularga izohlarni elektron pochta orqali jonatishga xizmat qiladigan elektron shakl yarating.

5. Quyidagilar uchun qo‘llanilishi mumkin bo‘lgan ehtimoliy standartlarni qisqacha tavsiflang:

- C, C # yoki Java da boshqaruv konstruksiyalarini qo‘llash;
- Universitetda oraliq loyihasi o‘rnida keltirilishi mumkin bo‘lgan hisobotlar;
- Dasturiy o‘zgarishlarni qabul qilish va tasdiqlash jarayoni(26 bobga qarang);
- Yangi kompyuterni harid qilish va unga dasturlarni o‘rnatish jarayoni;

6. Faraz qilaylik, siz kichik biznes va jismoniy shaxslar uchun ma‘lumotlar bazasini ishlab chiqish bilan shug‘ullanadigan tashkilotda ishlaysiz. Bu tashkilot o‘zining dasturiy ta‘minot ishlab chiqishda miqdoriy baholanishini xohlaydi.

7. Datur tekshiruvchilari bir dasturdagi hatoliklar texnik samarasi nima uchunligini tushuntiring? Qanday turdagi hatoliklar tekshiruv orqali hal etiladi?

8. Dizayn sifati va dizayin o‘lchami bir-biri bilan nima uchun metodlari kamligini tushintiring.

9. Aylana murakkabligi va tashqi atrebutlar sifati ichki maxsulotlar orasidagi sifatlar orasidagi munosabatlarni tasdiqlash qiyinligini tushuntiring.

10. Juda yaxshi dasturchi hamkasblari bilan dasturlar ishlab chiqaradi, lekindoimiy tashqi sifat standartlari pastligi uchun nazarga olinmaydi.

4. FORMALARNI BOSHQARISH

Dasturiy tizim rivojlanish va foydalanish davomida doimo o‘zgarib, yuksalib kelgan. Texnik nuqsonlar topilgan va bartaraf etilgan. Tizim talabini o‘zgartirib, uni o‘ziz qurilmangizga qarab yangi variant (versiya) ni sozlaysiz. kompyuterning apparat qismlari va sistemali platformaning yangi varianti yaroqli bo‘lsa va uni siz o‘z tizimingizga moslab ishlashingiz mumkin. Raqiblar siz bilan bor tizimlaridagi yangi xususiyatlar ega tizim taqdim etib musobaqalashishadi. Ya‘ni o‘zgartirib tayyorlangan dasturlar, bu tizimning yangi versiyasi (varianti) yaratilganidir. Ko‘p tizimlar shuning uchun tarmoqga e’tibor bera oladigab, har qaysi versiyasini boshqara oladigan va qo‘llab quvvatlamog‘i kerak.

Konfiguratsiya (formalar) boshqaruvi bu – tizimli dasturni boshqaruvini soddalashtirishga taaluqli bo‘lgan ko‘rsatmalarni boshqaradi. Sizga tizimni rivojlantirish zarur bo‘lsa, u tizim siz uchun soddalik qilsa, yangi versiyadagi tizimning tarkibiy qism qo‘shib olishingiz mumkin. Opertsion tizimlarning versiyalari o‘zgartirish taklifini amalga oshiradi, kamchliklarni to‘g‘rilashga va turli kompyuter qismlarini bir – biriga moslashtirishga yo‘naltirildi. Unda bir qancha versiyalar taraqqiy etgandan keyin va bir vaqtning o‘zida foydalanish mumkin. Agar siz joylashtirgan konfiguratsiya boshqaruvidan qoniqmasangiz, siz tizimning yaxshi bo‘lmagan versiyasiga zo‘r berayotgan bo‘lasiz, yoki sizga tizimning imtiyozsiz versiyasi yetkazib berilgan, yo bo‘lmasa tizimning maxsus versiyasining dasturiy kodi joylashgan manzili unitilgan, yoki komponentasi saqlangan.

Konfiguratsiya (formalar) boshqaruvi bu shaxsiy loyihalar uchun foydali, u oson yakka foylanuvchi. Bir vaqtda ishlaydigan dasturiy tizimni yaratuvchilarga bu jarayon loyihalar guruxi uchun muxim. Ba‘zan o‘sha yaratuvchilar barcha ishlarini bir joyning o‘zida tashkil etishadi, ammo yuksalgan jamoalar bu jarayonni dunyoning turli nuqtalarida joylashgan sheriklari bilan taqsimlashadi.

Boshqaruv Konfiguratsiya (formalar) tizimli dasturning 4 qismini o‘z ichiga oldi (4.1- rasm):

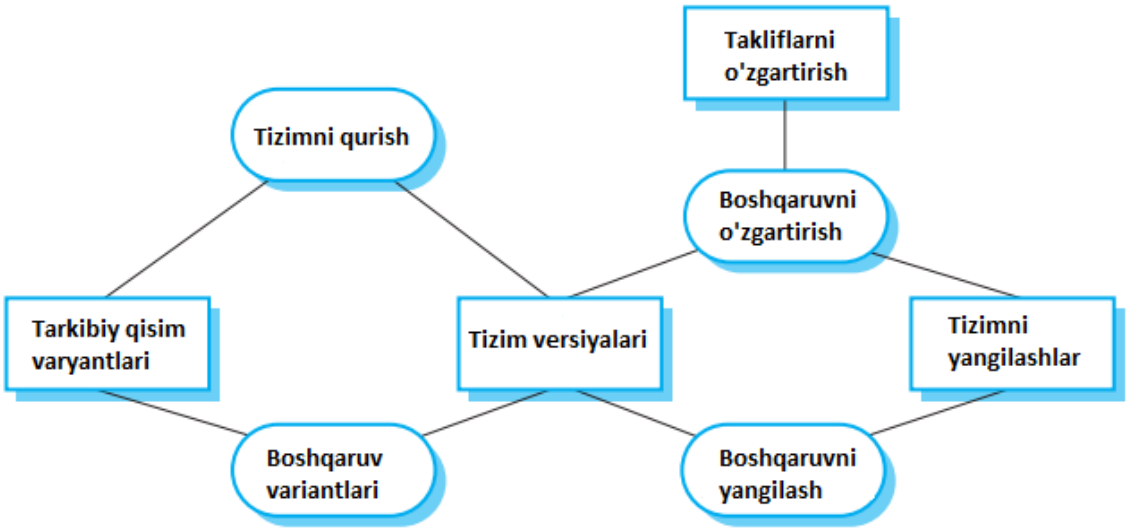
1. *Boshqaruvni o‘zgarishi.* Bu qismda ishlab chiqaruvchi va xaridordan dasturning o‘zgartirishga bo‘lgan talab qondiriladi, ishlab chiqarish va o‘zgartirish natijasida narx belgilanadi va ishlatiladi.

2. *Boshqaruv turi (versiyasi).* Bu qismda bir necha versiyalari kuzatishni o‘z ichiga oladi, Tizim qismlariga va ishlab turgan turli qismlariga qilingan o‘zgarishlar bir-biri bilan aralashib ketmasligini ta'minlash haqida so‘z yuritiladi.

3. *Tizim qurilishi.* Bu yig'ish jarayoni dasturiy ta'minot qismlari, ma'lumotlar va kutubxonalar, keyin kompilyatsiya qilinadi va tizimini yaratish, ularni bog‘lash haqida so‘z boradi.

4. *Boshqaruvni anglash.* Bu qism mijoz tomonidan foydalanish uchun ozod etiladigan tashqi muammolarni ko‘rish tizimi va versiyalari uchun dasturiy ta'minot tayyorlashni o‘z ichiga oladi.

Konfiguratsiya boshqarish siyosati va jarayonlari aniqlash qanday yozib olish va jarayon taklif etilgan bu tizim o‘zgarishlarini qayd etib borish, qanday tizim komponentlarini qaror qabul qilish tizimi va uning tarkibiy qismlari turli versiyalarini boshqarish uchun o‘zgartirish va qanday mijozlarga o‘zgarishlar tarqatishni ko‘rsatadi.



4.1- rasm. Konfiguratsiya boshqarish bo‘yicha faoliyati

Konfiguratsiya boshqarish vositalari boro‘zgartirish takliflar kuzatib, tizim komponentlarini do‘kon versiyalarida qolish uchun ishlatiladigan, Tizimlarni

qurish va ishdan chiqishini oldini olish, xaridorlarga tizim versiyalarini kuzatib borish tavsiya etiladi.

Konfiguratsiya boshqarish ba'zan dasturlarining bir qismi hisoblanadi. Shu boshqaruvga ega bo‘lgan sifat boshqaruvi (bob 24 bilan qoplangan), sifat boshqaruvi va konfiguratsiya boshqarish mas'uliyat bir biriga bog‘liqdir, Amalga oshirilgan dasturiy ta'minotni yangi versiyasi. Sifat Kafolati (SK) jamoasi rivojlantirish guruhi tomonidan topshiriladi. SK jamoasi tekshirgandan keyin tizim sifati qabul qilinadi. Agar tizim shunday bo‘lsa, u tizimiga barcha o‘zgarishlar haqida kelishib va qayd qilinishi degan ma'noni anglatadi.

Konfiguratsiya boshqarish standartlari quyidagi ikki xalqaro standartlar asosida tuziladi. ISO 9000 va CMM va CMMI standartlari ham sifatli sertifikatlash tizimidir. Bu CM standartlari rivojlanayotgan bo‘lib umumiy CM standartlarga asoslangan bo‘lib, IEEE sifatida organlari tomonidan ishlab chiqilgan. Misol uchun, IEEE 828-1998 standart bir aholi emas konfiguratsiya boshqarish rejalari uchun to‘g‘ri keladi. Ushbu standartlar CM jarayonlar va hujjatlar CM asosida ishlab chiqarilgan. Tashqi standartlardan foydalanishning boshlang'ich nuqtasi sifatida, kompaniyalar, kompaniya maxsus aholi rivojlantirishi, ularning o‘ziga xos ehtiyojlariga ko‘ra bajariladi.

Konfiguratsiya boshqaruv muammolardan biri bu turli kompaniyalar turli atamalar yordamida bir xil tushunchalar haqida gapirishning tarixiy sabablari bor.

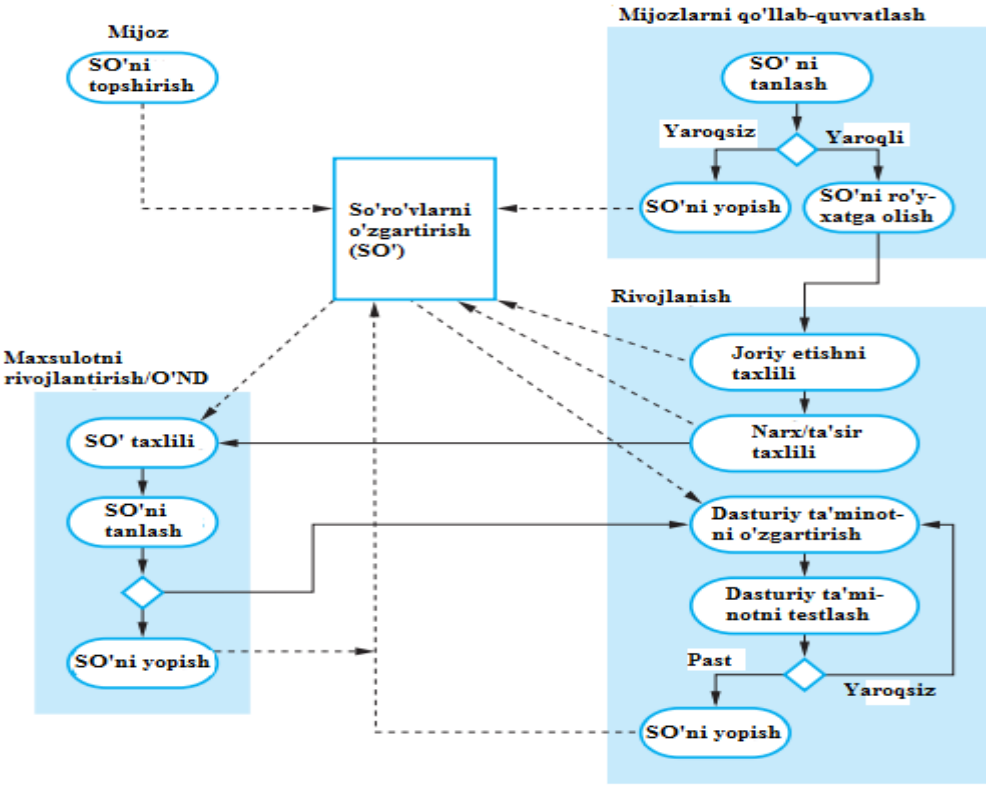
Atamalar	**Atamalarga berilgan ta‘rif (IZOX)**
Konfiguratsiya element yoki dasturiy konfiguratsiya	Dasturiy ta'minot loyihasi (dizayn, kod, test ma'lumotlar, hujjat bilan bog'liq element (SCI) konfiguratsiya nazorat ostida qoladi va boshqalar). Ko‘pincha konfiguratsiya elementi versiyalari tez-tez o‘zgarib boradi.
Konfiguratsiya nazorati	Konfiguratsiya tizimlari va komponentlarini versiyalari o‘sish jarayonini ta'minlash, nazorat qilish va o‘zgarishlarni boshqariladi, shuni saqlash uchun va komponentlarini barcha versiyalar

	aniqlangan va tizimi rivoji uchun saqlanadi.
Variantlar	Boshqa hollarda, ba'zi tarzda, konfiguratsiya element farq qiladi. Versiyalari har doim tez-tez tashkil topgan yagona identifikatorni bor, konfiguratsiya element nomi qo‘shimcha bir versiya raqami bilan birga qo‘llanadi.
Asosiy yo‘nalish	To‘plam va Tayanch tizimini tashkil etuvchi tarkibiy versiyalari bir. Tayanch – komponentlarini versiyalari tizimini tashkil etuvchi degan ma'noni anglatadi, nazorat o‘zgartirilishi mumkin emas. Bu har doim bir asos yakunini qayta yaratish mumkin bo‘lishi kerak degan ma'noni anglatadi va uni tashkil etuvchi komponentlari mavjud.
Kod yo‘nalis	Kod yo‘nalishi bir dasturiy komponent va boshqa konfiguratsiya versiyalari qatori bu qismi bog'liq mahsulot.
Bosh yo‘nalish	Mainline tizimi turli versiyalarini vakili boshlang'ich bir ketma-ketlikni asosiy yo‘nalishi.
Bildirishnoma (release)	Xaridorlar tomonidan (yoki muassasaning boshqa foydalanuvchilari tomonidan) foydalanish uchun bildirilgan tizimning versiyasi
Ish joyi (workspce)	Dasturni modifikatsiya qilishda tashqi tasirlarsiz, ishchi qanday qilib dasturni modifikarsiya qilishni hohlasa shunday xususiy ish maydoni isoblanadi.
Bog‘lash	Mavjud kod yo‘nalishi bir versiyasiga yangi kod yo‘nalishi yaratish. Yangi kod yo‘nalishi va mavjud kod yo‘nalishi keyin mustaqil rivojlanishi mumkin.
Marging	Alohida birlashtirib bir dasturiy komponent yangi versiyasini turli kod yo‘nalishi versiyalari yaratish, birlashtirish. Bu kod yo‘nalishis ishtiroki oldingi filiali tomonidan yaratilgan bo‘lishi.

Tizimning qurilishi	Kompilyatsiya va bog'lovchi tomonidan olib boriladi tizimi versiyasiga yaratishni barpo tizimi tizimini tashkil qilish komponentlar va kutubxonalar tegishli versiyalari.

4.2- rasm.

Harbiy dasturiy ta'minot tizimlari, ehtimol konfiguratsiya boshqarishda ishlatiladigan birinchi tizimlar edi va bu tizimi uchun terminologiyasi tiklablangan apparat iste'mol uchun joy jarayonlari va allaqachon tartib aks etgan ko‘rinishda boshqarishdir. Tijorat tizimlarida chiquvchilar tanish emas edi. Ko‘pincha harbiy tartib yoki termin bilan o‘z ixtirolarini atamalar bilan atashgan. Tezkor usullarini, shuningdek, ba'zan, yangi terminologiya bilan tanushtiradi. Eng so‘nggi texnologiyalar maxsus an'anaviy CM tezkor yondashuvni ajratish uchun metodlar qo‘llanadi. (Rasm 4.2) Bu bobda konfiguratsiya boshqarish terminidan foydalanilgan.



4.3- rasm O‘zgarish boshqarish jarayoni

4.1. O‘zgarishlarni boshqarish

Katta dasturiy tizimlar faoliyati uchun o‘zgarishlar haqiqatdir. Tashkiliy ehtiyojlari va talablarga tizimini hayoti davomida o‘zgarishlari, kamchiliklar ta'mirlanishi kerak, va ularning atrof-muhit o‘zgarishiga moslashish tizimlari bor. o‘zgarishlar ishonch hosil qilish uchun bir nazorat tizimi tarzda qo‘llaniladi, bunda sizga vositani qo‘llab-quvvatlaydigan, o‘zgarishni boshqarish jarayonlari majmui kerak. O‘zgarish boshqaruvi tizimining evolyutsiyasi ishonch hosil qilish uchun mo‘ljallangan muvaffaqiyat jarayoni va ustuvor eng dolzarb va iqtisodiy samarali o‘zgarishlar beriladi.

O‘zgarishni boshqarish jarayoni xarajatlarini va tahlil bilan bog'liq bo‘lgan daromadlar, Tavsiya etilgan o‘zgarishlar mos va munosib bo‘lgan o‘zgarishlar tasdiqlash va kuzatuv tizimi qismlariga o‘zgartirilgan. Modelning asosiy o‘zgarish boshqaruv jarayonini boshqarish faoliyati ko‘rsatadi (rasm 4.3). Foydalanishda bu jarayon ko‘p varyantlar bor lekin samarali bo‘lishi uchun, boshqaruv o‘zgartirish bor jarayonlar har doim anglamoq, tekshirish xarajat va tasdiqlash bir vositalarida o‘zgarish bo‘lishi kerak. Bu jarayon dasturiy ta'minot mijozlarga yoki tashkilot ichidagilarga tarqatish uchun topshiriladi.

Mijoz jarayonni tugallangach o‘zgarish boshqarish jarayonida boshlanadi va tizim uchun zarur bo‘lgan o‘zgarishlarni tushuntirib, o‘zgarish so‘rovini jo‘natadi. Bunday bo‘lishi mumkin bunda xato alomatlari tasvirlanadi, va xato hisobot yoki so‘rov uchun qo‘shimcha funktsional tizimiga qo‘shiladi. Ba'zi kompaniyalar xato band hisobotlar va yangi talablar alohida, asosan, bu shunchaki ham o‘zgartirish so‘rovlar. O‘zgartirish so‘rovlar talabi o‘zgarish shakli yordamida berilishi mumkin (CRF). Men «mijoz» atamasidan bu yerda foydalanish o‘zgarishlar bozor tomonidan taklif qilinishi mumkin bir qismi emas, balki har qanday bir kompaniya bo‘limiga ishlash jamoasi muddatini o‘z ichiga oladi.

Change Request Form

Project: SICSA/AppProcessing **Number:** 23/02
Change requester: I. Sommerville **Date:** 20/01/09
Requested change: The status of applicants (rejected, accepted, etc.) should be shown visually in the displayed list of applicants.

Change analyzer: R. Looek **Analysis date:** 25/01/09
Components affected: ApplicantListDisplay, StatusUpdater

Associated components: StudentDatabase

Change assessment: Relatively simple to implement by changing the display color according to status. A table must be added to relate status to colors. No changes to associated components are required.

Change priority: Medium
Change implementation:
Estimated effort: 2 hours
Date to SGA app. team: 28/01/09 **CCB decision date:** 30/01/09
Decision: Accept change. Change to be implemented in Release 1.2
Change implementor: **Date of change:**
Date submitted to QA: **QA decision:**
Date submitted to CM:
Comments:

4.4- rasm. O‘zgarish so‘ro‘v shaklining qisman yakunlanishi

O‘zgarish boshqaruvida ishtirok guruhlari elektron o‘zgartirishlar talabi orasidagi barcha ma'lumotni birgalikda ro‘yxatga olishdan hosil bo‘ladi. O‘zgartirish talabi ishlovi sifatida, ish haqqi haqidagi ma'lumot jarayonining har bir bosqichida qabul qilingan qarorlarni yozib uchun CRF qo‘shiladi. Har qanday vaqtda, u zarur o‘zgarishlarni oniy saqlashni anglatadi. CRF ga bog'liq holda tavsiyalarni yozib o‘zgartirish tasdiqlangan so‘ng amalga oshiriladi, va tasdiqlangan o‘zgarishda taxmin xarajatlar amalga oshirildi.

CRF bir sekundni o‘z ichiga olishi mumkin, ishlab chiqaruvchilar bir o‘zgarish amalga oshirish mumkin, qisman tugallangan o‘zgarishlar so‘rov shaklida 4.4- rasmda ko‘rsatilgan. Bu katta murakkab tizimlarida CRF bir turdagi muhandislik loyihalarida foydalanish mumkin. kichik loyihalar uchun, men kerak tavsiya va so‘rovlar o‘zgarishini rasman qayd qilinishini va CRF zarur o‘zgarishlarni amalga oshirish masalalari bo‘yicha kamroq e'tibor bilan bayonod

qilish kerakligini aytib o‘tmoqchiman. Tizim ishlab chiquvchilari o‘zgarishlar qarorini amalga oshirish va taxmin qilish uchun zarur vaqt mavjud.

O‘zgarish talabi topshirilgandan so‘ng, u ishonch hosil qilish uchun tekshirish amal qiladi. Tekshiruvchi mijozlardan yoki ilovalarni qo‘llab quvvatlovchilardan ichki so‘rovlarni quvvatlash uchun, rivojlantirish jamoasi a'zosi bo‘lishi mumkin. Tekshirish kerakli barcha o‘zgartirish so‘rovlarini qabul qila olmaydi. O‘zgartirish talabi bir xato haqida xabar bo‘lsa, bu avvalgi xabar bo‘lishi mumkin. Ba'zi hollarda, odamlar allaqachon amalga oshirilgan, biroq ular bilmaydigan narsalardan oddiy so‘rov xususiyatlari haqida xabar beradilar. Bu har qanday haqiqiy bo‘lsa, o‘zgarishi so‘rov yopiladi va shakli yopilishi uchun sabab bilan yangilanadi. U amal so‘rov o‘zgarishi bo‘lsa, u bir chiqayotgan kamchilik sifatida kirgan keyingi tahlil qilish uchun so‘rov turgan.

Amal o‘zgarish istaklari uchun, jarayonning keyingi bosqichi baholash va xarajat o‘zgarishidir. Bu, odatda, ishlab chiqish yoki xizmat jamoa sifatida javobgar hisoblanadi. Ular o‘zgarishlarni amalga oshirishda ishtiroki ta'siri tizimining qolgan o‘zgartirish tekshirilishi kerak. Buning uchun, siz barcha o‘zgarishi ta'sir komponentlarini aniqlashiz kerak. O‘zgarish qilish yanada bo‘lsa boshqa joyda tizimda zarur o‘zgarishlar amalga oshirishni anglatadi, bu o‘zgartirish xarajatlarni oshirilishini anglatadi. Keyingi tizim bor modul uchun zarur bo‘lgan o‘zgarishlar baholavadi. Nihoyat, o‘zgarish qilish qiymati hisobga olgan holda, taxmin qilingan tegishli qismlariga o‘zgartirish xarajatlar belgilanadi.

