O‘ZBEKISTON RESPUBLIKASI
OLIY VA O‘RTA MAXSUS TA’LIM VAZIRLIGI

OLIY TA’LIM TIZIMI PEDAGOG VA RAHBAR KADRLARINI QAYTA TAYYORLASH VA ULARNING MALAKASINI OSHIRISHNI TASHKIL ETISH BOSH ILMIY - METODIK MARKAZI

MUXAMMAD AL XORAZMIY NOMIDAGI TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI HUZURIDAGI PEDAGOG KADRLARNI QAYTA TAYYORLASH VA ULARNING MALAKASINI OSHIRISH TARMOQ MARKAZI

“AXBOROT TIZIMLARINING MATEMATIK VA DASTURIY TA'MINOTI” yo‘nalishi

“DASTURIY INJINIRING”

MODULI BO‘YICHA

O‘ Q U V –U S L U B I Y M A J M U A

TOSHKENT - 2018

O’ZBEKISTON RESPUBLIKASI
OLIY VA O’RTA MAXSUS TA’LIM VAZIRLIGI

OLIY TA’LIM TIZIMI PEDAGOG VA RAHBAR KADRLARINI QAYTA TAYYORLASH VA ULARNING MALAKASINI OSHIRISHNI TASHKIL ETISH BOSH ILMIY-METODIK MARKAZI

MUXAMMAD AL XORAZMIY NOMIDAGI TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI HUZURIDAGI PEDAGOG KADRLARNI QAYTA TAYYORLASH VA ULARNING MALAKASINI OSHIRISH TARMOQ MARKAZI

“DASTURIY INJINIRING” moduli bo'yicha

O’QUV-USLUBIY MAJMUA

TOSHKENT- 2018

Mazkur o`quv-uslubiy majmua Oliy va o`rta maxsus ta`lim vazirligining 20_ yil _____dagi ___-sonli buyrug`i bilan tasdiqlangan o`quv reja va dastur asosida tayyorlandi.

Tuzuvchi: TATU, “Axborot texnologiyalarining dasturiy ta`minoti” kafedrasi assistenti L.B. Boboyev

Taqrizchi: TATU, AKT bo`yicha maslahatchi prorektori, Janubiy Koreyalik mutaxassis ..... Li Chul Su

O`quv-uslubiy majmua Muxammad al Xorazmiy nomidagiToshkent axborot texnologiyalari universiteti Kengashining qarori bilan nashrga tavsiya qilingan (20__ yil _____dagi _____-sonli bayonnoma)

MUNDARIJA

I. BO`LIM

ISHCHI DASTUR

I. ISHCHI DASTUR

Kirish

Barchamizga sir emaski, xozirgi vaqtda jamiyatni axborot kommunikasiya texnologiyalarsiz tasavvur qilib bo’lmaydi. Umuman olganda, har bir mamlakatni rivojlanishini zamonaviy axborot-kommunikasiya texnologiyalarini sohalarga joriy qilinishi bilan baholash mumkin. Mamlakatimizda xam buning uchun bir qator ishlar amalga oshirilib kelinmoqda. Xususan, xukumatimiz tomonidan zamonaviy axborot-kommunikasiya texnologiyalarini joriy etish va rivojlantirish (21.03.2012 yildagi PQ-1730), axborot-kommunikasiya texnologiyalari sohasida kadrlar tayyorlash tizimini yanada takomillashtirish (26.03.2013 yildagi PQ-1942), dasturiy ta’minot vositalari ishlab chiquvchilarni rag’batlantirishni yanada kuchaytirish (20.09.2013 yildagi PQ-2042) bo’yicha chora-tadbirlarni keltirib o’tish mumkin. Bundan tashqari, “O’zbekiston Respublikasi Axborot Texnologiyalari va Kommunikasiyalarini Rivojlantirish Vazirligini tashkil etish to’g’risida”gi (04.02.2015 yil PF-4702) O’zbekiston Respublikasining Prezidentining Farmonida yangi tashkil etilgan vazirlikning asosiy vazifalaridan biri qilib “raqobatdosh dasturiy mahsulotlarning mamlakatimizda ishlab chiqarilishini va ichki bozorini hamda ularga ko’rsatiladigan xizmatlarni rivojlantirishga ko’maklashish va uning muvofiqlashtirilishini ta’minlash, iqtisodiyotning real sektori tarmoqlarida va iste’molchilarda zamonaviy dasturiy mahsulotlar, axborot tizimlari va axborot resurslarini joriy etish” etib belgilangan.

Ma’lumki, xozirgi kunda dasturiy ta’minot yoki dasturiy mahsulotni yaratish dasturiy ta’minot ishlab chiquvchi mutaxassislar jamoasi orqali amalga oshiriladi. Umuman olganda, xozirda, har bir dasturiy ta’minot yoki dasturiy mahsulotni ishlab chiqishga amaliy loyiha sifatida qarash mumkin. Chunki, har bir ishlab chiqilgan dasturiy ta’minot biror bir sohadagi (ijtimoiy, iqtisodiy, sanoat, xalq xo’jaligi va boshqalar) ma’lum bir muammolarni hal qilishga yo’naltirilgan bo’ladi.

Dasturiy ta’minotni ishlab chiqishga tizimli va kompleks yondashuv dasturiy

injiniring (Software Engineering) tushunchasini keltirib chiqardi. Kelinglar, dasturiy injiniring tushunchasini vujudga kelishi sabablariga to’xtalib o’taylik. Dasturiy ta’minotni ishlab chiqish faoliyatini rivojlanishini dasturiy ta’minot bilan bog’liq bo’lgan bir qator muammolarni vujudga kelishi va ularni hal qilish uchun amalga oshirilgan ishlar orqali izohlash mumkin. Dasturiy ta’minotni ishlab chiqish bo’yicha vujudga kelgan asosiy muammolar quyidagilardan iborat bo’lgan:

- dasturiy ta’minot tan-narxining juda yuqoriligi;
- talab etilayotgan dasturiy ta’minotni yaratishning murakkabligi;
- dasturiy ta’minotni ishlab chiqish jarayonlarini boshqarish va bashoratlash bo’yicha zarurat paydo bo’lishi va h.k.

Dasturiy injiniring – bu dasturiy ta’minot ishlab chiqishni samaradorligini oshirish va optimallashtirish bilan shug’ullanuvchi amaliy fan bo’lib, u dasturiy ta’minotni yaratishni loyihalashtirish(tahlil qilish), ishlab chiqish, joriy qilish va kuzatishni ilmiy asoslangan usullarini o’z ichida mujassamlashtirgan.

Ushbu o’quv uslubiy majmuada dasturiy injiniring tushunchalari, dasturiy ta’minot yashash sikli, dasturiy ta’minot ishlab chiqishning modellari va texnologiyalari, dasturiy ta’minot arxitekturasi, dasturiy ta’minotni testlash va tekshirish yo’llari, dasturiy ta’minotni ishlab chiqishdagi extimolli holatlar va ularni bartaraf etish yo’llari va usullari, dasturiy ta’minot sifati, dasturiy ta’minotni ishlab chiqish va testlash bo’yicha standartlar muammolari bayon etilgan.

Modulning maqsadi va vazifalari

Dasturiy injiniring **modulining maqsad va vazifalari:**

- dasturiy injiniring tushunchalari, dasturiy ta’minot yashash sikli, dasturiy ta’minot ishlab chiqishning modellari va texnologiyalari, dasturiy ta’minot arxitekturasi, dasturiy ta’minotni testlash va tekshirish yo’llari, dasturiy ta’minotni ishlab chiqishdagi extimolli holatlar va ularni bartaraf etish yo’llari va usullari, dasturiy ta’minot sifati, dasturiy ta’minotni ishlab chiqish va testlash;

Modul bo’yicha tinglovchilarning bilimi, ko’nikmasi, malakasi va kompetensiyalariga qo’yiladigan talablar

“Dasturiy injiniring” kursini o’zlashtirish jarayonida amalga oshiriladigan masalalar doirasida**:**

Tinglovchi:

- dasturiy ta’minotni yaratishda talablar va texnik topshiriq ishlab chiqishning qonuniy asoslari;
- dasturiy ta’minotni yaratish modellari;
- dasturlash tamoyillari va texnologiyalari bo’yicha nazariy va amaliy **bilimlarga ega bo’lishi**;

Tinglovchi:

- loyihani ishlab chiqishda obyektlar va ular bilan ishlash;
- dasturiy ta’minot arxitekturasi, talablar va ularni boshqarish;
- dasturiy ta’minot uchun holat diagrammalarini shakllantirish;
- loyiha dizayni va dasturiy ta’minotini ishlab chiqish;
- loyihani testlash;
- loyihada resurslarni taqsimlash va boshqarish **ko’nikma va malakalarini egallashi;**

Tinglovchi:

- loyiha sifatini boshqarish;
- loyihani yaratish uchun ketadigan sarf-harajatlarni hisoblash va oldindan bashorat qilish **kompetensiyalarni egallashi lozim.**

Modulni tashkil etish va o’tkazish bo’yicha tavsiyalar

“Dasturiy injiniring” kursi ma’ruza va amaliy mashg’ulotlar shaklida olib boriladi.

Kursni o’qitish jarayonida ta’limning zamonaviy metodlari, pedagogik texnologiyalar va axborot-kommunikasiya texnologiyalari qo’llanilishi nazarda tutilgan:

-ma’ruza darslarida zamonaviy kompyuter texnologiyalari yordamida prezentasion va elektron-didaktik texnologiyalardan;

-o’tkaziladigan amaliy mashg’ulotlarda texnik vositalardan, ekspress-so’rovlar, test so’rovlari, aqliy hujum, guruhli fikrlash, kichik guruhlar bilan ishlash, kollokvium o’tkazish, va boshqa interaktiv ta’lim usullarini qo’llash nazarda tutiladi.

Modulning o’quv rejadagi boshqa modullar bilan bog’liqligi va uzviyligi

“Dasturiy injiniring” moduli mazmuni o’quv rejadagi “Mobil ilovalar

yaratish”, “Operasion tizimlar” va “Veb dasturlash” o’quv modullari bilan uzviy bog’langan holda pedagoglarning loyihalarni ishlab chiqish bo’yicha kasbiy pedagogik tayyorgarligi va amaliy jihatdan bilim darajasini oshirishga xizmat qiladi.

Modulning oliy ta’limdagi o’rni

Modulni o’zlashtirish orqali tinglovchilar loyihalarni ishlab chiqishda talablarni shakllantirish, loyiha yaratish modellari va dizayni, loyihani testlash va tekshirish, amalda qo’llash va baholashga doir kasbiy kompetentlikka ega bo’ladilar.

Modul bo’yicha soatlar taqsimoti

№	Modul mavzulari	Tinglovchining o’quv yuklamasi, soat				
		Hammasi	Auditoriya o’quv yuklamasi			Mustaqil ta`lim
			Jami	jumladan		
				Nazariy	Amaliy mashg`ulot	
1.	Dasturiy injiniring tushunchasi. Dasturiy ta’minot yaratish jarayoni. Dasturiy ta’mi-not yaratish jarayoning klassik modellari.	6	6	2	4	
2.	Talablarni ishlab chiqish va model-lashtirish. UML holat diagrammalarini shakllantirish. Dasturiy ta’minotning arxitekturaviy dizayni. Dasturiy ta’minotni boshqarish.	8	6	4	4	
3.	Dasturiy ta’minot dizaynini qurish va moslashtirish. Dasturiy ta’minotni testlash. Dastriy ta’minot evalyusiyasi. foydalanuvchi interfeysi	6	4	2	2	2
4.	Dasturiy ta’minot xavfsizligi va ishonch-liligining xususiyatlari. Dasturiy ta’minot xavfsizligining ehtimolligini boshqarish.	6	6	2	4	2
	Jami:	**30**	**24**	**10**	**14**	**4**

NAZARIY MASHG’ULOTLAR MAZMUNI

1 - mavzu: Dasturiy injiniring tushunchasi. Dasturiy ta’minot yaratish jarayoni. Dasturiy ta’minot yaratish jarayoning klassik modellari.

Dasturiy injiniringa kirish. Dasturiy ta’minot jarayonlari va agile metodlari. Dasturiy ta’minotni ishlab chiqish jarayoni.

2 - mavzu: Talablarni ishlab chiqish va modellashtirish. UML holat diagrammalarini shakllantirish. Dasturiy ta’minotning arxitekturaviy dizayni. Dasturiy ta’minotni boshqarish.

Dasturiy ta’minot arxitekturasi va arxitekturaviy dizayn. Dasturiy ta’minot dizaynini yaratish va uni tizimga moslashtirish. Daturiy ta’minotni testlash. Dasturiy ta’minot evolyutsiyalarini ko’rib chiqish.

3 - mavzu: Dasturiy ta’minot dizaynini qurish va moslashtirish. Dasturiy ta’minotni testlash. Dastriy ta’minot evalyusiyasi. foydalanuvchi interfeysi

Ishlab chiqilgan dasturiy ta’minotning foydalanish yuzasidan ishonchliligini tekshirish. Ishchi stansiyalaridagi ma’lumotlarning dastur ta’sirida tarqalishini oldini olish. Tizim xavfsizligini o'rnat’sh va boshqarish.

4 - mavzu: Dasturiy ta’minot xavfsizligi va ishonchliligining xususiyatlari. Dasturiy ta’minot xavfsizligining ehtimolligini boshqarish.

Dasturiy ta’minotni boshqarishning asosiy turlari. Dasturiy ta’minotga bo’lgan xavfni boshqarish. Tizim foydalanuvchilarini boshqarish turlarini o’rganish.

AMALIY MASHG’ULOT MAZMUNI

1-amaliy mashg’ulot
Dasturiy injiniring tushunchasi. Dasturiy ta’minot yaratish jarayoni. Dasturiy ta’minot yaratish jarayoning klassik modellari.

Loyihaning holat diagrammalarini shakllantirish uchun UML muhitida ishlash uchun StarUML dasturiy vositasini o’rnatish, loyiha tasnifi uchun use case, class, sequence, activity va boshqa holat diagrammalarini yaratish ko’nikmalarini hosil qilish.

2-amaliy mashg’ulot
Talablarni ishlab chiqish va modellashtirish. UML holat diagrammalarini shakllantirish. Dasturiy ta’minotning arxitekturaviy dizayni. Dasturiy ta’minotni boshqarish.

Ishlab chiqarilayotgan dasturiy ta’minot uchun tizim talablari va funksional

talablarni ishlab chiqish va uning hujjatini shakllantirish.

3-amaliy mashg’ulot
Talablarni ishlab chiqish va modellashtirish. UML holat diagrammalarini shakllantirish. Dasturiy ta’minotning arxitekturaviy dizayni. Dasturiy ta’minotni boshqarish.

Berilgan loyiha uchun qo’yilgan tizim va funksional talablardan kelib chiqqan holda use case, class, sequence va boshqa mos holat diagrammalarini shakllantirish ko’nikmalarini hosil qilish.

4-amaliy mashg’ulot
Talablarni ishlab chiqish va modellashtirish. UML holat diagrammalarini shakllantirish. Dasturiy ta’minotning arxitekturaviy dizayni. Dasturiy ta’minotni boshqarish.

Dasturiy ta’minot arxitekturasi va arxitekturaviy dizayn yaratish va uni tizimga moslashtirish ko’nikmalarini hosil qilish.

5 – amaliy mashg’ulot
Dasturiy ta’minot dizaynini qurish va moslashtirish. Dasturiy ta’minotni testlash. Dastriy ta’minot evalyusiyasi. foydalanuvchi interfeysi.

Dasturiy ta’minot uchun dizayn na’munalar ishlab chiqish va uning amaliy ko’rinishini yaratish, o’z navbatida tizim uchun qo’yilgan talablar asosida dasturiy ta’minot ishlab chiqish usullar va texnologiyalarini aniqlash, dastur yaratish.

6 – amaliy mashg’ulot
Dasturiy ta’minot xavfsizligi va ishonchliligining xususiyatlari. Dasturiy ta’minot xavfsizligining ehtimolligini boshqarish.

Yaratilgan dasturiy ta’minotni testlash va testlash bosqichlarini o’rganish. Dasturiy ta’minotni testlash shartlarini shakllantirish

O’QITISH SHAKLLARI

Mazkur modul bo’yicha quyidagi o’qitish shakllaridan foydalaniladi:

- ma’ruzalar, amaliy mashg’ulotlar (ma’lumotlar va texnologiyalarni anglab olish, aqliy qiziqishni rivojlantirish, nazariy bilimlarni mustahkamlash);
- davra suhbatlari (ko’rilayotgan loyiha yechimlari bo’yicha taklif berish qobiliyatini oshirish, eshitish, idrok qilish va mantiqiy xulosalar chiqarish);
- bahs va munozaralar (loyihalar yechimi bo’yicha dalillar va asosli argumentlarni taqdim qilish, eshitish va muammolar yechimini topish qobiliyatini rivojlantirish).

BAHOLASH MEZONI

<table>
<tr><th>№</th><th>Baholash turlari</th><th>Maksimal ball</th><th>Ball</th></tr>
<tr><td rowspan="3">1.</td><td>Keys topshiriqlari</td><td rowspan="3">2.5</td><td>1.2 ball</td></tr>
<tr><td rowspan="2">Mustaqil ish topshiriqlari</td><td>0.5 ball</td></tr>
<tr><td rowspan="2">0.8 ball</td></tr>
<tr><td>Amaliy topshiriqlar</td></tr>
</table>

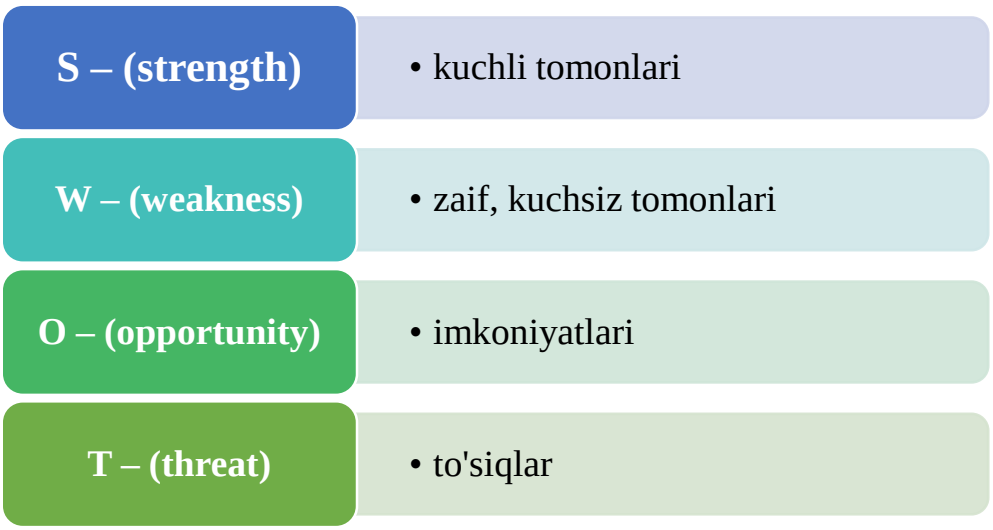
II. BO`LIM

MODULNI O`QITISHDA FOYDALANILADIGAN INTERFAOL TA`LIM METODLARI

II. MODULNI O’QITISHDA FOYDALANILADIGAN INTERFAOL TA’LIM METODLARI.

“SWOT-tahlil” metodi.

Metodning maqsadi: mavjud nazariy bilimlar va amaliy tajribalarni tahlil qilish, taqqoslash orqali muammoni hal etish yo’llarni topishga, bilimlarni mustahkamlash, takrorlash, baholashga, mustaqil, tanqidiy fikrlashni, nostandart tafakkurni shakllantirishga xizmat qiladi.



Namuna: Dasturiy ta’minotning MVC arxitekturasi SWOT tahlilini ushbu jadvalga tushiring.

S	Dasturiy ta’minotning MVC arxitekturasidan foydalanishning kuchli tomonlari	Veb ilovalarini yaratishda juda qulay
W	Dasturiy ta’minotning MVC arxitekturasidan foydalanishning kuchsiz tomonlari	Ko’p kodlardan tashkil topadi
O	Dasturiy ta’minotning MVC arxitekturasidan foydalanishning imkoniyatlari (ichki)	Ma’lumotlarni o’zgartirishning mustaqilligi
T	To’siqlar (tashqi)	Ma’lumotlar xavfsizligining to’laqonli ta’minlanmaganligi...

Xulosalash metodi

Metodning maqsadi: Bu metod murakkab, ko’ptarmoqli, mumkin qadar, muammoli xarakteridagi mavzularni o’rganishga qaratilgan. Metodning mohiyati shundan iboratki, bunda mavzuning turli tarmoqlari bo’yicha bir xil axborot beriladi va ayni paytda, ularning har biri alohida aspektlarda muhokama etiladi. Masalan, muammo ijobiy va salbiy tomonlari, afzallik, fazilat va kamchiliklari, foyda va zararlari bo’yicha o’rganiladi. Bu interfaol metod tanqidiy, tahliliy, aniq mantiqiy fikrlashni muvaffaqiyatli rivojlantirishga hamda o’quvchilarning mustaqil g’oyalari, fikrlarini yozma va og’zaki shaklda tizimli bayon etish, himoya qilishga imkoniyat yaratadi. “Xulosalash” metodidan ma’ruza mashg’ulotlarida individual va juftliklardagi ish shaklida, amaliy va seminar mashg’ulotlarida kichik guruhlardagi ish shaklida mavzu yuzasidan bilimlarni mustahkamlash, tahlili qilish va taqqoslash maqsadida foydalanish mumkin.

Metodni amalga oshirish tartibi:

trener-o’qituvchi ishtirokchilarni 5-6 kishildan iborat kichik guruhlarga ajratadi;

trening maqsadi, shartlari va tartibi bilan ishtirokchilarni tanishtirgach, har bir guruhga umumiy muammoni tahlil qilinishi zarur bo'lgan qismlari tushirilgan tarqatma materiallarni tarqatadi;

har bir guruh o'ziga berilgan muammoni atroflicha tahlil qilib, o'z mulohazalarini tavsiya etilayotgan sxema bo'yicha tarqatmaga yozma bayon qiladi;

navbatdagi bosqichda barcha guruhlar o'z taqdimotlarini o'tkazadilar. shundan so'ng, trener tomonidan tahlillar umumlashtiriladi, zaruriy axborotlar bilan to'ldiriladi va mavzu yakunlanadi

Namuna:

Dasturiy ta’minot arxitekturasi					
MVC		**Layered**		**Repository**	
afzalligi	kamchiligi	afzalligi	kamchiligi	afzalligi	kamchiligi
Xulosa:					

“Keys-stadi” metodi

«Keys-stadi» - inglizcha so’z bo’lib, («case» – aniq vaziyat, hodisa, «stadi» – o’rganmoq, tahlil qilmoq) aniq vaziyatlarni o’rganish, tahlil qilish asosida o’qitishni amalga oshirishga qaratilgan metod hisoblanadi. Mazkur metod dastlab 1921 yil Garvard universitetida amaliy vaziyatlardan iqtisodiy boshqaruv fanlarini o’rganishda foydalanish tartibida qo’llanilgan. Keysda ochiq axborotlardan yoki aniq voqyea-hodisadan vaziyat sifatida tahlil uchun foydalanish mumkin. Keys harakatlari o’z ichiga quyidagilarni qamrab oladi: Kim (Who), Qachon (When), Qayerda (Where), Nima uchun (Why), Qanday/ Qanaqa (How), Nima-natija (What).

“Keys metodi” ni amalga oshirish bosqichlari

Ish bosqichlari	**Faoliyat shakli va mazmuni**
1-bosqich: Keys va uning axborot ta’minoti bilan tanishtirish	✓ yakka tartibdagi audio-vizual ish; ✓ keys bilan tanishish(matnli, audio yoki media shaklda); ✓ axborotni umumlashtirish; ✓ axborot tahlili; ✓ muammolarni aniqlash
2-bosqich: Keysni aniqlashtirish va o’quv topshirig’ni belgilash	✓ individual va guruhda ishlash; ✓ muammolarni dolzarblik iyerarxiyasini aniqlash; ✓ asosiy muammoli vaziyatni belgilash
3-bosqich: Keysdagi asosiy muammoni tahlil etish orqali o’quv topshirig’ining yechimini izlash, hal etish yo’llarini ishlab chiqish	✓ individual va guruhda ishlash; ✓ muqobil yechim yo’llarini ishlab chiqish; ✓ har bir yechimning imkoniyatlari va to’siqlarni tahlil qilish; ✓ muqobil yechimlarni tanlash
4-bosqich: Keys yechimini yechimini shakllantirish va asoslash, taqdimot.	✓ yakka va guruhda ishlash; ✓ muqobil variantlarni amalda qo’llash imkoniyatlarini asoslash; ✓ ijodiy-loyiha taqdimotini tayyorlash; ✓ yakuniy xulosa va vaziyat yechimining amaliy aspektlarini yoritish

Keys. Ishlab chiqilgan dasturiy ta’minot windows 10 operatsion tizimi uchun mo’ljallangan, ammo mijoz kompyuterida windows 8.1 operatsion tizimi o’rnatilgan. Dasturni o’rnatishda muammo yuzaga keldi . Nima qilish kerak?

Keysni bajarish bosqchilari va topshiriqlar:

- Keysdagi muammoni keltirib chiqargan asosiy sabablarni belgilang(individual va kichik guruhda).
- Dasturni ishga tushirish uchun tizim talablari ketma-ketligini belgilang (juftliklardagi ish).

«FSMU» metodi

Texnologiyaning maqsadi: Mazkur texnologiya ishtirokchilardagi umumiy fikrlardan xususiy xulosalar chiqarish, taqqoslash, qiyoslash orqali axborotni o'zlashtirish, xulosalash, shuningdek, mustaqil ijodiy fikrlash ko'nikmalarini shakllantirishga xizmat qiladi. Mazkur texnologiyadan ma'ruza mashg'ulotlarida, mustahkamlashda, o'tilgan mavzuni so'rashda, uyga vazifa berishda hamda amaliy mashg'ulot natijalarini tahlil etishda foydalanish tavsiya etiladi.

Texnologiyani amalga oshirish tartibi:

- qatnashchilarga mavzuga oid bo'lgan yakuniy xulosa yoki g'oya taklif etiladi;
- har bir ishtirokchiga FSMU texnologiyasining bosqichlari yozilgan qog'ozlarni tarqatiladi:

F	• fikringizni bayon eting
S	• fikringiz bayoniga sabab ko'rsating
M	• ko'rsatgan sabablaringizni isbotlab bering
U	• fikringizni umumlashtiring

- ishtirokchilarning munosabatlari individual yoki guruhiy tartibda taqdimot qilinadi.

FSMU tahlili qatnashchilarda kasbiy-nazariy bilimlarni amaliy mashqlar va mavjud tajribalar asosida tezroq va muvaffaqiyatli o'zlashtirilishiga asos bo'ladi.

Namuna.

Fikr: "**Tezkor dasturiy ta'minot biznes ilovalar yaratishda muhimdir".**

Topshiriq: Mazkur fikrga nisbatan munosabatingizni FSMU orqali tahlil qiling.

“Assesment” metodi

Metodning maqsadi: mazkur metod ta’lim oluvchilarning bilim darajasini baholash, nazorat qilish, o’zlashtirish ko’rsatkichi va amaliy ko’nikmalarini tekshirishga yo’naltirilgan. Mazkur texnika orqali ta’lim oluvchilarning bilish faoliyati turli yo’nalishlar (test, amaliy ko’nikmalar, muammoli vaziyatlar mashqi, qiyosiy tahlil, simptomlarni aniqlash) bo’yicha tashhis qilinadi va baholanadi.

Metodni amalga oshirish tartibi:

“Assesment” lardan ma’ruza mashg’ulotlarida talabalarning yoki qatnashchilarning mavjud bilim darajasini o’rganishda, yangi ma’lumotlarni bayon qilishda, seminar, amaliy mashg’ulotlarda esa mavzu yoki ma’lumotlarni o’zlashtirish darajasini baholash, shuningdek, o’z-o’zini baholash maqsadida individual shaklda foydalanish tavsiya etiladi. Shuningdek, o’qituvchining ijodiy yondashuvi hamda o’quv maqsadlaridan kelib chiqib, assesmentga qo’shimcha topshiriqlarni kiritish mumkin.

Namuna. Har bir katakdagi to’g’ri javob 5 ball yoki 1-5 balgacha baholanishi mumkin.

Test

- 1. UML kengaytmasi nima?
- A) Unit Modeling Language
- B) Union Modeling Language
- C) Unified Modeling Language

Qiyosiy tahlil

- Dasturiy ta’minot arxitekturalarini tahlil qiling

Tushuncha tahlili

- IDE qisqartmasini izohlang…

Amaliy ko’nikma

- UML tili muhitini o’rnating

"Insert" metodi

Metodning maqsadi: Mazkur metod o'quvchilarda yangi axborotlar tizimini qabul qilish va bilmlarni o'zlashtirilishini yengillashtirish maqsadida qo'llaniladi, shuningdek, bu metod o'quvchilar uchun xotira mashqi vazifasini ham o'taydi.

Metodni amalga oshirish tartibi:

- o'qituvchi mashg'ulotga qadar mavzuning asosiy tushunchalari mazmuni yoritilgan input-matnni tarqatma yoki taqdimot ko'rinishida tayyorlaydi;
- yangi mavzu mohiyatini yorituvchi matn ta'lim oluvchilarga tarqatiladi yoki taqdimot ko'rinishida namoyish etiladi;
- ta'lim oluvchilar individual tarzda matn bilan tanishib chiqib, o'z shaxsiy qarashlarini maxsus belgilar orqali ifodalaydilar. Matn bilan ishlashda talabalar yoki qatnashchilarga quyidagi maxsus belgilardan foydalanish tavsiya etiladi:

Belgilar	1-matn	2-matn	3-matn
"V" – tanish ma'lumot.			
"?" – mazkur ma'lumotni tushunmadim, izoh kerak.			
"+" bu ma'lumot men uchun yangilik.			
"– " bu fikr yoki mazkur ma'lumotga qarshiman?			

Belgilangan vaqt yakunlangach, ta'lim oluvchilar uchun notanish va tushunarsiz bo'lgan ma'lumotlar o'qituvchi tomonidan tahlil qilinib, izohlanadi, ularning mohiyati to'liq yoritiladi. Savollarga javob beriladi va mashg'ulot yakunlanadi.

"Tushunchalar tahlili" metodi

Metodning maqsadi: mazkur metod talabalar yoki qatnashchilarni mavzu buyicha tayanch tushunchalarni o'zlashtirish darajasini aniqlash, o'z bilimlarini mustaqil ravishda tekshirish, baholash, shuningdek, yangi mavzu buyicha dastlabki bilimlar darajasini tashhis qilish maqsadida qo'llaniladi.

Metodni amalga oshirish tartibi:

- ishtirokchilar mashg'ulot qoidalari bilan tanishtiriladi;
- o'quvchilarga mavzuga yoki bobga tegishli bo'lgan so'zlar, tushunchalar nomi tushirilgan tarqatmalar beriladi (individual yoki guruhli tartibda);

- o'quvchilar mazkur tushunchalar qanday ma'no anglatishi, qachon, qanday holatlarda qo'llanilishi haqida yozma ma'lumot beradilar;
- belgilangan vaqt yakuniga yetgach o'qituvchi berilgan tushunchalarning tugri va tuliq izohini uqib eshittiradi yoki slayd orqali namoyish etadi;
- har bir ishtirokchi berilgan tugri javoblar bilan uzining shaxsiy munosabatini taqqoslaydi, farqlarini aniqlaydi va o'z bilim darajasini tekshirib, baholaydi.

Namuna: "Moduldagi tayanch tushunchalar tahlili"

Tushunchalar	**Sizningcha bu tushuncha qanday ma'noni anglatadi?**	**Qo'shimcha ma'lumot**
Pattern	Andoza	
Requirement	Talab	
Software	Dasturiy ta'minot	
Process	Jarayon	
Activity	Faoliyat	
Dependability	Ishonchlilik	
Agile	Tezkor	

Izoh: Ikkinchi ustunchaga qatnashchilar tomonidan fikr bildiriladi. Mazkur tushunchalar haqida qo'shimcha ma'lumot glossariyda keltirilgan.

Venn Diagrammasi metodi

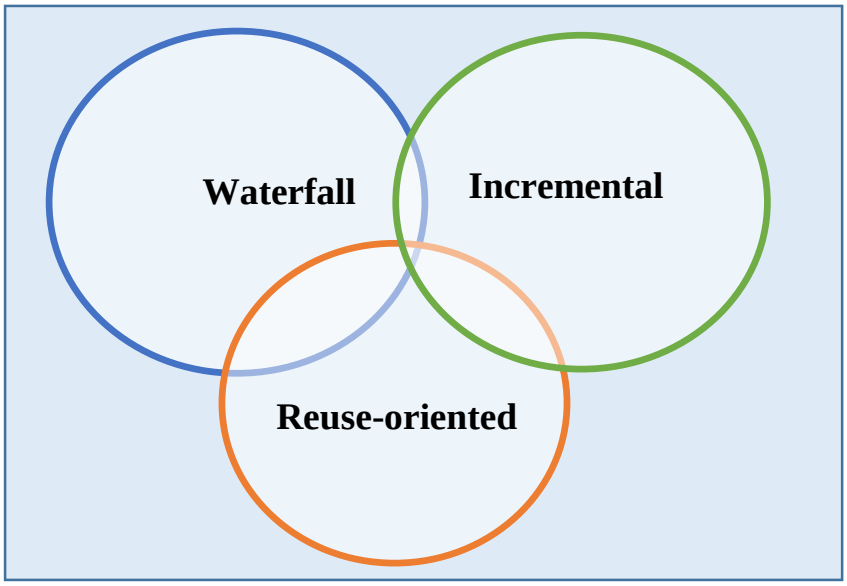
Metodning maqsadi: Bu metod grafik tasvir orqali o'qitishni tashkil etish shakli bo'lib, u ikkita o'zaro kesishgan aylana tasviri orqali ifodalanadi. Mazkur metod turli tushunchalar, asoslar, tasavurlarning analiz va sintezini ikki aspekt orqali ko'rib chiqish, ularning umumiy va farqlovchi jihatlarini aniqlash, taqqoslash imkonini beradi.

Metodni amalga oshirish tartibi:

- ishtirokchilar ikki kishidan iborat juftliklarga birlashtiriladilar va ularga ko'rib chiqilayotgan tushuncha yoki asosning o'ziga xos, farqli jihatlarini (yoki aksi) doiralar ichiga yozib chiqish taklif etiladi;
- navbatdagi bosqichda ishtirokchilar to'rt kishidan iborat kichik guruhlarga birlashtiriladi va har bir juftlik o'z tahlili bilan guruh a'zolarini tanishtiradilar;

- juftliklarning tahlili eshitilgach, ular birgalashib, ko’rib chiqilayotgan muammo yohud tushunchalarning umumiy jihatlarini (yoki farqli) izlab topadilar, umumlashtiradilar va doirachalarning kesishgan qismiga yozadilar.

