Г.Н. ФЕДОРОВА

БАҒДАРЛАМАЛЫҚ МОДУЛЬДЕРДІ БІРІКТІРУГЕ ҚАТЫСУ

«Білім беруді дамытудың федералды институты»
федералды мемлекеттік автономдық мекемесі «Компьютерлік жүйелерде бағдарламалау» мамандығы бойынша орта кәсіптік білім беру бағдарламаларын іске асыратын білім беру мекемелерінің оқу процесінде қолдану үшін оқу құралы ретінде
ұсынған.
Пікірдің тіркеу нөмірі №393 23 шілде 2015ж. «БДФИ» ФМАМ

ACADEMIA

Мәскеу
«Академия» баспа орталығы
2016

ӘӨЖ 004(075.32)
КБЖ 32.973.202ші723
Ф333

Бұл кітап Қазақстан Республикасының Білім және ғылым министрлігі және «Кәсіпқор» холдингі» КЕАҚ арасында жасалған шартқа сәйкес «ТжКБ жүйесі үшін шетел әдебиетін сатып алуды және аударуды ұйымдастыру жөніндегі қызметтер» мемлекеттік тапсырмасын орындау аясында қазақ тіліне аударылды. Аталған кітаптың орыс тіліндегі нұсқасы Ресей Федерациясының білім беру үдерісіне қойылатын талаптардың ескерілуімен жасалды.

Қазақстан Республикасының техникалық және кәсіптік білім беру жүйесіндегі білім беру ұйымдарының осы жағдайды ескеруі және оқу үдерісінде мазмұнды бөлімді (технология, материалдар және қажетті ақпарат) қолдануы қажет.

Аударманы «Delta Consulting Group» ЖШС жүзеге асырды, заңды мекенжайы: Астана қ., Иманов көш., 19, «Алма-Ата» БО, 809С , телефоны: 8 (7172) 78 79 29, эл. поштасы: info@dcg.kz

Пікір беруші -
СМАҚУ ЖКБ филиалының директоры, «Құрылыс» кафедрасының меңгерушісі,
доцент, техника ғылымдарының кандидаты *В.В.Кузьмин*

Федорова Г. Н.

Ф333 Бағдарламалық модульдерді біріктіруге қатысу: орта кәсіптік білім беру мекемелерінің студенттеріне арналған оқу құралы / Г.Н.Федорова. - М.: «Академия» баспа орталығы, 2016. – 304 б.
ISBN 978-601-333-335-9 (каз.)
ISBN 978-5-4468-2374-1 (рус.)

Оқу құралы «Компьютерлік жүйелерде бағдарламалау» мамандығының ПМ.03 «Бағдарламалық модульдерді біріктіруге қатысу» кәсіби модулін меңгеруге арналған.

Бағдарламалық қамсыздандыруды әзірлеудегі технологияларын ұйымдастыру принциптері, негізгі ережелері және даму перспективалары қарастырылған. Бағдарламалық өнімнің және оның сүйемелдеуші процестерінің өмірлік циклы ұғымы ашылған. Бағдарламалар мен бағдарламалық модульдерді біріктірудегі түрлі тәсілдер, бағдарламалардың сипаттамаларының өлшем әдістері, олардың тиімділігін бағалау сипатталған. Бағдарламалық қамсыздандыру сапасының стандарттары, бағдарламалық құжаттаманы әзірлеу әдістері мен құралдары берілген.

Орта кәсіптік білім беру мекемелерінің студенттеріне арналған.

ӘОЖ 004(075.32)
КБЖ 32.973.202ші23

© Федорова Г.Н., 2016
© «Академия» білім беру-баспа орталығы, 2016
© Рәсімдеу. «Академия» баспа орталығы, 2016

ISBN 978-601-333-335-9 (каз.)
ISBN 978-5-4468-2374-1 (рус.)

Құрметті оқырман!

Бұл оқу құралы «Компьютерлік жүйелерде бағдарламалау» мамандығы бойынша Оқу-әдістемелік жиынтықтың бөлігі болып табылады және «Бағдарламалық модульдерді біріктіруге қатысу» кәсіби модулін зерделеуге арналған.

Жаңа буынның оқу-әдістемелік жинақтары жалпы білім беру және жалпықәсіби пәндер мен кәсіби модульдерді танып білуді қамтамасыз етуге мүмкіндік беретін дәстүрлі және инновациялық оқу материалдарынан тұрады. Әрбір жинақ жалпы және кәсіби құзыреттілікті, сонымен қатар жұмыс берушінің талаптарын ескеруді меңгерту үшін қажетті оқулықтар мен оқу құралдарынан, оқыту және бақылау құралдарынан тұрады.

Оқу басылымдары электронды білім беру ресурстарымен толықтырылып отырады. Электронды ресурстар, интерактивті жаттығулары мен тренажерлары бар теориялық және практикалық модульдардан, мультимедиялық нысандар мен интернеттегі қосымша материалдар мен ресурстарға сілтемелерден тұрады. Жинаққа оқу процесінің негізгі параметрлерінде бекітілген: жұмыс уақыты, бақылау және тәжірибе жұмыстарының орындалу қорытындысы белгіленетін электронды журнал және терминологиялық сөздік енгізілген. Электронды ресурстарды оқу барысына енгізу жеңіл және оны басқа да оқу бағдарламаларына бейімдеп алуға болады

Алғысөз

Заманауи бағдарламалық қамсыздандыруды құру - бұл оған қатысатын мамандардың жоғары біліктілігін талап ететін ұзақ, логикалық күрделі және еңбекті көп қажетсінетін жұмыс. Ұсынылып отырған оқу құралының мақсаты - бағдарламалық қамсыздандыруды әзірлеу және пайдаланудың негізгі принциптері, сондай-ақ пайдаланушылар талаптарының артуына жауап беретін заманауи бағдарламалық өнімдерді алу үшін CASE-құралдарын пайдалану мүмкіндіктері жөнінде жүйелендірілген білімді қалыптастыру.

Оқу құралында бағдарламалық қамсыздандыруды әзірлеу технологияларын ұйымдастыру принциптері, негізгі ережелері мен даму перспективалары толық қарастырылған. Бағдарламалық өнім мен оның сүйемелдеуші процестерінің өмірлік циклының ұғымы айқындалған. Бағдарламалар мен бағдарламалық өнімдерді біріктіруге қойылатын түрлі тәсілдер ашылған. Бағдарламалардың сипаттамалары мен параметрлерін өлшеуге арналған құралдарға талдау жасалған, олардың тиімділігін бағалау әдістері қарастырылған. Бағдарламалық қамсыздандырудың сапа стандарттары, бағдарламалық құжаттаманы әзірлеу әдістері мен құралдары қарастырылған.

Оқу құралы Федералды мемлекеттік білім беру стандартының ОКБ талаптарына сәйкес «Компьютерлік жүйелерде бағдарламалау» мамандығы бойынша «Бағдарламалық модульдерді біріктіруге қатысу» кәсіби модулі бойыша даярланған және автордың практикалық қызметі процесінде жиналған материалдарға негізделеді.

Оқу құралы үш тараудан тұрады, олардың әрқайсысы кәсіби модульдің құрамындағы пәнаралық курсқа сәйкес келеді.

1. «Бағдарламалық қамсыздандыруды әзірлеу технологиясы».
2. «Бағдарламалық қамсыздандыруды әзірлеудің аспаптық құралдары».
3. «Құжаттау және сертификаттау».

«Бағдарламалық қамсыздандыруды әзірлеудің аспаптық құралдары» және «Құжаттау және сертификаттау» пәнаралық курсы практика-

бағдарлық сипатта, сол себепті оқу құралындағы теориялық материалдың негізгі көлемі «Бағдарламалық қамсыздандыруды әзірлеу технологиясы» бірінші тарауында берілген.

Тоғыз бөлімнен тұратын «Бағдарламалық қамсыздандыруды әзірлеу технологиясы» оқу құралының **I тарауында** халықаралық стандарттарға сәйкес қабылданған бағдарламалық жүйелердің өмірлік циклының ұғымы ашылады, жүйелі түрде өмірлік циклының барлық сатылары сипатталады, бағдарламалық қамсыздандыру сапасын қамтамасыз ету мәселелерінен, бағдарламалық өнімді тестіден өткізу технологиялары мен әдістерінен, оның тапсырыс берушіге жеткізілу процедурасынан хабардар етіледі. Ұсынылып отырған материалдың суреттер көлемі жеткілікті.

1-бөлім бағдарламалық қамсыздандырудың өмірлік циклын, оның негізгі, қосалқы және ұйымдастыру процестерін, сондай-ақ олардың арасындағы өзара байланысты қарастыруға арналған.

2-бөлімде бағдарламалық қамсыздандыруды әзірлеудің стратегиялары (каскадты, инкрементті, эволюциялық) беріледі, сондай-ақ бағдарламалық жүйелердің өмірлік циклының түрлі модельдері қарастырылады. Олардың салыстырмалы сипаттамалары беріледі және оларды қолдаушы процестері сипатталады.

3-бөлімде бағдарламалық қамсыздандыруға қойылатын талаптарды басқару жөніндегі мәліметтер, пәндік салаға тексеру жүргізу, тапсырыс берушінің талаптары бойынша функционалдық және пайдалану сипаттамаларын құру әдістері ашылады. Бұдан басқа, осы бөлімде бағдарламалық қамсыздандыру прототиптерін құру принциптері мен пайдалану артықшылықтары беріледі.

4-бөлімде бағдарламалық қамсыздандыруды жобалау мен әзірлеудің құрылымдық тәсілінің мәні түсіндіріледі, атап айтқанда, функционалдық диаграммаларды құру үшін SADT әдіснамасын қолдану мүмкіндіктері және DFD деректері ағынының диаграммаларын құру бірізділігі сипатталады.

5-бөлімде бағдарламалық қамсыздандыруды әзірлеудегі нысанға бағытталған тәсілге арналан. Мұнда UML модельдеу біріздендірілген тіліне сипаттама беріледі және UML кейбір диаграммаларын құру процедуралары, мысалы әрекет диаграммасы, бірізділік диаграммасы, пайдалану нұсқаларының диаграммасы және басқаларын құру процедурасын түсіндіреді.

6-бөлімде бағдарламалық қамсыздандыруды іске асыру кезеңдері қарастырылады, бағдарламалық жүйелердің түрлі сәулеті, модульді бағдарламалау принциптері мен модульдердің қасиеттері сипатталады. Бағдарламалық қателердің жіктемесі беріледі, бағдарламалық қамсыздандыруды дұрыстау тәсілдері ашылады, оның ішінде бағдарламалардың өрлемелі және төмендемелі әзірлемелердің әдістері

түсіндіріледі. Пайдаланушы интерфейсін құруға қойылатын талаптар беріледі.

7-бөлім бағдарламалық өнімнің сапасын қамтамасыз ету мәселелері және оның сенімділігін арттыру тәсілдеріне арналған. Бағдарламалық қамсыздандыру сапасын арттыру үшін өнімді құру процесі мен сүйемелденуін жақсарту қажет. Бағдарламалық жобалардағы метрикалар әзірлеушілер үшін уақытылы және дұрыс шешім қабылдауына көмек көрсетуге арналған. Бұл бөлімде бағдарламалық қамсыздандыруды әзірлеу процесіне сандық бағалау тәсілдері сипатталған, атап айтқанда жинау, талдау және өлшеуді қолдану әдістері мен өлшем модельдерін сипаттайды. Бағдарламалық өнімнің түрлі әзірлеу сатыларында сенімділігін қамтамасыз етуге, сондай-ақ оны құру барысында пайда болатын тәуекелге талдау жасауға басым назар аударылады.

8-бөлімде бағдарламалық қамсыздандыруды тестіден өткізу - бағдарламалық қамсыздандырудың талаптарға сәйкестігін тексеруге байланысты мәселелер қарастырылады. Бағдарламалық қамсыздандырудың өнімділігін жүктеу тесті, тұрақты тесті және басқа тестілеу түрлеріне ерекше назар аударылған.

9-тарау бағдарламалық қамсыздандыруды енгізу және пайдалану туралы мәліметтерден тұрады. Мұнда бағдарламалық қамсыздандыруды жеткізу сипаттамасы және өнімнің нұсқаларын басқару процедурасы сипатталады.

«Бағдарламалық қамсыздандыруды әзірлеудің аспаптық құралдары» **II тарауында** бағдарламалық қамсыздандыруды құруды қолдайтын аспаптық құралдармен жұмыс істеу құрылымы мен тәсілдері берілген. Бұл оқулықта автор бағдарламалық қамсыздандыруды әзірлеу құралдарын тиімді қолдану технологияларына шоғырландырылған бағдарламалау тілдері бойынша анықтамалық материалды толығымен шығарып тастайды.

10-бөлімде бағдарламалық қамсыздандыруды әзірлеудің аспаптық құралдары туралы жалпы түсінік беріледі. Сәулеті мен мүмкіндіктері жеткіліті түрде толық қарастырылады. NET Framework - бағдарламалық платформасы түрлі бағдарламалау тілдері үшін келетін жалпы орындау ортасы болып табылады. Қазіргі таңда бұл платформа әзірлеушілер арасында кең танымалдылыққа ие.

11-бөлімде заманауи CASE-құралдары - құру принциптері мен негізгі функционалдық мүмкіндіктері туралы мәліметтер беріледі. Мұнда CASE-құралдарының жіктемесі, олардың негізгі құрамдас бөлшектері мен функциялары, бағдарлама коды, пайдаланушы интерфейсі, құжаттамасы генерациясының құралдары берілген. Заманауи CASE-құралдарына шолу беріледі.

«Құжаттау және сертификаттау» **III тарауында** бағдарламалық

өнімдер метрологиясы мен оларды әзірлеуге және қолдануға арналған құжаттарды құрудың негізгі ережелері қарастырылады. Берілген пәнаралық курсты зерделеу үшін негізгі ақпарат көздері бағдарламалық қамсыздандыруды әзірлеу және пайдалану саласындағы халықаралық және ұлттық стандарттар болып табылады. Оқулықтың бұл тарауында бағдарламалық қамсыздандыруды әзірлеу және оны құжаттау кезінде осы стандарттарды қолдануға қатысты кейбір түсініктер беріледі.

12-бөлімде стандарттаудың негізгі міндеттері мен принциптері қалыптастырылған, стандарттау жөніндегі негізгі құжаттар - РФ мемлекеттік стандарттары, халықаралық (өңірлік) стандарттар, стандарттау жөніндегі қағидалар, нормалар мен ұсыныстар, жалпы ресейлік сыныптамалар және т.б. қарастырылған.

13-бөлім бағдарламалық қамсыздандыруды ресейлік стандарттарына сәйкес құжаттау мәселелерін қарастыруға арналған. Бағдарламалық қамсыздандыру сапасына қойылатын талаптардың артуы халықаралық стандарттарды құру мен оларды белсенді қолдануды ынталандырады. Бұл бөлімде ЕСПД және МемСТ Р стандарттарының жалпы сипаттамасы берілген. Ұсынылған стандарттар бағдарламалық қамсыздандырудың өмірлік циклының процестерін реттейді, сипаттамаларын сипаттайды, бағдарламалық қамсыздандыруды әзірлеуге арналған техникалық тапсырманы құруды, бағдарламалық және пайдалану құжаттарының және пайдаланушы құжаттамасын құруды реттейді.

14-бөлім бағдарламалық қамсыздандыруды сертификаттау мәселелеріне арналған. Мұнда сертификаттаудың нормативтік-құқықтық негіздері берілген және бағдарламалық өнімді сертификаттауды жүргізу тәртібі сипатталады.

Оқу құралының әрбір бөліміне студенттің зерделенген материалды бекітуіне мүмкіндік беретін өзіндік жұмысына арналған бақылау мәселелері қамтылған.

Оқу құралын «Бағдарламалық модульдерді біріктіруге қатысу» кәсіби модулін, сондай-ақ бағдарламалық қамсыздандыруды әзірлеу мен құжаттауға байланысты өзге модульдер мен пәндерде зерделеу кезінде оқытушылар мен аталған мамандық бойынша студенттер қолдана алады.

Ұсынылатын материал бағдарламалық өнімдерді әзірлеумен айналысатын мамандардың кең ауқымды тобы үшін пайдалы бола алады.

I

ТАРАУ

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ӘЗІРЛЕУ ТЕХНОЛОГИЯСЫ

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ ӨМІРЛІК ЦИКЛЫ

1.1. НЕГІЗГІ ТЕРМИНДЕР ЖӘНЕ АНЫҚТАМАЛАР

Әдетте ***бағдарлама*** деп компьютердің есепті шешу нәтижелерін алу үшін орындайтын командаларының (операторларының, нұсқаулықтарының) бірізділігі түсіндіріледі.

Бағдарламалық құралдар (БҚ) деп бағдарламаны немесе деректердің машиналы тасымалдағыштарында болатын және құжаттамамен жабдықталған бағдарламалардың логикалық байланысты жиынтығын атайды.

Бағдарламалық қамсыздандыру (БҚ) бағдарламалық құралдардың және кез келген қызмет саласындағы міндеттерді компьютерде шешуге, сондай-ақ ЭЕМ аппараттық құралдарының жұмыс істеуін қамтамасыз етуге мүмкіндік беретін оларды қолдаушы құжаттамалардың жиынтығы ретінде анықтама беруге болады.

Бағдарламалық қамсыздандыру техникалық (аппараттық), математикалық, ақпараттық, лингвистикалық, ұйымдастырушылық және әдістемелік қамтамасыз етумен қатар есептеу жүйесімен қамтамасыз етудің бір түрі ретінде қарастырылуы мүмкін.

Бағдарламалық өнім (БӨ) - өнеркәсіптік кез келген өнім түрі ретінде сатуға даярланған, көпшілік сұраныстың белгілі бір проблемаларын (міндеттерін) шешу үшін өзара байланысты бағдарламалық құралдар мен оларды қолдайтын құжаттамалардың кешені.

Бағдарламалық қамсыздандыру мен бағдарламалық өнімге берілген осы анықтамалардағы ерекшелік – тек соңғысының сатуға даярланған және көпшіліктің сұраныс міндеттерін шешуге арналғандығы болады. Бұдан әрі бұл терминдер бір ұғым ретінде қолданылатын болады, яғни көпшілік сұраныстың және сату аспектісі болып қарастырылмайды.

Бағдарламалық өнімдерге қарағанда ***тәжірибелік бағдарламалар*** («өзіне арналған бағдарламалар») әзірлеушілердің қажеттіліктерін қанағаттандыруға арналған. Көбінесе тәжірибелік бағдарламалар деректерді өңдеу технологиясында сервистің ролін орындайды, не болмаса кеңінен таралуға арналмаған функционалдық міндеттерді шешудегі бағдарламалар болып табылады.

Бағдарламалық қамсыздандыруды әзірлеу технологиясы деп берілген қасиеттер арқылы бағдарламалық өнімді алуды берілген жағдайларда қамтамасыз ететін бағдарламалау процесін жүргізудің оңтайлы тәсілдері туралы жалпы және жүйелі білімдерінің жиынтығы айтылады.

Бағдарламалық қамсыздандыруды әзірлеу технологиясы бағдарламалау процесінің мазмұнын кең ауқымды - кейбір бағдарламаларды құруға қажеттіліктің пайда болуынан оны шығаруға дейінгі, пайдаланушыға берілуі, пайдалану процесінде түрін өзгерту мен моральды ескіру салдарынан қолданылуын тоқтатуды қамтиды.

Кез келген бағдарламалық қамсыздандыруды әзірлеу болашақ бағдарламалық өнімге қойылатын талаптарды талдаудан басталады. Талдау нәтижесінде әзірленетін бағдарламалық қамсыздандырудың сипаттізімін алады.

Ерекшелік деп әзірленетін бағдарламалық қамсыздандырудың функциялары мен шектеулерін нақты сипаттауды атайды.

Бағдарламалық қамсыздандыру өзара байланысты бағдарламалық модульдермен құрылған ішкі ұйымдасуға немесе ішкі құрылымға ие. Бұл *бағдарламалық жүйелер* деп аталатын күрделі және көп функциялы бағдарламалық өнімдер үшін әділетті. Бағдарламаларды жеке құрамдас бөліктерге құрылымды түрде бөлу оларды құрудың аспаптық құралдарын таңдаудың негізі болып табылады, дегенмен кері ықпалы да орын алады. Бағдарламалық қамсыздандыруды әзірлеудің аспаптық құралдарын таңдау бағдарламалық модульдердің түрлерін айқындайды. Бағдарламалық өнімдерді құрған кезде бірнеше рет қолданылатын модульдер бөлінеді, оларды типтендіру мен бірегейлендіру жүргізіледі, оның есебінен жалпы бағдарламалық қамсыздандыруды әзірлеуге жұмсалатын еңбек шығындары мен мерзімдері қысқартылады. Кейбір бағдарламалық өнімдер стандартты бағдарламалардың дайын кітапханаларынан, процедуралардан, қосалқы бағдарламаларынан, нысандардан, деректерді өңдеу әдістерінен алынған модульдерді қолданады.

Көзбен шолу арқылы бағдарламалау - бұл бағдарламаны мәтін жазудың орнынан графикалық нысандар арқылы әрекет жасасумен құру тәсілі.

Кез келген бағдарламалық өнім өзі құрылған функцияларды орындауы тиіс. Сапалы БӨ оны ұзақ уақыт бойы пайдалануға мүмкіндік беретін бірқатар қасиеттерге ие болуы тиіс.

Бағдарламалық қамсыздандырудың сапасы - бұл бағдарламалық қамсыздандырудың пайдаланушы белгілеген қажеттіліктерін қанағаттандыру қабілетіне ықпал ететін оның сипаттамаларының жиынтығы. Бұл, алайда түрлі бағдарламалық өнімдер сандық көрсеткіштерінің бірдей мәндері бар қасиеттері жиынтығына ие болуы тиіс.

Техникалық құрылғылар жағдайында сияқты сапа көрсеткіштері қарама-қарсы болып табылады, бұл мынаны білдіреді: бір сапа көрсеткішінің жақсаруы басқалардың нашарлауынан қол жеткізілуі мүмкін. Егер де қасиеттерінің сандық көрсеткіштері оның табысты түрде қолданылатындығына кепілдік берсе, бағдарламалық қамсыздандырудың сапасы қанағаттандырылған болып табылады.

Бағдарламалық қамсыздандыру сапасының негізгі белгілеріне сипаттама келтіреміз.

Функционалдылық - бағдарламалық қамсыздандырудың сыртқы сипаттамасымен айқындалған функциялардың жиынтығын орындау қабілеті.

Сенімділік - бағдарламалық қамсыздандырудың жоғары дәрежедегі ықтималдылықпен берілген уақыт кезеңінде берілген жағдайларда берілген функцияларды тоқтаусыз орындау қабілеті. Сенімділік қатесіз дегенді білдіреді, сенімді бағдарламалық қамсыздандыру үшін қателердің бағдарламалық қамсыздандыруды қолданған кезде сирек шығуы және апатты салдарға әкелмеуі маңызды.

Қолданудағы жеңілдігі – бұл пайдаланушының бастапқы деректерді даярлау мен енгізуге жұмсалатын шығындарын азайту қабілеті, сондай-ақ пайдаланушының оң эмоциясының пайда болуы.

Тиімділігі - бұл қолданылатын есептеу ресурстарының көлеміне бағдарламалық қамсыздандыру ұсынатын қызметтердің деңгейінің қатысы. Қолданылатын есептеу ресурстарының көлемі сандық түрде машина уақытының шығындарымен және берілген функцияны орындауға жедел жадымен айқындалады.

Бағдарламалық өнімдердің ұтқырлығы - деректерді өңдеу жүйесінің техникалық кешеніне, операциялық ортаға, деректерді өңдеудің желілік технологияларға, пәндік саланың сипаттамасына және т.с.с. тәуелсіздігі. Ұтқыр (көп платформалы) бағдарламалық өнім түрлі компьютер модельдерінде және есептеу желісінің жағдайларындағы оның пайдаланылуы шектеусіз операциялық жүйелерге орнатылуы мүмкін.

Бағдарламалық қамсыздандырудың түр өзгергіштігі - өзгерістерді енгізуге қабілеті, мысалы өңдеу функцияларының кеңеюі, басқа техникалық өңдеу базасына өту және т.б.

Бағдарламалық өнімнің коммуникативтілігі оның басқа бағдарламалармен бірігуінің максималды мүмкіндігіне, деректермен алмасуын қамтамасыз етуге (деректер қорының экспорты/импорты, өңдеу нысандарын енгізу немесе байланыстыру және т.б.) негізделген.

Бағдарламалық қамсыздандырудың сүйемелденуі - пайдаланушының өзгеріп отыратын қажеттіліктеріне сәйкес бағдарламалық қамсыздандырудың қателерін жою немесе түрін өзгерту жөніндегі күш-жігерді азайтуға мүмкіндік беретін бағдарламалық қамсыздандырудың сипаттамалары.

Дәлдік - нәтижелердің олардың тағайындалуына сәйкес орындалу дәлдігін айқындайтын сипаттама. Мысалы, егер бағдарламада банкі операциялары бойынша есептер жүргізілсе, онда саналы түрдегі дәлдік үш таңбадан кейін келесілерді екі таңбаға дейін дөңгелектеумен құрылады. Егер де бағдарламалада молекулярлық деңгейде биологиялық сараптамалар бойынша есептер жүргізілсе, онда дәлдік 10-12 ондық таңбаға дейін қажет болуы мүмкін.

Қателерге тұрақтылық - бағдарламалық өнімнің кіріс деректерінің кез келген жағдайда жұмыс істеуі немесе жабдықтармен өзара байланыста қателердің болуы. Бұл тараптың айқындылығы ғарыш саласында бағдарламалық қамсыздандыру тұрақтылығының қажеттілігімен көрнекі түрде беріледі.

Ақпараттылық - бұл бағдарламалық қамсыздандырудың түсініктілігін қамтамасыз ететін сипаттамалардың бірі. Егер ақпарат жеке құрамдас бөлшектердің жұмысының мағыналық нәтижесінің мәнін, қабылданған шектеулерді, бағдарламалық қамсыздандырудың тағайындалуын қамтамасыз ететін ұғымнан тұрса, бағдарламалық өнім ақпараттылық сипаттамасына ие бола алады.

Бағдарламалық қамсыздандырудың ***ашықтығы*** оның бастапқы коды қолжетімді болғанда бағдарламалық қамсыздандырудың әрбір операторының тағайындалуын түсінуге мүмкіндік береді (мысалы, әрбір бағдарлама сәйкестендіргішінің мағыналық жүктемесі болуы керек).

Бағдарламалық қамсыздандырудың сипаттамасы ретіндегі ***келісушілік*** сыртқы және ішкі болады. Ішкі келісушілік бірыңғай терминологияны, ұғымдар мен мәндердің бірыңғай түсіндірілуін қамтамасыз етуі тиіс. Бұл сипаттама бағдарламалық кешендер құрылған кезде, мамандар тобы жобамен жұмыс істегенде және өзара байланысқан бағдарламалық модульдер бойынша байланыс қажет болатын жұмыс процесінде ерекше өз мәніне ие болады. Сыртқы

келісушілік құрылатын бағдарламалық қамсыздандырудың оның әзірленуіне арналған техникалық жобасында берілген талаптарға, сондай-ақ осы немесе басқа да салалық стандарттарға сөзсіз сәйкестігімен қамтамасыз етіледі.

1.2. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ ӨМІРЛІК ЦИКЛЫНЫҢ САТЫЛАРЫ

Өмірлік цикл (ӨЦ) ұғымы бағдарламалық қамсыздандыруды (БҚ) жобалау әдіснамасының негізгі ұғымдарының бірі болып табылады.

Бағдарламалық қамсыздандырудың өмірлік циклы (БҚ ӨЦ) бағдарламалық қамсыздандыруды құру туралы шешім қабылданған сәттен бастап оны пайдаланудан толық шығару сәтінде аяқталатын үздіксіз процесті білдіреді.

Бағдарламалық қамсыздандырудың өмірлік циклының ықтимал құрылымы туралы түсінік алу үшін ең алдымен технологиялық процестерді сипаттайтын тиісті стандарттарға жүгінеміз.

БҚӨЦ регламенттейтін халықаралық стандарттар ISO/IEC 12207 (ISO - International Standards Organization, Стандарттау жөніндегі халықаралық ұйым; IEC - International Electrotechnical Commission, Электротехника жөніндегі халықаралық комиссия) болып табылады. ISO/IEC 12207 стандарты бағдарламалық өнімнің құрылу кезінде орындалуы тиіс процестері, әрекеттері мен міндеттерін қоса алғандағы өмірлік циклы құрылымын айқындайды. Ресей аналогы ГОСТ Р ИСО/МЭК 12207 «Ақпараттық технология. Жүйелік және бағдарламалық инженерия. Бағдарламалық құралдардың өмірлік цикл процестері» болып табылады.

Бағдарламалық қамсыздандырудың өмірлік циклын оны құру мен пайдалану процесінде БҚ болатын оқиғалар қатары ретінде түсінуге болады. Бағдарламалық қамсыздандыруды әзірлеу, негізінен, белгілі бір тақырыптық сала үшін орындалады. Тақырыптық саланың және мұндағы бағдарламалық қамсыздандырудың жұмыс істеу ерекшеліктері сөзсіз, БҚ құрамына ықпалын тигізеді. Бағдарламалық қамсыздандыру жоба сияқты әзірленеді. Жобаны басқарудағы көптеген ерекшеліктер мен жобаны әзірлеу фазалары (өмірлік цикл фазалары) жалпы тақырыптық салаға, сонымен бірге жоба сипатына байланысты емес болып табылады. Әрбір жоба, оны орындауға қажетті жұмыстардың күрделілігі мен көлеміне қарамастан өзінің дамуында белгілі бір

күйлерден өтеді. Идея туындағаннан бастап жобаны аяқтағанға дейінгі даму сатыларының жиынтығын сатыларға немесе кезеңдерге бөлу қалыптасқан.

Сатыларының санын және олардың мазмұнын анықтауда кейбір ерекшеліктер бар, себебі бұл сипаттамалар көбінесе нақты жобаны жүзеге асыру шарттары мен негізгі қатысушылардың тәжірибесіне байланысты. Бұған қарамастан, ақпараттық жүйені әзірлеу процесінің логикасы мен негізгі мазмұны көптеген жағдайларда жалпы болып табылады.

Бағдарламалық қамсыздандырудың келесі даму сатыларын бөліп көрсетуге болады:

1) тақырыптық саланы талдау және талаптарды (тұжырымдамаларды) қалыптастыру;
2) жобалау;
3) іске асыру;
4) тестіден өткізу;
5) пайдалануға енгізу;
6) пайдалану (жобаны сүйемелдеу).

Бағдарламалық қамсыздандырудың өмірлік циклы оның пайдаланудан шығарылуымен аяқталады.

Әрбір саты үшін орындалатын жұмыстардың құрамы мен бірізділігі, алынатын нәтижелері, жұмыстарды орындауға қажетті әдістері мен құралдары, қатысушылардың рөлдері мен жауапкершіліктері және т.б. айқындалады.

Бағдарламалық өнімнің ӨЦ мұндай қалыпты сипаттамасы ұжымдық әзірлеу процесін жоспарлап, ұйымдастыруға және осы процестерді басқаруды қамтамасыз етуге мүмкіндік береді. Сатылардың әрбірін толығырақ қарастырайық.

БҚ қойылатын талаптарды қалыптастыру сатысы маңыздылардың бірі болып табылады, себебі бүкіл жобаның табысын айқындайды. Бұл сатыда жобаның мақсаттары мен міндеттері қалыптасады, БҚ қолданылу саласы белгіленеді және шектік шаралары айқындалады, олардың арасындағы базалық мәні мен өзара байланыстары бөлінеді. Әзірленетін БҚ өзара байланысуы болуы тиіс барлық сыртқы нысындар сәйкестендіріледі, және де бұл өзара байланыстың жоғарғы деңгейдегі сипаты айқындалады, яғни бағдарламалардың барлық функционалдық мүмкіндіктері сәйкестендіріледі және олардың ішіндегі маңыздыларына сипаттама беріледі. БҚ әзірлеу құны мен мерзімдері анықталады, оның әзірлеуге арналған техникалық тапсырма қалыптастырылады және қол қойылады. Осылайша, бағдарламалық өнімді бұдан әрі жобалау үшін негіздемесі

құрылады.

Талаптарды қалыптастыру және талдау сатысы келесі сатыларды қамтиды:

1) жұмыстарды жоспарлау. Бұл сатының негізгі міндеттері болып табылатындар:
   - әзірлеу мақсаттарын айқындау;
   - жобаны алдын-ала экономикалық бағалау;
   - жұмыстарды орындаудың жоспар-кестесін құру;
2) автоматтандырылатын нысанға тексеру жүргізу, оның шеңберінде жүзеге асырылады:
   - бағдарламалық қамсыздандыруға қойылатын талаптарды алдын-ала анықтау;
   - сыртқы көздерден келіп түсетін ақпаратқа талдау;
   - ішкі ақпараттық ағындарды талдау;
   - тақырыптық сала модельдерін тексеру нәтижелерінің негізінде құру;
3) қолданыстағы ұқсас бағдарламалық өнімдерге талдау.

Жобалау сатысы, негізінен бағдарламалық жүйенің сәулетін, оның функцияларын, сыртқы жұмыс істеу шарттарын, пайдаланушылар мен жүйе арасындағы функцияларды бөлу мен интерфейстерді, бағдарламалар мен ақпараттық құрамдас бөлшектерге қойылатын талаптарды анықтаудан тұрады. Жүйені жобалау талаптарды қалыптастыру нәтижелерінің негізінде жүргізіледі. Жобалау әдіснамасы жүйенің физикалық, логикалық, сондай-ақ динамикалық және статикалық модельдерінің тәсілдерін қамтиды. Бағдарламалық қамсыздандырудың функционалды ерекшелігі әзірленеді, жүйенің құрылысы таңдап алынады, көбірек келетін ДҚБЖ айқындалады, деректерді сақтау құрылымы жобаланады, аппараттық қамтамасыз етуге қойылатын талаптар келісіледі, бағдарламалық қамсыздандыруды енгізуге қажетті ұйымдастырушылық іс-шараларының жиынтығы, сондай-ақ оның қолданылуын реттейтін құжаттар тізбесі анықталады. Пайдалану құжаттамасы рәсімделеді.

Іске асыру сатысында тұтас бағдарламаның, сондай-ақ оның бөліктерінің прототиптері құрылады, деректердің құрылымын нақты жүзеге асырылады, бағдарламаның кодтары әзірленеді, ретке келтіруші тестілері орындалады, техникалық құжаттамасы құрылады. Іске асыру сатысының нәтижесінде өнімнің жұмыс нұсқасы пайда болады.

Бағдарламалық өнімді *тестіден өткізу* жобалау және іске асыру сияқты әзірлеу сатыларымен тығыз байланысты. Жүйеде арнайы механизмдер құрылады, олар бағдарламалық қамсыздандырудың оның талаптарына сәйкестігіне тестіден өткізу жүргізуге мүмкіндік береді,

қажетті құжаттама пакетін рәсімдейді және оның болуын тексереді. Тестіден өткізу нәтижесі бағдарламалық өнімнің барлық кемшіліктерін жою және оның сапасы туралы қорытынды болып табылады.

Бағдарламалық қамсыздандыруды *енгізу (пайдалануға енгізу)*, әдетте, келесі қадамдарды көздейді:

- бағдарламалық жүйені орнатуды;
- пайдаланушыларды оқытуды;
- құжаттауды (тиісті бұйрықтарды, қабылдау актілерін және т.с.с. беру).

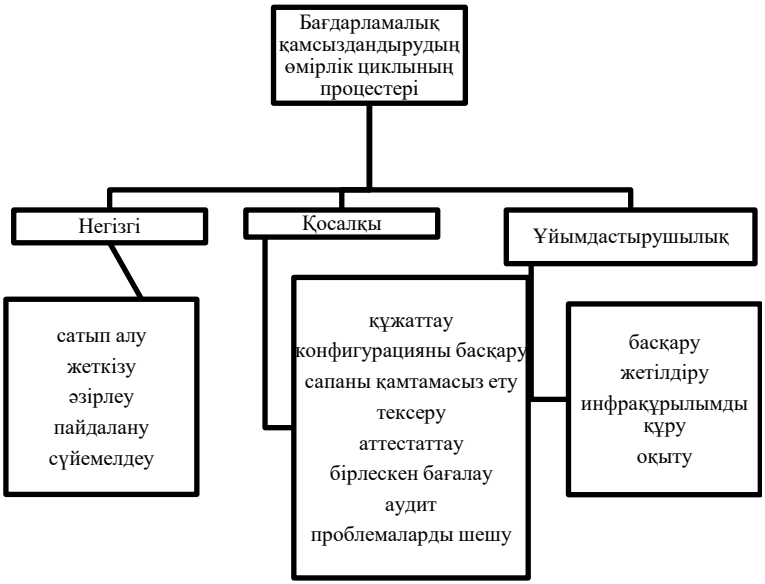
Кез келген әзірлемеге толық құжаттама пакеті қоса беріледі, ол бағдарламалық өнімнің сипаттамасынан, пайдаланушы нұсқаулығынан және жұмыс алгоритмінен тұрады. Бағдарламалық қамсыздандырудың жұмыс істеуін сүйемелдеу әзірлеушінің техникалық сүйемелдеу тобымен жүзеге асырылуы тиіс.

Сүйемелдеу - бұл жеткізілетін бағдарламалық қамсыздандырудың жаңа жағдайларға бейімделу, бағдарламалық қамсыздандыруға және оның негізгі функцияларының өзгертілмей сақталған кезде түрін өзгертуге қажеттіліктермен немесе пайда болған проблемалармен туындаған өзгерістерді енгізу процесі.

Бағдарламалық қамсыздандырудың өмірлік циклының әрбір сатысының шекаралары кейбір уақыт мезетімен анықталған, мұнда белгілі бір қиын шешімдерді қабылдау және белгілі бір түйінді мақсаттарға қол жеткізу қажет болады.

1.3. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ ӨМІРЛІК ЦИКЛЫНЫҢ ПРОЦЕСТЕРІ

Бағдарламалық қамсыздандырудың өмірлік циклы бойында өтетін барлық процестер әрекеттер жиынтығына бөлінген, әрбір әрекет - міндеттердің жиынтығына бөлінген. Бұрын айтылған ГОСТ Р ИСО/МЭК 12207-2010 «Ақпараттық технология. Жүйелік және бағдарламалық инженерия. Бағдарламалық құралдардың өмірлік циклының процестері» стандарты осы әрекеттерді айқындайды. Бұл стандарт қалыптасқан терминологияны қолдана отырып, бағдарлама индустриясымен бағдарланатын бағдарламалық құралдардың өмірлік цикл процестерінің жалпы құрылымын белгілейді. Стандарт бағдарламалық өнімді немесе қызметті сатып алынғанда, сондай-ақ бағдарламалық өнімдерді жеткізу, әзірлеу, мақсатына қарай қолдану, сүйемелдеу мен қолданылуын тоқтатқан кезде қолданылатын процестерді, қызмет түрлері мен міндеттерді айқындайды.



1.1-сурет. БҚ ӨЦ процестерінің топтары

ISO/IEC 12207-2010 сәйкес барлық процестер үш топқа бөлінген: негізгі, қосалқы, ұйымдастырушылық (1.1-сурет).

Процесс кейбір кіріс деректерін шығыс деректеріне түрлендіретін байланысты әрекеттердің жиынтығы ретінде айқындалады.

Әрбір процесс белгілі бір міндеттермен және оларды шешу әдістерімен, басқа процестерден алынатын бастапқы деректермен және нәтижелермен сипатталады.

Әрбір процесс әрекеттер жиынтығына, әрбір әрекет міндеттер жиынтығына бөлінген. Әрбір процесс, әрекет немесе міндет басқа процеспен қажеттілігіне қарай басталады және орындалады, мұнда алдын-ала белгіленген орындау реттілігі жоқ (әрине, кіріс деректері бойынша байланыстарды сақтағанда).

Өмірлік циклдың негізгі процестері

Бағдарламалық қамсыздандырудың өмірлік циклының бес негізгі процесі белгіленеді:

1) сатып алу;

2) жеткізу;
3) әзірлеу;
4) пайдалану;
5) сүйемелдеу.

Процестің *негізгі қатысушысы* ретінде бағдарламалық бұйымның әзірленуін, пайдаланылуын немесе сүйемелденуін бастайтын немесе орындайтын тарап аталады. Бұл бағдарламалық бұйымды сатып алушы, әзірлеуші, пайдаланушы қызметкер және сүйемелдеу қызметкері.

Сатып алу процесі бағдарламалық өнімді сатып алушы кәсіпорынның әрекеттерін айқындайды.

Бұл процесс төменде аталған әрекеттерді қамтиды:

1) сатып алуға бастама болу;
2) өтінім ұсынымдарын даярлау;
3) шарттарды даярлау және түзету;
4) жеткізушінің қызметін қадағалау;
5) жұмыстарды қабылдау және аяқтау.

Сатып алуға бастама болу мына міндеттерді қамтиды:

- тапсырыс берушінің бағдарламалық өнімді сатып алуға өзінің қажеттіліктерін анықтау;
- жүйеге қойылатын талаптарға талдау;
- сатып алуға қатысты шешімдерді қабылдау;
- бағдарламалық қамсыздандыруды сатып алған жағдайда қажетті құжаттаманың, кепілдіктердің, сертификаттардың, лицензиялардың және сүйемелдеудің болуын тексеру;
- жүйеге қойылатын талаптарды, шарттың үлгісін, тараптардың жауапкершіліктерін қамтитын сатып алу жоспарын даярлау және бекіту.

Өтінім ұсынымдары мыналардан тұрады:

- жүйеге қойылатын талаптар;
- бағдарламалық өнімдердің тізбесі;
- шарттар мен келісімдер;
- техникалық шектеулер (мысалы, жүйенің жұмыс істеу ортасы).

Өтінім ұсыныстары таңдап алынған жеткізушіге жіберіледі. Жеткізуші - бұл шартта келісілген шарттарда бағдарламалық қамсыздандыруды немесе бағдарлама қызметін, жүйені жеткізуге тапсырыс берушімен шарт жасасатын ұйым.

Шартты *даярлау және түзету* келесі міндеттерді қамтиды:

- тапсырыс берушінің жеткізушіні, ықтимал жеткізушілердің ұсынымдарын бағалау белгілерін қамтитын таңдау процедурасын анықтау;
- ұсынымдарды талдау негізінде нақты жеткізушіні таңдау;

- жеткізушілермен шартты даярлау және жасасу;
- шартқа оны орындау процесінде өзгерістер (қажет болғанда) енгізу.

Жеткізушінің қызметін қадағалау бірлескен бағалау және аудит процестерінде көзделген әрекеттерге сәйкес жүзеге асырылады.

Қабылдау процесінде қажетті тестілер даярланып орындалады. Шарт бойынша *жұмыстардың аяқталуы* қабылдау шарттарының барлығы қанағаттандырылған жағдайда жүзеге асырылады.

Жеткізу процесі тапсырыс берушіні жеткізушінің бағдарламалық өніммен немесе қызметпен жабдықтау үшін орындайтын әрекеттері мен міндеттерін қамтиды. Бұл процесс келесі әрекеттерді қамтиды:

1) жеткізуге бастама болу;
2) жоспарлау.

Жеткізуге бастама болу жеткізушінің өтінім ұсынымдарын қарастыруда және қойылған талаптармен келісу немесе өз талаптарын қоюмен шешім қабылдауда болады.

Жоспарлау келесі міндеттерді қамтиды:

- жеткізушінің өз күшімен немесе қосалқы мердігерлерді тарта отырып жұмыстарды орындауға қатысты шешім қабылдау;
- жеткізушінің жобаның ұйымдастырушылық құрылымынан тұратын жобаны басқару жоспарын әзірлеу, жауапкершіліктерді шектеу, әзірлеу ортасына қойылатын талаптар мен ресурстар, қосалқы мердігерлікті басқару.