Ularning iqtisodiy tahlil quyidagi alohida guruh bo‘lsa, keyin biznes nuqtai nazaridan samarali dasturiy ta'minot o‘zgarishlarni amalga oshirish uchun qaror kerak. harbiy va davlat tizimlari, bu guruh tez-tez o‘zgarishi (CCB) nazorat kengashi deb ataladi. Sanoatda u “mahsulot ishlab chiqish guruxi” deb atash mumkin. Dasturiy ta'minot tizimi haqida qaror qabul qilish uchun mas'ul bo‘lgan guruh ko‘rib chiqish va kerak bo‘lsa, barcha o‘zgarish so‘rovlarini tasdiqlash kerak. O‘zgarishlar ekran displeylari, veb-sahifalari yoki hujjatlarida kichik xatolarni tuzatishga jalb etiladi. Bu kichik so‘rovlar rivojlantirish jamoasi batafsil tahlil qilmasdan o‘tgan bo‘lishi kerak.

CCB yoki mahsulot ishlab guruhi tomonidan o‘zgarishining ta'sirini strategik va tashkiliy punkt o‘rniga texnik punkt ko‘rib chiqadi. Bu qaror o‘zgarishi iqtisodiy asoslangan va qabul darajasida ustuvor ahamiyat berilmoqda. Qabul o‘zgarishlar rivojlantirish uchun orqaga o‘tadi guruhi; rad etilgan o‘zgarish so‘rovlari yopiladi va yana harakat boshlanadi. Mazmunli o‘zgarish kerak yoki yo‘qligini qaror qabul qilish hisobiga olinishi lozim bo‘lgan omillar tasdiqlangan bo‘lishi kerak:

1. *O‘zgarish so‘rovini baholash oqibatida o‘zgarish qilish shart emas.* Agar o‘zgarish amalga oshirilmagan bo‘lsa nimani e'tiborga olish kerak. Agar o‘zgarishi xabar tizim xatosi deb jiddiy xato bilan bog'liq kamchilik hisobga olinishi kerak. Tizim xatosi tizimini blokirovka qilishga sabab bo‘lsa, bu juda jiddiy va operatsion buzilish bo‘lishi mumkin, bu xolda o‘zgarishsiz tizimdan foydalaning. Boshqa tomondan etishmovchilikning ta'siri kichik bo‘lsa, ekrandagi turli ranglar muammoni hal qilish uchun muhim emas, o‘zgarishda past bo‘lsada ustuvor bo‘lishi kerak.

2. *O‘zgarish foydalari*. Ko‘p foydalanuvchilari tizimi o‘zgarishida foyda beradi yoki birinchi navbatda, shunchaki bir taklif o‘zgartirish foyda bo‘ladimi?

3. *O‘zgarishi ta'sir foydalanuvchilari soni* bir necha foydalanuvchilar keyin, ta'sir bo‘lsa o‘zgarish past ustuvor tayinlanishi mumkin. Aslida, o‘zgarish qilish maqsadga muvofiq emas, bu tizim aksariyat foydalanuvchilar salbiy ta'sir bo‘lishi mumkin.

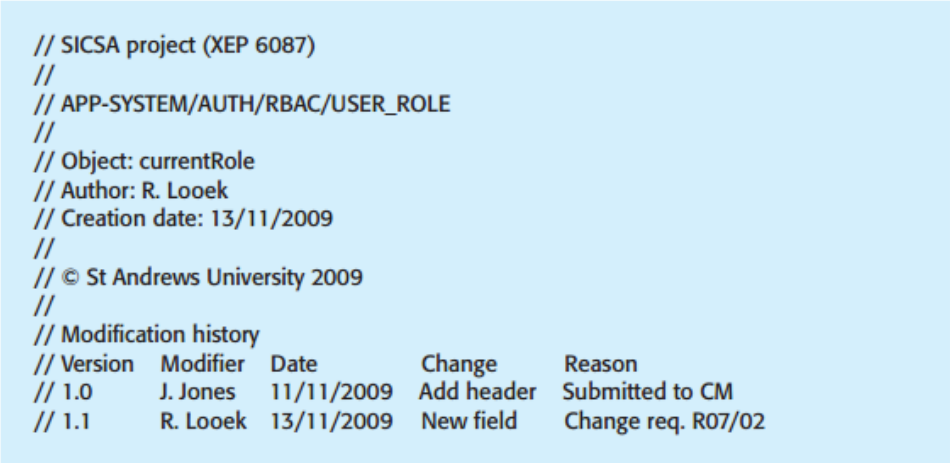
4. *O‘zgarish qilish xarajatlarin*. O‘zgarish qilish ko‘p tizimlar komponentalariga ta'siri bo‘lsa (shunda yangi xatolar joriy etish imkoniyatlari oshadi) va yoki qiymatni oladi, amalga oshirish uchun vaqt ko‘p, o‘zgarish yuqori berilgan keyin uni rad etilishi ham mumkin.

5. *Ishlab chiqarilgan mahsulotlarni aylanishi.* Dasturiy ta'minotni yangi versiyasi nafaqat ozod qilingan bo‘lsa mijozlar uchun u qadar o‘zgarish amalga oshirishni kechiktirish mantiqiy bo‘lishi mumkin. Keyingi rejalashtirilgan ishlab chiqarishlar (4.3- rasmga qarang) dasturiy mahsulotlar uchun zarurdir.

O‘zgarishni boshqarish (masalan, SAD tizimi mahsulot) o‘rniga xususan, muayyan mijoz uchun ishlab chiqilgan tizimlardan ko‘ra, ishlov kerak bir oz boshqacha tarzda amalga oshiriladi. Dasturiy mahsulotlar, mijozlar bevosita ishtirok emas tizimi evolyutsiyasi, shuning uchun mijozning uchun o‘zgarish dolzarb haqida qarorlar qabul ish muammo emas. Bu mahsulotlar uchun o‘zgaruvchi so‘rovlar mijozga kelib qo‘llab-quvvatlash jamoasi, kompaniya marketing jamoasi. Bu so‘rovlar mijozlar tomonidan taklif va mulohazalarni aks yoki mahsulotlari tomonidan taklif tahlillari raqobati bo‘lishi mumkin.

Mijozlarni qo‘llab-quvvatlash jamoasi hasharotlar bilan bog'liq o‘zgartirish talablarini topshirishlari mumkin, deb dasturiy ta'minot bo‘ldi keyin mijozlar tomonidan kashf va xabar qilingan ozod. Iste'molchilar xatolar hisobot uchun veb-sahifa yoki elektron pochtadan foydalanish mumkin. Nuqsonli boshqaruv jamoasi keyin xato hisobotlar tegishli ekanligini tekshiradi va ularni rasmiy tarjima qilishadi. Tizim o‘zgarish so‘rovlar marketing xodimlari mijozlar bilan tanishish va mahsulotlarni reklamasi bilan shug‘ullanadi. Ular buni amalga oshirishlari uchun kiritilishi kerak bo‘lgan o‘zgartirishlarni taklif qilishlari mumkin.

Muammolar va o‘zgarishlarni qayd etilish kerak, lekin faqat individual komponentlar va modul ta'sirdagi o‘zgarishlar mustaqil ravishda baholanadi ularni qayt etish shart emas. Nima uchun shart emas, chunki ular tizim ishlab turgan vaqt bevosita uzatiladi, yoki tizim ishlab ularni qabul qilad. Shu bilan birga, mustaqil hokimiyatda bunday tizim me'moriy baholash sifatida va turli jamoalar tomonidan ishlab chiqarilgan shu tizim modullari ta'sir o‘zgarishlar birinchi o‘ringa bo‘lishi kerak.



4.5- rasm. Manba tarixi

Bunday o‘ta shiddatli dasturlash kabi ba'zi tezkor usullar, yilda mijozlarda bevosita o‘zgarish amalga oshirish kerak yoki yo‘qligida foydalaniladi.

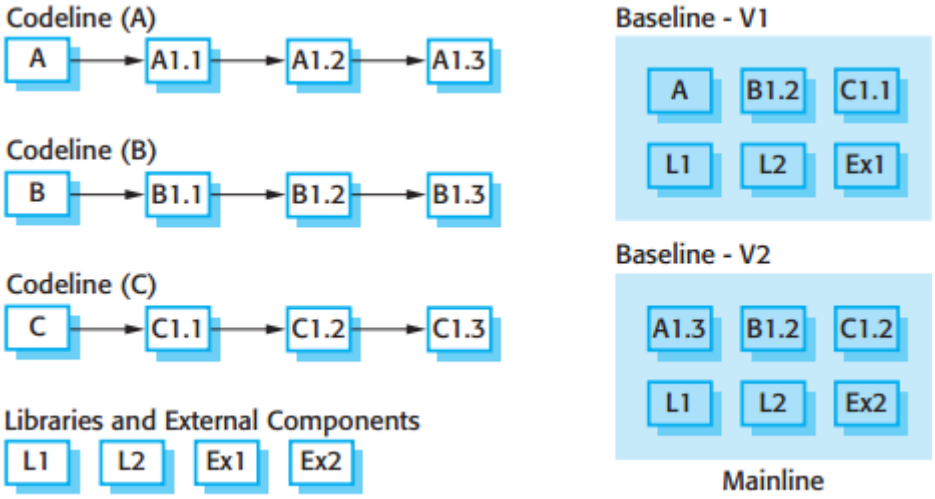
Ular tizim talablariga o‘zgarishlar, takliflar kiritishdi, ular tizimining keyingi o‘sishidan uchun rejalashtirilgan xususiyatlarni ustuvor o‘zgartirish qaror keyin o‘zgarishlar ta'sirini baholash va jamoa bilan ishlash. Shu bilan birga, dasturiy ta'minotni takomillashtirish o‘z ichiga o‘zgarishlar, tizim ustida ish ishlab chiquvchilari taqdir qolgan. dasturiy ta'minot doimo takomillashtirib borilmoqda Refactoring, balki ishlab chiqish jarayonining zaruriy qismi sifatida, xarajatlarni deb hisoblanmaydi.

Ular tizim talablariga bir o‘zgarish taklif bo‘lsa, ular o‘zgarishining ta'sirini baholash va keyingo‘zgarish tizimini keyingi o‘sishi uchun rejalashtirilgan ustivor xususiyatlarini jamoa bilan ishlash.qarorini qabul qiladi. Shu bilan birga, dasturiy ta'minotni takomillashtirish, tizim ustida ish ishlab chiquvchilar ishlarini o‘z ichiga oladi. Dasturiy ta'minot doimiy yaxshilash, qayta tartibga solishni bir yuk deb emas, balki rivojlantirish jarayonining zaruriy qismi sifatida qarashlari kerak.

Rivojlantirish guruhi dasturlar qismlarining o‘zgartiradi, ular har bir tarkibiy qism uchun qilingan o‘zgarishlarni bayonnomada saqlablari kerak. Manbaning kodi qismi manba tarixini saqlab qolish uchun bir yaxshi yo‘li (4.5- rasm). Dasturiy o‘zgarishni amalga oshirish uchun o‘zgarishga so‘rov ma'lumotnoma kerak. Hujjatlar uchun odatda hujjatning oldida alohida qog'ozga muhofaza

qilinadi har bir batianti kiritilgan o‘zgarishlarni qayd qilinadi. Men hujjatlar ustida web-bobdato‘htalib o‘tganman.

O‘zgarishlarni boshqarish odatda maxsus dasturiy ta'minot bilan qo‘llab-quvvatlanadi. Ular masalan, Bugzilla kabi nisbatan oddiy vebga asoslangan vositalari bo‘lishi mumkin, u ko‘p ochiq manba tizimlari bilan bog'liq muammolar haqida xabar berish uchun ishlatiladi. Shu bilan bir qatorda, yanada murakkab vositalar roziligini o‘zgartirish, boshlang'ich mijoz taklifiga o‘zgarish talablarini kiritib butun jarayonni avtomatlashtirish uchun foydalanish mumkin.



4.6- rasm. Kod chiziqlari(Codeline) va asosiy chiziq(baseline)

4.2. Uslubiy boshqaruv

Uslub menejmenti bu turli versiyadagi kompyuter dasturlari yoki konfiguratsion tizimlarni o‘zida saqlovchi proses hisoblanadi va bu tizimdan foydalaniladi. U yana dasturchilarning bir birini uslubini aks ettirmaydigan turli o‘zgarishlarni ham o‘z ichiga oladi. Shu sababli siz uslub menejmentini kodlashtirish va asoslashtirish deb o‘rganishiz mumkin.

4.6 diagrammada kodlar va asoslarning farqlari tasvirlangan. Asosiysi kodlar avvalgi versiyadagi muntazamlik bilan keyingi versiyadagi manba kodlari ketma ket ravishda bog'langan. Kodlar tizimning komponentlarga murojat qiladi, shuning

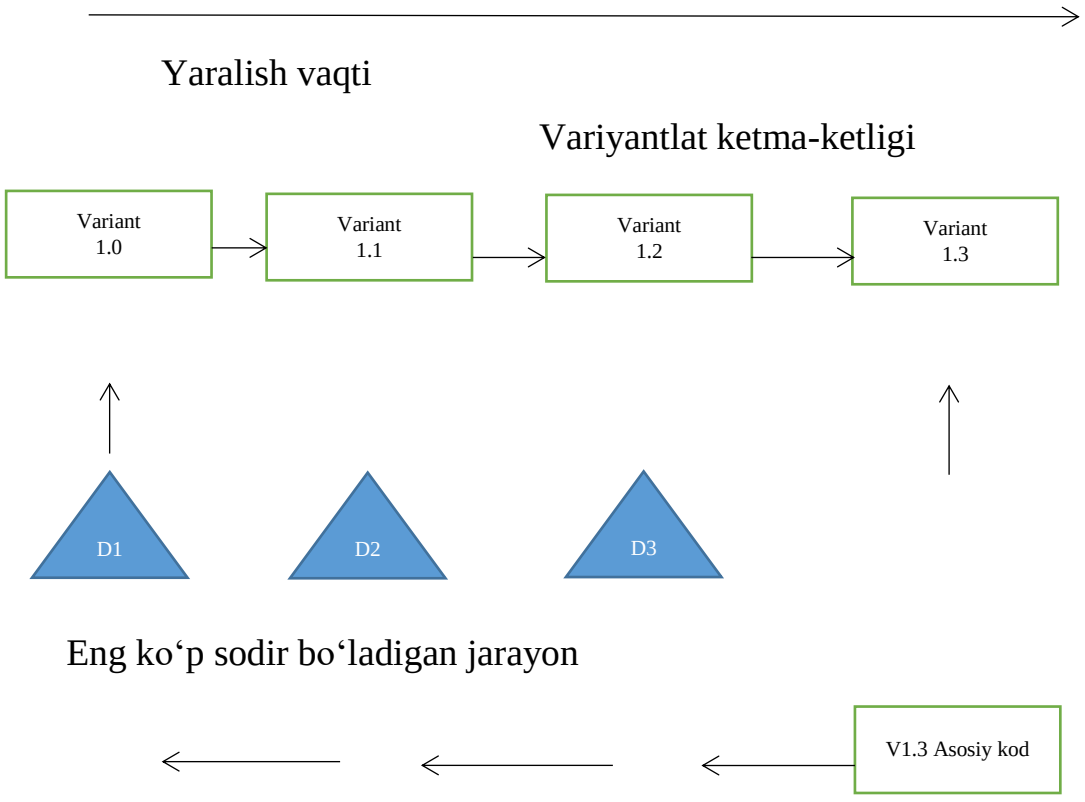
uchun har versiyaning komponentlari turlicha bo‘ladi. Asos bu berilgan tizimning ko‘rinishi. Asos o‘zida jamlagan tizim unga qo‘shimcha foydalaniladigan kutubxonalar va konfiguratsiyali fayllarni va boshqalarni o‘z ichiga oladi. 4.6 diagrammada siz turli asoslarning turli kodlardan foydalanayotganini korishingiz mumkin. Diagrammada biz boyalgan qutilarda asoslarga izoh berilayotganini bilamiz.

Asosga konfiguratsion til orqali ham izoh berishingiz mumkin, qaysiki berilgan tizimning versiyasida qanday komponentlarni o‘z ichiga olganini siz bilasiz. Unda (x1.2 ko‘rsatilgan) komponent versiyasi aniq tavsiflangan yoki oddiy komponent (izohlovchisi) IDsi tavsiflangan. Agar siz tavsiflovchidan foydalansangiz, bu shuni anglatadiki kopgina hozirgi versiyadagi komponent asosda foydalanishi kerak.

Asos muhim qism hisoblanadi, chunki siz tayyor tizimning berilgan versiyasini qayta o‘rnatishizga tog'ri keladi. Misol uchun dastur bir martalik bo‘lishi mumkin, shu sababli har bir ist'emolchi uchun foydalanishi mumkin bo‘lgan alohida tizim kodi bor. Shuning uchun siz iste'molchi tizimda qandaydir xatolik yoki kamchilikdan shikoyat etsa, adminstratorning alohida tayyor diskini olib qaytadan foydalanishingiz kerak.

Bu menejment versiyasini qo‘llab quvvatlash uchun, siz har doim uslub menejmenti uskunalaridan foydalanishingiz kerak.

Bu uskunalar saqlashga va turli versiyadagi komponentlarga kirishga tavsif beradi. Juda kopgina turli versiyadagi menejment tizimsi hozirda mavjud, shular ichida eng ommabop foydalanilgani (VS va Subversion) (Pilato atol 2004, Vespermain 2003) hisoblanadi.



4.7- rasm. Saqlash boshqaruvining detallaridan foydalanish

Menejment tizimlarning versiyasi o‘rtacha bir necha xil ko‘rinishlari bor. Bular:

1. Versiya va Izohlovchini anglab olish. Tayyor versiyalar Izohlovchini tizimga qo‘shilishi bilan aniqlab oladi. Bu Izohlovchilar odatda konfiguratsion nomerlar nomiga asoslanadi masalan (tugmacha menejeri) birdan boshlab ketma ketlikda kop raqamlar bo‘lishi mumkin. Shuning uchun Tugmacha menejeri komponentlari kodlarni 3-versiyasini anglatadi. Ba'zi CM tizimlarda yana versiya bilan birgalikda atributlar asotsiyatsiyasiga ham ruhsat beradi. Misol uchun (telefon, kichik ekran). Ular tavsiflovchi versiyalarda ham foydalana olinadi. To‘g'ri keladigan identifikatsion tizim muhim hisoblanadi,chunki u konfiguratsiyadagi berilgan muammolarni osonlashtiradi. U matni shartli belgilar bilan yozib olishda foydalanishimizni osonlashtiradi. Misol (V2 hamma komponentlarning 2-versiyasi).

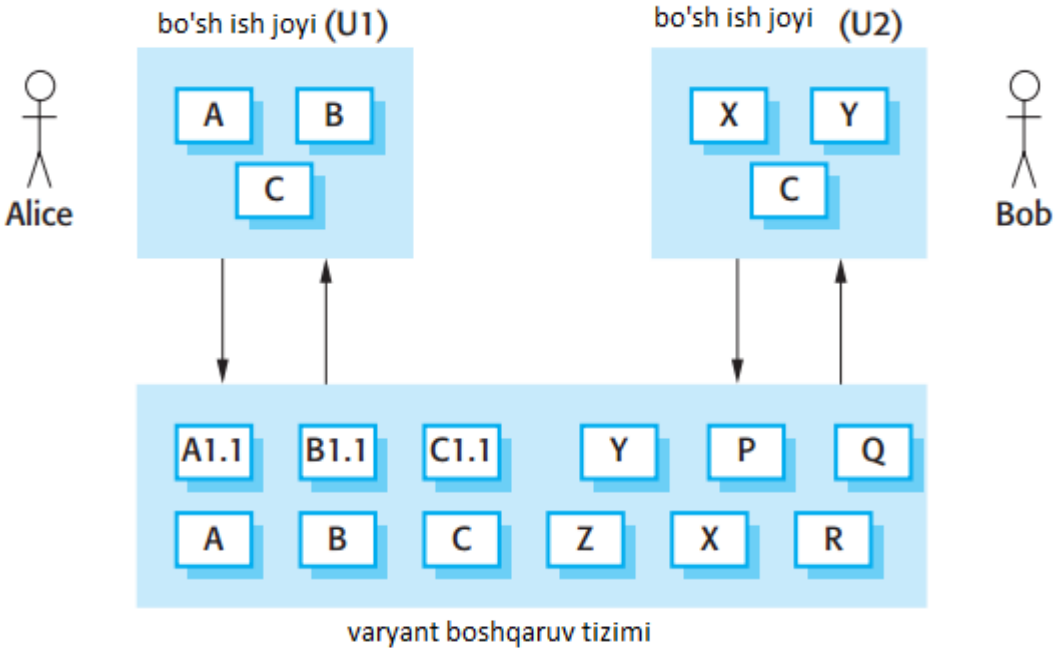
2. Saqlash menejmenti. Tizimlar menejmenti versiyasi odatda saqlash

menejmenti korinishlarini ta'minlaydi ya'ni bir-biridan ozginaga farq qiluvchi komponentlarni ko‘pgina versiyasi saqlash joyiga muhtoj bo‘ladi. Saqlashni o‘rniga nusxasi bilan tizimning bir versiyasining boshqasidan farqlovchi qismini to‘ldirish kerak. Bu ishni qilish orqali aniq versiyalar qaytatdan yaratiladi. Bu 4.7 rasmida tasvirlangan.

3. Tizimning kodiga qilingan har bir o‘zgarishlar va komponentlar yozib boriladi va to‘planadi. Ba'zi tizimlarda bu o‘zgarishlar tizimga kirish bilan ishlab ketadi. Bu o‘z ichiga o‘zgarish qilingan kalit so‘zlar bilan komponentlarni oladi. Siz keyin bu teglarda asosda komponentlarni tanlashda foydalanasiz.

4. Mustaqil rivojlanish turli dasturchilar bir xil vaqtda bir xil komponent ustida ishlashlari natijasida vujudga keladi. Uslub menejmenti tizimsi komponentlarni kuzatishlarini saqlaydi. Turli dasturchilarning o‘zgarish kiritgan komponentlarni qo‘shilishini tekshiradi.

5. Loyihani qollab quvatlash: Uslub menejment tizimsi bir nechta loyihalarni rivojlantirishga xizmat qiladi. Loyihani qollab quvatlash tizimlarda CVS kabi bir vaqtning o‘zida to‘g'ridan to‘g'ri bir fayl bilan ishlashdan ko‘ra loyihaga asoslangan hamma fayllarni tekshirish va topish imkonini beradi.