Namuna: Dasturiy ta’minot ishlab chiqish modellari



“Blis-o’yin” metodi

Metodning maqsadi: o’quvchilarda tezlik, axborotlar tizmini tahlil qilish, rejalashtirish, prognozlash ko’nikmalarini shakllantirishdan iborat. Mazkur metodni baholash va mustahkamlash maksadida qo’llash samarali natijalarni beradi.

Metodni amalga oshirish bosqichlari:

1. Dastlab ishtirokchilarga belgilangan mavzu yuzasidan tayyorlangan topshiriq, ya’ni tarqatma materiallarni alohida-alohida beriladi va ulardan materialni sinchiklab o’rganish talab etiladi. Shundan so’ng, ishtirokchilarga to’g’ri javoblar tarqatmadagi «yakka baho» kolonkasiga belgilash kerakligi tushuntiriladi. Bu bosqichda vazifa yakka tartibda bajariladi.

2. Navbatdagi bosqichda trener-o’qituvchi ishtirokchilarga uch kishidan iborat kichik guruhlarga birlashtiradi va guruh a’zolarini o’z fikrlari bilan guruhdoshlarini tanishtirib, bahslashib, bir-biriga ta’sir o’tkazib, o’z fikrlariga ishontirish, kelishgan holda bir to’xtamga kelib, javoblarini «guruh bahosi» bo’limiga raqamlar bilan belgilab chiqishni topshiradi. Bu vazifa uchun 15 daqiqa vaqt beriladi.

3. Barcha kichik guruhlar o’z ishlarini tugatgach, to’g’ri harakatlar ketma-ketligi trener-o’qituvchi tomonidan o’qib eshittiriladi, va o’quvchilardan bu javoblarni «to’g’ri javob» bo’limiga yozish so’raladi.

4. «To’g’ri javob» bo’limida berilgan raqamlardan «yakka baho» bo’limida berilgan raqamlar taqqoslanib, farq bulsa «0», mos kelsa «1» ball quyish so’raladi.

Shundan so’ng «yakka xato» bo’limidagi farqlar yuqoridan pastga qarab qo’shib chiqilib, umumiy yig’indi hisoblanadi.

5. Xuddi shu tartibda «to’g’ri javob» va «guruh bahosi» o’rtasidagi farq chiqariladi va ballar «guruh xatosi» bo’limiga yozib, yuqoridan pastga qarab qo’shiladi va umumiy yig’indi keltirib chiqariladi.

6. Trener-o’qituvchi yakka va guruh xatolarini to’plangan umumiy yig’indi bo’yicha alohida-alohida sharhlab beradi.

7. Ishtirokchilarga olgan baholariga qarab, ularning mavzu bo’yicha o’zlashtirish darajalari aniqlanadi.

«Dasturiy ta’minotni ishlab chiqish jarayoni» ketma-ketligini joylashtiring. O’zingizni tekshirib ko’ring!

Harakatlar mazmuni	Yakka baho	Yakka xato	To’g’ri javob	Guruh bahosi	Guruh xatosi
Software design and implementation					
Software evolution					
Software validation					
Software specification					

“Brifing” metodi

“Brifing”- (ing. briefing-qisqa) biror-bir masala yoki savolning muhokamasiga bag’ishlangan qisqa press-konferensiya.

O’tkazish bosqichlari:

1. Taqdimot qismi.
2. Muhokama jarayoni (savol-javoblar asosida).

Brifinglardan trening yakunlarini tahlil qilishda foydalanish mumkin. Shuningdek, amaliy o’yinlarning bir shakli sifatida qatnashchilar bilan birga dolzarb mavzu yoki muammo muhokamasiga bag’ishlangan brifinglar tashkil etish mumkin bo’ladi. Talabalar yoki tinglovchilar tomonidan yaratilgan mobil ilovalarning taqdimotini o’tkazishda ham foydalanish mumkin.

“Portfolio” metodi

“Portfolio” – (ital. portfolio-portfel, ingl.hujjatlar uchun papka) ta’limiy va kasbiy faoliyat natijalarini autentik baholashga xizmat qiluvchi zamonaviy ta’lim texnologiyalaridan hisoblanadi. Portfolio mutaxassisning saralangan o’quv-metodik ishlari, kasbiy yutuqlari yig’indisi sifatida aks etadi. Jumladan, talaba yoki tinglovchilarning modul yuzasidan o’zlashtirish natijasini elektron portfoliolar orqali tekshirish mumkin bo’ladi. Oliy ta’lim muassasalarida portfolioning quyidagi

turlari mavjud:

Faoliyat turi	**Ish shakli**	
	Individual	**Guruhiy**
Ta’limiy faoliyat	Talabalar portfoliosi, bitiruvchi, doktorant, tinglovchi portfoliosi va boshq.	Talabalar guruhi, tinglovchilar guruhi portfoliosi va boshq.
Pedagogik faoliyat	O’qituvchi portfoliosi, rahbar xodim portfoliosi	Kafedra, fakultet, markaz, OTM portfoliosi va boshq.

III. BO`LIM

NAZARIY MATERIALLAR

III. NAZARIY MATERIALLAR

1 - ma’ruza. Dasturiy injiniring tushunchasi. Dasturiy ta’minot yaratish jarayoni. Dasturiy ta’mi-not yaratish jarayoning klassik modellari.

Reja:

1.1. Kirish. Dasturiy ta’minot jarayonlari
1.2. Sifatli va tezkor dasturiy ta’minot ishlab chiqish
1.3. Tizim uchun talablarni shakllantirish
1.4. Dasturiy ta’minotni modellashtirish

Kalit so’zlar: *jarayon, talab, faoliyat, tezkor, hujjat, model, dasturiy ta’minot, injiniring, mahsulot, faza, professional, metod.*

Ushbu ma’ruzaning maqsadi dasturiy injiniringa kirish, dasturiy ta’minot jarayonlari va agile metodlar kabi muhim tushunchalar va dasturiy ta’minotni ishlab chiqish jarayoni ahamiyatini tushuntirishdan iborat.

1.1. Kirish. Dasturiy ta’minot jarayonlari

[1]Biz zamonaviy dunyoni dasturiy ta’minotsiz tasavvur qila olmaymiz. Milliy infrastrukturalar va utilitalar kompyuterga asoslangan tizimlar tomonidan nazorat qilinadi va ko’pgina elektrli mahsulotlar o’z ichiga kompyuter va nazorat dasturlarni oladi. Sanoatda ishlab chiqarish va tarqatish to’liq kompyuterlashtirilgan. Musiqa sanoati, kompyuter o’yinlari, film va televizorlarda dasturiy ta’minotdan foydalanishadi. Shu sababli dasturiy injiniring milliy va xalqaro jamiyatlar funksionalligi uchun muhimdir. Dasturiy ta’minot tizimlari mavhum va nomoddiy bo’lib ular materiallarning xususiyatlari yoki fizik qonunlar bilan chegaralanmagan. Dasturiy ta’minotning potensialiga hech qanday tabiiy cheklanuvlar yo’q.

Dasturiy injiniring tarixi

“Dasturiy injiniring” atamasi 1968 - yil konfrensiyada taklif qilingan. 1970 - 1980 - yillarda strukturali dasturlash va obyektga yo’naltirilgan ishlab chiqish kabi dasturiy injiniring usullari va metodlari ishlab chiqildi. Standard atamalar ishlab chiqildi va hozirgi kunda keng qo’llanilmoqda.

Professional dasturiy ta’minotni ishlab chiqish

Ko’p odamlar dasturlar yozishadi. Odamlar ishda o’z ishlarini osonlashtirish uchun elektron jadval ko’rinishli dasturlar tuzushadi, tadqiqotchilar va injinerlar

[1] “Software Engineering”, by Ian Sommerville, pages 4-8.

ilmiy tajribaga oid ma’lumotlarga ishlov berish uchun dasturlar tuzishadi, yoki qiziqish sabab dasturlar tuzishadi. Professional dasturiy ta’minot bu dasturchidan tashqari boshqa insonlar ham foydalanishi maqsadida ishlab chiqiladi va odatda individual bo’lib emas guruh bo’lib ishlanadi.

Dasturiy injiniring professional dasturiy ta’minotni qo’llab quyyatlashga yo’naltirilgan bo’lib u dasturning o’ziga xos xususiyatlari, dizayni va evolutsiyasini ta’minlovchi usullarni o’z ichiga oladi. Dasturiy injiniring nimaligi haqida sizga kengroq tushuntirish maqsadida quyida ko’p so’ralgan savollarga qisqacha javoblarni havola etamiz (Jadval 1.1)

Savol	**Javob**
Dasturiy ta’minot nima?	Kompyuter dasturlari va unga bog’liq hujjatlar. Dasturiy mahsulotlar olohida mijozlar uchun yoki umumiy bozor uchun ishlab chiqarilishi mumkin.
Yaxshi dasturiy ta’minotning attributlari nima?	Yaxshi dasturiy ta’minot talab qilingan funksionallikka ega, foydalanuvchiga qulay, ishonchli va davomiy bo’lishi lozim.
Dasturiy injiniring nima?	Dasturiy injiniring bu dasturiy mahsulotni ishlab chiqishning barcha qirralari bilan bog’liq muhandislik.
Dasturiy injiniringni asosiy faoliyatlari nima?	Dastur xususiyatlari, dasturiy ta’minot ishlab chiqish, dasturiy ta’minotni tekshirish, va dasturiy ta'minot evolyutsiyasi.
Dasturiy injiniring va kompyuter ilmi o’rtasida qanday farq bor?	Kompyuter ilmi nazariya va tushunchalarga e’tiborni qaratadi; dasturiy injiniring amaliy ishlab chiqish va foydali dasturiy ta’minotni yetkazib berishga e’tibor qaratadi.
Dasturiy injiniring va tizim injiniring o’rtasida qanday farq bor?	Tizim injiniring bu apparat ta’minot, dasturiy ta’minot va jarayon injiniringni o’z ichiga oluvchi kompyuter asosli

	tizimlarni ishlab chiqish. Dasturiy injiniring ushbu umumiy jarayonning bir qismidir.
Dasturiy injiniringni narxi qancha?	Taxminan 60% xarajatlar ishlab chiqarish uchun va 40% xarajatlar testlash uchun sarf bo'ladi.
Eng yaxshi dasturiy injiniring usuli va metodi qaysi?	Barcha dasturiy ta'minot loyihalari professional tarzda boshqarilishi va ishlab chiqilishi kerak, turli xil tizimlar uchun mos bo'lgan turli xil usulllar qo'llaniladi.
Dasturiy injiniringa veb qanday xilma-xillik olib keldi?	Veb dasturiy servislarni va yuqori taqsimlangan servisga asoslangan tizimlarni ishlab chiqishni olib kirdi.
Jadval 1.1.	

Dasturiy injinerlar dasturiy mahsulotni ishlab chiqish bilan shug'ullanadilar. Ikki xil turdagi dasturiy mahsulotlar mavjud:

1. *Umumiy mahsulotlar* Bular autonom tizimlardir ya'ni ishlab chiqarish tashkiloti tomonidan ishlab chiqiladi va ochiq bozorda sotib ola biladigan mijozlarga sotiladi. Bu turdagi dasturiy mahsulotlarga shaxsiy kompyuterlar uchun dasturiy ta'minotlar masalan ma'lumot bazalari, matn tahrirlovchi, rasmlar chizish, loyihalarni boshqarish uskunalari kabi dasturlarni misol qilib olishimiz mumkin.
2. *Buyurtma mahsulotlar* Bular alohida mijozlar tomonidan buyurtma qilingan tizimlardir. Bu turdagi dasturiy mahsulotlarga elektronik qurilmalar uchun nazorat tizimlari, maxsus ish jarayonlariga yordam berish uchun yozilgan tizimlarni misol qilib olishimiz mumkin.

Bu ikki turdagi mahsulotlarning asosiy farqi shundaki, umumiy mahsulotlarda dasturiy ta'minot xususiyatlarini dasturiy ta'minotni ishlab chiqaruvchi tashkilot nazorat qiladi. Buyurtma mahsulotlarda mahsulotni sotib oluvchi tashkilot dasturiy ta'minot xususiyatlarini ishlab chiqadi va nazorat qiladi.

Biroq hozirgi kunga kelib bu ikki turdagi mahsulotlar orasidagi farq tobora kamayib boryapti, chunki ko'pgina tizimlar umumiy mahsulot sifatida qurilyapti va

mijozlar qaladiga qarab moslashtirilyapti. Enterprise Resource Planning (ERP) tizimlar, masalan SAP tizimi bunga yaqqol misol bo'lishi mumkin.

Dasturiy injiniring ba'zida dasturiy ta'minot jarayoni ham deyiladi. Dasturiy ta'minot jarayoni dasturiy mahsulotni ishlab chiqish faoliyatlari ketma-ketligidir. Barcha dasturiy ta'minotlar uchun umumiy bo'lgan to'rtta asosiy faoliyat bor. Bular:

1. Dasturiy ta'minot xususiyati
2. Dasturiy ta'minotni ishlab chiqish
3. Dasturiy ta'minotni tekshirish
4. Dasturiy ta'minot evolutsiyasi

Dasturiy ta'minot jarayoni

Dasturiy ta'minot bu dasturiy mahsulotni tayyor holatga olib kelish faoliyatilari to'plamidir. Bu faoliyatlar Java yoki C kabi standart dasturlash tillarida dasturlarni ishlab chiqishni o'z ichiga olishi mumkin.

Dasturiy ta'minot jarayoni modeli bu dasturiy ta'minot jarayonining soddalashtirilgan ko'rinishidir.

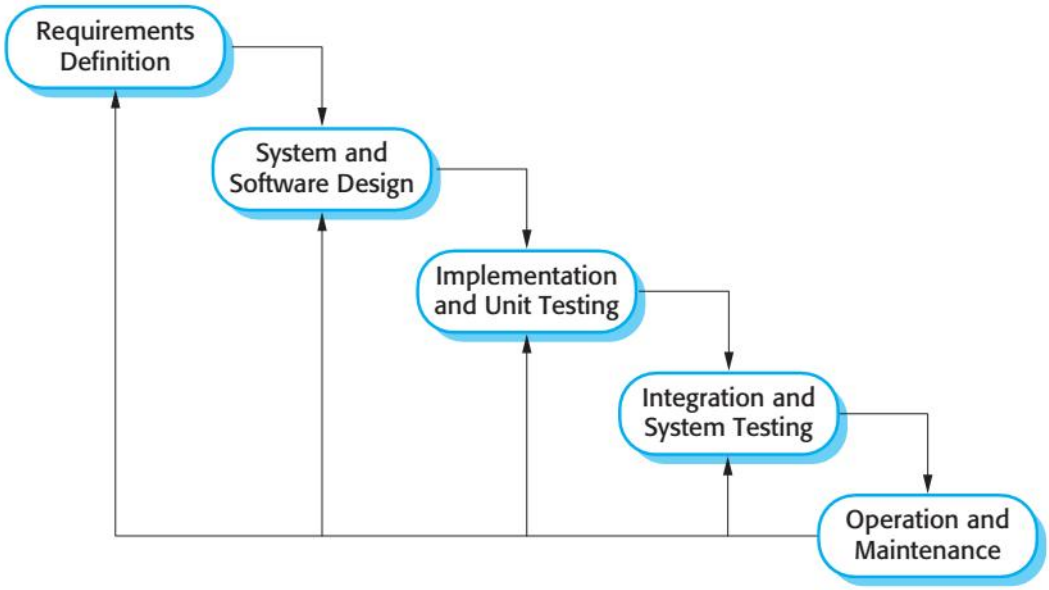
Jarayon modellari:

1. Sharshara modeli
2. Ortib borish modeli
3. Qaytib foydalanishga mo'ljallangan dasturiy injiniringi

Sharshara modeli

Sharshara modeli birinchi bo'lib e'lon qilingan dasturiy ta'minotni ishlab chiqish jarayoni modelidir (Royce, 1970).

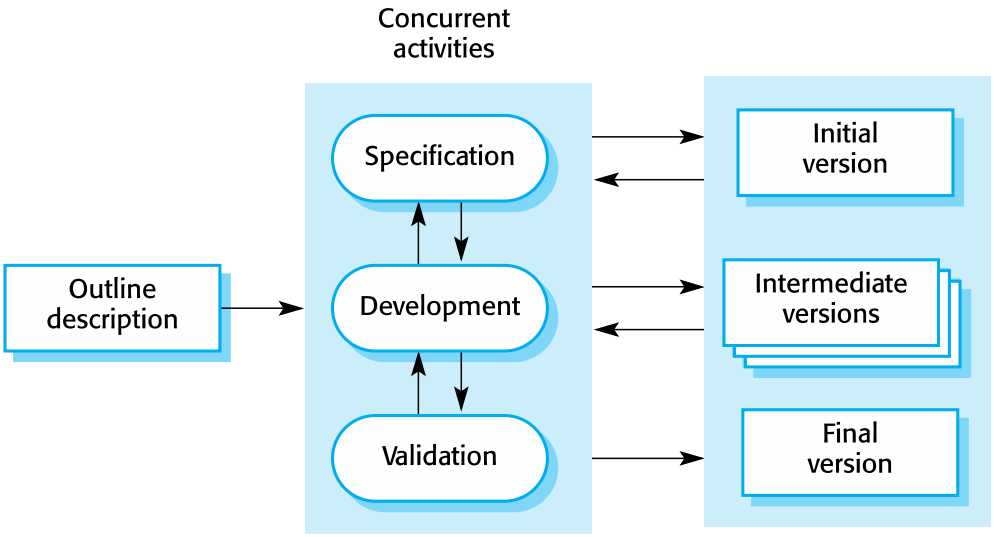
Sharshara modeli asosiy bosqichlari bevosita dasturiy ta'minotni rivojlantirish bosqichini aks ettiradi:



1. ***Requirements analysis and definition*** Tizim foydalanuvchilari bilan maslahatlashib tizimning servislari, chegaraklari va maqsadlari belgilab olinadi
2. ***System and software design*** Tizim dizayni jarayonida tizim arxitekturasini tashkil qilish orqali apparat yoki dasturiy tizimlarga talablar belgilanadi. Dasturiy ta'minot dizayni fundamental dasturiy ta'minot tizimlari mavhumliklari va ularning munosabatlarini identifikatsiyalash va tasvirlashni o'z ichiga oladi.
3. ***Implementation and unit testing*** Bu bosqichda dasturiy ta'minot dizayni dasturlar to'plami yoki dastur bo'limlar ko'rinishda amalga oshiriladi. Bo'lim testlash jarayonida har bir bo'lim alohida testlanadi.
4. ***Integration and system testing*** Individual dastur bo'limlari yoki dasturlar birlashtiriladi va to'liq tizim ko'rinishida testlanadi. Testlashdan keyin dasturiy ta'minot tizimi foydalanuvchiga yetkaziladi.
5. ***Operation and maintenance*** Bu eng uzun hayot sikli fazasi hisoblanadi. Tizim o'rnatiladi va amaliy foydalanishga qo'yiladi. Maintance o'z ichiga hayot siklining oldingi bosqichlarida tuzaqtilmagan xatoliklarni tuzatish, tizim bo'limlarining amaliy ko'rinishlarini yaxshilash va tizim servislarini yangi talablarga mos ravishda yaxshilashni o'z ichiga oladi.

Ortib borish modeli

Incremental(ortib borish) ishlab chiqish dastlabki amaliy ko'rinishni ishlab chiqish g'oyasiga asoslangan. Foydalanuvchi izohlariga qarab tizimning keyingi versiyalari ishlab chiqiladi.

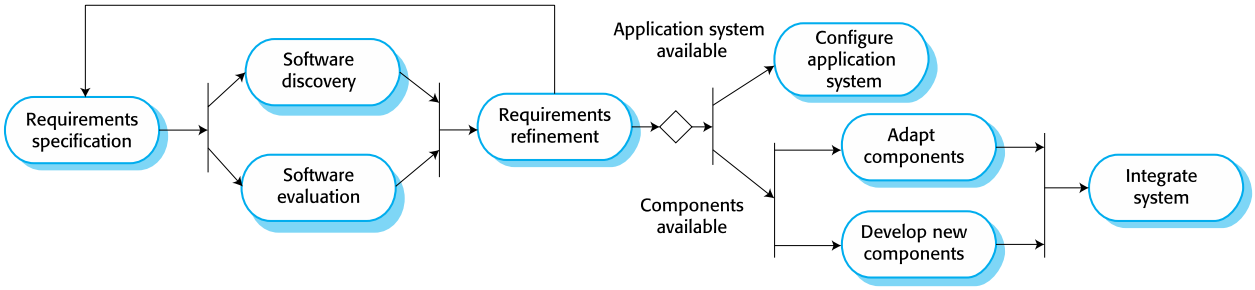


Qayta foydalanishga mo’ljallangan dasturiy injiniring

Ko’pgina dasturiy ta’minot loyihalarida bir nechta qayta foydalaniladigan dasturiy ta’minotlar mavjuv.

Qayta foydalanishga mo’ljallangan jarayonlarda foydalanish mumkin bo’lgan uch xil turdagi dasturiy ta’minot komponentalari bor:

1. Veb servislar servis standartlariga ko’ra ishlab chiqilgan
2. .NET yoki J2EE kabi componenta freymvorklarga integratsiya qilinadigan paketlangan obyektlar kolleksiyasi
3. Maxsus muhitlarda foydalanish uchun sozlangan autonom dasturiy ta’minot tizimlari.



1.2. Sifatli va tezkor dasturiy ta’minot ishlab chiqish

[2]Tezkor dasturiy ta’minot ishlab chiqish jarayoni foydali dasturiy ta’minotni tezda tayyorlashga mo’ljallangan.

Tezkor dasturiy ta’minot ishlab chiqishning bir nechta yondashuvlari mavjud bo’lsada ularning asosiy xarakteristikalari mavjud:

1. Xususiyatlarni aniqlash, dizaynlash va amaliy ko’rinishga keltirish jarayonlari birlashtirilgan. Tizimning xususiyatlari batafsil keltirilmaydi va dizayn hujjatlari minimumlashtirilgan. Foydalanuvchi talablari hujjatlari tizimning muhim xarakteristikasi hisoblanadi.
2. Tizim bir necha talqinda ishlab chiqariladi. Foydalanuvchi yangi talablarini qondirish maqsadida dasturiy ta’minotning novbatdagi talqini ishlab chiqariladi.
3. Tizimning foydalanuvchi interfeysini tezda tayyorlash maqsadida IDS(integrated development system)

1.3. Tizim uchun talablarni shakllantirish

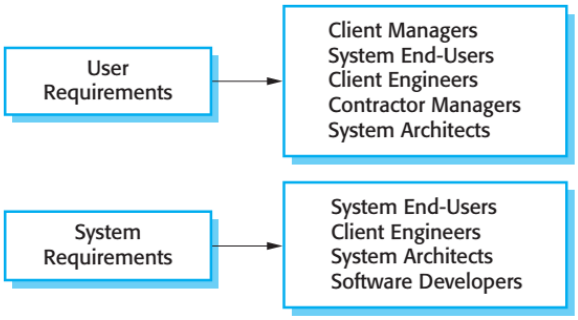
[2] “Software Engineering”, by Ian Sommerville, pages 57-60

[3]Tizim uchun talablar bu tizim nima ish bajarish lozimligini tasvirlashdir. Talablar tizim mijozlarini ehtiyojlarini aks ettiradi.

Talablar injiniringi jarayonida ko'pgina muammolar ko'tariladi. Foydalanuvchi talablari' va 'tizim talablari' terminlari orasida farq mavjud. Foydalanuvchi talablari va tizim talablari quyidagicha izohlanishi mumkin:

1. Foydalanuvchi talablari bu diagrammalar bilan tabiiy tildagi bayonotlar.
2. Tizim talablari bu dasturiy ta'minot tizimi funksiyalari, servislari va operativ cheklanishlarining batafsil tasvirlanishi.

Siz talablarni turli xil darajada yozishingiz kerak chunki turli xil o'quvchilar turli xil yo'lda foydalanishadi.



Dasturiy ta'minot tizimi talablari funksional va funksional bo'lmagan talablar sinflariga ajratiladi.

1. Funksional talablar Bu tizim taminlashi lozim bo'lgan servislarning bayonoti. Kiritilgan ma'lumotlarga tizim qanday reaksiya ko'rsatishi lozim, tizim o'zini bunday holatlarda qanday tutushi lozim
2. Funksional bo'lmagan talablar Bu tizim tomonidan taklif qilinayotgan servislar va funksiyalardagi cheklovlar. U o'z ichiga vaqt cheklanishi, ishlab chiqarish jarayoni cheklanishi, beriladigan standartlar tomonidan cheklanishlarni olishi mumkin.

Dasturiy ta'minot talablari hujjati

Dasturiy ta'minot hujjati bu tizimni ishlab chiquvchilar nimani oshirishi lozimligini ifodalovchi rasmiy hujjatdir. U tizim uchun foydalanuvchi talablarini ham tizim talablarining batafsil spesifikatsiyasini ham o'z ichiga oladi. Bazida foydalanuvchi va tizim talablari bitta qilib tavsiflanadi. Bazi hollarda esa foydalanuvchi talablari hujjatning kirish qismi va tizim talablari asosiy qismni tashkil qiladi.

Talablar hujjatidan foydalanuvchilar

[3] "Software Engineering", by Ian Sommerville, pages 83-88

Tizim mijozlari	Talablarni ko’rsatish va talablar bajarilganligiga tekshirish uchun o’qish. Mijozlar shuningdek talablarni o’zgartirishi mumkin.
Boshqaruvchilar	Tizimni narxlash va ishlab chiqishni rejalashritish uchun talablar hujjatidan foydalanish.
Tizim injinerlari	Ishlab chiqarilayotgan tizimni tushunish uchun talablardan foydalanish.
Tizimni testlovchi injinerlar	Tizimni haqiqiylikka tekshirish uchun tizm talablaridan foydalanish.
Tizimga xizmat ko’rsatuvchi injinerlar	Tizim va uning qismlari munosabatini tushunish uchun talablardan foydalanadi.

Talablar hujjatining strukturasi

Bo’lim	**Tavsifi**
Muqaddima	Hujjatni kutilgan o’quvchilarini aniqlash lozim
Kirish	Tizim muhimligini tasvirlash. Tizim funksionalligi qisqacha tasvirlanadi.
Glossariy	Hujjatda foydalanilgan texnik terminlarni aniqlash
Foydalanuvchi talablari	Foydalanuvchi uchun taminlangan servislarni tasvirlash
Tizim arxitekturasi	Kutilgan tizim arxitekturasini yuqori-darajali ko’rinishi

Tizim talablari	Funksional va funksional bo'lmagan talablarning batafsil ko'rinishi
Tizim modellari	Tizim componentalari orasidagi munosabatlarni grafik tizimini ko'rsatish
Tizim evolutsiyasi	Tizimga asoslanib fundamental taxminlarni tasvirlash
Ilova	Ishlab chiqarilayotgan ilova haqida batafsil ma'lumotlar; masalan, apparat ta'minot va ma'lumotlar bazasi
Index	Hujjat indekslari

1.4. Dasturiy ta'minotni modellashtirish

[4]Modellar talablar injiniringi jarayonida tizim uchun talablarni hosil qilishda foydalaniladi. Siz mavjud tizimlarning modelini va ishlab chiqarilayotgan tizimning modelini tuishingiz mumkin.

1. Mavjud tizim modellari talablar injiringi mobaynida foydalaniladi. Ular mavjud tizimning nima ish bajarishini aniqlashtiradi va tizimning kuchli va kuchsiz tomonlarini muhokama qilishga asos bo'ladi. Bu yangi tizim uchun talablar ishlab chiqishga olib keladi.
2. Yangi tizim modellari talablar injiniringi davomida yordam berish uchun ishlatiladi. Injinerlar dizayn bo'yicha takliflarni muhokama qilishda modellardan foydalanadi.

Tizim modelining eng muhim tomoni shundaki unda tizim haqidagi batafsil ma'lumotlar tashlab ketiladi. Model o'rganilayotgan tizimning mavhum ko'rinishidir.

Siz tizimni turli xil ko'rinishlarini ko'rsatish uchun turli xil modellarni ishlab chiqishingiz mumkin. Masalan:

1. Tashqi ko'rinish, tizimning konteksti yoki muhitini modellashtirish.
2. O'zaro munosabatlar ko'rinishi, tizim bilan muhit yoki tizim komponentalari o'rtasidagi o'zaro munosabatni modellashtirish.

[4] "Software Engineering", by Ian Sommerville, pages 119-121

3. Strukturaviy ko’rinish, tizim tomonidan ishlov berilayotgan ma’lumotlar strukturasi yoki tizim tashkilotini modellashtirish.
4. Xatti harakatlar ko’rinishi, tizimning dinamik xatti harakatlari va hodisalarga qanday javob berishini modellashtirish.

Tizimning turli xil modellarini yaratish uchun UML bir nechta diagrammalarga ega.

1. Faoliyat diagrammalari, jarayondagi faoliyatlarni ko’rsatadi
2. Foydalanish holati diagrammari, tizim va uning muhiti o’rtasidagi munosabatni ko’rsatadi.
3. Ketma-ketlik diagrammalari, shaxs va tizim va tizim komponentalari orasidagi munosabatlarni ko’rsatadi.
4. Sinf diagrammalari, tizimdagi obyektlar sinflari va ularning o’zaro munosabatini ko’rsatadi
5. Holat diagrammari, tizimning ichki va tashqi hodisalarga ta’sirini ko’rsatadi.

UML(Unified Modeling Language) - birlashgan modellashtirish tili dasturiy ta’minot tizimlarini modellashda 13 ta turli xil diagramma turlaridan foydalanadi. UML dasturiy ta’minot tizimlarini modelini yaratishda standart yondashuv deb qabul qilingan.

Konteks modellar

Konteks modellar tizimning tezkor kontekstini ko’rsatishda foydalaniladi. Arxitekturaviy modellar tizim va uning boshqa tizimlar bilan munosabatini ko’rsatadi.

Tizim chegaralari

Tizim chegaralari nima tizim ichida va nima tizim tashqarisidaligini ko’rsatadi. Ular ishlab chiqarilayotgan tizimda foydalanilayotgan yoki bog’liq bo’lgan boshqa tizimlarni ko’rsatadi.

Konteks modellar muhitdagi ishlab chiqarilayotgan tizimni emas balki muhitdagi boshqa tizimlarni ko’rsatadi.

Jarayon modellar ishlab chiqarilayotgan modellarni ko’rsatadi. UML diagrammalar jarayon modellarda foydalaniladi.

Nazorat savollari

1. Dasturiy ta’minot nima?
2. Yaxshi dasturiy ta’minotning attributlari nima?
3. Dasturiy injiniring nima?

4. Dasturiy injiniringni asosiy faoliyatlari nima?
5. Dasturiy injiniring va kompyuter ilmi o’rtasida qanday farq bor?
6. Dasturiy injiniring va tizim injiniring o’rtasida qanday farq bor?
7. Dasturiy injiniringni narxi qancha?
8. Eng yaxshi dasturiy injiniring usuli va metodi qaysi?
9. Dasturiy injiniringa veb qanday xilma-xillik olib keldi?

Foydalanilgan adabiyotlar

1. “Software Engineering”, by Ian Sommerville, 2015, pages – 790.
2. Holdener, A. T. (2008). Ajax: The Definitive Guide. Sebastopol, Ca.: O’Reilly and Associates.
3. Abrial, J. R. (2005). The B Book: Assigning Programs to Meanings. Cambridge, UK: Cambridge University Press.
4. http://www.SoftwareEngineering-9.com
5. http://www.pearsonhighered.com/sommerville

2 - ma’ruza. Talablarni ishlab chiqish va model-lashtirish. UML holat diagrammalarini shakllantirish. Dasturiy ta’minotning arxitekturaviy dizayni. Dasturiy ta’minotni boshqarish.

Reja:

2.1. Dasturiy ta’minotning arxitekturaviy dizayni
2.2. Dasturiy ta’minot dizaynini qurish va moslashtirish
2.3. Dasturiy ta’minotni testlash
2.4. Dasturiy ta’minot evolutsiyasi

Kalit so’zlar: *arxitektura, dizayn, testlash, evolutsiya, xavfsizlik, ishinchlilik, barqarorlilik, servis, uml, talab, servis.*

Ushbu ma’ruzaning maqsadi dasturiy ta’minot arxitekturasi va arxitekturaviy dizayn tushunchalariga kirish, dasturiy ta’minot dizaynini yaratish va uni tizimga moslashtirish, daturiy ta’minotni testlash hamda dasturiy ta’minot evolyutsiyalarini ko’rib chiqishdan iborat.

2.1. Dasturiy ta’minotning arxitekturaviy dizayni

[5]Dasturiy ta’minotning arxitekturaviy dizayn, tizimning umumiy tuzilishi dizayni va uni qanday tashkillashtirish tushunchalari bilan bog’liqdir.

Siz dasturiy ta’minot arxitekturasini ikkita ajralmas bosqichlarda dizaynlashtira olasiz. Bular, kichik arxitektura va katta arxitektura:

1. Kichik arxitekturaga shaxsiy dasturlarning arxitekturalarini o’z ichiga oladi.
2. Katta arxitektura boshqa tizimlarni, dasturlarni va dasturiy komponentalarni qamrab oladigan murakkab korxona tizimlari arxitekturasini o’z ichiga oladi.

Dasturiy ta’minot arxitekturasi tizimni ishlab chiqishda muhim o’rin tutadi, sababi u tizimni ishlab chiqilishiga, ishonchliligiga, keng ko’lamda qo’llanilishiga va qayta ishlab chiqilishiga ta’sir qiladi. Shaxsiy komponentalar funksional tizim talablarini bajaradi. Nofunksional talablar tizim arxitekturasidan kelib chiqadi – bu komponentalar tashkil qilinadigan va bir-biriga bog’lanadigan yo’ldir. Ko’plab tizimlarda, nofunksional talablar ham shaxsiy komponentalar tomonidan bajariladi, lekin bu yerda shubxasiz tizim arxitekturasi ustuvor hisoblanadi.

Bass va boshqalar(2003) dasturiy ta’minot arxitekturasini ochiq loyihalashtirish va hujjjatlashtirishning uchta afzalliklarini muhokama qilishdi:

[5] “Software Engineering”, by Ian Sommerville, pages 148-150

1. *Manfaatdor aloqalar.* Arxitektura tizimning yuqori darajadagi taqdimoti bo'lib, bir qator turli manfaatdor tomonlar tomonidan muhokama qilish uchun bir muammo markazi sifatida foydalaniladi.
2. *Tizim tahlili.* Tizimni ishlab chiqishning erta bosqichlarida tizim arxitekturasini yaratish bir qancha tahlillarni talab qiladi. Arxitekturaviy loyiha yechimlari tizimning ishlab chiqilish, ishonchlilik, keng ko'lamda qo'llanilish va qayta ishlab chiqilish talablarini qondira olishiga chuqur ta'sir ko'rsatadi.
3. *Keng ko'lamli qayta qo'llash.* Tizim arxitekturasi modeli tizimning qanday tashkil qilingani va komponentalar qanday o'zaro ta'sir qilishini ixcham, boshqarilga ta'rifidir.

Dasturiy tizim arxitekturasi alohida arxitekturaviy shablonlar va stillarga asoslangan. Arxitekturaviy shablonlar xuddi klient-server tashkillanishi yoki bosqichlangan arxitektura kabi tizimni tashkillashtirish tushunchasidir.