Әзірлеу процесі кәсіпорын-әзірлеушінің әрекеттерін анықтайды, стратегиялық жоспарлауды, талдауды, жобалау мен іске асыруды (бағдарламалауды), яғни бағдарламалық қамсыздандыруды құру жөніндегі барлық жұмыстар мен берілген талаптарға сәйкес оның құрамдас бөлшектерін, сондай-ақ жобалық және пайдалану құжаттамасын рәсімдеу, әзірленген бағдарламалық өнімдерді тестілеуден өткізу және қызметкерлерды оқытуға қажетті материалдарды даярлауды қамтиды.

Әзірлеу БҚӨЦ маңызды процестерінің бірі болып табылады және келесі сатыларды қамтиды:

1) даярлық жұмысы;
2) жүйеге қойылатын талаптарды талдау;
3) жүйе сәулетін жобалау;
4) БҚ қойылатын талаптарға талдау;
5) БҚ сәулетін жобалау;
6) БҚ егжей-тегжейлі жобалау;
7) БҚ кодтау және тестілеуден өткізу;
8) БҚ біріктіру;
9) БҚ білікті тестілеу;

10) жүйені бірыңғайландыру;
11) жүйені білікті тестілеу;
12) БҚ орнату;
13) БҚ қабылдау.

Даярлық жұмысы жобаның масштабы, мәні мен күрделілігіне сәйкес келетін БҚ ӨЦ моделін таңдап алудан басталады. Процестің әрекеттері мен міндеттері таңдап алынған әдіске сәйкес келуі тиіс. Әзірлеуші жобаны таңдап алып, жағдайларға бейімдеуі және тапсырыс берушімен келісілген стандарттарға, әзірлеу әдістері мен құралдарына сәйкес қолдануы, сондай-ақ жұмыстарды орындау жоспарын құруы тиіс.

Жүйеге қойылатын талаптарды талдау оның функционалдық мүмкіндіктерін, пайдаланушы талаптарын, қауіпсіздік пен сенімділік талаптарын, сыртқы интерфейстерге талаптарын және т.б. анықтау дегенді білдіреді. Жүйеге қойылатын талаптар іске асырылу белгілері мен тестіден өткізу кезіндегі тексеру мүмкіндіктеріне сүйене отырып, бағаланады.

Жүйе сәулетін жобалау жоғары деңгейде оның жабдықтарының, бағдарламалық қамсыздандыру мен жүйені пайдаланатын қызметкерлердің орындайтын операцияларының құрамдас бөлшектерін анықтауда болады. Жүйенің сәулеті жүйеге қойылатын талаптарға, сондай-ақ қабылданған жобалық стандарттар мен әдістерге сәйкес келуі тиіс.

БҚ қойылатын талаптарға талдау бағдарламалық қамсыздандырудың әрбір құрамдас бөлігі үшін келесі сипаттамаларды анықтауды болжайды:

- функционалдық мүмкіндіктер, құрамдас бөлшектің өнімділігі мен жұмыс істеу ортасының сипаттамалары;
- сыртқы интерфейстер;
- сенімділік және қауіпсіздік сипаттама;
- эргономикалық талаптар;
- қолданылатын деректерге қойылатын талаптар;
- орнату және қабылдап алуға қойылатын талаптар;
- пайдаланушы құжаттамасына қойылатын талаптар;
- пайдалану және сүйемелдеуге қойылатын талаптар.

Бағдарламалық қамсыздандыруға қойылатын талаптар жүйеге қойылатын талаптардың сәйкестілік критерийі, тестілеу кезінде тексерілу мүмкіндігі мен сатылу критерийлеріне сүйене отырып бағаланады.

Бағдарламалық қамсыздандырудың *сәулетін жобалау* келесі міндеттерді қамтиды (бағдарламалық қамсыздандырудың әрбір құрамдас бөлігі үшін):

- бағдарламалық қамсыздандырудың құрамы мен бағдарламалық қамсыздандыру құрылымының жоғары деңгейін айқындайтын сәулетіне қойылатын талаптарын түрлендіру;
- бағдарламалық қамсыздандырудың бағдарламалық интерфейстері мен деректер қорын әзірлеу және құжаттау;
- пайдаланушы құжаттамасының алдын-ала нұсқасын әзірлеу;
- БҚ біріктіру тестілері мен жоспарларына қойылатын болжамды талаптарын әзірлеу және құжаттау.

Бағдарламалық қамсыздандырудың құрамдас бөлшектерінің сәулеті оларға қойылатын талаптарға, сондай-ақ қабылданған жобалық стандарттар мен әдістерге сәйкес келуі тиіс.

Бағдарламалық қамсыздандыруды *егжей-тегжейлі жобалау* келесі міндеттерді қамтиды:

- құрамдас бөлшектері мен интерфейстердің арасындағы ең төмен деңгейде, олардың жеткілікті түрде кейінірек дербес кодтау және тестілеу деңгейіндегі сипаттамасы;
- деректер қорының жобасын егжей-тегжейлі әзірлеу жіне құжаттау;
- пайдаланушы құжаттамасын жаңарту (қажет болғанда);
- тестілер мен БҚ құрамдас бөліктерін тестіден өткізу жоспарын әзірлеу және оларға қойылатын талаптарды құжаттау;
- БҚ біріктіру жоспарын жаңарту.

Бағдарламалық қамсыздандыруды *кодтау және тестілеу* келесі міндеттерді қамтиды:

- БҚ әрбір компоненті мен деректер қорын әзірлеу және құжаттау, сондай-ақ тестілік процедуралар мен оларды тестіден өткізуге арналған деректердің жиынтығы;
- бағдарламалық қамсыздандырудың әрбір компоненті мен деректер қорын оларға қойылатын талаптарға сәйкес тестіден өткізу. Компоненттерін тестілеу нәтижелері құжатталуы тиіс;
- пайдаланушы құжаттамасын жаңарту (қажет болғанда);
- БҚ біріктіру жоспарын жаңарту.

Бағдарламалық қамсыздандыруды біріктіру агрегатталған компоненттерін біріктіру және тестіден өткізу жоспарына сәйкес БҚ әзірленген компоненттерін жинауды көздейді. Агрегатталған компоненттердің әрқайсысы үшін келесі біліктілік тестілеуінде біліктілік талаптарының әрқайсысын тексеруге арналған тестілер мен тест процедуралары әзірленеді.

Біліктілікке тестілеу - бұл бағдарламалық өнімді өзінің сипаттамасына сай және пайдалану жағдайларында қолдануға дайын бағдарламалық өнім ретінде білікті етуді орындауды қажет ететін белгілер мен шарттардың жиынтығы.

БҚ біліктілікке тестілеу тапсырыс берушінің қатысуымен (мүмкіндігінше) әзірлеушілердің бағдарламалық қамсыздандырудың өз сипаттамасын қанағаттандыратындығын және пайдалану жағдайларында қолдануға дайын екендігін көрсету үшін жүргізіледі. Біліктілікке тестілеу тестілерді кең ауқымда түрлендіру кезінде талаптарлың барлық талаптары бойынша бағдарламалық қамсыздандырудың әрбір компоненті үшін орындалады. Бұл ретте сондай-ақ, техникалық және пайдалану құжаттамасының толықтығы және бағдарламалық қамсыздандырудың компоненттерінің өздерінің барабарлығы тексеріледі.

Жүйені біріктіру бағдарламалық қамсыздандыру мен жабдықтарды қоса алғанда, оның барлық компоненттерін жинаумен болады. Біріктіруден кейін жүйе өз кезегінде оған қойылатын талаптардың жиынтығына сәйкестікке біліктілік тестінен өтеді. Бұл ретте де жүйеге арналған толық құжаттама кешені рәсімделеді және тексеріледі.

БҚ орнатуды әзірлеуші шартта көзделген ортада және жабдықтарда жоспарға сәйкес жүзеге асырады. Орнату процесінде бағдарламалық қамсыздандыру мен деректер қорының жұмысқа қабілеттілігі тексеріледі. Егер орнатылатын бағдарламалық қамсыздандыру қолданыстағы жүйені ауыстыратын болса, әзірлеуші шартқа сәйкес олардың параллель қызмет етуін қамтамасыз етуі тиіс.

БҚ қабылдап алу бағдарламалық қамсыздандыру мен жүйені біліктілікке тексеру нәтижелерін бағалау мен әзірлеушінің көмегімен тапсырыс беруші жүргізетін бағалау нәтижелерін құжаттауды көздейді. Әзірлеуші тапсырыс берушіне БҚ қорытынды тапсыруын қажетті оқыту мен сүйемелдеуді қамтамасыз ете отырып, орындайды.

Пайдалану процесі бағдарламалық қамсыздандыруды пайдаланушы мүддесінде жұмыс істеу процесінде қызмет көрсетілуін қамтамасыз ететін қызметкерлердің әрекеттерін анықтайды. Негізгі пайдалану жұмыстары тікелей пайдалануды, проблемалардың орнын анықтау және олардың пайда болу себептерін жоюды, бағдарламалық қамсыздандырудың түрін өзгертуді, жүйені жетілдіру жөніндегі ұсыныстарды даярлауды, жүйені дамыту мен жаңғыртуды қамтиды. Пайдалану процесіне деректер қорын және пайдаланушылардың жұмыс орындарын конфигурациялау, пайдаланушыларды пайдалану құжаттамасымен қамтамасыз ету, қызметкерлерді оқыту да кіреді.

Пайдалану процесі бірқатар әрекеттерді қамтиды:

1) даярлық жұмыстары;
2) пайдалану тестілеуі;
3) жүйені пайдалану;
4) пайдаланушыларды қолдау.

Даярлық жұмыстары оператордың келесі міндеттерді жүргізуін қамтиды:

- пайдалану процесінде орындалатын әрекеттер мен жұмыстарды орындау және пайдалану стандарттарын белгілеу;
- пайдалану процесінде туындайтын проблемалардың орнын анықтау мен шешу процедураларын айқындау.

Пайдалану тестілеуі бағдарламалық өнімнің кезекті редакциясы үшін жүзеге асырылады, бұдан кейін ол пайдалануға беріледі.

Жүйені пайдалану оған арналған ортада пайдалану құжаттамасына сәйкес орындалады.

Пайдаланушыларды қолдау бағдарламалық қамсыздандыруды пайдалану процесінде қателер табылған кезде көмек көрсету және кеңес беруде болады.

Сүйемелдеу процесі бағдарламалық өнімді сүйемелдеуді қамтамасыз ететін қызметкерлердің әрекеттерін анықтайды, ол бағдарламалық өнімнің түрлерін өзгертуді басқару, оның ағымдағы күйін сүйемелдеу мен функционалдық жарамдылығын қолдауды, сондай-ақ бағдарламалық қамсыздандыруды қосу мен жоюды қамтиды. Сүйемелдеу процесінде бағдарламалық қамсыздандыруға өмірлік циклының кез келген сатысында қабылданатын жобалық шешімдердің қайта қаралуын талап ететін қажетті өзгертулерді енгізеді. Қолданыстағы бағдарламалық қамсыздандыруға енгізілетін өзгертулер оның тұтастығын бұзбауы тиіс. Сүйемелдеу процесі бағдарламалық қамсыздандыруды басқа ортаға ауыстыру (көшіру) процесін қамтиды және бағдарламалық қамсыздандыруды пайдалануға енгізумен аяқталады.

Сүйемелдеу процесі келесі әрекеттерді қамтиды:

1) даярлық жұмыстары;
2) БҚ түрін өзгертуге сұратулар мен мәселелеріне талдау;
3) БҚ түрін өзгерту;
4) тексеру және қабылдап алу;
5) БҚ басқа ортаға ауыстыру;
6) БҚ пайдаланудан алу.

Сүйемелдеу қызметінің *даярлық жұмыстары* келесі міндеттерді қамтиды:

- сүйемелдеу процесінде орындалатын әрекеттер мен жұмыстарды жоспарлау;
- сүйемелдеу процесінде туындайтын проблемалардың орнын анықтау және шешу процедураларын анықтау.

Сүйемелдеу қызметі орындайтын *БҚ түрін өзгертуге сұрату және проблемаларына талдау* келесі міндеттерді қамтиды:

- БҚ түрін өзгертуге сұрату немесе туындайтын проблемаларының оның ұйымға, қолданыстағы жүйелер мен интерфейстерге және өзге жүйелерге әсеріне қатысты хабарламаларды талдау. Бұл ретте түр өзгеруінің келесі сипаттамалары анықталады: типі (түзетуші, жақсартушы, сақтандырушы немесе жаңа ортаға бейімдеуші); масштабы (түрін өзгерту өлшемдері, іске асыру құны мен уақыты); сыншылдығы (өнімділікке және қауіпсіздікке әсері);
- түрін өзгертуді жүргізудің мақсатқа сәйкестігін және оны жүргізудің ықтимал нұсқаларын бағалау;
- таңдап алынған түрін өзгерту нұсқасын бекіту.

БҚ түрін өзгерту БҚ компоненттерін, оның нұсқалары мен түрі өзгертілуге жататын құжаттамасын анықтауды және де әзірлеу процесінің ережелеріне сәйкес қажетті өзгертулерді енгізуді көздейді. Даярланған өзгертулер құжаттамада айқындалған белгілер бойынша тестілеуден өткізіледі және тексеріледі. Бағдарламаларға өзгерту енгізу дұрыс болып расталған кезде құжаттамаға түзету жүргізіледі.

Тексеру және қабылдау түрі өзгертілген жүйенің тұтастығын тексеруге және енгізілген өзгертулерді бекітуге негізделеді.

Бағдарламалық қамсыздандыруды басқа ортаға ауыстырған кезде ауыстырудың қолданыста бар немесе жаңа құралдары әзірленеді, содан соң бағдарламалар мен деректердің жаңа ортаға айырбасталуы жүргізіледі.

БҚ пайдаланудан шығару тапсырыс берушінің шешімі бойынша пайдаланушы ұйымның, сүйемелдеу қызметінің және пайдаланушылардың қатысуымен жүзеге асырылады. Бұл ретте бағдарламалық өнімдер мен тиісті құжаттама шартқа сәйкес мұрағаттандыруға жатады.

Өмірлік циклдың қосалқы процестері

Қосалқы процестер ақпараттық жүйенің өмірлік циклының ажырамас бөлігі бола отырып, негізгі процестерді белгілі бір мақсатпен және ақпараттық жүйенің тиісті сапасын қамтамасыз етумен жүзеге асырылуын қамтамасыз етеді. Қосалқы процестер қажеттілігіне орай басқа процестермен қолданылады және орындалады. Қосалқы процестер - бұл:

1) құжаттау;
2) конфигурацияны басқару;
3) сапаны қамтамасыз ету;
4) тексеру;

5) аттестаттау;
6) бірлесіп бағалау;
7) аудит;
8) проблемаларды шешу.

Құжаттау процесі бағдарламалық қамсыздандырудың өмірлік циклы бойында құрылатын ақпараттың қалыпты сипаттама берілуін көздейді. Бұл процесс жұмыстар жиынтығынан тұрады, олардың көмегімен әкімшілер, инженерлер, жүйе немесе бағдарламалық өнім пайдаланушылары сияқты барлық мүдделі тұлғалар қажет ететін құжаттарды жоспарлайды, жобалайды, әзірлейді, шығарады, редакциялайды, таратады және сүйемелдейді.

Құжаттау процесі келесі әрекеттерді қамтиды:

- даярлық жұмыстарын;
- жобалау мен әзірлеуді;
- құжаттаманы шығаруды;
- сүйемелдеуді.

Конфигурацияларды басқару процесі конфигурацияларды басқару жөніндегі әрекеттерді анықтайды. Бағдарламалық қамсыздандыру *конфигурациясы* деп техникалық құжаттамада белгіленген және бағдарламалық қамсыздандыруда іске асырылған оның функционалды және физикалық сипаттамаларының жиынтығы алынады. Конфигурацияларды басқару бағдарламалық қамсыздандырудың барлық өмірлік циклының барлық сатыларында бағдарламалық қамсыздандыруға өзгертулерді енгізуді ұйымдастыруға, жүйелі түрде есепке алуға және бақылауға мүмкіндік береді. Бұл, ең алдымен, әзірлеу мен сүйемелдеу процестерінің өмірлік циклының негізгі процестерін қолдайтын сол қосалқы процесі.

Көптеген компоненттерден тұратын, олардың әрқайсысы тәуелсіз түрде әзірленетін және бір/бірнеше іске асыру нұсқалары бар күрделі бағдарламалық жүйелердің жобаларын әзірлеу барысында олардың байланыстары мен функцияларын есепке алу проблемасы туындайды, бірыңғай құрылымды құру мен бүкіл жүйені дамытуды қамтамасыз ету проблемасы пайда болады. Конфигурацияларды басқару ақпараттық жүйенің түрлі компоненттеріне оның өмірлік циклының барлық сатыларында өзгерістерді енгізуді ұйымдастыруға, жүйелі түрде есепке алуға және бақылауға мүмкіндік береді.

Конфигурацияны басқару процесі мына әрекеттерді қамтиды:

- *даярлық жұмыстарын* - конфигурацияны басқаруды жоспарлау;
- *конфигурацияны сәйкестендіру* - БҚ компоненттері мен олардың нұсқаларын біркелі сәйкестендіруге және ажыратуға болатын ережелерді белгілейді. Сонымен қатар, әрбір компонент пен оның

нұсқалары бір мағыналы берілетін құжаттамалар жиынтығы сәйкес келеді. Нәтижесінде БҚ түрлі нұсқаларын сәйкестендіретін шектеуші және жеңілдетілген символдар жүйесін қолданатын БҚ компоненттерінің нұсқаларын бір мағыналы таңдау үшін база құрылады;

- *конфигурацияны бақылау* - әрбір модификацияны есепке ала отырып және оның орындалуына жұмсалатын шығындарымен болжанып отырған түрі өзгертілген БҚ мен олардың үйестіріліген сатылуын жүйелі түрде бағалауға арналған. Ол нақты өзгеретін компоненттердің және жиынтық құжаттамасының барабарлығын қамтамасыз етеді;
- *конфигурация күйін есепке алу* - БҚ компоненттерінің күйін тіркеуді, БҚ компоненттерінің нұсқаларының барлық сатылған және қабылданбаған нұсқалары туралы есептерді даярлауды білдіреді. Есептердің жиынтығы жүйенің және оның компоненттерінің бір мағыналы көрінісін, сондай-ақ түрін өзгертуді жүргізуді қамтамасыз етеді;
- *конфигурацияны бағалау* - БҚ компоненттерінің толық функционалдылығын бағалауды білдіреді, сондай-ақ олардың физикалық күйінің ағымдағы техникалық сипаттамасына сәйкестігі;
- *шығаруды және жеткізуді басқару* - бағдарламалардың және құжаттамалардың эталонды көшірмелерін шығаруды, олардың ұйымда қабылданған тәртіпке сәйкес сақталуы мен жеткізілуін қамтиды.

Сапаны қамтамасыз ету процесі объективті кепілдік әрекеттерін анықтайды, ол ақпараттық жүйе мен процестердің оларға қойылған талаптарға сәйкестігін және белгіленген мәнін сақтайды.

Бағдарламалық қамсыздандыру сапасы деп БҚ белгіленген талаптарды қанағаттандыру қабілетін сипаттайтын қасиеттерінің жиынтығы түсіндіріледі.

Жасалатын бағдарламалық қамсыздандыруды дұрыс бағалауды алу үшін қамтамасыз ету процесі БҚ әзірлеумен тікелей байланысты субъектілерге байланысты өтпейді.

Сапаны қамтамасыз ету процесі мына әрекеттерді қамтиды:

- *даярлық жұмысы* - қолданылатын стандарттарды, әдістерді, процедуралар мен құралдарды есепке ала отырып, сапаны қамтамасыз ету процесінің өзін жоспарлау және өзге қосалқы процестерімен үйлестірілуінде болады;
- *өнім сапасын қамтамасыз ету* - бағдарламалық өнімдердің және олардың құжаттамаларының шартта көзделген талаптарына тапсырыс берушінің толық сәйкестігіне кепілдік беретіндігін

білдіреді;

- *процесс сапасын қамтамасыз ету* - БҚ ӨЦ процестерінің, әзірлеу әдістерінің, қызметкерлердің белгіленген стандарттар мен процедураларға біліктілігі мен әзірлеу ортасының сәйкестігіне кепілдікті білдіреді;
- *жүйе сапасының өзге көрсеткіштерін қамтамасыз ету* - шарт талаптарына және ISO 9001 стандартына сәйкес жүзеге асырылады.

Бірлескен бағалау, тексеру, анықтау, аттестаттау сапа кепілдігінің тәсілдері ретінде қолданылуы мүмкін.

Тексеру процесі бағдарламалық қамсыздандыруды жобаға түрлі тәуелділікпен анықтап тексеруге арналған әрекеттерді (сатып алушы, жеткізуші немесе тәуелсіз тарап үшін) анықтайды. Тексеру шағын мағынасында бағдарламалық қамсыздандырудың дұрыстығын дәлдеп тексеруді білдіреді.

Тексерудің мақсаты кепілдікке жету болып табылады, анықталып тексерілетін нысан (талап немесе бағдарлама коды) көзделмеген функцияларсыз іске асырылмаған және жобалық сипаттамалар мен стандарттарды қанағаттандырады.

Тексеру процесі код инспекциясын, тестілеуді, тестілеу нәтижелерін талдау, проблемалар туралы есепті қалыптастыру мен талдауды қамтиды. Осылайша, тестіден өткізу процесі тексеру процесінің құрамдас бөлшегі болып табылатындығын есептеу қабылданған.

Тексеру түрлі тәуелсіздік деңгейлерімен жүргізілуі мүмкін.

Тәуелсіздік дәрежесі орындаушының тексеруді орындауынан немесе осы ұйымның басқа маманының орындауынан бастап түрлі нұсқалары бар өзге ұйымның мамандарының орындауына дейін болуы мүмкін. Егер тексеру процесі жеткізуші, әзірлеуші, операторы немесе сүйемелдеу қызметіне байланысты емес ұйыммен жүзеге асырылса, онда ол тексерудің *тәуелсіз процесі* деп аталады.

Тексеру процесінде келесі шарттар тексеріледі:

- жүйеге қойылатын талаптардың қарама-қайшы болмауы және пайдаланушылардың қажеттіліктерін есепке алу дәрежесі;
- жеткізушінің белгіленген талаптарын орындау мүмкіндіктері;
- бағдарламалық қамсыздандырудың өмірлік циклының таңдап алынған процестерінің шарт талаптарына сәйкестігі;
- бағдарламалық қамсыздандырудың өмірлік циклының процестерін әзірлеу ортасының, стандарттарының, процедураларының сәйкестігі;
- БҚ жобалық сипаттаманың берілген талаптарға сәйкестігі;
- кіру және шығу деректерінің жобалық сипаттамаларында, оқиғалардың бірізділігінде, интерфейстерде, логикада сипаттамасын

түзету;

- кодтың жобалық сипаттамасына және талаптарына сәйкестігі;
- кодтың тестіден өткізілуі және дұрыстығы, оның қабылданған кодтау стандарттарына сәйкестігі;
- бағдарламалық қамсыздандыру компоненттерінің жүйеге бірігуінің дұрыстығы;
- құжаттаманың барабарлығы, толықтығы және қарама-қайшы болмауы.

Аттестаттау процесі бағдарламалық қамсыздандыруды аттестаттауға арналған әрекеттерді (сатушы, жеткізуші, тәуелсіз тарап) анықтайды. *Аттестаттау* деп әдетте жүргізілген тестілеудің дұрыстығын растау мен бағалау түсіндіріледі. Аттестаттау БҚ сипаттамаларда, талаптарға және құжаттамаға толық сәйкестігін, сондай-ақ оны пайдаланушының қауіпсіз және сенімді пайдаланылуына кепілдік беруі тиіс. Аттестаттауды барлық мүмкін жағдайларда тестілеу арқылы орындауға және бұл ретте тәуелсіз мамандарды қамту ұсынылады. Аттестаттау бағдарламалық қамсыздандырудың өмірлік циклының бастапқы сатыларында немесе бағдарламалық қамсыздандыруды қабылдау жөніндегі жұмыстың бөлігі ретінде жүргізілуі мүмкін.

Аттестаттау тексеру сияқты тәуелсіздіктің түрлі дәрежелерімен жүзеге асырылуы мүмкін. Егер аттестаттау процесі жеткізушіге, әзірлеушіге, сүйемелдеу операторы немесе қызметіне қатысты емес ұйыммен орындалатын болса, онда ол *тәуелсіз аттестаттау процесі* деп аталады.

Бірлесіп бағалау процесі жоба және БҚ бойынша жұмыстардың күйін бағалауға арналған. Ол жобаның ресурстарын, қызметкерлерін, аппаратурасын және аспаптық құралдарын жоспарлау мен басқару бақылауына бағытталған.

Бағалау жобаны басқару деңгейінде қолданылатын сияқты, сондай-ақ шарт мерзімінің бойында жобаны техникалық іске асыру деңгейінде жүргізіледі. Бұл процесс шартқа қаысатын кез келген тараптың екеуімен орындалады, бұл ретте бір тарап басқасын тексереді.

Бірлесіп бағалау процесі әрекеттерді қамтиды:

- даярлық жұмыстары;
- жобаны басқаруды бағалау;
- техникалық бағалау.

Аудит процесі шарт талаптарына, жоспарларына және талаптарға сәйкес анықтаманы береді. Аудит бір тарап екінші тарапты тексерген кезде шартқа қаысатын екі кез келген тараппен орындалуы мүмкін.

Аудит - бұл БҚ немесе процестерінің белгіленген талаптарға

сәйкестілік дәрежесін тәуелсіз бағалауды қамтамасыз ету мақсатында құзырлы орган (тұлға) өткізетін ревизия (тексеру).

Аудит нақты жұмыстар мен есептердің талаптарға, жоспарларға және келісім-шартқа сәйкестігін белгілеуге қызмет етеді. Аудиторлар бағдарламалық қамсыздандыру әзірлеушілеріне тікелей байланыста болмауы тиіс. Олар жұмыстардың күйін, ресурстарды қолдануды, талаптар мен стандарттарға құжаттаманың сәйкестігін, тестілеу дұрыстығын анықтайды.

Проблемаларды шешу процесі проблемаларды талдау және жою процесін анықтайды (сәйкессіздікті қоса алғанда), олар әзірлеу, пайдалану, сүйемелдеу немесе басқа процестер кезінде табылғандығына қарамастан анықтайды. Әрбір анықталған проблема сәйкестендіріліп, сипатталып, талдау жасалып, шешілуі тиіс.

Ұйымдастыру процестері

Бағдарламалық қамсыздандырудың өмірлік циклының ұйымдастыру процестеріне мыналар жатады:

1) басқару;
2) жетілдіру;
3) инфрақұрылымды құру;
4) оқыту.

Басқару процессі өздерінің процестерін басқаратын кез келген тараптың бірімен орындалуы мүмкін әрекеттерден және міндеттерден тұрады. Бұл тарап (менеджер) өнімді шығаруды басқару, сатып алу, жеткізу, әзірлеу, пайдалану, сүйемелдеу және басқалар сияқты тиісті процестердің міндеттерін басқару үшін жауап береді.

Басқару процесі келесі әрекеттерді қамтиды:

- басқару саласына бастама болу және анықтау - менеджер басқару үшін оның басшылығында ресурстар (қызметкерлер, жабдықтар мен технология) жеткілікті көлемде екендігіне көз жеткізуі тиіс;
- жоспарлау - жұмыстарды орындау кестелерін құру, шығындарды бағалау, талап етілетін ресурстарды бөлу, жауапкершіліктерді бөлу, нақты міндеттерге байланысты тәуекелді бағалау, басқару инфрақұрылымын құру;
- орындау және бақылау;
- тексеру және бағалау;
- аяқтау.

Инфрақұрылымды құру процесі бағдарламалық қамсыздандыруды әзірлеу, пайдалану немесе сүйемелдеу үшін қолданылатын

технологияларды, стандарттарды және аспаптық құралдарды таңдау мен сүйемелдеуді (қолдауды), аппараттық және бағдарламалық құралдарды таңдау мен орнатуды қамтиды. Инфрақұрылымы тиісті процестерге қойылатын талаптардың өзгеруіне сәйкес түрі өзгертіліп, сүйемелденуі керек. Инфрақұрылым, өз кезегінде, конфигурацияны басқару нысандарының бірі болып табылады.

Жетілдіру процесі БҚ өмірлік циклы процестерін бағалауды, өлшеуді, бақылауды және жетілдіруді көздейді. БҚ өмірлік циклының процестерін жетілдіру бұған қатысатын барлық қатысушылардың еңбек өнімділігін қолданылатын технологияларды, басқару әдістерін, аспаптық құралдарды таңдау мен қызметкерлерді оқытуды жетілдіру есебінен арттыруға бағытталған. Жетілдіру әрбір процестің артықшылықтары мен кемшіліктеріне талдау жасауға негізделген. Мұндай талдауға көбінесе ұйымда іске асырылған жобалар бойынша тарихи, техникалық, экономикалық және өзге ақпараттың жиналуы ықпал етеді.

Оқыту процесі қызметкерлерді алғашқы оқыту мен олардың біліктілігін үнемі арттыруды қамтиды. БҚ сатып алу, жеткізу, әзірлеу, пайдалану және сүйемелдеу көбінесе қызметкерлердің білім деңгейі мен біліктілігіне байланысты болады. Оқыту процесінің мазмұны жобаға қойылатын талаптарымен айқындалады. Ол оқытудың қажетті ресурстары мен техникалық құралдарын ескеруі тиіс. Оқу жоспарына сәйкес пайдаланушыларды оқытуға қажетті әдістемелік материалдар әзірленіп, ұсынылуы тиіс.

Оқыту процесі даярлық жұмысын, оқу материалдарын әзірлеу мен оқу жоспарын іске асыруды қамтиды.

1.4. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ ӨМІРЛІК ЦИКЛЫ ПРОЦЕСТЕРІ АРАСЫНДАҒЫ БАЙЛАНЫС

ISO/IEC 12207 стандартымен реттелетін БҚ ӨЦ процестерін түрлі ұйымдар әр түрлі тәсілмен нақты жобаларда қолдануы мүмкін. Сонда да, стандарт әр түрлі көзқараспен алғандағы процестер арасындағы өзара байланыстың негізгі жиынтығын ұсынады. Мұндай аспектілер болып табылатындар:

- шарт аспекті;
- басқару аспекті;
- пайдалану аспекті;
- инженерлік аспекті;
- сүйемелдеу аспекті.

Шарт аспектісінде тапсырыс беруші мен жеткізуші шарт қарым-қатынасына түседі және тиісінше сатып алу және жеткізу процестерін іске асырады. Басқару аспектінде тапсырыс беруші, жеткізуші, әзірлеуші, оператор, сүйемелдеу қызметі мен БҚ ӨЦ қатысатын өзге қатысушылары өздерінің процестерін орындауды басқарады. Пайдалану аспектісінде жүйені пайдаланатын оператор пайдаланушыларға қажетті қызметтерді ұсынады. Инженерлік аспектісінде әзірлеуші немесе сүйемелдеу қызметі тиісті техникалық міндеттерді шешеді, ол бағдарламалық өнімдерді әзірлеуді немесе түрлерін өзгертеді. Сүйемелдеу аспектісінде қосалқы процестерді іске асыратын қызметтер жұмыстың қалған қатысушыларының барлығына қажетті қызметтерді көрсетеді.

Стандартта сипатталған процестер арасындағы өзара байланыстар тұрақты сипатта болады. Процестер мен іске асырушы тараптар арасындағы ең маңызды динамикалық байланыстар нақты жобаларда белгіленеді.

1.5. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ҰЖЫМДЫҚ ӘЗІРЛЕУДІ ҰЙЫМДАСТЫРУ

Жобаны дамыту барысында әзірлеушілердің командасы осы не басқа функцияларды орындайды. Түрлі жобалардағы функциялар түрлі

мазмұнда болады. Алайда, бағдарламалардың барлығы типтік функцияларды болжайды. Осылайша, кез келген бағдарламалық жобада *кодтау функциясы* - бағдарламаларды бар сипаттамаларына қарай алгоритм тіліне жазу, талаптарға талдау, яғни, құрылатын бағдарламалық өнімге деген қажеттілікті анықтау, тестілеу және ретке келтіру мен т.с.с. Кез келген жобаның менеджерінің міндеттемелері шеңберінде орындаушылар арасындағы жоба функцияларын таратуды ұйымдастыру қажет. Нәтижесінде жобаны орындайтын команда мүшелері тиісті рөлдерін ойнай бастайды.

Әдетте рөл жақын функцияларды біріктіреді. Сонымен бірге, кейде рөлдерді бас функциялар деп те атау қабылданған.

Әзірлеушіге белгілі бір рөлдің орындаушысы ретінде орындау ұйғарылған функцияларды жобадағы тапсырмалармен шатыстырудың қажеті жоқ. Тапсырма - бұл бір рет немесе жүйелі түрде берілетін тапсырма, әдетте бір немесе басқа функцияны орындау үшін қажетті әрекеттерден қалыптасады, Егер функция үшін тапсырмаларды орындау регламенті айқындалған болса, яғни оның тапсырмасын құраушыларды орындау бірізділігі орындаушы үшін қосымша түсіндірулерді талап етпесе, онда мұндай функция *технологиялық* деп аталады. Кез келген жобаны әзірлеу кезінде олардың технологиялылығын арттыруға, функциялардың көптеген санына регламентті белгілеуге деген талпыныс тән. Сондай-ақ, менеджер үшін бұған қол жеткізудің бірі қажетті біліктілігі бар қызметкерлерді қолдану болып табылады, олар үшін берілетін рөлдер технологиялық болады, яғни технологиялық функциялардан ғана тұратын, ол нақты жағдайларда әрдайым мүмкін емес.

Жобада әзірлеушілер орындайтын функциялар ұйымдастырушы және өндірістік деп ажыратылады. Алғашқылары жоба міндеттерін орындау үшін құрады, екіншілері олармен тікелей байланысты міндеттерді орындау үшін құрылады. Көбінесе жобаның сәтсіздіктері менеджердің ұйымдастыру функцияларын ескеруден туындайды. Осылайша, әдетте жобалық міндет тапсырыс берушіге нәтижелер ретінде тапсырыс берушіге не ұсыну қажеттілігін ғана белгілейді. Бір жағынан алып қарағанда, нәтижелерді білу қажет емес, мысалы, әзірлеушілер арасындағы жобалық материалдарды тапсыру қалай ұйымдастырылған, есептіліктің қандай процедурасы көзделеді, алайда жобада ақпараттық ағынды іске асыру міндеттерін ескермеу хаос тудыруы мүмкін де, соңында ол нәтижелерде көрінеді.

Әзірлеушілердің құрамы олардың рөлдерінің мәні мен олардың орындайтын жобаларының байланысы орындаушылардың ұжымына, қабылдаған технологияларына, басқа факторларына байланысты

болуына қарай ажыратылады. Рөлдерді таңдаудың біркелкі болмауы олардың тізімі тым үлкен болуына әкеледі. Рөлдер санының шамадан тыс артуы рөлдер мен функцияларды бірдей деп санау салдарына әкеледі, тиісінше, жақын функциялардың ұғымдарын елемеу. Сонда да, егер рөлдердің саны жеткілікті болмаса, онда жобада қажет емес барлық функциялар жоспарлаумен және басқарумен қамтылатын болады.

Нақты әзірлеуші бір уақытта бірнеше рөлдерді қатар жүргізе алса, рөл де солай бірнеше орындаушылар арасында бөлінуі мүмкін.

Рөлдердің нақты жиынтығы көптеген факторлармен - әзірлеме қатысушыларының санымен және жеке артықшылықтарымен, қабылданған әзірлеме әдістемесімен, жоба ерекшеліктерімен және өзге факторларымен анықталады.

Әрине, рөлдерді қоса атқару еркін болуы мүмкін емес. Кейбір қоса атқарулар ұнамсыз болса, кейбіреуі бейтарап, үшіншісі түрлі дәрежеде жоба үшін пайдалы. Рөлдерді қоса атқаруды болдырмау керек, олардың даулы немесе қарама-қарсы мүдделері бар. Егер қызметкерге бірнеше рөл берілсе, онда ол қазіргі кезде орындау қажеттілігін әрдайым білуі тиіс.

Төменде әзірлеушілер ұжымында ажыратып көрсетуге болатын рөлдер атап өтілді.

Тапсырыс беруші - бұл рөл әзірленіп жатқан бағдарламалық қамсыздандыруға тапсырыс берген ұйымның өкіліне жатады. Тапсырыс беруші тек жоба менеджерлерімен және сертификаттау немесе енгізу жөніндегі менеджерлермен ғана байланыс ұстайды. Әдетте, тапсырыс беруші әзірленетін бағдарламалық қамсыздандырудың техникалық емес ерекшеліктерін қозғайтын жобалық және сертификаттау құжаттамасын оқу, менеджерлермен келісімде бағдарламалық өнімге қойылатын талаптарды өзгертуге құқығы бар.

Тақырыптық сала сарапшысы осы саланың міндеттерін шешуге жоба бағытын қолдайды, қосымшалар саласын зерделеуге жауап береді.

Ресурстарды жоспарлаушы берілген әзірлемені жүзеге асыратын ұйымда жобаға қойылатын талаптарды жылжытады және үйлестіреді, сондай-ақ ұйымдастыру жағынан алғанда жобаны орындау жоспарын дамытады және бағыттайды.

Жоба менеджері жалпы жобаның дамуы үшін жауап береді, тапсырмалар мен ресурстарды бөлу жобаны орындауға мүмкіндік беретіндігіне, жұмыстар мен нәтижелерді қою кесте бойынша өтетіндігіне, нәтижелер талаптарға сәйкес келетіндігіне кепілдік береді. Осы функциялардың шеңберінде жоба менеджері тапсырыс берушімен және ресурстар жоспарлаушысымен өзара әрекет етеді.

Команда жетекшісі жобаны орындау процесінде командаға

техникалық басқаруды жүргізеді. Үлкен жобалар үшін жеке міндеттерге жауап беретін қосалқы командалардың бірнеше жетекшілерін тарту мүмкін.

Сәулетші жүйенің сәулетін жобалау үшін жауап береді, жобамен байланысты жұмыстардың дамуына келісім алады.

Әзірлеуші жобаланатын компоненттерді іске асырады, өзіне тән кластар мен әдістерді меңгереді және құрады, кодтау мен автономды тестілеуді жүзеге асырады, өнімді құрады. Бұл кең ұғым арнайы рөлдерге (мысалы, класс әзірлеушілеріне) ажыратылуы мүмкін. Нақты жоба шеңберінде әзірлеушінің рөлі мысалы, бағдарламалық қамсыздандыруды қосуды, өнімді немесе қызметті баптауды дегенді білдіреді. Әзірлеуші барлық жобалық құжаттамаға, оның ішінде тестілеу жөніндегі құжатқа қатынай алады, өзінің қызметтік міндеттемелерінің шеңберінде жүйенің бағдарлама кодына өзгерту енгізуге құқығы бар. Жобаның күрделілігіне байланысты команда әзірлеушілердің түрлі санынан тұруы мүмкін.

Пайдалану интерфейсі жөніндегі маман жүйені қолдану қолайлы болуы үшін жауапты. Тапсырыс берушімен пайдаланушы интерфейсі талаптарды қанағаттандыратындығына куәландыру үшін жұмыс істейді.

Тестілеу жөніндегі маман өнімнің функционалдылығын, сапасы мен тиімділігін тексереді, жобаны дамытудың әрбір фазасы үшін тестілер құрады және орындайды, тестілеу стратегиясын, жобаның әрбір фазасы үшін тест-талаптарды және тест-жоспарларды анықтайды, жүйені тестілеуді орындайды, тестілеудің өтуі туралы есептерді жинайды және талдайды. Тест-талаптар жүйелік талаптардың, функционалдық сипаттамаларды, сенімділік пен жүктеме қабілетіне қойылатын талаптарды, пайдаланушы интерфейстерін және бағдарлама кодының өзін орнын басады. Нақты алғанда тестілеу жөніндегі маманның рөлі көбінесе екіге – тестілерді әзірлеушіге және тестілеушіге бөлінеді. Тестілеуші тестіні орындау және ақпаратты жинау жөніндегі барлық жұмыстарды, тестілерді әзірлеуші – басқа қалған жұмыстарды орындайды.

Енгізу және сүйемелдеу жөніндегі маман әзірленетін жүйені енгізуді өткізу жоспарланып отырған тапсырыс берушінің алаңының ерекшеліктеріне талдау жасайды, жүйені орнату және баптау жөніндегі барлық жұмыс спектрін орындайды, пайдаланушыларды оқытады.

Қауіпсіздік жөніндегі маман құрылатын өнімнің қауіпсіздігі мәселелерінің барлық спектріне жауапты. Оның жұмысы өнімге қойылатын талаптарды жазуға қатысудан басталып, өнімді сертификаттау қорытындылау сатысымен аяқталады.

Техникалық жазушы әзірленген өнімге құжаттаманы даярлау жөніндегі міндеттерді, функционалдық мүмкіндіктерін қорытынды сипаттауды орындайды, ілеспе құжаттарды жазуға қатысады (көмек жүйесі, пайдаланушы нұсқаулығы).

Кітапханашы жобаның жалпы кітапханасын құру мен жүргізуге жауап береді, ол барлық жобалық жұмыс өнімдерден, сондай-ақ жұмыс өнімдерінің стандарттарға сәйкестігінен тұрады.

БАҚЫЛАУ СҰРАҚТАРЫ ЖӘНЕ ТАПСЫРМАЛАР

1. Өмірлік цикл деген не және оның қандай сатылары бар?
2. Бағдарламалық қамсыздандырудың өмірлік циклы немен реттеледі?
3. Өмірлік циклдың құрамына қандай процестер тобы кіреді және әрбір топтың құрамына қандай процестер кіреді?
4. Сатып алу процесінің құрамына қандай әрекеттер кіреді және олардың мақсаты не?
5. Жеткізу процесінің құрамына қандай әрекеттер кіреді және олардың мақсаты не?
6. Әзірлеу процесінің барысында қандай әрекеттер мен міндеттер орындалады?
7. Пайдалану процесінің құрамына қандай әрекеттер кіреді және олардың мақсаты қандай?
8. Сүйемелдеу процесі деп нені айтамыз?
9. Құжаттау процесінің құрамына қандай әрекеттер кіреді және олардың мақсаты қандай?
10. Конфигурацияны басқару процесінің құрамына қандай әрекеттер кіреді және олардың мақсаты қандай?
11. Сапаны қамтамасыз ету процесінің құрамына қандай әрекеттер кіреді және олардың мақсаты қандай?
12. Тәуелсіз аттестаттау процесі деп нені айтамыз?
13. Бірлесе бағалау процесінің аудит процесінен айырмашылығы неде?
14. Проблемаларды шешу процесінде қандай міндеттер орындалады?
15. Басқару процесінің құрамына қандай әрекеттер кіреді және олардың мақсаты қандай?
16. Инфрақұрылымды құру процесінде қандай міндеттер орындалады?
17. Оқыту процесі нені мақсат етеді?
18. Бағдарламалық қамсыздандыру әзірлеушілерінің ұжымында функциялар қалай бөлінеді?

2-бөлім

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ ӨМІРЛІК ЦИКЛЫНЫҢ МОДЕЛЬДЕРІ

2.1. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ӘЗІРЛЕУ СТРАТЕГИЯЛАРЫ

Бағдарламалық қамсыздандырудың өмірлік циклының моделі деп бағдарламалық қамсыздандырудың өмірлік циклы ішінде орындалатын процестердің, әрекеттердің және міндеттердің жүзеге асырылу жүйелілігін, сондай-ақ осы процестер, әрекеттер мен міндеттер арасындағы өзара байланысты айқындайтын кейбір құрылымды атайды.

Кез келген жүйенің немесе бағдарламалық өнімнің өмір процесі сатыларынан тұратын өмірлік циклы моделі арқылы сипатталуы мүмкін. Модельдер ағымдағы жобаға сәйкес келетін өмірлік циклдың бөлігін ойлаудан бастап қолдануы тоқтағанға дейінгі бүкіл өмірлік циклын көрсету үшін қолданылуы мүмкін.