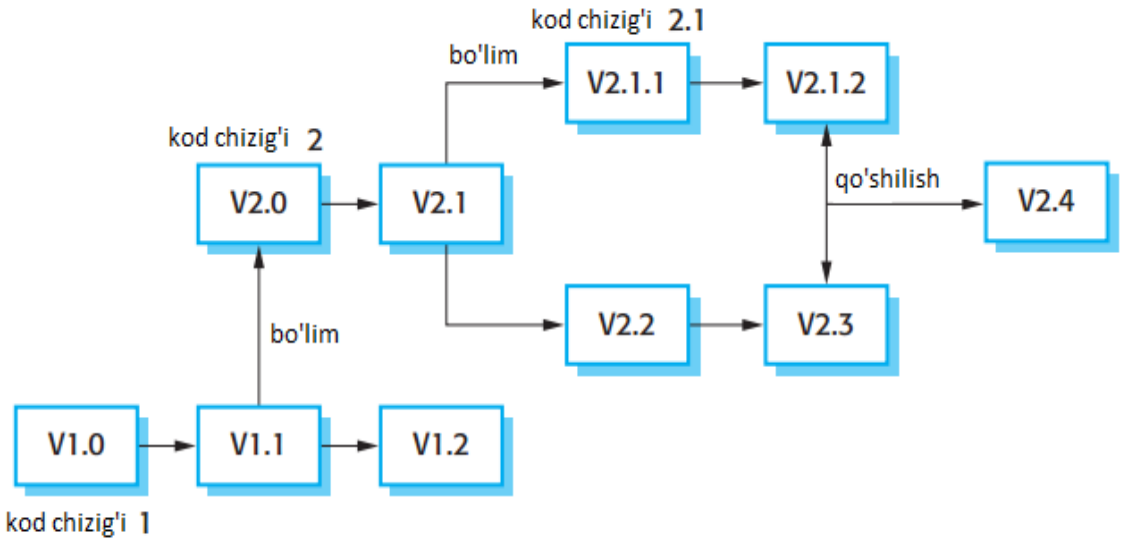


4.8- rasm. Varyantlarni saqlash joyidan tashqariga va ishkariga belgilash

Qachonki uslub menejmenti birinchi marta ixtiro qilinganda, saqlash menejmenti bir muxim funksiyalardan biri edi. Saqlash menejmentini koʻrinishi hamma tizim versiyalarida asosiy disk joylarini saqlashni kamaytirishni nazorat qiladi. Qachonki yangi versiya ixtiro qilinganda tizim oddiy yangi va eski versiyalarni foydalanilgan farqlarni saqlaydi va eski versiyadan yangi versiyani yaratishda foydalaniladi(4.7 diagramma pastki qismida koʻrsatilgan). 4.7 diogrammada boʻyalgan qutichalarda oldingi versiyadagi komponentlarni avtomatik ravishda xozirgi versiyadegi kompnentni qayta ishlashda foydalanildi. Delta odatda oʻzgarishlar listini saqlaydi va avtomatik ravishda bir versiyaga komponentlardan boshqasini ixtiro etiladi. Shunisi aniqki eng oxirgi versiyadagi komponentlar koʻpgina tizimni saqlshda foydalaniladi. Deltalar oldingi versiyalardagi tizimlarni qayta ishlashda foydalaniladi.

Koʻpgina dastur taminotni rivojlantirishda gurush bilan ishlanadi shunday holatlarda tez-tez turli guruh a'zolari bir xil vaqtda bir xil komponent ustida ishlash vujudga keladi; Misol uchun keling Alisa tizimga ba'zi oʻzgarishlar kirityapti deb olaylik. Berilgan A,B,C komponentlarni oʻz ichiga olgan oʻzgarishlar deb olaylik, huddi shu payt Bob ham shu ishni olib boryapti va u X,Y,C komponentlarni oʻzgarishlarini koʻryapti. Alisa ham Bob ham C komponentni oʻzgartirishga olib kelishdi. Shunisi muhimki bu oʻzgarishlar bir biriga aralashuviga olib keladi, ya'ni Bobning oʻzgartirishlari Alisanikini qayta yozilgani yoki ommabop usulidir.

Mustaqil rivojlanishda bir birini aralashuvini qoʻllab quvatlash uchun uslub menejment tizimsi alohida shaxsiy maydon va mahalliy ombor konseptsiyasidan foydalanadi. Dasturchilar komponentlarni mahalliy ombordan to shaxsiy maxsus ish maydoni boʻlgan oraliqda tekshirishadi va bu oʻzgarishlarni oʻzlari xoxlagandek maxsus ish maydoniga kiritishadi qacchonki ularning oʻzgarishlari yakunlanganda omborga kiritishadi. Bu 4.8 diogrammada tasvirlangan 2 yoki undan ortiq odamlar bir xil payt davomida ishlashla har biri buni ombordan tekshirib olishi kerak. Agar komponent tekshirilgan boʻlsa uslub menejment tizimsi oʻrtacha boshqa foydalanuvchilar xohishiga koʻra komponentni tekshirishgani bshqa bir odamni alla qachon tekshirganini koʻrsatib turadi.



4.9- rasm. Bo‘lim va birlashish

Bir komponentni mustaqil rivojlantirish ketma-ketligi kodlar qismida bo‘lishi mumkin. Diogramma 4.9 da bu normal tizim rivojlantirishi bo‘lib har hil dasturchilarning mustaqil ishlashi natijasida turli versiyadagi kodlarni turli yo‘l bilan o‘zgarishini ko‘rsatadi.

Ba'zi qisimlarda kodlarni birlashtirish yangi versiya komponentlarni o‘z ichiga olgan turli o‘zgarishlarni yaratadi u yana 4.9 diogrammada 2.1, 2.2 va 2.3 versiya komponentlarini qo‘shib yangi 2.4 versiyasi yaratilishini ko‘rsatib o‘tilgan. Agar o‘zgarish to‘laligicha boshqa qism kodlarini o‘z ichiga olsa komponent versiyasi avtomatik versiya menejment tizimsiga kodlarni qo‘shadi. Ko‘proq o‘zgarishlar bir-birinin qoplaydi. Dasturchilar ularni mos kelish kelmasligini tekshiradi va o‘zgarishlarni modifikatsiya qiladi. Shuning uchun ular bir vaqtda paydo bo‘ladi.

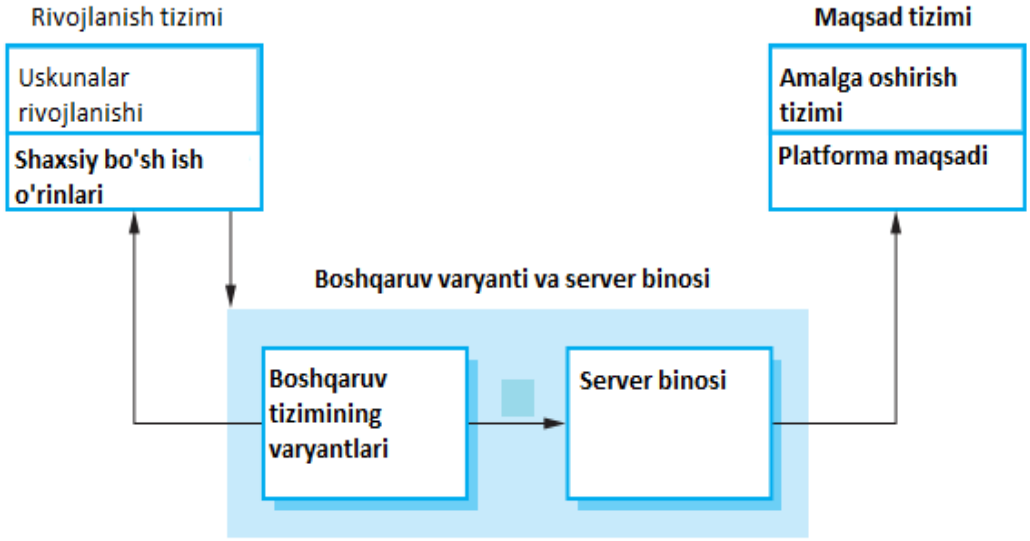
4.3. Tizimning qurilishi

Tizim tuzish bu to‘la, maqbul tizimni tizim komponentlari maxsus kutubxonalar konfiguratsiyon fayllar orqali kompilyatsiya qilish, o‘rnatishdir. Tizim tuzish uskunalari va uslub menejmenti uskunalari tizim tuzishda aloqada bog'liq bolishi kerak. Konfiguratsion tuzilmalar asosni tanishda va tizim tuzish

uskunalarda foydalaniladi.

Tizim tuzish murakkab jarayon bo‘lib u 3 turli tizim platformalarini o‘z ichiga olgan bo‘ladi (4.10 rasm).

1. Tizim rivoji deganda uskunalarni o‘zgarishi kompilyatorni kod o‘zgartirgichlarni rivojlanishi tushuniladi. Ixtirochilar tizimga o‘zgartirish kiritishdan oldin kodlarni uslub menejment tizimsida tekshirib maxsus ish joyiga qo‘yishadi.



4.10- rasm. Rivojlanish, bin ova maqsad platformalari

Ular o‘zgarishni qabul qilishdan oldin balkim tizim tuzishda muhitgaham etibor berishni xoxlashadi ya'ni uslub menejment tizimsida bu o‘z navbatida lokal tuzish uskunalardan foydalaniladi va undan komponentlar versiyasini maxsus ish joyida tekshirishadi

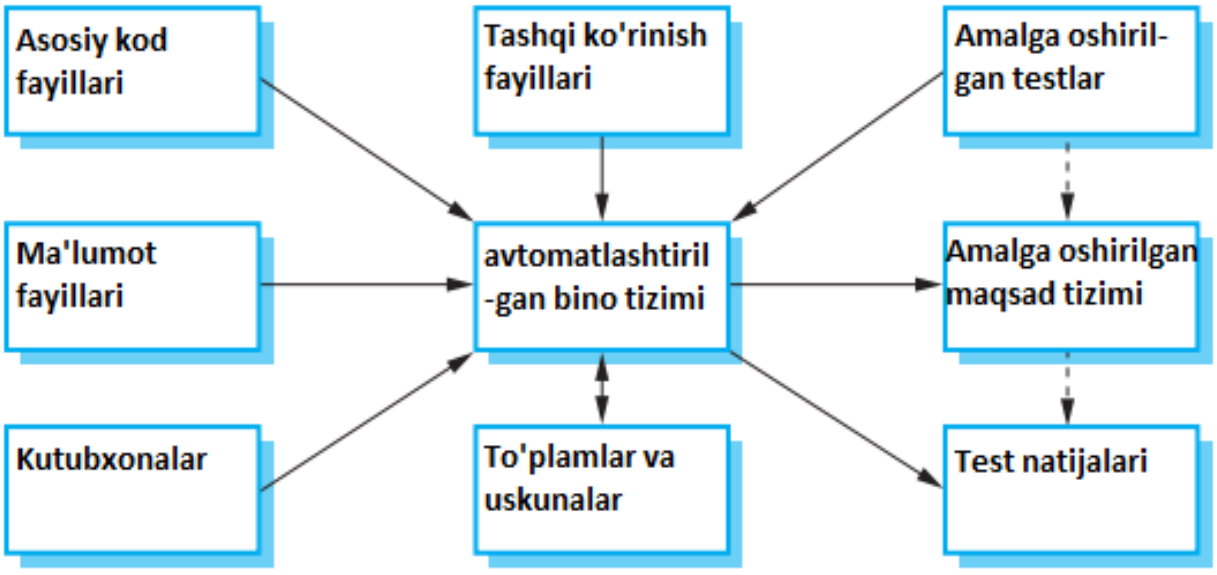
2. Tuzilgan server tizim versiyasining eng maqbul ifodalovchi foydalanuvchi qismidir. Bu menejment tizim versiyasiga juda ko‘p yordam beradi. Dasturchilar uni o‘rnatishdan oldin kodni uslub menejment tizimsiga kiritishadi tizim uslub menejment tizimsiga kiritilmagan kutubxonaga bog'liq bo‘ladi.

3. Berilgan muhit tizimning ijrosini ifodalaydi bu bir turdagi kompyuterlar rivojlantirishda va tizim o‘rnatishda foydalaniladi. Hattoki xozirgi vaqt uchun rivojlanayotgan muhitga qaraganda osonroq va kichikroq tez-tez bo‘ladigan mobil

qurilmalar hisoblanadi; misol uchun: telefon katta tizimlar uchun o‘z ichiga kunlar hisobi va boshqa ko‘pgina tizimlarni kiritishi mumkin. Lekin ularni hozirgi rivojlangan mobil qurilmalarga o‘rnatib bolmaydi. bu ikki holatda tizimni o‘rnatish va kompyuter rivoji uchun testdan o‘tkazish mumkin.

Tizimni rivojlantirish va server yaratish uslub menejment tizimsida bir biriga bog'liq. Uslub menejment tizimda host sifatida qurilgan serverda qolishi yoki o‘zi uchun alohida server yaratishi mumkin. Ichki tizimlar uchun esa simulyatsion muhit rivojlangan mobil qurilmalarga o‘rnatilishi mumkin. Masalan: aktiv tizim platformalardan foydalanganda bu samaraliroq.

Tizim o‘rnatish dastur haqida katta miqdordagi ma'lumot va uning ishlashi haqida ma'lumotni o‘z ichiga oladi. 4.11 rasm(figura)da hattoki kichik tizimlardan iborat har bir dasturchi tizim o‘rnatishda avtomatik ishlaydi shuni bilingki siz uchun uni o‘rnatishda manba kod fayllari zarur emas. Siz buni zaruriy taminlovchidan olishingiz mumkin. Siz kompilyatorni versiyasini va boshqa dasturiy uskunalarni izohlashingizga to‘g'ri keladi. Chunki bu tizimni o‘rnatishda foydalaniladi. Fikran siz kompyuter tizimsini oddiy bir komanda berish orqali yoki sichqonchani ezish orqali o‘rnata olishishingiz mumkin.



4.11- rasm. Tizimning qurilishi

Juda ko‘p o‘rnatiladigan tizimli dasturlar bo‘lib siz tizimni o‘rnatishda

hammasini taminlashingiz yoki quyidagi koʻrinishlarni berishingiz mumkin:

1. Agar zarur boʻlsa tizim tuzishda siz dasturni analiz qilishingiz kerak ya'ni tegishli komponentlarni izoxlash avtomatik generatsiya qilish(ba'zan konfiguratsion fayl deymiz) tizim yana qoʻlda tayorlash yoki oʻrnatishda oʻzgarish kirita olishni ham taminlashi lozim.

2. Uslub menejment tizimsida integratsiya tizim oʻrnatishda uslub menejment tizimsidan berilgan versiya komponentlari tekshiriloshi lozim.

3. Qoʻlda oʻrnatish: tizim oʻrnatishda siz qaysi kodlar qaytadan oʻrnatilishi kompilyatsiya boʻlishini bilishingiz kerak.

4. Oʻrnatiladigan tizimni tayorlash: tizim oʻrnatishda sizda kod fayllarni bir biri bilan boshqa fayllar kutubxona va konfiguratsion fayllarni yaratib oʻrnatish kerak.

5. Avtamotlashgan test: ba'zi oʻrnatilgan tizimda avtomatlashgan test uskunalaridan uni avtomatik ravishda tekshirish mumkin bu tekshirish oʻrnatishda yuzaga keladigon oʻzgarishlarni oldini oladi.

6. Xabar: oʻrnatilgan tizim muvoffaqiyatli yakuniga yetdi yoki yetmaganligi xabarini berilishi lozim.

7. Tizim oʻrnatilganda alohida u haqda ma'lumot beruvchi oyna boʻlishi lozim.

Tuzish yozuvlari tizim oʻrnatish toʻg'risida ma'lumotlardir u oʻz ichiga kompanentlar va ularga tegishlilik yana qoʻshimcha versiyada tashkil etilgan uskunalar haqida malumotni qamrab oladi. Oʻrnatish haqida ma'lumotlar berib borish, izoxlash tili va konfiguratsion til hisoblanadi. Konfiguratsion til oʻz ichiga konstruktorlar, tizim kompanentlari va ularga tegishli maʻlumotlarni oladi.

Kompilyasiya judayam tez jarayon boʻlib uskunalar tizim oʻrnatishda kompilatsiyalar sonini kamaytirishga yordam beradi. Ular buni kompiletsiyalangan versiyalarni borligini tekshirish orqali amalga oshirishadi. Shuning uchun qaytadan kompiletsiya qilish zarur emas.

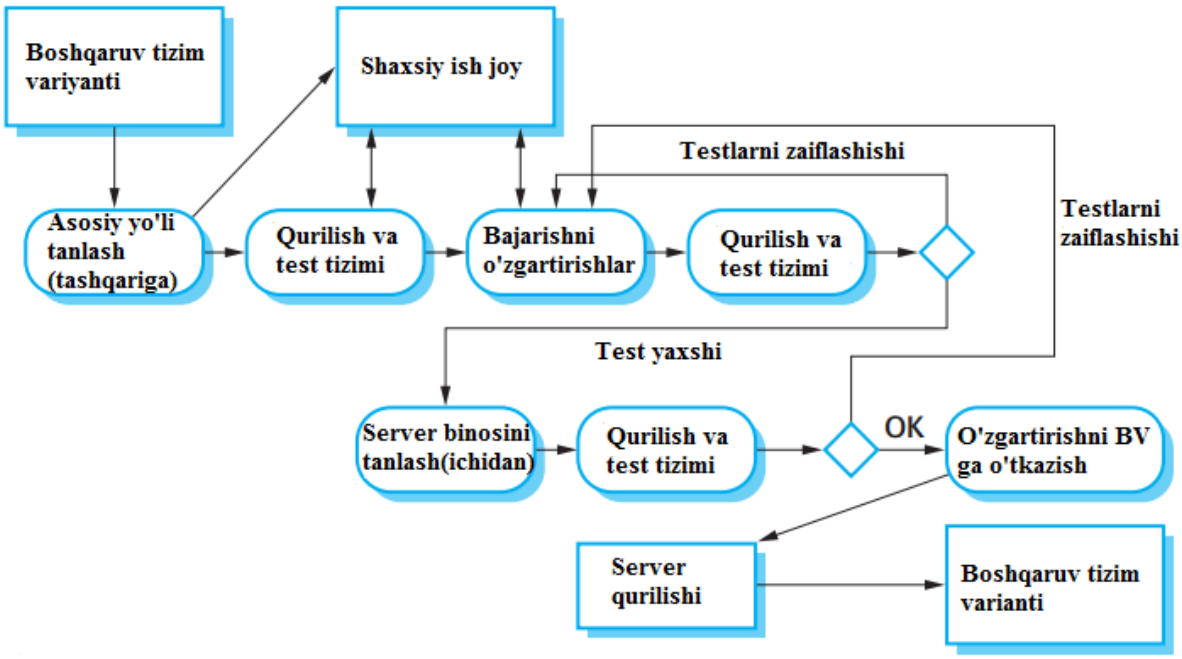
Ikki turdagi imzodan foydalanishimiz mumkin:

1. *Madifikatsion vaqti shtampi*. Dastur kod fayl imzosi bu vaqt va kun hisoblanib, qachonki fayl indifikatsiyasi bo‘lganda ishga tushadi. Dastur kodi fayli kompanenti madifikatsiya bo‘lgandan keyin dastur kod fayli tizimga agar zarur bo‘lsa yangi kod faylini yaratadi.

Misol uchun aytaylik kompanentlar va comb.java comb.class va ular 17:03:05:02:04:2009 va 02:12:2009 da madifikatsiya bo‘lgan deylik. Bu java kodi 14-fevral 2009-yilda soat 5 dan 3 minutu 16:34:25: sekund o‘tganda madifikatsiya bo‘lganini bildiradi. Kompilatsiya versiyasi esa 12-fevral 2009-yilning 4 dan 34 minut 25 sekund o‘tgandagi madifikatsiyasini bildiradi. Bu xolatda tizim avtamatik ravishda comp.javani kompilatsiya qiladi chunki kompilatik versiyasi 12-fevraldagi dastur kodlarini o‘z ichiga oladi.

2. *Dastur kodini tekshiruvchi funksiya*. Bu dastur kodi fayli yaratilgan kundanboshlab tekshirishni boshlaydi.Bu funksiya dastur matinlarini ham kiritilgan son sifatida foydalanadi va hisoblaydi. Agar siz dastur kodini o‘zgartirsangiz, bu yangi tekshirish funksiyasini yaratadi. Siz shuni yahshi bilingki har bir dastur kod fayli uchun alohida tekshirish funksiyasi mavjud. Tekshirish funksiyasi dastur kodi komplyasiya bo‘lishdan oldin tekshirib oladi va u haqida ma‘lumotberadi. Tizim o‘rnatilgamdan keyin, yangi teglar obyekti kod faylini generatsiya qiladi. Agar kod fayli mavjud bo‘lmasa dasturdastur kodlari tizimga qo‘shilgan bo‘ladi. U holda qaytadan komplyatsiya qilish muhim hisoblanadi.

Obyekt kodining fayllari odatda versiyalanmagani uchun tizimda obyektning eng oxirgi kompilatsiya qilinga fayli tizimda saqlanadi. Bu odatda dastlabki kodning fayli bilan nomi boyicha bog‘lanadi (bu degani uning nomi dastlabki kod fayli bilan bir-xil lakin boshqa sufikas bilan). Shunday qilib COMP Java fayli Comp klas faylini generatsiyalash mumkin. Dastlabki va obyekt fayllari nomi bo‘yicha bog‘langanligi uchun bitta kataligda dastlabki kodning har xil versiyani saqlash mumkin emas chunki ular shunday nomlangan obyekt fayllarini yaratishadi.



4.12- rasm. Uzluksiz integratsiya

Nazorat yig‘indisi yondashuvi komponentni obyekt kodining har-xil versiyasini bir vaqtda quvvatlashga imkon beradi. Manba va obyekt kodini fayl nomi emas balki imzo bog‘lab turadi. Dastlabki kod va obyekt kodini fayllari bir-xil signatiraga ega bo‘ladi. Shining uchun siz komponentni qayta komplizatsiya qilganizda u vaqtincha belgili bo‘lganda obyekt kodini qayta yozmaydi. U obyetk kodini yangi faylini yaratadi va dastlabki kod imzosi bilan belgilaydi. Parallel kompliizatsiyada komponentning har-xil versiyasi bie vaqtda kompilatsiya qilinishi mumkin.

Tezkor usullarda tizim yig‘ilganda dasturiy ta‘minotdagi muammolarni aniqlash uchun avtomatlashtirilgan testlar o‘tkazilishi kerak(bazida tutin testlar deb nomlanadi). Tez-tez o‘tkaziladigan qurishlar 4.12-rasmda ko‘rsatilgandek uzliksiz integratsiya jarayonining qismi bo‘lishi mumkin. Tezkor usullar tushunchasiga binoan uzliksiz integratsiyada dastlabki kodni kichik o‘zgartirishlar miritilgandan keyin asosiy yo‘nalishga qaytish ko‘zda tutiladi. Uzluksiz integratsiyani bosqichlari quyidagilar:

1. Yaratuvchilarning ishchi joyida versiyalarni boshqarish tizimining magistral tizimini tekshiring.

2. Tizimni quring va avtomatlashtirilgan testlarni ishga tushurining, o‘rnatilgan tizim hamma testdan o‘tishini tekshiring. Agar o‘tmasa tizim yig‘ilgani noto‘g‘ri hisoblanadi va siz bazaviy tizimni kim oxirgi tekshirganini aniqlashingiz kerak bo‘ladi. Ular bu muammoni bartaraf etishiga javobgar bo‘ladilar.