Arxitekturaviy shablonlar turli dasturiy tizimlarda qo'llanilgan arxitekturalar jamlanmasini o'z ichiga oladi. Tizim uchun arxitektura tanlashda ehtiyotkorlik bilan qaror qabul qilishingiz lozim.

Nofunksiyaviy talablar va dastur arxitekturasi o'rtasida yaqin bog'liqliklar bo'lganligi sababli, siz tanlayotgan arxitekturaviy stil va tizim, nofunksional tizim talablaridan kelib chiqqan holda tanlanishi lozim:

1. *Ishlab chiqish.* Ishlab chiqish bu muhim talabi bo'lsa, arxitektura kichik komponentlar soni doirasida muhim operatsiyalarini mahalliylashtirishga mo'ljallangan bo'lishi kerak, bu komponentlar bilan hammasi bitta kompyuterda yozilgan, butun tarmoq bo'ylab tarqalmagan bo'lishi lozim.

2. *Xavfsizlik.* Agar xavfsizlik muhim talab bo'lsa, arxitektura uchun qatlamli tuzilishi chuqur qatlamlarda himoyalangan eng muhim aktivlardan foydalanish kerak, bu qatlamlar uchun xavfsizlikni tekshirish yuqori darajada qo'llaniladi.

3. *Ichonchlilik.* Agar ishonchlilik muhim talab bo'lsa, arxitektura shunday mo'ljallangan bo'lishi kerakki, bunda xavfsizlik bilan bog'liq operatsiyalarning barchasi yoki yagona komponentada yoki kichik sonli komponentlarda joylashgan bo'lishi kerak. Bu xavfsizlik tekshirish xarajatlarni va muammolarini kamaytiradi va qobiliyatsiz taqdirda tizimini xavfsiz yopilishiga imkon beradigan tegishli himoya tizimlarini taqdim qiladi.

4. *Mavjudlik.* Agar mavjudlik muhim talab bo'lsa, arxitektura tizimi to'xtamasdan komponentlarini yangilash va almashtirish imkoniga ega ortiqcha komponentlarini o'z ichiga olgan bo'lishi kerak.

5. *Barqarorlik.* Agar barqarorlik muhim talab bo'lsa, tizim arxitekturasi tez-

tez o’zgarib turishi mumkin bo’lgan nozik, o’z-o’zini tarkibidagi komponentalardan foydalangan holda qurilishi lozim. Ma'lumotlar ishlab chiqaruvchilar iste'molchilardan ajratilgan bo'lishi kerak va birgalikda ma'lumotlar tuzilmalari yo'l qo'ymaslik lozim.

2.2. Dasturiy ta’minot dizaynini qurish va moslashtirish

[6]Dasturiy ta’minot dizayni va uni ishlab chiqish bajariladigan dasturiy ta'minot tizimi ishlab chiqiladigan dasturiy injiniring jarayonidagi bosqichdir.
Ba'zi oddiy tizimlar uchun, dasturiy ta'minot dizayni va dasturni ishlab chiqish dasturiy injiniringdir, va boshqa barcha faoliyatlar bu jarayonda bilan birlashgan. Shu bilan birga, katta tizimlar uchun, dasturiy ta'minot dizayni va dasturni ishlab chiqish dasturiy injiniringda ishtirok etadigan jarayonlar majmuidan biridir.

Dasturiy ta’minot dizayni va dasturiy ta’minotni ishlab chiqish faoliyati o’zgarishsiz ketma-ketlikda kelmoqda. Dasturiy ta’minot dizayni foydalanuvchi talablariga asoslangan dasturiy ta’minot komponentalari va ularning o’zaro bog’liqligi ta’minlashda yaratiladigan faoliyatdir. Dasturiy ta’minotni ishlab chiqish bu dizaynni dasturga moslab joriy etish jarayonidir. Ba’zi hollarda, bo’lingan dizayn bosqichlari ham mavjud va bu dizayn modellashtirilgan va dokumentlashtirilgan bo’ladi. Boshqa hollarda esa, dizayn dasturchi miyasida yoki doskada yoki qog’ozlarda aks etadi. Dizaynlashtirish bu muammoni qanday bartaraf etishdir, shuning uchun har doim dizaynlash jarayoni mavjuddir. Shunga qaramasdan, UML yoki boshqa dizayn yaratuvchi tillardan foydalanishda dizaynni tushuntirish bu doimo muhim yoki mos ravishda bo’lmaydi.

Dizaynlashtirish jarayonidagi eng muhim bosqich bu sizga kerak bo’lgan dizayn modellari va ularda talab qilingan detallarning bosqichlari ustida qarorlar qabul qilishdir. Bu ayni vaqtda ishlab chiqilayotgan tizim turidan kelib chiqadi.

Tizim loyihasini umumiylikdan detallarigacha ishlab chiqishda, obyektga yo’naltirilgan dizaynda quyidagi bir qanch narsalarni bilishingiz lozim:

1. Tizim bilan bo’ladigan tashqi ta’sirlar va kontekstni aniqlash va ularni tushunish.
2. Tizim arxitekturasini loyihalash.
3. Tizimdagi asosiy obyektlarni birlashtirish.
4. Loyiha modellarini rivojlantirish.
5. Interfeyslarni aniqlash.

Dastur dizaynini ishlab chiqishda UMLdan foydalanganingizda, siz ikki

[6] “Software Engineering”, by Ian Sommerville, pages 177-180

turdagi dizayn modellarini normal holda ishlab chiqasiz:

1. Tizimli modellar. Ular obyekt sinflari va ularning o'zaro bog'liqligidan foydalanadigan tizimning dinamik tuzilmasini tasvirlaydi.
2. Dinamik modellar. Ular tizimning dinamik tuzilmasini tasvirlaydi va tizim obyektlari o'rtasidagi o'zaro ta'sirlarni ko'rsatadi.

O'zaro ta'sirlar obyektlar tomonidan bajarilgan xizmat so'rovlarining ketma-ketligini o'z ichiga olgan holda hujjatlashtirilgan bo'lishi mumkin.

Ishlab chiqish jarayonining bir qismida, siz qanday qilib ishlab chiqilgan dasturiy ta'minot maqsaddagi platformada qanday tarqalgan bo'lishi haqida qarorlar qabul qilishingiz lozim. Tarqalgan tizimlar uchun tarqaladigan komponentalar mavjud maxsus platformalar ustida qaror qabul qilishingiz lozim bo'ladi. Qaror qabul qilishda ko'rib chiqishingiz kerak bo'lgan muammolar:

1. Komponentning qurilma va dasturiy talablari. Agar component maxsus qurilma arxitekturasi uchun loyihalashtirilgan bo'lsa, yoki boshqa dasturiy ta'minot tizimlarida qo'llanilsa, bu shubhasiz talab qilingan qurilma va dasturni qo'llab quvvatlovchi platformada tarqatiladi.
2. Tizim mavjudligi talablari. Yuqori-mavjud tizimlar bittadan ko'p platformada tarqalgan komponentalarni talab qilishi mumkin. Bu shuni anglatadiki, platform inkor qilgan holatda komponentning alternative ishlab chiqilishi mavjud.
3. Kommunikatsiya komponentlari. Agar komponentlar orasida yuqori darajali aloqa trafigi bo'sa, unda ularni o'sha platformada yoki fizik jihatdan bir-biriga yaqin bo'lgan platformalarda aralashtirish kerak. Bu aloqa ushlanib qolishini, vaqtlar orasidagi ushlanishni kamaytiradi, xabar bitta component tomonidan jo'natiladi va boshqasi tomonidan qabul qilinadi.

2.3. Dasturiy ta'minotni testlash

[7]Testlash, dasturiy ta'minotni foydalanishga qo'yishdan oldin dastur nuqsonlarini topish va ularni to'g'irlashga mo'ljallangan dasturlarni ko'rsatishga mo'ljallangan. Siz dasturiy ta'minotni testlagan chog'ingizda, sun'iy ma'lumotlardan foydalanib dasturni ishga tushirasiz. Siz dasturni testlash natijalarini xatolarga, anomaliya(normal holatdan chetlashish)ga yoki dasturning nofunksional sifatlari haqida ma'lumotga tekshirasiz.

Testlash jarayonida ikkita alohida maqsadlar mavjud:

1. Ishlab chiqaruvchi va buyurtmachiga ularning dasturiy ta'minoti talablari bajarilayotganini namoyish etish. Buyurtma qilingan dasturiy ta'minot uchun

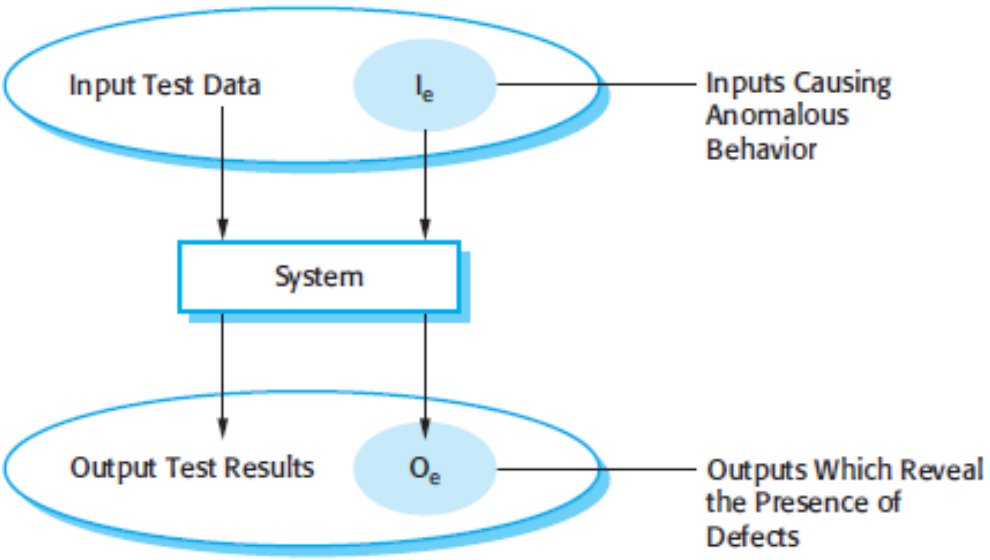
[7] "Software Engineering", by Ian Sommerville, pages 206-210

hujjatdagi talablarning har biri uchun kamida bitta testlash bo'lishi lozim. Umumiy dasturiy ta'minot mahsulotlari uchun esa, tizimning barcha funksiyalari uchun, shuningdek, tayyor mahsulotda ishlatiladigan funksiyalar aralashmasi uchun testlashlar bo'lishi kerak.

2. Dasturiy ta'minot noto'g'ri, ishonarsiz yoki spetsifikatsiyalarga mos kelmagan hollarni aniqlash. Ular dasturiy ta'minotning nuqsonlari hisoblanadi. Nuqsonlarni testlash keraksiz tizimlarning nuqsonlariga barham berish bilan bog'liq, masalan, tizimning to'xtab qolishi, boshqa tizimlar bilan keraksiz bo'g'lanishi, ma'lumotlarning noto'g'ri hisoblanishi va buzilishi.

Birinchi maqsad bu ishlatilishi kutilayotgan tizim tekshirishlarini testlashdir. Bunda ishlatilishi kutilayotgan tizimni tekshiruvchilarni berilgan testlar jamlanmasidan to'g'ri foydalangan holda testlash lozim. Ikkinchi maqsad, tizim nuqsonlarini testlashga olib keladi. Bunda tizim nuqsonlarini ko'rsatishi uchun nazorat misollari qo'yilgan bo'ladi. Albatta, testlashning bu ikki yo'li o'rtasida aniq bir chegara yoq. Tekshiruvlarni testlash vaqtida siz tizimdagi nuqsonlarni topasiz; Nuqsonlarni testlash vaqtida bazi testlar dasturiy ta'minot ularning talablari javob berayotganini ko'rsatadi.

Rasmda ko'rsatilgan sxema, tekshirishlarni testlash va nuqsonlarni testlash orasidagi farqlarni tushunishga yordam beradi. Testlovchi tizimingizni qora quti deb tasavvur qiling. I deb o'rnatilgan kiruvchi signallardan tizim kiruvchi signallar qabul qiladi va chiquvchi signallarni O deb o'rnatilgan chiqishga uzatadi. Chiqishlarning ba'zilari xatolikka tutilishi mumkin. Bular Oye jamlanmasidagi chiquvchilardir, ular Ie jamlanmasidagi kiruvchi signallarga javoban ishlab chiqilgandir. Nuqsonlarni testlashda birinchi o'rinda Ie da o'rnatilgan kiruvchilarni topishdir, chunki ular tizim bilan bog'liq muammolarni ochib beradi. Tekshirishlarni testlash Ie dan tashqarida joylashgan to'g'ri kiruvchilar bilan teslashni o'z ichiga oladi. Ular kutilayotgan to'g'ri natijalarni olish uchun tizimni kuchaytiradi.



Testlash dasturiy ta’minotning nuqsonlarga egaligi yoki bir aniq vaqtda u o’zini ko’rsatilgandek tutishini namoyon qilmaydi. Siz kuzatayotgan test kelgusida tizim bilan yuz beradigan muammolarni topishi imkoniyati doimo mavjud.

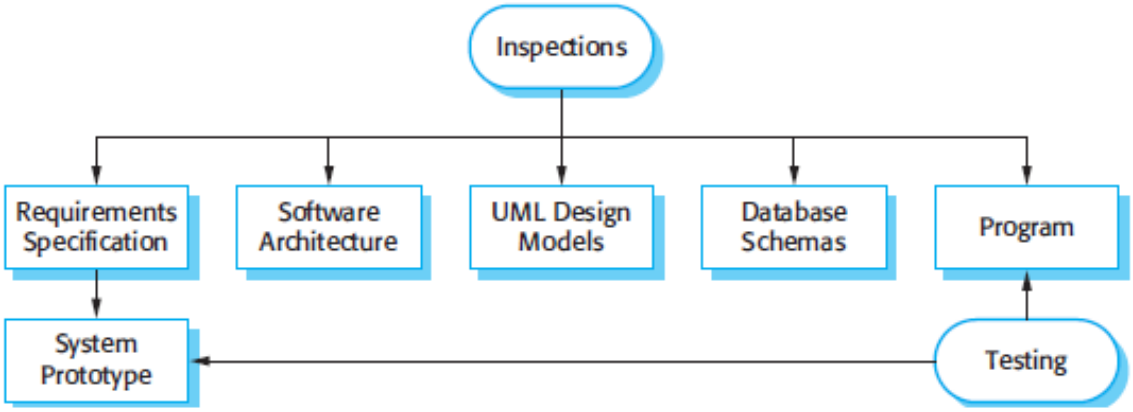
Verifikatsiya va validatsiya jarayonlari odamlar pul to’laydigan dasturiy ta’minotning funksional imkoniyatlarini ta’minlash va uning shartlariga mos kelishini tekshirish bilan bog’liq. Bu tekshirish jarayonlari talablar mavjud hollarda va ular ishlab chiqarish jarayonlarining barcha bosqichlarini davom ettirish vaqtida boshlanadi.

Verifikatsiyaning maqsadi dasturiy ta’minot unga qo’yilgan funksional va nofunksional talablarga javob berishini tekshirishdir. Verifikatsiya umumiy jarayon hisoblanadi.

Validatsiyaning maqsadi dasturiy ta’minot buyurtmachi kutayotgan natijalarga mosligini ta’minlashdir. Validatsiya muhim rol o’ynaydi, sababi, ma’lumotlarni tasniflash talablari har doim ham mijozlar va foydalanuvchilar talablari va istaklarini amalga oshiravermaydi.

Varifikatsiya va validatsiya jarayonining yakuniy maqsadi esa daturiy ta’minot tizimi “maqsadga mos kelishi” ga ishonchni o’rnatishdir.

Yetarli darajadagi ishonch bosqichi, tizimning maqsadi, tizim foydalanuvchilarning talablaridan, shuningdek, tizim uchun joriy marketing muhitidan kelib chiqadi:



1. *Dasturiy ta'minot.* Dasturiy ta'minotning eng kritik, eng muhimligi bu uning ishonchliligidir. Masalan, tanqidiy tizimlarni ehtiyot qilishni boshqarish uchun qo'llaniladigan dasturiy ta'minot uchun kerak bo'lgan ishonch darajasi, mahsulotni yangi g'oyalarini namoyon etish uchun ishlab chiqilgan prototip uchun bo'ladigan talabdan ancha yuqoridir.

2. *Foydalanuvchi kutayotgan natijalar.* Ularning buglar bilan, ishonarsiz dasturiy ta'minot bilan tajribalari bo'lganligi tufayli, ko'plab foydalanuvchilar sifatli dasturiy ta'minotni uqadar kutadilar. Ular dasturiy ta'minot ishdan chiqishidan hayratga tushmaydilar. Yangi tizimni o'rnatayotganlarida foydalanuvchilar muvoffaqiyatsizlikka ham chidashadi, sababi dasturiy ta'minotdan foydalanish dasturni to'xtashlaridan keyingi qayta tiklanishiga ketgan harajatlarni ustunroq keladi. Bu kabi holatlarda dasturiy ta'minotni testlashga ko'p vaqt ajratishingiz kerak bo'lmaydi. Biroq, dasturiy ta'minot tugallanishga yetishi bilan foydalanuvchilar uning ishonchli, ular xoxlaganidek bo'lishini kutadilar, shuning uchun dasturiy ta'minotni ishlab chiqishni oxirgi bosqichlarida ko'proq testlash talab qilinishi mumkin.

3. *Marketing muhiti.* Tizim bozorga chiqqanida, tizimni sotuvchilar raqobatchi mahsulotlarni, sotib oluvchilar to'lashga tayyor bo'lgan narxlarni, shuningdek, ushbu tizimni yetkazilib berilishi uchun talab qilinadigan grafikni inobatga olishlari lozim. Raqobatbardosh muhitda, dasturiy ta'minotni ishlab chiqaradigan korxonalar, uni umumiy testlanishidan oldin dasturni foydalanishga topshirishga qaror qabul qiladilar, sababi ular bozorda birinchi o'rinda turishni xoxlaydilar. Agar dasturiy ta'minot judayam arzon baholangan bo'lsa, foydalanuvchilar uning ishonarsizligiga sabr qilishga tayyor bo'ladilar.

2.4. Dasturiy maxsulotning evolutisiyasi

[8]Dasturiy maxsulotning o’zgarishi muqarrar

- Dasturiy maxsulotdan foydalanilayotganda yangi talablar yuzaga keladi
- Biznes muhitning o’zgadi
- Xatoliklarning tamirlanish majburiyati
- Yangi kompyuter va jihozlarning sistemaga qo’shilishi
- Sistemaning ish bajarishi yoki ishonchliligini oshirishga majbur bo’lish

Barcha tashkilotlar uchun asosiy muammo ularning mavjud dasturiy taminoti uchun o’zgarishlarni amalga oshirish va boshqarishdir

Evoulutsiyaning ahamiyati

- Tashkilolarning dasturiy taminot tizimlarida juda katta investitsiyasi bo’lishi bu katta mulkdir
- Bu mulkni biznesda qiymatini saqlab qolish uchun ular o’zgartirilishi va yangilanib borishi lozim
- Katta kompaniyalardagi dasturiy maxsulot mablag’ining katta qismi yangi dasturiy taminot yaratgandan ko’ra mavjjud dasturiy taminotni rivojlantirish va o’zgartirishga sarflanadi

Evalutsiya va servis xizmat

Evolutsiya

- Bu dasturiy taminot hayot siklining shunday bosqichiki bunda u tezkor oshlatishda bo’ladi va taklif qilingan yangi talablar bosqichma-bosqich shaklanadi.hamda sistemada amaga oshiriladi.

Servis xizmat ko’rsatish

- Bu bosqichda dasturiy maxsulot foydali bo’lib qoladi lekin faqatgina o’zgarishlar uning tezligini oshirish maqsadida qo’shiladi ya’ni dasturiy taminotda muhitida xatolarni to’grilash va o’zo’zgarishlarni tasvirlash amalga oshiriladi. Yangi funksiyalar esa qo’shilmaydi.

Bosqichdan chiqish

- Dasturiy taminot haligacha ishlatiladi lekin uni hosil qilish uchun yangi o’zgartirishlar kiritilmaydi

Elolutsiya jarayoni

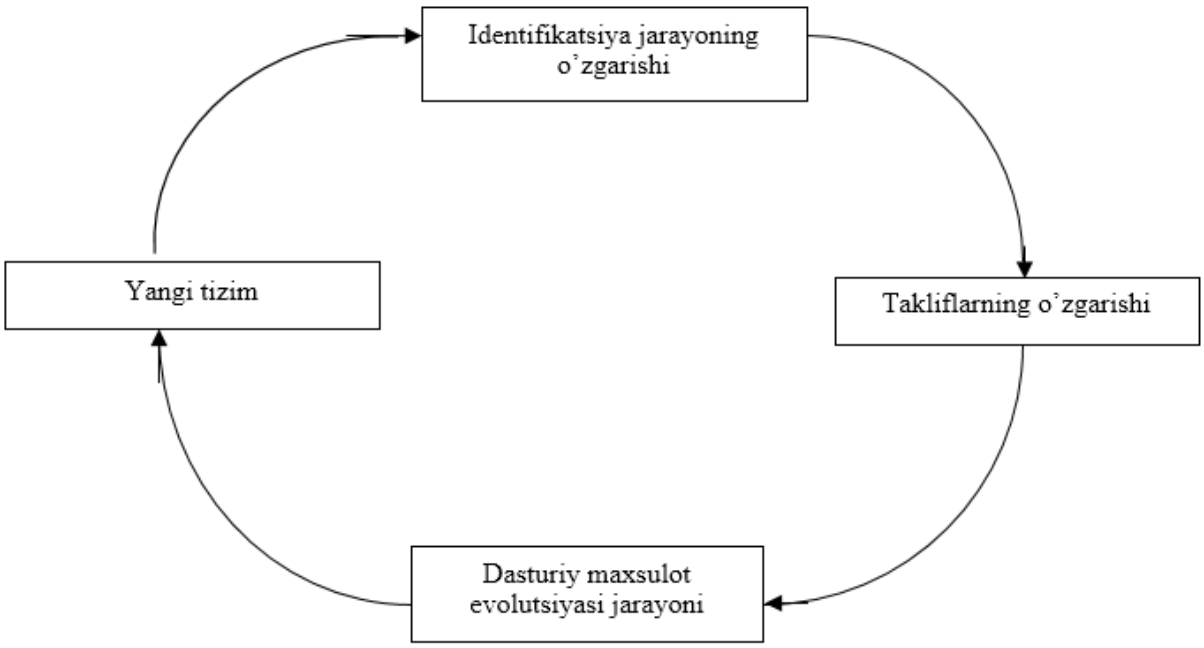
Dasturiy maxsulot evolutsiyasi quyidagilarga bog’liq:

[8] “Software Engineering”, by Ian Sommerville, pages 235-239

- Saqlanib kelayotgan dasturiy maxsulotning turi
- Foydalanilayotgan qurilish jarayoni
- Loyihaga jalb qilingan ishchilarning tajribasi va qobilyati

O’zgarishlar uchun takliflar tizim evolutsiyasi uchun asosiy hisoblanadi Inedtifikatsiya va evolutsiyaning o’zgarishi tizimning butun hayot sikli mobanida davom etadi.

Evolutsiya va identifikatsiya o’zgarish jarayonlari



Amalga oshirishning o’zgarishi

Sakllantirilhan amalga oshirilgan va test qilingan sistemani takrorlantiradigan yaratilish larayonining qaytishi(takrorlanishi)

Muhim farq shundaki amalga oshirish o’zgarishining birinchi bosqichi o’z ichiga dasturni tushunishni oladi. ayniqsa bu holathaqiqiy tizim tashkilotchilari amalga oshirishga javobgar bo’lmaganlaridasodir bo’ladi.

Dastruni tushunish bosqichi davomida siz dasturning tuzilish strukturasini, dasturga yaxshi tasir qiladigan o’zgartirishlarni qanday taklif qilishni va uning funksiyonalligini qanday taminlab berishni tushunishingiz shart.

Zarur o’zgarishlar talablari

Zarur o’zgarishlar dasturiy enginering jarayoning barcha bosqichlarida ham amalga oshirish shart bo’lmasligi mumkin

- Jiddiy tizim xatosi normal amaliyot davom ettirishga imkon berishni tuzatish kerak bo’lganda
- Agar sistema muhiti uchun kutilmagan tasirlar bo’lsa

- Agar tezda javob qaytarishni talab qiladigan bizness talablar bo’lsa.

Nazorat savollari

1. Nima uchun dasturiy ta’minot arxitekturasini loyihalash muhim?
2. Internetda musiqalar sotuvchi iTunes kabi tizimlarning arxitekturasini tuzing.
3. Ob-havo stansiyasi dizaynini ishlab chiqing.
4. Quyidagi obyekt sinflari uchun UML grafik notatsiyalardan foydalanib dizayni quring.
5. Regression testash nima?
6. Testlashning dastlabki bosqichlarida foydalanuvchilarni jalb qilishning ahamiyati nimada?
7. Dasturiy ta’minot evolutsiyasi nima?
8. Dasturiy ta’minotni mijozlarga yetkazilganidan keyingi hayot sikli fazasi qanday faoliyatlarni o’z ichiga oladi?

Foydalanilgan adabiyotlar

1. “Software Engineering”, by Ian Sommerville, 2015, pages – 790.
2. Holdener, A. T. (2008). Ajax: The Definitive Guide. Sebastopol, Ca.: O’Reilly and Associates.
3. Abrial, J. R. (2005). The B Book: Assigning Programs to Meanings. Cambridge, UK: Cambridge University Press.
4. http://www.SoftwareEngineering-9.com
5. http://www.pearsonhighered.com/sommerville

3 - ma’ruza. Dasturiy ta’minot dizaynini qurish va moslashtirish. Dasturiy ta’minotni testlash. Dastriy ta’minot evalyusiyasi. foydalanuvchi interfeysi

Reja:

3.1. Dasturiy ta’minot ishonchliligi.

3.2. Dasturiy ta’minot xavfsizligi.

3.3. Dasturiy ta’minot ishonchliligi va xavfsizligining xususiyatlari.

3.4. Dasturiy ta’minot xavfsizligining ehtimolligini boshqarish.

Kalit so’zlar: *Ishonchlilik, xavfsizlik, xususiyat, ehtimollik, boshqarish, mustahkamlik, himoyalanganlik, barqarorlik, xavf.*

3.1. Dasturiy ta’minot ishonchliligi

[9]Dasturiy ta'minot tizimlarining hajmi va murakkabligi oshib borgani sari, dasturiy injiniring sohasida uchraydigan eng muhim talab bu - biz tizimga ishonishimiz mumkinligini ta'milash ekanligi oydinlashmoqda. Biz biror tizimga ishonishimiz uchun bu tizim talab qilingan ishga mos kelishi va bu ishni to'g'ri bajarishi kafolatlanmog'i lozim. Buning ustiga tizim xavfsiz bo'lsin, ya'ni bizning PC larimiz yoki ma'lumotlarimiz bu tizim orqali xavf ostida qolmasin. Bizning ushbu ma'ruzamiz ishonchlilik va xavfsizlik borasidagi muhim ma'lumotlarni o'z ichiga oladi.

Hisoblash tizimlari shaxsiy hayotimiz hamda ishlarimizga chuqur kirib borgani sari tizim va dasturiy ta'minot nosozligi oqibatida kelib chiqadigan muammolar ham ortib bormoqda. Masalan, elektron tijorat bilan shug'ullanuvchi kompaniyaning serveri dasturiy ta'minotida paydo bo'lgan nosozlik ko'p miqdorda yillik daromad boy berilishi, mijozlarning yo'qotilishiga sabab bo'ladi.

Hozirda dasturiy ta'minot intensiv tizimlari hukumat, kompaniyalar va jismoniy shaxslar uchun juda ham zarur, shuning uchun keng qo'llanadigan dasturiy ta'minotlarga qo'yiladigan eng muhim talablardan biri bu ishonchlilik bo'ladi. Dasturiy ta'minot talab qilingan vaqtda javob berishi , vazifani to'g'ri bajarishi hamda autorizatsiyalanmagan ma'lumotlarni oshkor etilishi kabi ishning maqsadiga to'g'ri kelmaydigan ta'sirlardan yiroq bo'lishi kerak. 'dependability' ya'ni 'ishonchlilik' termini 1995 - yilda Lepray tomonidan tizimining tayyorlik, mustahkamlik, xavfsizlik va himoyalanganlik xususiyatlarini qamrab oluvchi atama sifatida taklif etilgan edi.

Quyidagi sabablarda ko'ra tizimlar ishonchliligi hozirda ularning barcha funsionalligidan ko'ra muhimdir:

[9] “Software Engineering”, by Ian Sommerville, pages 290-295

1. *Tizim nosozligi ko'p sondagi insonlarga zarar keltiradi.* Ko'pgina tizimlarning ichki funksionalligi kam ishlatiladi. Agar bu funksionalliklar ya'ni xizmatlardan biri tizimdan olib tashlansa oz miqdordagi foydaanuvchilar zarar ko'radi. Tizim yaroqliligiga zarar yetkazuvchi nosozlik esa bu tizimdan foydalanayotgan barcha iste'molchilarga zarar ketirishi mumkin. Nosozlik vaqtida normal ish yuritib bo'lmay qoladi.

2. *Foydalanuvchilar odatda mustahkam bo'lmagan, himoyalanmagan yoki xavfsiz bo'lmagan tizimlarni rad etadilar.* Agar foydalanuvchilar tizimni ishonchsiz va himoyalanmagan deb topsalar, uni ishlatishdan bosh tortadilar. Bu narsa esa keyinchalik ushbu tizimni ishlab chiqargan kompaniyaning boshqa mahsulotlariga nisbatan ham foydalanuvchilar ishonchining so'nishiga olib keladi.

3.*Tizim nosozligi juda qimmatga tushishi mumkin.* Yadro reaktorini nazorat qilish yoki airoplanlarni boshqaruvchi tizimlarga o'xshash tizimlarda paydo bo'ladigan nosozliklardan keladiga zarar, ularni boshqarishga sarflangan xarajatlar qiymatidan ham o'tib tushadi.

4. *Ishonchsiz tizimlar axborot yo'qotilishiga sabab bo'lishi mumkin.* Gohida hisoblash tizimiga joylashtirilgan ma'lumotlar shu tizimning o'zidan-da qimmat bo'ladi. Yo'qotilgan ma'lumotlarni qayta tiklash esa yanada qimmat turadi.

Ishonchli tizimni loyihalashda quyidagilarni e'tiborga olmoq lozim:

1. *Apparat ta'minotidagi nosozliklar*. Tizim apparat ta'minoti o'zining qurilishidagi xatolar yoki biror ehtiyot qismning o'z vazifasini o'tab bo'lgani sababidan nosozlikka duchor bo'ladi.

2. *Dasturiy ta'minot nosozligi.* Tizim dasturiy ta'minoti uning tavsifidagi, loyihasidagi yoki realizatsiyasidagi xatoliklar tufayli nosoz bo'lib qolishi mumkin.

3. *Faoliyatdagi nosozlklar*. Insonlar tizimdan to'g'ri foydalanishda va uni to'g'ri qo'llashda xato qilishlari mumkin. Apparat va dasturiy ta'minotlar ancha mustahka bo'lgan hozirgi davrda faoliyatdagi nosozliklar tizim nosozlilarining ko'pchiligini tashkil etadi deyish mumkin.

Bu nosozliklar ko'pincha bir-biriga bo'g'langan bo'ladi: nosoz apparat ta'minoti tizim operatorlariga qo'shimcha ishlar yuklashi ularni qiyin ahvolga tushirib qo'yishi mumkin. Bu narsa esa ulani asabiylashishiga sabab bo'ladi, insonning asabiylashganida xato qilishi esa tabiiy holdir. Nosoz dasturiy ta'minot bilan ham

shunday holatni kuzatish mumkin.

Hisoblash tizimi ishonchliligi - bu tizimga qanchalik ishonish mumkinligini o'zida aks ettiruvchi xususiyatdir. Bu bilan ishonchlilikni raqamlarda ifodalashni ko'zda tutmayapmiz. Balki nu o'rinda "ishonchsiz", "ishonchli", "juda ishonchli" kabi atamalar tzizm ishonchliligini aks ettirish uchun qo'llaniladi.

Ishonchlilikning to'rtta asosiy qismi bor:

1. *Tayyorlik.* Bu xususiyat tizim foydalanuvchi talab qilgan har qanday vaqtda oz' xizmatlarini taqdim eta olishidir.

2. *Mustahkamlik.* Bu xususiyat tizim o'ziga berilgan vazifani bexato, tavsiflarda keltirilganidek bajarishidir.

3. *Himoyalanganlik.* Bu tizim kishilarga yoki o'z muhitiga qanchalik ziyon yekazishi mumkinligini ko'rsatadigan xususiyat.

4. *Xavfsizlik.* Bu tizimning qasddan qilingan yoki tasodifiy tahdidlarga qanchalik qarshilik ko'rsata olishini aks ettiradigan xususiyatdir.

Bu asosiy qismlarga qo'shimcha ravishda quyidagi xususiyatlarni ham ishonchlilikning tarkibiga kiritish mumkin:

1. *Tuzatilish.* Tizim nosozliklari muqarrar hodisadir, lekin nosozlik natijasida kelib chiqqan buzilish agar tizimni tezda tuzatish imkoniyati bo'lsa minimallashtirilishi mumkin. Ochiq kodli dasturiy ta'minotlarda bu ish ancha oson, lekin komponentalarni qayta qo'llayverish buni qiyinlashtirishi mumkin.

2. *Qo'llab - quvvatlanish.* Tizim ishlatilgani sari unga yangi talablar qo'tib boriladi, shuning uchun talablar asosida tizimning yangi versiyalari ishlab chiqarilishi orqali uni qo'llab-quvvatlash muhimdir.

3. *Saqlanib qolish.* Bu Internetga asoslangan tizimlar uchun muhim xossadir. Saqlanib qolish bu - tizimning biror hujum ostida, hatto biror qismi o'chirib qo'yilganda ham ishda davom eta olish xususiyatidir. Albatta bunda minimal xizmat ko'rsata olish nazarda tutilmoqda. Saqlanib qolishni kuchaytirish uchun 3 ta strategiya qo'llaniladi - hujumga qarshilik qilish, hujumni aniqlash va hujum natijasida ko'rilgan ziyondan qayta tiklanish.

4. *Xatolarga chidamlilik.* Bunda ko'pincha foydalanuvchi xato ma'lumotlar kiritganida, iloji bo'lsa ularni tuzatish yo'qsa foydalanuvchiga bu haqdagi xabarni yetkazish tushiniladi.