Өмірлік циклының моделі сатылардың бірізді берілуі түрінде көрсетіледі, олар қолданылу саласына, өлшемдеріне, күрделілігіне, өзгертулерге деген қажеттіліктері мен мүмкіндіктеріне орай циклды түрде қатарласып және (немесе) қайталануы мүмкін. Әрбір саты мақсаттары мен шығуларын қоюмен сипатталады. Өмірлік циклдың процестері мен әрекеттері әрбір сатының мақсаттары мен нәтижелерін толық қанағаттандыру үшін осы сатыларда іріктеліп, орындалады.

Әртүрлі ұйымдар өмірлік циклының шеңберінде түрлі сатыларды қолдана алады. Алайда, әрбір сатыны осы сатыға жауапты ұйым өмірлік циклдың жоспарындағы ақпараттарды және алдыңғы сатыларда қабылданған шешімдерді тиісті түрде қараумен жүзеге асырады. Осылайша ағымдағы сатыға жауапты ұйым осы өмірлік циклдың келесі сатыларына қатысты қабылданған шешімдерді және құжаттамаларды жазуды жүргізеді.

Өмірлік циклдың моделі құрылатын және жұмыс істейтін жағдайларда және бағдарламалық қамсыздандырудың сипаттамасына байланысты болады.

Бағдарламалық қамсыздандыруды әзірлеудің каскадты стратегиясы

Каскадты стратегия әзірлеу сатыларының сызықтық бірізділігін (бір реттік өтуін) білдіреді.

Бұл стратегия әзірленетін бағдарламалық құралға немесе әзірлеу процесінің басындағы жүйеге қойылатын барлық талаптарды толық анықтауға негізделген. Әзірлеудің әрбір сатысы алдыңғы сатының аяқталуынан кейін басталады. Өнімнің аралық әзірлемелері бағдарламалық өнімнің рұқсаты ретінде таралмайды.

Каскадты стратегия кез келген қолданбалы саладағы түрлі жүйелерді әзірлеуге деген классикалық тәсілді көрсетеді. Бұл тәсіл бағдарламалық қамсыздандыруды әзірлеу үшін 1970 жылдары және 1980 жылдардың бірінші жартысында кеңінен қолданылды. Жобалаудың каскадты әдістері түрлі бағыттағы шетелдік және отандық әдебиеттерде: әдістемелік монографияларда, стандарттарда, оқулықтарда жақсы сипатталған. Каскадты схема бойынша жұмыстарды ұйымдастыру ресми түрде ұсынылып, түрлі салаларда кеңінен қолданылды.

Осылайша, тек теориялық негіздемелер ғана емес, сонымен қатар өнеркәсіптік әдістемелер мен стандарттардың болуы, сондай-ақ бұл әдістерді ондаған жылдар бойы қолдану каскадты әдістерді классикалық деп атауға мүмкіндік береді.

Каскадты стратегияны іске асыратын модульдердің өкілдері каскадты және V-тәрізді модель өкілдері болып табылады.

Каскадты стратегияның *жетістіктеріне* мыналарды жатқызуға болады:

- әзірлеменің ӨЦ бойында талаптарының тұрақтылығы;
- әзірлеудің сатыларының тек бір өту қажеттілігі, бұл стратегияны қолданудың қарапайымдылығын қамтамасыз етеді;
- жобаны жоспарлау, бақылау және басқару қарапайымдылығы;
- тапсырыс берушінің түсінуі үшін қолжетімділігі.

Каскадты *стратегияның негізгі кемшіліктері* оны жобада қолданғанда пайда болатын кемшіліктері болып табылатындар:

- әзірлеу процесінің басында талаптардың толық қалыптасу күрделілігі және өмірлік цикл ішінде олардың динамикалы түрде өзгеруіне мүмкіндігінің болмауы;

- әзірлеу процесінің сызықтық құрылымы;
- аралық өнімдердің болмауы;
- пайдаланушының талаптарды әзірлеу кезінде және қабылдау сынақтары кезінде әзірлеу процесіне жеткілікті түрде қатыспауы, бұл бағдарламалық қамсыздандыру сапасын пайдаланушының алдын-ала бағалай алмауына әкеледі.

Бұл тәсілді *қолдану тиімдірек:*

- аларды нақты, ӨЦ кезінде өзгермейтін талаптармен және түсінікті түрде іске асырумен әзірлеу кезінде;
- жобаларды жоғары емес қиындықпен әзірлеуге, мысалы:

✓ әзірлеушілер әзірлеген осы үлгідегі бағдарламалық құралды немесе жүйені құру;

✓ қолданыстағы бағдарламалық құралдың немесе жүйенің жаңа нұсқасын құру;

✓ қолданыста бар өнімді жаңа платформаға ауыстыру;

- басқа стратегияны іске асыратын ӨЦ модельдерінің құрамдас бөлшегі ретінде үлкен жобаларды орындау.

Бағдарламалық қамсыздандыруды әзірлеудің инкрементті стратегиясы

Инкрементті тәсіл нәтижені жақсартудың жоспарланып алынған әзірлеу сатыларынан бірнеше рет өтуін болжайды.

Инкрементті әзірлеме - бұл сатылы және келесі уақытша стратегия кестелеріне, мұндай жүйенің түрлі бөлшектері түрлі уақытта әзірленеді және түрлі қарқынмен әзірленеді, егер бір бөлігі дайын болса, онда оны жүйеге біріктіреді.

Балама стратегия жүйенің барлық бөлшектерін кодтау шешімі болса, ал содан кейін барлық кодты бірден біріктіру.

Бұл стратегия әзірлеу процесінің басында әзірленетін бағдарламалық қамсыздандыруға қойылатын барлық талаптарды толық анықтауға негізделген. Инкрементті әзірлеу негізіне жататын идея бағдарламалық жүйені өсу принципі бойынша әзірлеу қажет екендігінен тұрады, мұнда әзірлеуші әзірлеу кезінде бұрынғы бағдарламалық қамсыздандыру нұсқаларын (релиздердің) әзірлеу кезінде алынған деректерді қолдана алады. Жаңа деректер бағдарламалық қамсыздандыруды әзірлеу барысында алынғандай, сонымен қатар мүмкін болатын пайдалану барысында да алынады. Бұл процестің тірек сатылары - бағдарламаға қойылатын талаптардың көптегенін іске асыру мен бірізді релиздерді бағдарламалық қамсыздандыру іске аспағанша модельді жетілдіру.

Әрбір итерация барысында модель ұйымы өзгереді де, оған жаңа функционалдық мүмкіндіктер қосылады.

Бағдарламалық қамсыздандырудың сипаттамасы, жоба мен сату бірінен кейін бірі әзірленетін бөліктерге (increment) бөлінеді.

Осылайша, жүйенің бөлшектерін ауыстыруға қажеттілердің саны азаяды және клиенттер өздерінің ниеттерін біршама ұзақ уақытқа ойланып алуға мүмкіндік алады. Әдеттегі әрекеттерге сонымен қатар, әзірленетін өнімнің компоненттері (бөлшектері) пайдалануда болатындығын, олар клиентке өнімге қоятын талаптарын бұдан әрі анықтауында үлкен анықтылық алатындығын жатқызу керек.

Әрекеттерінің барысы келесідей болады: ең алдымен талаптар біршама жалпы түрде анықталады және өте маңызды және маңыздылығы жай деп ажыратылады. Бұдан кейін тапсырыс беруші ала бастайтын бағдарламалық қамсыздандырудың бөлігі анықталады.

Әрбір жеткізу жүйеге белгілі бір функционалдылықты қосады. Бұл ретте шығару басымдылығы жоғары болатын компоненттерден (бөлшектерден) басталады.

Жүйенің бөлшектері анықталған кезде бірінші бөлігін алады да, біршама сәйкес келетін процесті қолдана отырып, егжей-тегжейлі етуді бастайды. Осы уақыт ішінде тоқтатылып қалған осы жұмыстың талаптарының ағымдағы үйлесіміндегі басқа бөліктер үшін талаптарды анықтап алуға болады. Егер өте қажет болса, осы бөлігіне қайта келуге болады.

Егер бөлігі дайын болса, онда тапсырыс берушіге жұмысында қолдану үшін немесе кем дегенде, сынауға болатын бөлігі жеткізіледі. Бұл тапсырыс берушіге келесі компоненттер үшін немесе осы компоненттің келесі нұсқалары үшін қойылатын талаптарды анықтап алуына мүмкіндік береді. Жаңа бөліктері тиісті жүйемен жапсарласады.

Инкрементті стратегияның ерекшеліктері көршілес циклдердің итерациялары арасындағы циклдың азғантай ұзақтылығы мен кейбір ерекшеліктерінде әзірлеудің циклдарының көптеген саны болып табылады.

Инкрементті тәсіл әдетте каскадты модель элементтері мен прототиптеуді біріктіруге негізделген. Бұл ретте прототиптеуді қолдану әрбір инкрементті әзірлеудің ұзақтығы мен жалпы бүкіл жобаның ұзақтылығын біршама қысқартуға мүмкіндік береді.

Прототип деп әзірленетін бағдарламалық қамсыздандырудың жұмыс моделі айтылады, ол түрін өзгертуге жеңіл түсетін және ұлғайтылатын, оның толық іске асырылуына дейін тірек қасиеттері туралы түсінік алуға мүмкіндік береді.

Инкрементті стратегияны заманауи іске асыру экстремалды

бағдарламалау болып табылады.

БҚ және жүйелерді әзірлеудің бұл стратегиясы Microsoft компаниясында қолданылады. Мұнда бағдарламалық құралдың әрбір нұсқасына мыңға жуық инкремент әзірленеді. Инкрементті әзірлеу кезеңі бір тәулікті (мысалы, күндіз инкремент әзірленеді, кешке тестіленеді) құрайды. Бірқатар ұйымдарда инкрементті әзірлеудің апталық мерзімі қолданылады (көбінесе бес күн - әзірлеу, екі күн - тестілеу).

Инкрементті стратегияның *негізгі жетістіктері* оның тиісті жобаны әзірлеу кезінде пайда болатын:

- әрбір инкрементті іске асырғаннан кейін функционалдық өнімді алу мүмкіндігі;
- инкрементті жасаудың қысқа жалғасы; бұл бастапқы жеткізу мерзімдерін қысқартады, бағдарламалық өнімнің алғашқы және келесі жеткізілуіне жұмсалатын шығындарды азайтуға мүмкіндік береді;
- белгілі бір инкрементті құру кезіндегі талаптардың тұрақтылығы және өзгерген талаптарды есепке алу мүмкіндігі;
- каскадты стратегиямен салыстырғанда тәуекелдің азаюы;
- пайдаланушылар процесіне қосылу, ол әзірлеудің біршама бұрынғы кезеңдерінде өнімнің функционалдық мүмкіндіктерін бағалауға мүмкіндік беріп, бағдарламалық қамсыздандырудың сапасын арттырып, сондай-ақ оны әзірлеуге жұмсалатын уақыт пен шығынды азайтуға мүмкіндік беруі болып табылады.

Инкрементті стратегияның *негізгі кемшіліктеріне* жатқызуға болады:

- жүйенің немесе бағдарламалық құралдың өмірлік циклының басында инкременттерді жоспарлау мен жобаны басқаруды қамтамасыз ету үшін толық функционалдылығын толық анықтау қажеттілігі;
- жұмыстарды жоспарлау мен бөлудің күрделілігі;
- қиын проблемаларды кейінгі инкременттерге жылжыту үрдісіне байланысты адами фактордың туындауы, бұл бағдарламалық өнімнің сапасын төмендетіп немесе жұмыс кестесін бұзуы мүмкін.

Бұл стратегияны *қолдану* біршама *тиімді:*

- жобаларды әзірлеу кезінде мұнда көптеген талаптарды бұрынырақ қалыптастыруға болады (бірақ, олардың кейбіреуі белгілі бір уақыт аралығынан кейін анықталуы мүмкін);
- нарыққа негізгі функционалдық қасиеттері бар өнімді жылдам жеткізу қажеттілігі;
- тәуекел деңгейі төмен немесе орташа жобаларды әзірлеу.

Бағдарламалық құралдар мен жүйелерді әзірлеудің эволюциялық стратегиясы

Эволюциялық стратегия әзірлеу сатыларының бірнеше өткелдерін білдіреді.

Бұл стратегия әзірленетін бағдарламалық құралға немесе жүйеге әзірлеу процесінің басында қойылатын талаптарды жекелей анықтауға негізделген. Талаптар әзірлеудің бірізді циклдарында біртіндеп анықталады. Әзірлеудің әрбір циклының нәтижесі бағдарламалық өнімнің кезекті жеткізілетін нұсқасын береді.

Жалпы алғанда, эволюциялық стратегия үшін инкрементті стратегиямен салыстырғанда циклдың ұзақ жалғасуында әзірлеудің циклдерінің саны аз болуы тән. Бұл ретте әзірлеудің әрбір циклының нәтижесі (бағдарламалық өнімнің кезекті нұсқасы) алдыңғы циклдың нәтижелерінен біршама көбірек айырмашылығы бар.

Инкрементті стратегия кезінде сияқты эволюциялық стратегияны іске асырғанда көбінесе прототиптеу жиі қолданылады.

Бұл жағдайда прототиптеудің негізгі мақсаты талаптарды толық түсіну болып табылады. Ол жобаны әзірлеудің өнімділігі жоғары және шығындардың бір уақытта азаюымен қолжеткенде өнімге қойылатын талаптарды итеративті түрде анықтауға мүмкіндік береді. Прототиптеуді қолдану жаңа концепциялар немесе жаңа технологиялар жобада қолданылған жағдайларда біршама тиімдірек, себебі бұл жағдайларда әзірлеудің алғашқы ерте циклдарында жүйеге немесе бағдарламалық құралға қойылатын техникалық талаптарды егжей-тегжейлі толық және дұрыс әзірлеу жеткілікті дәреже күрделі болады. Прототиптеуді қолданған кезде талаптарды итеративті анықтап ау үшін әзірлеу циклына тапсырыс беруші қатысуы тиіс.

Эволюциялық стратегияны іске асыратын модельдердің өкілдері шиыршықты модельдер болып табылады.

Эволюциялық стратегияның *негізгі жетістіктері* болып табылатындар:

- әзірлеу процесінде жаңа талаптарды анықтап алу және енгізу мүмкіндігі;
- аралық өнімнің пайдалану үшін жарамдылығы;
- тәуекелді басқару мүмкіндігі;
- пайдаланушының жобаға кеңінен қатысуын қамтамасыз ету, ең алғашқы сатыларынан бастап, ол тапсырыс берушілер мен әзірлеушілердің арасындағы келіспеушіліктерді азайтып, жоғары сападағы өнімнің жасалуын қамтамасыз етеді;

- каскадты және инкрементті стратегиялардың артықшылықтарын іске асыру.

  Эволюциялық стратегияның оны таңдап алғанда туындайтын *кемшіліктеріне* жатқызу қажет:
- қажетті итерациялардың нақты санының белгісіздігі мен келесі итерацияда әзірлеу процесін жалғастыру үшін белгілерін анықтау қиындығы; бұл жүйенің немесе бағдарламалық құралдың соңғы нұсқасын іске асырудың кідіртуі мүмкін;
- жобаны жоспарлау мен басқарудың қиындығы;
- пайдаланушылардың жобаға белсенді қатысу қажеттілігі, ол әрдайым жүзеге аса бермейді;
- қуатты аспаптық құралдар мен прототиптеу әдістеріне қажеттілік;
- қиын проблемаларды шешуді келесі циклдарға жылжыту мүмкіндігі, бұл тапсырыс берушінің талаптарына алынған өнімнің сәйкессіздігіне әкелуі мүмкін.

Эволюциялық стратегияның кемшіліктер қатары инкрементті стратегияға да тән. Бұл стратегияны *пайдалану* біршама *тиімді:*

- талаптар тым күрделі, ертерек белгілі емес немесе анықтап алуды талап ететін жобаларды әзірлеу кезінде;
- күрделі жобаларды әзірлеген кезде, оның ішінде:
  - ✓ үлкен ұзақ мерзімді жобаларды;
  - ✓ аналогтары жоқ БҚ немесе жүйелердің жаңасын жасау жөніндегі жобаларды;
  - ✓ тәуекелі орташа және жоғары дәрежелі жобаларды;
  - ✓ концепцияны тексеру, техникалық жүзеге асуын көрсету жобалары немесе аралық өнімдер үшін;
- жаңа технологияларды қолданатын жобаларды әзірлеген кезде.

2.2. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ ӨМІРЛІК ЦИКЛЫНЫҢ КАСКАДТЫ МОДЕЛІ

Каскадты модель жұмыстарды жүйемен ұйымдастыруды көздейді. Бұл ретте ерекшелігінің негізі бүкіл әзірлемені сатыларға бөлу болып табылады, оның ішінде бір сатыдан келесі сатыға өту тек алдыңғы сатыдағы барлық жұмыстар аяқталғанда ғана мүмкін. Әрбір саты өзге әзірлеушілер командасы әзірлеуді жалғастыра алатындай жеткілікті құжаттаманың толық жиынтығын шығарумен аяқталады. Каскадты

модельдің сыртқы көрінісін есепке ала отырып, оны *сарқырамалы моделі* деп те атайды.

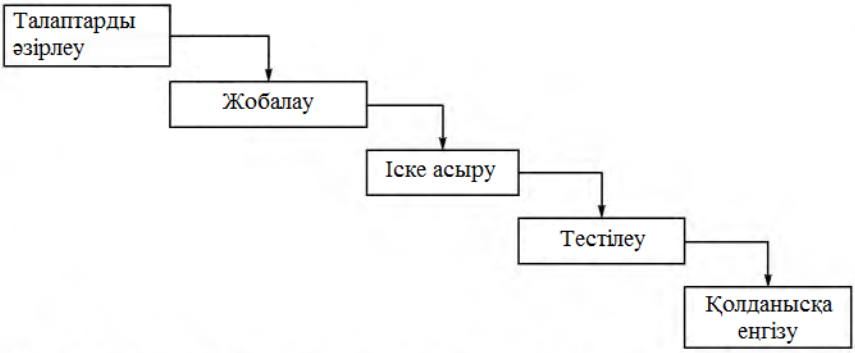
Каскадты модельдің пайда болған он жылы ішінде жұмыстарды сатыларға бөлу және осы сатыларға атау беру өзгерген жоқ. Сонымен қатар, біршама ақылға қонымды әдістемелер мен стандарттар нақты сатыларға белгілі бір жұмыстарды әдейі қоюдан алыс болды. Сонда да, тақырыптық салаға байланысты емес әзірлеудің тұрақты сатыларының қатарын бөліп көрсетуге болады (2.1-сурет).

Бірінші сатыда проблеманы зерделеу жүргізіледі, мұнда тапсырыс берушінің барлық талаптарының анық қойылуы шешіледі.

Бұл сатыда алынатын нәтиже барлық мүдделі тараптармен келісілген техникалық тапсырма (әзірлеуге арналған тапсырма) болып табылады.

Екінші сатыда техникалық тапсырмада анықталып берілген барлық талаптарды қанағаттандыратын жобалық шешімдер әзірленеді. Бұл сатының нәтижесі жобаны іске асыруға қажетті барлық қажетті деректерден тұратын жобалық құжаттама жиынтығы болып табылады.

Үшінші саты - жобаны іске асыру. Мұнда бағдарламалық қамсыздандыруды алдыңғы сатыда алынған жобалық шешімдеріне сәйкес әзірлеу (кодтау) жүзеге асырылады. Іске асыру үшін қолданылатын әдістер қатаң қабылданған мәнінде емес. Бұл сатының нәтижесі дайын бағдарламалық өнім болып табылады.



2.1-сурет. БҚ өмірлік циклының каскадты моделі

Тестілеу сатысында алынған бағдарламалық қамсыздандыруды техникалық тапсырыста көрсетілген талаптарға сәйкестікке тексеру жүргізіледі. Тәжірибелік пайдалану ақпараттық жүйенің жұмысының нақты жағдайларында көрінетін жасырын неше түрлі кемшіліктерді анықтауға мүмкіндік береді.

Соңғы саты - дайын жобаны тапсыру. Бұл сатының басты міндеті - тапсырыс берушіге барлық оның талаптары толық көлемде орындалғандығын дәлелдеп, көз жеткізу.

Модельдің барлық қадамдарында қажеттілігіне орай қосалқы және ұйымдастыру процестері орындалады, мысалы жобаны басқару, сапаны қамтамасыз ету, анықтап тексеру, аттестаттау, конфигурацияны басқару, құжаттау.

Модель қадамдарын аяқтау нәтижелері әзірлеудің аралық өнімдері болып табылады, олар келесі қадамдарда өзгере алмайды және бағдарламалық құралдың нұсқасы ретінде тапсырыс берушіге тапсырылмайды.

Классикалық каскадты модель әзірлеудің каскадты стратегиясына тән барлық жетістіктерге және кемшіліктерге ие.

Каскадты модельдің бірқатар оң жақтары бар, оның негізінде өзін түрлі инженерлік әзірлемелерді орындау кезінде жақсы көрсетіп, кеңінен таралу ауқымына ие болды. Оның *негізгі құндылықтарын* қарап шығайық.

Әрбір сатыда толықтылық және келісімділік белгілеріне жауап беретін жобалық құжаттаманың аяқталған жиынтығы қалыптасады. Қорытынды сатыларында да ақпараттық жүйенің стандарттармен көзделген барлық қамсыздандыру түрін (ұйымдастыру, әдістемелік, ақпараттық, бағдарламалық, аппараттық) қамтитын пайдаланушы құжаттамасы әзірленеді.

Логикалық жүйемен орындалатын жұмыс сатылары аяқтау мерзімдерін және тиісті шығындарды жоспарлауға мүмкіндік береді.

Каскадты модель түрлі текті инженерлік міндеттерді шешу үшін әзірленген және қазіргі уақытқа дейін қолданбалы аясы үшін өзінің мәнін жойған жоқ. Сонымен қатар, каскадты тәсіл белгілі бір бағдарламалық өнімдерді әзірлеу кезінде өзін жақсы ұсына білді. Яғни, әзірлеудің ең басында әзірлеушілерге ең жақсы техникалық жағынан алғанда барлық талаптарды толық және жеткілікті дәреже дәл қалыптастыруға болатын бағдарламалық жүйелер айтылады. Мұндай жүйелерге көбінесе күрделі есептеу жүйелері, нақты уақыттың жүйелерін жатқызады.

Сонда да, каскадты модельдің барлық құндылықтарына қарамастан оның ақпараттық жүйелерді әзірлеу кезінде қолданылуын шектейтін бірқатар кемшіліктері бар. Ал бұл кемшіліктер оны не толығымен қолданылмайтын етеді, не болмаса әзірлеудің мерзімін ұзартып, жоба құнын арттырады. Қазіргі таңда бағдарламалық жобалардың көптеген сәтсіздіктері әзірлеудің бірізді процестерімен түсіндіріледі.

Каскадты модельдің *кемшіліктерінің тізбесі* оны ақпараттық жүйелерді әзірлеуде қолдану үшін жеткілікті дәрежеде кең ауқымды. Алдымен оларды атап өтейік, ал содан соң олардың негізгілерін біршама толығырақ қарастырайық:

- нәтижелерді алуда кідірістің көбірек болуы;
- қателер мен толық жасалмау сатылардың кез келгенінде шығады, әрине ол келесі жұмыс сатыларында да, бұл кері қайту қажеттілігіне әкеледі;
- жоба бойынша жұмысты параллель жүргізудің қиындығы;
- жобаны басқару қиындығы;
- тәуекел деңгейінің жоғары болуы және инвестициялардың сенімсіздігі.

Нәтижелерді алудағы кідіру әдетте, каскадты схемасының бас кемшілігі болып саналады. Бұл кемшілік негізінде мүдделі тұлғалармен нәтижелерді келісу тек жұмыстың кезекті сатысы аяқталғаннан кейін ғана болу салдарынан туындайды. Әзірленіп отырған бағдарламалық қамсыздандыру пайдаланушылардың талаптарына сәйкес келуі мүмкін, мұндай сәйкессіздіктер әзірлеудің кез келген сатысында болуы мүмкін - бұрмалауды жобалаушы-талдаушылар да, бағдарламалаушылар да енгізуі әдейі болмауы мүмкін, себебі олар бағдарламалық қамсыздандыруды әзірлеу жүргізіліп отырған тақырыптық салада білікті болуы міндетті емес.

Сонымен қатар, бағдарламалық қамсыздандыруды әзірлеу кезінде қолданылатын ішкі келісім мен толық болу белгілеріне жауап беретін автоматтандырылған нысан модельдері әзірлеу уақыты ішінде түрлі себептерге байланысты (мысалы, заңнамаға өзгеріс енгізуге байланысты, валюта бағамының ауытқуына байланысты және т.с.с.) болуы мүмкін. Бұл функционалдық модельге де, ақпараттық модельге де, және де пайдаланушының интерфейсі жобаларына, пайдаланушы құжаттамасына да қатысты болуы мүмкін.

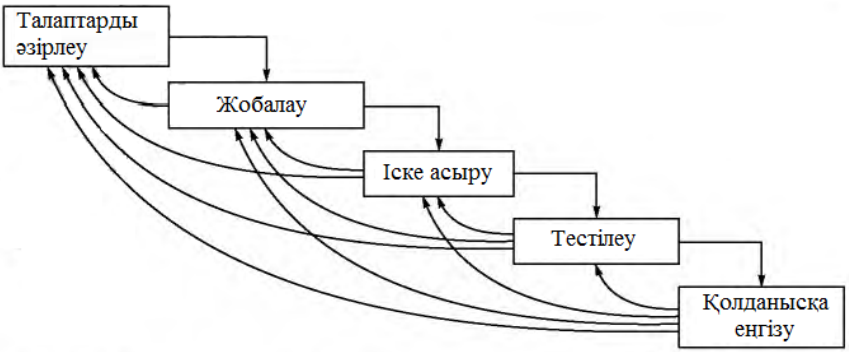
Бастапқы сатыларында жіберілген қателіктер, негізінде, жобамен жұмыс жасау барысындағы келесі сатыларында ғана анықталады. Сол себепті қателер шыққаннан кейін жоба алдыңғы сатысына қайтарылып, қайта толықтырылып, түзетіліп, келесі сатыға қайта тапсырылады. Бұл жұмыс кестесін бұзудың және жұмыстың жеке сатыларын орындайтын

топтардың арасында өзара қарым-қатынастың қиындауына себеп болуы мүмкін.

Ең жағымсызы алдыңғы сатының толық орындалмауы келесі сатыда бірден анықталмауы мүмкін, ол кейін (мысалы, тәжірибелік пайдалану сатысында тақырыптық саланың сипаттамасында қателер шығуы мүмкін) анықталуы мүмкін. Бұл жобаның бір бөлігінің бастапқы жұмыс сатысына қайтарылуы тиіс екендігін білдіреді. Жалпы, жұмыс кез келген сатысынан алдыңғы сатысына қайтарылуы мүмкін, онда әзірлеу схемасы 2.2-суретте көрсетілгендей болады. Бұл схема каскадты модель үзіндісін бұрын орындалған түрлі қадамдарға әзірлеу процесінің кейбір қадамдарынан қайтарылу мүмкіндігін суреттеп көрсетеді. Осыған байланысты жобаны жоспарлау мен қаржыландыру қиындықтары туындайды, әзірлеу кестесін бұзу тәуекелінің деңгейі жоғары.

Бұл жағдайдың себептерінің бірі тақырыптық саланың сипаттамасына қатысатын сарапшылар ретінде жүйенің пайдаланушылары болып табылады, олар не қажет екендігін анық түсіндіріп бере алмайды. Сонымен қатар, тапсырыс берушілер мен орындаушылар бірін-бірі жиі түсіне бермейді, орындаушылар әдетте шешілетін міндеттің тақырыптық саласындағы мамандар болып табылмайды, ал тапсырыс берушілер бағдарламалаудан алыс.

Жұмыстарды параллель жүргізу қиындығы да каскадты модельдің кемшіліктерінің бірі болып табылады. Аталған проблемалар жобамен жұмыс істеу бірізді қадамдардың тізбегі түрінде құрылатындықтан, соның салдарында пайда болады.



2.2-сурет. Алдыңғы сатыларға қайтып баратын кадкадты модель

Жобаның кейбір бөліктерін әзірлеуді параллель жүргізуге болатындығына қарамастан, каскадты схемада жұмыстардың параллель жүргізілуі бірқатар қиын. Жұмыстарды параллель жүргізудің қиындығы жобаның түрлі бөліктерін үнемі келісу қажеттілігімен байланысты. Жобаның жеке бөліктерінің өзара байланысы қаншалықты күшті болса, синхронизация соншалықты мұқият және жиі орындалуы тиіс, соншалықты әзірлеушілер тобы бір-біріне соншалықты байланысты болады. Сол себепті жұмыстарды параллель жүргізу артықшылықтары да жоғалады.

Параллелизмнің болмауы әзірлеушілер ұжымының барлығының жұмысын ұйымдастыруға да теріс әсерін береді. Бір топтың жұмысы басқа топпен іркіледі. Тақырыптық салаға талдау жүргізілгенше жобалаушылар, әзірлеушілер және тестілеумен және әкімшілік етумен айналысатын адамдар ешқандай жұмыспен жүктелмеген. Сонымен қатар, келесі әзірлеу кезінде жобаға сатыны аяқталғаннан кейін өзгерістер енгізу мен жобаны келесі сатыға беру қиын. Осылайша, мысалы егер жобаны келесі сатыға бергеннен кейін әзірлеушілер тобы біршама ұтымды шешім тапқан болса, онда ол қолданылмайды. Сондықтан да жобаны келесі сатыға бергеннен кейін толықтыруға болмайды (немесе, біршама қиындық туындатады).

Каскадты схеманы қолдану кезінде жобаны басқару қиындығы негізінен әзірлеу сатыларының қатаң бірізділігіне және жоба бөліктерінің арасындағы күрделі өазар байланыстың болуына негізделген. Жобаны әзірлеудің жүйелілігі бір әзірлеушілер тобының басқа командалардың жұмысының нәтижесін күтуіне әкеледі. Сондықтан да жұмыс мерзімдерін және берілетін құжаттама құрамын келісу үшін әкімшілік кірісу қажет. Орындалған жұмыста қате табылған жағдайда жобаны орындаудың алдыңғы сатысына қайту қажет. Қателік жіберген немесе дұрыс санамаған әзірлеушілер ағымдағы жұмысты (жаңа жобамен жұмысты) үзіп, қателерін түзетумен айналысуы керек. Әзірлеушілер командасынан әзірлеудің келесі сатысын аяқтауын талап ету ұтымды емес, себебі жұмыс уақытының біразын жоғалтады.

Әзірлеуші топтар арасындағы өзара байланысты жеңілдету және құжаттаманың ақпараттық шамадан тыс қанығуын азайту, жобаның жеке бөліктерінің арасындағы байланыс санын қысқарту арқылы мүмкін. Әдетте, бұл тым оңай емес.

Әрине, каскадты тәсілді қолданған кезде жобаның тәуекел деңгейі де артады. Жоба қаншалықты күрделі болса, әзірлеудің сатыларының әрқайсысының ұзақтығы да ұзақ және жобаның жеке бөліктерінің

арасындағы өзара байланыс та қиын, олардың саны да артады. Мұнда әзірлеу нәтижелерін нақты көріп, тестілеу сатысында ғана бағалауға болады, яғни талдау, жобалау және әзірлеу аяқталғаннан кейін - бұл сатыларды орындау көп уақыт пен қаражатты қажет етеді.

Бұрын да атап өткендей, кешіктіріліп берілген баға талдау мен жобалау қателерін анықтаған кезде қиын проблемаларды тудырады - жобаны алдыңғы сатыларға қайтару және әзірлеу процесін қайталау қажет болады. Алайда, алдыңғы сатыларға қайтару тек қателерге ғана байланысты емес, сонымен бірге тақырыптық салада болған өзгерістерге немесе тапсырыс берушінің әзірлеу уақыты ішіндегі талаптарындағы өзгерістеріне байланысты болады. Жобаны осы себептердің салдарынан толықтырып жасауға қайтару, тақырыптық саланың жобаның келесі нұсқасы дайын болғанша тағы өзгермеуіне кепілдік бола алмайды. Бұл әзірлеу процесінің «шырғаланып» қалу ықтималдылығын білдіреді және жүйе пайдалануға ешқашан берілмеуі мүмкін. Жобаға жұмсалатын шығындар да өседі, ал дайын өнімді тапсыру мерзімі үнемі кейінге қалдырылып отырады. Сондықтан, каскадты схема бойынша әзірленетін күрделі жобалардың тәуекел деңгейі жоғары болады.

Бұл әдіс бағдарламаны құру процесінің қаншалықты баяу екендігін, тіпті көбінесе бағдарламаға қойылатын талаптар әзірлеу аяқталғанға дейін өзгеріп үлгеретіндігін білдіреді.

Каскадты стратегияны қолдану салалары оның құндылықтарымен анықталады да, оның кемшіліктерімен шектеледі. Бұл стратегияны *қолдану* келесі жағдайларда біршама *ұтымды*:

- ӨЦ ішінде анық, өзгермейтін талаптары мен түсінікті іске асырылатын жобаларды әзірлеген кезде;
- күрделілігі жоғары емес жобаларды әзірлеген кезде (мысалы, егер осы үлгідегі жүйелер әзірленген болса);
- қолданыста бар бағдарламалық құралдың немесе жүйенің жаңа нұсқасын құрған кезде;
- бар өнімнің жаңа платформаға ауыстырылған кезінде;
- әзірлеудің басқа стратегияларын іске асыратын ӨЦ модельдерінің құрамдас бөлігі ретінде үлкен жобаларды орындаған кезде.

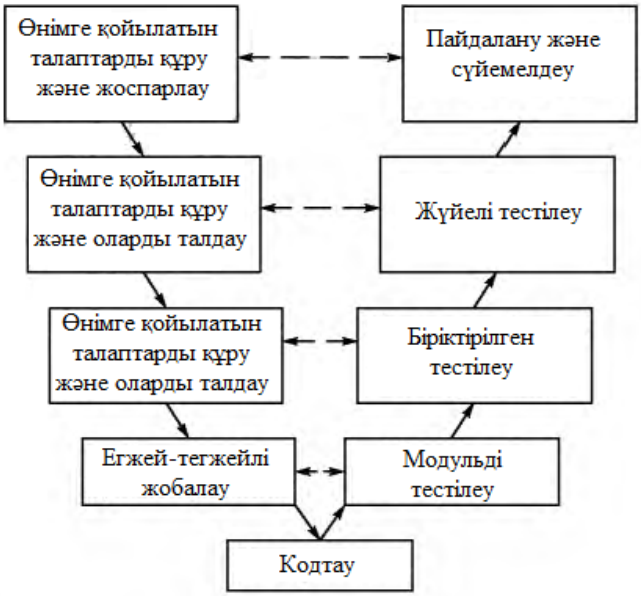
2.3. V-ТӘРІЗДІ МОДЕЛЬ

V-тәрізді модель каскадты модельдің бір түрі болып табылады және бағдарламалық өнімді әзірлеудің өмірлік циклының бастапқы сатыларынан бастап сынақты жоспарлауды қамтамасыз етеді. Бұл модельде БҚ анықтау мен аттестаттауға айрықша көңіл бөлінеді.

Бұл модель БҚ немесе жүйелерді әзірлеу процесінің сатыларын бір рет орындау стратегиясын қолдайды және талаптарды алдын-ала толық қалыптастыруға негізделеді. Классикалық V-тәрізді модельдің әрбір қадамы алдыңғы қадам сәтті аяқталғаннан кейін басталады.

V-тәрізді модельдің каскадты модельден ерекшелігі онда алдыңғы бағдарламалаудың қадамдарының арасында және тиісті тестілеу мен сынақ арасындағы байланыстың белгіленуі болады.

2.3-сурет бағдарламалық қамсыздандыруды әзірлеудің өмірлік циклының V-тәрізді модель нұсқасын көрсетеді. Модельде кодтау мен тестілеудің алдындағы жобалаудың аналитикалық фазалары арасындағы өзара байланыс жақсы көрінеді. Штрих көрсеткіштері параллель қарастырылу керек фазаларды көрсетеді.



2.3-сурет. Өмірлік циклдың V-тәрізді моделі

Бұл модель келесі сатылардан тұрады:

- өнімге қойылатын талаптарды құру және жоспарлау - жүйелік талаптар анықталады және жұмыстарды жоспарлау орындалады;
- өнімге қойылатын талаптарды құру және оларға талдау жасау - бағдарламалық өнімге қойылатын талаптардың толық сипаттамасы құрылады;
- жоғары деңгейлі жобалау - бағдарламалық қамсыздандыру құрылымы, оның компоненттері мен олардың іске асыратын функцияларының арасындағы өзара байланысы анықталады;
- егжей-тегжейлі жобалау - әрбір компоненттің алгоритмі анықталады;
- кодтау - алгоритмдерді дайын бағдарламалық қамсыздандыруға түрлендіру;
- модульді тестілеу - бағдарламалық қамсыздандырудың әрбір модулінің тексерілуі орындалады;
- біріктірілген тестілеу - жеке модульдердің бірлескен жұмысын тексеру;
- жүйелік тестілеу - жалпы алғанда жүйенің жұмысына байланысты проблемаларды анықтау. Бағдарламалық қамсыздандырудың нақты аппараттық платформада талаптардың сипаттамасына сәйкес тексерілуі орындалады.

V-тәрізді модель бағдарламалық қамсыздандыруды әзірлеудің каскадты стратегиясын қолдайды, онда ол осы стратегияның барлық артықшылықтарына ие. Бұдан бөлек, сәйкес түрде қолданғанда V-тәрізді модель келесі қосымша артықшылықтарға ие. Бұл модельде әзірлеудің ең бастапқы сатыларынан бастап ең үлкен рөл бағдарламалық қамсыздандыруды тексеру мен аттестаттауға беріледі, барлық әрекеттер жоспарланады. Бұл ретте тек бағдарламалық қамсыздандыру емес, барлық алынған ішкі және сыртқы деректер аттестатталады және тексеріледі. Артықшылығы әзірлеу процесінің барысын басқару мен бақылауды жеңілдету болып табылады.

V-тәрізді модельді қолданғанда оған сәйкес келмейтін жоба үшін келесі оның *кемшіліктері* анықталады:

- өмірлік циклде талаптарды тестілеудің кеш мерзімдері, ол жобаны орындау кестесіне талаптарды өзгерту қажеттілігінде біраз ықпалын тигізеді;
- басқа каскадты модельдерде сияқты тәуекелді талдауға бағытталған әрекеттердің болмауы.

V-тәрізді модельді бағдарламалық өнімдерді әзірлегенде, жоғары сенімділік басты талабы болып табылғанда *қолдану* мақсатқа сай.

2.4. RAD ЖЫЛДАМ ӘЗІРЛЕУ МОДЕЛІ

Қосымшаларды жылдам әзірлеу моделі (Rapid Application Development, RAD) өткен ғасырдың 80-нші жылдарында бағдарламалық құралдарды әзірлеудің аспаптық құралдарының қаулап дамуына байланысты шықты. RAD концепциясын визуалды бағдарламалау концепциясымен жиі байланыстырады. Сол себепті қосымшаларды жылдам әзірлеу процесінде негізгі назар бағдарламалау мен тестілеуге емес, талаптарды талдау мен жобалауға бөлінеді.

Бұл модель, оны іске асыру мен қолдану мақсаттарына сүйене отырып, бағдарламалық қамсыздандыруды әзірлеудің инкрементті сияқты, эволюциялық стратегиясын қолдай алады, негізінен, RAD-модельдер басқа модельдердің құрамында бағдарламалық құралды немесе жүйенің прототипін әзірлеу циклын жеделдету үшін қолданылады. RAD-моделі жобаларының жоғары емес қиындығы тәуелсіз модельдер ретінде қолданылуы мүмкін.

Әзірлеудің инкрементті немесе эволюциялық стратегиясын іске асыратын өмірлік цикл модельдері жылдам прототиптік ұғымын кеңінен қолданылады. RAD-моделі прототиптеу негізінде қолданылатын модельді береді.

Аспаптық құралдарды қолдану пайдаланушыны әрекетке қосуға мүмкіндік береді, яғни, салдарында оны әзірлеудің барлық сатыларында өнімге баға береді.

RAD-моделінің өзіне тән сипаты қысқа уақытта жүйенің немесе бағдарламалық құралдың талаптарын талдаудан толықтай оны құруға дейін өту болып табылады.

RAD тәсіл болуын болжайды:

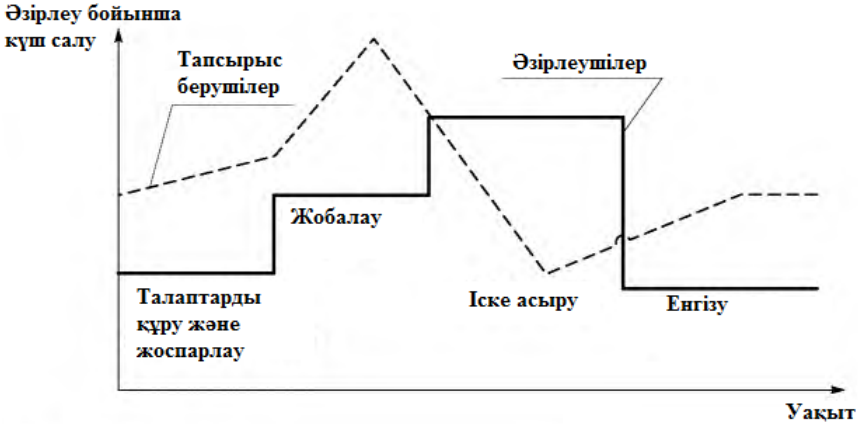
- бағдарламалық қамсыздандырудың қосалқы жүйелерін жобалау жөніндегі жұмыстарды орындайтын кәсіби әзірлеушілердің (7 адамға дейін) шағын топтары;
- жақсы жасалған өндірістік кестесі;
- тапсырыс берушімен өзара байланыс нәтижесінде алынатын талаптарды бағдарламалық өнімде циклды іске асыру.

Бағдарламалық қамсыздандырудың өмірлік циклы RAD тәсілімен бірге төрт сатыны қамтиды (2.4-сурет):

1) талаптарды талдау және қалыптастыру;
2) жобалау;
3) сату;
4) енгізу және сүйемелдеу.

Талаптарды талдау және қалыптастыру сатысында:

- жүйе орындауы керек функцияларды анықтайды;



2.4-сурет RAD-модель

- ең алдымен толықтай жасалуын талап етілетін басым функциялар бөлінеді;
- ақпараттық қажеттіліктерді сипаттайды.

Жүйеге қойылатын талаптарды қалыптастыру негізінде пайдаланушылар мен әзірлеуші-мамандардың бірлескен күшімен жүзеге асырылады. Сонымен қатар, осы сатыда келесі міндеттер шешіледі:

- жоба шекаралары белгіленеді;
- әрбір келесі сатылардың әрқайсысы үшін уақытша шеңберлері анықталады;
- жобаның іске асырылу мүмкіндігінің өзі техникалық қамтамасыз ету, қаржыландырудың берілген көлемінде шешіледі және т.с.с.

Сатыны аяқтау нәтижесінде болашақ бағдарламалық қамсыздандырудың функциялар тізбесі қалыптастырылып, БҚ алдын-ала модельдері құрылуы тиіс.

Жобалау сатысында пайдаланушылардың бірқатары әзірлеуші-мамандардың басшылығымен жобаны жобалауға қатысады. Қосымшаның жұмыс істейтін прототиптерін жылдам алу үшін тиісті аспаптық құралдар - CASE-құралдары қолданылады. Пайдаланушылар әзірлеушілермен тікелей өзара байланыса отырып алдыңғы сатыда анықталмаған жүйені техникалық жобалауға қойылатын талаптарды анықтап алады және толықтырады. Осы сатыда жүйенің процестері біршама егжей-тегжейлі қарастырылады, жиі прототиптері (экран формасы, диалог, есеп) құрылады, қажетті құжаттама құрамы анықталады.