3. Tizim komponentlariga o;zgartirish kiritish.

4. Xususiy ishchi o‘rinda tizimni quring va yana testdan o‘tkazing. Testdan yana o‘ta olmasa muharrirlashni davom ettiring.

5. Tizim sinovdat o‘tgandan so‘ng uni qurigan tizim deb belgilang lekin baza yabgi tizim sifatida emas.

6. Tizimni serverda quring va testlarni ishga tushiring. Siz tizimni tekshirganiz uchun boshqa komponentlar o‘zgartirilgan bo‘lsa shu ishni bajarishingiz kerak. Bu komponentlarni ham hususiy ishchi joyida tekshiring va muharrirlang.

7. Agar tizim testlarni qurilgan tizimga o‘tkazib yuborsa magistral yangi baza sifatida kiritilgan o‘zgarishlarni saqlab qolishingiz kerak.

Uzluksiz integratsiya afzalligi shundaki har-xil odamlar tomonidan yaratilgan komponentlar o‘rtasidagi muammolar ular paydo bo‘lishi bilan bartaraf etiladi. Magistral yo‘nalishidagi eng oxirgi tizim yakuniy tizim bo‘lib qoladi. Uzliksiz integratsiya yaxshi g‘oya bo‘lganiga qaramay tizimni qurishda bu yondashuvda qo‘llashni har doim ham ilojisi bo‘lmaydi. Buning sabalari quyidagilar:

1. Tizim juda katta bo‘lsa uni yaratish va testdan o‘tkazish jo‘p vaqt talab etadi. Shuning uchun uni kuniga bir necha marotaba qurish maqsadga muofiq emas.

2. Ishlab chiqarish platformasi mo‘ljallangan platformadan farq qilsa yaratilayotgan joyda tizimni sinovdan o‘tkazishni keragi yo‘q. Apparat taminoti, opetarsion tizim yoki dasturiy ta‘minotda farqlar bo‘lishi munkin. Demak tizimni testdan o‘tkazishga ko‘proq vaqt ketadi.

Katta tizim yoki platformasi boshqacha bo‘lgan tizimlar uchun uzluksiz integratsiyadan foydalanish maqsadga muofiq emas. Bu sharoitlarda tizimni har kuni yig‘ish yaxshi natija beradi buning hususiyatlari quyidagicha:

1. Tashkilot tizim komponentlarini yaratish uchun muddat belgilaydi. Agar yaratuvchilar komponentlarning yangi versiyalarini o‘sha vaqtga yetkazib berishi kerak. Komponentlar to‘liq bo‘lmasligi mumkin lekin tekshirish uchun bazi funksional imkoniyatlarga ega bo‘lishi kerak.

2. Tizimning yangi versiyasi bu komponentlarni birlashtirib to‘liq tizim shakillantiriladi.

3. Bu tizim sinov jamoasiga beriladi, ular tizimni sinovdan o‘tkazadilar. Shu paytda yaratuvchilar komponentlar ustida ishlaydi, funksional imkoniyatlar qo‘shadi va avvalgi testlarda aniqlangan hatoliklarni kuzatadi.

4. Test vaqtida aniqlanga hatolar rasmiylashtiriladi va tizim yaratuvchilariga beriladi. Ular bu hatolarni komponentning keying versiyasida bartaraf etadilar.

Dasturiy ta‘minotni tez-tez yig‘ishni afzalliklari shundaki, komponentlarni o‘zaro ta‘sirida paydo bo‘ladigan muammolar jarayonni boshida aniqlanadi. Tez-tez qurish component modullarini batafsil tekshirishni ko‘zda tutadi. Yaratuvchilar “qurishni sindirmaslik” uchun psixologik bosimda bo‘ladilar bu degani ular butun tizimni muavvaffaqqiyatsizlikka uchratadigan komponentlar versiyalarini tekshirishdan qochadilar.

Shuning uchun ular yaxshi tekshirilmagan komponentlarning yangi versiyalarini taqdim etishni hohlamaydi. Demak yaratuvchi tomonidan aniqlanishi mumkin bo‘lgan dasturiy ta‘minot hatoliklariga tizimni testlash vaqtida kamroq vaqt sarflanadi.

4.4 Foydalanishga chiqarishni boshqarish

Tizimli reliz – mijozlarga tarqatilgan dasturiy ta‘minot tizimining versiyasi. Omma uchun – dasturiy ta‘minot bozori, odatda relizning ikkita turi bor: asosiy relizlar ular yanfi funksionalaga ega bo‘ladi va kam miqdordagi relizlar ular

hatolarni to‘g‘rilaydi va e’lon qilingan foydalanuvchilar muammosini bartaraf etadi. Masalan bu kitob Apple Mac kompyuterida yozilyapti unda OS 10.5.8 operatsion tizimi o‘rnatilgan. Bu OS 10 tizimining asosiy 5 relizining qo‘shimcha 8 relizi. Asosiy relizlar sotuvchi uchun juda muhim chunki mijoz ularni pulga sotib oladi. Qo‘shimcha relizlar odatda bepul tarqatiladi.

Odatiy dasturiy ta‘minot yoki bir-xil yo‘nalishdagi dasturiy maxsulotlarning relizini boshqarish tizimi murakkab jarayondur. Tizimning maxsus relizlari har bir mijoz uchun alohida ishlab chiqilishi mumkin va alohida mijozlar bir vaqtni o‘zida tizimning bir nechta har-xil relizini boshqarishi mumkin. Bu degani dasturiy ta‘minot ishlab chiqaruvchi tashkilot maxsuslashtirilgan dasturiy maxsulotni sotib bu maxsulotning o‘nlab yoki yuzlab hat-xil relizlarini boshqarishiga to‘g‘ri keladi. Tashkilotda mijozdagi tizimning relizlari, reliz va tizim versiyalari o‘rtasidagi munosabatlar haqidagi ahborotni saqlovchi tizim bo‘lishi lozim. Muammo paydo bo‘lganda mijozga berilgan dasturiy ta‘minotni tiklash kerak bo‘lishi mumkin.

Shuning uchun tizimli reliz bajarilganda u ro‘yhatga olinishi kerak kelajakda uni yangilash imkonini kafolatlash uchun. Bu ayniqsa mashinalar kompleksini boshqaruvchi uzoq muddatga o‘rnatiladigan tizimlar uchun muhim. Mijozlar bu tizimning yagona relizini ko‘p yil davomida ishlatishi va yaratilgandan ancha keyin dasturiy ta‘minotga ma‘lum o‘zgartirishlarni kiritishni talab qilishi mumkin.

Relizni ro‘yhatga olish uchun siz bajariluvchi kodni yaratishda ishlatgan komponentning dastlabki kodining ma‘lum versiyalarini yozishingiz kerak. Siz dastlabki kod fayllarini hamma ma‘lumotlarini va komfiguratsiya fayllari nushasini saqlab qolishingiz kerak. Shuningdek siz dasturiy ta‘minot yaratishda ishlatilgan operatsion tizimlar versiyalari bilan kutubxona kompleator va boshqa instrumentlarni saqlab qolishingiz kerak. Ular huddi shu tizimni biroz kechroq sana bilan yaratishda ishlatish mumkin. Bu degani maqsadli tizimning dastlabki kodi bilan birga platform va instrumentlarni dasturiy ta‘minotning nushasini ham olishingiz kerak.

Tizimli relizni tayyorlash va tarqatish qimmatbaxo jarayondur ayniqsa dasturiy ta‘minot ommaviy bozorining uchun. Reklamani materiallarini shunday

tayyorlash kerakki mijozni tizimning yangi ko‘rinishini olishga undashi kerak. Relizga ko‘nikish uchun vaqt berish kerak. Agar relizlar ortiqcha tez chiqarilsa yoki apparat vositalarini talab qilsa mijozlar yangi relizga qiziqmasligi mumkin ayniqsa uni sotib olmoqchi bo‘lsa. Agar tizimli relizlar juda kam chiqarilsa bozordagi o‘rnini yo‘qotishi mumkin chinki mijoz alternative tizimlarga o‘tib ketishi mumkin.

Omil	Tasnif
Tizimning texnik sifati	Ko‘p mijozlar ishlatadigan tizimga muhim hatolik haqida habar berishganda hatoliklar bartaraf etilgan relizni chiqarish mukin. Unchalik muhum bo‘lmagan hatolarni bartaraf etish uchun tizimning qisimlari chiqarilishi mumkin.
Platformani o‘zgarishi	Opertasion tizimning yangi verisiyasi chiqarilganda siz ilovaning yangi relizini chiqarishingizga to‘g‘ri kelsa kerak.
Lehmanning 5 qonuni (9 bobni ko‘ring)	Bu ‘qonun’ shuni nazarda tutadiki siz tizimga kata hajmdagi yangi funksionallikni qo‘shayotgan bo‘lsangiz siz keyingi relizda qo‘shishingiz mumkin bo‘lgan funksionalning miqdorini cheklovchi hatoliklarni ham nazarda tutishingiz kerak. Shuning uchun yangi funksionali ko‘p bo‘lgan tizimli reliz odatd muammolarni bartaraf etuvchi va ishni yaxshilovchi reliz bilan kuzatiladi.
Musobaqa	Raqibingiz o‘z maxsulotlariga yangi ko‘rinishlar qo‘shgan, siz ham maxsulotingizga yangicha ko‘rinishlar qo‘shmasangiz o‘z mijozlaringizni yo‘qotishingiz mumkin.
Bozor talablari	Tashkilotning marketing bo‘limi relizlar alohida sanagacha ishlashini ta‘minlashni o‘ziga oladi.
Mijoz talablarining o‘zgartirishi	Pullik tizimlar uchun mijozlar o‘zgartirish takliflarini bildiradilar va bu o‘zgartirishlar amalga oshirilgan tizimni kutadilar.

4.13- rasm. Tizim ko‘rinishlarini rejalashtirishga ta‘sir etuvchi omollar

Tizimning yangi versiyasini qachon chiqarishni rejalashtirishda siz etiborga olishingiz kerak bo‘lgan har-xil texnik va tashkiliy omillar 4.13 rasmdda ko‘rsatilgan.

Tizimli reliz faqat tizimning bajariluvchi kodi degani emas. Relizning ichiga shuningdek quyidagilar kirishi mumkin:

- Relizni o‘rnatilishini belgilovchi konfigratsion fayllar;
- Ma‘lumotlar fayllari, tizimi muovaffaqqiyatli ishlashi uchun hatolar haqidagi habarlarni saqlovchi fayllar;
- Instialatsiya, u tizimni apparat vositalarga o‘rnatishga yordam beradi;
- Tizimga tasnif beruvchi electron va qog‘oz hujjatlar;
- Bu reliz uchun ishlab chiqilgan qadoq va reklama;

Relizni yaratish – tizimli relizning hamma kopanentlarini ichiga olgan fayl va hujjatlar to‘plamini yaratish. Dasturning bajariluvchi kodi va unga biriktirilgan ma‘lumotlar fayillari variantlarni boshqarish tizimida ro‘yhatga olinishi va relizni raqami bilan belgilanishi kerak.

Konfiguratsiya tasniflari har-xil appatar vositalari va operatsion tizimlari uchun yozilishi kerak va mijozlar ularni o‘z tizimlarida o‘rnatishi uchun yo‘riqnimalar tayyorlashi kerak. Agar mashina o‘qiydigan yo‘riqnamalar bo‘lsa ularning electron nusxalari dasturiy ta‘minot bilan berilishi kerak. Dasturni o‘rnatish skriptlari ham yozilishi kerak. Vanihoyat hamma axborot tayyor bo‘lganda dasturiy ta‘minotning savdo belgisi tayyorlanishi va mijoz yoki savdo nuqtalarida tarqatilishi kerak.

Yangi tizimli relizlarni o‘rnatilishini rejalashtirayotganda siz mijozlar yangi tizimli relizlarni doim ham o‘rnatmasligini nazarta tutishingiz kerak. Bazi foydalanuvchilar mavjut tizimda mamnun bo‘lishi mumkin. Ular yangi rrelizdagi o‘zgarishlar ortiqcha ssarf harajatga loyiq emas deb hisoblashlari mumkin. Shuning uchun tizimni yengi relizini yaratayotganda oldingi relizlarrga asoslanmasligi mumkin. Bu muammoni namoyish etish uchun quyidagi stenariyni ko‘rib chiqamiz:

1. Tizimning birinchi relizi tarqandi va foydalanishga qo‘yildi.

2. 2-reliz yangi fayillarni o‘rnatishni talab qiladi lekin mijozlar 2 relizdagi vositalarga extiyoj sezmaydi va birinchi relizni qoldiradi.

3. 3-reliz 2-relizda o‘rnatilgan fayllarni talab qiladi va o‘zida xech qanday yangi fayl olib kelmaydi.

Dasturiy ta‘minot distribyutri 3-relizga kerak bo‘lgan fayillarni hamma joyda o‘rnatilgan deb xisoblash kerak emas. Bazi joylarda birinchi relizdan bevosita 3-relizga o‘tishlari mumkin. Bazi joylarda o‘zidagi sharoitdan kelib chiqib 2-relizdagi fayllarni o‘zgartirgan bo‘lishi mumkin. Shuning uchun ma‘lumotlar saqlangan fayllar tizimning 3-relizi bilan tarqatilishi va o‘rnatilishi kerak.

Dasturiy ta‘minotning yangi relizlari bilan bog‘liq marketing va qadoqlashga ketadigan sarflar shunaqa kattaki maxsulot yaratuvchilar odatda yangi platformalar uchun yangi relizlar yaratadi yoki muhumyangi funksionallikni qo‘shishadi. Bu yangi dasturiy ta‘minot uchun ular foydalanuvchilarni ayiblaydilar. Mavjud relizda muammolar aniqlanganda yaratuvchilar ularni bartaraf qiluvchi qismni yaratadilar va mijozlar uni web saytdan yuklab oladi.

Yuklab olinadigan qismlarni ishlatishni muammosi shundaki, ko‘p mijozlar muammoni bartaraf etuvchi bunday tuzatishlar mavjudligini va ular nima uchun o‘rnatishligini tushunmaydilar. Buning o‘rniga ular o‘zlaridagi hato ishlaydigan tizimdan foydalanishni davom etadilar va o‘z bizneslari uchun havf yaratadilar. Bazi holatlarda tuzatishlar havfsizlikni ta‘minlash uchun ishlab chiqilganda ularni o‘rnatishda muavfaqqiyatsizlikka uchraganda, biznes tashqaridan bo‘lagn hujumlarga himoyasizligini bildiradi. Bu muammoni bartaraf etish uchun Adobe, Apple va Microsoft kabi ommaviy bozorga dasturiy ta‘minot maxsulotlarini taqdim etuvchi tashkilotlar odatda avtomatik yangilanishni yo‘la qo‘ydilar, bunda relizning yangi kichkina o‘zgarish bo‘lsa ham tizimlar o‘zi yangilanib oladi. Lekin pullik tizimlar uchun bunday qilinmaydi, chunki bu tizimlar hamma mijozlar uchun standart versiyada mavjud bo‘lmaydi.

Asosiy tushunchalar

- Konfiguratsiyani boshqarish bu dasturiy ta‘minot tizimini rivojlanishini boshqarishdur. Tizimni quvvatlashda konfiguratsiyani boshqarish jamoasi tizimga kiritilayotgan o‘zgarishlarni nazoratini ta‘minlaydi va xisobga olib boradi.

- Komfiguratsiyani boshqarishni asosiy jarayonlari o‘zgarishlarni boshqarish, versiyalarni boshqarish, tizimni qurish va relizlarni boshqarish bilan bog‘liq. Hamma jarayonlarni quvvatlash uchun dasturiy vositalar mavjud.
- O‘zgarishlarni boshqarish deganda tizimni o‘zgartirish bo‘yicha mijozlar va boshqa tegishli tomonlarni takliflarini baxolash va ular maqul kelsa tizimning yangi versiyasida qo‘llash tushuniladi.
- Versiyalarni boshqarish deganda o‘zgarish sifatida dasturiy komponentlarni har-xil versiyalarini kuzatish tushuniladi.
- Tiizmni qurish deganda komponentlarni maqsadli kompyuter tizimida ishlaydigan bajariluvchi dasturga yig‘ish jarayoni tushuniladi.
- Dasturiy ta‘minot tez-tez qayta qurilishi va yangi versiya paydo bo‘lishi bilan tajribada sinalishi kerak. Bunda oxirgi qurishdan keyin paydo bo‘lgan hat ova muammolarni aniqlash osonroq bo‘ladi.
- Tizimli relizlar o‘z ichiga bajariluvchi kod, ma‘lumotlar fayllari konfiguratsiya fayllari va hujjatlarni oladi. Relizlarni boshqarush deganda relizni muddati haqida qaror qabul qilish, tiizmning har bir relizi uchun tarqatiladigan axborot va hujjatlarni tayyorlash tushuniladi.

Nazorat savollari:

1. Agar tashkilot KB ning samarali siyosati va jarayonlarini rivojlantirmasa 5 ta paydo bo‘lishi mumkin bo‘lgan muammolarni ayting.

2. O‘zgartirishni boshqarish jarayonida markaziy hujjat sifatida o‘zgartirishga so‘ro‘v berish shaklidan foydalanish afzalliklari qanaqa?

3. O‘zgartirishni boshqarish jarayonlarini quvvatlash uchun vosita sifatida kiritilishi kerak bo‘lgan 6 ta muhum belgilarni tasniflab bering.

4. Komponentning har bir versiyasi nima uchun yakka ma;noda aniqlanishi kerakligini tushuntiring. Faqat versiya raqamiga asoslangan versiyalarni identifikatsiyalash sxemasidan foydalanish paydo bo;lgan muammolarni izohlang.

5. Tasavvur qiling ikkita yaratuvchi bir vaqtning o‘zida 3 ta har-xil dasturiy komponentni o‘zgartirmoqda. Ular bajargan o‘zgartirishlarni birlashtirishda qanday muammolar paydo bo‘lishu mumkin?

6. Dasturiy ta‘minot hozir har – xil joylarda ishlatilayotgan jamoa tomonidan yaratiyapti. Dasturiy ta‘minotni bunday yaratish sharoiti uchun versiyalarni boshqarish tizimida kerak bo‘lgan funksiyalarni taklif eting.

7. Tizimni uni komponentlaridan qurishda paydo bo‘lishi mumkin bo‘lgan muammolarni tasniflab bering. Tizim host kompyuterda qaysidur mashina uchun qurilganda qanday moammolar paydo bo‘lishi mumkin?

8. Tizimni qurushga kelganda siz tushuntiringchi nima uchun katta dasturiy tizimlar ishlab chiqilgan eski kompyuterlarni saqlash kerak bo‘ladi?

9. Tizimni qurishning umumiy muammosi shundaki tizim kod va fayllar tuzilmasida ishlatilgan fayllar nomalri mashinadagindan farq ilishi mumkin. Shunga o‘xshash muammolarni oldini olish uchun dasturchi uchun qoidalar to‘plamini yarating.

10. Katta dasturiy ta‘minot tizimining relizini qurush jarayonida muhandislar inobga olishi kerak bo‘lgan 5 ta omilni tasniflang.

5. JARAYONNI MUKAMMALLASHTIRISH

Hozirgi vaqtda sanoatdan doimiy talab shunaqaki arzonroq, yaxshiroq dasturiy ta‘minot tez muddatda yetkazib berilishi lozim. Buning natijaasida ko‘plab dasturiy taminot tashkilotlari dasturiy ta‘minot jarayonlarini mukammallashtirishni sarfni kamaytirib yoki yaratish jarayonlarini tezlashtirib dasturiy ta‘minot sifatini ko‘paytirishni maqsad qilib qo‘ydilar. Jarayonni mukammallashtirish deganda mavjud jarayonlarni tushunish va maxsulot sifatini ko‘tarish hamda sarflar va yaratish vaqtini kmaytirish maqsadida ularni o‘zgartirish degani.

Jarayonni mukammallashtirish va o‘zgartishda ikkita juda katta farq qiluvchi yondashuvlar ishlatiladi:

1. Jarayon yetukligi yondashuvi, bunda tashkilotdagi loyihani boshqarish va dasturlashni yaxshi amalyotida foydalanish jarayonini yaxshilashga etibor qaratiladi. Jarayon yetuklik darajasi dasturiy ta‘minot ishlab chiqishning tashkiliy jarayonlarida qabul qilingan teznik va boshqaruv amalyotining saviyasini aks ettiradi. Bu yondashuvning asosiy maqsadlari maxsulot sifatini yaxshilash va jarayon ketishini oldindan bilish.

2. Tezkor yondashuv, bunda takroriy yaratishga va dasturiy ta‘minot jarayonlaridagi sarf harajatlarni kamaytirishga etibor qaratiladi. Tezkor usullarning asosiy hususiyatlari funksionallikka tez erishish va mijozning talablari o‘zgarishiga tez javob berishi.

Har bir yondashuv tarafdorlari ikkinchisini foydasiga nisbatan salbiy fikirdadur. Jarayon yetukligi yondashuvi boshqaruv rejasini rivojlantirishga kiritiladi va odatda “tepada” o‘zgartirishni talab qiladi, chinki kiritilgan harakatlar bevosita dasturlashga aloqasi yo‘q. Tezkor yondashuvlar kodni revojlantirishga etibor qaratadi va rasmiyatchilik hamda hujjatlarni kamaytirishga harakat qiladi.

Tezkor usullarni men 3 bobda va boshqa joyda muhokama qilganman shuning uchun bu bobda me jarayonlarni boshqarish va yetuklikka asoslangan jarayon mukammallashtirishga etibor qaratataman. Men bu yondashuvni tezkor

usullardan afzal deb xisoblamayman. Mening ishonchim komilki o‘rtacha o‘lchamdagi loyihalar uchun tezkor usullar jarayonni mukammallashtirish eng yaxshi strategiyasidur. Lekin katta tizim, kritik tizim hamda har-xil kompaniyalardan kelgan xodimlar tomonidan yaratilayotgan tizimlar uchun boshqaruv muammolari ko‘pincha qiyinchilik tug‘diradi. Katta murakkab tizimlarni loyihalash bilan shug‘ullanuvchi korxonalar uchun yetuklikka asoslangan yondashuv ko‘proq mos keladi.

3-bobda aytganimdek dasturiy ta‘minot tizimini yaratishda ishlatilgan rivijlantirish jarayoni bu tizimni sifatiga ta‘sir qiladi. Shuninguchun ko‘plab odamlar dasturiy ta‘minot ishlab chiqarish jarayonining yaxshilanishi dasturiy ta‘minotning yuqari sifatiga olib keladi deb xisoblaydi. Jarayonni mukammallashtirishning bu tushunchasi sifatni yaxshilashga yordam berishi uchun ikkinchi jaxon urushidan keyin yapon sanoati bilan ishlagan amerikalik muhandis B.E.Deming maxsuludir. Yapon sanoati mukammallshtirishning uzluksiz jarayoniga ko‘p yillar sarf qildi, buning natijasida yapon sanoatining maxsulotlarining yuqori sifati tan olindi.