3.2. Dasturiy ta’minot xavfsizligi

[10]Biz yuqorida xavfsizlik bu tizimning tashqi qasddan uyushtirilgan yoki tasodifiy hujumlardan o'znin himoyalay olish xususiyati ekanligi haqida so'z yuritgan edik. Bu tashqi hujumlar muqarrardir, chunki ko'pchilik kompyuterlar hozirda internetga ulanadi va bu bilan tashqi tomondan nishonga aylanishi hech gap emas.Bunday hujumlarga viruslar tushishi, tizim xizmatlaridan ruxsatsiz foydalanish, tizimga uatorizatsiyasiz ulanib uning ma'lumotlarini o'zgartirish kabilarni misol tariqasida keltirish mumkin. Agar siz haqiqatdan xavfsiz tizimda ishlashni xohlasangiz, unda yaxshisi internetga ulanamay qo'ya qoling. Shunda agar autorizatsiya qilingan foydalanuvchilar ishonchli bo'lsa sizning xavfsizlik bo'yicha muammolaringiz o'z yechimini topadi. Amalda esa katta tizimlar online rejimida ishlagani uchun yuqori darajada foyda ko'radilar, internetdan uzilish ular uchun daromadlarning keskin pasayishiga sabab bo'ladi.

Ko'pgina tizimlar uchun xavfsizlik bu ishonchlilikning asosiy mezonidir. Harbiy tizimlar, elektron savdo uchun yaratilgan tizimlar hamda o'ta maxfiy ma'lumotlarga ishlov berish bilan shug'ullanuvchi tizimlar yuqori darajada xavfsizlik ta'minlangan holda loyihalashtirilishi zarur. Masalan, agar havo tarnsportlariga chiptalarni buyurtma qiluvchi tizimda tayyorlik xususiyati past bo'lsa, bu ishonchning yo'qolishi hamda ba'zi chiptalardagi kechikishga sabab bo'lishi mumkin. Agar bu tizim xavfsizligi past bo'lsa unda hujum qiluvchilar unga kirib barcha buyurtmalarni o'chirib tashlashlari, buning natijasida esa normal havo yo'llari harakatlarini davom ettirishga imkoniyat bo'lmay qolishi mumkin. Ishonchlilikning boshqa qismlari kabi xavfsizlik ham o'zining maxsu atamalariga ega.

Pflegeer tomonidan muhim atamalar quyidagicha ta'riflanadi:

Mulk (asset) - himoyalanadigan va biror qiymatga ega bo'lgan narsa. Mulk bu dasturiy ta'minot tizimining o'zi yoki bu tizim tomonidan ishlatiladigan ma'lumot bo'lishi mumkin.

Zararlanish (exposure) - Hisoblash tizimi zaralanishi yoki undagi elementlar yo'qotilishi bo'lishi mumkin. Bunda zarar yoki yo'qotish ma'lumotlarda, vaqtda yoki xavfsizlik buzilganda keyingi tiklash ishlariga ketgan mehnatda ko'rinadi.

Zaif himoyalanganlik (vulnerability) - Hisoblash tizimi zaifligi, bundan foydalanib tizimga zarar yetkazilishi mumkin.

Hujum (attack) - Tizimning himoyasi zaifligidan foydalanib qolish. Odatda bu tashqi tarafdan bo'ladi va bunda zarar qasddan yetkazilishi nazarda tutiladi.

[10] “Software Engineering”, by Ian Sommerville, pages 302-305

Tahdidlar (threats) - zarar yetkazishi mumkin bo'lgan holatlar, vaziyat va sharoitlar. Bularga tizimga hujum uchun yo'l ochib beruvchi zaif himoyaga qaragandek qarash lozim.

Nazorat (Control) - tizim himoyasi zaifligini ketkazuvchi chora. Bunga shifrlashni misol qilib keltirish mumkin.

Ixtiyoriy tarmoqqa ulangan tizimda, uch xil asosiy xavfsizlikka qilinadigan tahdidlar uchraydi:

1. *Tizim va uning ma'lumotlari maxfiyligiga tahdidlar.* Bular axborotlarning autorizatsiyadan o'tmagan shaxslar yoki dasturlarga ochilishiga sabab bo'lishi mumkin.
2. *Tizim va uning ma'lumotlari sofligiga tahdid.* Bu tahdidlar dasturiy ta'minot yoki ma'lumotlarga zarar yekazishi, ularni buzishi mumkin.
3. *Tizim va uning ma'lumotlari tayyorligiga tahdidlar.* Bu tahdidlar autorizatsiyadan o'tgan foydalanuvchilarga ruxsatlarni chegaralab qo'yishi mumkin.

Albatta bu tahdidlar o'zaro ichki bog'lanishga ega Agar hujum tizim tayyorligiga zarar yetkazsa, unda siz vaqt o'tishi bilan o'zgarib turadigan axborotlarni yangilay olmaysiz. Bu o'z navbatida tizim sofligini yo'qqa chiqaradi. Shunday qilib zararlar bir - biriga ulanib ketadi.

Amalda, sotsialtexnik tizimlardagi ko'pchilik himoya zaifligi texnik muammolardan ko'ra ko'proq insonlarning xatolari natijasida paydo bo'ladi. Odamlar oson parollar tanlaydilar, yoki parollarini topib olish oson bo'lgan joylarga yozib qo'yadilar, tizim administratorlari ruxsatlarni belgilashda yoki fayllarni joylashtirishda xato qiladilar bundan tashqari foydalanuvchilar himoyalovchi dasturiy ta'minotlarni qo'llamaydilar.

Siz tizim xavfsizligini kuchaytirish uchun qo'yishingiz mumkin bo'lgan nazoratlar quyidagilardir:

1. *Himoya zaifligidan chetlanish.* Qilinayotgan hujumlar muvaffaqiyatsiz bo'lishiga ishonch hosil qilish uchun qo'yiladigan nazoratlar. Bu yerda strategiya tizimni xavfsizlikka oid muammolardan chetda loyihalashdan iborat. Masalan, harbiy tizimlar mahalliy tarmoqlarga ulanmagan bo'ladi, shuning uchun ularga tashqi kirish yo'llari berkdir. Ma'lumotlarni shifrlashni ham bu turdagi nazoratlarga kiritish mumkin. Shifrlangan ma'lumotga har qanday

autorizatsiyasiz kirishda, bu ma'lumot hujumchilar tomonidan o'qib bo'lmaydigan ko'rinishda bo'ladi. Amalda, kuchli shifrlangan ma'lumotlarni deshifrlash ko'p vaqt talab qiladi va qimmatga tushadi.

2. Hujumni aniqlash va uni bartaraf etish. Bu turdagi nazoratlar hujumlarni aniqlab ularni yo'q qilishga mo'ljallangan. Bu nazoratlar tizimda bajarilayotgan amallarni kuzatib turadi va g'ayrioddiy holatni aniqlaganda chora ko'radi: tizimning ushbu qismini o'chirib qo'yishi yoki aniqlangan foydalanuvchiga kirish yo'lini yopib qo'yishi mumkin.

3. Chegaralar qo'yish va tiklash. Bu nazoratlar muammolardan keyin qayta tiklanishni qo'llab-quvvatlaydi.

Talabga javob beradigan xavfsizliksiz, biz tizimning tayyorligi, mustahkamligi hamda himoyalanganligiga ishonolmaymiz.

Tizimni ishlab chiqarishdagi xatoliklar keyinchalik xavfsizlikni aylanib o'tilishiga olib kelishi mumkin. Agar tizim ko'zda tutilmagan kiruvchi parametrlarga javob bermasa yoki kiritilayotgan massiv ko'riniishidagi ma'lumotlarning chegarasi aniqlanmasa, hujumchilar bu zaifliklardan tizimga ruxsatsiz kirish uchun foydalanishlari mumkin. Asosiy xavfsizlik buzilish hodisalari ushbu zaifliklar orqali kelib chiqadi. C# tilida tuzilgan dasturlar massiv chegarasini tekshirishni o'z ichiga olmaydi, bu esa tizimga ruxsatsiz kirish orqali xotiraning biror qismini qayta yozishga imkon yaratadi.

3.3. Dasturiy ta’minot ishonchliligi va xavfsizligining xususiyatlari

[11]Tizimning ishonchliligi apparat ta'minot ishonchliligi, dasturiy ta'minot ishonchliligi hamda tizim operatorlari ishonchliligiga bog'liq. Tizim dasturiy ta'minoti bu yerda alohida o'rin tutadi. Bu o'z ichiga dasturiy ta'minot nosozligini qoplaydigan talablarni olish bilan birga operator hamda apparat ishonchlilik talablariga bog'liq bo'lib apparatdagi nosozliklar hamda operator xatolarini aniqlashda yordam berishi mumkin.

Ishonchlilik xavfsizlik hamda himoyalanganlikdan farqli ravishda tizimning o'lchasa bo'ladigan xususiyatdir. Tizim ishonchlilik darajasini tavsiflash mumkin, biror vaqt daomida tizim amallari kuzatib turiladi, hamda talab qilingan ishonchlilikka erishilgan bo'lsa bu belgilab qo'yiladi. Masalan, ishonchlilikka quyidagicha talab qo'yish mumkin: tizim qayta yuklanishiga sabab bo'ladigan tizim nosozligi haftada bir martadan ortiq ro'y bermasin. Aytilgan nosozlik har ro'y berganda siz bundan xabar topasiz hamda belgilab qo'yasiz, shunday qilib talab etilgan ishonchlilikka erishildimi yo yo'qmi bilib olasiz. Agar erishilmagan bo'lsa

[11] “Software Engineering”, by Ian Sommerville, pages 310-313

ishonchlilik talablarini qayta ko'rib chiqasiz yoki tizimdagi muammolarni tuzatishga kirishasiz. Siz past darajadagi ishonchlilikka ham rozi bo'laverishingiz mumkin, chunki ishonchlilikni oshirish uchun tizimga kiritiladigan o'zgartirishlar juda qimmatga tushishi mumkin.

Ishonchlilik talablari ikkiga bo'linadi:

1. Nofunksional talablar. Bular tizim normal ishlab turganida yoki tizim ishga tayyor bo'lmaganida qabul qilinishi mumkin bo'lgan nosozliklar sonini aniqlaydi.
2. Funksional talablar. Bular tizim va dasturiy ta'minotning dasturiy ta'minot nuqsonlarini chetlatish, aniqlash va ularga bardosh berish funksiyalarini aniqlaydi va bu nuqsonlar tizim nosozligiga olib kelmasligini ta'minlaydi.

Ishonchlilik talablari shunga bogliq bo'lgan funksional tizim talablariga yo'l ochadi. Biror darajadagi ishonchlilikka erishish uchun bu tizimning funksional hamda loyihaviy talablari aniqlanadigan xatoliklarni hamda ular tizim nosozligiga olib kelmasligini ta'minlash uchun ko'riladigan choralarni tavsiflamog'i lozim.

Umuman olganda tizim ishonchliligini o'sha tizim biror operatsion muhitda ishlatilganida tizim nosozligi ro'y berishi ehtimoli bilan tavsiflash mumkin. Maslan 1000 ta ixtiyoriy xizmatdan bittadida nosozlik ro'y bersa unda nosozlik ehtimolligi 0.001 bo'ladi. Albatta bu har 1000 ta amalda, aniq bitta nosozlik uchraydi degani emas. Bu agar siz 1000 * N ta amalni kuzatsangiz shunda nosozliklar soni N atrofida bo'ladi degan ma'noni anglatadi.

Ishonchlilikni tavsiflash uchun ikkita asosiy miqdordan va bunga qo'shimcha ravishda ishonchlilikka bo'gliq bo'lga xususiyat tayyorlikni tavsiflash uchun yana bitta miqdordan foydalaniladi:

1.*Talab qilingan nosozlik ehtimolligi - Probability of failure on demand (POFOD).* Agar siz bu miqdorni qo'llasangiz, unda tizim tomnidan biror xizmat uchun belgilangan tizim nosozligi ehtimolligini natija sifatida olasiz. Shunday qilib POFOD = 0.001 ifoda talab bajarilganida nosozlik ro'y berishi imkoniyati 1/1000 ga bo'lishini ko'rsatadi.

2. Nosozliklar sodir bo'lish darajasi - Rate of occurrence of failures (ROCOF) . Bu biror vaqt davomida yoki biror sondagi amallar bajarilish jarayonini kuzatish davomida qayd etilgan nosozliklar soni bilan belgilanadi. Masalan bir soatda ikkita nosozlik yuz bersa unda nosozlik yuz berish oralig'i yarim soat bo'ladi.

3.Tayyorlil -Availability(AVAIL). Tizimning tayyorligi so'rovlar

jo'natilganida xizmatlarni yetkazib berishida aks etadi. Masalan AVAIL = 0.9999 bu tizim har vaqt amallarni bajarishga 99.99% tayyor degani.

Tizimlarning xavfsizligiga qo'yiladigan talablar tavsifi bir jihatdan olib qaraganda himoyalanganlik talablari bilan umumiydir. Shunday bo'lsa ham xavfsizlik himoyalanganlikka qaraganda muhimroq muammodir. Buning sabablari quyidagicha:

1. Himoyalanganlikni olib qaraydigan bo'lsak, siz tizim o'rnatilgan muhitni "dushman" sifatida qaramasligingiz mumkin. Hech kim himoyalanganlik tomonidan muammo chiqarishga urinib ko'rmaydi. Ammo xavfsizlik tarafdan yondashuv mutlaqo boshqa natijaga olib keladi. Bunda tashqaridan tizimning zaif nuqtalaridan yaxshigina xabardor qandaydir g'arazli kimsa tizimga tashqi tomondan ta'sir o'tkazishga harakat qiladi.

2. Agar nosozlik himoyalanganlikdagi tavakkalchilikdan kelib chiqsa, siz nosozlikka sabab bo'lgan xatolarni va bo'shliqlarni ko'rishingiz mumkin. Tashqi hujumlar tizim nosozligini keltirib chiqarganida esa, ildizni topish qiyinlashib ketadi. Chunki hujumchilar nosozlik sababini yashirishga intiladilar.

3. Odatda tizimni o'chirib qo'yish, yoki uning biror xizmatlarini to'xtatsih himoyalanganlik buzilishi natijasida paydo bo'lgan nosozliklar uchun eng ma'qul yechimlardan hisoblanadi. Tashqi hujumlar esa ko'pincha tizimni o'chirib qo'yishga yo'naltirilgan bo'ladi.Tizim o'chirib qo'yilsa hujum muvaffaqiyatli yakunlanibdi deb hisoblayvering.

4. Himoyalanganlik bilan bo'gliq harakatlar aqlli "dushman" tomonidan amalga oshirilmaydi. Tashqi tomondan hujum qiluvhchi shaxs esa bir qancha tizimlarga hujum qilib tajriba orttirgan bo'lishi, tizim va uning javoblari haqida ega bo'lgan bilimlarini qo'llab o'z hujumlarini o'zgartirib turishi mumkin.

Yuqoridagilardan xavfsizlikka qo'yiladigan talablarrning himoyalanganlik talablariga nisbatan nechog'lik keng miqyosda bo'lishini ko'rishimiz mumkin. Qiyoslashlar natijasida, tizimga duch keladigan turli tahdidlarni o'z ichiga qamrab oluvchi bir necha tur xavfsizlik talablarini keltirish mumkin. Firesmith (2003 - yili) tizim tavsifida mavjud bo'lishi mumkin bo'lgan 10 ta xavfsizlik talablarini keltiradi:

1. Identifikatsita talablari. Tizim o'z foydalanuvchilari bilan muloqotga kirishishdan oldin ular identifikatsiya qilingan yo qilinmaganini tavsiflaydi.

2. Autentifikatsiya talablari. Foydalanuvchilar qanday identifikatsiyalanganini tavsiflaydi.

3. Avtorizatsiya talablari. Identifikatsiyalangan foydalanuvchining

imtiyozlari va kirish ruxsatlarini tavsiflaydi.

4. Qarshi turish (immunitet) talabari. Tizim viruslar, vormlar va shunga o'xshash tahdidlardan o'zninqanday himoyalashini ko'rsatadi.

5. Soflik talablari. Ma'lumotlar buzilishidan qanday saqlanish mumkinligini ko'rsatadi.

6. Ruxsatsiz kirishni aniqlash talablari. Tizimga qilinayotgan hujumlarni aniqlashda qanday mexanizm qo'llanishini ko'rsatadi.

7. Rad etilmaslik talablari. Biror xizmat a'zolaridan biri xizmatning o'ziga tegishli qismini inkor etmasligini ko'rsatadi.

8. Sir saqlash talablari. Ma'lumotlarni sir saqlash qanday qo'llab-quvvatlanishini ko'rsatadi.

9. Xavfsiz nazorat talablari. Tizimdan foydalanish qanday kuztilishi va tekshirilishi mumkinligini ko'rsatadi.

10. Tizimni qo'llab-quvvatlashning xavfsizligi talablari. Ilova qanday qilib xavfsizlik mexanizmidagi tasodifiy muvaffaqiyatsizlikdan so'ng autorizatsiyada yuzga keladigan o'zgarishlardan saqlanishini ko'rsatadi.

Albatta siz yuqoridagi xavfsizlik talablarning har birini barcha tizimlarda ham uchratavermaysiz. Bu talablarning qo'llanishi tizim turiga, undan foydalanish holatiga va undan foydalanishi mumkin bo'lgan iste'molchilarga bo'gliqdir.

3.4. Dasturiy ta’minot xavfsizligining ehtimolligini boshqarish

[12]Dasturiy ta'minot xavfsizligining ehtimolligini baholash va boshqarish samarali xavfsiz mexanizm qurishda juda muhimdir. Ehtimollikni boshqarish tizim mulkiga hujum natijasida kelib chiqishi mumkin bo'lgan yo'qotishlarni baholash hamda bu yo'qotishlarni ularni yo'qotishi mumkin bo'lgan xavfsizlik xizmati qiymati bilan muvozanatlashtirish ishlari bilan ham uzviy bog'liqdir. Kredit kartochka kompaniyalari buni har doim amalga oshiradilar. Kredit kartochkalardagi tovlamachiliklarni yo'qotish uchun yangi texnologiyalar ishlab chiqish nisbatan oson ishdir. Shunga qaramay, tovlamachilikka yo'l qo'ymaydigan tizimni sotib olish va uni o'rnatishdan ko'ra kompaniyalarga foydalanuvchilarining zararlarini qoplash uchun tovlamachilikdan ko'rilgan ziyonni qoplab berish arzonga tushadi. Yo'qotish va hujumlar qiymatiga qarab bu muvozanat buzilishi mumkin.

Ehtimollikni boshqarish texnik ishlardan ko'ra tadbirkorlik ishlariga yaqinroqdir. Dasturiy ta'minot muhandislari tizimda qanday nazoratlar o'rnatilishi

[12] “Software Engineering”, by Ian Sommerville, pages 394-397

haqida qaror qabul qilmaydilar. Xavfsizlik tizimi qiymatini qabul qilish yoki natijalarni xavfsizlik xizmatlaridan mahrum qilish bo'yicha qarorni yuqori menejment qabul qiladi. Shunga qaramasdan dasturiy ta'minot muhandislarining texnik yo'riqnomalar berish hamda xavfsizlik muammolarini hal qilish yo'llarini ko'rsatishdagi rollari beqiyosdir. Shuning uchun ham ular ehtimolli holatlarni boshqarish jarayonida asosiy ishtirokchilardan bo'lib qolveradilar.

Ehtimollikni baholash tizim qurilishidan oldin boshlanadi, tizim ishlab chiqarish jarayonida hamda tizim iste'molga chiqarilganda ham u davom etaveradi. Ehtimollikni baholash uch bosqichdan iboratdir:

1. *Dastlabki ehtimollikni baholash. Preliminary risk assessment.* Bu bosqichda hali tizimga qo'yilgan barcha talablar, tizim arxitekturasi yoki realizatsiyasi borasida hali qaror qabul qilinmagan bo'ladi.

2. Yashash siklidagi ehtimollikni baholash. Life-cycle risk assessment. Bu ehtimollikni baholash tizimni ishlab chiqish yashash sikli davomida amalda bo'ladi hamda tizimning texnik arxitekturasi va realizatsiya haqidagi qarorlardan ma'lumotlarni qabul qiladi.

3. Faoliyatdagi ehtimollikni baholash. Operational risk assessment. Tizim ishlab chiqarilgani va iste'molga chiqarilganidan so'ng, foydalanuvchilarga tizim qanday qo'llanishini ko'rsatish hamda yangi va o'zgargan talablarga mos ravishda takliflar kiritish uchun kerak. Faoliyatdagi talablarga doir taxminlar tizim noto'g'ri tavsiflanganda qilinadi. Tashkiliy o'zgarishlar tizim asl rejadan tashqari maqsadlarda qo'llanilayotganini ko'rsatadi. Faoliyatdagi ehtimolliklarni baholash tizim rivojlangani sari unga yangi xavfsizlik talablarini qo'yishga olib keladi.

Ehtimolliklarni baholash uchun sizi avvalo tizimga duch kelishi mumkin bo'lgan tahdidlarni aniqlab olishingiz lozim. Buni amalga oshirishning bir yo'li "noto'g'ri holatlar" (misuse cases) to'plamini ishlab chiqishdir. "Noto'g'ri holatlar" bu tizim bilan tizimga zararli bo'lgan o'zaro ta'sirlarga kirishdigan holatlardir. Siz bu holatlarni mumkin bo'lgan tahdidlarni aniqlash va ularni muhokama qilishda qo'llashingiz, binobarin bundan tizim xavfsizligiga qo'yiladigan talablarni ishlab chiqishda foydalanishingiz mumkin. Pflegeer tahdidlarni mumkin bo'lgan noto'g'ri holatlarni aniqlashda boshlang'ich nuqta bo'lishi mumkin bo'lgan to'rtta bo'lim ostida aks ettiradi. Bular quyidagilar:

1. Ko'rib olish hujumlari. Hujum qiluvchiga tizim va uning ma'lumotlariga kirish yo'lini ochadi.

2. Uzib qo'yish hujumlari. Bular hujum qiluvchilarga tizimning biror qismini

tayyorlik holatidan chiqarish imkonini beradi.

3. O'zgartirish hujumlari. Bu hujumlar natijasida hujum qiluvchi tizimning muhim ma'lumotlarini o'zgartirish imkoniyatiga ega bo'lishi mumkin.

4. Soxtalashtirish hujumlari. Bular hujum qiluvchiga tizim ichiga noto'g'ri axborotlar kiritish imkonini beradi.

Faoliyat davomida ehtimolliklarni baholash. Dasturiy ta'minot xavfsizligi ehtimolligini baholash va uni boshqarish, bu mahsulot yashash siklidan keyin ham davom etishi mumkin. Chunki vaqt o'tishi bilan yangi ehtimolli holatlar paydo bo'ladi va tizim ular bilan kurasha olishi uchun unga o'zgartirish kiritilishi mumkin. Mana shu jarayon faoliyat davomida ehtimolliklarni boshqarish deyiladi. Yangi ehtimolli holatlar tizimga qo'yilgan talablar o'zgarishi natijasida kelib chiqishi mumkin. Chunki tizimga qo'yilgan talablar o'zgarishi tizim infrastrukturasi o'zgarishiga yoki tizim qo'llanilayotgan muhitning o'zgarishiga sabab bo'ladi.

Faoliyat davomida ehtimolli holatlarni boshqarish, yashash siklida ehtimolliklarni boshqarishga o'xshaydi, biroq qo'shimcha ravishda tizim ishlatilayotgan muhit haqidagi axborotlarni ham o'z ichiga oladi. Muhitning xususiyatlarini bilish juda muhim ahamiyatga ega. Chunki ular yangi ehtimolli holatlarni keltirib chiqarishi mumkin.

Nazorat savollari

1. Tizim ishonchliligining asosiy olti asosiy xususiyatini ayting.
2. Tizimda uchrashi mumkin bo'lgan nosozliklar turlarini sanang.
3. Nima uchun mustahkamlik talablari oshgani sari tizim ishonchliligini ta'minlashning qiymati favqulodda tez ko'tarilib ketadi?
4. Tizim mustahkamligiga qanday erishiladi?
5. Mustahkamlik va xavfsizlik o'zaro bo'g'liq bo'lgan ishonchlilik xususiyatlaridir, ayni paytda ular alohida xususiyatlardir. Ular orasidagi eng muhim farqni bayon qiling.
6. Xavfning jiddiyligi qanday baholanadi?
7. Himoyalanganlik i qanday mezonlar asosida baholanadi?
8. Himoyalanganlikdagi tahdid va hujum tushunchalari qanday farqlanadi?
9. Ilova darajasidagi hamda infrastruktura darajasidagi himoyalanganlik bi-biridan qanday farq qiladi?
10. Tizimda yuzaga keladigan nosozliklarning katta qismi qaysi turdagi tizim nosozliklari tashkil etadi?

Foydalanilgan adabiyotlar

1. “Software Engineering”, by Ian Sommerville, 2015, pages – 790.

2. Holdener, A. T. (2008). Ajax: The Definitive Guide. Sebastopol, Ca.: O'Reilly and Associates.
3. Abrial, J. R. (2005). The B Book: Assigning Programs to Meanings. Cambridge, UK: Cambridge University Press.
4. http://www.SoftwareEngineering-9.com
5. http://www.pearsonhighered.com/sommerville

4 - ma’ruza. Dasturiy ta’minot xavfsizligi va ishonch-liligining xususiyatlari. Dasturiy ta’minot xavfsizligining ehtimolligini boshqarish.

Reja:

4.1. Loyihani boshqarish.
4.2. Loyihani rejalashtirish.

Kalit so’zlar: *Boshqarish, rejalashtirish, sarf harajat, talab, taklif, mijoz, biznes, nomoddiy, loyiha, baholash, bashorat, tahlillash.*

Ushbu ma’ruzaning maqsadi dasturiy ta’minotni boshqarishning ikkita asosiy turlari, ya’ni xavfni boshqarish hamda odamlarni boshqarish turlari haqida ma’lumot berish.

4.1. Loyihani boshqarish.

[13]Loyihani boshqarish dasturiy injiniringning muhim qismi hisoblanadi.Loyihani boshqarish aniq jadvallar asosida ketishi uchuin uni proffesional darajada amalga oshirish kerak bo’ladi. Bunda loyihani boshqaruvchinining o’rni katta .uning asosiy vazifasi loyihani bajarishda vujudga keladigan cheklovlarni aniqlash hamda bartaraf etishdan iborat.

Loyihani boshqarishni bir nechta muvvofiqiyatli mezonlari mavjud ular qo’yidagilar.

1. Loyihani belgilangan vaqt ichida mijozga yetkazish.
2. Umumiy narxni budjetga mos holda saqlash.
3. Mijozning talablariga mos holda loyihani bajarish .
4. Yaxshi faoliyt olib boruvchi guruh bilan ishlash.

Yuqorida berilgan mezonlar mezonlar dasturiy injiniringda yagona mezonlar hisoblanmaydi. Dasturiy ta’minotni boshqarish qiyin bo’lganligi uchun uni bajarishda bir nechta asosiy tushunchalarni bilish talab etiladi.

1. Dasturiy mahsulot nommodiy hisoblanadi:

Loyihani bajarishda menejer yo’lboshchi hisoblanadi.Agar loyihaning biron qismida kamchilik ko’zatilsa bu butun loyiha strukturasiga tasir etadi.

Dasturiy ta’minot nomoddiy bo’lganligi uchun uni ko’rib ham ushlab ham bo’lmaydi.Dasturiy ta’minot menejerlari loyiha qanday tarzda ketayotganligini oddiy kuzatish orqali aniqlay olmaydilar. Shuning uchun ular boshqa dalillarga

[13] “Software Engineering”, by Ian Sommerville, pages 594-595

tayanib jarayonlarni nazorat qiladilar.

2. Katta dasturiy loyihalar ko'pincha "one off" loyihalar hisoblanishadi.

Katta dasturiy loyihalar oddiy loyihalardan ancha farq qilishadi. Shuning uchun ushbu loyiha menejerlari bu loyihalarda duch kelishi mumkin bo'lgan muammolarni aniqlashga qiynaladilar. Komyuterlardagi texnik o'zgarishlar va ular o'rtasidagi aloqalar menejerning ishini kuchaytiradi.

3. Dastur jarayonlari o'zgaruvchan hamda aniq tashkillashtirilgan.

Muhandislik jarayonlari ba'zi bir sistemalarda xususan ko'prik hamda binolarning strukturasida tushunarli bo'lishadi. Lekin dasturiy jarayonlarni loyihlash bundan mustasno. Chunki hech bir menejer aniq bir dasturiy loyiha jarayonlarini rivojlantirishda qachon muammolarga duch kelishini bashorat qilishga qodir emas.

4.2. Loyihani rejalashtirish.

[14]Dasturiy loyihalar menejerlari uchun aniq standart vazifalar tasnifini keltirish mushkul.chunki menejerlarning ishlari tashkilotlar hamda dasturiy mahsulotlarning ishlab chiqishiga qarab o'zgaradi.
Shunga qaramasdan ko'p menejerlar qo'yida keltirilgan harakatlaning ba'zilariga yoki hammasiga javobgar bo'lishadi:

1. Loyihani rejalashtirish.

Loyiha menejerlari loyihani rtejalashtirish,ulani aniqlash hamda guruhdagi odamlarga o'zlarining vazifalarini bo'lib berishga javobgar bo'lishadi. Ular vazifalarni talablarga mos holda bajarilishini monitoring qilishadi va ishning belgilangan vaqt ichida borishi hamda budjetga mosligini nazorat qilishadi.

2. Hisobot berish.

Loyiha menejerlari bajarilayotgan ishlarni qay darajada ketayotganli haqida mijozga hamda kompaniya menejerlariga hisobot berishga majbur .

Ular texnik axborotlardan tortib boshqarish xulosalarigacha bolgan bosqichlarda ma'lumot berishga majbur. Ular bu ma'lumotlarni loyiha jarayonlari ketayotgan paytda berishlari shart.

3. Xavfni boshqarish.

[14] "Software Engineering", by Ian Sommerville, pages 619-621

Loyiha menejerlari loyihaga ta’sir ko’rsatishi mumkin bo’lgan xavflarni baholashlari , ularni monitoring qilishlari shuningdek muammo kuchayganda ular ustida ish olib borishlari kerak bo’ladi.

4. Odamlarni boshqarish.

Loyiha menejerlari lyihani bajarayotgan guruh odamlari ustidan nazorat olib borishlari kerak bo’ladi.

5. Taqdim yozish.

Dasturiy ta’minotni loyihalashning birinchi bosqichi loyiha bo’yicha shartnomani yutish uchun taqdim yozishga jalb qiladi. Taqdim o’z ichiga loyihaning maqsadi va uni qanday olib borilishini oladi.

Xavfni boshqarish

Xavfni boshqarish bu menejer uchun loyihani boshqarishdagi eng muhim vazifa hisoblandi. Xatarlar loyihaga tahdid soladilar.

Xatarlarni uchta turga bo’lib ko’rsatish mumkin:

1. Loyihadagi xatarlar.

Bular shunday xatarlarki, ular loyiha grafigiga hamda loyiha resurslariga ta’sir qiladilar.bunga misol qilib loyiha disaynerining yuqolishini keltirish mumkin. Agar dizayner ketsa uning o’rnig boshqa tajribali dizayner topishga ko’p vaqt ketadi.

2. Mahsulotdagi xatarlar.

Bu shunday xatarlarki, ular mahsulotning sifatiga hamda loyiha imkoniyatlariga ta’sir ko’rsatadi. Bunga misol qilib qabul qilinga komponentlarning tartibdan chiqib ketishini ko’rsatish mumkin.

3. Biznes xatarlar.

Bu shunday xatarlarki, ular tashkilotning dasturiy ta’minot jarayonlariga hamda ularni rivojlantirishlariga tasir ko’rsatadi. Bunga misol qilib raqobatchilarning shunga o`xshash yangi loyihani taqdim etishlari kiradi.

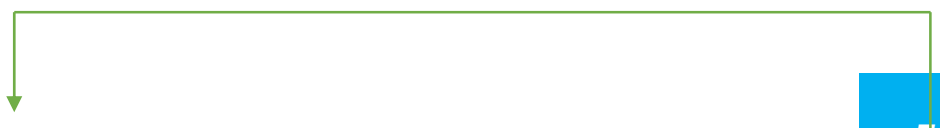
Loyiha menejerlardan loyihani rejalashtirishdagi xatarlar oqibatlarini tahlil qilish talab qilinadi. Xatarlarni effektiv boshqarish bu yuzaga keilishi mumkin bo’lgan xatarlarni aniqlash hamda ularni yechishning samarali yo’lini topishdir. Loyihaga tasir ko’rsatadigan shunday muayyan xavf-xatarlar borki, ular loyiha hamda dasturiy ta’minot yaratilayotgan tashkilot muhitiga bog’liq bo’ladi. Biroq shunday xavf-xatarlar borki ular yaratiulayotgan dasturiy ta’minotning turiga bog’liq bo’lmaydilar va ularga har qanday loyihalarda duch kelish mumkin bo’ladi. Bunday xavf-xatarlarning ba’zi ko’p uchraydigan ko’rinishlari qo’yidagi jadvalda

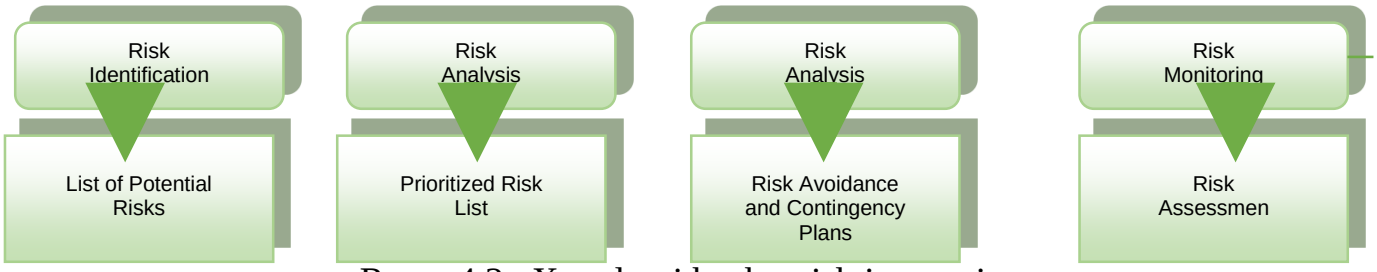
keltirilgan:
Loyiha,mahsulot va biznesga bo’lgan ko’p uchraydigan xatarlarga misollar.

Jadval-4.1.

Xatarlar	Ta’sir ko’rsatadi	Tasnifi
Xodimlar aylanmasi	Loyiha	Tajribali xodim loyihani tugashidan oldin tark etishi.
O’zgarishlarni boshqarish	Loyiha	Bu tashkiliy boshqarishning turli xil ustuvorlik bilan o’zgarishi
Apparat vositalarining yo’qligi	Loyiha	Loyihaga zarur bo’lgan apparat vositalarining oz vaqtida kelmasligi
Talablar o’zgarishi	Loyiha va mahsulot	Talablarga kutilgandan ko’p miqdorda o’zgarishlar kiritish
Xususiyatlarning kechikishi	Loyiha va mahsulot	Muhim inerfeyslarning xususiyatlarini jadvalda bo’lmasligi
Hajmni baholamaslik	Loyiha va mahsulot	Sistema hajmiga yetarlicha baho berolmaslik
“CASE” vositalarining qoniqarsizligi	Loyiha	Loyihaga yordam beradigan CASE vositalarining kutilgan darajada amalga oshirmasligi
Texnologiyalar o’zgarishi	Biznes	Dastur yozilgan texnologiyalarning yangi texnologioyalar bilan almashishi.
Loyihaga raqobat	Biznes	Raqobatbardosh mahsulotning loyiha tugashidan oldin bozorga chiqishi

Xaxf-xatar jarayonlarinin boshqarish chizmasi quyidagir rasmda keltirilgan:





Rasm-4.2. Xatarlarni boshqarish jarayoni.