Іске асыру сатысында қосымшаларды жылдам әзірлеу өтеді. Әзірлеушілер бағдарламалық қамсыздандыруды модельдің алдыңғы сатыларында алынған негізде итеративті құруды жүргізеді, сонымен қатар сенімділікке, өнімділікке және т.с.с. қойылатын талаптарды құрады. Пайдаланушылар алынатын нәтижелерді бағалайды және түзетілер енгізеді, егер БҚ әзірлеу процесінде бұрынғы талаптарды қанағаттандырмайтын болса. Бағдарламалық қамсыздандыруды тестілеу әзірлеу процесінде жүзеге асырылады. Сатының нәтижесі барлық келісілген талаптарды қанағаттандыратын дайын бағдарламалық өнім болып табылады.

Енгізу сатысында пайдаланушыларды оқыту жүргізіледі, ұйымдастырылған өзгерістер және жаңа бағдарламалық өнімді енгізумен параллель қолданыстағыны жаңа толық ендірілгенше пайдалану жалғаса береді. Жоспарлау мен енгізуге дайындау бағдарламалық өнімді жобалау сатысында басталуы тиіс.

RAD тәсілі әмбебаптылыққа ұсына алмайтындығын ескере кеткен дұрыс. Ол ең алдымен, салыстырмалы түрде үлкен емес, нақты тапсырыс беруші үшін әзірленетін жобалар үшін жақсы, және де тапсырыс беруші әзірлеу процесінде тікелей қатыса алады. Егер ауқымды бағдарламалық жүйе әзірленетін болса, онда мұндай жоба үшін жоспарлаудың жоғары деңгейі, қатаң жобалау, алдын-ала әзірленген хаттамаларды қатаң сақтау қажет, ол әзірлеу жылдамдығын да азайтады. RAD тәсілі күрделі есеп бағдарламаларын, нақты уақыт масштабында күрделі нысандарды басқару бағдарламалары немесе операциялық жүйелерін құру үшін қолданылмайды, яғни үлкен көлемдегі кодтардан тұратын бағдарламалар үшін. Жылдам әзірлеу адамдардың қауіпсіздігіне қатысты (мысалы, ұшақты немесе атом электр станциясын басқару) қосымшалар үшін қолданылмайды, себебі итеративті тәсіл алдыңғы бірнеше нұсқа толыққанды жұмысқа қабілетті болмайды, ол бұл жағдайларда болмауы қажет.

Қарастырылған тәсілді БҚ қойылатын талаптары жақсы анықталған модельдеуге жақсы келетін бағдарламалық өнімдерді әзірлеу кезінде қолдануға болады.

RAD-модельдерді оған сәйкес жобада қолданған кезде оның келесі *артықшылықтары* шығады:

- әзірлеу циклының және жалпы бүкіл жобаның ұзақтығын қысқарту;
- әзірлеушілердің санын қысқарту;
- алдыңғы факторлардың салдары - жоба құнын азайту (сонымен бірге қуатты аспаптық құралдарды қолдану есебінен);
- кестені сақтамау тәуекелін қысқарту;
- тапсырыс берушінің алынған бағдарламалық өнімге қанағаттанбауымен байланысты тәуекелді әзірлеу циклына оны

тарту есебінен қысқарту;

- әзірленген компоненттерді қайта пайдалану мүмкіндігі; бұл артықшылығы RAD-моделін инкрементті немесе эволюциялық модель құрамында қолданғанда пайда болады. Бұл жағдайда функционалдық мүмкіндіктерін көбейту бұрын әзірленген компоненттер негізінде жүзеге асырылады. RAD-моделіне сәйкес келмейтін жобада қолданғанда оның *негізгі кемшіліктері* болып табылады:
- пайдаланушылардың әзірлеу процесінде үнемі қатысуға қажеттілігі, бұл әрдайым орындала бермейді және соңғы өнімнің сапасына әсер етуі мүмкін;
- прототипті әзірлеуге уақытша шектеулердің қатаңдығы;
- өніммен жұмыс істеуді аяқтау мерзімдері мен шығындарын анықтау және шектеу қиындығы.

RAD-моделі бағдарламалық өнімдерді келесі жағдайларда әзірлеген кезде *ұтымды қолданылуы* мүмкін:

- бағдарламалық өнімдер модельдеуге келеді;
- бағдарламалық өнімдер сын көзбен қарайтын емес болып табылады;
- бағдарламалық өнімнің көлемі үлкен емес;
- бағдарламалық өнімдердің өнімділігі төмен;
- бағдарламалық өнімдер әзірлеушілерге танымал тақырыптық салаларға жатады;
- бағдарламалық өнімдер ақпараттық жүйелер болып табылады;
- бағдарламалық қамсыздандыруға арналған талаптар жақсы танымал;
- әзірленетін БҚ компоненттерге қайта пайдалануға жарамдылары бар;
- егер пайдаланушы әзірлеу процесінде үнемі қатысатын болса;
- егер жобаға аспаптық құралдарды әзірлеуді қолдануда жеткілікті дағдысы бар әзірлеушілер қатысатын болса;
- БҚ әзірлегенде функционалдық мүмкіндіктерді жылдам өсіру қажет болса;
- техникалық тәуекелдердің дәрежесі жоғары болмағанда;
- өмірлік циклдың басқа модельдерінің құрамында.

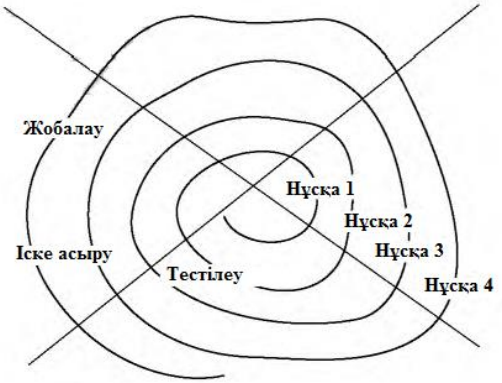
2.5. ӨМІРЛІК ЦИКЛДЫҢ ШИЫРШЫҚТЫ МОДЕЛІ

Өмірлік циклдың шиыршықты моделі каскадты модельдің аталған проблемаларын жеңу үшін ұсынылған болатын және жобалау сияқты, сатылы прототиптеу сияқты бағдарламалық қамсыздандыруды әзірлеу процесін береді. БҚ процесі шиыршықты түрінде берілуінде, әрбір шиыршықты орамында бағдарламалық өнімнің жаңа нұсқасы немесе жаңа прототип құрылады.

Өмірлік циклдың шиыршықты моделі бағдарламалық қамсыздандыруды әзірлеудің эволюциялық стратегиясын қолдайды, мұнда өмірлік циклдың басында барлық талаптар анықтала бермейді. Шиыршықты модельдің негізгі концепциясы келесіде болады. Бағдарламалық өнім нұсқалардың бірізді түрінде құрылады. Нұсқалардың әрқайсысы талаптардың кейбір көптеген түрлерін іске асырады. Әрбір нұсқаны іске асырғаннан кейін талаптар анықталып алынады.

Шиыршықты модель (2.5-сурет) каскадты модельге қарағанда бағдарламалық қамсыздандыруды әзірлеудің эволюциялық (итерациялық) процесін болжайды. Бұл ретте талдау мен жобалау сияқты өмірлік циклдың бастапқы сатыларының мәні артады. Бұл сатыларда прототиптерді құру арқылы техникалық шешімдердің іске асырылуы тексеріледі және негізделеді.

Әрбір итерация әзірлеудің аяқталған циклын береді, ол аяқталған жүйе болу үшін итерациядан итерацияға жетілдірілетін бұйымның (немесе көптеген соңғы өнімнің) ішкі немесе сыртқы нұсқаларын шығаруға әкеледі.



2.5-сурет. БҚ өмірлік циклының шиыршықты моделі

Әрбір итерацияда жоба бөлшектері тереңдетіліп және ізінше нақтыланады, соның нәтижесінде соңғы іске асыруға дейін жеткізілетін негізделген нұсқа таңдап алынады.

Шиыршықты модельді қолдану жобаны орындаудың келесі сатысына өтуді жүзеге асыруға мүмкіндік береді, ол ағымдағы сатының аяқталуын күтуді қажет етпейді - толық орындалмаған жұмысты келесі сатыда орындауға болады. Әрбір итерацияның басты міндеті - жүйе пайдаланушыларына көрсетуге болатын жұмысқа қабілетті өнімді тезірек көрсету. Осылайша, жобаға анықтап алу мен толықтыруларды енгізу процесі де біршама жеңілдейді.

Шиыршықты модельдің артықшылықтары. Бағдарламалық қамсыздандыруды әзірлеуге шиыршықты тәсіл каскадты модельдің біршама кемшіліктерін өткеруге мүмкіндік береді және сонымен қатар, әзірлеу процесін икемді ете отырп, қосымша мүмкіндіктердің қатарын қамтамасыз етеді.

Маңызды артықшылықтарының бірі тәуекел деңгейін азайту болып табылады. Тәуекел біріктіру кезінде анықталады, сол себепті тәуекел деңгейі жобаны әзірлеуді бастаған кезде максималды. Әзірлеудің алға жылжуына қарай тәуекелдің күтілген деңгейі азаяды. Бұл кез келген әзірлеу моделі кезінде әділетті, алайда шиыршықты модельді қолданғанда тәуекел деңгейінің азаюы көбірек үлкен жлдамдықпен болады. Бұл итерациялы тәсілде біріктіру бірінші итерацияда болатындығымен байланысты, және бастапқы итерацияларда жобаның көптеген аспектілері, мысалға, қолданылатын аспаптық құралдардың және бағдарламалық қамсыздандырудың жарамдылығы, әзірлеушілердің біліктілігі және т.с.с. анықталатын болады. 2.6-суретте кестелерді салыстырғанда тәуекел уақытының каскадты және шиыршықты модель үшін әзірлеу уақытына тәуелділігі берілген.

Итерациялық әзірлеме жобаны басқаруда үлкен икемділікті қамтамасыз етеді, әзірленетін бұйымға тактикалық өзгерістерді енгізуге мүмкіндік береді. Мысалы, әзірлеу мерзімдерін жүйенің функционалдылығын қысқарту арқылы болады немесе жүйенің құрамдас бөлшегі ретінде өз әзірлемелерінің орнына басқа тараптық фирмалардың өнімдерін қолдану. Бұл бәсекелестердің ұсынған өнімдерінің алға жылжуына қарсы келе алатын бәсекелестік жағдайларда өзекті болуы мүмкін.

2.6-сурет. Тәуекел деңгейінің әзірлеу уақытына тәуелділік графикалары

Итерациялық тәсіл компоненттердің қайта пайдаланылуын жеңілдетеді (бағдарламалауға компонентті тәсілді іске асырады). Бұл жобаның жалпы бөлімдерін, олар жекелей әзірленгендігін анықтауға қолайлы болу үшін (сәйкестендіру) қажет, оларды жобаның басында белгілеп алуға қарағанда жеңіл. Бірнеше бастапқы итерация өткізуден кейін жобаға талдау жасау жетілідірілетін келесі итерацияларда қолданылатын компоненттерді бірнеше рет анықтауға мүмкіндік береді.

Шиыршықты модель едәуір сенімді және тұрақты жүйені алуға мүмкіндік береді. Бұл жүйені дамытуға қарай қателер мен әлсіз орындарды анықтауға және әрбір итерацияда түзетілуіне байланысты бір уақытта тиімділіктің сыни параметрлері түзетілуі мүмкін, ол каскадты модель жағдайында тек жүйеге енгізу алдында қолжетімді.

Итерациялық әдіс әзірлеу процесін жетілдіруге мүмкіндік береді - әрбір итерацияның соңында өткізілетін талдау әзірлеуді ұйымдастыруда не өзгертілуі тиіс және келесі итерацияда оны жақсарту қажеттілігіне бағалауға мүмкіндік береді.

Шиыршықты модельдің кемшіліктері. Шиыршықты циклдың негізгі проблемасы – келесі сатыға өту сәтін анықтау. Оны шешу үшін өмірлік циклдың сатыларының әрқайсысына уақытша шектеу енгізу қажет. Әйтпесе, әзірлеу процесі жасалғанның өзін үздіксіз шексіз жетілдіруге айналып кетуі мүмкін. Итерациялық тәсіл кезінде «ең жақсысы - жақсының қасы» деген принципті ұстанған дұрыс. Сол себепті тіпті егер де барлық жоспарланған жұмыс аяқталған болса да, итерацияны аяқтау тек жоспарға сәйкес қатаң жүргізілуі тиіс. Жұмыстарды жоспарлау әдетте алдыңғы жобаларда алынған статистикалық деректер мен әзірлеушілердің жеке тәжірибесі негізінде өткізіледі.

2.6. ЭКСТРЕМАЛДЫ БАҒДАРЛАМАЛАУДЫҢ ИНКРЕМЕНТТІ МОДЕЛІ

Экстремалды бағдарламалау (XP) - анық емес немесе жылдам өзгеретін талаптар жағдайында бағдарламалық өнімді жасаумен айналысатын шағын және орташа әзірлеушілер командасына арналған бағдарламаларды ұйымдастырудың жеңілдетілген әдіснамасы.

Экстремалды бағдарламалау негізіне - бір-үш аптаны құрайтын әзірлеудің үнемі қайталатын, өте қысқа циклы. Әрбір циклдың соңына Сіз қосымшалардың толық функционалды жұмыс және тестіден өткізілген релизіңіз болуы тиіс. Бұл циклдер бүкіл жоба бойы қайталанатын және үздіксіз болуы керек. Мұндай жұмыс режимі үшін алғышарт бірнеше рет тексерілген факті болып табылады, талаптар толық, уақытылы және дұрыс болуы сирек. Үнемі өзгеріп отыратын талаптардың салдары ретінде басқа принцип келеді - шешімдерді кеш қабылдау.

Модель икемділігімен ерекшеленеді, себебі ол талаптардың белгілі емес сипаттамасының жоғары дәрежесіне бағдарланған.

Осы модельге сәйкес әзірлеудің бірінші сатысында (сәулетті жобалау) әзірлеушілердің күші жүйені және/немесе бағдарламалық құралдың сәулетін мұқият жобалауға кетеді. Кейіннен сәулетті анықтап алу немесе өзгерту көзделмейді. Бұл сатының нәтижесі жүйенің моделі болып табылады.

Әзірлеудің екінші сатысында жүйенің (бағдарламалық құралдың) кезекті нұсқасын әзірлеу орындалады. Тапсырыс берушіден ағымдағы сәтке келіп түскен әрбір жаңа функционалды талап оның құны мен іске асыру уақытын есепке ала отырып, бағаланады. Бұл ретте берілген талаптардың белгісіздігін және оларды іске асырудың тәуекелдерін бағалау нәтижелері есепке алынады. Орындалған бағалауды есепке ала отырып, тапсырыс беруші жүйенің (бағдарламалық құралдың) кезекті нұсқасында іске асырылатын жаңа талаптарды таңдап алады және бекітеді.

Іске асырудың келесі сатылары итерациялық түрде орындалады. Тапсырыс берушінің жаңа талаптары жүйенің (бағдарламалық құралдың) жаңа нұсқасында іске асады, осы нұсқаны қабылдау сынақтары орындалады. Егер жүйенің (бағдарламалық құралдың) кезекті нұсқасында қателер табылған болса, онда бұл нұсқа іске асырудың келесі итерациясына қайтарылады. Процесс қабылдау сынақтарының кезекті нәтижелері тапсырыс берушімен бекітілгенге дейін қайталана береді. Жүйенің (бағдарламалық құралдың) бекітілген

нұсқасы кезекті нұсқа ретінде пайдалануға түседі.

ХР негізгі мақсаттары тапсырыс берушінің бағдарламалық өнімге деген сенімін әзірлеу процесінің даму сәттілігін анық дәлелдермен беру арқылы және өнімді әзірлеу мерзімдерін қысқартумен арттыру болып табылады. Бұл ретте ХР әзірлеудің алғашқы сатыларында қателерді азайтуға бағытталған. Бұл дайн өнімді шығарудың максимады жылдамдығына қолжеткізуге мүмкіндік береді және жұмыстың болжамы туралы айтуға мүмкіндік береді. ХР барлығы дерлік тәсілдері бағдарламалық өнімнің сапасын арттыруға бағытталған.

ХР процесі белгілі тәртіпте болып табылмайды, бірақ өзіндік тәртіптің жоғары деңгейін талап етеді. Егер бұл ереже орындалмайтын болса, онда ХР бірден бейберекет және бақыланбайтын процеске айналады.

ХР бағдарламашылардан көптеген есептерді жазуын және қаншама модельдерді құруын қажет етпейді.

ХР әрбір бағдарламалаушы білікті қызметкер болып табылады, ол өз жауапкершіліктеріне кәсіби түрде және үлкен жауапкершілікпен қарайды. Егер командада бұл болмаса, онда ХР енгізудің мәні де жоқ - ең жақсысы алдымен команданы құрумен айналысу қажет. Әзірлеу тәуекелі ХР керемет келетін командада ғана азаяды, ал басқа ХР жағдайларында - бұл тәуекелдің біршама жоғары әзірлеу процесі, себебі коммерциялық тәуекелді азайту әдістері, адами фактордан басқа, ХР жоқ.

БАҚЫЛАУ СҰРАҚТАРЫ ЖӘНЕ ТАПСЫРМАЛАР

1. БҚ әзірлеудің базалық стратегияларын атаңыз.
2. БҚ әзірлеудің каскадты стратегиясының мәнін сипаттап беріңіз.
3. БҚ әзірлеудің инкрементті стратегиясының мәнін сипаттап беріңіз, осы стратегияны қолдану артықшылықтарын, кемшіліктері мен саласын атап беріңіз.
4. БҚ әзірлеудің эволюциялық стратегиясының мәнін сипаттап беріңіз, осы стратегияны қолдану артықшылықтарын, кемшіліктері мен саласын атап беріңіз.
5. БҚ әзірлеудің каскадты, инкрементті және эволюциялық стратегиясының салыстырмалы сипаттамасын беріңіз.
6. Өмірлік циклдың каскадты модельдерінің жалпы қырларын атаңыз.

7. Өмірлік циклдың классикалық каскадты моделін бейнелеп, сипаттап беріңіз.
8. Өмірлік циклдың каскадты моделін кері байланыстарымен бейнелеп, сипаттап беріңіз. Классикалық каскадты модельмен салыстырғанда оның артықшылықтары мен кемшіліктері неде?
9. Өмірлік циклдың V-тәрізді моделін бейнелеп, сипаттап беріңіз. Классикалық каскадты модельмен салыстырғанда оның ерекшеліктері, артықшылықтары және кемшіліктері неде?
10. ӨЦ базалық RAD-моделін бейнелеп, сипаттап беріңіз. Классикалық каскадты модельмен салыстырғанда оның ерекшеліктері, айырмашылықтары және кемшіліктері неде?
11. RAD-модельдердің негізгі артықшылықтарын, кемшіліктерін және қолданылу саласын атаңыз.
12. Өмірлік циклдың инкрементті модельдерінің жалпы қырын атаңыз.
13. Өмірлік циклдың эволюциялық модельдерінің жалпы қырларын атаңыз.
14. Шиыршықты модельді бейнелеңіз және сипаттап беріңіз. Өмірлік циклдың осы моделінің артықшылықтары мен кемшіліктерін атаңыз.

3-бөлім

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУҒА ҚОЙЫЛАТЫН ТАЛАПТАРДЫ ҚАЛЫПТАСТЫРУ

3.1. ТАЛАПТАРДЫ БАСҚАРУ ЖӨНІНДЕГІ ЖАЛПЫ МӘЛІМЕТТЕР

Бағдарламалық өнімді құру оған қойылатын талаптарды жинаудан және жеңілдетуден басталады.

Талап - бұл бағдарламалық қамсыздандыруды қанағаттандыруы тиіс шарт немесе сипаттамасы.

Бастапқыда жиналатын талаптар *тапсырыс берушінің алғашқы талаптары* (ТАТ) деп аталады. Тапсырыс берушінің алғашқы талаптары жиналыс хаттамаларын, тапсырыс берушімен және пайдаланушылармен сұхбаттарды, түрлі құжаттардың көшірмелері мен түпнұсқаларын, ұқсас бағдарламалық өнімдер және өзге материалдар туралы мәліметтерді қамтиды.

Бұдан кейін *бағдарламалық қамсыздандырудың компоненттеріне қойылатын талаптар* - бағдарламалық және техникалық құралдарға, деректер қорына сәйкестігі.

Тапсырыс берушінің алғашқы талаптары бір-біріне қарама-қайшы болмауы тиіс және де пайдалы ақпараттан тұратын «артық» құжаттар болмауы тиіс. негізінде, талаптар мен тілектер құрылымдалмаған, көбінесе қарама-қайшы, барлық келісімдер, жұмыс жиналыстарының хаттамалары, тексерудің алғашқы жазылған материалдары бойынша шашыраңқы болуы мүмкін.

Салдарында, осы талаптарды орындауды тіркеу мен бақылау жөніндегі ұйымдаспаған күшсіз барлық талаптарды есепке алмау үлкен қауіп.

Проблеманы шешу жеткілікті екендігі сөзсіз: жиналатын талаптарды есепке алуды жүргізу қажет және олардың өңделуін, бағалануын және іске асырылуын (немесе іске асырылудан істен шығу) бақылау керек. Мұндай жұмыс талаптарды басқару жөніндегі жұмыс деп аталады.

Бағдарламалық қамсыздандыруға қойылатын ***талаптарды басқару***

(requirements management) білдіреді:

- бағдарламалық қамсыздандыруға қойылатын талаптарды анықтауға, ұйымдастыруға және құжаттауға жүйелік тәсіл;
- тапсырыс берушілер мен әзірлеушілердің арасындағы бағдарламалық қамсыздандыруға қойылатын талаптарға қатысты келісімді белгілейтін процесс.

Бағдарламалық қамсыздандыруға қойылатын талаптарды басқару келесі мақсаттарды көздейді:

- тапсырыс беруші мен пайдаланушының бағдарламалық өнім не істеу керектігі туралы келісімге жету;
- әзірлеушілердің тарапынан бағдарламалық қамсыздандыруға қойылатын талаптарды түсінуді жақсарту;
- бағдарламалық қамсыздандыру шекараларын белгілеу, яғни компьютер аппаратурасына, операциялық ортаға және бағдарламалық қамсыздандырудың мүмкіндіктеріне қойылатын техникалық талаптарды анықтау;
- жоспарлау үшін базисті анықтау.

Басқару мүмкіндігі әзірлеу кезінде басымдықтарды белгілеу керек болады, ол барлық талаптарды үш санатқа бөледі: 1) «орындау қажет»; 2) «орындау керек»; 3) «орындауға болады».

Функционалды және функционалды емес талаптар бар.

Функционалды талаптар бағдарламалық қамсыздандыру орны іске асырумен байланысты шектеулерді ескерусіз орындауға болатын әрекеттерді анықтайды. Басқаша айтар болсақ, олар бағдарламалық өнімнің ақпаратты өңдеу процесіндегі әрекетін анықтайды.

Функционалдық емес талаптар бағдарламалық өнімнің әрекетін анықтамайды, бірақ оның атрибуттарын немесе жүйелік ортаның атрибуттарын сипаттайды. Функционалдық емес талаптардың келесі түрлерін ажыратуға болады:

- *қолдануға қойылатын талаптар* пайдаланушы интерфейсінің, құжаттама мен оқу курстарының сапасын анықтайды;
- *өнімділікке қойылатын талаптар* функционалдық талаптарға шектеулерді қояды, ресурстарды пайдаланудың қажетті тиімділігін, өткізу қабілеті мен әрекет ету уақытын береді;
- *іске асыруға қойылатын талаптар* белгілі бір стандарттарды, бағдарламалау тілінің, операциялық ортаның және т.б. қолданылуына алғышарт болады;
- *сенімділікке қойылатын талаптар* рауалы жиілікті негіздейді және бағдарламалық қамсыздандырудың жұмысына істен шығудың әсеріне негіз болады, сондай-ақ істен шығудан кейін БҚ қалпына келу мүмкіндіктерін негіздейді;

- *интерфейске қойылатын талаптар* сыртқы мәнін анықтайды (яғни, пайдаланушылардың және кез келген сыртқы құрылғының), жүйе өзара әрекет ететін және осы өзара әрекет етудің регламенті.

Талаптарды қалыптастыру сатысын болашақ бағдарламалық қамсыздандырудың сәулетінің қалыпты емес анықтамасының сатысы деп атауға болады. Бұл саты біршама күрделі және жауапты болып табылады. Жобамен жұмыс істеу процесінде тапсырыс берушіде де, орындаушыда да бағдарламалық қамсыздандыру туралы түсінік өзгереді. Сол себепті ең алғашқы сатыларында құрылатын бағдарламалық қамсыздандыру қандай сипаттамаларға ие болуын анықтау қиын. Талаптарды қалыптастыру сатысы тапсырыс берушінің алғашқы талаптарын, талаптарды құрылымдауды және бәлкім, прототипті салуды талдауды қамтиды.

Талаптарды құру циклды процесс болып табылады. Егер талап қорытындысында құрылған және бекітілген болса, онда әзірлеу бағдарламалық қамсыздандырудың өмірлік циклының келесі фазасы - жобалауға өтеді. Әйтпесе, алғашқы талаптарға талдау қайта өткізіледі. Әрбір циклде бағдарламалық қамсыздандырудың сипаттамасындағы семантикалық қателер анықталып, жойылады. Бұл ретте талаптарды құру процесінің аяқталу сәтін анықтау мүмкін емес. Ол аяқталуы үшін қадамдардың бірінде саналы түрде жеткіліктілік принципін басшылыққа ала отырып, құрылған талаптардың семантикалық қателері жоқ және бағдарламалық қамсыздандыру нені сипаттау керектігі жайлы шешім қабылдайды.

3.2. ТАПСЫРЫС БЕРУШІНІҢ АЛҒАШҚЫ ТАЛАПТАРЫНА ТАЛДАУ ЖӘНЕ ҚҰРЫЛЫМДАУ

Тапсырыс берушінің алғашқы талаптары бағдарламалық қамсыздандырудың жалпыландырылған сипаттамасынан тұрады, ол жеке функциялардың сипаттамасы егжей-тегжейлі болмайтындай, не болмаса болмауы. Алғашқы талаптарда талдау тапсырыс берушімен бірлесе жүргізілуі тиіс. ТАТ талдау кезінде талап етілген бағдарламалық қамсыздандыруды құру мүмкін бе, және аталған мерзімдер қаншалықты мүмкін, сондай-ақ сипаттамада қандай бөлшектер жетпейді және нені анықтап алу қажеттілігін анықтап алу керек. Сонымен қатар, жеткілікті емес ақпараттарды алу мерзімдері мен тәсілдерін, тестілеу тәсілдерін және дайын БӨ қабылдау шарттарын анықтау қажет.

Талаптарды құрылымдау тиісті түрде өтеді. Алдымен

бағдарламалық өнімнің жалпы тағайындалу мақсаты, БҚ іске асырылатын аппараттық құралдардың ерекшеліктері көрсетіледі, бағдарламалық өнімді пайдалану ерекшеліктері анықталады, бағдарламалық қамсыздандырудың негізгі компоненттерін бөледі.

Бұдан әрі жалпыланған талаптардың құрылымына егжей-тегжейлі орындауды жүргізеді: негізгі компоненттердің өзара байланысының сипаттамасын анықтайды және әрбір құрылымдық элементтің егжей-тегжейлі сипаттамасын құрайды. Әрбір құрылымдық элемент үшін оның функцияларына бөлшектеуді жүзеге асырады және барлық құрылымдық элементтердің әрбір функциясына жалпы сипаттама құрады. Бұл ретте барлық талаптарды олардың аталған мерзімде орындалу мүмкіндіктері жағынан талдау жасайды. БӨ талаптарының құрылымына егжей-тегжейлі болуы құрылымдық элементтердің өзара байланысы жайлы толық түсініктеме мен оның барлық функциялары туралы жалпы түсінік беруі тиіс.

Бағдарламалық қамсыздандыруға қойылатын талаптарды құрылымдау әрбір құрылымдық элементтің жеке функцияларына сипаттама құрумен аяқталады. Әрбір функция үшін кіру деректері көрсетілуі керек, кіріс деректеріне жасалатын әрекеттер егжей-тегжейлі сипатталуы керек, шығыс деректері анықталып, сондай-ақ тестілеу тәсілдері сипатталуы тиіс.

Әрбір талаптың егжей-тегжейлі тереңдігін жоба жетекшісі айқындайды. Ол жеке функциялардың сипаттамасы барлық қажетті мәліметтерден тұрып, әзірлеушілерге бағдарламалық өнім деген не және ол не істей алатындығы туралы анық түсінік береді.

Сипаттаманы техникалық әдебиеттерде қабылданған терминологияны қолдана отырып құруға ұсыныс беріледі.

3.3. ТАҚЫРЫПТЫҚ САЛАНЫ МОДЕЛЬДЕУ

Тақырыптық сала моделі деп зерделенетін тақырыптық саланың құрылымын немесе жұмыс істеуіне бастама болатын және негізгі талапқа - осы салада барабар болуға жауап беретін кейбір жүйе түсініледі. Модель бағдарламалық қамсыздандырудың жұмыс істеуінің барлық аспектілерін беруі тиіс және бағдарламалық қамсыздандыруды өмірлік циклының барлық сатыларында қажет. Бірақ, тақырыптық саланың моделінің басты рөлін бағдарламалық өнімді оны құрған кезде қойылатын талаптарын қалыптастыру сатысында алады.

Тақырыптық саланы алдын-ала болжау жобалық жұмыстарды

жүргізу уақыты мен мерзімдерін қысқартуға және едәуір тиімді және сапалы өнім алуға мүмкіндік береді. Тақырыптық саланы модельдеуді жүргізбей стратегиялық мәселелерді шешуде көп қате жіберу мүмкіндігі көп, ол өнімді келесі қайта жобалауға шығындарды көбейтуге және экономикалық ысырапқа әкеп соғады.

Тақырыптық саланың моделіне келесі талаптар қойылады:

- тақырыптық саланың құрылымын біржақты сипаттауды қамтамасыз ететін нысандандырылуы;
- тапсырыс берушілер мен әзірлеушілер үшін модельді графикалық бейнелеу құралдарын қолдану негізінде түсініктілігі;
- тақырыптық сала моделін нақты іске асыру құралдарының болуын білдіретін іске асырушылық;
- тақырыптық сала моделінің іске асырылу тиімділігін белгілі бір әдістер мен есептелетін көрсеткіштердің негізінде мүмкіндігін қамтамасыз ету.

Аталған талаптарды іске асыру үшін модельдердің жүйесін құру қажет:

- тақырыптық саласының ақпараттық нысандары мен материалдық процестерінде өзара байанысты құрамын көрсететін нысанды модельдің;
- процестерде нысандардың түрленуі бойынша функциялардың (әрекеттердің) өзара байланысын білдіретін функционалды модельдің;
- техникалық құралдар кешенінің орналасу типологиясы мен коммуникация тәсілдерін сипаттайтын техникалық модельдің.

Мүмкін, процестерді орындауға әсер ететін оқиғалар мен бизнес-ережелерді білдіретін басқару моделі, сондай-ақ ұйымдастыру бірліктерінің өзара байланысын білдіретін ұйымдастыру бірлігі қажет болады.

Тақырыптық сала моделінің барабарлылық моделінің басты белгісі әзірленетін БӨ функционалды толықтығында болады.

Модельдеумен жобалық шешімдерді ұсыну тілін таңдау проблемасы тікелей байланысқан.

Модельдеу тілі - бұл жобаларды сипаттау үшін қолданылатын графикалық нотация.

Нотация модельде қолданылатын графикалық нысандардың жиынтығын білдіреді және модельдеу тілінің синтаксисі болып табылады.

Модельдеу тілі бір жағынан жобалаушының шешімін пайдаланушыға түсінікті ету керек болса, басқа жағынан - жобалаушыларға тұтастай жүйені құрайтын бағдарламалық кешендер

түрінде іске асырылатын жобалық шешімдерді бір мағынада шешу құралдарын білдіреді.

Әдетте, модельдерді үш деңгейде құрады:

1) сыртқы деңгейде (талаптарды анықтау);
2) тұжырымдамалық деңгейде (талаптардың сипаттізімі);
3) ішкі деңгейде (талаптарды іске асыру).

Осылай, *сыртқы деңгейде* бағдарламалық қамсыздандырудың негізгі компоненттерінің құрамы анықталады: нысандардың, функциялардың, оқиғалардың, ұйымдастыру бірліктерінің, техникалық құралдардың.

Тұжырымдамалық деңгейде жүйе компоненттерінің өзара байланысының сипаты анықталады.

Ішкі деңгейде модельдің көмегімен «Қандай бағдарламалық-техникалық құралдардың көмегімен жүйеге қойылатын талаптар іске асырылады?» деген сауалға жауап беріледі.

Үлкен және күрделі бағдарламалармен жұмыс жасағанда проблема басты проблема болып табылады. Әзірлеушілердің ешбірі адам мүмкіндіктерінің шегінен шығуға қауқарлы емес және бүкіл жүйені жалпы түсінуге қауқарлы емес. Осы проблеманы шешуге жалғыз тиімді тәсіл ірі бөлшектердің бірқатар санынан күрделі жүйені құруда болады, олардың әрқайсысы өз кезегінде, кіші өлшемді бөлшектерден құрылады. Ең үлкен бөлшектер қабылдау үшін және түсіну үшін жеңіл болғанша қолжетімді. Бұл күрделі бағдарламалық жүйені кішкентай қосалқы жүйелерге бөлуге (декомпозитрлеу) болады, олардың әрқайсысын бір-біріне тәуелсіз қарастыруға болады.

3.4. ТАҚЫРЫПТЫҚ САЛАҒА ТЕКСЕРУ ЖҮРГІЗУ ӘДІСТЕРІ

Тақырыптық салаға тексеру жүргізу жөніндегі жұмыстарды бастамас бұрын тексеру жүргізу әдісін таңдап алу керек. Барлық әдістерді түрлі белгілері бойынша топтарға біріктіруге болады.

1. Тексеру мақсаттары бойынша:

- міндеттердің кешені үшін немесе жеке міндеттегі жобаны әзірлеу үшін қолданылатын тексеруді жергілікті жүргізудің ұйымдастыру әдісі;
- жалпы бағдарламалық қамсыздандыру жобасы үшін бүкіл нысанды әзірлеу мақсатымен қолданылатын нысанды жүйелі тексеру әдісі.

2. Тексеру жүргізетін орындаушылардың саны бойынша:

- бір жобалаушы жүзеге асыратын жеке тексеру;
- тақырыптық саланың барлық аспектілерін зерделейтін бригадалар-

орындаушылар қатары мен үйлестіруші бір бригаданың бөлінуімен бригадалық тексеру.

3. Тақырыптық саланы қамту дәрежесі бойынша:

- өндірістік немесе экономикалық жүйенің барлық бөлімшелерін қамтитын тұтас тексеру әдісі;
- құрылымы жағынан типтік нысандар болғанда қолданылатын таңдаулы тексеру.

Сонымен бірге жұмыстарды жүйелі және параллель жүргізу әдістері бар:

- жұмыстарды жүйелі жүргізу әдісі, бұл ретте жобалаушылар алдымен тақырыптық сала туралы деректерді жинайды, ал содан кейін оларды зерделейді (мұндай текті жұмыстарды орындауда тәжірибесі болмаған жағдайда жиі қолданады);
- жұмыстарды параллель орындау әдісі, мұнда алынған тексеру материалдарын жинаумен қатар зерделеу болады да, бұл жоба алдындағы сатыларға жұмсалатын уақытты біраз қысқартады және алынатын нәтижелердің сапасын арттырады.

Тақырыптық саланы тексеру және материалдарды жинау жөніндегі жұмыстарды орындауды әдістердің алдын-ала таңдалу негізінде өткізуге болады, олардың жиынтығын келесі топтарға ажыратуға болады:

- жинау әдістері, жобалаушы-орындаушылардың күшімен орындалады, олар әңгімелесу мен сауалнама өткізу әдістерін, тексеру материалдарын талдау, жеке қадағалау әдістерін (мысалы, жұмыс күнін суретке түсіру мен бір не басқа жұмысты орындау кезінде маманның жұмыс уақытына хронометраж жасау) қамтиды;
- жинау әдістері, тақырыптық саланың мамандары орындайды, олар жүзеге асыратын жұмыстарына күнделік-дәптерді толтыруды, не болмаса жұмыс орнына құжаттық түгендеу жүргізуді, не болмаса жұмыс күнін өзі суретке түсіру әдісін қолдану арқылы ұсынылады, олар операциялардың құрамы мен бұл ретте алынатын құжаттарды анықтауға мүмкіндік береді;
- басшылармен әңгімелесу және кеңес алу әдісі, ол көбінесе кәсіпорындардың және бөлімшелердің басшыларымен, болашақ пайдаланушылар тобымен қарапайым әңгімелесу түрінде жиі өткізіледі, не болмаса проблемаларды анықтауға қатысты өзекті сипаты бар мәселелер бойынша мамандармен кеңес алу түрінде өткізіледі;
- орындаушыларға жұмыс орнында сауал қою тәсілі, ол мамандардан тікелей мәлімет жинау процесінде қолданылады. Әңгімелесу өткізілетін қызметкерлердің тізімін алдын-ала құрады да, нысанның

қызметіндегі жұмысының рөлі мен тағайындалу мақсаты, оларды орындау тәртібі туралы мәселелердің тізбесін әзірлейді;

- операцияларды талдау әдісі, қарастырылып отырған іс процесін, жұмыстарды құрамдас бөлшектерге, міндеттерге, есептеулерге, операцияларға, элементтерге бөлу болып табылады. Бұдан кейін әрбір бөлігі жеке талданады да, жеке операциялардың бірнеше рет қайталануы анықталады, бір операцияға бірнеше рет жүгіну, олардың бір-біріне тәуелділігі анықталады.

3.5. ТАПСЫРЫС БЕРУШІНІҢ ТАЛАПТАРЫ БОЙЫНША СИПАТТАМАЛАРДЫ ҚҰРУ

Кез келген бағдарламалық қамсыздандыруды әзірлеу болашақ бағдарламалық өнімге қойылатын талаптарды талдаудан басталады. Талдау жүргізу нәтижесінде әзірленетін бағдарламалық қамсыздандырудың сипаттамасы алынады: шешілетін міндеттердің декомпозициясы мен мазмұндық құрылымы орындалады, олардың өзара байланысы анықталып, пайдалану шектерін анықтайды.

Сипаттамалар деп әзірленетін бағдарламалық өнімнің функцияларын және шектеулерін нақты дәл сипаттауды айтады.

Функционалдық сипаттама бағдарламалық қамсыздандыруды әзірлеуде - бұл жүйенің талап етілген сипаттамаларын (функционалдылығын) сипаттайтын құжат.

Пайдалану сипаттама - бұл бағдарламалық қамсыздандыруды пайдалану ережелерінің сипаттамасы. Пайдалану сипаттама техникалық құралдарға, сенімділікке, қауіпсіздікке және т.б. қойылатын талаптарды анықтайды. Сипаттамалардың жиынтығы жобаланатын бағдарламалық қамсыздандырудың жалпы логикалық моделін білдіреді.

Жалпы сипаттамаларды анықтау процесінде тақырыптық саланың жалпы моделін құрады, ол нақты әлемнің бір бөлігі ретінде, ол әзірленетін бағдарламалық қамсыздандырудың қалай да болса бір тәсілмен өзара байланысын қамтамасыз етеді және оның негізгі функцияларын нақтылайды.

Функционалдық сипаттамалар толық және нақты болуы тиіс. Талаптардың толық болуы сипаттізімдердің барлық маңызды ақпараттан тұратындығын, маңызды ештеңе назардан тыс қалмауын, және маңызды емес ақпараттардың болмауын, мысалға, іске асыру бөлшектерін, олар әзірлеушілерге біршама ұтымды шешім қабылдауға кедергі келтірмейтіндей болу қажеттілігін білдіреді. Талаптардың дәлдігі сипаттамаларды тапсырыс беруші де, әзірлеушілер де бірдей

түсіну керектігін білдіреді.

Функционалды сипаттамалар төрт бөліктен тұруы тиіс.

1. *Сыртқы ақпараттық ортаның сипаттамасы,* онымен әзірленетін бағдарламалық қамсыздандыру өзара жұмыс істейтін болады. Қолданылатын барлық енгізу және шығару каналдары мен әзірленетін БҚ қолданылатын барлық ақпараттық нысандар, сондай-ақ осы ақпараттық нысандардың арасындағы маңызды байланыстар анықталуы керек.

2. *Бағдарламалық қамсыздандыру функцияларын анықтау*, осы ақпараттық жүйенің көптеген жағдайында анықталған. Барлық анықталатын функциялардың шартты атаулары енгізіледі, олардың кіріс деректері мен орындалу нәтижелері сипатталып тізіледі, деректердің түрлері мен осы деректер мен нәтижелерді қанағаттандыру қажет барлық шектеу тапсырмалары көрсетіледі. Осы функциялардың әрқайсысының мазмұны анықталады.

3. *Айрықша жағдайлардың сипаттамасы,* егер мұндай жағдайлар бағдарламаны орындау кезінде пайда болуы мүмкін болса және осы жағдайларға әсері тиісті бағдарламаны қамтамасыз ететін болса көрініс табады. Бағдарламалық қамсыздандыруды өзінің осы немесе басқа функциясын қалыпты орындай алмайтын барлық маңызды жағдайлар аталуы тиіс. Әрбір мұндай жағдай үшін бағдарламаның реакциясы анықталуы керек.

Әзірленетін бағдарламалық қамсыздандырудың барлық функционалды сипаттамалары өңделетін деректердің құрамы мен функцияларының тізбесін сипаттайды. Олар әзірлеушілердің талаптарына талдау жасау мен сипаттамаларын анықтау процесінде қолданатын басымдықтарға (акценттерге) ғана жүйемен ажыратылады.

Орындаушы мен тапсырыс берушінің талаптарын бекіту талаптардың сипаттамаларының барлық тармақтары бойынша олар келісімге қол жеткізген сәтте анықталады. Тапсырыс беруші бағдарламалық қамсыздандыруға қойылатын кейбір талаптардың тексерілуін көрсететін прототипті немесе өзге мысалды ұсынуын талап етуі мүмкін, БӨ қойылатын жеке талаптардың параметрлерін бірнеше рет өзгертуін немесе көптеген тексеру үлгілерін анықтауын талап етуі мүмкін. Бұл жағдайда тапсырыс берушінің мұндай тілектерін талаптардың сипаттамасына қосымшаларға рәсімдеу қажет.

Сипаттамаларды құрған кезде бір мәнді түсіндірмейтін сөздерді және сөз тіркестерін қолданбаған дұрыс. Әзірленетін бағдарламалық қамсыздандырудың дәл сипаттамаларын осы бағдарламалық қамсыздандырудың кейбір модельдерін әзірлеп барып қана анықтауға болады.

3.6. ПРОТОТИПТІ САЛУ

Прототипті салу (прототиптеу) кейбір әзірлеушілердің пікірі бойынша бағдарламалық қамсыздандыруды әзірлеудің ең маңызды сатыларының бірі болып табылады.

Алдымен прототиптер не үшін қолданылатындығын анықтап алайық.

Біріншіден, пайдаланушыға жүйеден не күтетіндігін, оған қойылатын талаптарын қалыптастыру қиын болады. Бұл жағдайда пайдаланушы интерфейсінің прототипі (User Interface, UI), әңгімелесу нәтижелері бойынша шұғыл құрылған, ол әзірлеушінің жүйенің тиісті бөлігін қалай көріп, жүйелі түрде қалай іске асыратындығын көруге мүмкіндік береді. Прототиптерді қолданған кезде кез келген нәтиже маңызды. Егер бағларламалаушы тапсырыс берушінің талаптарын дұрыс түсінген болса, онда әрине, бағдарламалық жүйені әзірлеу дұрыс бағытта жылжитын болады. Егер пайдаланушы өзіне ұсынылған жүйені іске асыру нұсқаларына көңілі толмаған болса, пайдасы бағдарламалаушы түсінбеген міндеттерді көрсетуінде болады.

Екіншіден, прототиптеу альтернативті тұжырымдамалық шешімдердің бірін таңдауға мүмкіндік береді. Кез келген техникалық міндетті әртүрлі тәсілдермен шешуге болады. Бұл бағдарламалық жүйеге қойылатын талаптарды қалыптастыру міндеті сияқты, оның пайдаланушы интерфейсін іске асыруға да қатысты.