Deming va boshqalar sifatni statistik nazorat qilish g‘oyasini kiritgan. Bunda maxsulot defektlari miqdori o‘lchanadi va bu difektlar jarayon bulan aloqasi aniqlanadi. Bunda maqsad shundaki o‘zgartirish jarayonlari taxlil qilinib maxsulot defectlari miqdori kamaytiriladi. Defektlarning eng past miqdoriga yetganda jarayon standartlashtiriladi va mukammallashtirishning keying sikli boshlanadi.

5.1- rasm. Dasturiy ta‘minot maxsulotiga ta‘sir etuvchi omillar

Xamfri(1988) jarayonlarni boshqarish haqidagi kitobida shu usullar dasturlashda ham ishlatilishi mumkinligini keltirib o‘tgan. U aytyapti:

“B.E.Deming ikkinchi jaxon urushidan so‘ng yapon sanoatida ishlaganda jarayonni statistik boshqarish tushinchasini sanoatga nisbatan ishlatdi. Muhum farq bo‘lsa ham, bu tushunchalar avtomabil, kamera, qo‘lsoati va po‘lat singari dasturiy ta‘minotga ham qo‘llanilishi mumkin”

Uuimiy hossalar bo‘lganiga qaramy men Xamfri ni gapini maqullamayman chunki mashinasozlik natijalari dasturiy injiniringa asonlikcha qo‘llash mumkin emas. Ishlab chiqarishda jarayon va maxsulot munosabatlari yaqqol ko‘rinib turadi. Ishlab chiqishda odatda avtomatlashtirilgan vositalar o‘rnatiladi va jarayonning maxsulotga ta‘siri tekshiriladi. Agar birortasi mashinani kalibrovkasida hatoga yo‘l qo‘ysa bu mashina chiqargan hamma maxsulotda bilinadi. Mashinalarni o‘tnatishda hatoliklarni oldini olish va tekshirishning samarali usullarini ishlatilishi maxsulot sifatini yaxshilaydi.

Bunday sifat-maxsulot munosabatlari maxsulot mavxum ekanligida ko‘rinishi qiyin va avtomatlashtirilishi mumkin bo‘lmagan intelektual jarayonlarga bog‘liq. Dasturiy ta‘minot sifatiga uning ishlab chiqarish jarayoni emas balki uni

loyihalsh jarayoni ta‘sir etadi. Bazi hollarda ishlatilayotgan jarayon maxsulot sifatining eng muhum diterminanti bo‘lishi mumkin. Lekin hususan innovatsion ilovalar uchun jarayonda qatnashayotgan odamlar ishlatilayotgan jarayondan ko‘ra sifatga ko‘proq ta‘sir qilishi mumkin.

Dasturiy maxsulotlar yoki kino hamda film kabi boshqa intelektual maxsulotlarda maxsulot sifati uning dezayniga bog‘liq bo‘lgani uchun maxsulot sifatiga ta‘sir qiluvchi 4 ta muhim omillar maxjud. Ular 5.1- rasmda ko‘rsatilgan.

Bu omillarning har-birining ta‘siri loyiha o‘lchami va turiga bog‘liq. Alohida ost tizimlardan iborat bo‘lgan juda katta tizimlarni yaratuvchi jamoalar har-xil joyda ishlaydi, bunda maxsulot sifatiga ta‘sir etuvchi asosiy omil – dasturiy ta‘minot jarayonidir. Katta loyihalardagi asosiy muammolar integratsiyalash, loyihani boshqarish va aloqalar. Odatda jamoa azolarini qobilyat va tajribasi har-xil bo‘ladi. Yaratish jarayoni bir necha yil davom etishi mumkin. Shuning uchun jamoa tarkibi o‘zgarib turadi. Loyiha hayot sikli davomida u to‘liq o‘zgarishi mumkin.

Kichik loyihalarda jamoa a‘zosi bir nechta bo‘ladi shining uchun jamoa sifati ishlatiladigan jarayondan muhumroq bo‘ladi. Demak tezkor manifest jarayonni emas odamlarning muhumligini e’lon qoladi. Agar jamoaning qobilyati va tajriba darajasi yuqori bo‘lsa maxsulot sifati ishlatiladigan jarayonga bog‘liq bo‘lmagan holda yuqori bo‘ladi. Agar jamoa tajriba va malakasi past bo‘lsa jarayon sifati hatoliklarni kamaytirishi mumkin lekin dasturiy ta‘minotning yuqori sifatini ta‘minlamaydi.

Jamoa tarkibi kichik bo‘lganda rivojlanishning yaxshi texnologiyasi juda muhim bo‘ladi. Kichik jamoa ma‘muriy pretseduralarga ko‘p vaqt ajrata olmaydi. Jamoa azolari ko‘p vaqtini tizimni loyihalash va dasturlashga sarflaydi, shuning uchun yaxshi vositalar unumdorlikka ta‘sir qiladi. Katta loyihalar uchun bazaviy darajadagi rivojlanish texnologiyasi axborot boshqaruvi muhum. Qizig‘i shundaki katta loyihalarda murakkab dastiruy vositalar kamroq axamiyatga ega. Jamoa qatnashchilari o‘z vaqtining ozroq qismini tajraba ishlariga sarflaydi va ko‘proq vaqt muloqatta bo‘lib tizimning boshqa qismlarini tushunishga harakat qiladi.

Instrumentlarni rivojlanishi bunda xech qanday axamiyatga ega bo‘lmaydi. Lekin Wikis va blog kabi aloqalarni ta‘minlovchi Web 2.0 instrumentlari tarqalgan jamoalar a‘zolari o‘rtasida olaqani anchi yaxshilashi mumkin.

Agar loyihaning byudjeti noto‘g‘ri xisoblangan bo‘lsa yoki yetkazib berish grafigini noto‘g‘ri tuzilgan bo‘lsa maxsulot sifati odamlar, jarayon yoki instrument omillariga bog‘liq bo‘lmagan holda o‘zgaradi. Yaxshi jarayon samarali bajarilishi uchun yetarlicha manbalarni talab qiladi. Bu manbalar yetarli bo‘lmasa jarayon samarali bajarilmaydi. Agar manbalar mos kelmasa faqat zo‘r odamlar loyihani qutqarib qolishi mumkin. Yetishmovchilik juda katta bo‘lsa maxsulor sifati yomonlashadi. Yaratish uchun vat yetarli bo‘lmasa yaratilgan dasturiy ta‘minotning funksionalligi past bo‘lishi mumkin, yoki ishinchlilik va ishchanlik darajasi past bo‘ladi.

Ko‘p hallarda dasturiy ta‘minot sifatidagi muammolarning haqiqiy sababi yomon boshqaruv, mos kelmagan jarayon yoki past sifatli o‘qotishda emas, ko‘pincha bu tashkilot yashab qolishi uchun raqobatning natijasi. Shartnomani olish uchun tashkilot talab qilingan harakatni to‘g‘ri baxolamaydi yoki tizimni tezroq yetkazib berishni vada qiladi. Bu majburiyatlarni bajarish maqsadida bajarilishi qiyin bo‘lgan jadval tuziladi. Demak dasturiy ta‘minot sifatiga salbiy ta‘sir yetkaziladi.

5.1 Jarayonni takomillashtirish

2-bobda men faoliyatlarning izchilligi singari dastur jarayonining umumiy g‘oyalari haqida tanishtirib o‘tganman. Men umumiy jarayonni tasvirlab, ular: sharshara modeli va qayta foydalanishga asoslangan yaratish va men eng muhim jarayon faoliyatlari haqida muhokama qilib o‘tganman. Bu umumiy jarayon bir tashkilot ichida o‘ziga xos dastur va jarayonni yaratish uchun bajariladi.

Dastur jarayonlari yakka shaxsli kompaniyalardan ko‘p millatli kompaniyalargacha barcha tashkilotlarda o‘rganiladi. Bu jarayonlar yarattilgan mahsulot turiga, tashkilot o‘lchamiga va jarayon rasmiyatchiligining darajasiga bog‘liq holda turli tiplarda bo‘ladi.

Texnologik xususiyatlari	Kalit masalalar
Tozalik	Qay darajada jarayon aniq belgilangan va qanday qiliob bu jarayonning ta'rifini oson qilib tushunish bo‘ladi?
Standartlashtirish	Bu jarayon standarti umumiy jarayonda qaysi darajada asoslangan? Bu belgilangan jarayon standartlari ba'zi mijozlar uchun muhim bo‘lishi mumkin. Bu bir xil jarayon kompaniyaning barcha joylarida qay darajada ishlatiladi?
Ko‘rinish	Jarayonining rivojlanishi tashqi paydo bo‘ladi, jarayon faoliyati, ochiq-oydin ravishda natijalari bilan tugaydimi?
Nisbatan chiqishlar	Jarayon ma'lumotlar yig'ish yoki jarayon yoki mahsulot xususiyatlari o‘lchanadi imkonini beradi, boshqa faoliyatni o‘z ichiga oladi?
O‘lchanadigan	Ma'lumotlarni to‘plash yoki mahsulot xususiyatlarini boshqa faoliyatdagi maxsulot hususiyatlari orqali o‘lchaydigan jarayon bo‘lishi kerak, shundaymi?
Qo‘llab-quvvatlash imkoniyatlari	Faoliyatini qo‘llab-quvvatlash uchun dasturiy ta'minotdan qay darajada foydalanishingiz mumkin?
Maqbulligi	Dasturiy ta'minot ishlab chiqarish uchun mos jarayon, mas'ul va kerakli muhandislarni ishlatilishi emasmi?
Ishonchliligi	Jarayonda yuz berishi mumkin bo‘lgan hatoliklar yakunda mahsulotdagi hatoliklarga olib kelmasligi uchun jarayon to‘g‘ri tashkil qilinganmi?
Chidamliligi	Kutilayotgan muammolar yuz bergan taqdirda ham jarayon davom eta oladimi?
Ta‘mirlash mumkinliligi	Tashkiliy ta‘lablarning o‘zgarish yoki mukammallashtirishni hisobga olib rivojlantirish

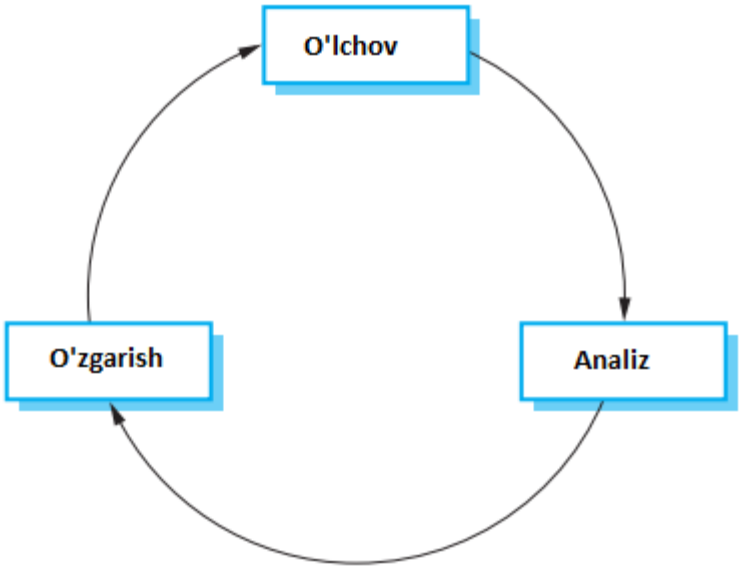
	jarayonini o‘zgartirish mumkinmi?
Tezlilik	Bu spesifikasiyadagi tizimni yetkazib berish jarayoni qancha tez yakunlanishi mumkin?

5.2- rasm. Jarayon attributlari

Bu yerda “ideal” yoki “standart” singari o‘ziga xos tipli dastur mahsulotlarining barchasi uchun barcha tashkilotlarda ishlatsa bo‘ladigan dasturlash jarayoni yo‘q. Har bir kompaniya o‘zining o‘lchamiga, xodimlarining qobiliyatlariga, yaratilayotgan dasturning turiga, xaridor va bozor talabiga va kompaniya madaniyatiga bog‘liq holda o‘zining jarayonini yaratishi kerak.

Jarayonni yaxshilash, shuningdek, o‘ziga xos metodlar yoki uskunalarni qabul qilish yoki nashr etishdan foydalanish kabi ma‘nolarni anglatmaydi. Shunga qaramay tashkilotlar dasturning bir xil tipini yaratishda oddiylikka egalar, u yerda mahalliy tashkiliy faktorlar, jarayonga ta‘sir etuvhci standartlar bo‘ladi. Siz jarayonni mukammallashtirishda kamdan-kam muvafaqqiyatga erishasiz, agar barcha yerda qo‘llaniladigan jarayonni o‘zgartirishga urinsangiz. Siz har doim mahaliiy muhit va madaniyatni va bu jarayhonni o‘zgartirish takliflariga qanday ta‘sir ko‘rsatishini hisobga olishingiz shart.

Siz yana shuningdek mukammallashtirishni xohlayotgan jarayonni ba‘zi bor tushunchalarini ham e’tiborga olishingiz kerak. Sizning maqsadingiz dastur sifatini mukammallashtirish bo‘lishi mumkin va shuning uchun siz dastur yaratilgan va tekshirilgan yo‘lni o‘zgartiradigan yangi jarayon faoliyatlarini kiritishingiz mumkin. Siz jarayonning ba‘zi bir attributlariga qiziqishingiz mumkin va siz kompaniyangiz uchun qanday jarayon attributlar muhim ekanligiga qaror qilishingiz kerak. Jarayon attributlariga misollar mukammallashtirish nishonlari bo‘lib, 5.2-rasmda ko‘rsatilgan.



5.3- rasm. Jarayonni takomillashtirish halqasi

Bu attributlar aniq bog‘lanadi,ba‘zan ijobiy ba‘zan salbiy. Jarayon, ya‘ni yuqori tezlikli attribut tushunarli bo‘ladi. Jarayonni kuzatuvchi ishlab chiqarilgan mahsulotdan faoliyatning mavjudligiga xulosa qiladi.Boshqa tomondan ko‘rinuvchi jarayon tezlikka bog‘langan bo‘ladi. Jarayonni tayyorlash jarayon haqidagi axborotlarni ishlab chiqara oladigan odamlarni talab qiladi. Bu hujjatlarni tayyorlashni oladigan vaqt sababli dastur mahsulotini pastga tushirib yuborishi mumkin.

Bir vaqtda sodir bo‘ladigan barcha jarayon attributlarini optimizatsiyalaydigan jarayonni mukammallashtirishni bajarishni imkoni yo‘q. Misol uchun siz jarayonni tez yaratishni maqsad qilsangiz, keyin jarayon ko‘rinishini kamaytirishga majbur bo‘lasiz.Agar siz jarayonni tayyorlashni istasangiz, keyin kompaniyaning turli qismlarida qo‘lllaniladigan va tashkiliy amaliyotni aks ettiruvchi protsedura va uskunalarni qabul qilishingiz kerak. Bu jarayonning mahalliy to‘g‘ri kelishini kamaytirishi mumkin. Injenerlar ish usullarini ta‘minlash uchun standartsiz uskunalar va mahalliy protseduralarni kiritishadi.

Jarayonni mukammallashtirish jarayoni siklik jarayon 5.3-rasmda ko‘rsatilgan. U 3 ta jarayonni o‘z ichiga oladi:

1. *Jarayon o‘lchovi*. Joriy loyihaning attributlari(xususiyatlari) o‘lchanadi. Bundan maqsad jarayonni mukammallashtirishda tashkilot maqsadiga ko‘ra, o‘lchashni yaxshilash. Bu uslublar agar mukammallashtirish samarali bo‘lsa qaror qilishingizda yordam beradi.

2. *Jarayon analizi*. Joriy jarayon tahlil qilinadi va zaifligi aniqlanadi. Jarayon modellari (ba‘zan jarayon xaritalari deb ataladi) nu bosqich davomida yaratilishi mumkin bo‘lgan jarayonni tasvirlaydi. Analiz jarayon harakteristikalari shuningdek tezlik va mustahkamlik ni hisobga olish orqali markazda turishi mumkin.

3. *Jarayon o‘zgarishi*. Jarayon o‘zgarishi jarayon aniqlangan jarayon zaifligini adreslash uchun taklif qilinadi. Bu yerga o‘zgarishlarning samaraliligi haqidagi ma‘lumotlarni to‘plash uchun siklning qisqacha xulosalari kiritiladi.

Jarayondagi aniq ma‘lumotlarsiz yoki dasturni yaratishda jarayondan foydalanishsiz jarayonni mukammallashtirishning qiymatini tahlil qilishni imkoni yo‘q. O‘zgarishlar siklining birinchi qismida siz dastur mahsuloti harakteristikalari o‘lchovi va dastur jarayoni haqidagi ma‘lumotlarni to‘plash uchun jarayon faoliyatlarini kiritishingiz kerak.

Jarayonni mukammallashtirish uzoq muddatli faoliyat, har bir bosqich bir necha oylab davom etishi mumkin. Bu shuningdek davomli faoliyat bo‘lib, yangi jarayonlar kiritiladi,biznes muhiti o‘zgartiriladi.

5.2. Jarayon o‘lchovi

Jarayon o‘lchovi dastur jarayoni haqidagi miqdoriy ma‘lumotlar bo‘lib, jarayon faoliyatlarini bajarilishi uchun ketadigan vaqt hisoblanadi. Misol uchunsiz dastruni yaratish uchun talab qilingan vaqtni o‘lchashingiz mumkin. Humphrey(1989) u o‘z kitobida jarayonni mukammallashtirishda isbotlar ya‘ni jarayon o‘lchovi va mahsulot attributlari jarayonni mukammallashtirishuchun ahamiyatli hisoblanadi.U shuningdek kichik-masshtabli jarayonni mukammallashtirishda muhim rolga ega o‘lchovni taklif etadi.

Jarayon o‘lchovlari jarayon mukammallashtirilishining ta‘siri bo‘ladimi yo yo‘qmi shuni tahlil etish uchun qo‘llaniladi.Misol uchun natija va vaqt tekshirish uchun ajratiladi. Jarayonni tekshirishning samarali yaxshilanishlari natija yoki tekshirish vaqtini qisqartirishi kerak.Shunday qilib jarayon o‘lchovlari ularning o‘zida niqlanmaydi, agar mahsulot sifati yaxshilansa. Mahsulot sifati ma‘lumotlari jarayon faoliyatlariga bog‘langan va to‘plangan bo‘ladi.

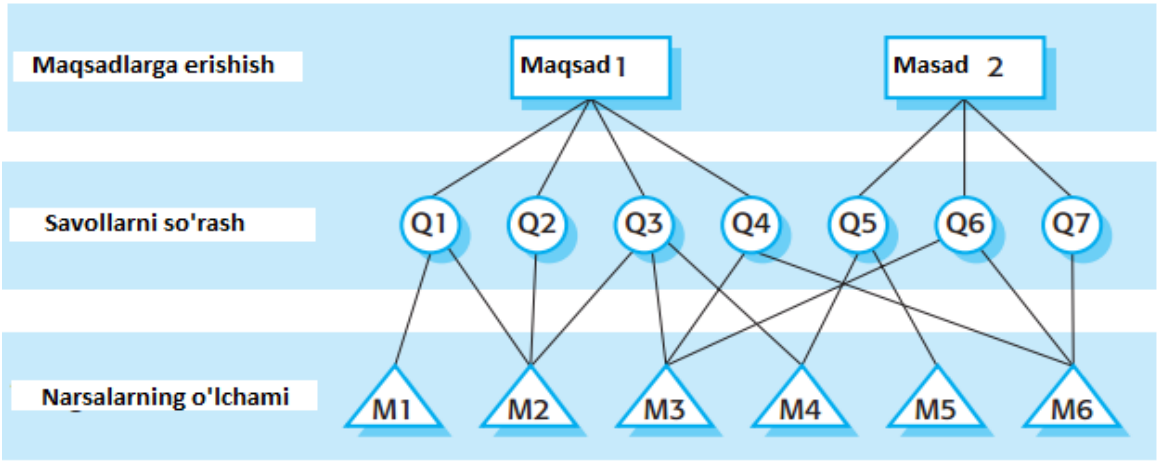
Jarayonning metr o‘lchovidagi 3 xil turi bor:

1. *Tugatiladigan o‘ziga xos jarayon uchun olinadigan vaqt.* Bu jarayon uchun ajratilgan umumiy vaqt., calendar vaqti, maxsus injenerlar tomonidan jarayonda sarflangan vaqt.

2. *O‘ziga xos jarayon uchun talab qilingan resurslar*. Resurslar person-days(shaxs-kuni), sayohat narxlari,kompyuter resurslaridagi umumiy natijalarni o‘z ichiga oladi.

3. *O‘ziga xos hodisalarning bir qancha soni*. Hodisalarga misollar kodni nazoratdan o‘tkizish davomida vujudga keladigan nuqsonlarni, so‘ralgan talablar o‘zgarish sonini va o‘rtacha o‘zgartirilgan kod qatorlarini o‘z ichiga olish uchun tekshirilishi mumkin.

O‘lchovning birinchi ikkita turi agar jarayon o‘zgarishlari samaradorligi yaxshilansa yaratish uchun qo‘llaniladi. Bu yerda dasturni yaratish jarayonida o‘rnatilgan tushunchalar bor, shuningdek talablar maqbulligi, arxitekturaviy loyihaning yakunlanishi kabilar.



5.4- rasm. GQM masalasi

Siz vaqt va talab qilingan natijani bir o‘rnatilgan tushunchadan boshqasiga ko‘chirish uchun o‘lchay olasiz. O‘zgarishlar kiritilgandan so‘ng, agar jarayon o‘zgarishlari vaqtning qisqarishida muvafaqqiyatli bajarilsa, tizim attributlarining o‘lchovlari ko‘rsatiladi.

Hodisalarning o‘lchovlari dastur sifatida to‘g‘ri olib boriladi. Misol uchun dasturni nazoratdan o‘tkazishda vujudga keladigan bir qancha nuqsonlarning oshishi yaxshilangan mahsulot sifatida ehtimol aks etadi.Shunday qilib, bu keyingi mahsulot o‘lchovida tasdiqlanadi.

Jarayon o‘lchovidagi fundamental qiyinchilik-jarayonni mukammallashtirishni ta‘minlash uchun to‘planishi kerak bo‘lgan jarayon haqidagi axborotlarni bilish demakdir. Basili va Rombach (1988) ular GQM(Goal-question-metric) deb ataydigan dastur va jarayon o‘lchovida keng qo‘llaniladigan namuna(paradigma) ni taklif etishgan. Basili va Green (1993) AQSH, mahsus agentlik NASA da bu takrorlanish uzoq muddatda qanday qo‘llanilganligi, o‘lchovga asoslangan jarayon dasturini tasvirlab berishgan.