Xatarlarni boshqarish jarayoni bir nechta bosqichlarni o'z ichiga oladi:

1. Xatarni aniqlash.(risk identification) -bo'lishi mumkin bo'lgan loyihaviy,mahsulot va biznes xatarlarni aniqlash.
2. Xatarni tahlil qilish.(risk analysis) – xatarlarning bo'lish ehtimolini hamda oqibatini baholash.
3. Xatarlarni rejalashtirish.(risk planning)- xatarlardan qutilish yoki ta'sirini kamaytirish uchun rejalar tuzish.
4. Xatarlarni monitoring qilish.(risk monitoring) xatarlarni nazorat qilish hamda bunga qarshio tuzilgan rejalarda kamchilik ko'zatilsa bartaraf etish.

Xatarlarni tahlillash

Xatarlarni aniqlash jarayonida oldindan har bir aniqlangan xatarlar uchun hukm ciqarilishi kerak bo'ladi. Xatarlarning sodir bo'lish ehtimoli va uning qanday darajada xavf tug'dirishini oldindan baholashi kerak bo'ladi.

Jadval - 4.3.

Xatarlarni boshqarish strategiyalari

XATARLAR	STRATEGIYALAR
Tashkiliy moliyaviy muammolar	Yuqori boshqaruvga loyihani maqsadga erishda qanday muhim hissa qo'shayotganligi hamda loyihaga berilgan mablag'ni qisqartirsa samarasiz bo'lishiga sabablar keltirilgan holda katalog hujjatlarni tayyorlash

Xodimlarni ishga yollash muammolari	Mijozga qiyinchiliklar hamda kechikishlar haqida xabar berish,komponentlarni sotib olishni nazorat qilish
Xodimlar betobligi	Xodimlarni bir-birini ishlarini tushunadigan qilib tashkillarshtirish
Nuqsonli qizimlar	Nuqsonli qismlarni yangi komponentlar bilan almashtirish
Talablar o'zgarishi	Talablar ta'sirini baholash hamda o'zgartirish uchun maksimal darajada keraksizma'lumotlarni kamaytirish
Tashkiliy qayta qurish	Yuqori boshqaruvga loyihani biznez yo'lioda maqsadga erishda qanday muhim hissa qo'shayotganligi haqida katalog hujjatlarni tayyorlash
Ma'lumotlar bazasi bilan ishlash	Yuqori darajada ishlovchi ma'lumotlar bazasini izlash
Loyiha yaratilish vaqtining qisqarilishi	Yangio qisimlar sotib olish hamda yaratuvchi dasturlar izlah

Yuqoridagi jadvalda xatarlarni boshqarish strategiyalari ko'rsatligan bo'lib, ularni uchta kategoriyaga bo'lish mumkin.

1. Oldini olish strategiyasi:

Bunda xatarlarning sodir bo'lish ehtimolini yo'q qiladi.misol qilib nuqsonli qismlarni almashtirish strategiyasini keltirish mumkin.

2. Minimallashtirish strategiyasi:

Bunda xatarlarning oqibatlari minimallashtiriladi.misol qilib bemor xodimlarning ishini boshqa wu ishni biladigan xodimga berib turishni keltirish mumkin.

3. Favqulotdagi rejalar:

Bu shuni ko'rsatadiki,xatarni oldini olish uchun har qanday narsaga tayor bo'ladi menejer.bunga misol qilib tashkiliy-moliyaviy muammolar ustidan qilinadigan strategiyalarni ko'rsatish mumkin.

Kelayotgan xatarlarga o’z vaqtida unga qarshi strategiya qo`llash talab etiladi. Loyiha menejeri ushbu loyiha yoki mahsulotga keladigan xatar oqibatlarini yaxshilashi kerak bo’ladi.

Jadval-4.4.

Xatarlar indikatori

Xatar turlari	Potensial ko’rsatkichlari
Texnologiya	Apparat vositalarining yoki yordamchi dasturlarning kechiktirilishi
Xodimlar	Xodimlarning moliyaviy kelib chiqishiga qarab munosabat qilishlari
Tashkiliy	Tashkiliy g’iybatlar; boshqaruvchilarning kam harakatda bo’lishi
Qurilmalar	jamoa a’zolarining Qurilmalarni ishlatishni istamasliklari
Talablar	Ko’p talablar so’roqlarni o’zgartirishi, mijozlar shikoyatlari
Baholash	Kelishilgan ish grafiginining buzilishi, aytilgan kamchiliklarni qilmaslik

Xodimlarni boshqarish

Dasturiy ta’minot yaratish tashkilotida ishlaydigan xodimlar tashkilotning muhim bo’lagi hisoblanishadi. Yaxshi xodimlarni ishga yollash tashkilotga qimmatga tushadi va bu masalani hal qilish menejerlarga bog’liq bo’ladi. Bundan tashqari loyiha menejerlari datsuriy ta’minot ishlab chiqishda texnik masalalarni tushunishlari kerak bo’ladi. Lekin shunda zo’r dasturiy ta’minot muhandislari borki, ularda yaxshi menejer chiqmaydi. Ular ko’chli texnik qobiliyatga ega bo’lishlariga qaramasdan ularda guruhda yetakchilik hamda guruh a’zolariga dalda berish qobiliyati bo’lmaydi.

Loyiha menejerlari xodimlarni boshqarish haqida xabardor bo’lishlari va ularni boshqarishni rivojlantirishi kerak bo’ladi.

Xodimlarni boshqarishning asosiy 4 ta omili mavjud:

1. **Mustahkamlik:**

Loyihadagi xodimlar barobar o’qitilishi kerak bo’ladi. Hech bir xodim erishiladigan yutuqlarni oziniki bo’lishini xohlamasligi hamda ularning ishlari rahbar tomonidan qullab quvatlab turilishi kerak.

2. Hurmat:

Har bir xodimda har xil qobiliyat bo’ladi va menejer bularni hurmat qilishi kerak bo’ladi.har bir guruh a’zosiga loyihaga o’z hissalarini qo’shish imkoniyatini berish kerak. ba’zi hollarda xodimlarni ushbu loyihaga mos emasligini menejerlar sezishadi, shunda ulardan astalik bilan ish ko’rishlari talab etiladi.

3. Hisobga olish:

Xodimlarning loyihaga samarali hissa qushishlari boshqa guruh a’zolarini ularni tinglashlari va bergan takliflarini hisobga olishlariga bog’liq bo’ladi. Ishchi muhitinio rivojlantirish judayam muhim hisoblanadi.

4. Haqqoniylik:

Menejerlar guruhda kimning ishi yaxshi ketayotgani yoki yomon ketayotganini haqqoniylik bilan aytishi kerak.bundan tashqari menejer o’zining qanday texnik darajada bilimga ega ekanligini bilib, undan yuqori malakali xodimlar bilan bilim almashishi kerak.

Menejer o’z xodimlariga motivatsiya berish qobiliyatiga ega bo’lishi kerak.motivatsiya berish bu ishni boshqarish va odamlarga ishchi muhitini yaxshi tashkil qilish hisoblanadi. Buning natijasida ishchilar ozlarining eng yaxshi ishlarini ko’rsatadilar.

Xodimlarga yaxshi ishchi muhitini yaratish uchun menejer xodimlarga nma motivatsiya berishi mumkinligini bilishi talab qilinadi. Odamlar ozlariga kerak narsalarni ularga yetkazilib berishidan qanoatlanadilar.xodimlarning nimalarga muhtoj bo’lishlarini quyidagi rasmdan ko’rish mumkin:

Chizma-4.5. Xodimlarning ehtiyojlari ierarxiyasi

Dasturiy ta’minot ishlab chiqish tashkilotida ishlovchi xodimlarga odatda och qolmaslik uchun fiziologig tahdidga tushmaslik uchun muhit yaratib berishadi.

Guruh a’zolarini tanlash:

Menejerlar yoki guruh yetakchilarining ishi samarador guruh yaratsih hamda guruhni birgalikda effektiv ishlashlarini ta’minlashdan iborat. Bu xodimlarnining xarakterlari va texnik mahoratlarini bir maromda ushlashga yordam beradi.

Jamoaviy Munosabatlar

Guruh a’zolarining bir birlari bilan va boshqa loyiha aksiyadorlari(stackeholder) effektiv va samarali munosabatda bolishlari judam muhim.Guruh a’zolari loyihaga doir ma’lumotlarni ,loyiha dizayni bo’yicha takliflarni bir-birlari bilan almashishlari kerak. Xodimlar o’z navbatida sheriklarida loyiha bo’yicha muammolar to’gilsa ularni bartaraf etishda ko`maklashishlari kerak. Bundan tashqari guruh a’zolari boshqalarning kuchli va kuchsiz tomonlarini bilishlari kerak bo’ladi.

Effektiv va samarali munosabatlarga quyidagilar ta’sir qiladi.

1. **Guruhning hajmi.**

Guruh qanchalik kattalashsa guruh a’zolarining bir-birlari bilan samarali munosabat qo’rishlari shunchalik qiyinlashadi. Guruh a’zolarining har birlari boshqalar bilan bir martadan aloqa qilishlari soni n*(n+1) formula orqali topiladi.masalan 8 ta a’zosi bo’lgan xodimlarning bir martadan o’zaro munosabatda bo’lishlari soni 56 taga teng bo’ladi.

2. Guguh tuzulmasi.

Norasmi tuzulgan guruh a’zolari rasmi tuzilga guruh a’zolariga nisbatan effektiv aloqada bo’lishadilar. Turli tizimlarda ishlaydigan xodimlar bir-birlari bilan faqat menejer orqali ma’lumot almashadilar bu esa tushinmovchiliklarga olib kelishi mumkin.

3. Guruh tarkibi.

Bir xil toifadagi xodimlar bir birlari bilan tuqnash kelishlari mumkin va natijada munosabatlar buzilishi kuzatiladi. Shuning uchun xodimlar tarkibi ayollar hamda erkaklar dan tashkil topishi kerak.

4. Jismoniy ish muhiti.

Tashkilotning xodimlar ishlashlari uchun yaratilgan muhiti munosabatlarni yaxhilash yoki yon to’sqinlik qilishga asosiy omil hisoblanadi.

5. Aloqa kanallarining mavjudligi.

Turli xil aloqa qilish usullari mavjud. Bular yuzma-yuz aloqa, e-mail xabarlar orqali aloqa, rasmiy hujjatlar orqali, telefon bo’lishi mumkin.

Agar loyiha a’zolarini ajratilgan holda ishlashlariga to’gri kelsa bunda ularga aloqani yengillashtiradigan texnologiyalar kerak bo’ladi.

Nazorat savollari

1. Nima uchun doim ham eng yaxshi dasturchi eng yaxshi dasturiy ta’minot menejeri bo’la olmaydi?
2. Doimiy narx(fixed-price) nima?
3. Dasturiy loyiha plani har bir qismini tushuntiring.
4. Narx smetasini kamaytira olishi mumkin bo’lgan to’rtta yo’l ayting.
5. Dasturiy ta’minotni ishlab chiqishda tezkor(agile) va yetuk(maturity) yondashuvlar orasidagi muhim farqlarni ayting.
6. Dizayn sifatini bashorat qiluvchi qanday metodlar mavjud?

Foydalanilgan adabiyotlar

1. “Software Engineering”, by Ian Sommerville, 2015, pages – 790.
2. Holdener, A. T. (2008). Ajax: The Definitive Guide. Sebastopol, Ca.: O’Reilly and Associates.
3. Abrial, J. R. (2005). The B Book: Assigning Programs to Meanings. Cambridge, UK: Cambridge University Press.

IV. BO`LIM

AMALIY MASHG`ULOT MATERIALLARI

IV. AMALIY MASHG'ULOT MATERIALLARI

1 - amaliy mashg'ulot. UML muhitida ishlash uchun kerakli dasturiy vositani o'rnatish. Loyiha uchun UML holat diagrammalarini shakllantirish.

Ishdan maqsad: Loyihaning holat diagrammalarini shakllantirish uchun UML muhitida ishlash uchun StarUML dasturiy vositasini o'rnatish, loyiha tasnifi uchun use case, class, sequence, activity va boshqa holat diagrammalarini yaratish ko'nikmalarini hosil qilish.

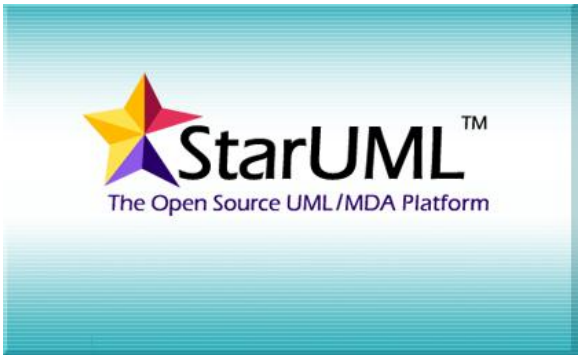
Ishni bajarish tartibi

1. StarUML dasturini o'rnatish.
2. StarUML dasturida holat diagrammalrini yaratish uchun ishlatiladigan uskunalar paneli bilan tanishish.
3. Loyihaning use case diagrammasini shakllantirish bo'yicha ko'rsatmalar.
4. Loyihaning class diagrammasini shakllantirish bo'yicha ko'rsatmalar.
5. Loyihaning sequence diagrammasini shakllantirish bo'yicha ko'rsatmalar.

StarUML dasturini o'rnatish

StarUML – bu ochiq kodli, qulay va tezkor UML/MDA platforma bo'lib, loyihaning holat diagrammalarini shakllantirishda ishlatiladi.

- Asosiy xususiyatlari:
  - UML 2.0
  - MDA
  - Plug-in Architecture
  - Usability



1-rasm. StarUML dasturi dastlabki interfeysi

UML (Unified modeling language) – OMG (Object Management Group) ning kengaytirilgan ko'rinishi bo'lib, hozirgi kunda UML 2.0 versiyasi mavjud va barcha

standartlari StarUML dasturi tomonidan qo'llab quvvatlanadi.

MDA (Model Driven Architecture) - OMG (Object Management Group) ning yangi texnologiyasi bo'lib, dasturiy ta'minotni modellashtirish uskunasi o'zgaruvchilar va metodlar kiritishni ham qo'llab quvvatlaydi.

- **Afzalliklari**
  - import va export qilish imkoniyati mavjud.
  - Dasturda kodlar va dokumentatsiyalar mavjud.
  - foydali va bepul.
- **Kamchiliklari**
  - Boshlang'ichlar uchun qiyin.
  - Kross platformali emas. (faqat Windows uchun ishlaydi)

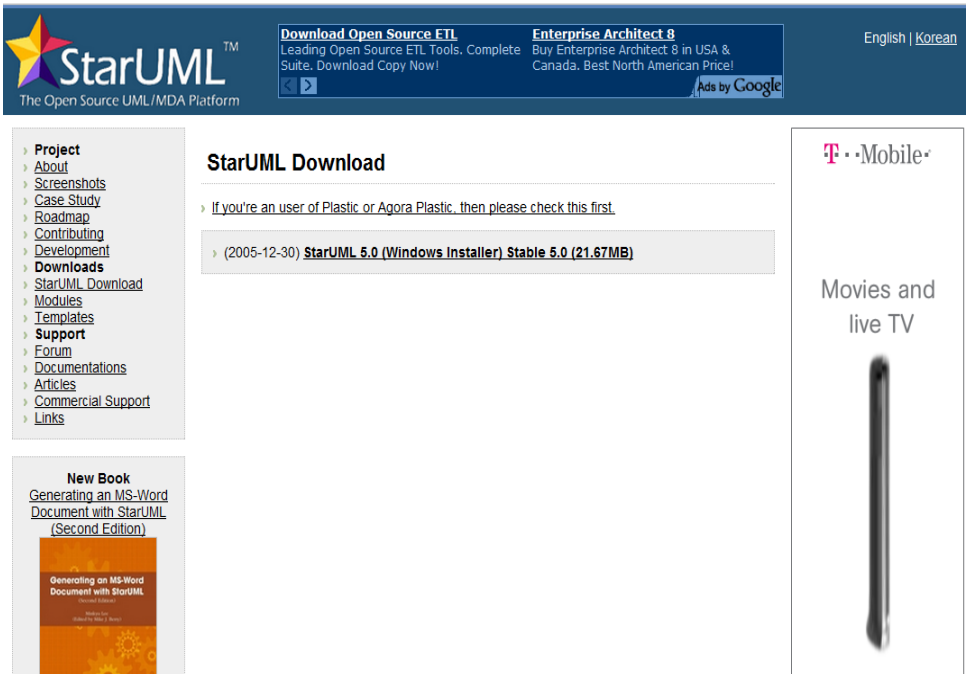
StarUML dasturidan foydalanish uchun tizimga minimum quyidagicha talablar qo'yiladi:

- Intel® Pentium® 233MHz yoki undan yuqori
- Windows® 2000, Windows XP™,yoki undan yuqori
- Microsoft® Internet Explorer 5.0 yoki undan yuqori
- 128 MB RAM (256MB tavsiya qilinadi)
- hard diskdan 110 MB bo'sh joy (150MB tavsiya qilinadi)

StarUML loyiha uchun quyidagi ko'rinishdagi diagrammalarni yaratish uchun imkoniyatini yaratadi:

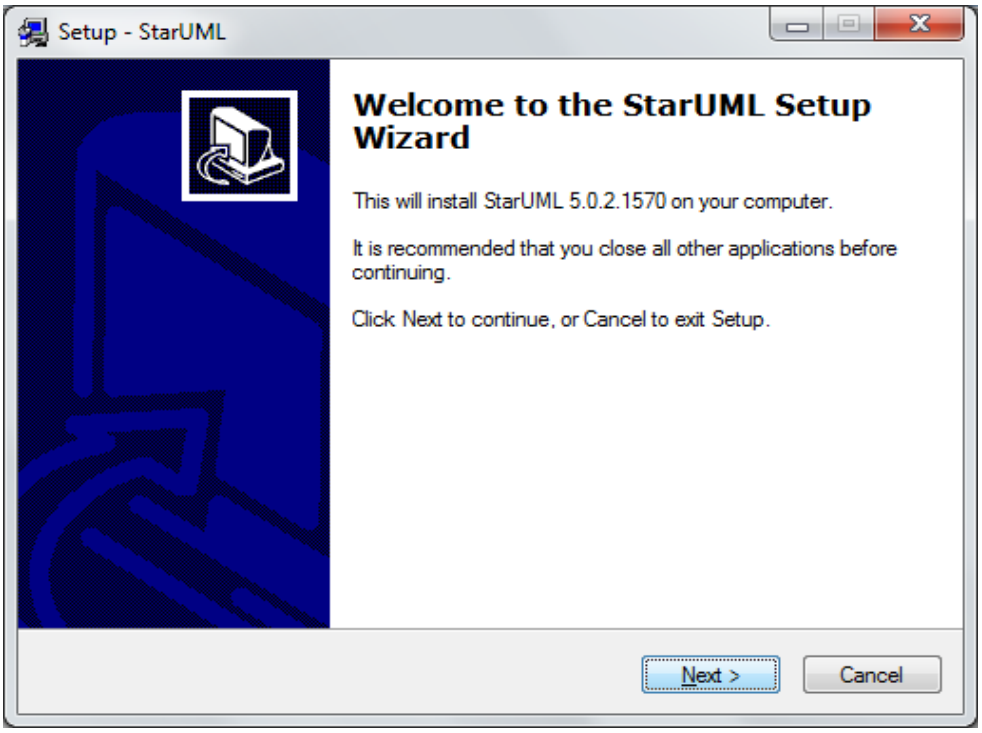
1. Use Case Diagram
2. Class Diagram
3. Sequence Diagram
4. Collaboration Diagram
5. State chart Diagram
6. Activity Diagram
7. Component Diagram
8. Deployment Diagram
9. Composite Structure Diagram (UML 2.0)

Dasturni o'rnatish uchun quyidagi ssilkaga kiramiz
http://staruml.sourceforge.net/en/download.php



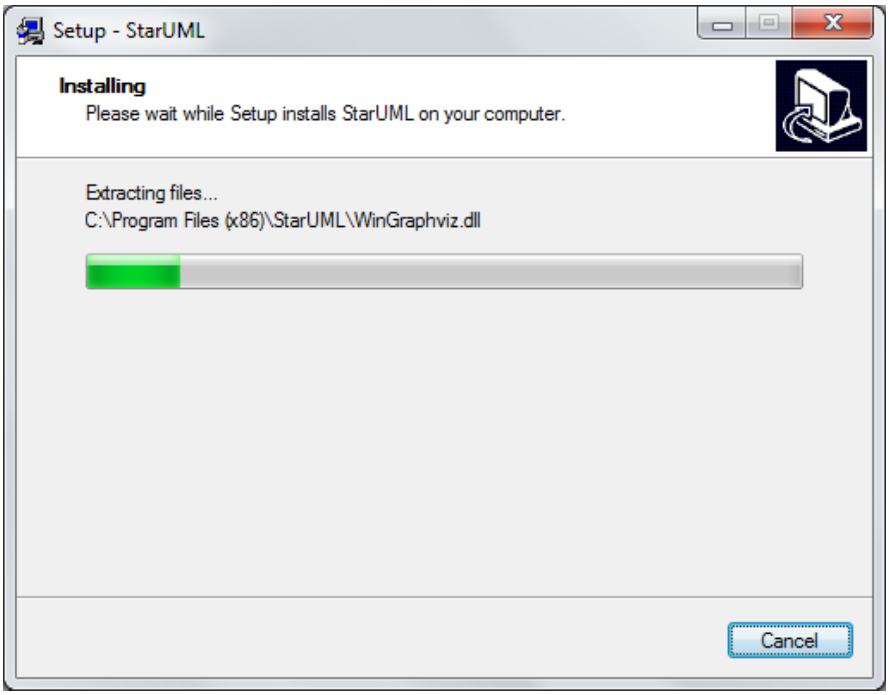
2-rasm. StarUML dasturini yuklab olish uchun web sahifa oynasi

Yuklab olingan faylni ishga tushirganimizda quyidagicha oyna xosil bo'ladi va undan "Next" tugmasini bosamiz.

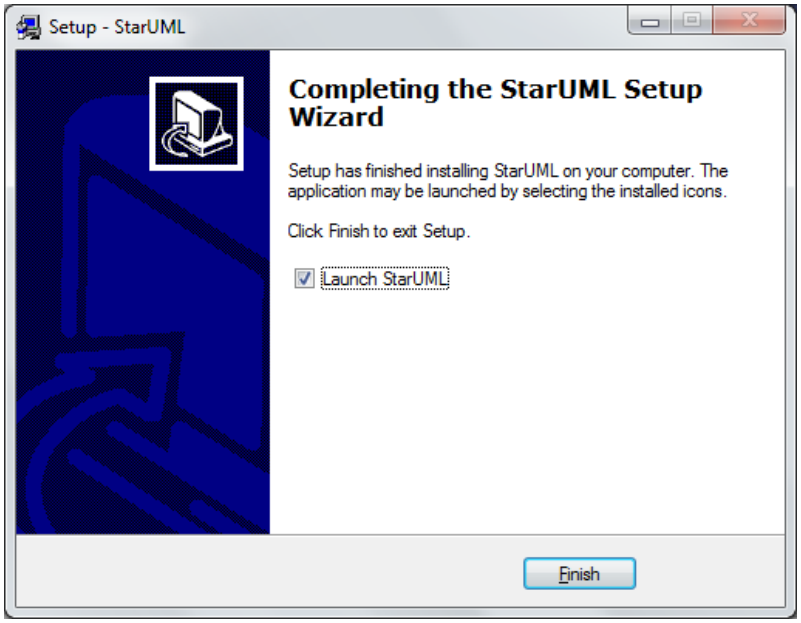


3a-rasm. StarUML dasturining o'rnatilish jarayoni

Natijada dasturni qayerga o'rnatishni yo'lini ko'rsatib (Odatda C:\Program Files (x86)\StarUML papkasiga o'rnatiladi) "Next" tugmasini belgilaymiz.



3b-rasm. StarUML dasturining o'rnatilish jarayoni

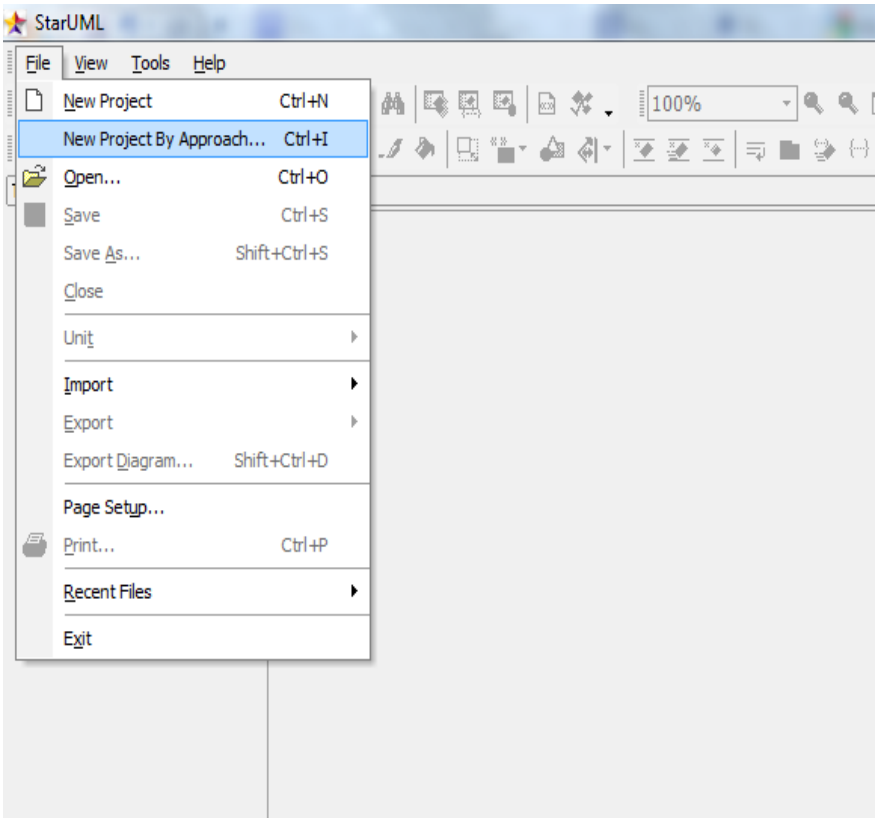


3c-rasm. StarUML dasturining o'rnatilish jarayoni tugashi va uni ishga tushurish

Agar StarUML dasturi muvoffaqiyatli o'rnatilsa yuqoridagi 3c-rasm paydo

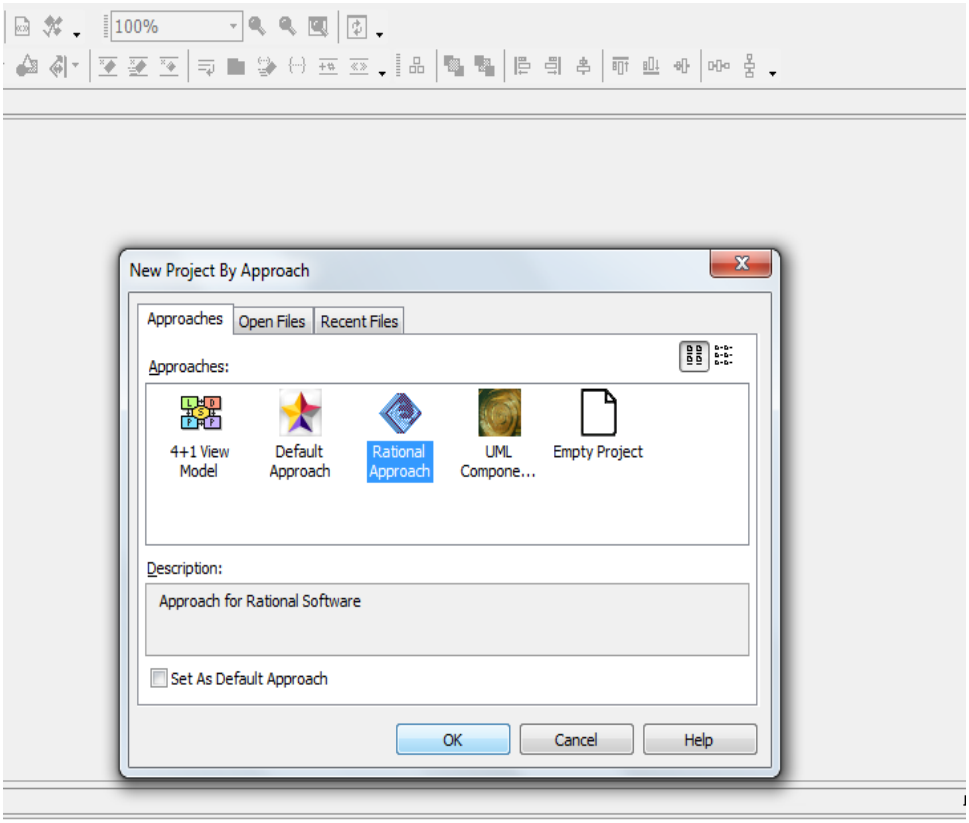
bo'ladi va bundan "Finish" tugmasini bosadigan bo'lsak, dastur o'rnatilishi nihoyasiga yetadi va StarUML dasturi avtomatik tarzda ishga tushuriladi. StarUML da loyiha yaratish ketma-ketligi quyidagicha amalga oshiriladi.

[File] -> [New Project By Approach] menu.

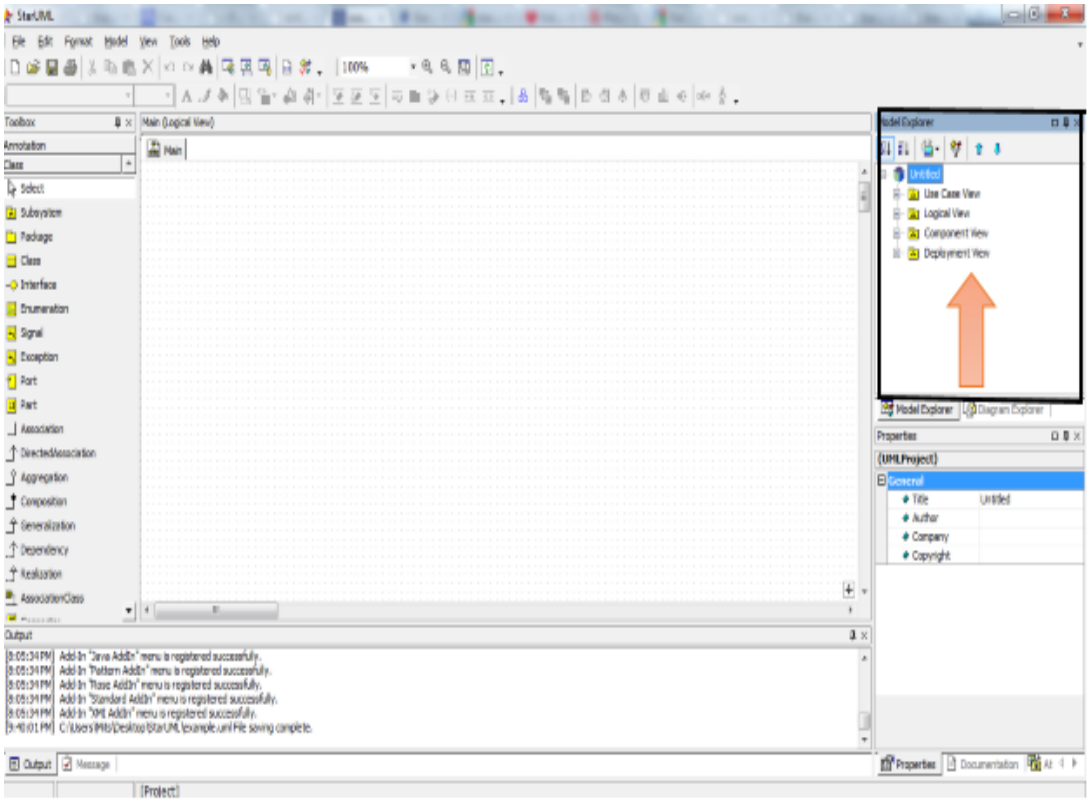


4a-rasm. StarUML dasturida yangi loyiha yaratish

Select **[Rational Approach]**



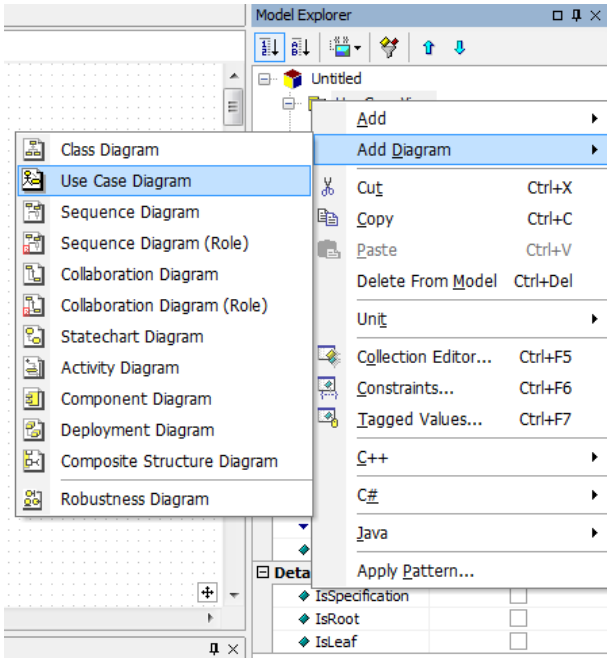
4b-rasm. StarUML dasturida yangi loyiha yaratish



5-rasm. Yaratilgan loyihaning umumiy oynasi

Loyihaning Use case diagrammasini chizish. Buning uchun quyidagi ketma-ketlik amalga oshiriladi

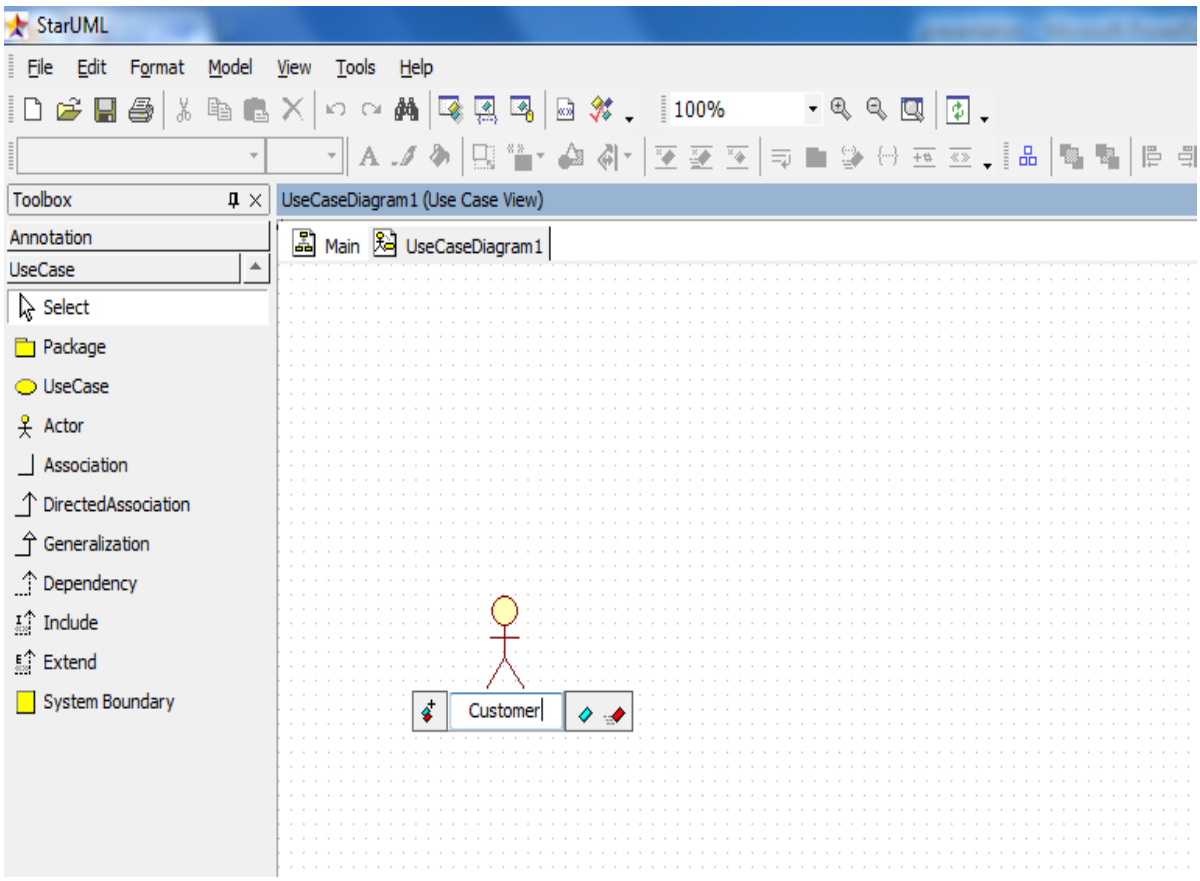
- Yuqorida ko'rsatilgan rasmdan **Use Case view** bo'limiga sichqoncha chap tugmasi ikki marta bosiladi.
- "**Main**" bo'limida to'g'ridan to'g'ri diagrammani chizish mumkin (yoki **Use Case View** menuyusidan sichqoncha o'ng tugmasini bosib **[Add Diagram] -> [Use Case Diagram]** ketma-ketliklari tanlanadi.