Үшіншіден, функионалды және функционалды емес талаптардың үйлесімі оларды іске асыру мүмкіндігі қауіп туғызатындай болады. Негізінде, мұндай тәуекел оны іске асыру ортасына белгілі шектеулер болғанда жылдам әрекет етуге қойылатын талаптармен байланысты. Бұл жағдайда жүйенің тиісті бөлігін іске асыратын, деректердің ағынын имитациялайтын, оның кірісіне келіп түсетін және осы деректердің өңделуін көрсететін прототиптер құрылады (пайдаланушы интерфейсімен міндетті түрде байланысты емес).

Прототиптеу - бұл жалпы жүйенің жұмысына талдау үшін негізгі функционалдылықты жылдам «алғашқы» іске асырылуын айтуға болады.

Прототиптеу сатысында құрылатын бағдарламалық жүйе ұтымды жұмыс істемеуі мүмкін, қателермен және толық болмауы мүмкін. Бұдан басқа, прототиптеу әзірленетін жүйе сияқты сол технологиялармен (мысалы, бағдарламалау тілі) жүзеге асырылуы міндетті емес.

Прототиптер көлденең және тік, бір реттік және эволюциялық, қағаз және электронды болуы мүмкін.

Көлденең (мінез-құлық) прототипі деректердің құрылымы мен өңдеу логикасын қатыстырмай, қосымшаларды пайдаланушы интерфейсін модельдейді. Көлденең прототиптерді көпбалама түрде іске асырылатын анық емес талаптарды анықтап алу қажет болғанда қолданған дұрыс. Көлденең прототиптер үшін жүйенің қорытынды нұсқасы әзірленетін тура сол бағдарламалық іске асыру ортасын қолдану міндетті емес. Егер әзірленген интерфейсте деректер қоры қолданылатын болса, онда олар бағдарлама кодында ұқсастырылып беріледі. Бұл ретте экранда берілетін мәтіндер проблемалық саланың нақты сипаттамасын көрсетеді, әйтпесе, пайдаланушы үшін жиналу қиын болады.

Прототипті құрған кезде есептеулер мен сұратулар нәтижесі, мысалы, сыртқы жүйелерге сұрату ұқсастырылып беріледі. Пайдаланушыға түсінікті болу үшін жүйе оның әрекеттеріне қалай әрекет ететіндігін пайдалану нұсқаларын қолдану процесіндегі экрандағы орын алмасуларға жауап беретін кодтың бір бөлігін көрсету дұрыс болар еді. Көлденең прототипті құрмас бұрын қандай негізгі экрандардан болатындығын, қандай терезелер ашылатындығын, олар арасында қандай өтулер болатындығын анықтап алған дұрыс. Бұл үшін диаграмма моделін қолданған жөн, мұндай әртүрлі экрандар (терезелер) арқылы күйі салыстырылады, ал актив басқару элементтері кейбір интерфейс элементтерін жапса, басқалары - өткелдерді шақыртады.

Тік (құрылымдық) прототип пайдаланушының интерфейсін жобалауға қаншалықты бағытталса, іске асырудың барлық деңгейлерін қозғайтын жүйенің тік «кесінділерін» іске асыруға соншалықты бағытталған. Мұндай прототипті құрған кезде тұтас желіні құрған кезде қолданылатын тілдер мен іске асыру ортасын қолдану ұсынылады. Мұндай тектес прототиптер қолданылуды талдау, сәулеттік тұжырымдаманы тексеру үшін қолданылады.

Бір реттік, немесе *зерттеу прототипі* әзірленетін бағдарламалық құралдың макетін, оның осы немесе басқа аспектілері мен компоненттерін жылдам алу қажет болғанда құрылады. Зерттеу прототиптерін құру мақсаты RAD технологиялары (rapid application development) - қосымшаларды жылдам әзірлеу болады. Бір реттік прототип жылдам құрылуы тиіс. Оны әзірлеген кезде кодты қайта пайдалану мәселелеріне, сапасына, жылдам әрекет етуіне, технологиялылығына және т.с.с. назар аудару қажет. Нәтижесінде «шикі» код алынады, ол қаншама ақауларынан тұруы мүмкін. Мұндай тектес прототиптерді іске асыратын кодтың үзіндісі тұтас жүйенің бөлігі

болмауы үшін шаралар қолдану қажет.

Эволюциялық прототип жүйенің өзінің салдарында алғаш жақындаған жүйе ретінде құрылады. Эволюциялық прототиптің бағдарламалық коды жүйелі түрде тұтас қосымшалардың кодына өсуі керек. Сол себепті бұл прототиптердің түрі бір реттік прототиптерді құру кезінде бас тартуды қажет етеді: мұқият әзірлеу, технологиялық әдістер мен тәсілдерді қолдану, нәтижелерді тестілеу және т.с.с.

Прототипті құру кезеңі ұзақ болмауы керек. Ұтымды прототиптер идеяны мүдделі тұлғаға дейін жеткізуге арналған. Идея жеткізілгеннен кейін прототип қабыл алмауы мүмкін.

Прототипті құру процесі келесі қадамдардан тұрады:

1) бастапқы талаптарды анықтау;
2) жүйенің пайдаланушы интерфейсінен тұратын прототиптің алғашқы нұсқасын әзірлеу;
3) тапсырыс беруші мен соңғы пайдаланушының прототипті зерделеу сатысы. Қажетті өзгертулер мен толықтырулар туралы кері байланысты алу;
4) алынған ескертулер мен ұсыныстарды ескере отырып, прототипті өңдеу.

Прототип болашақ пайдаланушылардан кері байланысты алуға мүмкіндік береді, ал бұл ең қажет жағдайларда өте қажет: жоба басында жобалаудың қателерін шығынсыз өзгерту мүмкіндіктері бар болған кезде қажет.

Прототиптің қасиеттерін зерделеу тапсырыс берушінің талаптарын оны бекіткенге дейін анықтау немесе тексеру мақсатымен жүзеге асырылады. Бағдарламалық қамсыздандыруға тапсырыс берушілер мен пайдаланушыларға әдетте әзірленетін бағдарламалық өнімге қойылатын талаптарды қалыптастыру қиынға түседі. Бағдарламалық қамсыздандыру өзге бағдарламалармен қалай өзара байланысады және пайдаланушылардың орындайтын қандай операцияларын автоматтандыру қажеттілігін алдын-ала көру қиын. Талаптарға мұқият талдау бағдарламалық өнімнің не істеу керектігін анық түсінуге көмектеседі.

Прототип жүйеге салынған тұжырымдамаларды көрсету, талаптардың нұсқаларын тексеру, сондай-ақ әзірлеу барысында сияқты, БӨ пайдалану кезінде пайда болуы мүмкін проблемаларды іздеу және осы проблемаларды шешудің мүмкін нұсқаларын іздеу үшін қолданылатын бағдарламалық қамсыздандырудың бастапқы нұсқасы болып табылады.

Бағдарламалық қамсыздандыруға қойылатын талаптарды құрумен жұмыс істегенде прототип бірқатар артықшылықтарын қамтамасыз

етеді. Пайдаланушылар прототиппен тәжірибе жасап көре алады, ол БӨ қалай жұмыс істейтіндігін тексеруге мүмкіндік береді. Олар өнімнің әлсіз және күшті жақтарын анықтай алады, оның нәтижесінде жаңа талаптарды құруға болады.

Прототип бұрын қабылданған талаптарда кеткен кемшіліктер мен қателерді анықтауға мүмкіндік береді. Мысалы, бағдарламалық қамсыздандырудың талаптарында анықталған кейбір функцияларды пайдаланушылар басында маңызды және қажет деп санауы мүмкін, алайда осы функцияларды қолдану процесінде олар туралы пікірі өзгереді. Нәтижесінде БҚ қойылатын талаптары өзгереді, БӨ функцияларын пайдаланушы жаңа қырынан түсіне бастайды.

Прототиптеуді жоспарлау сатысында тәуекелдерді талдау кезінде қолдануға болады. Бағдарламалық өнімді әзірлеу кезіндегі жаңа қауіп талаптардағы қателер мен жіберілген кемшіліктер болып табылады. Талаптардағы қателерді жоюға жұмсалатын шығындар әзірлеу процесінің кейінірек сатыларында өте жоғары болуы мүмкін.

Прототиптеу жүйені әзірлеудің жалпы құнын да азайтады. Осы себептеріне қарай оларды талаптарды әзірлеу процесінде жиі қолданады.

3.7. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ЖОБАЛАУ ТЕХНОЛОГИЯСЫ

Жобалау сатысының міндеті әзірленетін бағдарламалық қамсыздандырудың толық сипаттамаларын анықтау болып табылады. Бағдарламалық қамсыздандыруды жобалау процесі әдетте мыналарды қамтиды:

1) жалпы құрылымды жобалау - негізгі бөліктерін (компоненттерін) және олардың басқару мен деректермен өзара байланыстарын анықтау;
2) компоненттердің декомпозициясы және блок-иерархиялық тәсіл ұсыныстарына сәйкес құрылымдық иерархияның құрылуы;
3) компоненттерді жобалау.

Жобалау нәтижесі әзірленетін бағдарламалық қамсыздандырудың оның барлық деңгейлерінің компоненттерінің сипаттамасымен бірге егжей-тегжейлі моделі болып табылады. Модель типі таңдап алынған немесе белгіленген тәсілге (құрылымды, нысанға бағытталған) және нақты жобалау технологиясына байланысты болады. Алайда, жобалау процесінің кез келгенінде бағдарламаны (қосалқы бағдарламаны)

жобалау мен олардың арасындағы өзара байланысты сияқты, осы бағдарламалар немесе қосалқы бағдарламалар өзара байланысатын деректердің жобалауын қамтиды.

Сонымен бірге, жобалаудың екі қырын ажырату қабылданған:

1) логикалық жобалау, олар болашақ бағдарламалық өнімнің жұмыс істеу ортасын қамтитын техникалық және бағдарламалық құралдарға тікелей байланысты емес жобалау операцияларын қамтиды;
2) физикалық жобалау, нақты техникалық және бағдарламалық жұмыс істеу құралдарына байланысады.

Жобалау әдістемесі, технологиясы мен аспаптық құралдары кез келген бағдарламалық өнім жобасының негізін құрайды. Әдістеме нақты технологиялар мен оларды қолдайтын стандарттар, өмірлік циклдағы процестердің орындалуын қамтамасыз ететін әдістемелер мен аспаптық құралдар арқылы іске асады.

Жобалау технологиясы жобалаудың технологиялық операцияларының және оларды жобаны әзірлеуге әкелетін өзара байланысының жиынтығы ретінде айқындалады. Жобалау технологиясын үш құрамдас бөлшектердің жиынтығы ретінде көрсетуге болады:

1) жобалаудың технологиялық операцияларының бірізділігін анықтайтын қадамдық процедуралар;
2) технологиялық операцияларды орындау нәтижелерін бағалау үшін қолданылатын белгілер мен ережелер;
3) жобаланатын бағдарламалық жүйенің сипаттамасы үшін қолданылатын нұсқаулар (графикалық және мәтіндік құралдар).

Нақты ұйымда және нақты жобада бағдарламалық өнімді жобалаудың, әзірлеудің және қолдаудың кез келген технологиясының нақты қолданылуы әзірлеушілер ұстануы тиіс бірқатар стандарттарды (ережелерді, келісімдерді) жасаусыз мүмкін емес. Мұндай стандарттарға жобалау стандарты, жобалық құжаттаманы рәсімдеу стандарты, пайдаланушы интерфейсінің стандарты жатады.

Таңдап алынатын жобалау технологиясына қойылатын негізгі талаптар:

- бұл технологияның көмегімен құрылған жоба тапсырыс берушінің талаптарына жауап беруі тиіс;
- таңдап алынған технология жобаның ӨЦ барлық сатыларын максималды түрде көрсетуі тиіс;
- таңдап алынатын технология жобаны жобалауға және қолдауға жұмсалатын құн шығындары мен минималды еңбек шығындарын қамтамасыз етуі тиіс;

- технология жобаны жобалау мен қолдау арасындағы байланыс негізі болуы тиіс;
- технология жобалаушының еңбек өнімділігінің артуына ықпал етуі тиіс;
- технология жобаны жобалау және пайдалану процесінің сенімділігін қамтамасыз етуі тиіс;
- технология жобалық құжаттаманы қарапайым жүргізуге ықпал етуі тиіс.

Бағдарламалық қамсыздандыруды жобалау технологиясының мәнін, негізгі айрықша технологиялық ерекшеліктерін анықтайтын әдістеме құрауы тиіс. Әдістеме нақты технологиялар мен оларды қолдайтын стандарттар, өмірлік циклдың процестерінің орындалуын қамтамасыз ететін әдістемелер мен аспаптық құралдары арқылы іске асырылады. Жобалау әдістемесі кейбір тұжырымдамалардың, жобалау қағидаттарының, жобалауды іске асыратын әдістердің жиынтығының болуын болжайды, олар өз кезегінде кейбір жобалау құралдарымен сүйемелденуі керек.

Бағдарламалық қамсыздандыруды жобалау әдістерін жіктеуге болады.

Автоматтандырылу дәрежесіне қарай жобалау әдістері мына әдістерге бөлінеді:

- қолмен жобалау, мұнда БӨ компоненттерін жобалау аспаптық бағдарламалық құралдарды қолданусыз, ал бағдарламалау - алгоритм тілдерінде қолданусыз жүзеге асырылады;
- компьютерлік жобалау, ол арнайы аспаптық бағдарламалық құралдарды қолдану негізінде жобалық шешімдердің конфигурациясын (бапталуын) немесе қосылуын жүргізеді.

Типтік жобалық шешімдерді қолдану дәрежесіне қарай келесі жобалар әдістерін ажыратады:

- түпнұсқалық ерекше (жеке) жобалау, мұнда жобалық шешімдер бағдарламалық қамсыздандыруға қойылатын талаптарға сәйкес «нөлден» бастап әзірленеді. Түпнұсқалық жобалау бағдарламалық қамсыздандыруды барлық жобалық жұмыстардың түрлері әрбір жоба нысандары үшін жеке оның барлық ерекшеліктерін барынша көрсете алатындай құруға бағдарлануы керектігімен сипатталады;
- типтік жобалау, БӨ конфигурациясын дайын типтік жобалық шешімдерден (бағдарламалық модульдерден) болжайды. Типтік жобалау жеке жобаларды әзірлеу кезінде алынған тәжірибе негізінде орындалады, типтік жобалар кейбір ұйымдасқан-экономикалық жүйелердің топтары немесе нақты жағдайда жұмыс түрлері үшін кқптеген өзіне тән ерекшеліктерімен байланысқан және

орындалатын жұмыстар мен әзірленетін жобалық құжаттаманы, басқару функцияларын қамту дәрежесіне қарай ажыратылады.

БАҚЫЛАУ СҰРАҚТАРЫ ЖӘНЕ ТАПСЫРМАЛАРЫ

1. Бағдарламалық қамсыздандыруға қойылатын талаптарды басқарудың мақсаты неде?
2. Сипаттама деген нені білдіреді?
3. Бағдарламалық қамсыздандыруға қойылатын талаптарды басқару деген қандай?
4. Функционалды сипаттама деген қандай бөліктерден тұрады?
5. Тақырыптық сала моделіне қандай талаптар қойылады?
6. Прототиптеу деп нені атайды?
7. Прототипті құру процесі қандай қадамдардан тұрады?
8. Прототип бағдарламалық қамсыздандыруға қойылатын талаптармен жұмыс істегенде басымдылықты қалай қамтамасыз етеді?
9. Бағдарламалық қамсыздандырудың жобалау процесі қандай қадамдардан тұрады?
10. Тақырыптық саланы тексеру әдістері қалай сыныпталады?
11. Тақырыптық сала моделіне қандай талаптар қойылады?
12. Логикалық және физикалық жобалаудың мәні неде?
13. БӨ жобалаудың қандай әдістері бар?

4-бөлім

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ЖОБАЛАУҒА ЖӘНЕ ӘЗІРЛЕУГЕ ҚОЙЫЛАТЫН ҚҰРЫЛЫМДЫҚ ТӘСІЛ

4.1. ҚҰРЫЛЫМДЫҚ ТӘСІЛДІҢ МӘНІ

Бағдарламалық қамсыздандыруды әзірлеуге құрылымдық тәсілдің мәні оның автоматтандырылатын функцияларына декомпозициясы (бөлінуі): жүйе өз кезегінде міндеттерге және т.б. бөлінетін қосалқы функцияларға бөлінетін функционалдық қосалқы жүйелерге бөлінеді. Бөлу процесі нақты процедураларға дейін бөлуге дейін жалғасады. Бұл ретте автоматтандырылатын жүйе барлық құрамдас компоненттері өзара байласқан тұтас көрінісін сақтайды. Атап өтсек, жүйені кері әзірлегенде («төменнен - жоғарыға») жеке міндеттерден барлық жүйеге тұтастық жоғалады, жеке компоненттердің ақпараттық түйісу кезінде проблемалар туындайды.

Құрылымдық тәсілдің ең көп таралған әдістемелері келесі принциптерге негізделеді:

- *«бөл де билік ет» принципі* - күрделі проблемаларды түсінуге және шешуге жеңіл болу үшін көптеген кішкене міндеттерге бөлу арқылы шешу принципі;
- *иерархиялық қарапайымдандыру принципі* - проблеманың құрамдас бөліктерін әрбір деңгейде жаңа тетіктерді қоса отырып, ағаш тәрізді иерархиялық құрылымда ұйымдастыру принципі.

Құрылымдық талдауда жүйемен орындалатын және деректер арасындағы қатынаспен функцияларды бейнелеп көрсететін құралдар тобы қолданылады. Әрбір құралдар тобына белгілі бір модельдер (диаграммалар) түрлері сәйкес келеді, олардың ішінде ең көп таралғандары болып табылатындар:

- SADT әдісі (Structured Analisis and Design Technique) - құрылымдық талдау және жобалау әдісі, модельдер және тиісті диаграммалар;
- DFD әдісі (Data Flow Diagrams) - деректер ағындарының диаграммасы;

- ERD әдісі (Entity-Relationship Diagrams) - «мәні-байланыс» диаграммасы (деректер моделі).

Аталған модельдер жиынтығында бағдарламалық қамсыздандырудың қолданыста немесе жаңадан әзірленіп жатқандығына қарамастан толық сипаттамасын береді. Әрбір нақты жағдайда диаграммалардың құрамы бағдарламалық қамсыздандырудың қажетті толық сипаттамасына байланысты болады.

«Мәні-байланыс» моделі «Деректер қорын әзірлеу және әкімшілік ету» кәсіби модулінде зерделенеді және сол себепті осы оқу құралында қарастырылмайды. SADT және DFD модельдерінің құрылу ережесі бұдан әрі сипатталып беріледі.

4.2. SADT ФУНКЦИОНАЛДЫ МОДЕЛЬДЕУ ӘДІСНАМАСЫ

Функционалды диаграммалар әзірленетін бағдарламалық қамсыздандыру функцияларының өзара байланыстарын көрсетеді. Олар жүйені жобалаудың ерте сатыларында құрылады, олар жобалаушыға жобаланатын бағдарламалық жүйенің негізгі функцияларын және құрамдас бөлшектерін анықтауға, және де мүмкіндігінше маңызды қателерді анықтап, оларды жоюға көмектесу үшін керек. Функционалдық диаграммаларды құру үшін Д.Росс ұсынған SADT әдістемесін қолдану ұсынылады.

SADT әдіснамасының негізінде IDEF0 (Icam DEFinition) күрделі жүйелердің танымал сипаттамасы құрылған болатын, ол АҚШ ВВС бастамасымен жүргізілетін ICAM (өндірісті біріктірілген компьютерлендіру) бағдарламасының негізгі бөлігі болып табылады.

SADT әдіснамасын қолдану нәтижесі бір-біріне сілтемесі бар мәтіндердің үзіндісі мен глоссарий, диаграммалардан тұратын модель болып табылады.

SADT әдіснамасы мыналарды қолданылуы мүмкін:

- бағдарламалық қамсыздандырудың тақырыптық саласын модельдеу және талаптары мен функцияларын анықтау үшін;
- бағдарламалық қамсыздандыруды әзірлеу, ол осы талаптарды қанағаттандырады және осы фунцияларды іске асырады.

SADT қолданыстағы бағдарламалық өнімді талдауда қолданылуы мүмкін:

- БӨ орындалатын функцияларды талдау үшін;

- олар жүзеге асырылатын механизмдерді белгілеу үшін.

Диаграммалар - модельдің негізгі компоненттері. Жүйенің функциялары мен интерфейстер блоктар мен доғалар сияқты диаграммаларда берілген. Доғаның блокпен жалғанған орны интерфейс типін анықтайды. Басқару ақпараты блокқа жоғарыдан кіреді, ал өңделетін ақпарат өңдеуге түседі, ол блоктың сол жағынан көрсетілген, ал шығу нәтижелері - оң жағынан. Операцияны жүзеге асыратын механизм (адам немесе автоматтандырылған жүйе) төменнен блокқа кіретін доғамен беріледі (4.1-сурет).

SADT моделі блоктар түрінде берілген құрамдас бөлшектерге күрделі нысанды бөлетін ілеспе құжаттамамен диаграммалар сериясын білдіреді. SADT әдістемесінің маңызды ерекшеліктерінің бірі модельді білдіретін диаграммаларды құру жағынан барлық егжей-тегжейлі деректерін біртіндеп енгізу болып табылады. Негізгі блоктардың әрқайсысының бөлшектері басқа диаграммаларда блоктар түрінде көрсетілген. Әрбір *егжей-тегжейлі* диаграмма жалпы диаграммадан блоктардың декомпозициясы болып табылады. Декомпозицияның әрбір қадамында жалпы диаграмма толық диаграмма үшін *түпкі* деп аталады.

SADT-модельді құру қарапайым компоненттер түрінде бүкіл жүйені көрсетуден басталады - бір блок пен жүйеден тыс функциялары бар интерфейстерді көрсететін доға. Жалғыз блок блокта көрсетілген атауды біртұтас жүйені білдіретіндей жалпы болып табылады. Бұл дұрыс және интерфейсті доғалар үшін - олар жалпы жүйелердің сыртқы интерфейстерінің толық жиынтығын білдіреді.

Бұдан кейін жүйені біртұтас модуль түрінде көрсететін блок басқа диаграммада интерфейстік доғалармен жалғанған бірнеше блоктардың көмегімен егжей-тегжейіленеді. Бұл блоктар бастапқы функциялардың негізгі қосалқы функцияларын көрсетеді.

4.1-сурет. Функционалды блок және интерфейстік доғалар

Бұл декомпозиция шекаралары интерфейстік доғалармен айқындалған блок ретінде әрбірі берілетін қосалқы функциялардың толық жиынтығын анықтайды. Осы қосалқы функциялардың әрқайсысы толығырақ көрсету үшін тиісті түрде декомпозияланған. Барлық жағдайларда әрбір қосалқы функция бастапқы функцияға кіретін элементтерден тұруы мүмкін. Сонымен қатар, бұрын айтылғандай, модель қандай да бір элементті түсіре алмайды, түпкі блок пен оның интерфейстерді мәнмәтінді қамтамасыз етеді. Оларға қосуға болмайды және одан жоюға болмайды.

Блокқа кіретін және одан шығатын доғалар жоғарғы деңгейдің диаграммасында төменгі деңгейдің кіретін және одан шығатын доғалары сияқты болып табылады, себебі блок пен диаграмма жүйенің сол бір бөлігін көрсетеді.

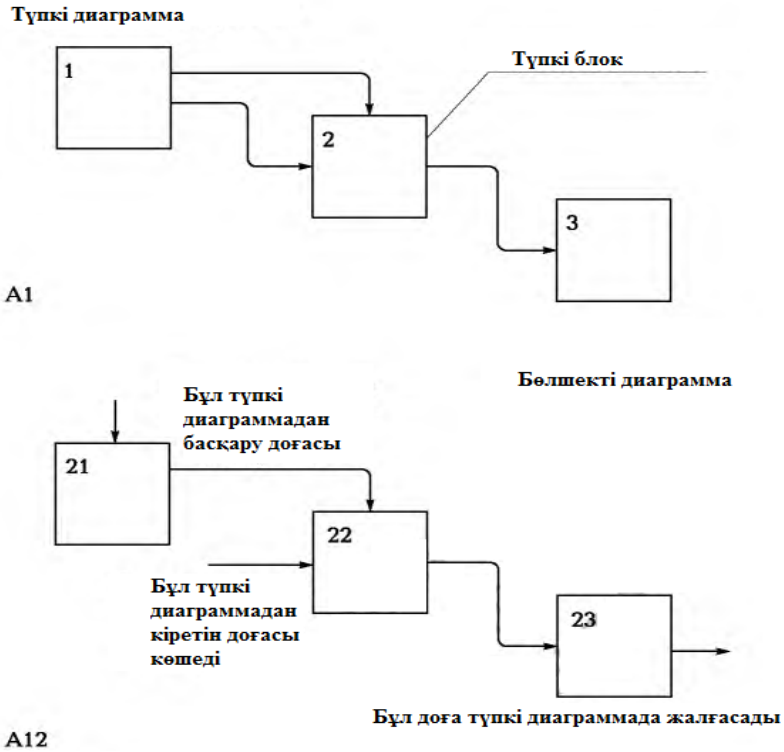
Барлық диаграммаларды бір-бірімен блоктардың иерархиялық нөмірленуімен байланыстырады: бірінші деңгей - А0, келесі - А1, А2 және т.с.с., келесі - А11, А12, А13 және т.б., мұнда алғашқы цифрлар - түпкі блоктың нөмірін, ал соңғысы - түпкі блоктың нақты қосалқы блогының нөмірін (4.2-сурет) білдіреді. Әрбір диаграмма блоктың түпкі диаграммадағы «ішкі салынуын» көрсетеді. Егжей-тегжейін функцияларды алғанда аяқтайды, олардың тағайындалуы тапсырыс берушіге де, әзірлеушіге де жақсы ұғымды. Бұл функциялар табиғи немесе жалған кодтарды қолдана отырып жақсы сипаттайды. Диаграммалардың иерархиясын құру процесінде барлық анықталатын ақпаратты жазып алып, деректердің сөздігін құрады, мұнда диаграммада көрсетілген деректердің элементтері мен құрылымын анықтайды.

Осылайша, нәтижесінде функционалдық диаграммалардың иерархиясынан, бір-біріне сілтемесі бар сөздік пен төменгі деңгей функцияларының сипаттамасынан тұратын сипаттаманы алады.

SADT-диаграммаларда жүйелілігі де, уақыты да көрсетілмегені анық. Кері байланыстар, итерациялар, жалғасатын процестер мен қатарласып келетін функциялар (уақыты жағынан) доғалардың көмегімен де көрсетілуі мүмкін. Кері байланыстар түсініктемелер, ескертулер, түзетулер және т.б. түрінде көрсетілуі мүмкін.

SADT әдістемесінің көмегімен жүйені модельдегенде маңызды сәттердің бірі функциялар арасындағы байланыс типтерін дәл келісу болып табылады. Кем дегенде байланысудың жеті типін ажыратады:

1) кездейсоқ;
2) логикалық;
3) уақытша процедуралық;



4.2-сурет. Түпкі және бөлшекті диаграммалардың интерфейстік доғаларының сәйкестігі

4) коммуникациялық;
5) кезекті;
6) функционалды.

Кездейсоқ байланыс функциялар арасындағы нақты байланыс аз немесе мүлде болмағанда пайда болады. Бұл SADT-доғаларда деректердің аттары диаграммаларда бір-бірімен аз байланысы болған жағдайларға қатысты.

Логикалық байланыстыру деректер мен функциялар олар жалпы класқа түскенде немесе элементтер жиынтығы түскенде болады, бірақ олардың арасындағы қажетті функционалдық қарым-қатынас байқалмайды.

Уақытша байланыс уақытпен байланысқан функцияларды береді, деректер бір уақытта қолданылады немесе функциялар параллель қосылады, олар кезекпен қосылмайды.

Процедуралық байланыс - функциялар бірге топтастырылған, себебі олар бір циклдың немесе процестің ішінде орындалады.

Диаграммалар к*оммуникациялық байланысты* көрсетеді, бұл кезде олар бір деректерді қолданса және/немесе шығыс деректерін шығаратын болған соң топтастырылады.

Жүйелі байланыстары бар диаграммаларда бір функцияның шығысы келесі функция үшін кіріс деректері болады. Элементтер арасындағы байланыс диаграммада біршама тығыз болады, ол байланыс деңгейлерінен жоғары болады, себебі себеп-салдарлық тәуелділік модельденеді.

Диаграмма толық *функционалды байланысты* көрсетеді, бір функцияның басқасынан толық тәуелсіздік болғанда. Жиі функционалды болып табылатын диаграмма бірізді немесе біршама әлсіз байланыс түріне жататын бөгде элементтерден тұрмайды.

Талапкерлерді қабылдау және оқуға алу процестерін автоматтандыру үшін функционалды диаграмманы құру үлгісін мысалға аламыз. 4.3-суретте көрсетілген диаграмма жоғарғы деңгей диаграммасы болып табылады. Мұнда жүйе үшін бастапқы деректер не болып табылатындығы және қандай нәтижелер күту керектігі жақсы көрінеді. Функционалды диаграммада нөлдік деңгей (4.4-сурет) түпкі блок А0 функционалдық блоктарға А1, А2, А3, А4 бөлінеді. Өз кезегінде кезекті блок А1 1-інші деңгейдің функционалды диаграммасында төрт еншілес блокқа А11, А12, А13, А14 көрсетілуі мүмкін (4.5-сурет).

SADT әдісі функциялар мен талаптарды анықтау үшін және ең түрлі жүйелерді модельдеу үшін қолданылуы мүмкін. Қолданыстағы жүйелерде SADT әдісі жүйе орындайтын функцияларды талдау үшін қолданылуы мүмкін.

4.3-сурет. Бастапқы деңгейдің функциялық диаграммасы

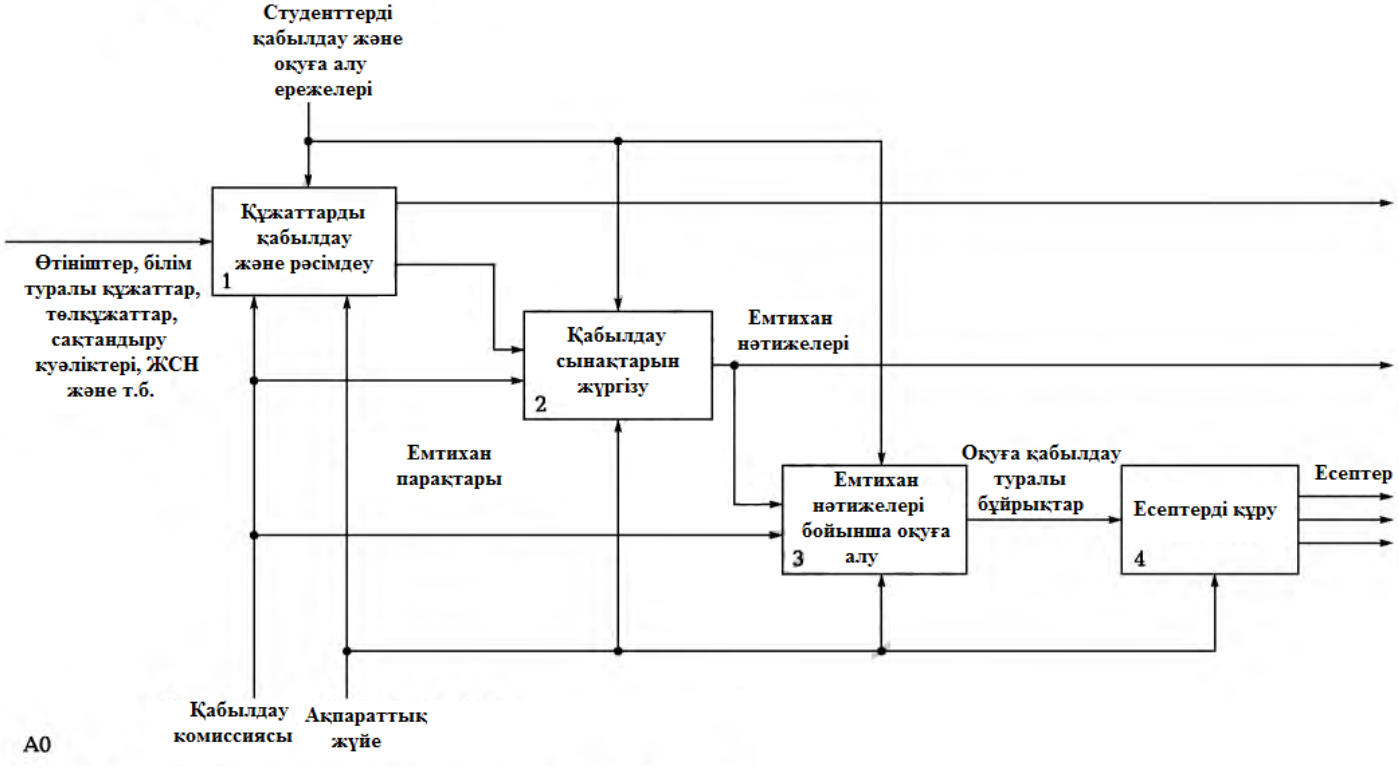
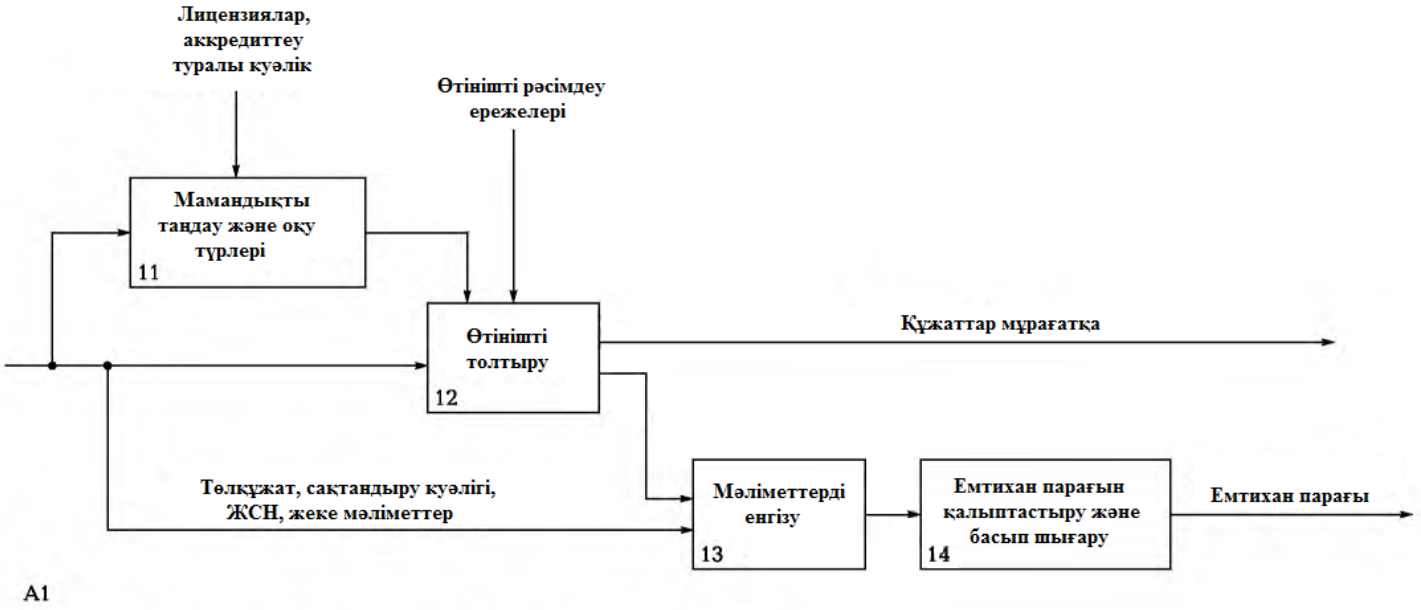


Рис. 4.4. Функциональная диаграмма нулевого уровня (более подробный вариант)



4.5-сурет. «Құжаттарды қабылдау және рәсімдеу» 1-деңгейінің функционалдық диаграммасы

4.3. DFD ДЕРЕКТЕР АҒЫНЫНЫҢ ДИАГРАММАСЫ

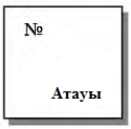
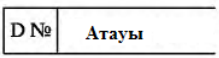
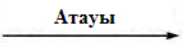
Деректер ағынының диаграммасы (DFD) жүйеге - жобаланатын немесе нақты қолданыстағы функционалдық талаптарды модельдеу құралы болып табылады.

DFD деректерді өңдеу тораптарынан, деректер тұтынушылар немесе көздеріне қолданылатын диаграммаға қатысты сыртқы және деректерді сақтау құралдарынан тұрады.

Модельдің негізінде процестің, деректерді сақтау орнының (жинағыштың), сондай-ақ деректер ағынының мәні жатыр. Ақпарат көздері (сыртқы мәні) қосалқы жүйелерге немесе процестерге ақпаратты өткізетін ақпараттық ағындарды (деректер ағынын) тудырады. Олар өз кезегінде ақпаратты түрлендіреді де, жаңа ағындарды тудырады, олар ақпаратты басқа процестерге немесе қосалқы жүйелерге, деректер жинағышына немесе сыртқы мәнге - ақпарат тұтынушыларына түрлендіреді.

Деректер ағынының диаграммасын көрсету үшін дәстүрлі түрде екі түрлі нотация қолданылады: Йордан және Гейн-Сарсон нотациялары, олар 4.1-кестеде көрсетілген.

4.1-кесте. DFD қолданылатын нотациялар

Ұғым	Йордан нотациясы	Гейн-Сарсон нотациясы
Сыртқы мәні		
Жүйе, қосалқы жүйе немесе процесс		
Деректерді жинағыш		
Ағын		

Сыртқы мәні ақпарат көзін материалдық зат немесе қабылдағышын көрсететін жеке тұлға, мысалы, тапсырыс берушілер, қызметкерлер, жеткізушілер, клиенттер, қойма. Кейбір нысанды немесе сыртқы мәні ретіндегі жүйені анықтау ол талданатын жүйенің шекараларын тыс екендігін көрсетеді. Кейбір талдау процесінде кейбір сыртқы мәндері ішіне ауыстырылуы мүмкін, егер бұл қажет болса немесе керісінше, ақпараттық жүйенің процестерінің бөлігі сыртқы мәні ретінде көрсетілетін және диаграммадан тыс шығарылуы мүмкін.

Сыртқы мәні ретінде жүйенің немесе кейбір нысанды анықтау талданатын ақпараттық шекарадан тыс болатындығын көрсетеді. Талдау процесінде кейбір сыртқы мәндері ішіне ауыстырылуы мүмкін, егер бұл қажет болса, немесе керісінше ақпараттық жүйелердің процестерінің бөлігі диаграммадан тыс шығарылған және сыртқы мәні ретінде көрсетілген.

Күрделі ақпараттық жүйе моделін құрған кезде ол біртұтас түрдегі бір жүйе түріндегі мәнмәтіні диаграмма деп аталатын жалпы түрінде көрсетілуі мүмкін немесе қосалқы жүйелер қатарына декомпозирленуі мүмкін.

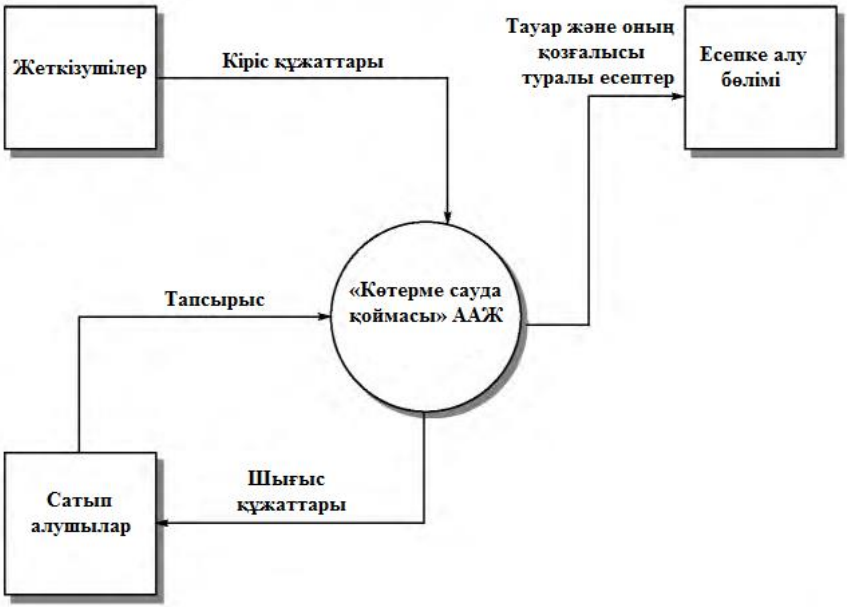
Қосалқы жүйенің нөмірі оны сәйкестендіру үшін қызмет етеді. Аты өрісіне тиісті анықтамалары мен толықтырулары бар ұсыныстар түріндегі қосалқы жүйенің атауы енгізіледі.

Процесс белгілі бір алгоритмге сәйкес деректердің кіру ағынын шығыс деректерге түрленуін білдіреді. Нақты алғанда процесс түрлі тәсілдермен іске асырылуы мүмкін: бұл кіріс құжаттары мен есептерді шығаруды өңдеуді орындайтын ұйымның бөлімшесі (бөлімі) болуы мүмкін, ол аппаратты іске асқан логикалық құрылғы және т.б. Процесс нөмірі оны сәйкестендіру үшін қызмет етеді. Аты өрісіне процестің атауы енгізіледі, бұл ретте «өңдеу», «жаңғырту» немесе «редакциялап алу» сияқты етістіктерді қолдану негізінен, бұл процесті жеткілікті мәнінде түсінбеу мен бұдан әрі талдауды білдіреді. Процестің атауы нақты әрекеттерді анық айқындауы тиіс. Нақты іске асыру өрісіндегі ақпарат ұйым бөлімшесі, бағдарлама немесе аппараттық құрылғы осы процесті орындай керектігін көрсетеді.

Деректердің жинақтағышы жинағышқа орналастыруға болатын кез келген сәтте ақпаратты сақтауға арналған абстрактылы құрылғыны білдіреді және біршама уақыттан кейін алып шығуға болады. Деректерді жинақтаушы картотекадағы жәшік, жедел жадыдағы кестелер, магнитті тасымалдағыштағы файл түрінде нақты іске асырылуы мүмкін.

Гейн-Сарсон нотациясында деректердің жинақтағышы D әрпімен сәйкестендіріледі де, туынды санмен болады. Жинақтағыштың атауы көп ақпараттылықтан таңдап алады.

Деректердің жинақтағышы жалпы жағдайда деректер қорының көрінісі болып табылады және онда сақталатын деректер ақпараттық модельмен байланысуы керек.