GQM paradigmasi jarayonni mukammallashtirishda 3 ta kritik savolga javob berishda yordam berishda qo‘llaniladi(5.4- rasm). Ular:

1. Jarayonni mukammallashtirishni nega kirityapmiz?

2. Mukammallashtirishni aniqlash va tahlil qilishda qanday axborotlar bizga yordam beradi?

3. Qanday jarayon va mahsulot o‘lchovlari bu axborotni ta‘minlashni talab qiladi?

Bu savollar GQM dagi abstraktsiyaga bog‘langan(maqsad-vazifalar,savollar,o‘lchovlar):

1. *Maqsad vazifalar*. Maqsad – bu shunday narsaki, tashkilot muvafaqqiyatga erishish uchun harakat qiladi. U jarayon attributlariga aloqador bo‘lmasligi kerak. Vazifalarga misollar oxiriga yetgan jarayon bosqichida, qisqa mahsulot yaratish vaqtida yoki mahsulotning ishonchliligi oshganda mukammallashtiriladi(5.5-qismda ko‘ring).

2. *Savollar*. Aniqlanadigan vazifalarga bog‘langan maxsus sohalardagi vazifalarning yaxshi tarafga o‘zgarishi.Bir vazifa javob berilishi kerak bo‘lgan bir qancha savollarga ega bo‘ladi. Savollarga misollar “bizning joriy jarayonimizdagi o‘tish joyi qayerda?”,”Mijozlar kamaytirilgan va mahsulot talabini tugatish uchun qancha vaqt talab qilinadi?” yoki “Qancha tekshiruvda vujudga kelayotgan mahsulot nuqsonlarida samarador bo‘ladi”singari bo‘lishi mumkin bo‘lgan mahsulot yaratish vaqtining vazifalariga bog‘langan bo‘ladi.

3. *O‘lchovlar*. Bular jarayonni mukammallashtirish muvafaqqiyatga erishadimi yo‘qmi tasdiqlash uchun o‘lchovlar Yuqoridagi savollarga javob berishga yordam berish uchun, siz har bir jarayon faoliyatini tugatish uchun ketadigan vaqtda, klient va mijoz o‘rtasidagi formal munosabatlarda yoki har tekshiruv ishida vujudga kelgan nuqsonlar sonida ma‘lumotlarni to‘plashingiz mumkin.

Jarayonni mukammallashtirishda GQM takrorlanishidan foydalanishning afzalligi –u mahsus jarayon aloqalaridan(savollar) tashkiliy aloqalarni (vaifa-maqsad) ajratadi. U qanday ma‘lumotlar to‘planganligini hal qilish uchun asos bilan ta‘milaydi va turli usullarda analiz qilinishi kerak bo‘lgan to‘plangan ma‘lumotlarni taklif qiladi.

GQM takrorlanishi AMI(Analyze, Measure, Improve)(Analiz, o‘lchov,yaxshilash)usulida SEI ning tugatilgan modeli bilan yaratilgan (Paulk et al., 1995). AMI metodini yaratuvchilar jarayonni mukammalllashtirish uchun pog‘onalashtirilgan takrorlanishni taklif etadilar. AMI qo‘llanmasi (Pulford et al., 1996) o‘lchovga asoslangan jarayonni mukammallashtirishda amaliy maslahat va ko‘rsatmalar bilan ta‘minlaydi.

3-bobda muhokama qilganimdek, o‘lchovlar ba‘zan shubhali bo‘ladi. Masaln, dasturdagi yozilgan nuqsonlarni tuzatish uchun ketadigan o‘rtacha vaqtni o‘lchaysiz. Bu “tozalangan” tarzda belgilangan xabar vaqti va xabar vaqtini qabul qiluvchi jamoa o‘rtasidagi vaqtdir. Siz xatoni yozib olish uchun yangi web ga asoslangan uskunalarni kiritasiz, bu uskuna biror vaqt uchun foydalanilgandan keyin kamaygan nuqsonlarni tuzatish uchun ketgan vaqtni tahlil qilasiz.

Siz nuqsonni tuzatishda qisqargan vaqtga ega xatoni yozib olish uskunalarini kiritishni tasdiqlashingiz mumkin. O‘lchovlar uchun o‘zgarishini kuzatganingizda, bu sizga tanishtirilgan jarayon o‘zgarishlariga bu o‘zgarishlarni attributlar.Shunday qilib, u mukammallashtirish haqidagi soddalashgan taxminlarni bajarish uchun xavfli.

O‘lchovdagi o‘zgarishlar biror narsani: loyiha jamoasidagi odamlar o‘zgarishiga, jarayon jadvalining o‘zgarishiga yoki menjment o‘zgarishiga sabab bo‘ladi.O‘zgarishlar quyidagi kuzatuvlarni o‘z ichiga oladi:

1. Tizim boshidan qisqarishi mumkin,shuning uchun ko‘proq vaqt nuqsonlarni bartaraf etish uchun foydali bo‘lib qoladi. Bu o‘rtacha “nuqsonni tuzatish” vaqtidagi qisqarishni boshqaradi.Jarayonni mukammallashtirish asl farqni keltirib chiqarishi mumkin.

2. Yangi tizim nuqsonlarni tuzatish uchun ketadigan aniq vaqt uchun farqsiz tayyorlanishi mumkin. Lekin u axborotni yozib olishga osonroq tayyorlanadi. Shuning uchun nuqsonlarni tuzatish vaqtlari ko‘proq yangi tizim bilan aniq o‘lchanadi. Nuqsonlarni o‘rnatish o‘rtacha vaqtida aniq o‘zgarishlar bo‘lmaydi.

3. Yangi tizim kiritilishidan oldin o‘lchovlar tizimni nazoratdan o‘tkazish orqali bajariladi. Tuzatish eng oson va eng tez nuqsonlar allaqachon tuzatilgan, faqat “og‘ir nuqsonlar” tiklash uchun ko‘p vaqtni oladi. Shunday qilib, nuqsonlarni yozib olish tizimi kiritilgandan so‘ng, o‘lchovlar yangi tizimni nazoratdan o‘tkazishni boshlanishida tayyorlanadi va nuqsonlar “oson nuqsonlar ” sifatida tezda tuzatiladi.

4. Nazoratdan o‘tkazish jamoasining yangi menejeri nuqsonlardek foydalanuvchi interfeysiga xabar berish uchun jamoa a‘zolarni o‘qitadi. Bu shuni anglatadiki, ko‘p “oson nuqsonlar” tez o‘rnatilganligi haqida axborot beriladi.

O‘lchov jarayon va jarayon o‘zgarishlari haqidagi asoslarning genratsiyalsh usuli. Shunday qilib, asos jarayonni o‘zgartirish samarali bo‘lishiga ishonch hosil qilishingizdan oldin jarayon haqidagi boshqa axborot bilan birga tushuntirilishi kerak. Siz har doim o‘zgarishlarning sifatli tahlili bilan aloqada o‘lchovlardan foydalanishingiz kerak. Bu o‘zgarishlarning samaradorligida odamlarning

tassurotini olish va tanishtirilgan o‘zgarishlar haqidagi jarayonlarni odamlarg gapirib berish. Nafaqat bu jarayonga ta‘sir etishi mumkin bo‘lgan boshqa faktorlarni ko‘rsatadi. U taklif qilingan o‘zgarishlardan foydalangan jamoaga o‘lchovni va bular aniq yaratish amaliyotiga qanday ta‘sir ko‘rsatishini anglatadi.

5.3. Jarayon analizlari

Jarayon analizlari kalit harakteristikalarni tushunishga yordam berishga jarayonlarning o‘rgatilishi va qanday jarayonlar odamlar tomonidan amaliyotda bajarilishi tushuniladi. Men 5.3-rasmda jarayon analiziga olib keluvchi jarayon o‘lchovini taklif qilganman. Bu bir soddalashtirish, chunki, haqiqatda, bu faoliyatlar bir-biriga o‘ralgan. Siz o‘lchovni bilish uchun ba‘zi analizlarni yakunlashingiz kerak. Siz o‘lchangan jarayonni yanada chuqurroq tushunishni o‘rganasiz.

Jarayon analizi bir-biriga bog‘langan xususiyatlardan iborat:

1. Jarayonda o‘z ichiga olingan faoliyatlar va ular orasidagi bog‘liqliklarni tushunish.

2. Jarayon faoliyatlari va o‘lchovi o‘rtasidagi bog‘liqlikni tushunish.

3. Tashkilotdagi solishtirib bo‘ladigan jarayonlarga analiz qilayotgan jarayonlarni bog‘lash yoki bir xil tipdagi jarayonlarni benuqson qilib ko‘rsatish.

Jarayonni analizlash davomida jarayon qanday davom etayotganini tushunishga harakat qilasiz. Jarayon samarasizliklarini va muammolari haqidagi axborotlarni qidirasiz. Siz jarayon o‘lchovi haqida, jarayonni ta‘minlash uchun foydalaniladigan dastur uskunalari, jarayonga qanday tashkiliy omillar ta‘sir etishiga ham qiziqishingiz kerak. 5.5-rasmda jarayonni analizlash davomida tekshirishingiz mumkin bo‘lgan jarayon tushunchalari ko‘rsatilgan.

Eng oson qo‘llaniladigan jarayon texnikalari:

1. *Intervyu oluvchilar va intervyular*: Loyiha ustida ishlayotgan menejerlar va injenerlardan davom etish haqida so‘raladi. Javoblar shaxsiy intervyu davomida

yaxshilanadi. Quyida muhokama qilganimdek, muhokama dastur jarayon modellari atrofida strukturalanadi.

2. *Etnografik qo‘llanmalar*. Etnografik qo‘llanmalar(4-bobga qarang) inson faoliyatida dastur yaratish tabiatini tushunishda foydalanilishi mumkin.Shu kabi analizlar intervyu oluvchilar va intervyular tomonidan ko‘rsatilishi mumkin bo‘lmagan murakkablik va nozikliklarni ko‘rsatadi.

Bu takrorlanishlarning har biri o‘z yaxshi va yomon tarafiga ega. Savol beruvchiga asoslangan analizlar to‘g‘ri savol aniqlanganida tezda yakunlanadi. Agar savol nomunosib bo‘lsa,siz jarayonni tushunishni to‘liqsiz yakunlashingiz mumkin.Bundan tashqari bu analizlar tahlil shaklidek paydo bo‘ladi. Injenerlar siz eshitishni istaganjarayon haqidagi haqiqatga javob berishadi.

Jarayonlar yo‘nalishi	**Savollar**
Qabul qilish va standartlashtirish	Jarayon tashkilot orqali ro‘yhatga olingan va standartlashtirilganmi? Agar bunday bo‘lmasa har qaysi o‘zgartirish jarayonning yagona holati uchunmi? Agar jarayonlar standartlashtirilmagan bo‘lsa bitta jarayonni o‘zgartirilishi korxinaning bishqa joydagi shunga o‘xshash jarayonlarga qo‘llanilmaydimi?
Dasturiy injiniring amalyoti	Jarayonga kiritilmagan yaxshi dasturlash usullaridan habardormi? Nima uchun ular kiritillmagan? Ular yo‘qligi maxsulotga ta‘sir etmaydimi, masalan dasturiy ta‘minot tizimidagi nosozliklar miqdoriga?
Tashkiliy cheklovlar	Jarayon dizayini va u bajarilgan usullariga qanday tashkiliy cheklovlar qo‘yilgan? Masalan agar jarayon maxfiy ma‘lumot bilan aloqada bo‘lsa tekshirib ko‘ring maxfiy ma‘lumotlar tashqi korxonalarga ketadigan malumotlar ichiga qo‘shilib qolmaydimi? Tashkiliy cheklovlar natijasida jarayon o‘zgartirishlari amalga oshmasligi mumkin.

Aloqalar	Jarayondagi aloqalar qanday boshqariladi? Bajarilgan o‘zgartirishlar jarayonda qanday aloqa muammolariga olib keladi? Aloqa muammolari ko‘plab jarayondagi asosiy muammo va aloqaviy tor joylar loyihaning kechikishini sabalari bo‘ladi.
Ichki kuzatish	Jarayon refliksivmi? Bu degani (unda qatnashganlar jarayonni qanday yaxshilash mumkinligi haqida muhokama qilishga undaydi). Jarayon qatnashchilari jarayonni mukammallashtirishga oid takliflarini kiritish mumkin bo‘lgan mexanizimlari bormi?
O‘rganish	Jamoaga yangi qo‘shilgan odamlar dasturiy maxsulot jaraytonlari haqida qanday bilim olishadi? Korxonada jarayonno boshqarish yoriqnomasi yoki o‘qitish dasturi bormi?
Aloqalarni qo‘llab-quvvatlash	Jarayonning qaysi tomonlari dastiruy vositalar tomonidan quvvatlangan va quvvatlanmagan? Quvvatlanmagan sohalar uchun rentable vositalar bormi? Quvvatlangan sohalardagi vositalar samaralimi? Eng yaxshi vositalar haqiqatdan foydalanishga qulaymi?

5.5- rasm. Jarayon analizlarining tushunchalari

Siz savollarning tayyorlangan yozuvi bilan inte-rvyuni boshlashingiz mumkin, ammo odamlardan oladigan javobingizga ko‘ra ularni moslashtirasiz. Agar siz qatnashuvchilarga maqolangizni kengroq muhokama qilishlariga imkon bersangiz, jarayon qatnashuvchilari jarayon muammolari va jarayonni amaliyotda o‘zgartirilishi haqida gapirishadi.

Deyarli barcha jarayonlarda, odamlar mahalliy sharoitlarga jarayonlarni moslash uchun mahalliy o‘zgarishlar qilishadi. Etnografik analiz intervyuga qaraganda ko‘proq mos. Shunday qilib analizning bu turi faoliyatlarni so‘nggi bir

necha oyga uzaytiradi. U jarayonning tashqi kuzatuvlariga tayanadi. Analizni tugatish uchun,mahsulotni ta‘minlash va qo‘llab-quvvatlash uchun loyihaning boshlang‘ich bosqichlariga jalb etilishingiz kerak. Katta loyihalar uchun bu bir necha yillarni talab etadi, shuning uchun bu katta loyihalarda jarayonning etnografik analizini yakunlash uchun amaliy emas deb hisoblaniladi. Etnografik analiz talab qilingan jarayon qismlarinin tushunishda foydali.

Jarayon analiz qilinayotganda, jarayondagi faoliyatlarni aniqlaydigan jarayon modellar bilan boshlash va bu faoliyatlarni kiritish va chiqarishda foydali. Bu model jarayon aktyorlari - odamlar yoki bajarilayotgan harakatlar uchun javobgarlik rollari haqidagi axborotni o‘z ichiga oladi. Siz jarayon modellari yoki ko‘proq rasmiy jadval shaklidagi qaydlarni tasvirlash uchun norasmiy qayddan foydalanishingiz mumkin, UML harakat diagrammasi yoki biznes jarayon modellashini qayd etish, shuningdek BPMN Ushbu kitobda jarayon modellariga ko‘plab misollar keltirilgan.

Jarayon modellari jarayondagi harakatlarga e’tibor qaratishning va bu harakatlar o‘rtasida ma‘lumotlar uzatishning yaxshi usulidir. Bu jarayonlar rasmiy va yakunlangan bo‘lmasligi kerak, ularning maqsadi jarayon hujjatlariga qaraganda ozroq muhokamani yuzaga keltirish. Odamlar bilan muhokama va jarayonning kuzatuvlari savollar atrofiga quriladi.Bu savollarga misollar quyidagicha bo‘lishi mumkin:

1. Amaliyotda qanday faoliyatlar olib boriladi ammo modelda ko‘rsatilmaydimi? Modellar to‘liq bo‘lmaydi lekin agar odamlar turli o‘tkazib yuborilayotgan harakatlarni aniqlashsa. Bu sizga shuni aytadiki, jarayon tashkilotga to‘g‘ri keladigan tarzda bajarilmaydi.

2. Jarayon harakatlarimi?,modelda ko‘rsatiladimi? Jarayon aktyorlari uni samarasiz deb o‘ylaydimi? Qanday samarasiz harakatlar jarayon o‘lchoviga ta‘sir ko‘rsatadi?

3. Narsalarning xato ketgani qanday sodir bo‘ladi? Jamoa modeldagi jarayonni aniqlashda davom etadimi,yoki jarayon voz kechiladimi va favqulodda

harakat olinadimi? Agar jarayon voz kechilsa, bu dastur injenerlari jarayon yaxshi yetarli yoki istisnolarga moslashuvchanlikka ega emasligiga ishonishmaydi.

4. Jarayondagi turli bosqichlarda kim aktyor? Va ular qanday munosabatda bo‘lishadi? Axborot almashinuvida qanday o‘tish joylari sodir bo‘ladi?

5. Qanday uskunalar modelda ko‘rsatilgan faoliyatlar uchun foydalaniladi? Uskunalar qanday mukammallashtiriladi.

Dastur jarayonini analiz qilishni yakunlasangiz,jarayon va uni mukammallashtirishni chuqurroq anglab yetasiz.Siz shuningdek jarayonni mukammallashtirish omillarini bilishingiz va bular mukammallashtirish ko‘lamini qanday chegaralaydilar?

5.3.1. Jarayon istisnolari

Dasturlash jarayoni murakkab mazmunga ega. Biror tashkilotda jarayon modellari aniqlanishi mumkin, ammo bu faqat holatni taqdim etadi, ya‘ni yaratish jamoasi kutilmagan muammolarga duch kelishmaydi. Haqiqatda, kutilmagan muammolar loyiha boshqaruvchilari uchun har kungi hayotning asosi.”Ideal” jarayon modeli topilgan muammolarga yechimlarni dinamik o‘zgartirishi kerak. Istisno turlariga misollar, loyiha boshqaruvchisi quyidagilarni taqsimlashi kerak:

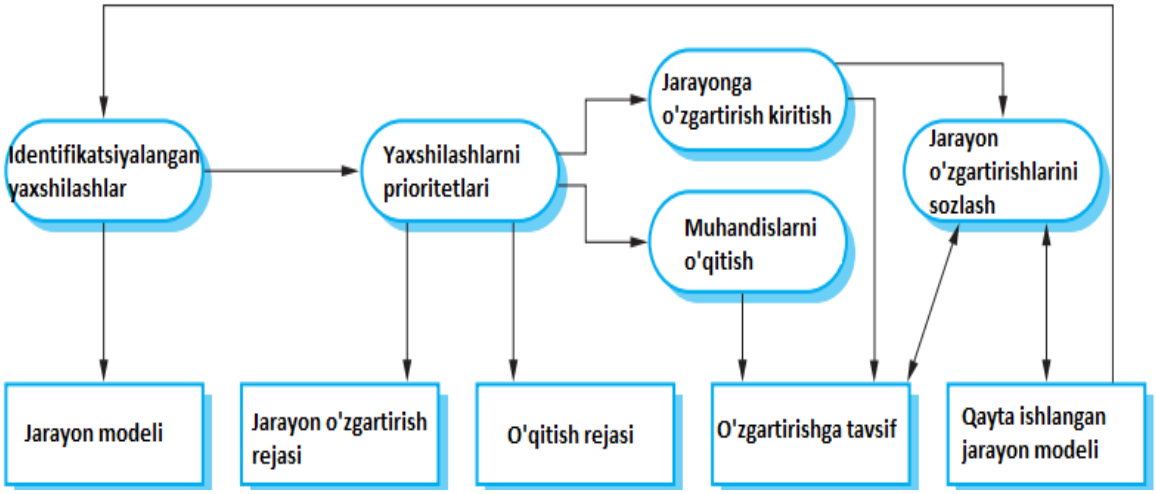
- Kritik loyihalar qayta ko‘rilishidan oldin bir xil vaqtda bir qancha kalit odamlar kasal bo‘lib qolishi.
- Kompyuter xavfsizligidagi jiddiy uzilish shuni anglatadiki barcha tashqi kommunikatsiyalr bir necha kun mobaynidan harakatdan to‘xtaydi.
- Kompaniyani qaytadan tuzish shuni anglatadiki, boshqaruvchilar loyihani boshqarishga qaraganda ko‘proq tashkiliy muammolarga ko‘p vaqt sarflashlari kerak.
- Yangi loyiha uchun takliflarni yozish uchun kutilmagan talablar shuni bildiradiki, natija joriy loyihadan taklifni yozishga o‘tkaziladi.

Muhim jihatdan, istisno ta‘sir ko‘rsatadi va odatda o‘zgartiradi ba‘zi usullarda: resurslar byudjetlar yoki loyiha jadvallari. Barcha istisnolarni aniqlash

va ularni formal jarayon modeliga kiritish qiyin. Siz istisnolarni hal qilishingiz va keyin kutilmagan vaziyatlarni yengish uchun “standart” jarayonlarni o‘zgartirishingiz kerak.

5.4. Jarayon o‘zgarishi

Jarayon o‘zgarishi mavjud jarayonga o‘zgartirish kiritish tushuniladi. Taklif qilganimdek siz buni yangi amaliyotlar, metodlar, uskunalarni kiritish,jarayon faoliyatlarini o‘zgartirish yoki jarayondan yetkazib beradiganlarni o‘chirish yoki u yerga kiritish,aloqalarni yaxshilash yoki yangi rol va majburiyatlarni kiritish orqali qilishingiz mumkin.



5.6- rasm. Jarayonni o‘zgartirish

Jarayon o‘zgarishlari mukammallashtirilgan vazifa–maqsadlar: “25% nazoratdan o‘tkazish davomida vujudga kelgan nuqsonlar sonini kamaytirish” tomonidan bajariladi.O‘zgarishlar bajarilgandan so‘ng, siz jarayon o‘lchovlaridan o‘zgarishlar samaradorligini tahlil qilishda foydalanasiz.

Jarayonni o‘zgartirish jarayonida (5.6- rasmda) 5 ta kalit bosqichlar keltirilgan:

1. *Mukammallashtirishni tasdiqlash*. Bu bosqich sifatli muammolarga, jarayonni analizlash davomida aniqlangan narx samarasizliklari uchun yo‘nalishni aniqlash uchun jarayon analizlarining natijasidan foydalanish bilan aloqador

bo‘ladi.Siz jarayon muammolari uchun yangi jarayonlar,jarayon strukturalari,metodlar, uskunalarni taklif qilishingiz mumkin. Misol uchun biror kompaniya shunga ishonadiki, ko‘plab dasturlash muammolari talablar muammosidan kelib chiqadi. Talablar injiniringidan foydalanishning eng yaxshi amaliyoti boshqaradi (Sommerville and Sawyer, 1997), turli talablar injiniringi amaliyoti kiritiladi yoki o‘zgartiriladi.

2. *Mukammallashtirish afzalligi*. Bu bosqich jarayonga mumkin bo‘lgan o‘zgarishlarni tahlil qilish aloqador. Ko‘plab mumkin bo‘lgan o‘zgarishlar aniqlanganda, dastlab ularni kiritishni imkoni bo‘lmaydi va siz bunda qaysi biri eng muhimligiga qaror qilishingiz kerak.Siz o‘zgarishlarni kiritsh narxi,tashkilotda o‘zgarishning ta‘siri,mahsus jarayon hududlarini yaxshilash kerakligi va boshqa faktorlarga asoslangan qarorlarni tayyorlashingiz mumkin. Masalan, kompaniya eng yuqori darajali jarayonni o‘z ichiga oluvchi talablarni boshqarishi uchun talablar menejmenti jarayonini tanishtirishni e’tiborga olishi mumkin.