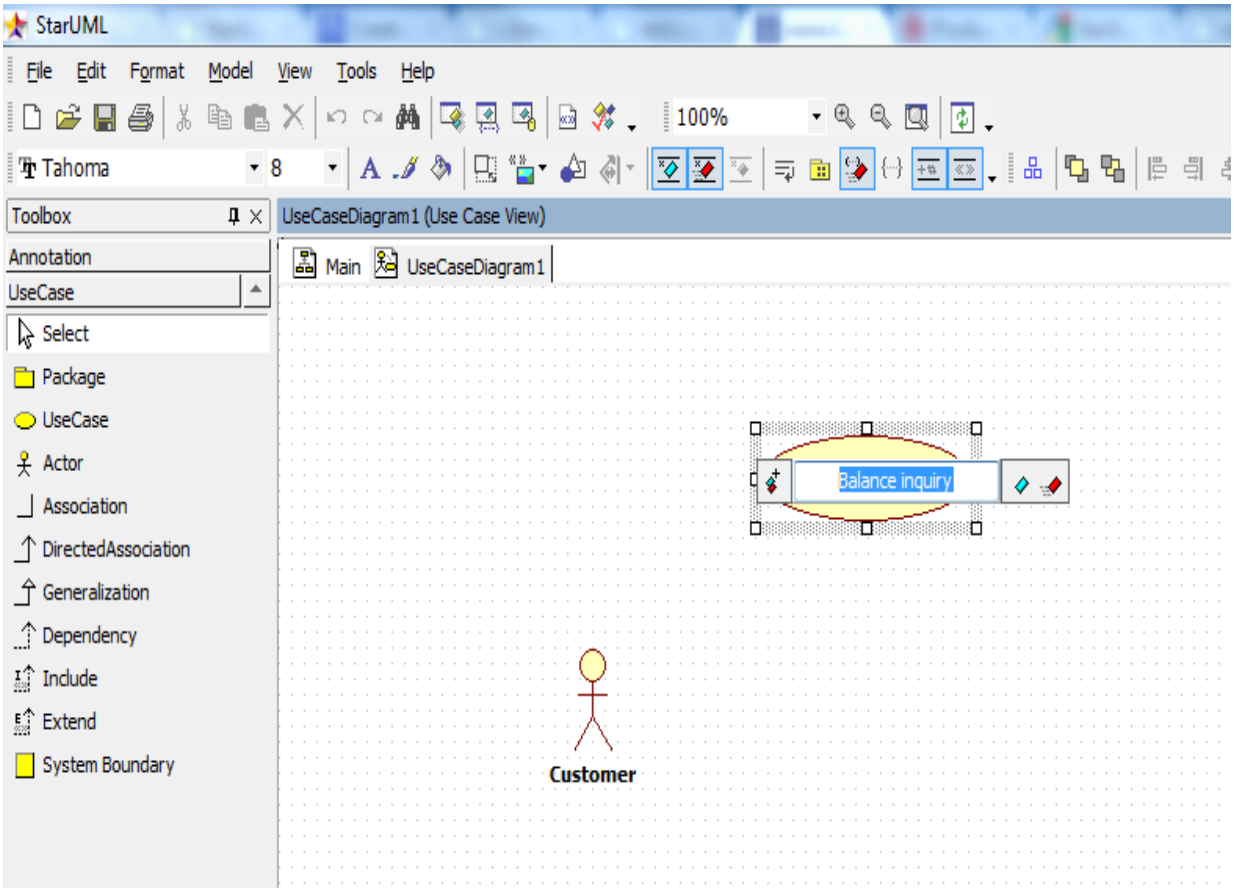
6-rasm. Loyiha uchun Use Case diagrammasini yaratish

Use case diagrammasini shakllantirish jarayonida quyidagi elementlardan foydalaniladi:

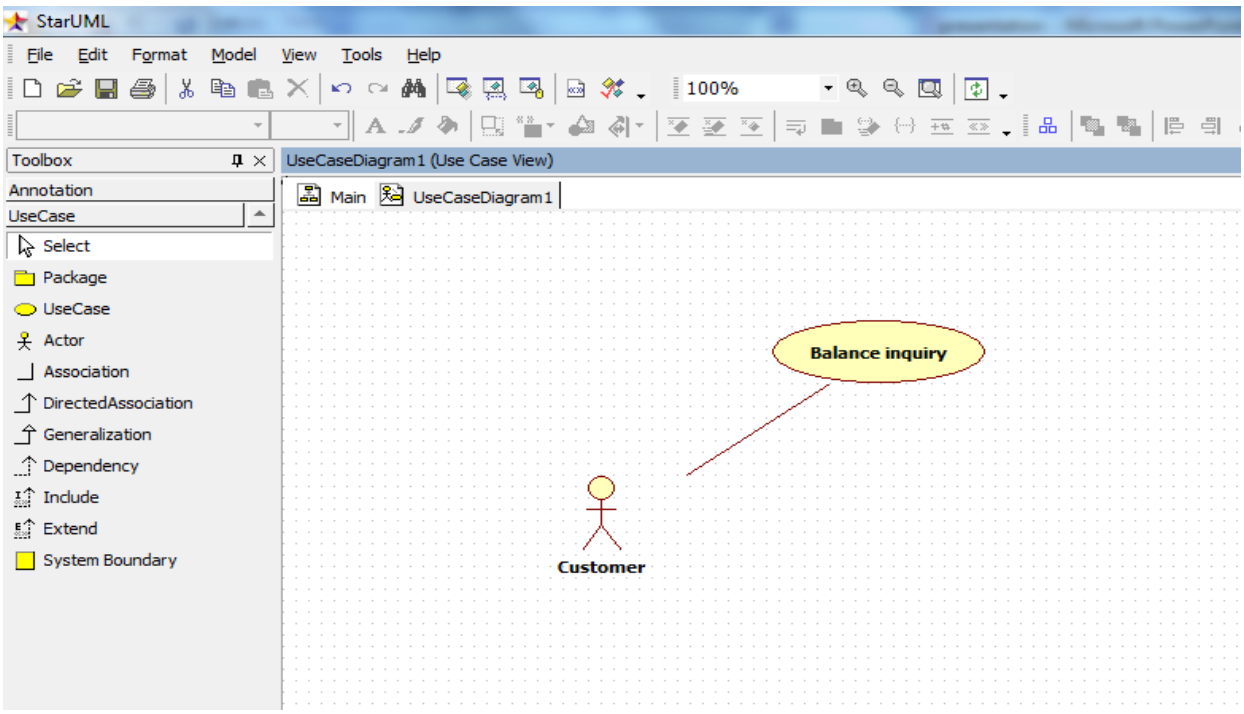
- Actor
- Use Case
- Association
- Directed Association
- Generalization
- Dependency
- Include
- Extend
- System Boundary
- Package



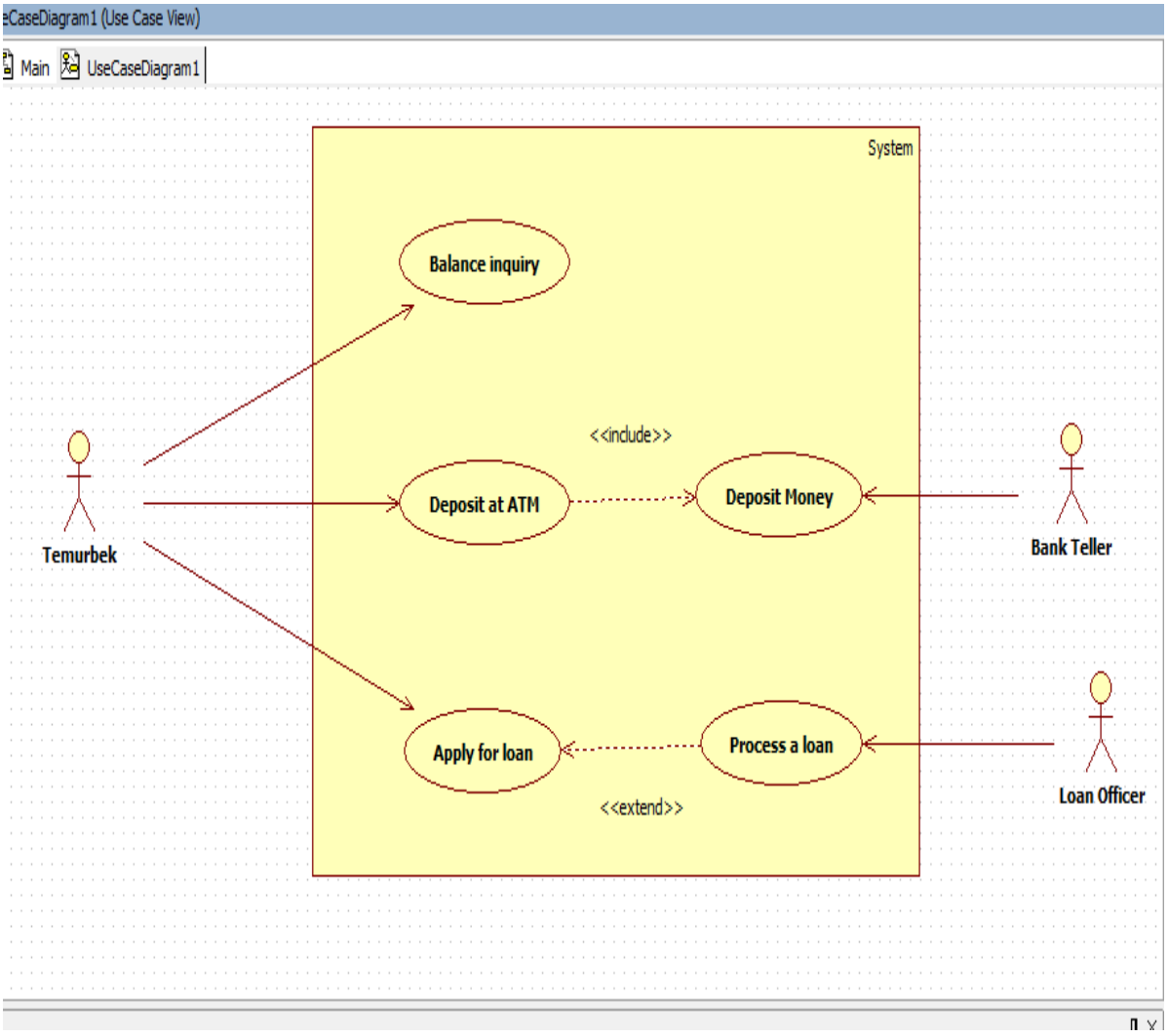
7a-rasm. Actor elementi



7b-rasm. Use case elementi



7c-rasm. Assotsatsiya (komponentalarni bog'lash) elementi



8-rasm. Loyihaning Use case diagrammasi

Foydalanilgan adabiyotlar

1. "Software Engineering", by Ian Sommerville, 2015
2. UML Distilled: A Brief Guide to the Standard Object Modeling Language (3rd Edition)by Martin FowlerSep 25, 2003
3. http://www.SoftwareEngineering-9.com
4. http://www.uml-diagrams.org/

2 - amaliy mashg'ulot. Loyiha uchun tizim talablari va funksional talablarni ishlab chiqish.

Ishdan maqsad: Ishlab chiqarilayotgan dasturiy ta'minot uchun talablar hujjatini shakllantirish.

Tizim uchun talablar bu tizim nima ish bajarish lozimligini tasvirlashdir. Talablar tizim mijozlarini ehtiyojlarini aks ettiradi.

Talablar injiniringi jarayonida ko'pgina muammolar ko'tariladi. 'Foydalanuvchi talablari' va 'tizim talablari' terminlari orasida farq mavjud. Foydalanuvchi talablari umumiy va tizim talablari batafsil bo'ladi. Foydalanuvchi talablari va tizim talablari quyidagicha izohlanishi mumkin:

3. Foydalanuvchi talablari bu diagrammalar bilan tabiiy tildagi bayonotlar.
4. Tizim talablari bu dasturiy ta'minot tizimi funksiyalari, servislari va operativ cheklanishlarining batafsil tasvirlanishi.

Ushbu amaliy ishni tushuntirish maqsadida biz sizga Ruhiy Bemorlar Sog'ligini Saqlash Boshqaruv Tizimi (RBSSBT) misol qilib olamiz.

Foydalanuvchi talablari

1. RBSSBT belgilangan har bir shifoxonalar uchun dorilar

Tizim talablari

1.1. Har oyning oxirgi ish kunida belgilangan dorilar qisqacha izohi, ularning bahosi va belgilangan shifoxonalar hosil qilinadi

1.2. Tizim automatik tarzda har oyning oxirgi ish kunida soat 17:30 dan keyin hisobotni bosmaga chiqarishga tayyorlaydi

1.3. Hisobot har bir shifoxona uchun yaratilgan bo'lishi va individual dori nomlari ro'yxati, shifokorlar yozib bergan dori ro'yxatilari soni, yozib berilgan dori ro'yxatidagi dori dozalari soni yozib berilgan dorilarning umumiy bahosi

1.4. Agar dorilar turli xil dozalarda(masalan, 10mg, 20mg) bo'lsa ularni alohida ajratib hisobot tayyorlash

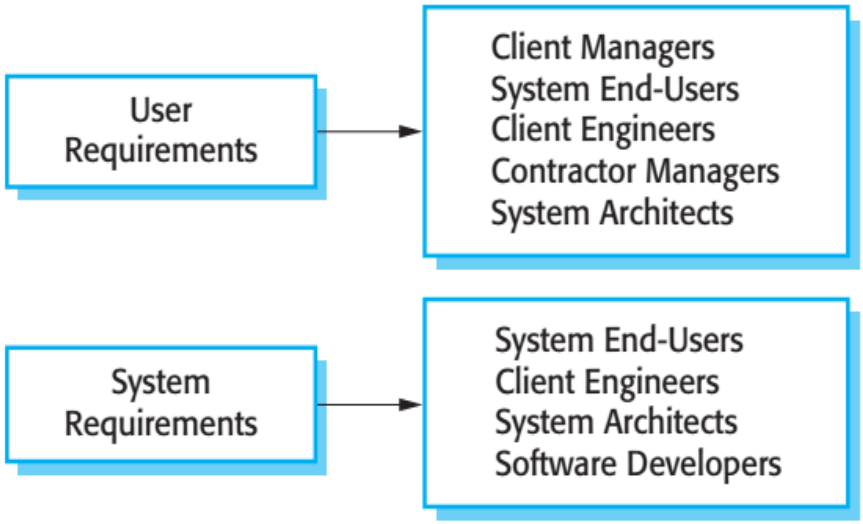
1.5. Hisobotga ruxsatlarni chegaralash

Dasturiy ta'minot tizimi talablari funksional va funksional bo'lmagan talablar sinflariga ajratiladi.

3. Funksional talablar Bu tizim taminlashi lozim bo'lgan servislarning bayonoti. Kiritilgan ma'lumotlarga tizim qanday reaksiya ko'rsatishi lozim, tizim o'zini bunday holatlarda qanday tutushi lozim

4. Funksional bo'lmagan talablar Bu tizim tomonidan taklif qilinayotgan servislar va funksiyalardagi cheklovlar. U o'z ichiga vaqt cheklanishi, ishlab chiqarish jarayoni cheklanishi, beriladigan standartlar tomonidan cheklanishlarni olishi mumkin.

Siz talablarni turli xil darajada yozishingiz kerak chunki turli xil o'quvchilar turli xil yo'lda foydalanishadi.



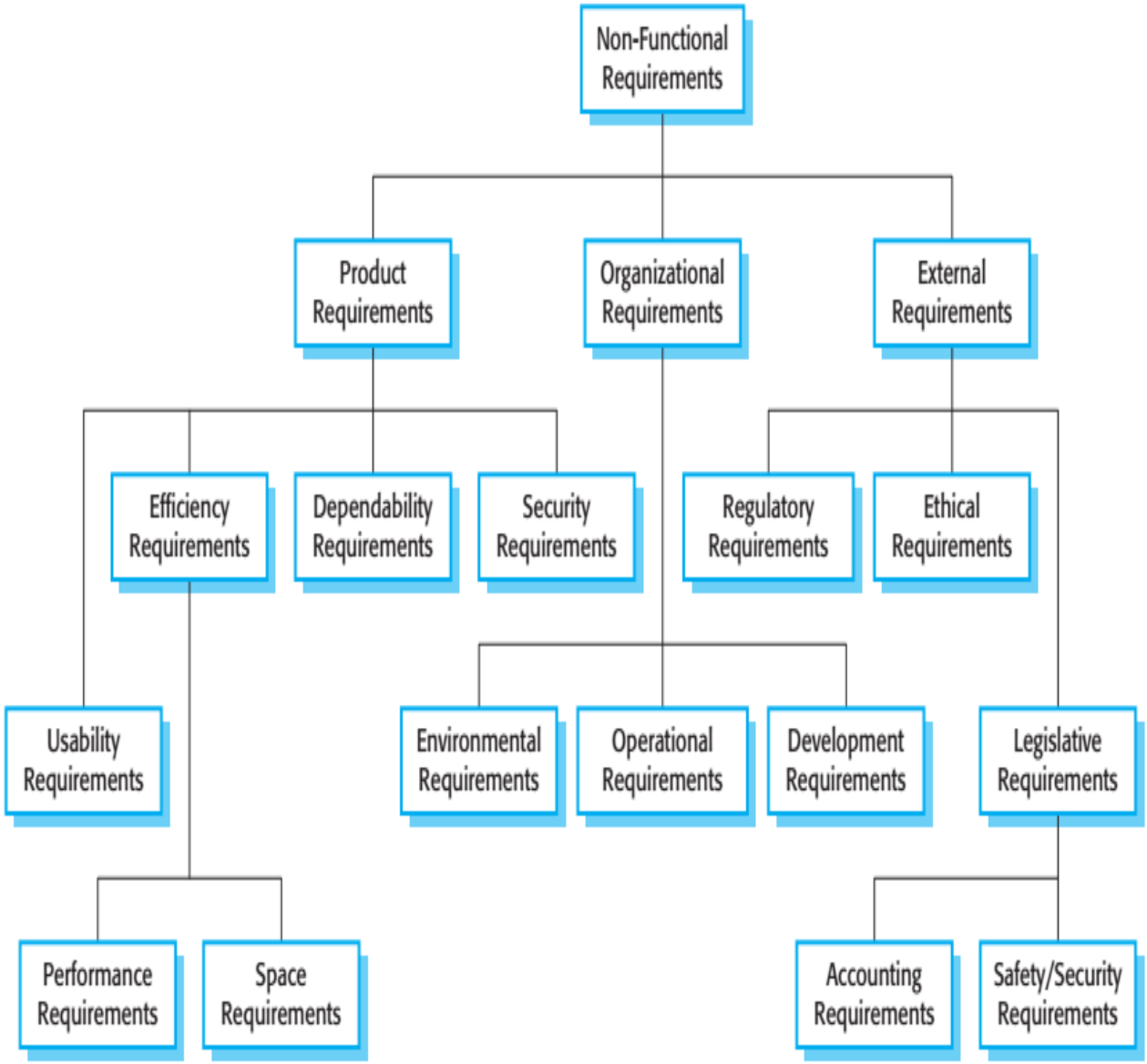
RBSSBT tizimi uchun funksional talablar, ruhiy kasallikka chalinayotgan bemorlar uchun

1. Foydalanuvchi barcha shifoxonalar uchun uchrashuv vaqtlarini qidira olishi kerak
2. Tizim har bir kunga har bir shifoxona uchun novbat kutib turgan bemorlarga uchrashuv kunini tuzib berishi lozim
3. Har bir ishchi xodim o'zining sakkiz xonali ishchi nomeri orqali identifikatsiyadan o'ta olishi lozim

RBSSBT tizimi uchun funksional bo'lmagan talablar

1. Tizim tibbiyot xodimlari uchun foydalanishga qulay bo'lishi lozim
2. Tibbiyot xodimlari to'rt soatlik treyningdan so'ng tizimning barcha funksiyalaridan foydalana olishi lozim

Funksional bo'lmagan talablar



PRODUCT REQUIREMENT - RBSSBT normal ish vaqtida barcha klinikalar uchun mavjud bo'ladi(Dushanba-Shanba, 08.30-17.30). Uzilishlar bir kunda bir marta va 5 sekunddan oshmasligi kerak.

ORGANIZATIONAL REQUIREMENT - RBSSBT tizimi foydalanuvchilari o'z kartalari orqali identifikatsiya qilinishi lozim.

EXTERNAL REQUIREMENT - tashqi talablar.

Dasturiy ta'minot talablari hujjati

Dasturiy ta'minot hujjati bu tizimni ishlab chiquvchilar nimani oshirishi lozimligini ifodalovchi rasmiy hujjatdir. U tizim uchun foydalanuvchi talablarini

ham tizim talablarining batafsil spesifikatsiyasini ham o'z ichiga oladi. Bazida foydalanuvchi va tizim talablari bitta qilib tavsiflanadi. Bazi hollarda esa foydalanuvchi talablari hujjatning kirish qismi va tizim talablari asosiy qismni tashkil qiladi.

Talablar hujjatidan foydalanuvchilar

Tizim mijozlari	Talablarni ko'rsatish va talablar bajarilganligiga tekshirish uchun o'qish. Mijozlar shuningdek talablarni o'zgartirishi mumkin.
Boshqaruvchilar	Tizimni narxlash va ishlab chiqishni rejalashritish uchun talablar hujjatidan foydalanish.
Tizim injinerlari	Ishlab chiqarilayotgan tizimni tushunish uchun talablardan foydalanish.
Tizimni testlovchi injinerlar	Tizimni haqiqiylikka tekshirish uchun tizm talablaridan foydalanish.
Tizimga xizmat ko'rsatuvchi injinerlar	Tizim va uning qismlari munosabatini tushunish uchun talablardan foydalanadi.

Talablar hujjatining strukturasi

Bo'lim	Tavsifi
Muqaddima	Hujjatni kutilgan o'quvchilarini aniqlash lozim
Kirish	Tizim muhimligini tasvirlash. Tizim

	funksionalligi qisqacha tasvirlanadi.
Glossariy	Hujjatda foydalanilgan texnik terminlarni aniqlash
Foydalanuvchi talablari	Foydalanuvchi uchun taminlangan servislarni tasvirlash
Tizim arxitekturasi	Kutilgan tizim arxitekturasini yuqori-darajali ko'rinishi
Tizim talablari	Funksional va funksional bo'lmagan talablarning batafsil ko'rinishi
Tizim modellari	Tizim componentalari orasidagi munosabatlarni grafik tizimini ko'rsatish
Tizim evolutsiyasi	Tizimga asoslanib fundamental taxminlarni tasvirlash
Ilova	Ishlab chiqarilayotgan ilova haqida batafsil ma'lumotlar; masalan, apparat ta'minot va ma'lumotlar bazasi
Index	Hujjat indekslari

Foydalanilgan adabiyotlar

1. "Software Engineering", by Ian Sommerville, 2015
2. Davis, A. M. (1993). Software Requirements: Objects, Functions and States. Englewood Cliffs, NJ: Prentice Hall.
3. http://www.SoftwareEngineering-9.com
4. http://www.pearsonhighered.com/sommerville

3 - amaliy mashg'ulot. Loyiha uchun kerakli holat diagrammalarni ishlab chiqish (Use case, class va sequence diagrammalari)

Ishdan maqsad: Berilgan loyiha uchun qo'yilgan tizim va funksional talablardan kelib chiqqan holda use case, class, sequence va boshqa mos holat diagrammalarini shakllantirish ko'nikmalarini hosil qilish.

Ishni bajarish tartibi

1. StarUML dasturini ishga tushirish;
2. Loyiha uchun use case diagrammasini ishlab chiqish;
3. Berilgan loyihaning class diagrammasini ishlab chiqish
4. Loyihaning sequence diagrammasini ishlab chiqish

UML (Unified modeling language) – ishlab chiqiladigan dasturiy ta'minotni tasniflash, visual tasvirlash, qurish vahujjatlashtirish uchun standart til hisoblanadi. UML tizimning strukturali va bog'langan ko'rinishlarini tasvirlashda ishlatiladi. Bunda UML ning turli ko'rinishdagi diagrammalaridan (asosan 9 ta diagramma mavjud) foydalaniladi. Ushbu diagrammalar orqali tizimda bajariladigan jarayonlarni visual tasvirlash va obyektlar orasidagi bog'lanishni ko'rsatish mumkin.

Mazkur amaliy ishda biror bir tizimning umumiy strukturasi, use case, class va sequnce diagrammalrini yaratishini ko'rib o'tamiz.

Use case diagram – bu tizimdagi case lar va actor lar orasidagi bog'lanish diagrammasi bo'lib, bunda tizimning foydalanuvchilari va tizim tashkil etuvchi qismlari orasidagi bog'lanish tasvirlanadi.

Class diagram – bu turdagi diagrammada sinflar, interfeyslar, hamkorlik va ularning bog'lanishlari ko'rsatiladi. Bu diagramma tizimdagi mavjud sinflarni, ularning atributlari, metodalari va interfeys xususiyatlarini ko'rsatadi.

Sequence diagram – o'zaro munosabat diagrammasi bo'lib, bunda tizim tasjkil etuvchi modullari orasida o'zao bog'likliklar aniq ko'rsatiladi. O'z navbatida UML diagrammalridan "collobaration diagram" ham obyektlar yoki modullar aro o'zaro munosabatni ta'minlaydi va tizining dinamik ko'rinishda holatini ko'rsatib beradi.

Ishni bajarish namuna

Topshiriq: Ko'p qavatli binolarda lift mavjud bo'lib, ushbu liftning ishlash tizimini loyihalashtirish va UML diagrammalarini shakllantirish lozim.

Dastlab lift tizimining qisqacha tasnifini keltiramiz. Umumiy ishlash

prinsipiga ko'ra lift foydalanuvchi tomonidan chaqrilida, yo'nalish ko'rsatiladi va ko'rsatilgan yo'nalish bo'yicha belgilangan qavatga ko'tariladi yoki aksincha.

Liftda belgilangan qavatga yetib olish algoritmi quyidagicha:

1. liftni chaqirish (lift turgan joyiga nisbatan pastdan yoki yuqoridan);
2. lift eshigi ochiladi;
3. kerak bo'lgan qavat belgilanadi;
4. left eshigi yopiladi;
5. belgilangan qavatga harakatlanadi;
6. lift ehigi ochiladi;
7. lift eshigi yopiladi;
8. lift neytral holatga o'tadi.

*Izoh: Agar zarur bo'lgan hollarda lift tizimi ishdan chiqib qolsa favqulotda yordam tugmasi (emergency call) orqali navbatchini (dispatcher) chaqirish mumkin.

Demak yuqoridagilardan kelib chiqqan holda mazkur tizimning talablarini ishlab chiqishimiz mumkin:

Lift tizimining talablari

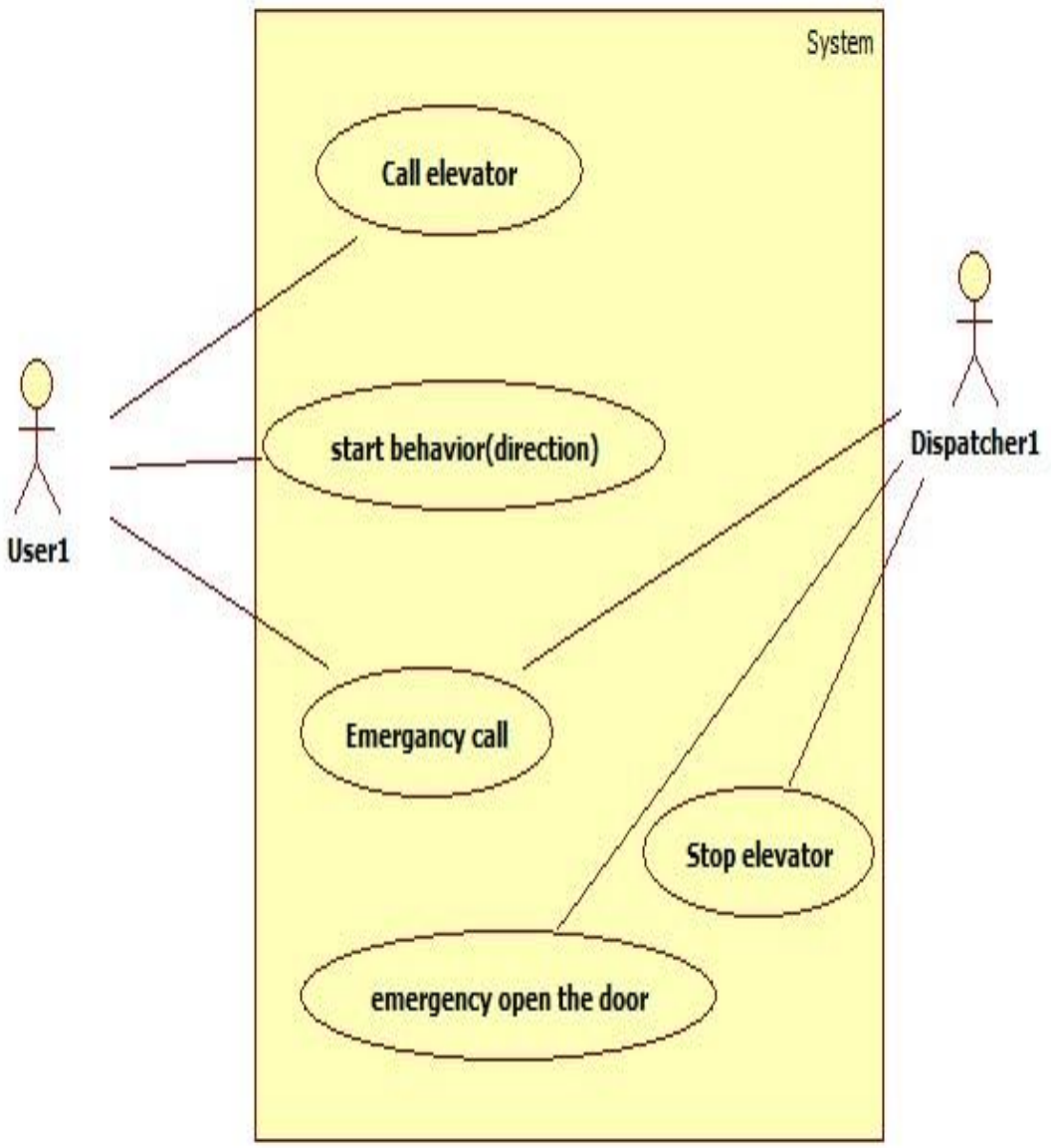
- ✓ T-01: foydalanuvchi tomonidan chaqiruv amalga oshirilganda kerakli qavatga harakatlanishi kerak;
- ✓ T-02: lift belgilangan qavatga yetib kelganida to'xtashi va eshil ochilishi kerak;
- ✓ T-03: biror bir qavat ko'rsatilmaguncha ma'lum vaqt neytral holatda turishi kerak;
- ✓ T-04: harakatlanish uchun qavat ko'rsatilgandan keyin eshik yopilishi kerak;
- ✓ T-05: harakat ko'rsatilgan qavatgacha amlaga oshishi kerak;
- ✓ T-06: ko'rsatilgan qavatga yetib borganidan keyin lift to'xtashi bilan eshik avtomatik ochilishi kerak;
- ✓ T-07: ma'lumo vaqtdan keyin eshik avtomatik yopilishi kerak

Lift tizimining funksional talablari

- ✓ FT-01: foydalanuvchi (user) – liftdan foydalanich jarayonida ketma-ketlikni to'g'ri bajarishi va harakatni amalga oshirishi lozim;
- ✓ FT-02: navbatchi (dispatcher) – liftda favqulotda vaziyat bo'lganida yoki ishdan chiqqanida birinchi yordam ko'rsatishi lozim;
- ✓ FT-03: lift qurilmasiga javobgar tashkilot boshqaruvchisi navbatchi faoliyatini muvoffiqlashtirib tuishi lozim;
- ✓ FT-04: lift tizimi xatosiz ishlashi va chidamli bo'lishi lozim.