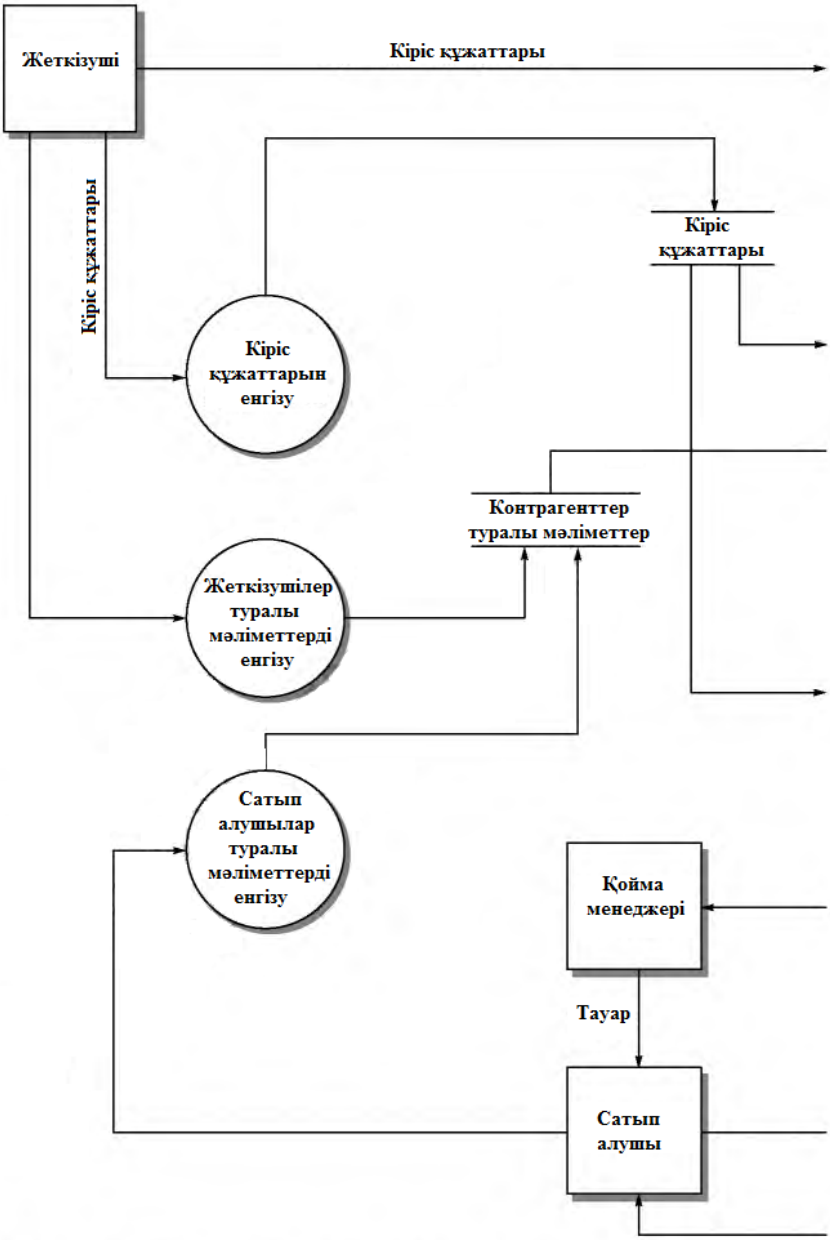


4.6-сурет. «Көтерме сауда қоймасы» ААЖ үшін Йордан нотациясындағы бастапқы мәнмәтінді диаграмма (нөлдік деңгей диаграммасы)

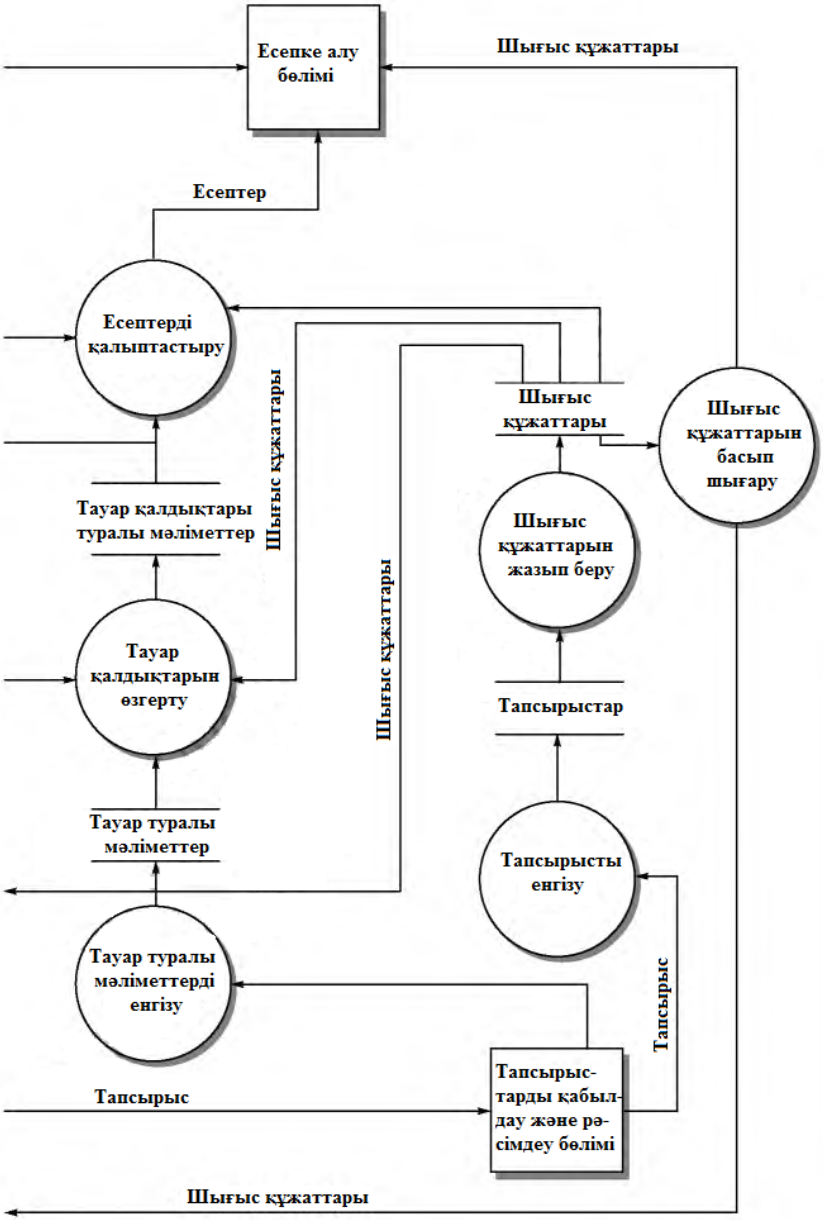
Деректердің ағыны көзінен бастап қабылдағышқа бірнеше жалғағыштар арқылы берілетін ақпаратты анықтайды. Деректердің нақты ағыны екі құрылғы арасында пошта, магнитті таспалар немесе дискеттер арқылы, бір компьютерден басқасына берілетін ақпарат болуы мүмкін. Әрбір деректер ағынының оның мазмұнын білдіретін аты болады.

Деректер ағыны диаграммасының иерархиясын құру мәнмәтінді диаграммадан басталады, ол жүйенің біршама жалпы түрін анықтайды. Мұндай диаграммада жүйенің сыртқы әлеммен интерфейсін сипаттайды, яғни әзірленетін жүйенің тұтынушылармен және ақпарат көздерімен өзара байланысын көрсетеді. Әдетте бастапқы мәнмәтіндік диаграмманың пішіні жұлдыз пішіндес. Егер жүйенің сыртқы мәнінің саны көп болса (10-нан астам), таратылған табиғатында немесе қолданыстағы қосалқы жүйелерін қамтиды, онда мәнмәтіндік диаграммалардың иерархияларын құрайды.

Егжей-тегжей процесінде теңгерім ережесін сақтайды - қосалқы жүйенің егжей-тегжейінде талданатын қосалқы жүйенің ақпараттық байланысы бар қосалқы жүйенің компоненттері қолданылуы мүмкін.



4.7-сурет. «Көтерме сауда қоймасы» ААЖ деректері ағынының диаграммасы


Есепке алу бөлімі
Шығыс құжаттары
Есептер
Есептерді қалыптастыру
Шығыс құжаттары
Шығыс құжаттарын басып шығару
Тауар қалдықтары туралы мәліметтер
Шығыс құжаттарын жазып беру
Шығыс құжаттары
Тауар қалдықтарын өзгерту
Шығыс құжаттары
Тапсырыстар
Тауар туралы мәліметтер
Тапсырысты енгізу
Тауар туралы мәліметтерді енгізу
Тапсырыс
Тапсырыс-тарды қабыл-дау және рә-сімдеу бөлімі
Тапсырыс
Шығыс құжаттары

Егжей-тегжейленбейтін процестерде осы процестің функцияларының сипаттамасынан тұратын функцияларды құрайды. Мұндай сипаттама табиғи тілінде орындалуы мүмкін.

Көтерме сауда үшін сатып алынған тауарлар мен олардың қозғалысы туралы деректерді алуға арналған «Көтерме сауда қоймасы» автоматтандырылған ақпараттық жүйесінің бағдарламалық қамсыздандыруын әзірлеу үшін деректер ағынының диаграммасы үлгісін келтіреміз.

Деректер ағынының диаграммасы иерархиясын құру мәнмәтіні диаграммадан басталады, яғни әзірленетін жүйе ақпарат көздерімен және қабылдағыштарымен өзара әрекет ететіндігін анықтаймыз (4.6-сурет).

Бастапқы құжаттардың тауарлардың келіп түсуі бойынша тауарлардың келіп түсу журналында белгіленіп жазылады. Тауарларды сатуды рәсімдеу мен есепке алу сатып алушылар мен сатушының арасында сатып алынатын тауарларға есептесу тәсіліне байланысты болады. Қойма менеджері тауарларды сатып алу және жіберуді есепке алу журналын жүргізеді. Бастапқы құжаттардың деректері тиісті жинақтағыштарда сақталады.

Әзірленетін жүйе үшін сыртқы мәні жеткізушілер, сатып алушылар, қойма менеджері, есепке алу және бақылау бөлімі, тапсырыстарды қабылдау және рәсімдеу бөлімі болып табылады. Олар туралы мәліметтер тиісті кестелерде (анықтағыштарда) сақталады. Жеткізуші тауарды қоймаға тапсырады.

Тауарды жеткізуге арналған құжаттар (жөнелтпе, шот-фактуралар) деректер қорына енгізіледі.

Сатып алушы тауарларды сатып алуға тапсырыс береді. Тапсырыстарды қабылдау және рәсімдеу бөлімінде келіп түскен тапсырыстың әрбір жолы тексеріледі. Қандай да бір позициясы болмаған жағдайда қоймада жеткізушіге тапсырыс рәсімделеді, яғни қажетті тауарды жеткізу жүргізіледі.

Тапсырыстың негізінде деректер қорында сақталатын тауарларды сатуға құжаттар басып шығарылады да, сатып алушыға беріледі.

Әрбір күннің соңында барлық бастапқы құжаттар бухгалтерге беріледі. Тауардың келіп түсуі мен сатылуы туралы мәліметтердің негізінде қойма менеджерлері мен есепке алу бөлімінің қызметкерлері қоймадағы тауарлардың айналымы мен қалдықтары жөніндегі есептерді қалыптастырады.

«Көтерме сауда қоймасы» ААЖ деректер ағынының диаграммасы 4.7-суретте берілген.

4.4. БАҒДАРЛАМАНЫҢ ФУНКЦИОНАЛДЫ СХЕМАСЫ

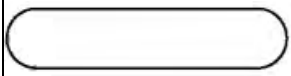
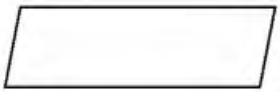
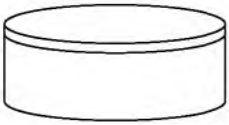
Бағдарламалық қамсыздандыруды жобалау процесі бағдарламалық жүйенің құрылымдық компоненттері мен олардың арасындағы байланысты анықтауды қамтиды. Құрылымды анықтап алу нәтижесі түрлі схемалар түрінде берілуі мүмкін, ол жобаланатын бағдарламалық қамсыздандыру туралы біршама толық түсінік береді.

Схемалар түрлі егжей-тегжей деңгейлерінде қолданылуы мүмкін, мұнда деңгейлердің саны деректерді өңдеу міндеттерінің көлемі мен күрделілігіне байланысты болады. Егжей-тегжей деңгейі түрлі бөліктер мен жалпы арасындағы өзара байланыс түсінікті болатындай болуы керек.

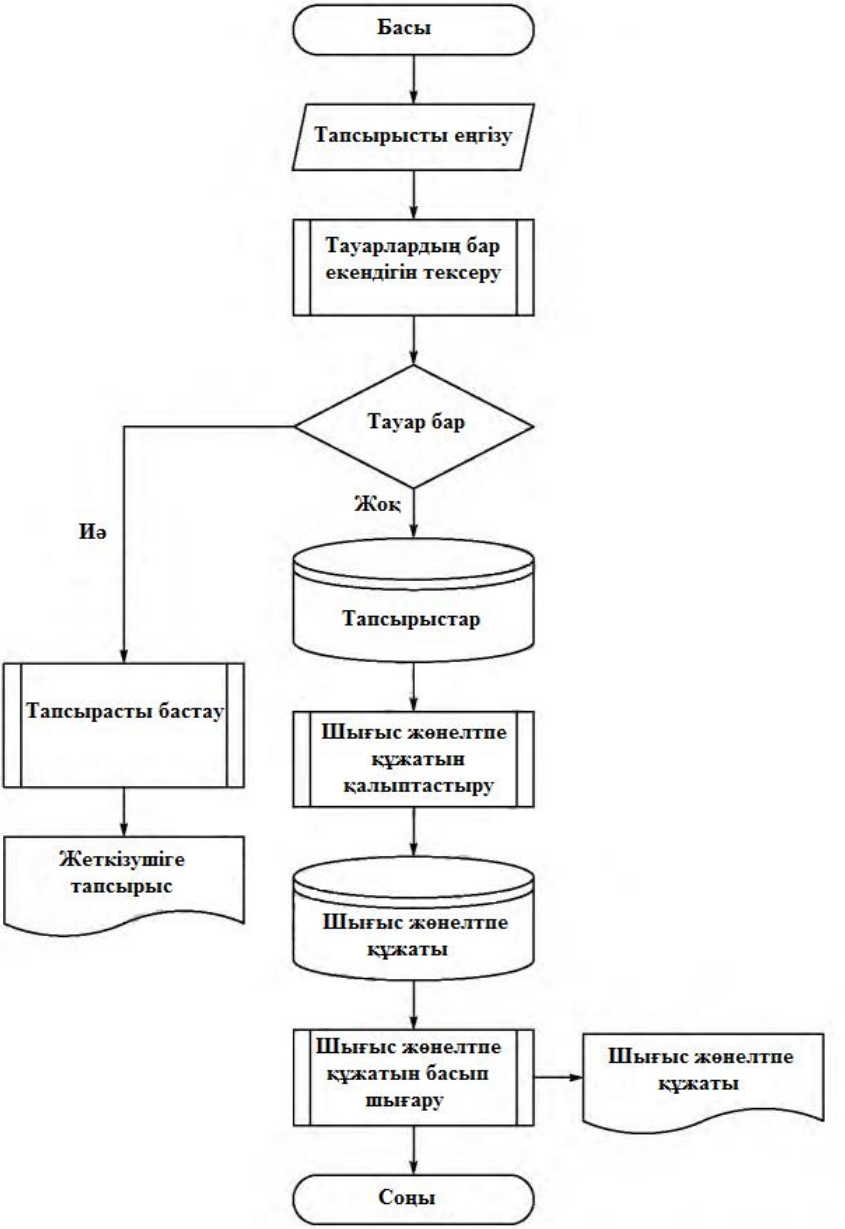
Функционалды схема - бағдарламалық қамсыздандыру компоненттерінің ақпараттық ағындарының сипаттамасымен өзара байланыс схемасы, деректердің ағындағы құрамы мен қолданылатын файлдар және құрылғылардың көрсетілуімен.

Функционалды схемаларды көрсету үшін стандартпен белгіленген (МемСТ 19.701-90) арнайы шартты белгілерді қолданады. Функционалдық схемалардың барлық компоненттері сипатталуы тиіс (4.2-кесте).

4.2-кесте. Алгоритмдердің негізгі блоктарының графикалық шартты белгілері

Атауы	Графикалық шартты атауы	Тағайындалуы
Терминатор		Символ сыртқы ортаға шығуды және ішкі ортаға кіруді көрсетеді (бағдарлама схемасының басы немесе соңы, сыртқы қолдану мен деректер көзі немесе тармағы)
Деректер		Символ деректерді көрсетеді, деректер тасымалдағышы анықталмаған
Тіке қатынайтын есте сақтаушы құрылғы		Символ тіке қатынаумен есте сақтау құрылғысында сақталатын (магнитті дискі, магнитті барабан, икемді магнитті дискі) деректерді көрсетеді

Атауы	Графикалық шартты атауы	Тағайындалуы
Құжат		Символ тасымалдағышта берілген оқуға ыңғайлы түрдегі деректерді (машинограмма, оптикалық немесе магнитті санауға арналған құжат, микрофильм, қорытынды деректері бар таспа орамы, деректерді енгізу блоктары) көрсетеді
Қолмен енгізу		Символ кез келген түрдегі құрылғыдан (пернетақта, ажыратып қосқыштар, жарық құралы, штрих коды бар жолақшалар) өңдеу кезінде қолмен енгізілетін деректерді көрсетеді
Дисплей		Символ көрсететін құрылғы түріндегі тасымадағышта адам оқитын түрде деректерді (көзбен бақылау үшін экран, ақпаратты енгізу индикаторлары) көрсетеді
Процесс		Символ кез келген түрдегі деректерді өңдеу функциясын көрсетеді (белгілі бір операцияларды немесе өзгертуге әкелетін операциялар тобын орындау, ақпараттарды орналастыру түрлері немесе ағынның бірнеше бағытының бірін анықтауға қозғалу керек)
Алдын-ала анықталған процесс		Символ басқа орында (қосалқы бағдарламада, модульде) анықталатын бағдарламаның қадамдарынан немесе бір немесе бірнеше операциядан тұратын алдын-ала анықталған процесті көрсетеді



4.8-сурет. «Көтерме сауда қоймасы» ААЖ сату қосалқы жүйесінің бағдарламалық қамсыздандыру функционалды схемасының үлгісі

4.8-суретте «Көтерме сауда қоймасы» автоматтандырылған ақпараттық жүйесінің тауар сату операциясын іске асыратын бағдарламалық жүйесінің функционалды схемасы берілген.

Құрылымдық тәсілде бағдарламааралық интерфейстердің сипаттамаларын мұқият орындау керек, себебі сипаттама сапасына ең қымбат қателердің саны байланысты болады. Ең қымбат қателерге тестілеу кезінде анықталатын қателер жатады, себебі оларды жою үшін қалыптасқан мәтіндерді өзгерту қажет болады.

БАҚЫЛАУ СҰРАҚТАРЫ МЕН ТАПСЫРМАЛАРЫ

1. Тақырыптық саланың модельдеу мақсаты қандай?
2. Тақырыптық сала модельдерін құрудың үш деңгейін атаңыз.
3. Күрделі жүйелердің проблемаларын шешудің қандай тәсілдері бар?
4. Бағдарламалық қамсыздандыруды әзірлеудің құрылымдық тәсілінің мәнін түсіндіріңіз.
5. Құрылымдық тәсіл әдістемесінің принциптері қандай принциптерге негізделеді?
6. Тақырыптық сала моделіне қандай талаптар қойылады?
7. SADT әдістемесінің ерекшеліктерін атаңыз.
8. SADT қандай блоктар түпкі, ал қандай - бөлшекті деп аталады?
9. SADT диаграммаларын құрудың қандай ережелері бар?
10. Деректер ағыны диаграммасының компоненттерін атаңыз (DFD).
11. DFD теңгерім ережесі қалай шығады?
12. Бағдарламалық қамсыздандырудың функционалдық схемасын құрған кезде қандай компоненттер қолданылады?

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ӘЗІРЛЕУГЕ НЫСАНҒА БАҒЫТТАЛҒАН ТӘСІЛ. UML МОДЕЛЬДЕУ ТІЛІ

5.1. НЫСАНҒА БАҒЫТТАЛҒАН ТӘСІЛДІҢ МӘНІ

Құрылымдық және нысанға бағытталған тәсіл арасындағы принциптік ерекшелік бағдарламалық жүйенің декомпозияция тәсілінде. Нысанға бағытталған тәсіл нысандық декомпозицияны қолданады, бұл ретте жүйенің құрылысы нысандарының терминдерде және арасындағы байланыстарда сипатталады, ал жүйенің жүріс-тұрысы нысандар арасындағы хабарлармен алмасу терминдерінде сипатталады. Нысанға бағытталған тәсілдің негізгі ұғымдары нысан мен класс болып табылады.

Нысан - белгілі бір күйі мен әрекетіне ие кейбір мәні, олар үстінен операциялар мен қасиеттері (атрибуттарының) берілген мәніне ие болмыс.

Класс - құрылымы мен жүріс-тұрысының жалпылығымен байланысқан көптеген нысандар.

Нысанды тек белгіленетін жүріс-тұрысы бар тақырыбы ғана емес, сонымен қатар құбылыс ретінде көрсетуге болады. «Класс данасы» мен «нысан» терминдері эквивалентті болып табылады. Нысанның жағдайы осы нысанның барлық мүмкін (статистикалық) қасиеттерімен және осы қасиеттердің бірінің ағымдағы мәнімен (динамикалық) сипатталады. Жүріс-тұрыс нысанының басқа нысандарға әсерін сипаттайды және керісінше, осы нысандардың күйін өзгерту мен хабарламаларды беруге қатысты. Басқаша айтар болсақ, нысанның жүріс-тұрысы оның әрекеттерімен анықталады. Дербестілік – бұл барық өзге нысандардан ерекшеленетін нысанының қасиеттері.

Операция деп тиісті реакцияны шақыру мақсатымен бір нысанының басқаға белгілі бір әсерін шақыратын әрекетті атайды. Негізінде, нысандық және нысанға бағытталған тілдерде осы нысандармен орындалатын операциялар әдістер деп аталады да, класты анықтаудың

құрамдас бөлігі болып табылады.

Нысандар мұрагерлік ету, инкапсуляция және полиморфизм қасиеттеріне ие.

Мұрагерлік ету деректер мен әдістерді қосу немесе қайта анықтау мүмкіндігімен қолданыстары деректер мен әдістер негізінде жаңа кластарды құруды білдіреді.

Инкапсуляцияны өзінің ішкі іске асыру нысандарын жасыру мүмкіндігі ретінде анықтауға болады.

Полиморфизм кластың бірден артық типке тиесілілік қабілеті ретінде анықтауға болады.

Нысанға бағытталған жүйе бастапқыдан оның эволюциясын есепке алумен құрылады. Мұрагерлік ету мен полиморфизм туынды кластарды - базалық класс ұрпақтарын құру көмегімен кластардың жаңа функционалдылығын анықтау мүмкіндігін қамтамасыз етеді. Ұрпақтары түпкі кластардың өзгеріссіз алғашқы сипаттамасын алады да, қажет болғанда деректер мен әдістердің жеке құрылымдарын қосады. Туынды кластарды анықтау, мұнда тек айырмашылықтары немесе анықтап алу беріледі, ол көбінесе өндіріс кезінде уақыт пен күшті үнемдейді және бағдарламалық код пен сипаттаманы қолданғанда беріледі.

Нысанға бағытталған тәсілдің тұжырымдамалық негізі нысанды модель болып табылады, оның негізгі элементтері - бұл:

- дерексіздендіру;
- инкапсуляция;
- модульділік;
- иерархия.

Дерексіздендіру (абстракция) модельге өзінің мақсатты тағайындау немесе өзінің функцияларын жүйенің орындауына тікелей қатынасы бар жобаланатын жүйенің аспектілерін қосуды болжайды. Бұл ретте барлық қосалқы бөлшектер түсіріледі, олар алынған модельді талдау және зерттеу процесін тым қиындатпау үшін қажет (осылайша, мысалы, тұрғын үйді жобалаған кезде оның ішкі қабырғаларын желімдеу үшін түсқағаздарды іріктеп алуға алаңдаудың қажеті жоқ).

Инкапсуляция оның құрылысы мен жүріс-тұрысын анықтайтын нысанының элементтерін ашуды болжайды. Инкапсуляция ережесі - сенімділікті қамтамасыз ету үшін нысан өрістерінің өзіне тіке қатынаудың қажет жоқ, оқу және олардың мазмұнын жаңарту тиісті әдістерді шақыру арқылы жүргізілуі тиіс. Инкапсуляция дәне дерексіздендіру - өзара толықтырылатын ұғымдар: дерексіздендіру

нысанының сыртқы жүріс-тұрысын белгілейді, ал инкапсуляция осы жүріс-тұрысты қамтамасыз ететін іске асырудан тұрады және ашады. Инкапсуляцияға ақпараттық жабық көмегімен қол жеткізіледі. Әдетте нысандардың құрылымы және оларды іске асыру көрінбейді.

Модульділік - бұл модульдер қатарына оның декомпозиция мүмкіндігімен байланысты бағдарламалық құралдың қасиеті. Модульдер физикалық контейнерлер болып табылады, мұнда логикалық өңдеу кластары мен нысандары жарияланады. Модульдерге декомпозицияның жалпы мақсаты - бағдарламалық қамсыздандырудың жобалануы мен өзгеруі тәуелсіз болатын модульдердің есебінен бағдарламақ қамсыздандыру қиындығын азайту. Модульдерді іске асыруды өзгерту олардың жүріс-тұрыстарына әсерсіз және басқа модульдердің іске асырылуын білусіз жүргізілуі керек.

Иерархия (иерархиялық ұйымдастыру) - бұл иерархиялық құрылым абстрациясынан қалыптасу, яғни олардың деңгейлері бойынша орналасуы. Иерархиялық құрылымның негізгі түрлері күрделі жүйелерге қатысты болып табылуы:

- класс құрылымы (бір класс бір немесе бірнеше басқа кластардың бөлігін құрылымдық немесе функционалдық бөлігін қолданады);
- нысандардың құрылымы (агрегация, мысалы «жазба» типінің құрылымы).

Нысанға бағытталған тәсіл келесі *артықшылықтарға* ие:

- нысанды декомпозиция анық көрінетін құралдардың қажетті үнемділігін қамтамасыз ететін жалпы механизмдерді қолдану арқылы аз көлемді модельдерді құру мүмкіндігін береді. Нысанды тәсілді қолдану әзірлеу деңгейін арттырады және қайта қолдану үшін жарамдылығын арттырады, ол модельдерді жинап құруға және әзірлеу ортасын құруға әкеледі;
- нысанды декомпозиция күрделі модельдерді құрудан бас тартуға мүмкіндік береді, себебі ол азғантай қосалқы жүйелер негізінде модельдің дамытудың эволюциялық жолын болжайды;
- нысанды модель табиғи, себебі әлемді адамның қабылдауына бағдарланған.

Нысанға бағытталған тәсілдің *кемшіліктеріне* жобалау кезіндегі жоғары бастапқы шығындар жатады. Бұл тәсіл өзін бірден өтемейді. Оны қолданудан тиімділігі екі-үш жобаны әзірлеуден кейін және қайта қолданылатын компоненттерінің жиналуынан кейін көрінеді.

5.2. UML - БІРІЗДЕНДІРІЛГЕН МОДЕЛЬДЕУ ТІЛІ

Нысанға бағытталған тәсілдің көптеген қолданыстағы әдістері модельдеу тілі мен модельдеу процесін сипаттауды қамтиды.

Модельдеу процесі - бұл жобаны әзірлеу кезінде орындау қажет қадамдардың сипаттамасы.

Нысанды тәсілдің модельдеу тілі ретінде ***UML біріздендірілген модельдеу тілі*** (Unified Modeling Language) қолданылады, модельдеу үшін диаграммалардың стандартты жиынтығынан тұрады.

UML - бағдарламалық қамсыздандыруды әзірлеу саласындағы нысанды модельдеуге арналған графикалық сипаттама тілі.

UML-модель деп аталатын жүйенің дерексіз моделін құру үшін графикалық шартты атауларын қолданатын ашық стандарт. UML негізгі бағдарламалық жүйеде анықтау, көзбен көру, жобалау және құжаттау үшін құрылған. UML бағдарламалау тілі болып табылмайды, бірақ UML-модельдерінің негізінде бағдарлама кодының генерациясы мүмкін.

UML модельдеудің әмбебап тілін әзірлеу өткен ғасырдың 90-ншы жылдарының ортасынан нысанға бағытталған бәрнеше тәсілдердің негізінде және бағдарламалық қамсыздандырудың сипаттама нотациясы негізінде басталды.

Бағдарламалық қамсыздандырудың әмбебап сипаттама тілін құруға себеп болған шешілетін міндеттердің өз кезегінде қиындауымен жобаланатын бағдарламалық өнімдердің үнемі болатын қиындықтары болып табылады.

Ақпараттық жүйе нысандардың саны 7-8 асқанда (адам қосымша жазбасыз ақпаратты негіздей алмайтын психологиялық бөгеттері) жүйені жобалау кезінде туындайтын қиындықтар арнайы құралдарсыз мүмкін емес. Мұндай бағдарламалық қамсыздандыру (бір жұмыс орны үшін немесе шағын компания үшін) бір адамды құруға қабілетті.

Нысандардың, жағдайлардың және олардың арасындағы өткелдердің саны бірнеше мыңға жеткенде, онда ешбір маман қаншалықты тәжірибелік және білімді болғанына қарамастан бүкіл жүйені толық қамтуы мүмкін болмайды.

Мұндай жүйелер әзірлеушілер тобының функционалды рөлдерін бөлумен құрылады. Онда қарым-қатынас құралы қажет болады, ол UML болып табылады.

UML әзірлеудің басты мақсаттары мыналар болды:

- пайдаланушыларға ойланып өлшенген модельдерді әзірлеу мен модельдеу тілінің көмегімен алмасуына мүмкіндік береді;
- тақырыптық нақты салада жүйенің модельдерін біршама толық беру үшін сипаттау мен ұлғайту механизмдерін көздеу;
- әзірлеу процестері мен бағдарламалаудың нақты тілдеріне тәуелсіздікті қамтамасыз ету;
- осы модельдеу тілін түсіну үшін қалыпты негізін қамтамасыз ету;
- нысанға бағытталған аспаптық құралдардың нарығының өсуін ынталандыру;
- үздік практикалық тәжірибені біріктіру.

Қазіргі таңда UML ақпараттық жүйелер мен олардың бағдарламалық қамсыздандыруын құру процестерін құжаттаудың көпшілік қабылдаған стандарты болып табылады.

UML тілін конструктивті пайдалану күрделі жүйелердің жалпы принциптерін және нысанға бағытталған талдау процесінің және жобалау ерекшеліктерін түсінуге негізделеді, көбінесе:

- дерексіздендіру принципі;
- көп модельділік принципі;
- күрделі жүйелердің модельдерін иерархиялық құру принципі.

Бұл принциптер жоғарыда еске салынған болатын.

Көп модельділік принципі ешқандай жалғыз модель күрделі жүйенің түрлі аспектілерін сипаттау барабарлығының жеткілікті дәрежесімен сипатталмайтыны туралы пікірді береді.

Нысанға бағытталған модель әдістемесіне қатысты күрделі жүйенің толық моделі өзара байланысты түсініктердің кейбір санын беретіндігін білдіреді, жүйе құрылымы немесе оның кейбір аспектісін білдіреді.

Иерархиялық құру принципі модельді түрлі абстракция деңгейлерінде немесе белгіленген түсініктер шеңберінде детализация деңгейлерінде құрылуын айтады. Бұл ретте күрделі жүйенің бастапқы немесе алғашқы моделі жалпы көріністе (метакөрініс) болады. Мұндай модель жобалаудың бастапқы сатысында құрылады және модельделетін жүйенің көптеген тетістері мен аспектілерінен тұрмайды.

UML концептуалды моделі үш құрамдас бөлшектерін қамтиды: негізгі құрылыс блогы тілі, олардың үйлесім ерекшелігі мен барлық механизм тілі үшін кейбірі. UML - бұл тіл, оған кіретін сөздерді және мағынасы бар конструкцияларды алатын сөздіктер мен ережелерден тұрады. Модельдеу тілінде сөздік пен ережелер жүйені тұжырымдамалық және физикалық көрінісін беруге бағдарланды.

UML тіл сөздігі үш құрылыс блогын қамтиды:

1) мәні;
2) қатынасы;
3) диаграммалар.

Мәндері - бұл модельдің негізгі элементтері болып табылатын дерексіздік. *Қатынастар* түрлі мәндерді байланыстырады. *Диаграммалар* - бұл арнайы графикалық конструкциялар, олар түрінде модельдер көрінеді. Диаграммалар мәндердің жиынтығынан тұратын мүддені топтастырады.

Тақырыптық саланы немесе жобаланатын жүйенің түрлі аспектілерін UML тілінде модельдеу үшін келесі диаграмма түрлері көзделген:

- қолдану нұсқаларының диаграммасы (Use Case Diagram);
- кластар диаграммасы (Class Diagram);
- жүріс-тұрыс диаграммасы (Behavior Diagram), оның ішінде:
  - ✓ жағдай диаграммасы (Statechart Diagram);
  - ✓ қызмет диаграммасы (Activity Diagram);
- өзара байланыс диаграммасы (Interaction Diagram), оның ішінде:
  - ✓ жүйелілік диаграммасы (Sequence Diagram);
  - ✓ кооперация диаграммасы (Collaboration Diagram);
- іске асыру диаграммалары (Implementation Diagram), оның ішінде:
  - ✓ компоненттер диаграммасы (Component Diagram);
  - ✓ бұрау диаграммасы (Deployment Diagram).

UML-диаграммалармен сипатталған жүйе жобалау процесінде қолжеткізу керек болатын нәтидені әзірлеушіге көрсетеді.

UML тілінің модельдері екі түрге бөлінеді:

1) *құрылымдық модельдер (тұрақты модельдер)* тақырыптық саланың мәнін сипаттайды немесе модельденетін жүйенің компоненттерін, олардың кластары, атрибуттары, байланыстары, интерфейстерін қоса алғанда сипаттайды; бұл модель түрлеріне пайдалану, класс, компоненттер, бұрау нұсқаларының диаграммалары жатады;
2) *жүріс-тұрыс модельдері (динамикалық модельдер)* тақырыптық саланың мәнінің немесе жүйенің компоненттерінің жұмыс істеуін сипаттайды, олардың әдістерін, арасындағы өзара байланысты сипаттайды, жалпы жүйенің жеке мәндерінің, компоненттерінің өзгеруі; бұл модель түріне күйінің диаграммасы, қызмет диаграммасы, жүйелілік диаграммасы, кооперация диаграммасы жатады.

UML тілінің барлық модельдері үш деңгейге бөлінеді.

1. *Тұжырымдамалық модельдер* модельденетін жүйенің жоғарғы,

біршама жалпы және дерексіздік деңгейін білдіреді. Бұл деңгейге пайдалану нұсқаларының диаграммасы жатады. Тұжырымдамалық модель деңгейінен тақырыптық саланы немесе жобаланатын жүйені (бағдарламалық құралды) модельдеу басталуы керек.

2. *Логикалық модельдер* модельденетін жүйені сипаттаудың екінші деңгейін береді. Бұл деңгей моделінің элементтері нақты толығу жоқ және тақырыптық саланың немесе жүйенің құрылымы мен жүріс-тұрысының логикалық аспектілерін көрсетеді. Логикалық модельдер тұжырымдамалық модельдерден кейін құрылуы тиіс. Бұл деңгейге кластар, күйлер, қызмет, жүйелілік, кооперация диаграммалары жатады

3. *Физикалық модельдер* модельденетін жүйені сипаттаудың төменгі деңгейін білдіреді. Бұл деңгей модульдерінің элементтері нақты жүйенің нақты материалдық мәнін білдіреді. Бұл модельдерді соңғы кезекке құру ұсынылады. Бұл деңгейге компоненттер мен бұрау диаграммалары жатады.

UML тілінде графикалық конструкциялардың төрт негізгі түрі қолданылады.

1. *Белгілер немесе пиктограммалар.* Белгілер белгіленген өлшемдегі және пішіндегі графикалық фигуралар түрінде келеді. Ол өз ішінде қосымша символдарды орналастыру үшін өлшемдерін ұлғайта алмайды. Белгілер өзге графикалық конструкциялардың ішінде сияқты, оның сыртына да орналаса алады. Белгілердің мысалдары болып диаграммалардың элементтерінің байланысы немесе кейбір өзге қосымша шартты атаулар (сәнді белгілер) бола алады.

2. *Жазықтағы графикалық символдар.* Мұндай екі өлшемді символдар кейбір геометрикалық фигуралардың көмегімен бейнеленеді және UML тілінің өзге конструкцияларының фигураларының ішіне орналастыру мақсатымен ені мен биіктігі әртүрлі болады. Осындай символдардың ішінде жиі орналасатын мәтін жолдары, олар UML тілінің тиісті элементтерінің жеке қасиетінің семантикасын анықтайды немесе белгілейді. Жобаланатын жүйенің нақты моделі үшін фигуралардың ішіндегі ақпараттың мәні маңызды, себебі бағдарламалық кодтағы тиісті элементтердің іске асырылуын реттейді.

3. *Бөлек графикалық символдарды жалғастыратын сызықтардың кесінділерінен тұратын жүйелілікті көрсететін жолдар.* Бұл ретте сызықтардың кесінділерінің соңғы нүктелері граф теориясында қабылданғандай диаграммалардың шыңын белгілеу үшін қызмет ететін геометриялық фигуралармен түйісуі керек. Тұжырым жағынан алғанда UML тілінде жолдарға ерекше мән беріледі, себебі олар қарапайым

топологиялық мәндері болып табылады. Басқа жағынан, жолдың жеке жолдары немесе сегменттері олардың жолынан тыс болмауы мүмкін. Жолдар тиісті сызықтар кесіндісінің екі шекарасында өзге графикалық символдармен түйісуі керек. Басқаша айтар болсақ, жолдар сызықтар диаграммасында үзілмеуі керек, олар ешбір графикалық символмен түйіспеуі керек. Жоғарыда аталып өткендей, жолдар арнайы графикалық фигура - белгінің аяқталуы ретінде немесе терминатор болуы мүмкін, ол осы жолдың сегменттері болып табылатын сызықтардың ұштарының бірінде көрсетіледі.

4. *Мәтін жолдары*. Түрлі ақпарат түрлерінің грамматикалық формасында берілуі үшін қызмет етеді. Мәтіннің жолдарын әрбір қолдану UML тілінің нотациясындағы синтаксиске сәйкес келуі тиіс, оның көмегімен осы жолдың грамматикалық талдауы іске асырылуы мүмкін. Соңғысы модель туралы толық ақпарат алу үшін керек. Мысалы, түрлі секциялардағы мәтіннің жолдары осы кластың атрибуттарына немесе оның операцияларына сәйкес келуі мүмкін. Жолдарды қолдануға маңызды шарты қатарласып келеді - барлық мүмкін символдардың семантикасы алдын-ала UML тілінде анықталуы тиіс немесе нақты модельде оны ұлғайту тақырыбы болуы керек.

UML салу блоктарын бір-бірімен біріктіруге болмайды. Басқа тілдер сияқты UML тілі ережелер жиынтығымен сипатталады, олар жақсы рәсімделген модель қалай көріну қажеттілігін анықтайды, яғни семантикалық өзіндік келісілген және онымен байланысты барлық модельдермен үйлесімде.

Жоғарыда аталып өтілген диаграммалардың көбісі өз кезегінде арнайы түрдегі бағандарды негізіне алатын болып табылады, олар геометриялық фигуралар түріндегі шыңдарынан тұрады, олар қабырғаларымен немесе доғаларымен өзара байланысқан. Бағандардан тұратын ақпараттың топологиялық сипаты болғандықтан, геометриялық өлшемдер де, диаграмма элементтерінің орналасуы да (кейбір ерекшеліктерін алмағанда, уақыт осі өлшенетін жүйелілік диаграммасы сияқты) қатып қалған мәні жоқ.

Бұдан әрі біршама жиі қолданылатын кейбір UML-диаграммалар қарастырылатын болады.

Пайдалану нұсқаларының диаграммасы модельденетін тақырыптық саланың немесе жүйенің функционалды тағайындалуын сипаттайды.

Кластар диаграммасы қосымшалар кодын құруға арналған негіздеме болып табылады. Ол бағдарламалық құралдың ішкі құрылымын, оның мұрагерлік етуін және кластың өзара жағдайларын

сипаттайды.

Күй диаграммасы кейбір оқиғаларға жауап ретінде орындалатын өтулер мен күйлердің мүмкін бірізділігін сипаттайды. Бұл диаграмма өмірлік цикл бойы модельдің элементтерінің жүріс-тұрысын сипаттайды.

Қызметтік диаграммасы жүйеде операцияларды орындаудың алгоритмдік және логикалық іске асыруын модельдейді және нысанға бағытталған қосымшаларда қолдануға арналған алгоритмдердің схемаларының аналогы болып табылады.

Жүйелілік диаграммасы модель нысандарының уақыттағы өзара байланысын сипаттайтын синхронды процестерін көрсетеді. Бұл модельдегі уақыт анық түрінде болады.

Компоненттер диаграммасы жобаланатын жүйенің физикалық көрінісін сипаттайды және оның сәулетін модульдер, алғашқы және орындалатын кодтардың, файлдардың терминдеріндегі сәулетін анықтауға мүмкіндік береді.

Орналастыру диаграммасы (бұрау диаграммасы) таратылған жүйенің жалпы конфигурациясы мен топологиясын көрсетеді. Бұл диаграмма бағдарламалық компоненттерді жүйенің жеке тораптары мен олардың арасында ақпараттың берілу маршруты бойынша таратуын көрсетеді.

5.3. ПАЙДАЛАНУ НҰСҚАЛАРЫНЫҢ ДИАГРАММАСЫ

Пайдалану нұсқалары диаграммасы тақырыптық саланы, жүйені немесе бағдарламалық құралды модельдеу кезінде әзірленетін диаграммалардың алғашқысы болып табылады. Ол функционалдық талаптардың сипаттама тізімін әзірлеу кезінде негіз болып табылады және жобаланатын жүйені одан әрі модельдеудің толықтығы мен дұрыстығы жағынан алып қарағанда негізгі орын алатын мәніне ие.

Бағдарламалық қамсыздандыру сипаттамаларын әзірлеуді техникалық тапсырмада көрсетілген функционалдылыққа қойылатын талаптарға талдаудан бастайды. Осы талдау процесінде бағдарламалық қамсыздандырудың сыртқы пайдаланушыларын және нақты пайдаланушылармен өзара байланыстыру процесінде оның қырларының тізбесін анықтайды. Бағдарламалық қамсыздандыру

әрекетінің қырлары «пайдалану нұсқасы» немесе «үлгі боларлық» (use cases) деп аталады. Пайдалану нұсқасы пайдаланушы мен жүйе арасындағы типтік өзара байланысты сипаттайды.

Пайдалану нұсқалары диаграммасының басты мақсаты жүйеге қойылатын функционалдық талаптарды рәсімдеуде болады. Пайдалану нұсқаларының диаграммасы жобалаудың ерте сатыларында жүйеге қойылатын функционалдық талаптарды тапсырыс берушімен келісу үшін негіз болуы мүмкін.

Пайдалану нұсқасы - жүйенің тек адамдар ғана емес, сондай-ақ өзге жүйелер мен құрылғылар болуы мүмкін кейбір сыртқы нысандар (қолданыстағы тұлға) бастайтын оқиғаға жауап ретінде орындалатын әрекеттерінің бірізділігі (транзакциялары). Пайдалану нұсқасы пайдаланушы мен жүйе арасындағы типтік өзара байланысты сипаттайды.

Әрбір пайдалану нұсқасы жеке мәні бар мақсатпен байланысқан, мысалы, «Тақырыбын қалыптастыру» мәтіндік редакторы үшін - бұл пайдалану нұсқасы, ал «Арнайы стильдермен тақырыптарын байланыстыру» - тақырыптың автоматты құрылуы мүмкін болатын операция.

Процедураны орындау мақсатына байланысты келесі пайдалану нұсқаларын ажыратады:

- *негізгі (басты)* - әзірленетін БҚ талап етілетін функционалдылығын қамтамасыз етеді;
- *қосалқы* - жүйені баптаудың және оған қызмет көрсетудің (мысалы, ақпараттарды мұрағаттау және т.с.с.) орындалуын қамтамасыз етеді;
- *қосымша* - пайдаланушы үшін қосымша ыңғайлылықты қамтамасыз етеді (ереже бойынша, әзірлеу де де, пайдалануда да қандай бір ресурстар талап етілмегенде ғана іске асады).