3. *Jarayon o‘zgarishini tanishtirish.* Buning ma‘nosi yangi protseduralar,metodlarni, uskunalarni joyiga qo‘yish va ularni boshqa jarayon faoliyati bilan birlashtirish. Siz o‘zgarishlarni kiritish uchun yetarli vaqt berishingiz va bu o‘zgarishlar boshqa jarayon faoliyatlari,tshkiliy protsedura va standartlar bilan taqqoslasa bo‘lishini ta‘minlashingiz shart. Bu o‘z ichiga talabni boshqarish uchun uskunalarni egallash va bu uskunalardan foydalanish uchun jarayonlarni loyihalashni o‘z ichiga oladi.

4. Jarayon mashg‘uloti. Mashg‘ulotsiz jarayonni o‘zgartirishning to‘liq foydasiga erishishning umkoni yo‘q. Injenerlar taklif qilingan o‘zgarishlarni va yangi va o‘zgargan jarayonlarni qanday bajarishni tushunishi kerak. Jarayon o‘zgarishlari ko‘pincha yetarli mashg‘ulotlarsiz yuklanadi va bu o‘zgarishlarning ta‘siri mahsulot sifatini yaxshilashga qaraganda tezroq yomonlashadi. Talablar menejmenti bo‘lsa, mashg‘ulot talab menejmenti qiymati,jarayon faoliyati izohi va tanlangan uskunalarni kiritish qiymatini muhokamasini o‘z ichiga oladi.

5. O‘zgarishni sozlash.Taklif qilingan jarayon o‘zgarishlari hech qachon ular kiritilishi bilanoq to‘laliicha samarali bo‘lmaydi. Siz vujudga kelgan katta

muammolarda vaqtni sozlashingiz kerak va jarayon uchun o‘zgarishlar taklif qilinadi va kiritiladi. Bu sozlash davri yaratiuvchi injenerlar yangi jarayon bilan xursand bo‘lgunicha bir necha oy davomida o‘tkaziladi.

Bir vaqtda ko‘p o‘zgartirish kiritish yahshi emas. Tajribaviy qiyinchilikdan tashqari bunda hae bir o‘zgartirish jarayonga qanday ta‘sir qilishini aniqlash qiyin bo‘ladi. O‘zgartirish kiritilgandan kegin mukammallashtirish jarayonini tahlil qilish mumkin, texnologik muammolarni yechish uchun o‘zgartirishlar kiritish taklif qilinadi va hokazo.

O‘zgartirish jarayonlarida ikki turdagi qiyinchiliklar uchrashi mumkin:

1. *O‘zgartirish kiritishga qarshilik*. Guruh a‘zolari yoki loyiha boshqaruvchilari jarayonga o‘zgartirish kiritishga qarshilik qalishlari mumkin, o‘zgartirishlar ta‘sir ko‘rsatmasligini sabablarini keltirishlari mumkin va o‘zgartirish kiritishni orqaga surishlari mumkin. Jarayonga taklif etilgan o‘zgartirishni samarasiz ekanligini ko‘rsatish maqsadida ular ma‘lumotlarni o‘zgartirish va interpritasiya qilish jarayonini qastdan qiyinlashtirishlari mumkin.

2. *Qat’iylikni o‘zgartirish*. Bu o‘zgartirishlarni jarayonga joriy etish mumkin bo‘lsada, bu yangiliklar qisqa vaqt ichida bekor qilinib jarayon yana o‘z holiga kelib qoladi.

O‘zgartirishga qarshilik o‘zgartirilayaotgan jarayonda qatnashilayotgan loyiha boshqaruvchisi va muhandisidan bildirilishi mumkun. Loyiha boshqaruvchilari ko‘pincha nomalum havf-hatarlarga duch kelishi mumkun bo‘lganligi uchun har qanday yangilikka qarshi chiqadilar. O‘zgartirish dasturiy ta‘minotni ishlab chiqarilishini tezlatish yoki dasturiy ta‘minotdagi defektlarni kamaytirishga qaratilgan bo‘lishi munkun. Lekin jarayonga kiritilgan o‘zgartirishlar samarasiz bo‘lishi yoki o‘zgartirish kiritishga ketgan vaqt tejalishi mumkin bo;lgan vaqtdan oshib keshi mumkin. Loyiha boshqaruvchilari dasturiy ta‘minot liyihasi vaqtida va byudjetdan chiqmagan holda bajarilishini nazorat qiladi. Shuning uchun ular samarasiz bo‘lsa ham o‘ziga ma‘lum bo‘lgan jarayonni qisqa muddatli havf-hatarlar uchraydigan yaxshilangan jarayondan afzal biladilar.

Muhandislar o‘zgartirishlarga shu sabablarga ko‘ra qarshilik ko‘rsatishi mumkin yoki ular bu jarayonlarni o‘zining proffessionallizimiga hatar sifatida qabul qiladi.

Yangi jarayon ularning qobilyat va tajribasini yo‘qqa chiqaradi deb xisoblashadi. Yangi jarayonda kam odam ishlatilishi va ular ishini yo‘qotishi mumkin deb o‘ylashadi. Balki ular yangi usul, vosita va ko‘nikmalarni o‘rganishni hohlashmas.

Boshqaruvchi sifatida siz o‘zgartirilayotgan jarayonda qatnashayotgan odamlarni tuyg‘ulariga etibor berishingiz lozim. Siz jamoani to‘lig‘icha jarayon o‘zgartirishiga chorlashingiz kerak, ularning shubhalarini tushunishingiz kerak. va yangi jarayonni rejalashtirishga tortishingiz lozim. Jarayonni o‘zgartirishga qiziqtira olsangiz ular bu o‘gartirish ishlashini o‘zlari hohlaydilar. 1990 yillarda urufga kirgan jarayonda radikal o‘zgartirishlar kiritgan biznis-jarayonni qayta ishlashlar (Hammer, 1990; Ould, 1995) ko‘p holatlarda muvaffaqiyatsiz bo‘lgan chunki ularda odamlarning tashvishlari inobatga olinmagan.

Loyiha grafik va sarflarga salbiy ta‘sir ko‘rsatuvchi o‘zgartirishlar bilan ishlaydigan loyiha boshqaruvchilarini muammolarni yechish uchun siz o‘zgartirishga ketadigan qo‘shimcha sarflarni hisobga olib loyiha byudjetini ko‘paytirishingiz lozim. Shuningdek siz o‘zgartishning samarasi qisqa muddatli bo‘lishiga nisbatan realist bo‘lishingiz kerak. o‘zgartirishlar katta mashtabdagi mukammallashtirishlarga olib kelmaydi. Jarayonni o‘zgartirishni natijasi uzoq muddatda bilinadi shuning uchun siz jarayon o‘zgartirishni bir nechta loyihada quvvatlashingiz lozim.

Kelgusida rad etilgan o‘zgartirishlarni muammosi bitta. Mukammallashtirishga olib keladigan o‘zgartirishlarni ularga qattiq ishongan “evangelist” tomonidan taklif etilgan bo‘lishi mumkin. U o‘zgartirishlar samarali ekanligini kafolatlash uchun qattiq ishlashi mumkin va yangi jarayon qabul qilinadi. Lekin agar ‘evangilist’ biror joyga ketsa uning o‘rniga yangi jarayonga o‘zini kamroq baxshida etadigan shaxs qo‘yilishi mumkin. Shuning uchun jarayonda qatnashadigan oodamlar oldingi usullarga qaytishi mumkin. Bu

ayniqsa kiritilgan o‘zgartirishlar universal bo‘lganda va jarayon o‘zgartirishning to‘liq foydasi bilinmagan hollarda bilinadi.

5.5. bobda ko‘rilgan CMMI modelida doimiylik muammosini jarayon o‘zgartirishini institutsionalizatsiyasi haqida ko‘p baxs brogan. Buda o‘zgartirish jarayoni alohida ko‘rilmaydi balkim butun kompaniya tomonidan qabul qilingan umumiy amalyotga aylanadi.

5.5. CMMI jarayonini qurishni takomillashtirish

U.S. Dasturish Injinering Instituti (Software Engineering Institute (SEI)) Amerika dusturlari sanoatining salohiyatini takomillashtirish uchun barpo etilgan. 1980-yilning yarmida, SEI dasturiy taminot bitim tuzuvchilarining salohiyatini baholash yo‘llarini o‘rganish boshlagan. Qobilyatli bu baholovchilarning natijasida SEI dasturiy taminoti Qobilyatli Yetuk Model (Capability Maturity Model (CMM)) paydo bo‘lgan (Paulk et al., 1993; Paulk et al., 1995). Muhim jarayoni rivojlantira olishda bu dasturiy injenringi jamoasi ishonchiga jiddiy ta‘sir ko‘rsadigan.

CMM dasturiy taminotini boshqa solohiyatili yetuk modellarning orqali mustahkamlantirilgan, shu jumladan bularga Qobilyatli Yetuk Odamlar Modeli(including the People Capability Maturity Model (P-CMM) (Curtis et al., 2001)) va Salohiyatli Model Injeneringi Tizimlari)the Systems Engineering Capability Model (Bate, 1995)) kiradi.

Boshqa tashkolotlar shuningdek qulay yetuk modellarni yaratishgan. SPICE usuli baholash qobilyatida va jarayonlar rivojlantirishda(Paulk va Konrad, 1994) SIE modeliga qaraganda ancha yaxshiroqdir. U CMM yuzalari bilan qiyoslangan yetuk yuzalarni o‘z ichiga oladi, lekin jarayonlarni turini ham aniqlaydi, shuningdek mijozlarga qulaylik tug‘diradi, bu esa yuzalarni(ishni) tezlashtirishga yordam beradi. So‘ngi natijani ortirishda, qisqartiriladigan bu jarayonlarni ko‘paytirish juda muhumdir.

1990-yildagi Bootstrap loyihasi kengaytirishning maqsadida va SEI yetuk modelini uzgartirish ishlatiladi, u kompanyalarning bir-biri bilan aloqada bo‘lishini ta‘minlaydi. Bu model (Haase et al., 1994; Kuvaja, et al., 1994) SEIning yutuk bo‘limlarini qullaydi(5.5.1-bo‘limda muhokama qilingandi). Shuningdek u jarayonlar bazasini model vazifasini bajaradi(bazalashtirilgan modil Yevro Koinot Agentligida qullanilgan) bu esa jarayonlarni ta‘rishlash uchun qullaniladigan maqsad bo‘lishi mumkin. U jarayonlarni o‘stirishni qullab quvvatlashda katta komlaniya qurish harakterishtikasi tizimi uchun ma‘lumotlarni o‘z ichiga oladi.

Sinab ko‘rishda yetuk jarayonlarning bazalashtirilganlarini eng yaxshi moldellarning barchasini birlshtiradi(ular shaxsiy modellarni o‘z ichiga oladi), SEI birlashtirilgan eng yaxshi modelni (CMMI) yaratishda yangi dasturni ishga tushuradi. CMMI fremvorki Dasturlarni va CMM Injenerlik Tizimlarini va boshqa yetuk modellarni qo‘shishni uzgartiradi. U ikki bo‘limdan iborat, markazda turish va davom etish, shuningdek CMM Software dagi nosozliklarning bazi nomlarini ham ko‘rsatadi.

CMMI modeli jarayonlarni rivojlantirish uchun fremwork sifarida yaratilgan bo‘lib bunda juda katta birlashgan kompaniyalarni o‘z ichiga oladi. Ular Software CMM bilan ishni qoniqarli bajarishni ta‘minlaydi va tashkilotning tizimini yuksaltirishga yo‘l ochadi va jarayonlar boshqaruvini natishsi bo‘ladi va 1-bo‘limdan 5-bo‘limga junatadi. Ularning davom ettirish bo‘limi yetuk jarayonlarning kattalashish-kichiklashish holati uchun ruhsat beradi. Bu model eng kerakli 22 ta jarayon atrofidagi yo‘ni 0 dan 5 gacha tezligini oshirishni ta‘minlaydi(5.7- rasmda ko‘rsatilgan).

CMMI modeli juda kompleks bo‘lib 1000 sahifa yoki undan ko‘proq tasvirlash mumkin. Bu joyda uni muxokama qilish uchun menda asosiy sabab bor. Asosiy model komponentalari quyidagilar:

1. Jarayonlarn maydonlarini bog‘lash bunda dasturlar jarayonlarining faoliyatiga bog‘liqdir. CMMI 22 bir hil jarayon maydonlari bu qulay dasturiy jarayonlarga va takomillashtirishga bog‘liqdir. CMMI modelini davom ettirish turt

guruhni tashkil etadi. 5.7-rasmda bu guruhlar va jarayonlar maydonnini aloqadorligi ko‘rsatilgan.

2. Maqsadlarning biri bu kerakli holatining qaysidir tur abstraklari qulay tashkillashtirilgan bo‘lishi kerak. CMMI ning aniq maqsadlari bu har bir jarayon maydoni va bu maydon uchun kerakli joyni to‘g‘ri tanlash orqali birlashtirishdir. Shungdek u umumiy maqsadlarni tushuntiradi bu esa yaxshi tajriba joyi bilan birlashadi.

3. Yaxshi tajribalarning bog‘lanishi, oxiriga yetgan maqsad yullarining tasvirlanishi. Bir qancha aniq va umumiy tajribalar jarayonlar maydoni ichidagi har bir maqsad bilan birlashishi mumkin. Tavsiya qilingan tajribalarning ba‘zilari 5.9-rasmda ko‘rsatilgan. Biroq, CMMI maqsadga erishish yo‘liga qaraganda u ancha afzalrog‘ini ko‘rsatadi, bu esa muhumdir. Tashkilotlar CMMI maqsadlarini ba‘zan natijaga erishishda kerak tajribalarni qo‘llashi mumkin – ular CMMIdagi tavsifa qilingan misollarni o‘zgartirishlari shart emas.

Kategoriya	Jarayon maydoni
Jarayon boshqaruvi	Tashkiliy aniqlik jarayoni(Organizational process definition (OPD)) Tashkiliy markaziy jarayoni(Organizational process focus (OPF)) Tashkiliy tayyorgarlik(Organizational training (OT)) Tashkiliy jarayon ishlashi(Organizational process performance (OPP)) Tashkiliy riojlanish va innovatsiyasi(Organizational innovation and deployment (OID))
Loyiha boshqaruvi	Loyihani rejalashtirish(Project planning (PP)) Loyihani kuzatish va nazorat qilish(Project monitoring and control (PMC)) Yetkazib beruvchi kelishuv boshqaruvi(Supplier agreement management (SAM))

	Loyiha boshqaruviga qo‘shilish(Integrated project management (IPM)) Havf-hatar boshqaruv(Risk management (RSKM)) Miqdoriy loyiha boshqaruvi(Quantitative project management (QPM))
Injiniring	Talablar boshqaruvi(Requirements management (REQM)) Talablar ishlanmasi(Requirements development (RD)) Texnik yechim(Technical solution (TS)) Maxsulotni integratsiyalash(Product integration (PI)) Tekshirish(Verification (VER)) Tekshirish(Validation (VAL))
Himoya vositasi	Konfiguratsion boshqaruv(Configuration management (CM)) Boshqaruvni jarayoni va sifati(Process and product quality management (PPQA)) O‘lchash va tahlil qilish(Measurement and analysis (MA)) Qaror va qarorni tahlil qilish(Decision analysis and resolution (DAR)) Sabab, qaor va tahlil(Causal analysis and resolution (CAR))

5.7- rasm. CMMI dagi jarayon maydoni

Birlashgan g‘oyalar va tajribalar texnik bo‘lmaydi lekin yaxshi tajribalarning joyi bilan bog‘lanadi. Tashkilotning so‘ngisigahi bog‘liqliklar nima ma‘no bildiradi. Yetuk rivojlanishning erta bosqichi, odatiy kafolar bo‘lishi mumkin bu rejalar esa tashkillashtirilgan va kompaniyadagi barcha rivojlangan dasturlar uchun belgilangan jarayonlar hissoblanadi. Biroq, ko‘plab rivojlanish bilan tashkilot uchun taraqiylashgan jarayonlar, joylashuv statistik qo‘llashni boshqarish jarayonini tanishtirish ma‘nosi bildiratigan bo‘lishi mumkin va boshqa tashkillashdagi kerakli texnologiyalarni sonini qisqartirish hamdir.

Maqsad	Jarayon maydoni
Aniq harakatlar loyihani qachondir bajarilishi tashkillashtirish yoki rejadan ma‘noli natijalar olishni ta‘minlaydi.	Loyiha rahbarligi va nazorat (aniq maqsad)
Haqiqiy ijro va loyiha rejasiga qarshi o‘rgatiladigan loyiha progressi.	Loyiha rahbarligi va nazorat (aniq maqsad)
Talablarni tahlil qilish va ma‘qullash, shuningdek talab etilgan qoida vazifasini yaratish.	Talablarni rivojlantirish (aniq maqsad)
Kamchiliklarning sabablari va boshqa muammolarni tizimini aniqlash.	Aloqani tahlil qilish va qaror (aniq maqsad)
Jarayonni joylashtirish so‘ng uni o‘rnatish.	Umumiy maqsad

5.8- rasm. CMMI dagi jarayon maydonlari

CMMI baholashda tashkillotdagi imtixon jarayonlarini o‘z ichiga oladi va bu jarayonlarni reytingi yoki 6 jihatli jarayon maydonlarini bu esa har bir jarayon maydonidagi so‘ngi bosqichga bog‘lanadi. Bu esa ko‘plab yetuk jarayonlarga qulaydir. 6 jihatli bog‘lanishni jarayon maydonidagi so‘ngi boshqichini aniqlash quyidagicha:

1. *Tugallanmagan*- aniq maqsadlarning eng kichigi bo‘lib jarayon maydonlarini moslashtirish bilan birlashtiriladi. Bu yuzada umumiy maqsadlar yo‘q bo‘lib tugallanmagan jarayonlarning joylashuvi hech qanday ma‘no anglatmaydi.

2. *Bajarilgan*- g‘oyalar(maqsadlar) jarayon maydonini moslashtirish bilan birlashtiriladi, va ishlash doirasidagi barcha jarayonlar uchun jamoaviy a‘zolar sifatida harakatda bo‘lishini va aniq aloqada bo‘lishni ta‘minlaydi.

3. *Boshqarish*- bunda maqsadlar jarayon maydonini uchrashtirish orqali birlashtiriladi va qachondir har bir jarayonni qo‘llash kerak bo‘lganda ularni aniq

joyda to‘playdi. Loyiha rejalarida loyihadagi maqsadlar aniq hujjatlashtirilgan bo‘lishi shart. Boshqaruv manbasi va jarayon tahlil usullarini tashkil qilish orqali joylashtirilgan bo‘lishi shart hissoblanadi.

4. *Aniqlangan*- bunda tashkillashtirish standarti va jarayonlar joylashuvi markazlashtiriladi. Barcha loyihani boshqaruv jarayoni bo‘ladi bu esa tashkillashtirish jarayonlarining aloqasini aniqlash orqali loyiha talablari moslashtiriladi. Foydali jarayon va jarayon o‘lchamlari kelajagdagi jarayon rivojlanishlari uchun to‘plangan va qo‘llab ko‘rilgan bo‘lishi shart.

5. *Boshqarilgan miqdor*- bu esa ma‘suliyatli tashkillashtirish bo‘lib statistik holda qo‘llaniladi va jarayonlarni almashtirishni boshqarishda boshqa miqdordagi metodlar ishlatiladi; bu esa to‘plangan jarayon va savdo darajalari boshqaruv jarayonida qo‘linilgan bo‘lishi shartligini anglatadi.

6. *Optimistlik*- bu eng yuqori bosqich bo‘lib , bunda rivojlantirish jarayonini boshqarishdagi savdo darajalari va jarayonni ishlata olish tashkillangan bo‘lishi shart. G‘oyalar taxlil qilingan bo‘lishi shart va jarayonlar biznesdagi kerakli almashinuvlarni moslashtira olishi zarurdir.

Maqsad	**Tajribalarni birlashtirish**
Talablarni tahlil qilish va tasdiqlash, shuningdek talab qilshish vasifasini aniqlashni yaratish.	Tahlil qilshish talablar tizimiga kafolat berishni boshqaradi bu esa ularni kerakli va yetarli bo‘lishini ta‘minlaydi.
	Tasdiqlash esa talablarga kafolat beradi bu esa savdo natijasini foydalanuvchilarning atrof-muhitagi istagini amalga oshirish uchun hizmat qiladi, ko‘plab mos keladigan texnikalarni qullashga imkon yaratadi.
Kamchiliklarning sabablari va boshqa muammolarni tizimini	Tahlil qilish uchun muhum kamchiliklarni va boshqa muammolarni saralash.

aniqlash.	
	Saralangan kamchiliklarning tahlillarini va boshqa muammolar aloqasi bajarish shuningdek rejalashtiriladigan vazifalarni adreslash.
Jarayonni joylashtirish so‘ng uni o‘rnatish.	Jarayonlarni yaratish talablarini bajarish va reja uchun tashkillashtirish hujjatini yaratish va qo‘llab quvvatlash.
	Harakatlantirish jarayoni uchun huquq va ma‘suliyat tayinlash, mahsulotlarni ishlashini rivojlantirish va rivojlanish jarayoni talablarining hizmatlarini ta‘minlash.

5.9- rasm. CMMI dagi maqsadlar va tajribalarni birlashtirish

Bu yuqori boshqichlarning juda sodda turi bo‘lib, tajribada sinaladi, to‘la turlari bilan ishlashingiz kerak bo‘ladi. Bosqichlar rivojlanadi, eng yuqori yuzadagi dasturlar va jarayonning darajalarini boshqarishni rivojlantirish va jarayon almashinuvi ta‘minlanadi, jarayonni standartlashtirish orqali eng pastgi yuzalarda jarayon turlari aniqlanadi. Jarayonlarni rivojlantirishda kompaniya jarayon guruhlarining yetuk bosqich o‘stirishida maqsadga erishishlari kerakdur bu esa biznesda juda katta ahamiyatga egadir.

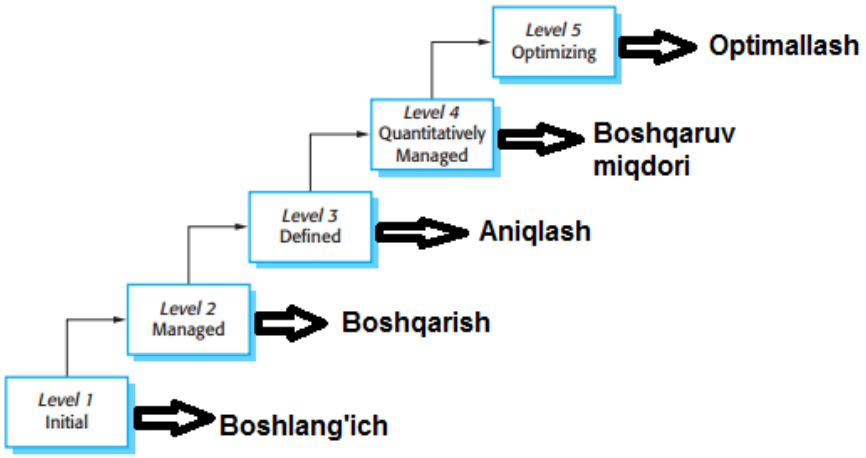
5.5.1. CMMI model bosqichi

CMMI model bosqichi Software Capability Maturity Model ya‘ni yetuk modelli eng zo‘r dasturlar deb ta‘riflanadi bunda uning asm ma‘nosi besh boshqichning biridagi tashlikkashtirishning eng mukammal jarayoni aniqlash deganidir, shuningdek maqsadlarni tayinlangan qoidalari bo‘lib bunda besh

boshqichlarning har biridagi shun maqsadlarga erishish kerak bo‘ladi. Rivojlantirish jarayoni har bir bosqichda tajribalardan o‘tkaziladi, pastdan yuqori boshqichga o‘tkazish model ichida amalga oshiriladi.