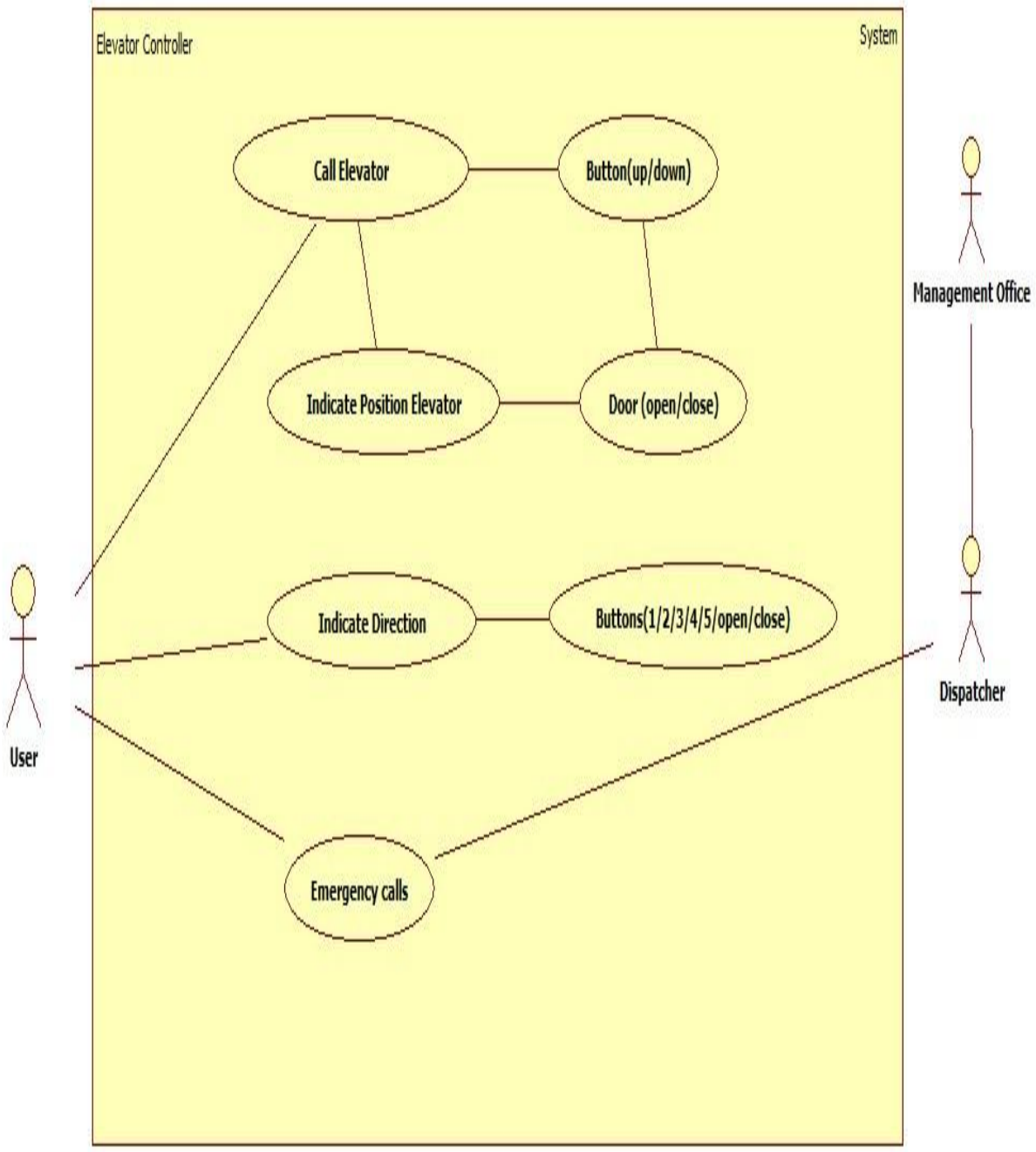
Ushbu talablarni tizim xususiyatidan kelib chiqqan holda hohlagancha

o'zgartirish va davom ettirish mumkin, Ushbu talablar ishlab chiqiladigan tizimning asosiy qismini tashkil etadi. Endi quyidagi rasmda lift tizimining umumiy sxemasini keltiramiz:



3.1-rasm. Lift tizimining umumiy sxemasi

Tizimning use case diagrammasi quyidagicha bo'ladi:

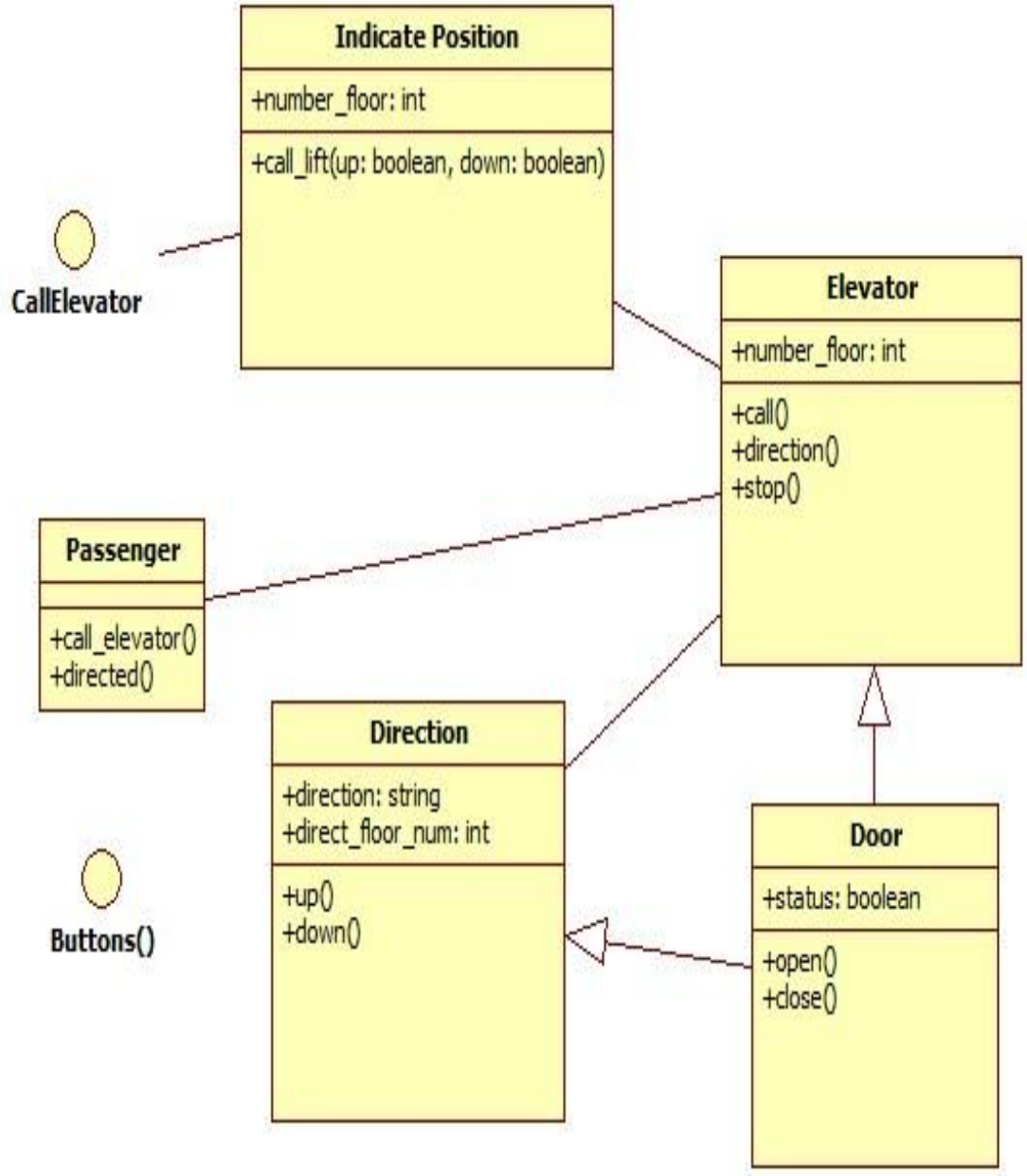


3.2-rasm. Lift tizimining use case diagrammasi

Tizimning use case diagrammasini ishlab chiqishda asosan quyidagi elementlardan foydalaniladi:

- Use case (tizimning tashkil etuvchi modullari)
- Actor (tizim foydalanuvchisi, administrator, dispatcher)
- Dependency, generalization, association (tizim modullarini bog'lash qonuniyatlari)

Tizimning class diagrammasi quyidagicha bo'ladi:

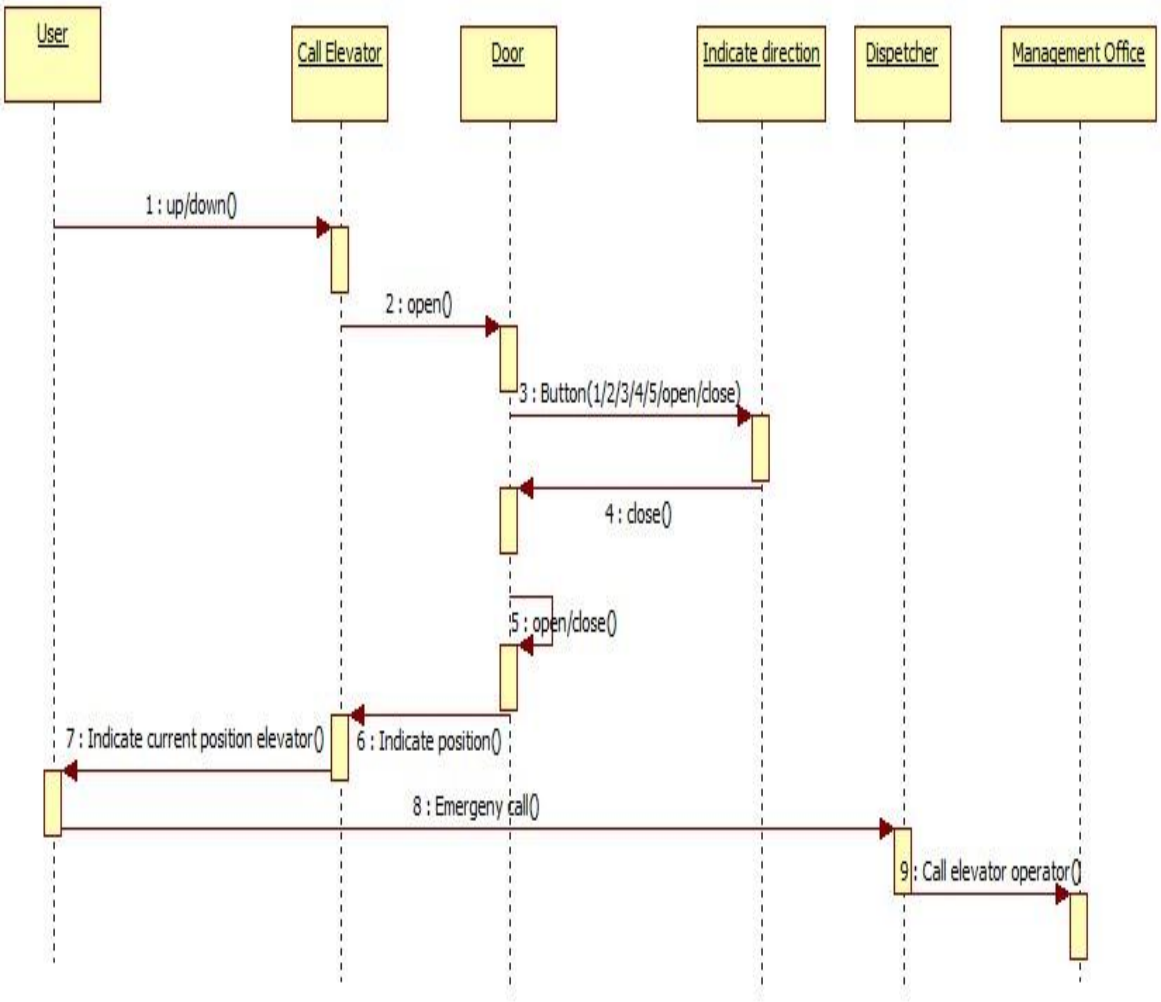


3.3-rasm. Lift tizimining class diagrammasi

Tizimning class diagrammasini shkllantishda esa quyidagi asosiy elementlardan foydalaniladi:

- Classes (sinflar)
- Interfaces (interfeyslar)
- Collaborations (hamkorliklar)
- Dependency, generalization, association (sinflar va interfeyslarni bir – biriga bog'lash qonuniyatlari).

Tizimning sequence diagrammasi

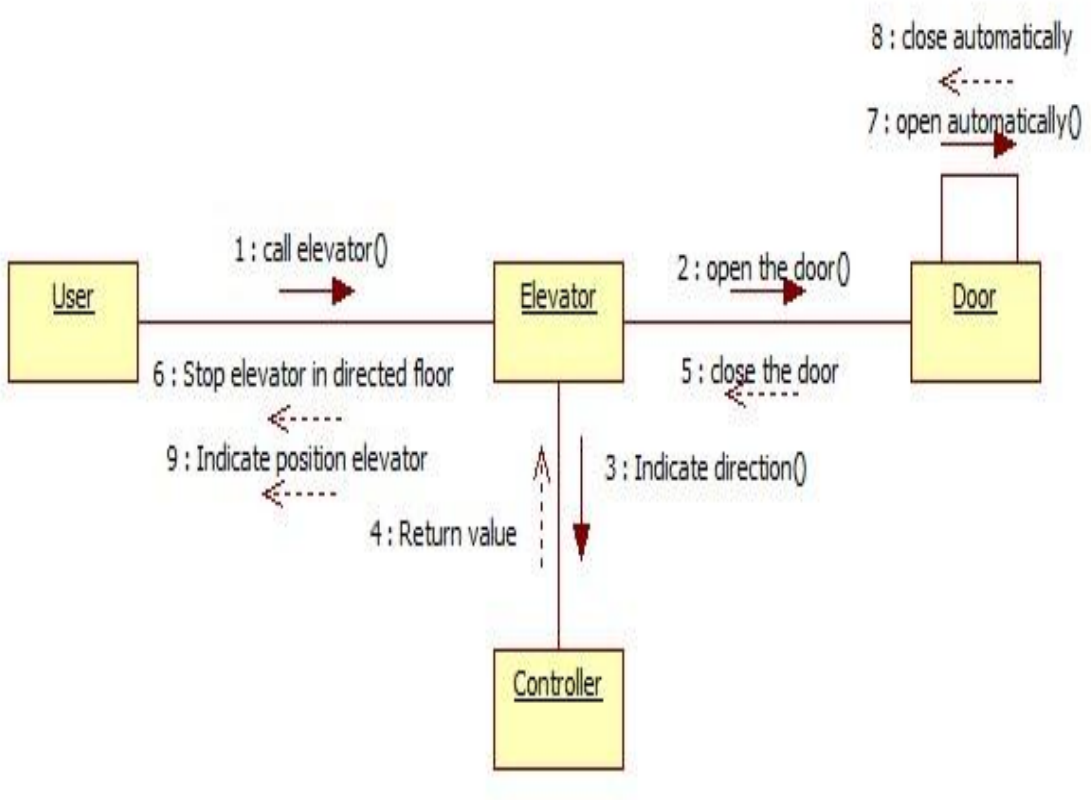


3.4-rasm. Lift tizimining sequence diagrammasi

O'z navbatida quyidagi elementlar tizimning sequence diagrammasini shakllantirishda ishlatiladi:

- Objects (tizim obyetlari)
- Links (obyekt bog'lash va yo'naltirish belgisi)
- Messages (bog'lanish holatini bildiruvchi xabarlar)
- Respond Time (qayta aloqa vaqtini ko'rsatish)

Tizimning collobaration (hamkorlik) diagrammasi



3.5-rasm. Lift tizimining collobaration diagrammasi

Foydalanilgan adabiyotlar

1. Learning UML 2.0 by Russ Miles and Kim Hamilton May 2, 2006
2. "Software Engineering", by Ian Sommerville, 2015
3. Davis, A. M. (1993). Software Requirements: Objects, Functions and States. Englewood Cliffs, NJ: Prentice Hall.
4. http://www.uml-diagrams.org/
5. http://www.SoftwareEngineering-9.com
6. http://www.pearsonhighered.com/sommerville

4 - amaliy mashg'ulot. Tizim arxitekturasi va dizaynini qurish

Ishdan maqsad: Dasturiy ta'minot arxitekturasi va arxitekturaviy dizayn yaratish va uni tizimga moslashtirish.

Dasturiy ta'minot arxitekturasi tizimni ishlab chiqishda muhim o'rin tutadi, sababi u tizimni ishlab chiqilishiga, ishonchliligiga, keng ko'lamda qo'llanilishiga va qayta ishlab chiqilishiga ta'sir qiladi.

Arxitekturaviy dizayn jarayonida tizim arxitektori tizim va tizimni ishlab chiqish jarayoniga samara keltiradigan strukturaviy qarorlar chiqarishi lozim. Bilim va tajribaga asoslanib tizim haqida quyidagi qarorlarni qabul qilishi kerak:

1. Loyihalanayotgan tizim biror bir umumiy loyihalar arxitekturasiga mos keladimi?
2. Tizim yadro yoki protsessorlarga qanday taqsimlanadi?
3. Qanday arxitekturaviy andozalar yoki usullardan foydalanish mumkin?
4. Tizim strukturasiga qanday fundamental yondashuvlar foydalaniladi?
5. Tizimdagi strukturaviy komponentalar qanday qilib qismlarga ajratiladi?
6. Tizimdagi komponentalarning operatsiyasini nazorat qilishda qanday strategiyadan foydalaniladi?
7. Tizimning funksinal bo'lmagan talablarini taminlash uchun qanday qanday arxitekturaviy tashkilot eng yaxshi?
8. Arxitekturaviy dizayn qanday baholanadi?
9. Tizimning arxitekturasi qanday hujjatlashtirilishi lozim?

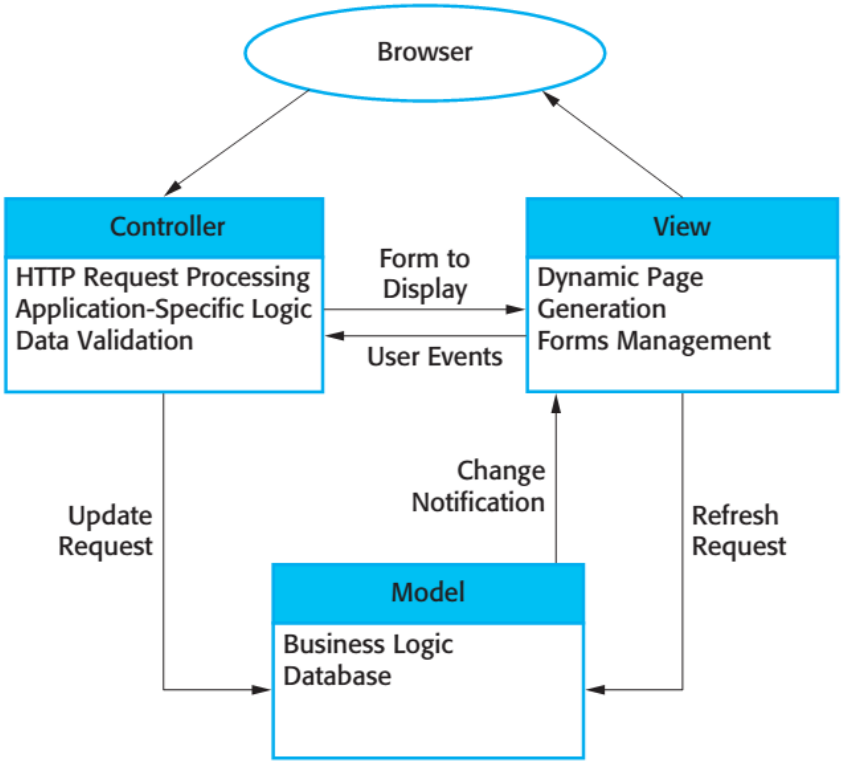
Garchi har bir dasturiy ta'minot tizimi unikal bo'lsada o'xshash arxitekturalardan foydalanilishi mumkin.

Dasturiy ta'minot tizimi haqidagi bilimlarni ko'rsatishda andozalar g'oyasidan foydalanish hozirgi kunda ko'p qo'llanilmoqda.

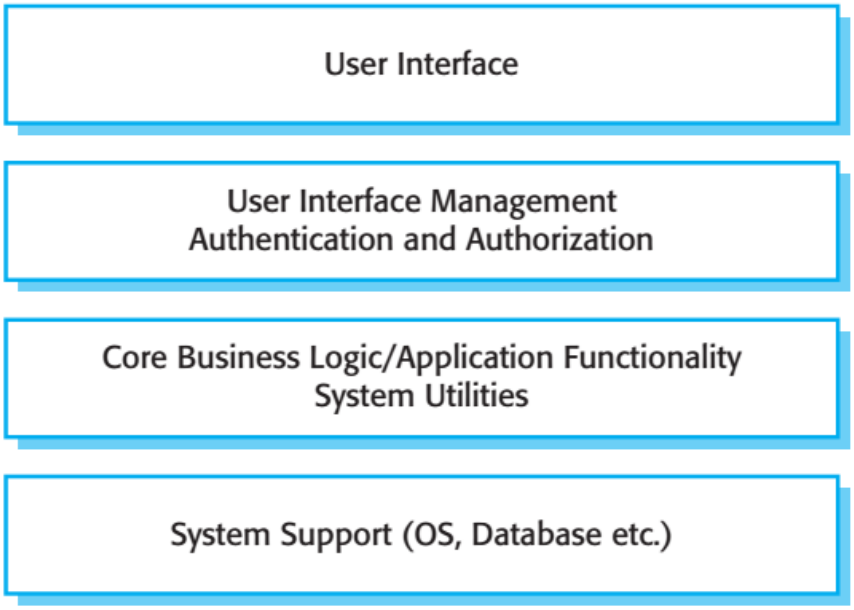
Har bir arxitekturaning yaxshi va yomon tomonlari mavjud

Quyida eng ko'p qo'llaniladigan arxitekturaviy andozalarni keltiramiz

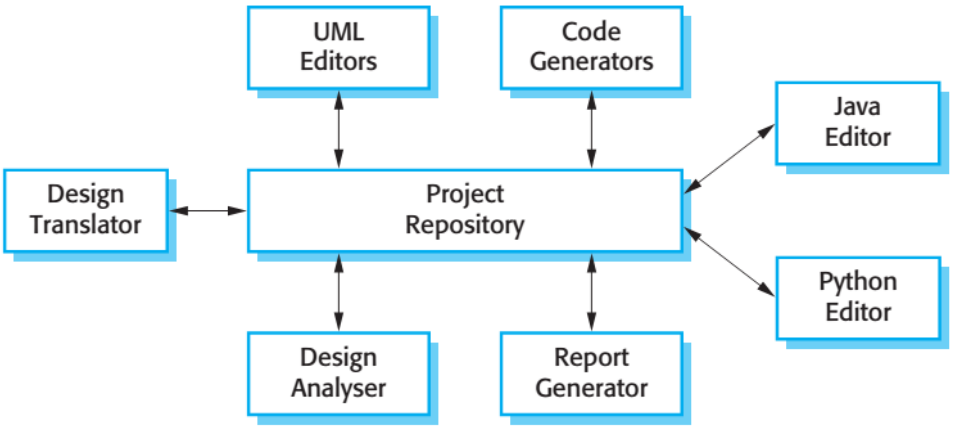
Veb ilova arxitekturasida MVC andozasidan foydalanish



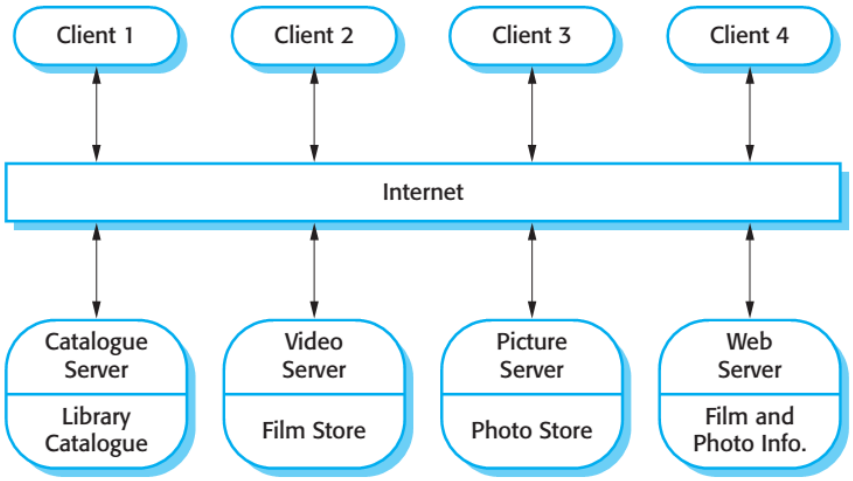
Qatlamli arxitektura



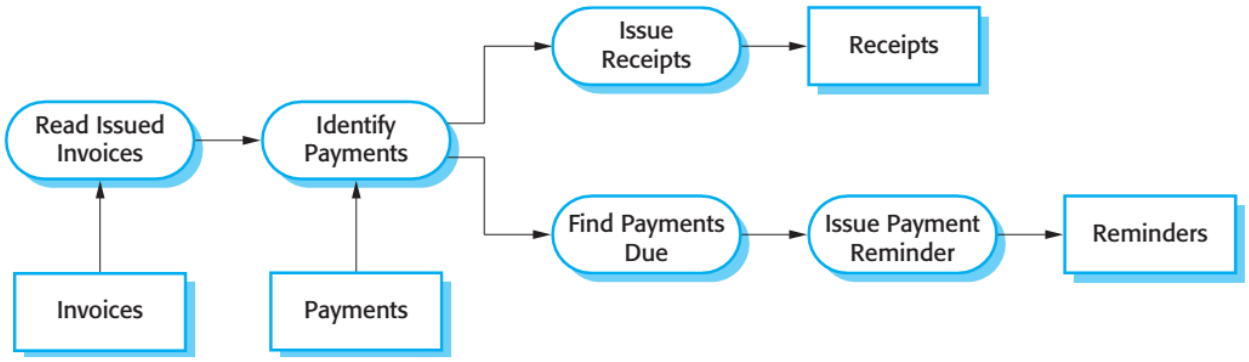
Jihojlangan arxitektura



Klient - Server arxitekturasi



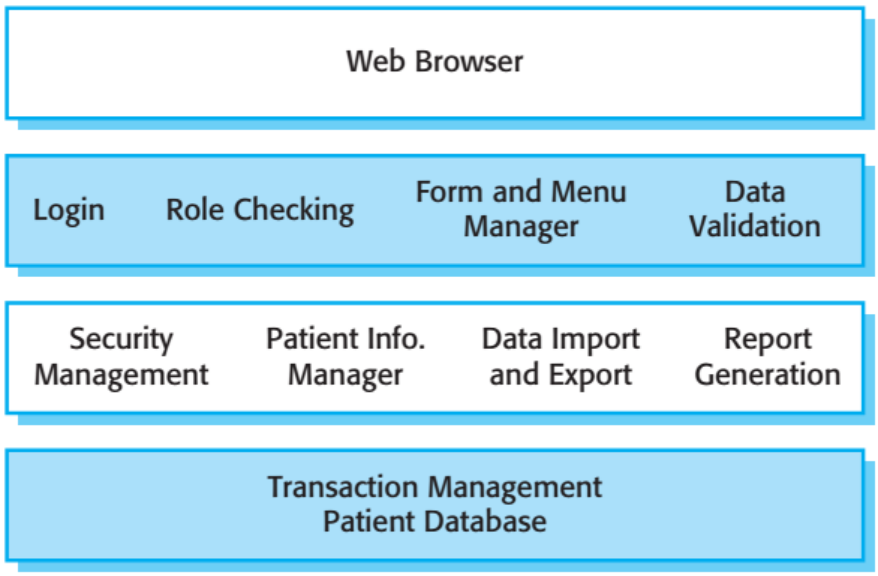
Quvur and filter arxitekturasi



Ruhiy kasallikka chalingan bemorlar sog'ligini saqlash tizimi dasturiy ta'minotini

yaratishda quyidagi arxitekturani qo'llanilishini ko'rsatib beramiz.

Qatlamli arxitektura



Foydalanilgan adabiyotlar

1. "Software Engineering", by Ian Sommerville, 2015
2. Bass, L., Clements, P. and Kazman, R. (2003). Software Architecture in Practice, 2nd ed. Boston: Addison-Wesley.
3. http://www.SoftwareEngineering-9.com
4. http://www.pearsonhighered.com/sommerville

5 - amaliy mashg'ulot. Dasturiy ta'minot dizayni va amaliy ko'rinishi.

Ishdan maqsad: Dasturiy ta'minot uchun dizayn na'munalar ishlab chiqish va uning amaliy ko'rinishini yaratish.

Dasturiy ta'minotlar uchun ma'lum bir dizayn na'munalari mavjudki, ular ustida bir necha bosqichli amallarni bajargandan so'ng ushbu na'munani tizim uchun qo'llash mumkin bo'ladi. Dizayn na'muna bir vaqtning o'zida qo'yilgan muammoning tavsifi va uning yechimi uchun mohiyatini tashkil etadi.

Siz dizaynni ishlab chiqishda UML dan foydalansangiz, siz odatda ikki xil turdagi dizayn modellarni ishlab chiqasiz:

1. Strukturaviy modellar, Obyekt sinflari va ularning munosabatlaridan foydalanib tizimning statik strukturasi tasvirlanadi.
2. Dinamik modellar, tizimning dinamik strukturasini tasvirlaydi va tizim obyektlari orasidagi munosabatni ko'rsatasdi.

Dasturiy injiniring tizimning dastlabki talablaridan tortib tizimni ishlab chiqib amaliyotga qo'llashgacha bo'lgan barcha faoliyatlarni o'z ichiga oladi. Bu jarayonning eng kritik bosqichi albatta tizimni amaliy ko'rinishidir. Ya'ni siz dasturiy ta'minotning ishlatib bo'ladigan talqinini ishlab chiqish. Amaliy ko'rinish yuqori yoki past darajali dasturlash tillarida amalga oshirilishi mumkin. Dasturiy injiringda amaliy ko'rinish uchun muhim bo'lgan jihatlarni keltiramiz:

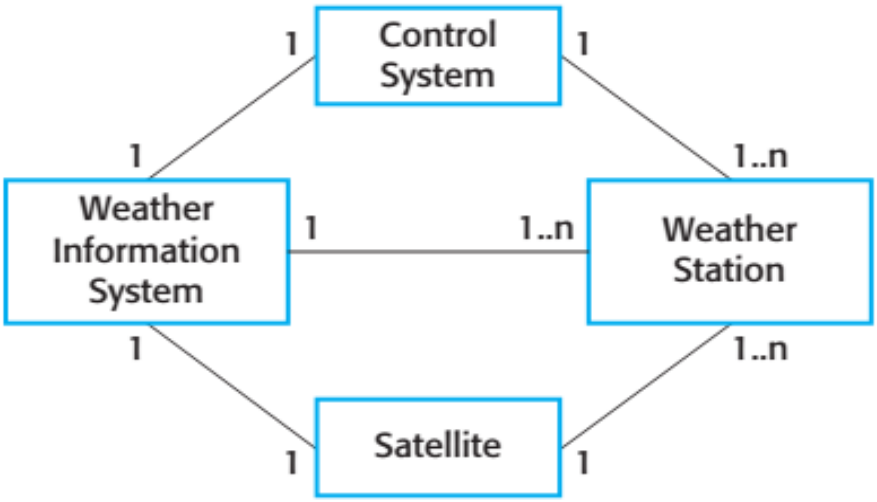
1. *Reuse* Ko'pgina zamonaviy dasturiy ta'minotlar mavjud komponentalar yoki tizimlardan qayta foydalanish orqali quriladi. Siz dasturiy ta'minot ishlab chiqayotganda imkon qadar mavjud kodlardan foydalanishingiz lozim.
2. *Configuration management* Ishlab chiqish jarayonida har bir dasturiy ta'minot komponentasi bir necha xil talqinda yaratiladi. Agar siz sozlamalarni boshqarish tizimida ushbu talqinlarni yozib bormasangiz tizimda xato talqindagi komponentalardan foydalanishingiz mumkin.
3. *Host-target development* Dasturiy mahsulot ishga tushiriladigan kompyuter bilan dastur ishlab chiqarilgan komptuer bir xilda bo'lmaydi. Shu sababdan tizimni turli xildagi kompyuterda ishlashini ham hisobga olish lozim.

Ob-havo stansiyasi *use case* lari

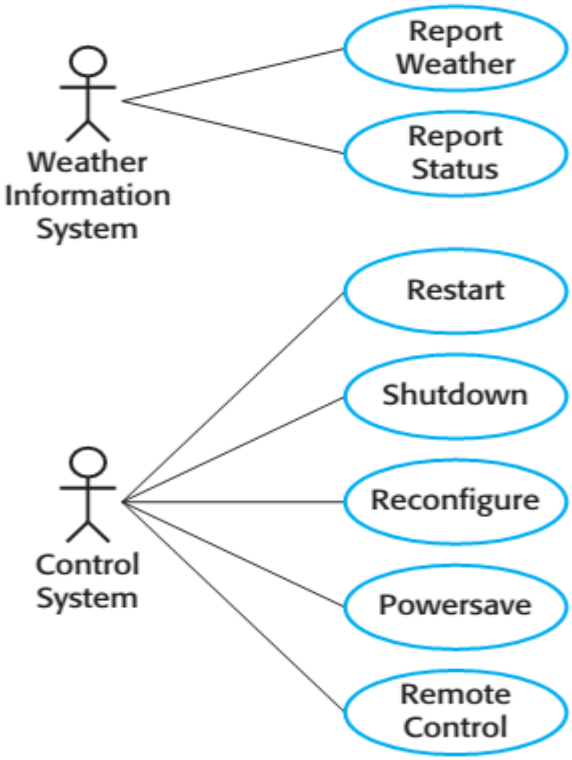
Report weather - ob-havo axborot tizimiga ob-havo ma'lumotlarini jo'natish.

Report status - ob-havo axborot tizimiga holat ma'lumotlarini jo'natish.
Restart - agar tizimi o'chgan holatda bo'lsa tizimni qayta yuklash.
Shutdown - ob-havo stansiyasini o'chirish.
Reconfigure - ob-havo stansiyasi dasturiy ta'minotini qayta sozlash.
Powersave - ob-havo stansiyasini quvvatni tejash rejimiga o'tkazish.
Remote control - ob-havo stansiyasi tizim ostilarga nazorat komandalarini jo'natish.