UML тілінің талаптарының бір жобаланатын жүйелердің модельдері туралы ақпаратты беруге арналған диаграммалардың жеткіліктілігі болып табылады. Алайда, UML тілінің бейнелеу құралдары күрделі жүйенің функционалды әрекет етуінің ерекшеліктерін пайдалану нұсқалары диаграммасында есепке алуы үшін жеткіліксіздігі анық. Осы мақсатпен бұл диаграммалар типін мәтіндік сценариймен толықтыруға ұсыныс беріледі, оны пайдалану нұсқаларын орындаған кезде атқаратын жүйелердің әрекеттерін анықтайды немесе егжей-тегжейін ашады.

Сценарий (***scenario***) - әрекеттердің белгілі бір бірізділігі, ол қарапайым мәтін формасында актерлардың әрекеттері мен модельденетін жүйенің әрекетін сипаттайды.

UML тілінің мәнмәтінінде сценарий актерлер мен пайдалану нұсқаларының өзара байланысын қосымша көрнекі түрде көрсету үшін қолданылады. Осыған ұқсас сценарийлерді ұсыну немесе жазу тәсілдері ұсынылады.

Мұндай үлгілердің бірі төменде қарастырылады және тұжырымдамалық модельдеудің алғашқы сатыларында қолдану үшін ұсынылуы мүмкін. Дерексіздік деңгейіне байланысты пайдалану нұсқасы қысқа немесе біршама толық сипатталуы мүмкін. Сипаттаманың қысқаша түрі: пайдалану нұсқасының атауынан, оның мақсатынан, әрекет ететін тұлғаларынан, пайдалану нұсқасының типінен (негізгі, жанама немесе қосымша) және оның қысқаша сипаттамасынан тұрады. Төменде жеке пайдалану нұсқасының сценарийін жазуға арналған үлгіден тұратын кесте (5.1-кесте) берілген.

Пайдалану нұсқаларының сценарийлерін жазған кезде сценарий мәтіні пайдалану нұсқасы диаграммасын анықтауға немесе толықтыруға тиіс, оны толығымен ауыстыруға қолданылмайтындығын түсіну маңызды. Керісінше жағдайда модельдерді көзбен көрсетудің артықшылықтары жоғалатын болады.

Пайдалану нұсқаларының диаграммалары жүйенің күтілетін әрекеттерін көрнекі түрде көрсетуге мүмкіндік береді. Пайдалану нұсқалары диаграммасының негізгі ұғымдары табылады: әрекет етуші тұлға, пайдалану нұсқасы және байланыс. 5.1-суретте пайдалану нұсқалары диаграммаларының көрінісінде қолданылатын шартты атаулар көрсетілген.

Әрекет етуші тұлға (актер) - әзірленетін жүйеге қатысты мәні, ол қандай да бір ақпаратты алу немесе ұсыну мақсатымен онымен өзара байланысады. Жоғарыда аталып өткенде әрекет ететін тұлғалар пайдаланушылар, басқа БҚ немесе жүйемен өзара байланысты қандай да бір техникалық құрал болуы мүмкін.

Пайдалану нұсқасы - әрекет ететін тұлға оның нақты міндетін шешетін кейбір сөзсіз міндеті. Барлық пайдалану нұсқалары әзірленетін жүйенің функционалдылығына қойылатын талаптарымен қалай да байланысқан және орындалатын жұмыстарының көлеміне қарай қатты ерекшеленуі мүмкін.

Байланыс - әрекет ететін тұлғалар мен тиісті пайдалану нұсқаларының өзара байланысы.

Пайдалану нұсқалары сондай-ақ өзара байланысуы мүмкін. Бұл ретте пайдалану және кеңею байланыстары белгіленеді.

5.1-кесте. Пайдалану нұсқасының сценарий үлгісі

Пайдалану нұсқасының аты	Пайдалану нұсқасын сәтті орындауға әкелетін оқиғалардың типтік барысы	1-ерекшелік	1-ескертпе
Актерлер		2-ерекшелік	2-ескертпе
Мақсаты		3-ерекшелік	3-ескертпе
Қысқаша сипаттамасы			
Типі			
Өзге пайдалану нұсқаларына сілтемелер		*n*-ерекшелік	*n*-ескертпе

5.1-сурет. Пайдалану нұсқалары диаграммасының негізгі шартты белгілері

а- әрекет етуші тұлға; б- пайдалану нұсқасы; в- байланыс

Пайдалану (uses/include) әзірленетін БҚ бірнеше рет пайдаланылу нұсқасында кейбір әрекеттерінің үзіндісін білдіреді. Бұл үзіндіні жеке пайдалану нұсқасы ретінде рәсімдейді және онымен «пайдалану» типімен байланысын көрсетеді.

Ұлғайту (extends) қолданылады, егер екі ұқсас пайдалану нұсқасы болса, олардың бірінде кейбір қосымша әрекеттердің болуымен ерекшеленеді. Бұл жағдайда қосымша әрекеттер «ұлғайту» типті байланысу нұсқасымен байланысқан жеке қолдану нұсқасы ретінде анықтайды.

Пайдалану нұсқалары диаграммасының басты мақсаты жобалаудың алғашқы сатыларында алынған модельді тапсырыс берушімен келісу мүмкіндігі және жүйеге қойылатын функционалдық талаптарды нысандандыруда болады. Пайдалану нұсқаларының кез келгені алғашқы мәнін түзетін жеке элементтерге пайдаланудың көптеген қосалқы нұсқаларына бұдан әрі декомпозияцияға жатуы мүмкін.

Функционалдық талаптардың сипаттамасының ерекшеліктерін көрсету үшін пайдалану нұсқалары диаграммасында «Көтерме сауда қоймасы» ақпараттық жүйесін бағдарламалық қамсыздандырудың моделін қарастыруға болады.

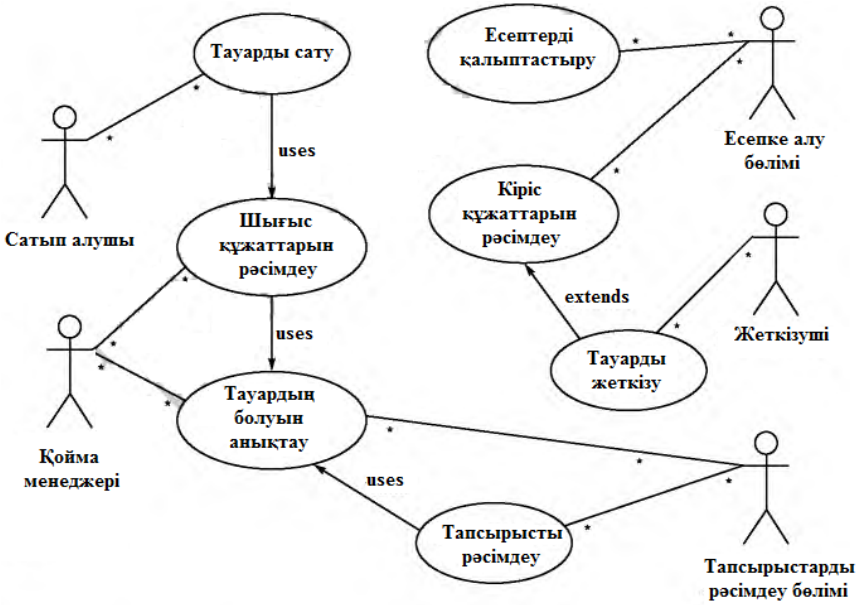
Бағдарламалық жүйенің құрылымын бастапқыдан түсіну үшін әрекет етуші тұлғалар (адамдар немесе олардың арасында өзара байланыс болатын жүйелер) анықталады. Қарастырылып отырған жүйенің бес актері бар, олардың екеуі контрагенттер, ал басқалары барлық операциялардың орындалуын жүзеге асыратын қойма менеджерлері болып табылады. Осы актерлардың әрқайсысы жүйемен өзара байланысты, әйтсе де басты актерлар жеткізушілер мен сатып алушылар (контрагенттер) болып табылады, себебі олар жүйенің функционалдылығына бастама болады. Бұдан әрі пайдалану нұсқалары қалыптасады, яғни әрекет ететін тұлғалардың (актерлардың) қарым-қатынасын жүзеге асыру үшін жүйе орындайтын әрекеттер.

Актерлардың (әрекет етуші тұлғалардың) әрқайсысы жүйеге қатысты белгілі бір тізбекпен жүреді: жеткізуші - тауарды қоймаға

тапсырады, сатып алушы - тауарды сатып алады, қойма менеджері - тауарды қабылдап алады және жібереді, есепке алу бөлімінің менеджері - келіп түсу және сату көлемін анықтайды және тауар қорына талдау жүргізеді. Осы мақсаттардың негізінде негізгі пайдалану нұсқаларын қалыптастыруға және олардың арасындағы өзара байланысты талдауға болады (5.2-сурет).

Шын мәнінде пайдалану нұсқалары бұдан да үлкен болуы мүмкін. Мысалы, клиенттің төлем қабілеттілігін тексеру, тауар туралы ақпарат алу, қоймадағы тауар қорларын бағалау, төлем алу, ақпаратты экранға шығару және т.б. Алайда, бұл диаграмма жалпы жүйенің қалай жұмыс істейтіндігін түсіндіреді.

Қарастырылып отырған жүйенің пайдалану нұсқаларының модельдерін әзірлеудің келесі сатысында бұл диаграмманы мәтіндік сценариймен толықтыруға болады, ол бұрын ұсынылған үлгі негізінде жазылған. Бұл



5.2-сурет. «Көтерме сауда қоймасы» ААЖ бағдарламалық қамсыздандыруды жобалау үшін пайдалану нұсқаларының диаграммасы

сценарий тауар келіп түсетін және сату процесінде жүйе мен актерлардың орындайтын жеке әрекеттерінің мазмұны мен логикалық бірізділігін аша отырып, диаграмманы толықтыратын болады. Бұл жағдайда сценарийді кесте түрінде көрсету ыңғайлы, олардың әрқайсысы шаблонның жеке тарауын сипаттайды.

Сценарийдің басты тарауында (5.2-кесте) пайдаланудың қарастырылатын нұсқасының аты, олармен байланысты актерлардың аттары, нұсқаларды орындау мақсаты, өзге пайдалану нұсқалары шартты типі мен сілтемелері көрсетіледі.

Сценарийдің келесі тарауында (5.3-кесте) қарастырылып отырған пайдалану нұсқасын сәтті орындауға әкелетін әрекеттердің бірізділігі сипатталады. Бұл ретте әрекеттердің бастамашасы болып Сатып алушы актер шығуы тиіс. Келесі сілтемелердің ыңғайлы болуы үшін әрбір әрекет жүйелілікпен реттік нөмірмен белгіленеді.

Сценарийдің үшінші тарауында (5.4-кесте) ерекше жағдайларда немесе ерекшеліктер пайда болғанда орындалатын әрекеттердің жүйелілігі сипатталады.

Берілген сценарийді толықтыруға болады, ұқсас тәсілмен «Тапсырыстарды рәсімдеу» және «Тауарлардың бар екендігін анықтау» пайдалану нұсқаларын сипаттап қана қоймай, сонымен бірге басқа ерекшеліктерді қарастырамыз, мысалы тұрақты сатып алушыларға жеңілдіктерді рәсімдеу және т.с.с.

5.2-кесте. Сценарийдің басты тарауы

Пайдалану нұсқасы	Тауарды сату
Актерлер	Сатып алушы, тапсырыстарды рәсімдеу бөлімінің менеджері, қойма менеджері
Қысқаша сипаттамасы	Сатып алушы тауарға сұраныс береді. Тапсырыстарды рәсімдеу бөлімі тауарды резервке қояды да, тапсырысты рәсімдеп, қойма менеджеріне тапсырысты береді. Сатып алушы тауарды төлейді, тауарды қоймадан алады
Мақсаты	Қажетті тауарды алу
Типі	Базалық
Өзге пайдалану нұсқаларына сілтеме	Пайдалану нұсқаларын қамтиды: тауардың бар екендігін анықтау; тапсырысты рәсімдеу

5.3-кесте. Пайдалану нұсқасын сәтті орындау кестесі

Актерлардың әрекеттері	Жүйенің жауабы
1. Сатып алушы тауарды сұратады. №1 ерекшелік. Қоймада сұратылған тауар қажетті мөлшерде жоқ	2.Тапсырыстарды рәсімдеу бөлімінің менеджері қоймада қажетті тауардың болуын тексереді. 2. 3.Тапсырыстарды рәсімдеу бөлімінің менеджері қажетті тауарды резервке қояды.
4.Сатып алушы тауарды төлейді. №2 ерекшелік.Сатып алушы тауарды төлеген жоқ	5.Тапсырыстарды рәсімдеу бөлімінің менеджері тауарды алуға рұқсат береді. 6.Тапсырыстарды рәсімдеу бөлімінің менеджері тапсырысты қоймаға береді. 5. 7.Қойма менеджері сатып алушыға тауар мен шығыс жөнелтпе құжатын береді.

Бұл ретте пайдалану нұсқасының сценарийлері мен модельдерінің толықтығы нақты жоба шеңберінде қалыптасатын функционалдық талаптармен анықталатын болады.

Пайдалану нұсқалары мәтіндері бар ескертпелермен қосымша сипатталып тізілуі мүмкін, олар кейін атрибуттарымен бірлескен операциялардың прототиптері мен әдістері болуы мүмкін.

5.4-кесте. Ерекше жағдайларды өңдеу

Актерлардың әрекеттері	Жүйенің жауабы
№1 ерекшелік. Қоймада сұратылған тауардың қажетті мөлшері жоқ	
4.Сатып алушы тауарды төлейді	5.Тапсырыстарды рәсімдеу бөлімінің менеджері қажетті тауардың жеткізілуіне бастама болады
Ерекшелік №2. Сатып алушы тауарды төлеген жоқ	
	6. Тапсырыстарды рәсімдеу бөлімінің менеджері сатып алушының тауарды алуын жауып тастайды

Ескертпе (note) модельге туынды еркін ақпаратты қосуға арналған,

ол әзірленетін жобаның мәнмәтініне тікелей қатысы бар.

Мұндай ақпарат ретінде әзірлеушілердің пікірлері (мысалы, диаграмманы немесе оның жеке компоненттерін әзірлеу күні мен нұсқасы) шектеулер (мысалы, жеке байланыстар мәні немесе мәнінің даналары) және белгіленген таңбалары. Пайдалану нұсқаларының диаграммаларына қатысты ескертпелерге осы немесе басқа пайдалану нұсқаларының мәнмәтініне қатысты анықтаушы ақпараттар орын алуы мүмкін. Барлық типтегі диаграммалардағы графикалық ескертпелер жоғарғы оң жақ бұрышы «бүктелген» тік төртбұрыштармен белгіленеді (5.3-сурет). Ескертпе мәтіннің өзі осы тікт төртбұрыштың ішіне орналасады.

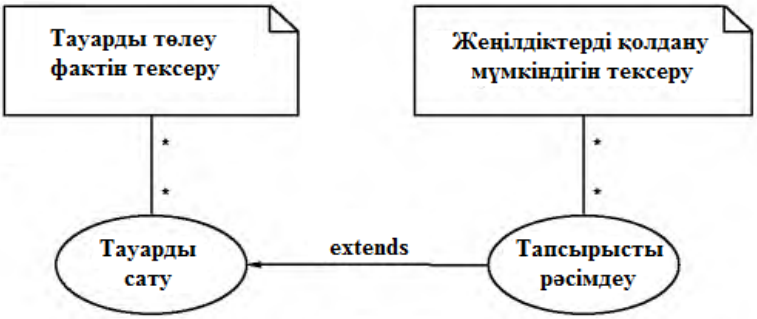
Ескертпе диаграмманың кез келген элементіне қатысты болуы мүмкін, бұл жағдайда олар сызықтармен жалғанады. Егер ескертпе бірнеше элементтерге жатса, онда одан тиісінше бірнеше сызық кетеді.

Бұрын атап өтілгендей, ескертпелер пайдалану нұсқалары диаграммасында ғана емес, сондай-ақ канондық өзге диаграммаларда да болуы мүмкін.

Базалық пайдалану нұсқалардың бірінен келесіде жиі пайдалану нұсқаларына декомпозицияға түсуі мүмкін. Бұл ретте модельдегі актерлардың жалпы саны 20-дан аспауы, ал пайдалану нұсқалары - 50 аспауы тиіс. Керісінше жағдайда модель өз көрнекілігін жоғалтады, және де өзі басқа бірнеше диаграммаларды басуы мүмкін.

Пайдалану нұсқалары диаграммасын әзірлеу үшін әрекеттердің кейбір бірізділігі ұсынылады:

1) басты немесе алғашқы және қосалқы актерларды анықтау;
2) бас актерлардың жүйеге қатысты мақсаттарын анықтау;
3) жүйеге қойылатын функционалды талаптарды сипаттайтын негізгі пайдалану нұсқаларын қалыптастыру;
4) оларды тәуекелдің азаюын сату арқылы пайдалану нұсқаларын жеңілдету;



5.3-сурет. Пайдалану нұсқалары диаграммаларында ескертпелер мысалы

5) таңдап алынған пайдалану нұсқасының орындалу қатысушыларын, мүдделерін, алғышарттары мен қорытындыларын белгілеу;
6) таңдап алынған пайдалану нұсқасын іске асырудың сәтті сценарийін жазу;
7) пайдалану нұсқасының сценарийін орындауда табысқа жетпеу немесе ерекшеліктерді анықтау;
8) барлық ескертулер үшін сценарий жазу;
9) *uses/include* стереотипі бар өзара байланысты пайдаланудың жалпы нұсқаларын бөлу және көрсету;
10) алып тастау үшін пайдалану нұсқаларын белгілеп көрсету және *extends* стереотипі бар өзара байланысты көрсету;
11) диаграмманы пайдалану нұсқалары мен актерларды қайталаудың болмауына тексеру.

5.4. ҚЫЗМЕТ ДИАГРАММАЛАРЫ

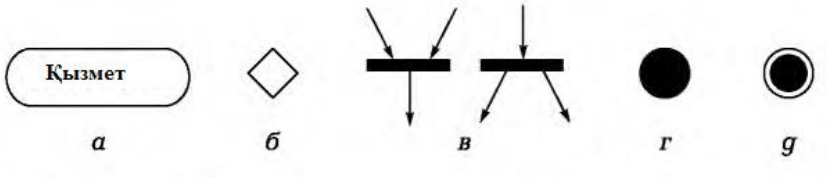
Егер пайдалану нұсқаларының диаграммасы бағдарламалық қамсыздандырудың функционалдылығына «жоғарыдан көрінісін» беретін болса, онда қызмет диаграммасы, керісінше, пайдаланудың жеке нұсқасы мен оның сценарийін толық көрнекі көрсетуге мүмкіндік береді.

Бұл жағдайда *қызмет* деп қолмен немесе автоматтандыру құралдарының көмегімен орындау қажет болатын міндетті (операцияны) түсінеді. Әрбір пайдалану нұсқасына міндеттердің өз бірізділігі сәйкес келеді.

Теориялық қырынан алғанда ***қызмет диаграммасы*** - талдау жасалатын пайдалану нұсқасын іске асыратын алгоритмнің жалпы көрінісі.

Қызмет диаграммасында қызмет бұрыштары бүктелген тік төртбұрыштар болып белгіленеді (5.4-сурет, *а*).

Қызмет диаграммалары баламалы және параллель процестерді сипаттауға мүмкіндік береді. Баламалы процесті шартты атау үшін (5.4-сурет, *б*) ромб қолданылады, ал «иә», «жоқ» баламалары - тиісті шығыстармен қатар қолданылады . Осы блоктың көмегімен циклды процесті құруға болады.



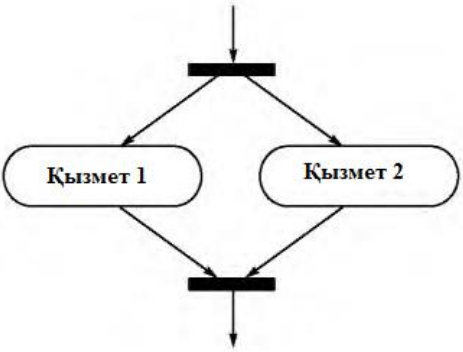
5.4-сурет. Қызмет диаграммасының шартты белгілері:
А - қызмет; б - таңдау; в – синхронизациялау сызғышы; г – басы; д – соңы

Қызметтерді актив етудің көптілігі «*» символымен шартты белгіленеді, олар қызмет активі нұсқарымен қатар орналасқан, және қажет болғанда «әрбір жол үшін» түрлі жазуын анықтайды.

Параллель процестерді шартты белгілеу үшін синхронизациялау процестерін (5.4-сурет, *в*) қолданады, мұнда синхронизациялау шартын оны диаграммада көрсете отырып анықтауға болады (5.5-сурет).

Енді қызмет диаграммасын салу үлгісін қарастырамыз.

Алдыңғы мысалда пайдалану нұсқаларын сипаттауды нақтылау үшін «Көтерме сауда қоймасы» автоматтандырылған ақпараттық жүйесінде мәтіндік сценарий әзірленген болатын. Бұл сценарий диаграмманы толықтырады, жүйемен және актерлармен орындалатын жеке қызметтердің мазмұнын ашады. Алайда пайдалану нұсқаларының сипаттамасының орнына немесе оларға қосымша ретінде қызметтер диаграммасын қолдануға болады. Қызмет диаграммасы талап етілген қажеттілік дәрежесімен пайдалану нұсқасын көрнекі етіп көрсетуге мүмкіндік береді. Пайдалану нұсқаларын анықтап алудың маңызы бар, олардың қысқаша сипаттамасы шешілетін проблемалардың мәнін түсіну үшін жеткілікті емес.



5.5-сурет. Процестердің параллельдігі көрсетілген қызметтер диаграммасының мысалы

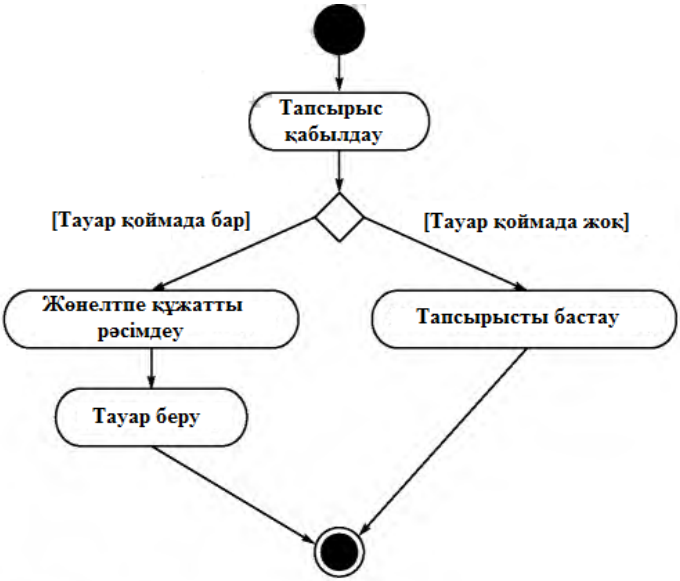
Әзірленетін модель шеңберінде «Тауар сату» және «Тауарды жеткізу» пайдалану нұсқаларын іске асыру үшін қызмет диаграммаларын құрамыз. Төменде 5.6-суретте тауарларды жеткізген кезде қызметтердің бірізділігін көрсету үшін қызметтер диаграммасының нұсқасы берілген.

Алайда, қызмет диаграммасы пайдалану нұсқасын түрді толықтыру қажеттілігімен көрнекі көрсетуге мүмкіндік береді және «Тауар жеткізу» пайдалану нұсқалары үшін қызмет диаграммасын келесі түрде көрсетуге болады (5.7-сурет).

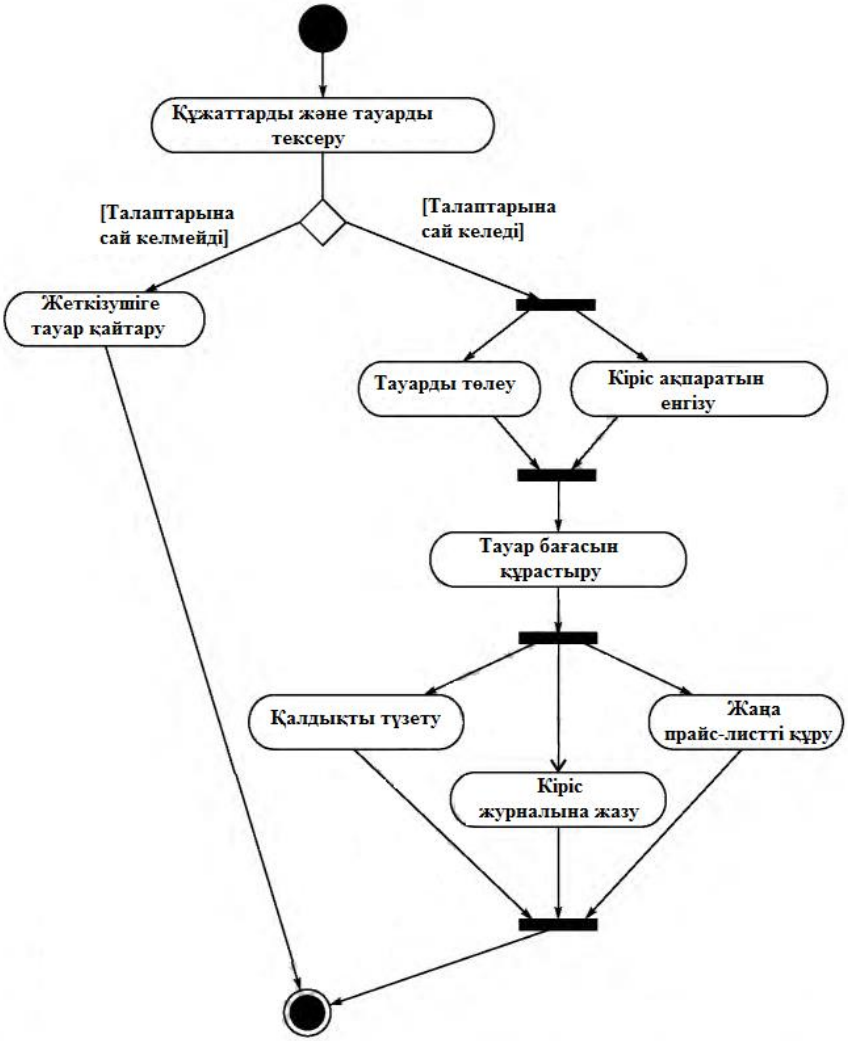
«Тауар сату» пайдалану нұсқасын талдай отырып, «Тауар сату» пайдалану нұсқасы үшін қызмет диаграммасын құрамыз (5.8-сурет).

Жүйенің толық моделі бірнеше қызметтер диаграммасынан тұрады, олардың әрқайсысы маңызды пайдалану нұсқаларын іске асырудың бірізділігін (оқиғалардың типтік барысы және барлық ерекшеліктер), не болмаса кластардың таптаурын емес операцияларын сипаттайды.

Стандартты сипаттау форматынан басқа, UML «жүзу жолы» бар нұсқаны ұсынады. Бұл формат қызмет ететін бәрнеше тұлғалар қатысатын пайдалану нұсқасында жағдайды сипаттау үшін ыңғайлы.



5.6. - сурет. Қызмет диаграммасының мысалы

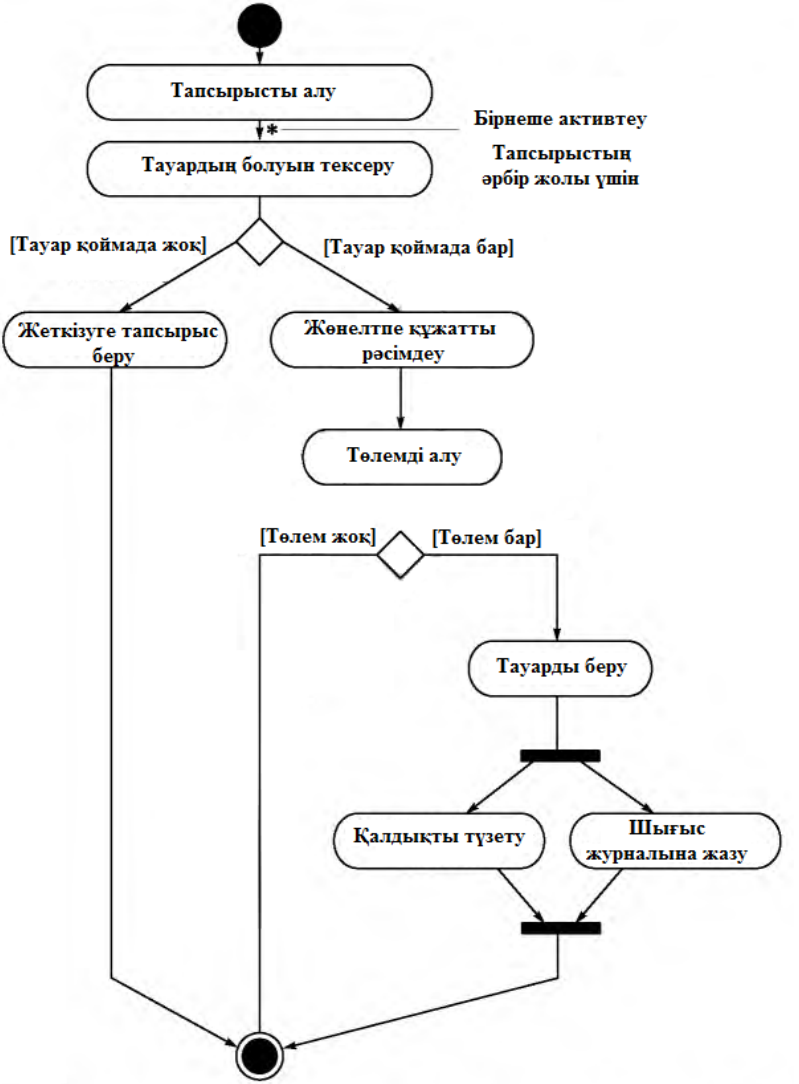


5.7-сурет. «Тауар жеткізу» пайдалану нұсқасына арналған қызмет диаграммасы

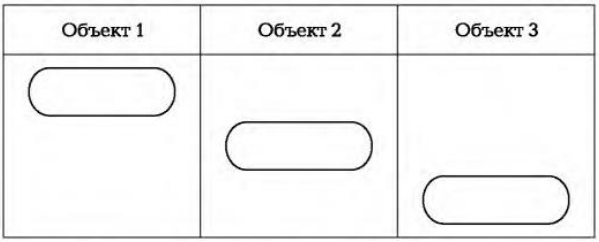
Бұл ретте қызмет диаграммасында барлық **қызмет** жағдайлары бір-бірімен тік сызықтармен бөлінетін жеке топтарға бөлінеді (5.9-сурет). Осылайша, мысалы, жолдарды қолдану кәсіпорындардың бөлімшелерінің қызметін сипаттап тізуге мүмкіндік бере отырып, бизнес-процестерді көрнекі көрсету үшін қосымша мүмкіндіктерін

ашады (5.10-сурет).

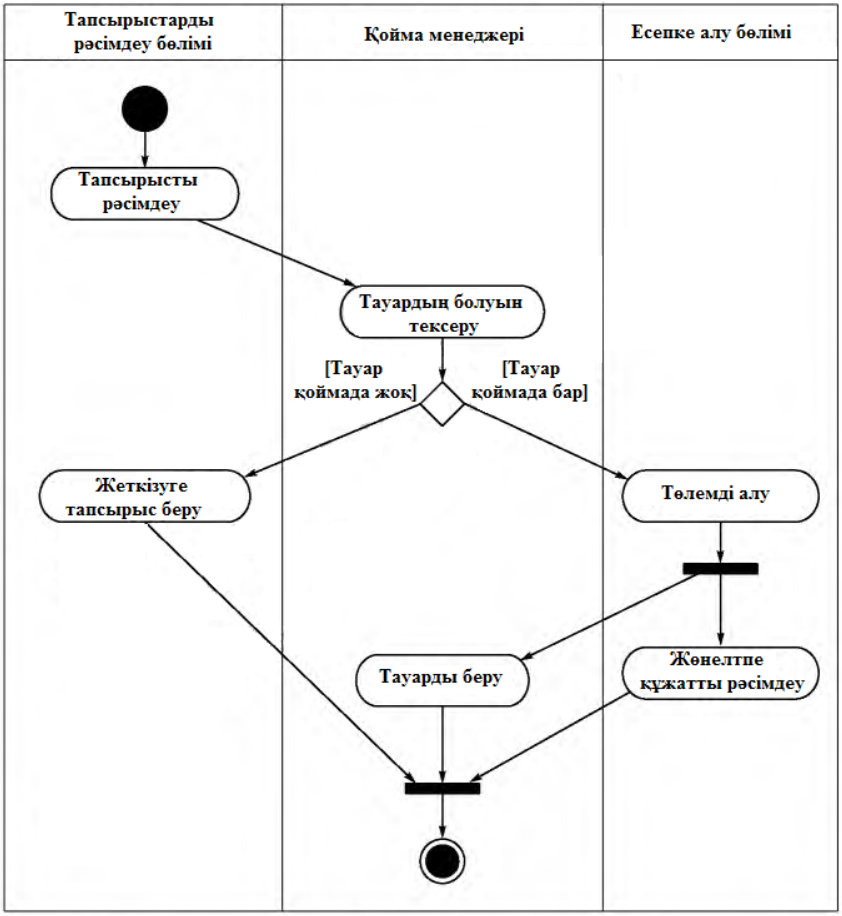
Үлгілік жоба жағдайында қызметтерді іске асыру тетіктерінің көбі қолданыстағы жүйелерге немесе жүйе-прототиптерді әзірлеудің алдыңғы тәжірибесіне талдау негізінде алдын-ала белгілі болуы мүмкін. Типтік шешімдерді пайдалану әзірлеу уақытын біраз қысқартуға және жобаны іске асырғанда мүмкін қателерді болдырмауға мүмкіндік береді.



5.8-сурет. «Тауар сату» пайдалану нұсқасына арналған қызметтер диаграммасы



5.9-сурет. Жолдары бар қызмет диаграммасының нұсқасы



5.10-сурет. «Тапсырысты рәсімдеу» пайдалану нұсқасы үшін жолдары бар қызмет диаграммасын қолдану мысалы

5.5. БІРІЗДІЛІК ДИАГРАММАЛАРЫ

Бірізділік диаграммалары өзара байланыс топтарына (interaction diagrams) - өзара байланысты нысандар топтарының әрекеттерін сипаттайтын модельдерге жатады. Негізінде, өзара байланыс диаграммасы тек бір пайдалану нұсқасының шеңберінде нысандардың әрекетін қамтиды. Мұндай диаграммада олар өзара алмасатын нысандар мен хабарламалар қатарын көрсетеді.

Қарастырылған қызмет диаграммалары жүйенің әрекет динамикасының сипаттамасы үшін қолданылады, оларда уақыттың қатысы жоқ. Алайда, әрекеттің уақытша қыры нысандардың өзара байланысын сипаттайтын (мысалы, нақты уақыт жүйелері үшін) синхронды процестерді модельдеу кезінде біршама мәнге ие болуы мүмкін. Әсіресе мұндай мақсат үшін UML тілінде бірізділік диаграммалары қолданылады.

Жүйе бірізділігінің диаграммасы - графикалық модель, уақытта нысандардың өзара байланысы динамикасын көрсететін пайдалану нұсқасының белгілі бір сценарий үшін графикалық модель.

Жүйенің бірізділігі диаграммасын құру үшін қажет:

- әрбір әрекет етуші тұлғаны (нысанды) сәйкестендіру және ол үшін өмір тізбесін бейнелеу керек;
- пайдалану нұсқасының сипаттамасынан көптеген жүйелік оқиғаларды және олардың бірізділігін анықтау;
- әрекет ететін тұлғалар мен жүйенің тізбектерінің арасындағы соңында нұсқары бар сызықтар түріндегі жүйелік оқиғаларды көрсетуге болады, сонымен қатар оқиғалардың аты мен берілетін мәндерінің тізімін көрсету.

Бірізділік диаграммасында өзара байланыста тікелей қатысатын нысандар ғана бейнеленеді және басқа нысандармен мүмкін статикалық ұқсастықтар көрсетілмейді.

Бірізділік диаграмма үшін тірек сәті нысандардың өзара байланыс динамикасының уақыты болып табылады. Бұл ретте бірізділік диаграммасы екі өлшемді болады. Біреуі - солдан оңға қарай тік сызық түрінде, олардың әрқайсысы өзара әрекетке қатысатын жеке нысанының сызықтарын көрсетеді.

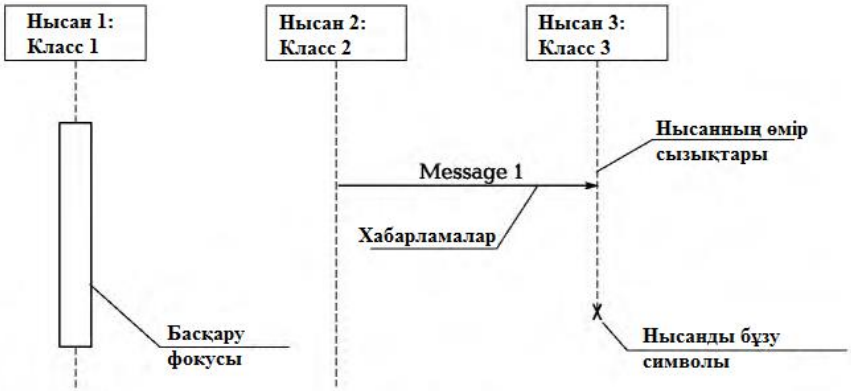
Графикалық жағынан әрбір нысаны тік төртбұрыш болып бейнеленеді және өз өмірінің сызығының жоғарғы жағында орналасады. Тік төртбұрыштың ішіне нысанының аты мен кластың аты жазылады, олар қос нүктемен бөлінеді. Бұл ретте барлық жазу астынан сызылады, бұл кластың өз данасын көрсететін нысанының белгісі болып табылады (5.11-сурет).

Нысанның аты бірізділік диаграммасында болмаған жағдайда

алынбайды. Бұл жағдайда тек класс аты ғана көрсетіледі, ал нысанның өзі жасырынды болып саналады.

Диаграммада сол жақтан шеткі нысан бейнеленеді, ол өзара әрекеттесудің бастамасы болып табылады (1 нысан 5.11-суретте). Оң жағынан біріншімен өзара әрекеттесетін басқа нысан бейнеленеді. Осылайша, барлық нысандар бірізділік диаграммасында кейбір тәртіпті құрады, ол бір-бірімен өзара әрекеттескенде осы нысандардың актив белсенділігін анықтайды.

Бірізділік диаграммасының екінші өлшемі - жоғарыдан төменге бағытталған уақытша тік ось. Уақыттың алғашқы сәті диаграмманың ең жоғарғы бөлігі сәйкес келеді. Бұл ретте нысандардың өзара әрекеті бір нысаннан басқасына берілетін хабарламалар арқылы іске асады. Хабарламалар көлденең нұсқар түріндегі хабарламалар атымен бейнеленеді және уақыттағы өздерінің пайда болу тәртібін құрады. Басқаша айтқанда, бірізділік диаграммасында орналасқан хабарламалар төменде орналасқандардан бұрын бастама алады. Бұл ретте уақыт осіне масштаб көрсетілмейді, себебі бірізділік диаграммасы тек «бұрын-кейін» үлгісіндегі өзара әрекеттесудің уақытша қарапайымдылығын модельдейді.



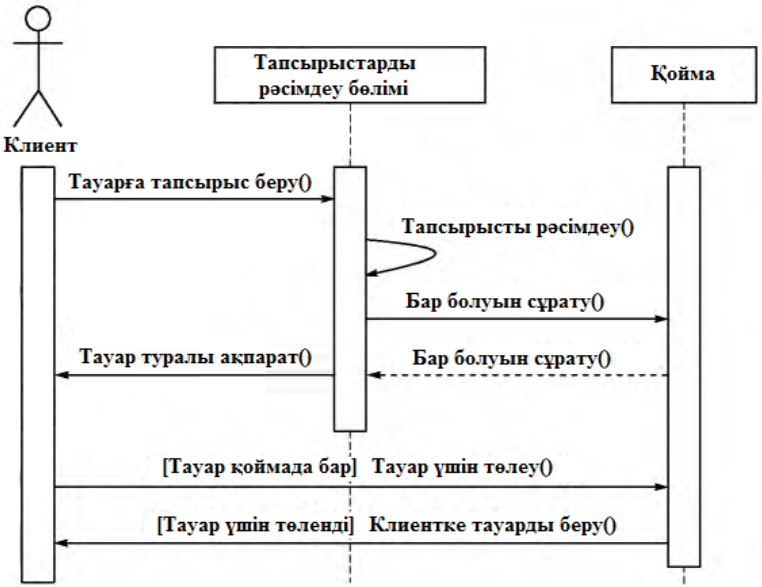
5.11-сурет. Бірізділік диаграммасының түрлі графикалық примитивтері

Нысанның өмір сызығы уақыт кезеңін шартты белгілеу үшін қызмет етеді, оның бойында нысан жүйеде болады және салдарында оның барлық өзара әрекет етуіне басымдылықпен қатыса алады. Өмір сызығының диаграммасында үзік сызықты тік сызықтармен жалғыз нысанымен бейнеленеді. Егер нысаны жүйеде тұрақты болса, онда оның өмір сызықтары бірізділік диаграммасының бүкіл жазықтығында жалғасатын болады.

Бірізділік диаграммасын құруды бүкіл жиынтықтан модельденетін өзара байланысқа қатысатын нысандар, кластар барлық жиынтықтан бөліп алуды мақсатқа сай басталады. Бұдан кейін барлық нысандар диаграммаға хабарламаларды бастаудың кейбір тәртібін сақтаумен енгізіледі. Қандай нысандар тұрақты, ал қандайы уақытша - тек талап етілетін әрекеттерді орындау кезеңіне орнату қажет. Нысандар анықталған кезде хабарламалардың сипаттамасына келетін болады. Бұл ретте жүйеде алатын хабарламалардың рөлін ескеру керек.

Мысал ретінде «Көтерме сауда қоймасы» ақпараттық жүйесінің «Тауар сату» пайдалану нұсқасын іске асыруға арналған бірізділік диаграммасын құрамыз (5.12-сурет).

Бұл диаграмма екі нысаннан және бір актердан тұрады. Нысандар үнемі белсенді болып табылмайды, ол тиісті басқару фокусының көмегімен көрсетілген. Хабарламалар аты ретінде тиісті кластарда сипаттап тізілген операциялар аттары көрсетілген.



5.12-сурет. Тауар сату операциясын модельдеуге арналған бірізділік диаграммасының нұсқасы

Кейбір хабарламалардың ұсыныс-шарттары квадратты жақшалар ішіндегі қарапайым мәтінмен жазылған. Бұл шарттарды тиісті пайдалану нұсқасының ерекше сценарийлерін орындау мен сату процесін тармақтау деңгейін береді, алайда басқа пайдалану нұсқалары бұл диаграммада көрсетілмеген.

5.6. КҮЙЛЕРІНІҢ ДИАГРАММАЛАРЫ

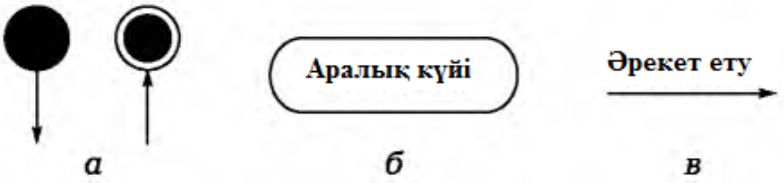
Күйлерінің диаграммалары әзірленіп отырған бағдарламалық жүйенің басқару әрекеттерін алған кездегі әрекеттерін көрсетеді.

Басқарушы өзара әрекеттер немесе *сигналдар* деп бұл жағдайда шеттен жүйе алатын басқару ақпаратын түсінеді, мысалы, басқарушы өзара әрекеттер деп пайдаланушы командалар мен компьютерлік жүйелерге қосылған датчиктердің командаларын санайды. Мұндай басқару әсерін ала отырып, әзірленетін жүйе белгілі бір әрекеттерді орындауы тиіс, ал содан немесе сол күйінде қалуы керек немесе басқа күйге көшуі тиіс, ол жүйедегі кейбір өзгерістерді белгілеп алуы керек.