CMMI model boshqichidagi besh boshqich 5.10- rasmda ko‘rsatilgan. Ular model davomiyligi ichida 1- bosqichdan 5- boshqichga o‘tishni samarali tarzda moslashtiradi. Kalit CMMI modellar davomiyligi va bosqich orasidagi farq hissoblanadi bu esa model bosqichini to‘liq tashkillashtirish mahorati bilan aniqlab uni qo‘llashga imkon yaratib beradi, holbuki model davomiyligi tashkillashtirish doirasidagi jarayon aniq maydonlarining eng so‘ngisini aniqlaydi.

Har bir yetuk model umumiy maqsadlar va jarayon maydonlarining bog‘lashni birlashtirish xusuyatiga egadir. Yaxshi dasturiy injineringi, boshqaruv tajribasi va rivojlantirish jarayoni joyi haqida chuqur muluhazalar yuritiladi chunki bu uchulasi juda muhum ahamiyatga ega hissoblanadi. Pastgi boshqichlarda tajribani yaxshi tanishtirishga erishish mumkin bo‘ladi; biroq, yuqori bosqichlar esa jarayoni darajasini oshirish va rivojlantirishni talab qiladi.



5.10- rasm. CMMI mukammal model bosqichi

Misol uchun, ikkinchi bo‘lim (boshqarish bo‘limi) bilan birlashtirilgan modeldagi jarayon maydonlarini aniqlaylik:

1. *Talablar boshqaruvi* – Loyiha mahsulotlarining talablarini va mahsulot komponentalarini boshqarish, shuningdek talablar, loyiha rejalari va ishlaydigan mahsulotlar orasidagi o‘xshash jihatlarini aniqlash.

2. *Loyihani rejalashtirish*- Tashkil qilish va rejalarni qo‘llab quvvatlash bu esa loyiha faoliyatlarini aniqlaydi.

3. *Loyiha rahbarligi va nazorat*- Loyihaning rivojlanishi uchun choralar ko‘rish shunga bu aniq moslashtirish holatlari qachondir rejadan chiqish loyihasini bajarishni nazarda tutush imkonini beradi.

4. *Kelishuv boshqaruvini ta‘minlash*- Mahsulotlar boshqariviga erishish va mavjud rasmiy kelishuv uchun loyihaga tashqi ta‘minotlar hizmatlari muvofiqlashtirish.

5. *O‘lchash va tekshirish*- Mukammal o‘lchashni yaratish va qo‘llav quvvatlash bunda axborotga kerak bo‘ladigan boshqaruv yordamidan foydalaniladi.

6. *Ishonchli jarayon va mahsulot sifati*- Hodimlar va jarayonlardagi toza obyekni boshqarish choralarini ko‘rish shuningdek ishlaydigan mahsulotlarni birlashtirish.

7. *Konfiguratsiya boshqaruvi*- Ishlaydigan mahsulotlar softligini konfiguratsiya tengligidan foydalanishni, konfiguratsiya nazoratini, konfiguratsiya huquqiy holati hissobini, konfiguratsiya tekshirishlarini yaratish va qo‘llab quvvatlash.

Shuningdek ushbu aniq tajribalar, CMMI modelidagi ikkinchi boshqichda tashkillashtirishlar operatsiyasi jarayon boshqaruvidagi har jarayonlar joylashuvining umumiy maqsadga erishish uchun imkon yaratib beradi.

Tajribalar tashkilotining misollari rejalashtirish loyihasi bilan birlashgan bo‘lib bu boshqariv jarayoni bo‘layotganda rejalashtirish loyihasi jarayonini nazorat qilishga imkon yaratiladi:

- Jarayonlarni rejalashtirish loyihasini bajarish va reja uchun tashkillashtirish hujjatini yaratish va qo‘llab quvvatlash.
- Boshqariv jarayoni loyihasini bajarish uchun loyiq keladigan resurslarni, ishlaydigan mahsulotlarni yaratishni, shuningdek jarayon servislari ta‘minoti choralarini ko‘rish.

- Rejaga qarshi loyiha rejalashtirish jarayonini boshqarish va nazorat qilish hamda holatga to‘g‘ri keladigan munosibini olish.
- Faolliklarni, huquqiy holatni, eng yuqori bosqich boshqaruvi bilan loyiha rejalashtirish jarayonining natijalarini va ba‘zi masalalar qarorini qaytadan tekshirib chiqish.

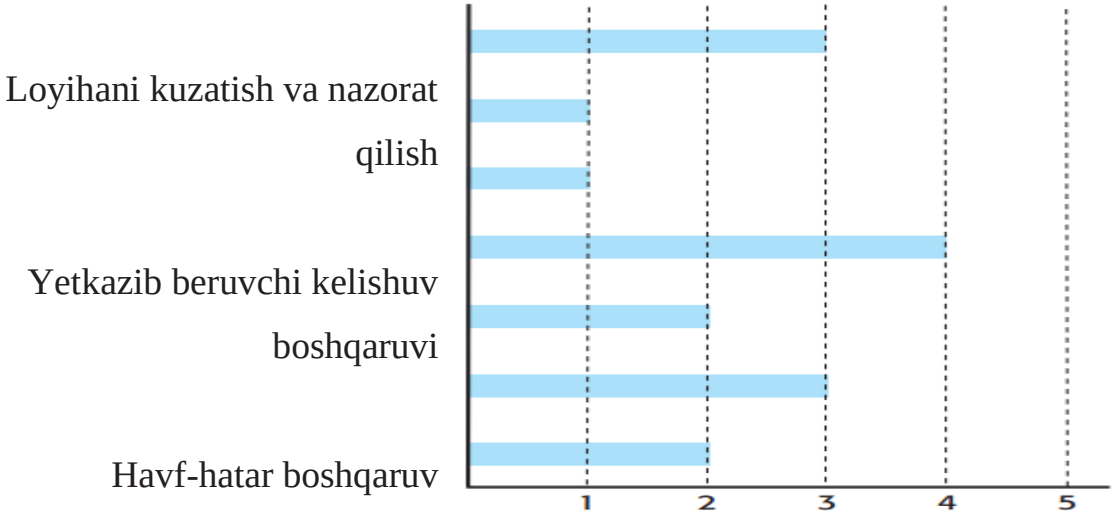
CMMI bosqichining afzalligi u yetuk modelli mukammal dasturlar bilan uygu’nligi bo‘lib bu 1980-yilning ohiriga rejalashtirilgan edi. Ko‘plab kompaniyalar tushunishadi va rivojlantirish jarayoni uchun bu modelni qo‘llab ko‘rishgan. U shuning uchun ularga sodda bo‘lib bundan CMMI model bosqichiga o‘tish davrini yaratishda foydalanishdi. Bundan tashqari model bosqichi tashkilotlar uchun yuqori rivojlantirishning eng qisqa yo‘nini aniqlaydi. Ular ikkinchidan uchinchiga o‘tishni rejalashtira olishadi va undan yuqoriroqni ham.

Model bosqichining zarari meyor (CMM dasturlarining ham) biroq, ular tabiiy buyruq beradi. Har bir yetuk bosqichning shaxsiy maqsadlari va tajribalari bor. Model bosqichi keyingi bosqichga o‘tish davridan oldin bir boshqichdagi barcha maqsadlar va tajribalarni bajarib ko‘rishni zimmasiga oladi. Biroq, tashkillashtirish muhitlari ko‘p bo‘lishi mumkin bu esa pastgi bosqich tajribalaridan oldin yuqori bosqichlardagi tajribalarni va maqsadlarni uyg‘unroq amalga oshirish imkonini yaratadi. Qachondir biror tashkilot buni bajarganda, so‘ngi baholash uning xato tushunchasini namoyon qiladi.

5.5.2. CMMI model davomiyligi

Yetuk modellar davomiyligi bosqichlar muhovazasiga muvofiq ravishda tashkillashtirish sinifiga ajratmaydi. Yana, ular modellarni bezak qisimi hissoblanadi bu esa mustaqi yoki tajribalarning guruhlari nazarda tutadi va har bir jarayon gurugidagi yaxshi tajribalarning qo‘llanilishini aniqlaydi. Eng so‘ngi baholash yo‘q, shuning uchun, yagona qiymat hissoblanadi lekin jarayon guruhi yoki har bir jarayon uchun tashkillashtirishning eng so‘ngi ko‘rinishi qiymatlarining bog‘lanishi deb ham ataladi.

CMMI davomiyligi 5.7- rasmda jarayon maydonlari hissobga olgan holda ko‘rsatilgan va U 0 dan 5 gacha har bir jarayon maydonida mukammal baholash bosqichini aniqlashga hizmat qiladi. Odatda, tashkillotlar har hil jarayon maydonlari uchun har hil yetuk bosqichlarda operatsiyalarni amalga oshirishadi. Demak, CMMI baholash davomiyligining natijasi har bir jarayonning mukammal profil ko‘rinishi hissoblanadi va ular yuksak darajali baholashni bog‘laydi. Mukammal profilning fragmenti har hil mukkammal boshqichlardagi jarayonlarni ko‘rsatadi (5.11- rasm). Bu boshqaruv konfiguratsiyasidagi yetuk boshqichni ko‘rsatasi, misol uchun, u yuqori ko‘rasatsa unda havfli boshqaruv pastgisini ko‘rsatadi. Kompaniya haqiqiy va mukammal maqsad profillarini yaratishi mumkin, maqsad profile salohiyatli bosqichni aks ettiradi bu esa ular jarayon maydoniga erishishga intilishini anglatadi.



5.11- rasm. Qobilyat profil jarayoni

Model davomiyligining asosiy foydasi bu kompaniyalar haridorning shaxsiy talablari va istaklariga munosib jarayonlarni tanlar va to‘play olishidir Mening tajribamda, tashkilotning har hil turlarini yaxshilash jarayoni uchun har hil talablari mavjud. Misol uchun, biror kompaniya belgilangan tizimdagi yaxshilanishlarda

markazda bo‘lishi mumkin bo‘lgan airoplan sanoati, konfiguratsiya boshqaruvi, qonuniylik uchun dasturlarni yaratadi, baholanki web rivojlanish kompaniyasi savdo jarayonlari bilan ko‘proq aloqador bo‘lishi mumkin. Model bosqichi navbatdagi har hil bosqichlarda kompaniyalarni jamlaydi. Taqqoslash, CMMI davomiyligi ehtiyotkorlikka va o‘zgaruvchanlikka ruxsat beradi, hali hanus kompaniyalarga CMMI rivojlanish doirasida ishlashi uchun imkoniyat yaratadi.

Asosiy tushunchalar

- Jarayonni mukammallashtirish maqsadlar-maxsulotning yuqori sifati, ishlab chiqarishga ketadigan sarflarning kamayishi va dasturiy ta‘minotning tezlik yetkazib berish.
- Jarayonni mukammallashtirishda jarayonning sariflarini kamaytirish bilan bog‘liq bo‘lgan tezkor yondashuv va jarayonlarni yaxshu boshqarishga va dasturlashning yaxshi amalyotidan foydalanishga asoslangan yetuk yondashuvlar.
- Jarayonni mukammallashtirish sikliga jarayonni o‘lchash, taxlil qilish, modellashtirish va o‘zgartirish kiradi.
- Jarayondagi amallarni va dasturiy ta‘minot maxsulotlari bilan munosabatni ko‘rsatuvchi jarayon modellari jarayonni tasniflash uchun ishlatiladi. Amalda dasturiy ta‘minotni ishlab chiqaruvchi muhandislar doimo modellarni o‘z sharoitlariga moslab oladilar.
- O‘lcho‘v dasturiy ta‘minotda ishlatilayotgan jarayon haqidagi aniq savollarga javob toppish uchun ishlatilishi kerak. Bu savollar mukammallashtirishning tashkiliy maqsadlariga asoslanishi kerak.
- O‘lchash jarayonida ishlatiladigan 3 ta turdagi parametirlar-vaqt parametri, foydalanilgan resurs parametri va voqealar paremetri.
- CMMI jarayonining yetuklik modeli bu uzluksiz tashkillangan jarayonning quvvatlash va mukammallashtirishga asoslangan jarayonning mukammallashtirishning integratsionallangan modeli.
- CMMI modelida jarayonni mukammallashtirish qator maqsadlarga erishishga asoslangan, dasturlashning yaxshi amaliyoti, tasnifi, standartlashtirish

va usullarni boshqarishorqali bu maqsadlarda erishiladi. CMMI modelini ichiga ishlatilishi tavsiya etilgan usullar kiradi lekin ular ishlatilishi shart emas.

Nazorat savollari:

1. Tezkor yondashuv va jarayon yetukligi yondashuv o‘rtasidagi muhum farqlar nimalar?

2. Qanday sharoitlarda maxsulot sifatiga jamoa sifati ta‘sir etadi? Alohida iqtidor va qobilyatga bog‘liq bo‘lgan dasturiy ta‘minot turlarini misolini keltiring.

3. Tashkilotda jarayonni mukammallashtirish dasturini quvvatlovchi 3 ta maxsusu dasturiy maxsilotni taklif qiling.

4. Tasavvur qiling tashkilotdagi jarayonlarni mukammallashtirishdan maqsad rivojlantirish vaqtida ishlab chiqilgan ko‘p marotaba ishlatiladigan komponentlar miqdorini ko‘paytirishdur. Bunga olib keladigan GQM padigma doirasida 3 ta savol taklif qiling.

5. Mukammallashtirish jarayoni qismi sifatida yig‘iladigan dasturiy ta‘minot jarayonini 3 turdagi metrikaga tasnif bering. Har bir turdagi metrikani misolini keltiring.

6. Jarayonni baxolash uchun loyihalang va jarayonni o‘zgartirish bo‘yicha takliflarni priareiti bo‘yicha joylashtiring. Bu jarayonni bu jarayonga qatnashgan ro‘llarni ko‘rsatgan jarayon modeli sifatida rasmiylashtiring. Jarayonni tasniflash uchun UML faoliyat diagrammalari yoki BPMN dan foydalanishingiz kerak.

7. CMMI kabi jarayonni mukammallashtirish tuzilmasida aks ettirilgan baxolash va mukammallashtirishniamalga oshiruvchi yondashuvning ikkita afzalligi va ikkita noqulayligini ayting.

8. Qanday sharoitlarda siz CMMI instenirovkasidan foydalanishni tavsiya qilasiz?

9. Jarayon yetuklik modelidan foydalanishning afzalliklari va noqulayliklari nimada?

10. Jarayondagi odamlarning ishini o‘zgartiruvchi jarayonni mukammallashtirish dasturlarni degumanizatsiya deb tushunish mumkinmi?

Jarayonni mukammallashtirish dasturiga qanday qarshilik paydo lishi mumkin va nimaga?

ASOSIY ADABIYOTLAR RO‘YHATI

1. O‘zbekiston Respublikasini yanada rivojlantirish bo‘yicha harakatlar strategiyasi to‘g‘risida. O‘zbekiston Respublikasi Prezidentining PF-4947-son farmoni. Toshkent, 2017 yil 7 fevral.

2. Aron, J. D. (1974). The Program Development Process. Reading, Mass.: Addison-Wesley.

3. Baker, F. T. (1972). “Chief Programmer Team Management of Production Programming”. IBM Systems J., 11 (1), 56–73.

4. Bass, B. M. and Dunteman, G. (1963). ‘Behaviour in groups as a function of self, interaction and task orientation’. J. Abnorm. Soc. Psychology., 66 (4), 19–28.

5. Beck, K. (2000). extreme Programming Explained. Reading, Mass.: Addison-Wesley.

6. Boehm, B. W. (1988). ‘A Spiral Model of Software Development and Enhancement’.
IEEE Computer, 21 (5), 61–72.

7. Brooks, F. P. (1975). The Mythical Man Month. Reading, Mass.: Addison-Wesley.

8. Hall, E. (1998). Managing Risk: Methods for Software Systems Development. Reading, Mass.: Addison-Wesley.

9. Marshall, J. E. and Heslin, R. (1975). ‘Boys and Girls Together. Sexual composition and the effect of density on group size and cohesiveness’. J. of Personality and Social Psychology, 35 (5), 952–61.

10. Maslow, A. A. (1954). Motivation and Personality. New York: Harper and Row. Ould, M. (1999). Managing Software Quality and Business Risk. Chichester: John Wiley & Sons.

11. Beck, K. (2000). extreme Programming Explained. Reading, Mass.: Addison-Wesley.

12. Boehm, B. 2000. ‘COCOMO II Model Definition Manual’. Center for Software Engineering, University of Southern California. http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf

13. Boehm, B., Clark, B., Horowitz, E., Westland, C., Madachy, R. and Selby, R. (1995). ‘Cost models for future life cycle processes: COCOMO 2’. Annals of Software Engineering, 1 57–94.

14. Boehm, B. and Royce, W. (1989). ‘Ada COCOMO and the Ada Process Model’. Proc. 5th COCOMO Users’ Group Meeting, Pittsburgh: Software Engineering Institute.

15. Boehm, B. W. (1981). Software Engineering Economics. Englewood Cliffs, NJ: Prentice Hall.

16. Boehm, B. W., Abts, C., Brown, A. W., Chulani, S., Clark, B. K., Horowitz, E., Madachy, R., Reifer, D. and Steece, B. (2000). Software Cost Estimation with COCOMO II. Upper Saddle River, NJ: Prentice Hall.

17. Londeix, B. (1987). Cost Estimation for Software Development. Wokingham: Addison-Wesley.

18. Myers, W. (1989). ‘Allow Plenty of Time for Large-Scale Software’. IEEE Software, 6 (4), 92–9

19. Putnam, L. H. (1978). ‘A General Empirical Solution to the Macro Software Sizing and Estimating Problem’. IEEE Trans. on Software Engineering., SE-4 (3), 345–61.

20. Schwaber, K. (2004). Agile Project Management with Scrum. Seattle: Microsoft Press.

Qo‘shimcha adabiyotlar ro‘yhati:

1. Mirziyoev Sh.M. Tanqidiy tahlil, qat’iy tartib-intizom va shaxsiy javobgarlik – har bir rahbar faoliyatining kundalik qoidasi bo‘lishi kerak. O‘zbekiston Respublikasi Vazirlar Mahkamasining 2016 yil yakunlari va 2017 yil istiqbollariga bag‘ishlangan majlisidagi O’zbekiston Respublikasi prezidentining nutqi. // Xalq so’zi gazetasi. 2017 yil 16 yanvar, №11.

2. Ahern, D. M., Clouse, A. and Turner, R. (2001). CMMI Distilled. Reading, Mass.: Addison-Wesley.

3. Basili, V. and Green, S. (1993). ‘Software Process Improvement at the SEL’. IEEE Software, 11 (4), 58–66.

4. Basili, V. R. and Rombach, H. D. (1988). ‘The TAME Project: Towards Improvement-Oriented Software Environments’. IEEE Trans. on Software Eng., 14 (6), 758–773.

5. Bate, R. 1995. ‘A Systems Engineering Capability Maturity Model Version 1.1’. Software Engineering Institute.

6. Chrissis, M. B., Konrad, M. and Shrum, S. (2007). CMMI: Guidelines for Process Integration and Product Improvement, 2nd edition. Boston: Addison-Wesley.

7. Curtis, B., Hefley, W. E. and Miller, S. A. (2001). The People Capability Model: Guidelines for Improving the Workforce. Boston: Addison-Wesley.

8. Haase, V., Messnarz, R., Koch, G., Kugler, H. J. and Decrinis, P. (1994). ‘Bootstrap: Fine Tuning Process Assessment’. IEEE Software, 11 (4), 25–35.

9. Hammer, M. (1990). ‘Reengineering Work: Don’t Automate, Obliterate’. Harvard Business Review, July–August 1990, 104–112.

10. Humphrey, W. (1989). Managing the Software Process. Reading, Mass.: Addison-Wesley.

11. Humphrey, W. S. (1988). ‘Characterizing the Software Process’. IEEE Software, 5 (2), 73–79.

12. Humphrey, W. S. (1995). A Discipline for Software Engineering. Reading, Mass.: Addison-Wesley.

13. Kuvaja, P., Similä, J., Krzanik, L., Bicego, A., Saukkonen, S. and Koch, G. (1994). Software Process Assessment and Improvement: The BOOTSTRAP Approach. Oxford: Blackwell Publishers.

14. Osterweil, L. (1987). ‘Software Processes are Software Too‘. 9th Int. Conf. on Software Engineering, IEEE Press, 2–12.

15. Ould, M. A. (1995). Business Processes: Modeling and Analysis for Re-engineering and Improvement. Chichester: John Wiley & Sons.

16. Paulk, M. C., Curtis, B., Chrissis, M. B. and Weber, C. V. (1993). 'Capability Maturity Model, Version 1.1'. IEEE Software, 10 (4), 18–27.

17. Paulk, M. C. and Konrad, M. (1994). 'An Overview of ISO's SPICE Project'. IEEE Computer, 27 (4), 68–70.

18. Paulk, M. C., Weber, C. V., Curtis, B. and Chrissis, M. B. (1995). The Capability Maturity Model: Guidelines for Improving the Software Process. Reading, Mass.: Addison-Wesley.

19. Pulford, K., Kuntzmann-Combelles, A. and Shirlaw, S. (1996). A Quantitative Approach to Software Management. Wokingham: Addison-Wesley.

20. Sommerville, I. and Sawyer, P. (1997). Requirements Engineering: A Good Practice Guide. Chichester: John Wiley & Sons.

21. White, S. A. 2004. 'An Introduction to BPMN'. http://www.bpmn.org/Documents/ Introduction%20to%20BPMN.

MUNDARIJA

“Dasturiy loyihalarni boshqarish” fani bo‘yicha o‘quv qo‘llanma.

ATDT kafedrasi majlisida ko‘rib
chiqildi va nashr etishga ruxsat etildi,
2019 yil 9 yanvar
10 - sonli bayonnoma

DI fakulteti IUKmajlisida
ko‘rib chiqildi va nashr etishga ruxsat etildi,
2019 yil 29 yanvar
6 - sonli bayonnoma

TATU ilmiy-uslubiy Kengashi majlisida
ko‘rib chiqildi va nashr etishga ruxsat etildi,
2019 yil 26 mart
9(121)- sonli bayonnoma

Tuzuvchi(lar): Boboev L.B., Abduraxmanova N.N.

Mas’ul muharrirlar: Arzikulov S.D.
Shaazizova M.E.