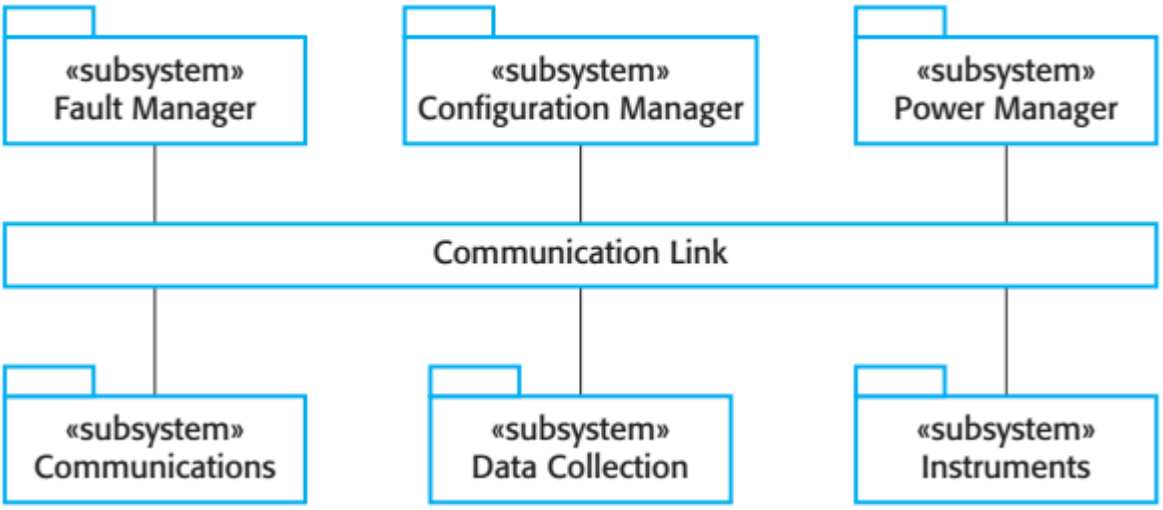
Ob-havo stansiyasi uchun tizim konteksti



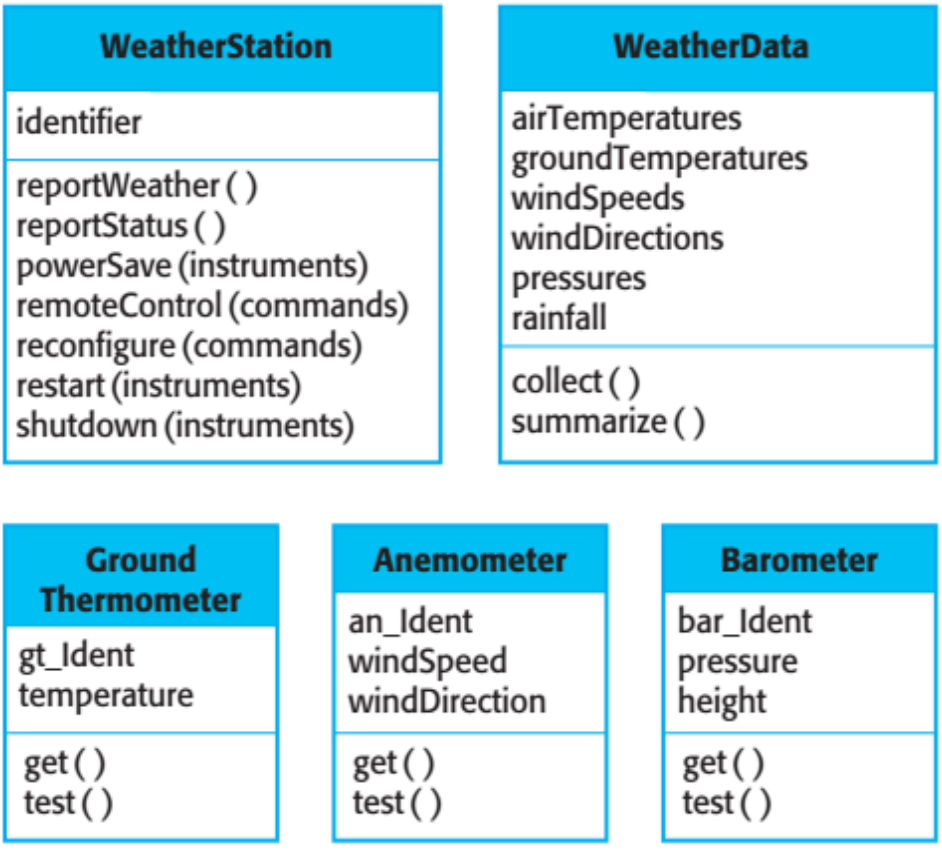
Ob-havo stansiyasi uchun use case lar



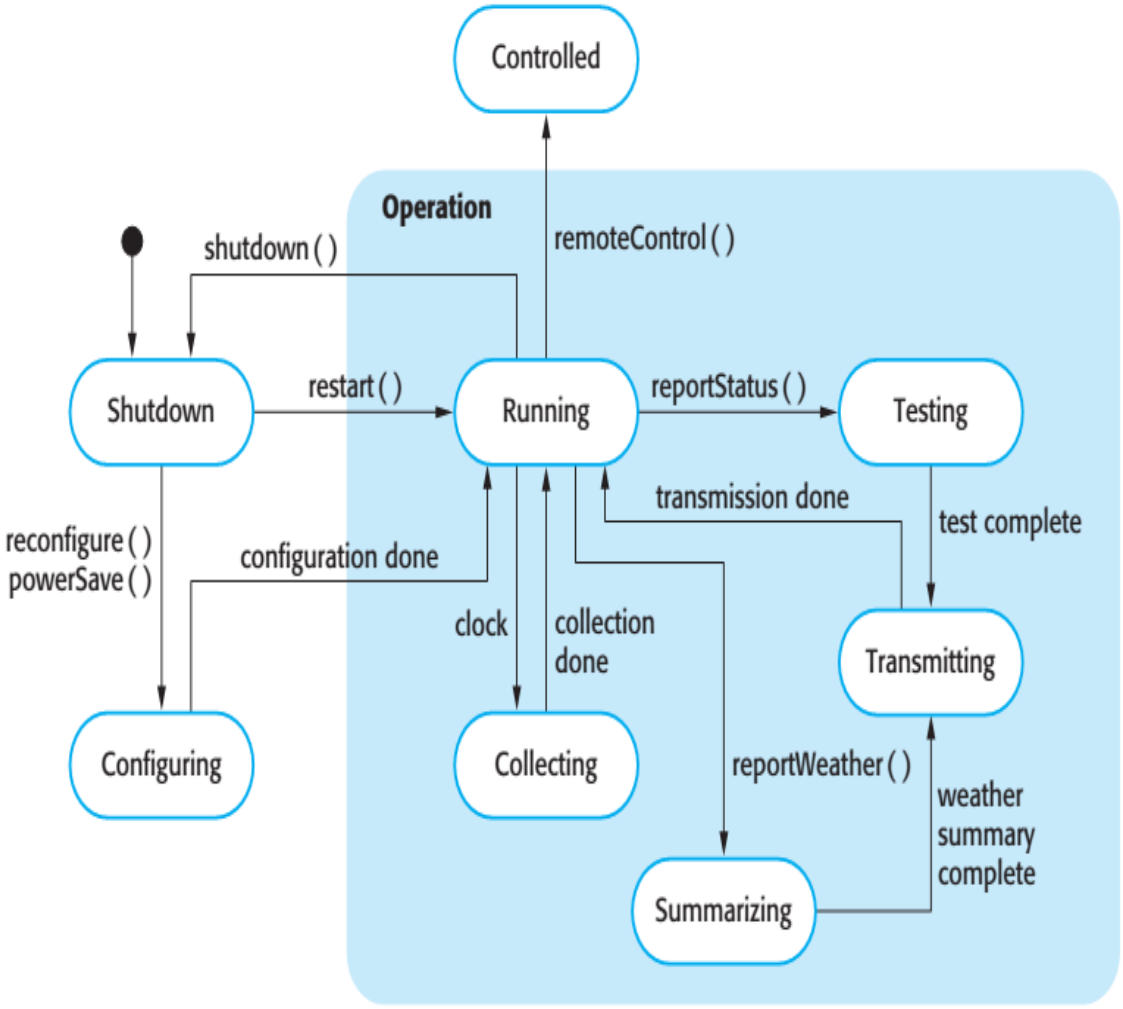
Ob-havo stansiyasining yuqori darajali arxitekturasi



Ob-havo stansiyasining obyektlari



Ob-havo stansiyasining holat diagrammasi



Foydalanilgan adabiyotlar

1. "Software Engineering", by Ian Sommerville, 2015
2. Bass, L., Clements, P. and Kazman, R. (2003). Software Architecture in Practice, 2nd ed. Boston: Addison-Wesley.
3. http://www.SoftwareEngineering-9.com
4. http://www.pearsonhighered.com/sommerville

6 - amaliy mashg'ulot. Dasturiy ta'minotni testlash va tekshirish

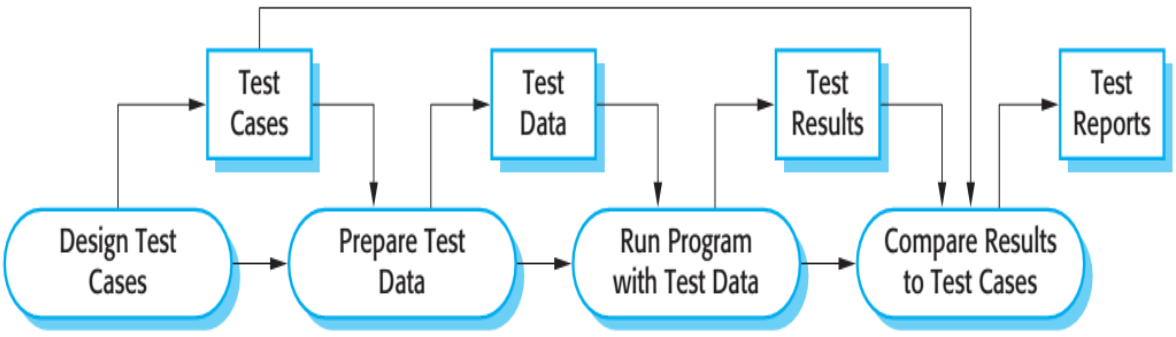
Ishdan maqsad: Dasturiy ta'minotni testlash va testlash bosqichlarini o'rganish.

Testlash, dasturiy ta'minotni foydalanishga qo'yishdan oldin dastur nuqsonlarini topish va ularni to'g'irlashga mo'ljallangan dasturlarni ko'rsatishga mo'ljallangan. Siz dasturiy ta'minotni testlagan chog'ingizda, sun'iy ma'lumotlardan foydalanib dasturni ishga tushirasiz. Siz dasturni testlash natijalarini xatolarga, anomaliya(normal holatdan chetlashish)ga yoki dasturning nofunksional sifatlari haqida ma'lumotga tekshirasiz.

Testlash jarayonida ikkita alohida maqsadlar mavjud:

1. Ishlab chiqaruvchi va buyurtmachiga ularning dasturiy ta'minoti talablari bajarilayotganini namoyish etish. Buyurtma qilingan dasturiy ta'minot uchun hujjatdagi talablarning har biri uchun kamida bitta testlash bo'lishi lozim. Umumiy dasturiy ta'minot mahsulotlari uchun esa, tizimning barcha funksiyalari uchun, shuningdek, tayyor mahsulotda ishlatiladigan funksiyalar aralashmasi uchun testlashlar bo'lishi kerak.
2. Dasturiy ta'minot noto'g'ri, ishonarsiz yoki spetsifikatsiyalarga mos kelmagan hollarni aniqlash. Ular dasturiy ta'minotning nuqsonlari hisoblanadi. Nuqsonlarni testlash keraksiz tizimlarning nuqsonlariga barham berish bilan bog'liq, masalan, tizimning to'xtab qolishi, boshqa tizimlar bilan keraksiz bo'g'lanishi, ma'lumotlarning noto'g'ri hisoblanishi va buzilishi.

Dasturiy ta'minotning testlash jarayoni modeli



Odatda tijorat dasturiy ta'minot tizimi quyidagi uchta testlash bosqichidan o'tishi kerak:

1. *Development testing*, tizim ishlab chiqarilayotgan paytda testlanadi. Tizim dizaynerlari va dasturchilari tomonidan testlanadi.

2. *Release testing*, tizim foydalanuvchilarga taqdim etilishidan oldin to'liq tizim testlovchi guruh tomonidan testlanadi.
3. *User testing*, foydalanuvchilar o'zining muhitida tizimni testlaydi.

Amaliyotda testlash ikki usul - avtomatik va qo'lda tekshirishning birgalikda qo'llanishi orqali amalga oshiriladi. Qo'lda tekshirish jarayonida testlovchilar dasturni bir necha testlovchi ma'lumotlarni kiritgan holda ishga tushirib hosil bo'lgan natijalarni kutilgan natijalar bilan solishtiradi. Ular testlash jarayoni mobaynida vujudga kelgan farqlar va xatolarni yozib olib dasturiy ta'minotni ishlab chiqaruvchiga yetkazadilar. Avtomatik testlash jarayoni esa dasturiy ta'minotni ishlab chiqish davomida bir necha marta ma'lum bir testlovchi tizim tomonidan amalga oshirilib boriladi. Teslashning avtomatik uslubi qo'lda olib boriladigan testlashdan tezroq va unumliroqdir, ayniqsa, testlash natijasida vujudga kelgan xatoliklarni tuzatishdan hosil bo'lishi mumkin bo'lgan yangi nosozliklarni tekshirish kerak bo'lganda, ya'ni qayta teslash jarayonida avtomatik testlash usuli foydaliroq hisoblanadi.

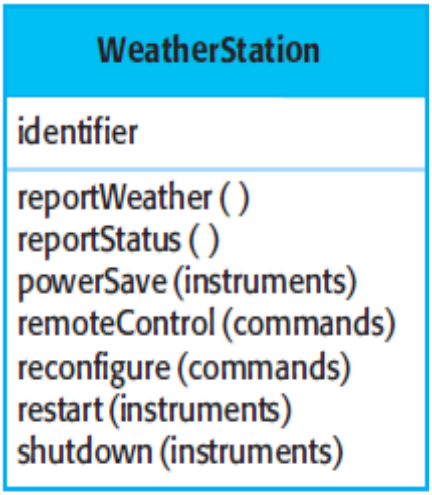
Endi tijorat dasturiy ta'minotlarining testlash bosqichlariga batafsil to'xtalib o'tamiz:

I. Ishlab chiqarishdagi testlash (Development testing).

Ishlab chiqarishdagi testlash dasturiy ta'minotni ishlab chiqaruvchilari tomonidan shu jarayonda olib boriladigan barcha testlarni o'z ichifa oladi. Odatda testlovchi dasturiy ta'minotni ishlab chiqarishda ishtirok etgan dasturchi hisoblanadi. Lekin ba'zan dasturiy ta'minotni ishlab chiqaruvchi jamoa alohida testlovchi va dasturchilardan ham tashkil topgan bo'lishi mumkin.

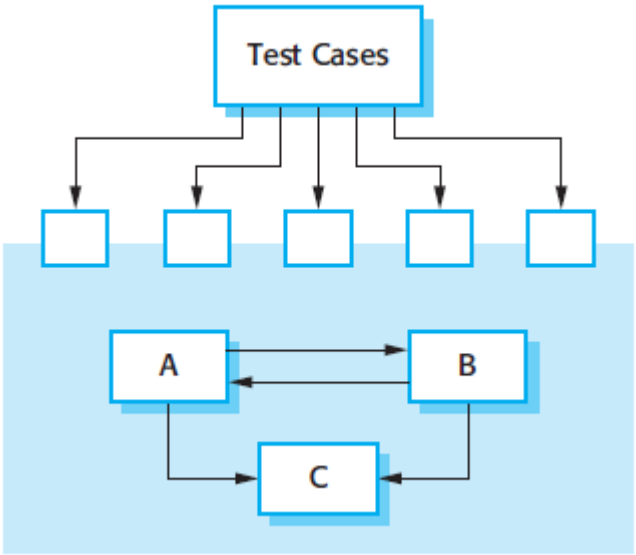
Ishlab chiqarishdagi testlash, odatda quyidagi 3 bosqichni o'z ichiga oladi:

1. *Unit testing* – dasturiy ta'minotning alohida olingan bo'limlari(sinflar, usullar)ni testlash.



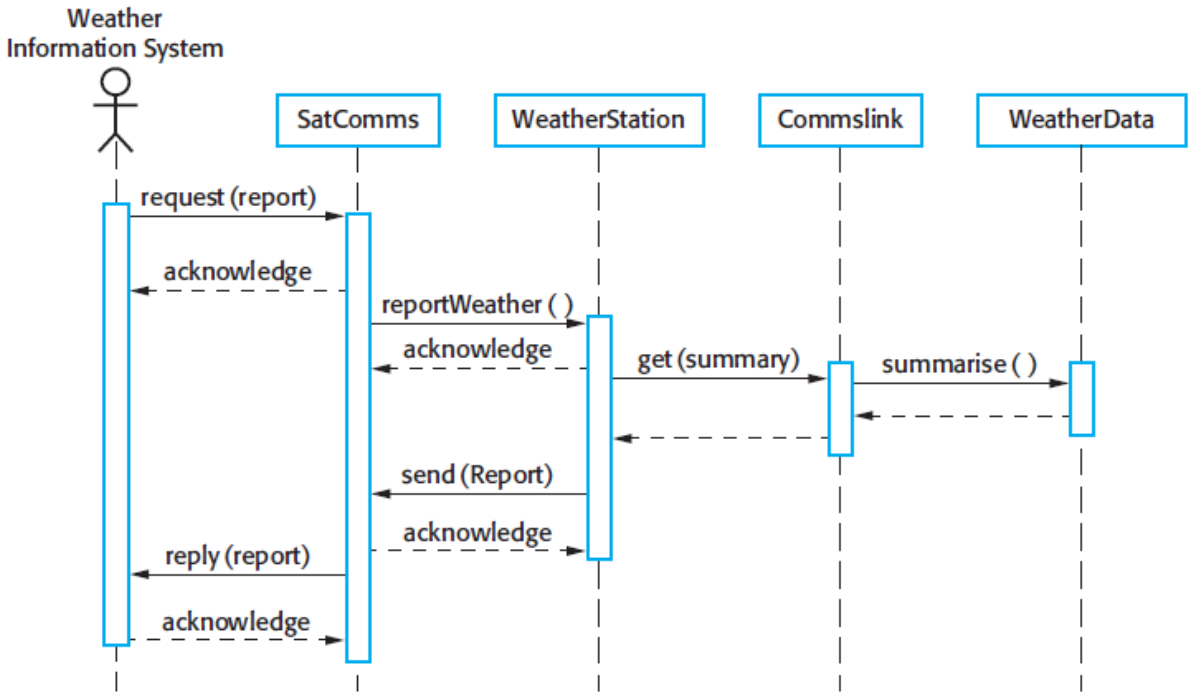
Misol tariqasida ob-havoni aniqlovchi stansiya uchun dasturiy ta'minot ishlab chiqarilayotgan bo'lsa unit testing da uning yuqorida ko'rsatilgan ma'lum bir usullarini tekshirish yetarlidir.

2. *Component testing* – ma'lum bir umumiy vazifani bajaruvchi bir nechta bo'limlardan tashkil topgan biror bir komponentani testlash



Bunda bitta koponentani tashkil qilgan A, B va C bo'limlar umumlashtirilgan holda testlanadi.

3. *System testing* – ba'zi yoki barcha komonentalarni o'z ichiga olgan butun bir tizimni yaxlit holda testlash jarayoni. Bunda testlash butun bir tizimga qaratilgan bo'ladi.



O'z nomi bilan ma'lumki bunda barcha komponentalarni o'z ichiga olgan butun bir tizim yakuniy bosqichda teslanadi.

Testlashning ishlab chiqarish jarayonidagi bosqichi (development testing) testlarning professional dasturchilar tomonidan amalga oshirilganligi bilan muhim va aniqdir.

II. Dasturiy ta'minotni amaliyotga tadbiq etish jarayonidagi teslash(release testing)

Ushbu testlash jarayoni dasturiy ta'minotni ishlab chiqaruvchilar jamoasidan tashqarida, uni amaliyotga tadbiq etish davridagi testlash jarayoni bo'lib hisoblanadi. Testlashning bu bosqichidan ko'zlangan asosiy maqsad dasturiy ta'minotni amaliyotga qo'llash uchun yetarli darajada tayyor ekanligiga ishonch hosil qilishdan iborat. Bu jarayon nafaqat foydalanuvchi talablari. Balki dasturiy ta'minot qo'llanilayotgan butun bir tizimning talablarini hisobga olgan holda amalga oshirilishi lozim.

Bu jarayon ham o'z navbatida bir necha bosqichlarni o'z ichiga oladi va ular quyidagilar:

Requirements-based testing(talablarga asoslangan testlash) – bunday testlash talablarning qay darajada bajarila olinayotganini tetslashga asoslangan. Shuning uchun dasturiy ta'minot uchun talablar ishlab chiqilayotganda ularni keyinchalik testlash uchun qulay bo'lishini ham inobatga olish kerak.

Senario – testing (ssenariyni testlash) – bunda dasturiy ta'minot qay tarzda ishlashini belgilab beruvchi hujjat – ssenariy talablari bajarilayotgani testlanadi.

Performance testing(ijroni yoki dasturiy ta'minotning ishlash jarayonini testlash) – bu dasturiy ta'minotni amaliyotga tadbiq qilish jarayonidagi testlashning so'nggi bosqichi bo'lib bunda asosiy e'tibor dasturning barcha mayda detallarini ham inobatga olgan holda testlash lozim bo'ladi.

III. Foydalanuvchi tomonidan testlash(user testing)

Bunda foydalanuvchi yoki iste'molchi dasturiy mahsulotdan foydalanish davomida yuzaga kelgan muammolar bo'yicha o'z fikr va maslahatlari bergan holda testlash jarayonini amalga oshiradi. Deyarli barcha tizimlar uchun dasturchi va ishga tushirish davomida olib borilgan testlashning o'zi yetarli bo'lib qolmaydi. Chunki murakkab tizimlarda shunday vaziyatlar bo'ladi-ki, faqatgina foydalanuvchi yoki mijoz dasturiy mahsulotdan foydalanayotgan vaqtda muammolar yuzaga keladi. Shu jihati bilan ham testlashning ushbu bosqichi muhim hisoblanadi. Buning ham bir necha usullari mavjud:

a) *Alfa testlash* – bunda foydalanuvchi dasuturiy ta'minotni ishlab chiqaruvchi jamoa bilan birga faoliyat olib borgan holda testlashni amalga oshiradi.

b) *Beta testlash* – dasturiy ta'minotning ma'lum bir funksiyalari foydalanuvchilar testlashlari uchun ochiq bo'ladi va ular testlash jarayonini o'z kasbiy vazifalarini bajara turib amalga oshiradilar.

c) *Tasqidlovchi testlash* – bunda foydalanuvchi dasturiy ta'minotni izchil o'rganib uni o'zining ish jarayoni uchun tadbiq qilish mumkin yoki mumkin emasligi haqida yakuniy qarorni qabul qiladi.

Foydalanilgan adabiyotlar

1. Andrea, J. (2007). 'Envisioning the Next Generationof Functional Testing Tools'. IEEE Software, 24(3), 58–65.
2. "Software Engineering", by Ian Sommerville, 2015
3. http://www.SoftwareEngineering-9.com
4. http://www.pearsonhighered.com/sommerville

V. BO`LIM

KEYSLAR BANKI

V. KEYSLAR BANKI

1-keys.

Bank bir nechta automat bankomatlardan tashkil topgan bo'lib, ular geografik jihatdan taqsimlangan bo'lib tarmoq orqali markaziy serverga ulanadi. Har bir bankomat plastik karta o'quvchi, naqd pul pispanseri, klaviatura/displey va printerdan tashkil topadi. Bankomatdan foydalangan holda bank mijozlari naqd pullarini yechib olishlari, hisob raqamidagi balansni tekshirish yoki bir hisob raqamidan boshqa hisob raqamiga pul ko'chirishlari mumkin. Tranzaksiya mijoz bank kartasini *card reader* ga qo'yishi bilan boshlanadi. Karta orqasidagi magnit shtrix kodlaridan karta nomeri, karta tayyorlangan kun va amal qilish kuni aniqlanadi. Karta haqidagi ma'lumotlar olingandan so'ng tizim kartaning amal qilish muddati o'tib ketmaganligiga tekshiradi. Agar karta amal qilish muddatida va foydalanuvchi kiritgan PIN kodi tizimdagi PIN kod bilan bir-xil bo'lsa karta yo'qotilmagan yoki o'g'rilanmagan. Agar foydalanuvchi uch marta noto'g'ri PIN kod kiritsa karta banomat ichida qoladi. Agar karta barcha tasdiqlashlardan o'tsa pulni yechib olish, balans haqidagi so'rovlar va transferlar bo'limlar ochiladi. Naqd pul yechib olish bo'limi tanlanganda so'ralgan summa hisob raqamida mavjudligi va kundalik pul yechib olish limitidan oshib ketmaganligiga va lokal naqd pul pispanserida mavjud summaga tekshiriladi. Agar tranzaksiya tasdiqlansa so'ralgan miqdordagi naqd pul chiqariladi, tranzaksiya haqidagi barcha ma'lumotlar cheki printerdan chiqariladi va karta egasiga qaytariladi. Mijoz ixtiyoriy paytda tranzaksiyani bekor qilishi mumkin. Tranzaksiya to'xtatilganda karta qaytariladi.

Keysni bajarish tartibi

1.1. Use Case modelini tuzing(individual)
1.2. Static modelini tuzing(individual)
1.3. Obyekt strukturasini tuzing(guruhlarda)
1.4. Dinamik modelini tuzing(individual)
1.5. Tizim arxitekturasi va dizaynini yarating(guruhlarda)
1.6. Tizimni amaliy ko'rinishini yarating (individual)
1.7. Tizimni testlang (guruhlarda)

2-keys.

Vebga asoslangan Onlayn Xarid Tizimi, mijoz buyurmani yetkazib beruvchidan bir yoki bir nechta tovarlarni sotib olishni so'rashi mumkin. Mijoz shaxsiy ma'lumotlarini beradi masalan manzil, kredit karta ma'lumotlari. Bu ma'lumotlar mijoz accountida saqlanadi. Agar kredit karta haqiqiyligi tasdiqlansa buyurtma yaratiladi va yerkazib beruvchiga yetkaziladi. Yetkazib beruvchi tovar mavjudligiga tekshiradi, buyurtmani tasdiqlaydi va yetkazib berish kunini belgilaydi. Qachonki buyurtma yetkazilganda mijozga xabar beriladi va mijozning kredit kartasidan mablag' yechib olinadi.

Keysni bajarish tartibi

2.1. Use Case modelini tuzing(individual)
2.2. Static modelini tuzing(individual)
2.3. Obyekt strukturasini tuzing(guruhlarda)
2.4. Dinamik modelini tuzing(individual)
2.5. Tizim arxitekturasi va dizaynini yarating(guruhlarda)
2.6. Tizimni amaliy ko'rinishini yarating (individual)
2.7. Tizimni testlang (guruhlarda)

3-keys.

Favqulotda holatlar monitoring tizimi bir nechta masofadan boshqariladigan monitoring tizimlari va monitoring sensorlaridan tashkil topadi. Turli xil sensorlar orqali tashqi muhit holati monitoring qilib boriladi. Sensorlarning bir nechtasi masofadan turib monitoring serveriga ma'lumot uzatadi. Tashqi muhit xavfli holatlarga duch kelganda odamlar sensorlar orqali ogohlantirishi mumkin. Ogohlantirishlar ogohlantrish servisida saqlanadi. Monitoring operatori turli-xil sensorlarning holatini ko'rib boradi va ogohlantirish holatlarini yangilab turadi.

Keysni bajarish tartibi

3.1. Use Case modelini tuzing(individual)
3.2. Static modelini tuzing(individual)
3.3. Obyekt strukturasini tuzing(guruhlarda)
3.4. Dinamik modelini tuzing(individual)
3.5. Tizim arxitekturasi va dizaynini yarating(guruhlarda)
3.6. Tizimni amaliy ko'rinishini yarating (individual)
3.7. Tizimni testlang (guruhlarda)

Muammo turi	Kelib chiqish sabablari	Hal etish yo`llari

***Keysdagi muammolarni shakllantirishda tinglovchilar (professor-**

VI. BO`LIM

VI. MUSTAQIL TA’LIM MAVZULARI

Mustaqil ishni tashkil etishning shakli va mazmuni

Tinglovchi mustaqil ishni muayyan modulni xususiyatlarini hisobga olgan xolda quyidagi shakllardan foydalanib tayyorlashi tavsiya etiladi:

- me’yoriy xujjatlardan, o’quv va ilmiy adabiyotlardan foydalanish asosida modul mavzularini o’rganish;
- tarqatma materiallar bo’yicha ma’ruzalar qismini o’zlashtirish;
- avtomatlashtirilgan o’rgatuvchi va nazorat qiluvchi dasturlar bilan ishlash;
- maxsus adabiyotlar bo’yicha modul bo’limlari yoki mavzulari ustida ishlash;
-tinglovchining kasbiy faoliyati bilan bog’liq bo’lgan modul bo’limlari va mavzularni chuqur o’rganish.

Mustaqil ta’lim mavzulari

1. Kompomentaga asoslangan dasturiy ta’minot injiniringi(Component-based software engineering).
2. Qayta foydalansa bo’ladigan dasturiy ta’monot(Software reuse).
3. Taqsimlangan dasturiy ta’minot injiniringi(Distributed software engineering).
4. Servisga yo’naltirilgan dasturiy ta’minot injiniringi(Service-oriented architecture).
5. Ajratilgan(Embedded) dasturiy ta’minotlar.
6. Aspect-oriented dasturiy injiniring.

VII. BO`LIM

VII. GLOSSARIY

Termin	O’zbek tilidagi sharhi	Ingliz tilidagi sharhi
Actor	Mavzu bilan o’zaro hamkorlikdagi o’ynalayotgan shaxs roli	A type of role played by an entity that interacts with the subject (e.g., by exchanging signals and data)
Adaptability	Dasturiy mahsulotning turli xil muhitlarga moslashuvchanligi	The capability of the software product to be adapted for different specified environments without applying actions
Agile software development	Ketma-ket oshib boruvchi dasturiy mahsulotlar ishlab chiqish metodologiyalar guruhi	A group of software development methodologies based on iterative incremental development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams
Artefact	Dasturiy ta’minot ishlab chiqarilish mobaynida olingan natijalardan biri	One of outcomes produced during the development of software
Attribute	Obyektning xarakteristikasi	A characteristic of an object.
Behavioral diagram	Tizimning xususiyatlarini ko’rsatuvchi UML diagrammasi	In UML a type of diagram that depicts behavioral features of a system or business process. This includes activity, state machine, and use case diagrams as well as the four interaction diagrams.
Business Analysis (BA)	Biznes ehtiyojlarni aniqlashda va biznes muammolarni yechishda talab qilingan vazifalar, bilim, uskunalar va usullar to’plami	The set of tasks, knowledge, tools and techniques required to identify business needs and determine solutions to business problems
Business Process	Alohida mijozlar yoki bozor uchun mahsulotlar ishlab chiqish faoliyatlari to’plami	A collection of activities designed to produce a specific output for a

		particular customer or market
Class	O’xshash obyektlar to’plamini tasvirlash	A class describes a set of objects that share the same specifications of features, constraints, and semantics.
Class diagram	Tizimining statik strukturasini ko’rsatuvchi diagramma. Unda tizimning sinflari, sinf xususiyatlari va metodlari va sinflar orasidagi munosabatlar ko’rsatiladi.	A type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes.
Component diagram	Tizimni tashkil qiluvchi komponentalar ko’rsatiladigan diagramma	In UML a diagram that depicts the components that compose an application, system, or enterprise.
Conceptual model	Dasturiy ta’minot yoki apparat ta’minotning texnologik xususiyatlarini tasvirlovchi model	A model describing technological software/hardware specifications.
Customer	Mahsulotni sotib oluvchi foydalanuvchi yoki tashkilot	Current or potential buyer or user of the products or service of an individual or organization, called the supplier, seller, or vendor.
Dependency	Boshqa komponentalar to’plamidagi bir komponentda to’plami	A dependency is a reliance of some kind, of one set of components on another set of components
Deployment diagram	Tizimning ishlash arxitekturasini ko’rsatuvchi diagramma	In UML a diagram that shows the execution architecture of systems.
Entity	Alohida mavjudlikka ega element yoki elementlar to’plami	An element or set of elements that has a distinct, separate existence, although it need not be a material existence.
Entity-relationship model (ERM)	Ma’lumotlarning abstrakt ko’rinishi. ERM borliqlar to’plamidan tashkil topadi.	An abstract and conceptual representation of data. Entity-relationship model consists of a set of entities,

		characterized by attributes and linked by relationships.
Error	Noto’g’ri natija chiqaradigan inson harakati	A human action that produces an incorrect result
Function	Tizim nima ish bajarishini tasvirlash.	A description of “what” a system does. A function has a corresponding implied purpose and is a fundamental part of a system description
Lifecycle	Mahsulotning hayotni bo’limlarga bo’lish	A partitioning of the life of a product or project into phases
Maintenance	Dasturiy mahsulot yetkazilganidan so’ng uni yaxshilash, muhitga moslashtirish	Modification of a software product after delivery to correct defects, to improve performance or other attributes, or to adapt the product to a modified environment
Modeling language	Axborotni ifodalashda foydalansa bo’ladigan ixtiyoriy sun’iy til	Any artificial language that can be used to express information or knowledge or systems in a structure that is defined by a consistent set of rules.
Non-functional requirement	Funksional bo’lmagan talablar	A requirement that does not relate to functionality, but to attributes such as reliability, efficiency, usability, maintainability and portability.
Object-oriented analysis and design	Tizimni birlashgan obyektlar guruhi ko’rinishida modellashtirish	A software engineering approach that models a system as a group of interacting objects.
Portability	Dasturiy mahsulotni bir muhitdan boshqa muhitga o’tkazganda oson ko’chish imkoniyati	The ease with which the software product can be transferred from one hardware or software environment to another
Process	Jarayon, bog’liq faoliyatlat to’plami	A set of interrelated activities, which transform

		inputs into outputs
Process requirement	Ishlab chiqish jarayoniga oid bo’lgan talablar	A requirement related to the development process.
Product requirement	Ishlab chiqish jarayoni mahsulotiga oid bo’lgan talablar. Mahsulotning sifatiga ta’sir ko’rsatadi.	A requirement related to the product of the development process. They affect quality of the product.
Product	Jarayondan chiquvchi natija	An output of a process
Quality	Tizimning talablarga javob berish darajasi	The degree to which a component, system or process meets specified requirements and/or user/customer needs and expectations
Requirement	Talab	Requirement

VIII. BO`LIM

ADABIYOTLAR RO`YXATI

VIII. ADABIYOTLAR RO'YXATI

Maxsus adabiyotlar

1. "Software Engineering", by Ian Sommerville, 2015, pages – 790.
2. Holdener, A. T. (2008). Ajax: The Definitive Guide. Sebastopol, Ca.: O'Reilly and Associates.
3. Abrial, J. R. (2005). The B Book: Assigning Programs to Meanings. Cambridge, UK: Cambridge University Press.
4. Anderson, R. (2001). Security Engineering: A Guide to Building Dependable Distributed Systems. Chichester, UK: John Wiley & Sons.
5. Baier, C. and Katoen, J.-P. (2008). Principles of Model Checking. Cambridge, Mass.: MIT Press.
6. Ball, T., Bounimova, E., Cook, B., Levin, V., Lichtenberg, J., McGarvey, C., Ondrusek, B., S. K., R. and Ustuner, A. (2006).
7. 'Thorough Static Analysis of Device Drivers'. Proc. EuroSys 2006, Leuven, Belgium. Ahern, D. M., Clouse, A. and Turner, R. (2001). CMMI Distilled. Reading, Mass.: Addison-Wesley.
8. Basili, V. and Green, S. (1993). 'Software Process Improvement at the SEL'. IEEE Software, 11 (4), 58–66.

Internet resurslar

1. http://www.SoftwareEngineering-9.com
2. http://www.pearsonhighered.com/sommerville
3. https://en.wikipedia.org/wiki/Software_engineer
4. http://www.ece.rutgers.edu/~marsic/books/SE/projects/