Бұл диаграмманың басты тағайындалу мақсаты - өмірлік циклы бойы модельдің элементтерінің жүріс-тұрысын сипаттайтын жиынтықтағы өткелдер мен күйлердің ықтимал бірізділігін сипаттау. Күйлерінің диаграммасы кейбір нақты оқиғаларды қабылдауға оның реакциясының сипаттамалары негізінде мәнінің динамикалық қозғалысын көрсетеді. Басқа жүйелердің немесе пайдаланушылардың сыртқы әрекеттеріне жауап беретін жүйелерді кейде реактивті деп атайды. Егер мұндай әрекеттер еркін уақыт мезеттерінде бастама алатын болса, онда модельдің асинхронды әрекеті туралы айтады.

Күйлерінің диаграммалары көбінесе кластардың (нысандардың) жеке даналарының әрекеттерін сипаттау үшін жиі қолданса да, олар да модельдердің басқа компоненттерінің функционалдылық сипаттамасы үшін қолданылуы мүмкін, мысалға пайдалану нұсқалары, әрекет етуші тұлғалар, қосалқы жүйелер, операциялар мен әдістер.

Күйлерінің өту диаграммаларын құру үшін соңғы автоматтардың теориясына сәйкес негізгі күйлерін, басқарушы әсерлерін (немесе өту шарттарын), орындалатын әрекеттерді және әзірленетін бағдарламалық қамсыздандырудың ықтимал өтулерін анықтайды. Өту диаграммасын құру кезінде қолданылатын шартты белгілер 5.13-суретте көрсетілген.



5.13-сурет. Күйлердің өту диаграммасының шартты белгілері: а – бастапқы және соңғы күйі; б – аралық күйі; в – өту

Күй диаграммасы мағынасына қарай кейбір автомат болатын арнайы түрдегі сызбалы бейне болып табылады. Автоматты UML мәнмәтінінде түсіну автоматтар теориясына негізделген арнайы өзіне тән мәні бар. Бұл сызбалы бейненің шыңы тиісті графикалық символдармен бейнеленетін автоматтың өзге элемент типтері (жалған күйі) мен күйі болып табылады. Сызбалы бейненің иіндері бір күйден бір күйге өтуді көрсетуге арналған. Өткелдер мен күйлерді көзбен көрсетуге арналған қарапайым мысалы автоматтар формализмі негізінде компьютер сияқты, техникалық құрылғының дұрыс жұмыс істеуіне байланысты жоғарыда қарастырылған жағдай болады. Бұл жағдайда екі ең жалпы күйі «бұзылған» және екі өткел «істен шығу» мен «жөндеу» қарастыруға шығарылады. Графикалық бұл ақпарат төменде көрсетілген компьютер күйінің диаграммасы түрінде берілуі мүмкін (5.14-сурет).
Автомат формализміне кіретін негізгі ұғым күйі мен өткелдері болып табылады. Олардың арасындағы басты айырмашылық жүйенің жеке күйінде көп уақыттан артық болуында, ол басқа күйге өтуіне жұмсалады. Өту уақыты шегінде бір күйден екінші күйге өту нөлге тең екендігі болжанады (егер қосымша ештеңе айтылмаған болса). Басқаша айтсақ, нысанның бір күйден бір күйге өтуі лезде болады.

5.14-сурет. Компьютер үлгісіндегі техникалық құрылғыға арналған күй диаграммасының қарапайым үлгісі

Нақты күйі диаграммасының семантикасын түсіну үшін модельденетін мағынаның әрекет ету ерекшеліктерін біліп қана қоймай, сонымен қатар автоматтар теориясы жөніндегі жалпы мәліметтерді де білу қажет.

Пайдаланушы интерфейсі дамыған интерактивті бағдарламалық қамсыздандыру үшін негізгі басқарушы әрекет - пайдаланушының командалары, нақты уақытта бағдарламалық қамсыздандыру үшін - датчиктерден және/немесе өндірістік процесс операторларынан сигналдар. Интерактивті бағдарламалық қамсыздандыру үшін түрлі үлгідегі командаларды алу тән, ал нақты уақыттағы бағдарламалық қамсыздандыру үшін - бір үлгідегі сигналдар, не болмаса көптеген датчиктерден алу, не болмаса кейінірек өңдеуді талап ететін. Бағдарламалық қамсыздандырудың бұл үлгілері үшін кезекті басқарушы әрекетті алғанға дейін жұмысты жүйе тоқтата тұратын күту күйінің болуы болып табылады. Интерактивті жүйелердің айырмашылығында нақты уақыт жүйелері үшін әдетте желідегі жұмысқа бағдарланған бағдарламалық қамсыздандыру жатады. Мұндай шектеу уақытта жүйенің әрекеттерін қосымша зерттеуді орындау қажет болады.

Бағдарламалық қамсыздандыруға, күйлердің өту диаграммасын құру арқылы олардың ерекшеліктерін анықтап алу талап етіледі және бұл ретте әдетте сервер мен клиенттің әрекет моделі жеке құрады, ол басқарушы әрекеттер түрінде берілетін хабарламаларды береді.

Күйлер диаграммасын құру үлгісін келтіреміз.

Ұсынылатын диаграммада (5.15-сурет) күйлер мен өткелдердің мүмкін бірізділігі сипатталады, олар жиынтығында «Көтерма сауда қоймасы» автоматтандырылған ақпараттық жүйесінің «Тапсырыс» нысанының әрекеттерін сипаттайды (келіп түсу, өңдеу, жеткізуді қалыптастыру). Мұнда белгілі бір күйде «Тапсырыс» нысаны орындайтын функциялар көрсетіледі.

Әрекеттің таңбалары келесі синтаксисте болады: орындау/< әрекет > (мысалы, орындау/жолды тексеру).

Өткелдердің таңбалары бар, олардың синтаксисі міндетті емес үш бөліктен тұрады: <Оқиға> <[Шарт]> </Әрекет> (мысалы, [барлық жолдар алынған жоқ]/келесі жолды алу).

5.15-сурет. «Құжат-Тапсырыс» нысанының күй диаграммасы

Күйлердің диаграммаларын әр түрлі пайдалану нұсқаларында кейбір нысандардың әрекеттерін сипаттау үшін қолданған жақсы. Олар өзара байланысты нысандардың бірқатарының әрекетін сипаттау үшін жарамды бола бермейді. Күй диаграммаларын мағынасына қарай басқа құралдармен үйлестіру тиімді.

5.7. КЛАСТАР ДИАГРАММАСЫ

Кластар диаграммасы жүйенің ішкі құрылысын түсіндіреді, оның компоненттері (кластар, нысандар) арқылы құрылатын бағдарламалық жүйені, байланысы мен олардың арасындағы өзара байланысты сипаттайды. Кластар диаграммасында кластар арасындағы байланыста қатарласып келетін кластардың атрибуттары, класс операциялары мен шектеулер бейнеленеді.

UML нотациясы қосымша ақпаратты (дерексіз операциялар мен кластар, стереотиптер, жеке және жалпы әдістер, егжей-тегжейлі интерфейстер, параметрлендірілген кластар) бейнелеу үшін ауқымды мүмкіндіктерін береді.

Класс диаграммасы нысанға бағытталған бағдарламалау кластары терминологиясында жүйе моделінің статикалық құрылымын көрсетуге қызмет етеді. Кластар диаграммасы көбінесе тақырыптық саланың жеке

мәндері арасындағы түрлі өзара байланысты, мысалға нысаны мен қосалқы жүйелер арасындағы байланысты көрсете алады, сондай-ақ олардың ішкі құрылымы мен қарым-қатынастарының үлгілерін сипаттайды.

UML кластар диаграммасының олардың егжей-тегжейлі дәрежесіне қарай үш деңгейін қолдану болжанады.

1. *Тұжырымды деңгей,* мұнда класс диаграммалары бұл жағдайда мәнмәтінді деп аталып, тақырыптық саланың негізгі ұғымдарының арасында байланысты көрсетеді. Бұл ұғымдар, әрине, оларды іске асыратын кластарға тиісті болуы керек, алайда, мұндай сәйкестілік көбінесе болмайды. Шынымен де, тұжырымды модель біршама әлсіз қатынасы болады немесе әзірленіп жатқан бағдарламалық қамсыздандыруға ешбір қатысы жоқ, сол себепті оны бағдарламалау тілінен жеке қарастыруға болады.

2. *Сипаттамалардың деңгейі,* мұнда класс диаграммасы тақырыптық сала кластарының интерфейстерін көрсетеді, яғни осы класс нысандарының байланыстары. Бұл жағдайда біз бағдарламалық жүйені қарастыруға өтеміз, бұл ретте оның іске асырылуын емес, интерфейстерін ғана қарастырамыз. Нысанға бағытталған әзірлеу интерфейс пен іске асыру арасындағы біршама мәнді көрсетеді. Нысанға бағытталған тиімді бағдарламалаудың тірек факторы класс интерфейсі болып табылады, оның іске асырылуы емес.

3. *Іске асыру деңгейі,* мұнда кластар диаграммасы нақты кластардың өрістері мен әдістерін тікелей көрсетеді. Осы деңгейдегі кластар диаграммасын қолдану шамамен тыс жиі кездеседі, алайда көптеген жағдайларда сипаттамасы жағынан алғанда талдаушы үшін біршама артық болып табылады.

Шамамен осы үш түрлі модель, олардың арасындағы байланыс бір мағыналы емес. Егер тұжырымды модель тақырыптық саланың класс сияқты кейбір ұғымдарын анықтайтын болса, онда бұл ұғымды іске асыру үшін жеке класс қолданылатын болады. Алайда, барлық үш модельде де бірыңғай нотациясын қолдануға мүмкіндік беретін нысандар (класстар) түрі мен олардың статикалық қатынастарын көрсетеді.

Атап өтілген модельдердің әрқайсысы бағдарламалық қамсыздандыруды әзірлеудің нақты сатысында қолданылады:

- тұжырымдық модель - талдау сатысында (тақырыптық сала ұғымдарымен, осы ұғымдардың атрибуттары мен олардың арасындағы байланыс);
- сипаттама деңгейінің класс диаграммалары - жобалау сатысында;
- іске асыру деңгейі кластар диаграммасы - іске асыру сатысында.

Диаграмманы құру кезінде жалғыз қырды таңдау керек. Диаграмманы оқыған кезде оның қандай қырға сәйкес құрылғандығын

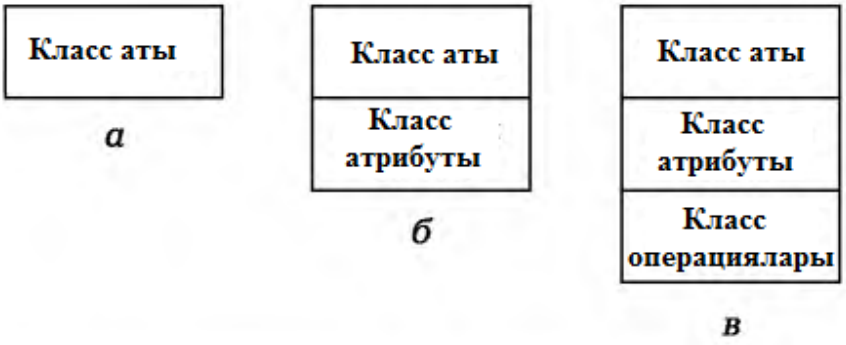
анықтап алу қажет.

Модельде негізгі ұғым *класс* сәйкес қойылады. Класс бұл ретте тақырыптық саланың кейбір топтарының жалпы белгілерінің жиынтығы ретінде қабылдау дәстүрлі. Осы анықтамаға сәйкес кластар диаграммасында әрбір класқа жалпы белгілері класты белгілейтін нысандар топтарына сәйкес келеді. Осылай, класс «Студент» жоғары оқу орындарында оқитын адамдардың топтарын жалпы белгілерімен біріктіріледі. Класс немесе нысан данасы (мысалы, И.И. Иванов) өзінің класының белгілер жиынтығының барлығына ие және класта белгіленбей қалған өз белгілеріне ие болуы мүмкін. Осылай, мысалы И.И. Иванов студент болып қана қоймай, ол спортшы, музыкант және т.б. осылайша, мұндай жеке белгі студенттің атын сәйкестендіруші болып табылады.

Қауымдастықтар класс даналарының арасындағы байланысты білдіреді. Тұжырымдық жағынан алғанда қауымдастықтар кластар арасындағы тұжырымдық байланыстарды білдіреді.

Диаграммаларда класс тік төртбұрыш түрінде бейнеленеді (5.16-сурет, *а*). Класс аты тік төртбұрыштың оң жақ жоғары секциясында көрсетіледі. Қажет болғанда класс сипаттамасын көрсетуге болады (5.16-сурет, *б*), сондай-ақ класс операциясы (5.16-сурет, *в*) көрсетіледі.

Кейбір жалпы қасиеттерге ие кластарды топтастыру үшін пакеттер қолданылады. Пакет - топтағы модельдің элементтерін ұйымдастыруға арналған жалпы механизм. Модельдің әрбір элементі тек бір пакетке ғана кіре алады.



5.16-сурет. Кластардың концептуалды диаграммасында кластарды шартты көрсету:
а – сипаттамаларын анықтаусыз; б – атрибуттарды анықтау арқылы; в – операцияларды көрсету арқылы

Класс аты класс диаграммаларының кейбір жиынтығымен сипатталатын (мүмкін, бір диаграммасымен) пакет шеңберінде бірегей болуы керек.

Класс аттары нысанға бағытталған талдау мен жобалау кезінде тақырыптық саланың сөздігін құрады. Класс аттарының үлгілері зат есімдер, мысалы «Қызметкер», «Басшы», «Клиент», «Сатушы», «Менеджер» және модельденетін тақырыптық салаға тікелей қатысы бар және жобаланатын жүйенің функционалды тағайындалуына қатысты көптеген басқа зат есімдер болады.

Кластың даналары немесе нысандары болмайды. Бұл жағдайда ол *дерексіз* деп аталады.

Кейбір жағдайларда қандай пакетке қандай класс жататындығын көрсету қажет. Бұл мақсат үшін арнайы бөлгі - қатарланған қос нүкте «: : » қолданылады. Класс атының жол синтаксисі бұл жағдайда мына түрде болады <Пакеттің_аты>::<Кластың _аты>. Басқаша айтсақ, класс атының алдында оны жатқызу қажет пакеттің атын көрсету анық қажет. Мысалы, «Банк» атымен пакет анықталса, онда «Шот» класын мына түрде жазамыз: «Банк::Шот».

Атрибуттың аты мәтіннің жолын білдіреді, ол тиісті атрибуттың идентификаторы ретінде қолданылады және сол себепті осы кластың шеңберінде бірегей болуы керек. Атрибуттың аты атрибутты синтаксистік атаудың жалғы міндетті элементі болып табылады.

Атрибуттың дүркінділігі жеке кластың құрамына кіретін осы типтің нақты атрибуттарының жалпы санын сипаттайды. Жалпы жағдайда дүркінділік тиісті атрибуттың атынан кейін квадрат жақшалардың ішіне мәтін жолдары түрінде жазылады:

[төменгі_шекара$_1$ .. жоғарғы_шекара$_1$, төменгі_шекара$_2$ . .
жоғарғы_шекара$_2$, . . . , төменгі_шекара$_к$ ..жоғарғы_шекара$_к$],

мұндағы төменгі шекара мен жоғарғы шекара оң бүтін сандар болып табылады, олардың әрбір сыңары бүтін сандардың тұйық аралығын белгілеуге арналған, онда төменгі (жоғарғы) шекарасы төменгі шекараның (жоғарғы шекараның) мәніне тең.

Жалпы алғанда шартты атау тиісті аралықтарды теориялық-көбейту біріктіруге сәйкес келуі тиіс. Жоғарғы шекара ретінде туынды оң бүтін санды білдіретін «*» арнайы символды қолдана алады. Басқа символдармен бұл тиісті атрибуттың дүркінділігінің мәнін жоғарыдан беруді білдіреді.

Мысал ретінде атрибуттардың дүркінділіктерін берудің келесі нұсқаларын көрсетуге болады:

- [0..1] атрибут дүркінділігі 0 немесе 1 мәндерін алуы мүмкін екендігін білдіреді. Бұл ретте 0 берілген атрибут үшін мәннің жоқтығын білдіреді;
- [0..*] атрибут дүркінділігі 0 тең немесе одан үлкен кез келген оң бүтін мәнді алу керектігін білдіреді. Бұл дүркінділік [*] қарапайым символ түрінде қысқаша жазылуы мүмкін;
- [1..*] атрибут дүркінділігі 1 тең немесе одан үлкен кез келген оң бүтін мәнді алатындығын білдіреді;
- [1..5] атрибуттың дүркінділігі 1, 2, 3, 4, 5 сандарынан кез келген мәнді алатындығын білдіреді.

Егер атрибут дүркінділігі көрсетілмеген болса, онда келісім бойынша 1..1 тең оның мәні алында, дәлірек 1.

Атрибут типі өрнек түрінде болады, оның семантикасы тиісті модельдің сипаттама тізімі тілімен айқындалады. UML нотациясында атрибут типі кейде бағдарламалау тіліне байланысты анықталады, оны осы модельдің іске асырылуы үшін қолдану болжанады. Ең қарапайым жағдайда атрибут типі мәтін жолдарымен көрсетіледі, ол пакет немесе қарастырылып отырған класс жататын модель шеңберінде мәнді мағынасы бар.

Мысалы, атрибут қызметкердің_есімі [1..2]:String - мұнда қызметкердің есімі атрибут аты болып табылады, ол есімі туралы ақпарат беру үшін, сондай-ақ нақты қызметкердің әкесінің аты туралы ақпарат беруге қызмет етеді.

Атрибут типі String (Жол) есімнің жеке мәні бір немесе екі сөзден тұратын мәтіннің жолын көрсетеді (мысалы, «Иван» немесе «Сергей Владимирович»). Бағдарламалау тілдерінің көбінде String деректер типі бар, бұл ағылшын тілді тиісті терминнің көптеген бағдарламалаушыларында жаңсақтық туғызбайды. Алайда, UML тілінде барлық терминдер ағылшын тілінде көрсетілген, атрибут типі ретінде пайдалану бұл жағдайда алынып тасталмайды және тек ыңғайлы болуымен анықталады.

Класс диаграммаларын егжей-тегжей дәрежесіне байланысты атрибуттарды шартты көрсету аты, типі, көріну сипаттамасынан басқа келесі түрде келісім бойынша мәнін қамтуы мүмкін:

<көріну белгісі> <аты>:<типі>=<келісім бойынша мәні>.

Көріну белгісі үш мүмкін мәннің бірін қабылдауы мүмкін және тиісінше арнайы символдардың көмегімен көрсетіледі.

«+» символ көпшілікке қолжетімді көріну түрінің саласымен атрибутты білдіреді (public). Атрибут осы көріну аясынан қолжетімді немесе диаграмма анықталған өзге кластан көрінеді.

«#» символы қорғалған түрдегі көріну типінің облысы бар атрибутты білдіреді (protected). Бұл облыстағы атрибут қолжетімді емес немесе осы кластың қосалқы кластарынан басқа барлық кластар үшін көрінбейді.

Және «-» белгісі жабық түрдегі көріну аясынан атрибутты көрсетеді (private). Атрибут бұл көріну аясынан қолжетімді емес немесе барлық класттар үшін көрінбейді.

Көріну белгісі төмен түсірілуі мүмкін. Бұл жағдайда оның болмауы атрибуттың көрінуі көрсетілмейтіндігін білдіруі мүмкін. Шартты графикалық шартты атаулардың орнына тиісті тірек сөздерді жазуға болады: public, protected, private.

Атрибуттарды берген кезде қосымша екі синтаксистік конструкциялар қолданылуы мүмкін - бұл атрибут жолын сызып көрсету мен фигуралық жақшалардың ішіндегі түсініктемелік мәтін.

Атрибут жолының астын сызу тиісті атрибуттың оның типімен анықталатын атрибут мәндерінің кейбір аясынан көптеген мәндерді қабылдай алатындығын білдіреді. Бұл мәндерді бір типті жазбалардың немесе кластың әрбір нысанын жиынтықтап сипаттайтын массивтердің жинағы ретінде қарастыруға болады.

Тік төртбұрыштың жоғарғыдағы үшінші секциясына операциялар немесе класс әдістері жазылады.

Операция белгілі бір талап бойынша кластың әрбір данасы көрсететін кейбір сервисті білдіреді.

Операциялардың жиынтығы класс әрекеттерінің функционалдылық қырын сипаттайды. Класс операцияларын жазу UML тілінде белгілі бір синтаксистік ережелермен стандартталған және соларға бағынады.

Операцияның аты тиісті операторының идентификаторы ретінде қолданылатын мәтіннің жолын береді және сол себепті осы класс шеңберінде бірегей болуы керек. Атрибуттың аты операцияларды синтаксистік шартты белгілеудің жалғыз міндетті элементі болып табылады. Мұнда жобалаудың атрибуттары сияқты, операцияларын да қосымша сипаттімздеу мақсатқа сай болмақ.

Класс диаграммасында операцияның толық сипаттамасы UML келесі түрде көрсетілуі мүмкін:

<көріну белгісі> <аты>(<параметрлердің тізімі>): <қайтарылатын мәннің типі>

Көріну белгісі, класс атрибуттары жағдайында да үш мүмкін мәннің бірін қабылдауға болады және тиісінше, арнайы символдардың көмегімен («+», «#», «-») көрсетіледі. Операциялар үшін көріну белгісі төмен түсірілуі мүмкін. Шартты графикалық белгілердің орнына тиісті тірек сөздерді жазуға болады: public, protected, private.

Параметр түрі - тірек сөздердің бірі келісім бойынша in, out немесе inout мәндерімен in, егер параметр түрі көрсетілмеген болса. Параметрдің аты тиісті формалды параметрдің идентификаторы. Типтің көрінуі тиісті формалды параметр үшін қайтарылатын мәнінің сипаттама тізімін бағдарламалау нақты тіліне байланысты болады.

Келісім бойынша мәні жалпы жағдайда формалды параметр мәні үшін өрнекті білдіреді, оның синтаксисі нақты бағдарламалау тіліне байланысты және мұндағы шектеулерімен бағынады. Формалды параметрлердің тізімі мен қайтарылатын мәнінің типі көрсетілмейді.

Класс *жауапкершілігі* класс нысандарының негізгі функцияларын формалды емес атауларын атайды. Класс жауапкершілігі әдетте жобалаудың бастапқы сатыларында атрибуттар мен класс операциялары әлі анықталмаған болса. Бұл ақпаратты класс диаграммаларында кластың шартты бейнелерінің класс диаграммаларында көрсетіледі.

Атрибуттардың көбісі тақырыптық салаға, техникалық тапсырма талаптары мен оқиға ағынының сипаттамаларын талдау кезінде анықталады.

Егер әр түрлі пакеттегі кез келген екі класс арасында кейбір тәуелділік болса, онда осы екі пакет арасындағы байланыс та орын алады. Пакеттердің диаграммасы класс пакеттерінен және олардың арасындағы тәуелділіктен тұратын диаграмманы білдіреді. Пакеттер диаграммасы класс диаграммасының формасы болып табылады. Пакеттердің диаграммаларын жүйенің жалпы құрылымын басқаратын негізгі құрал деп санауға болады.

Кластардың ішкі құрылымы немесе құрылысынан басқа бағдарламалық жүйенің әзірлеу кезінде маңызды рөл алатын кластар арасындағы байланыс, оны класс диаграммасында да көрсетуге болады:

- *ұқсастық байланысы* - кластар арасындағы туынды өзара байланыстың болуы;

- *жалпылау байланысы* - біршама жалпы элементтер мен біршама жиі элементтер (түпкі және тармақталған) арасындағы байланыс;
- *композиция байланысы* - агрегация байланысының жеке жағдайы, мұнда бөліктері тұтастан жеке шыға алмайды және тұтасты жойғанда оның барлық құрамдас бөлшектері де жойылады;
- *агрегация байланысы* - кластардың бірі кейбір мәнін береді, ол кейбір мәнінің құрамдас бөлшегі ретінде қамтиды.

UML тілінің мәнмәтінінде арнайы класс бар, ол *интерфейс* деп аталады, оның тек операциялары бар және атрибуттары жоқ. Диаграммадағы интерфейстер басынан көрінетін, бірақ клиенттерден ішкі құрылымы жабық болып қала беретін модельдің элементтері үшін қызмет етеді.

Класс диаграммаларын әзірлеу процесі күрделі жүйелердің жобаларын әзірлеген кезде орталық орынды алады. Класты дұрыс таңдау білу мен олардың арасында өзара байланыс орнату тек жобалау процестерінің табысына ғана емес, бағдарлама өнімділігіне байланысты болады.

5.8. КОМПОНЕНТТЕР ДИАГРАММАЛАРЫ

Компоненттер диаграммалары жүйе моделінің физикалық деңгейде қалай көрінетіндігін көрсетеді. Диаграммада бағдарламалық қамсыздандырудың компоненттері мен олардың арасындағы байланыс бейнеленген. Бұл ретте компоненттердің екі түрін көрсетеді: орындалатын компоненттер мен код кітапханалары.

Әрбір модель класы бастапқы код компонентіне түрленеді. Құрғаннан кейін олар компоненттер диаграммасына бірден қосылады. Жеке компоненттердің арасында компиляция сатысында немесе бағдарламаны орындау сатысында тиісті байланыстарды бейнелейді.

Компоненттерді бір компоненттің талап етілген интерфейсі басқа компоненттің бар интерфейсімен жалғанған кездегі байланыс арқылы байланысады. Осылайша, «клиент-көз» қатынасы екі компонент арасында бейнеленіп көрсетеді.

Тәуелділік бір компоненттің басқа компонентке қажетті сервисін беретіндігін көрсетеді. Тәуелділік интерфейстен немесе клиент портынан импортталатын интерфейске нұсқармен белгіленеді. Компоненттер диаграммасы компоненттердің ішкі құрылымын көрсету үшін қолданылады, ол ұсынылатын және талап етілетін интерфейстер ішкі компоненттердің тиісті интерфейстеріне берілуі мүмкін. Делегация ішкі компоненттердің осы әрекеті ішкі іске асыру компонентінің сыртқы контракт байланысын көрсетеді.

Компоненттер диаграммасы жүйенің компиляциясына жауап беретін жоба қатысушыларымен қолданылады. Бұдан қай тәртіпте компоненттерді құрастыру керектігі көрінеді, сондай-ақ қандай орындалатын компоненттер жүйемен құрылатындығы көрінеді. Ол код генерациясы қайда басталса, сол жерге керек.

5.9. ОРНАЛАСТЫРУ ДИАГРАММАСЫ

Орналастыру диаграммасы жүйенің бағдарламалық және аппараттық компоненттері арасындағы нақты өзара байланысты көрсетеді. Ол таратылған жүйедегі объектілер мен компоненттердің жылжу бағытын көрсетуге арналған жақсы құрал болып табылады.

Әрбір торап орналастыру диаграммасында есептегіш құрылғының кейбір үлгісін, ал көбінесе - аппаратураның бөлігін білдіреді. Тораптар арасындағы байланысты коммуникациялық арналарды көрсетеді, олардың көмегімен жүйелік өзара байланыс жүзеге асырылады.

Жылжу диаграммасында компоненттер бағдарлама кодының нақты модулін көрсетеді. Негізінде, олар компоненттер диаграммасындағы компоненттерге сәйкес келеді. Осылайша, орналастыру диаграммасы жүйедегі әрбір компоненттің орындалуын көрсетеді.

Орналастыру диаграммасының практикасында тым жиі қолданылмайды. Көптеген әзірлеушілер осындай диаграммаларды қолданады, алайда, олар қарапайым формалды емес суреттерді береді. Басқа жағынан, әрбір жүйе өзінің физикалық сипаттамаларына ие, олар одан әрі орналастыру диаграммасының көмегімен бірінші кезекте шешу керек проблемаларды жақсы түсінуге қол жеткізу бойынша үлкен формализм дәрежесі қажет болады.

5.10. ДИАГРАММАЛАРДЫ ҚҰРУ КЕЗІНДЕГІ ҰСЫНЫСТАР

Диаграммаларды графикалық бейнелеу кезінде төменде сипатталған негізгі ұсыныстарды ұстанған дұрыс.

1. Әрбір диаграмма модельденетін тақырыптық саланың тиісті үзіндісінің аяқталған түсінігіне қызмет етеді. Диаграмманы әзірлеу процесінде берілген модельдің және диаграмманың мәнмәтіні жағынан алғанда барлық мәнін есепке алу қажет. Диаграммада осы немесе басқа элементтердің болмауы модельдің толық еместігі қызметі болады және оның келесі өңделуін талап етуі мүмкін.

2. Диаграммадағы барлық мәндер бір тұжырымдық деңгейде болуы тиіс. Бұл жерде бірдей элементтердің атаулары ғана емес, сонымен қатар жеке диаграммаларды бір-біріне түсініктің толыққанды болуына қол жеткізу үшін салып отыр. Жүйенің жеткілікті дәрежедегі модельдеріне қол жеткенде жеке диаграммаларды жүйелілікпен анықтау немесе егжей-тегжей стратегиясын ұстанған дұрыс.

3. Мәні туралы барлық ақпарат диаграммаларда анық көрсетілген. UML тілінде диаграммада кейбір символдар болмағанда олардың мәндерін келісім бойынша қолдану туралы сөз болып отыр (мысалы, атрибуттар мен класс операцияларының анық көрінбеген жағдайында), диаграммалардың барлық элементтерінің қасиеттерін анық көрсетуге ұмтылу қажет.

4. Диаграммада қарама-қайшы келетін ақпарат болмауы тиіс. Модельдің қарама-қайшылығы оны іске асырғанда және кейін практикада қолданғанда маңызды проблемалардың себебі болуы мүмкін. Мысалы, біріктіру немесе композиция байланыстарын бейнелегенде тұйық жолдардың болуы бағдарламалық кодтағы қателерге әкеледі, ол тиісті кластарды іске асыратын болады. Бірдей атаулары мен түрлі атрибут қасиеттері бар элементтердің болуы бір кеңістікте осындай бірегей емес түсіндіруге әкеледі және проблемалардың көзі болуы мүмкін.

5. Диаграммаларды мәтіндік ақпаратпен шамадан тыс толтырмау қажет. Модельді көзбен көргенде біршама тиімді болады деп есептеуге болады, егер оның түсіндірмелік мәтіні аз болса. Негізінен, толық мәтіннің үлкен үзінділердің болуы модельді жеткілікті түрде толықтырмай себебі немесе оның біртекті болмауының себебі болады, мұнда бір модельдің шеңберінде сипаттамасы жағынан түрлі ақпарат ұсынылады. Себебі модельдің жалпы декомпозициясы жеке диаграмма типтеріне жүйе туралы әзірлеушілердің ең егжей-тегжейлі түсініктерін қанағатттандырып, канонды диаграммалардың тиісті элементтеріне модельдеу қырлары мен олардың көрсете білу қажет.

6. Әрбір диаграмма оның барлық элементтерін дұрыс түсіндіру және барлық қолданылатын графикалық символдардың семантикасын түсіну үшін өзі жеткілікті болуы тиіс. Кез келген түсіндірмелік мәтіндер, олар диаграмманың жеке элементтері болып табылмайды (мысалы, түсініктемелер) әзірлеушілердің назарына қабылданбауы тиіс. Жеке жеткілікті жалпы үзінділер салынған немесе бағынышты диаграммаларды құра отырып, осы типтегі өзге диаграммаларда анықталуы немесе егжей-тегжейленуі мүмкін. Осылайша, жүйенің UML тіліндегі моделі иерархиялық салынған диаграммалардың пакетін білдіреді, егжей-тегжей тиісті жүйенің жобаны іске асыратын бағдарламалық кодын келесіде генерациялау үшін жеткілікті болуы керек.

7. Қосымшаның нақты моделі үшін диаграмма типтерінің саны қатаң белгіленген болып табылмайды. Қарапайым қосымшалар үшін барлығы дерлік диаграммаларды құру қажеттілігі жоқ екендігі туралы сөз болып отыр. Олардың кейбіреулері жүйенің жобасында мүлде болмауы мүмкін және де бұл факті әзірлеушінің қатесі болып саналмайды. Мысалы, жүйенің моделі пайдаланушының компьютерінде жергілікті орындалатын қосымшалар үшін жазу диаграммасынан тұрмауы мүмкін. Диаграммалардың тізбесі жүйенің нақты жобасының сипаттамасына байланысты болатындығын түсіну маңызды.

8. Жүйе модельдерінің кез келгені UML тілі нотациясында анықталған элементтерден ғана тұруы тиіс. Жобаны әзірлеуді бастау талабына алады, тек UML метамоделінде анықталған конструкциялар ғана қолданылады. Тәжірибе көрсетіп отырғандай, бұл конструкциялар бағдарламалық құралдардың үлгілік жобаларының көбісін ұсыну үшін біршама жеткілікті. Және UML тілінің негізгі элементтерінің болмаған кезінде жүйенің нақты моделін дұрыс ұсыну үшін ұлғайтудың механизмдерін қолдану қажет. Бұл ретте UML тілінің метамоделінің негізгі нотациясына жатқызылған элементтердің семантикасын қайта анықтауға болмайды.

Диаграммалардың жеке типтерін құру процесі осы диаграммалардың элементтерінің семантикасына байланысты өзінің ерекшеліктері бар. Нысанға бағытталған жобалау процесінің өзі UML тілінің мәнмәтінінде арнайы атауын - ұтымды бірыңғайланған процесс (Rational Unified Process, RUP) атауын алды. RUP тұжырымдамасын және оның негізгі элементтерін А.Джекобсон UML тілімен жұмыс барысында әзірленген. RUP тұжырымдамасының мәні жүйе моделінің тиісті диаграмма типтерін әзірлеу жүзеге асырылатын жеке сатыларға бөлетін нысанға бағытталған жобалау процестері немесе бірізділік декомпозициясында болады. Бұл ретте RUP алғашқы сатыларында жүйе құрылымының статикалық моделінің логикалық көрінісі құрылады, ал содан кейін - әрекет моделінің логикалық көрінісі, және тек осыдан

кейін - жүйе моделінің физикалық көрінісі.

БАҚЫЛАУ СҰРАҚТАРЫ ЖӘНЕ ТАПСЫРМАЛАРЫ

1. Нысанға бағытталған тәсілдің мәні неде?
2. Нысанға бағытталған тәсілдің тұжырымдық негізі не болып табылады?
3. Нысандық модельдің негізгі элементтерін атаңыз.
4. UML ұғымына сипаттама беріңіз.
5. UML қолданудың артықшылықтары қандай?
6. Жүйенің әрекеті қандай мәндерін сипаттайды?
7. UML диаграмма түрлерін атап өтіңіз.
8. Әдетте класс диаграммалары қандай мәннен тұрады?
9. Көп модельдік принципі нені көрсетеді?
10. Жүйені иерархиялық құру принципі нені білдіреді?
11. UML қандай диаграмма түрлері айқындалған?
12. Нысанға бағытталған тәсілдің артықшылықтарын атаңыз.
13. UML пайдалану нұсқасы нені білдіреді?
14. Неліктен пайдалану нұсқалары диаграммалары мәтіндік сценариймен толықтыру ұсынылады?
15. Әрекет диаграммасын не үшін қолданады?
16. Бірізділік диаграммасының ерекшелігі неде?
17. Класс диаграммасы графикалық қалай бейнеленеді?
18. Компоненттер диаграммасы қалай қолданылады?
19. Модельдеудің құрылымдық және нысанға бағытталған әдістерінің салыстырмалы сипаттамасын беріңіз.

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ІСКЕ АСЫРУ САТЫСЫ

6.1. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ СӘУЛЕТІ

Бағдарламалық қамсыздандыру сәулеті - бұл өзара әрекеттесетін қосалқы жүйелердің кейбір жиынтығынан тұратын жүйе ретіндегі бағдарламалық қамсыздандырудың көрінісі.

Бағдарламалық қамсыздандырудың сәулеті - бұл бағдарламаның немесе есептегіш жүйенің құрылымы, ол бағдарламалық компоненттерді, осы компоненттердің сыртынан көрінетін қасиеттерін, сондай-ақ олардың арасындағы байланысты қамтиды.

Сәулетті әзірлеу тәуелсіз компоненттерге қатысты бөліну принципі іске асырылатын бағдарламалық өнімнің қиындығымен күресудің бірінші сатысы болып табылады.

Бағдарламалық қамсыздандырудың сәулетін әзірлеудің негізгі міндеттері:

- бағдарламалық қосалқы жүйелерді бөлу және бағдарламалық қамсыздандырудың сыртқы функцияларын оларда көрсету (сыртқы сипаттамаларымен берілген);
- бөлінген бағдарламалық қосалқы жүйелердің арасындағы өзара байланыс тәсілдерін анықтау.

Осы сатыда қабылданатын шешімдерді есепке ала отырып, бұдан әрі нақтыландыру жүргізіледі және функционалдық сипаттамаларды құрады.

Бағдарламалық жүйе *дербес орындалатын бағдарламалық кешенін* білдіреді, яғни осындай бағдарламалар жиынтығы:

- осы бағдарламалардың кез келгені пайдаланушылармен активтендірілуі мүмкін;
- активтендірілген бағдарлама орындалған кезде осы жиынтықтағы басқа бағдарламалар активтендірілген бағдарлама орындалуын аяқтағанша активтендіріле алмайды;
- бұл жиынтықтың барлық бағдарламалары осы не басқа ақпараттық ортаға қолданылады.

Басқару бойынша осы жиынтықтың бағдарламалары ешбір байланыспайды - өзара әрекеттері олардың арасында жалпы ақпараттық орта арқылы ғана жүзеге асырылады.

Көп қабатты сәулет жүйесі бағдарламалық қосалқы жүйелердің бірыңғайландырылған қабаттарынан тұрады, олар:

- әрбір қатпарында келесілердің қасиеттері туралы ештеңе белгілі емес, біршама жоғары қатпарлары;
- әрбір қатпары тікелей келесі компоненттерге келуі мүмкін, яғни біршама төмен қатпары, алдын-ала анықталған интерфейс арқылы, ол алдыңғы қатпарлардың барлығының ішкі құрылысы туралы ештеңе білмейді. Бұл басқару жөніндегі өзара байланыс деп аталады;
- әрбір қатпардың белгілі бір ресурстары бар, олар не басқа қатпарлардан жасырады, не болмаса аталған интерфейс арқылы келесі қатпарға олардың дерексіздендірілуінен алынады.

Көп қатпарлы сәулеті бар бағдарламалық жүйеде әрбір қатпар кейбір деректер дерексіздігін жүзеге асыра алады. Қатпарлар арасындағы байланыс әрбір қатпардың келесі қатпарына келу параметрлерінің мәнімен және төменгі қатпардан жоғарғыға оның нәтижелерін берумен шектелген. Бірнеше қатпарлармен ауқымды деректерді қолдану мүмкін емес.

Көп қатпарлы сәулет операциялық жүйелермен қолданылады: төменгі қатпары аппаратураны, аралық - ядро, жоғарғы - бағдарламалар мен қосымшаларды өңдейтін утилиттерді қолданады. Әрбір қатпар тек аралас қатпарлармен өзара байланысуы мүмкін. Операциялық жүйе осылай ұйымдасқан кезде тікелей аппаратурамен, ал ядро қатпары арқылы тікелей өзара байланысуы мүмкін емес.

Көп қатпарлы тәсіл кез келген типтегі күрделі жүйелердің, оның ішінде бағдарламалық жүйелердің декомпозициясының әмбебап және ұтымды тәсілі болып табылады. Әрбір қатпары жоғарыда жатқан қатпарға қызмет көрсетеді, ол үшін қатпараралық интерфейсін түзетін функциялардың кейбір жиынтығын орындайды. Төмен жататын қатпардың функцияларының негізінде келесі (иерархиясы бойынша жоғарыға) қатпары өз функцияларын құрады - біршама күрделі және біршама қуатты, олар өз кезегінде, жоғары жатқан қатпардың біршама қуатты функцияларын құру үшін қарапайым болады. Қатаң қадағалар тек жүйенің қатпарлары арасындағы өзара байланысқа қатысты болады, ал модульдердің арасындағы байланыс қатпарлары туынды болуы мүмкін. Жеке модуль өз жұмысын жеке орындауы мүмкін немесе өз қатпарының басқа модульдеріне жүгінеді, не болмаса қатпар аралық интерфейсі арқылы төмен жатқан қатпарға жәрдемге жүгінеді.

Мұндай жүйені ұйымдастырудың көптеген артықшылықтары бар:

- жүйені әзірлеу жеңілдейді, мысалы («жоғарыдан төменге») қатпарлардың функциялары мен қатпараралық интерфейстерінің

функцияларын анықтайды, ал содан кейін егжей-тегжейлі іске асырғанда «төменнен жоғарыға» қозғала отырып, функциялардың біртіндеп азаюын анықтауға болады;

- жүйе жаңғыртылған кезде модульдің басқа қатпарларда қандай да бір өзгерістер жүргізу қажетсіз қатпардың ішіндегі модульдерді интерфейстер күшінде қалады.

Параллель әрекет ететін бағдарламалардың ұжымы орындау сатысында бір уақытта болатын өзара әрекет етуге қабілетті бағдарламалардың жиынтығын білдіреді. Бұл бағдарламалар нені білдіреді, біріншіден, жедел жадыға жүктелген, активтендірілген және уақытпен бір немесе бірнеше орталық процестерді бөледі, ал екіншіден, өзара динамикалық өзара байланысты жүзеге асырады, оның негізінде синхрондау жүргізіледі. Әдетте, мұндай процестер арасындағы өзара байланыс бір-біріне кейбір хабарламаларды беру арқылы жүргізіледі.

Мұндай сәулеттің қарапайым түрі конвейер болып табылады, ол UNIX операциялық жүйеде орналасады. Конвейер UNIX терминологиясында - бұл енгізу-шығару келесі қайта бағыттау орындалатын көптеген кейбір процестері; ол алдыңғы процестің стандартты шығу ағынына шығарады, келесі процестің стандартты енгізу ағынына түседі. Басқаша айтқанда, конвейер бағдарламалардың бірізділігін береді, мұнда стандартты шығу әрбір бағдарламаның, соңғы кезектіден басқасы, осы бірізділіктің келесі бағдарламасының стандартты енгізуімен байланысты.

Конвейер хабарламалардың кейбір ағынын өңдейді. Осы ағынның әрбір хабарламасы бірінші бағдарлама енгізуіне келіп түседі, ол оны өңдеп, өңделген хабарламаны келесі бағдарламаға береді, ал өзі ағынның кезекті хабарламасын өңдеуді бастайды.

Осылайша, конвейердің әрбір бағдарламасы әрекет етеді: алдыңғы бағдарламадан хабарлама алып, оны өңдеп, ол өңделген хабарламаны келесі бағдарламаға береді, ал конвейердің соңғы бағдарламасы бүкіл конвейердің жұмыс нәтижесін (нәтижелендіруші хабарламасын) шығарады. Осылайша, n-бағдарламаларынан тұратын конвейерде бір уақытта N-хабарламаларды өңдеуде болады.

Әрине, конвейердің әртүрлі бағдарламалары уақыттың кез келген кесіндісінде кезекті хабарламаларға өңдеуге жұмсауы мүмкін, осы процестердің синхрондауын қалай да қамтамасыз ету қажет. Кейбір процестер күту сатысында болуы мүмкін, не болмаса өңделген хабарламаны беру мүмкіндігі, не болмаса кезекті хабарламаны алу мүмкіндігі.

6.2. МОДУЛЬДІК БАҒДАРЛАМАЛАУ

Бағдарламалық қамсыздандыру, негізінде, үлкен жүйе болып табылады, сол себепті оны жеңілдету үшін шараларды қабылдау қажет. Бұл үшін мұндай бағдарламаны бағдарламалық модульдер деп аталатын бөліктерге әзірлейді. Ал мұндай бағдарламаны әзірлеу әдісі модульдік бағдарламалау деп аталады.

Бағдарламалық модуль (МемСТ 19781-90 сәйкес) - бағдарлама немесе бағдарламаның функционалды аяқталған үзіндісі, ол басқа бағдарламалық модульдермен біріктіру, сақтау, трансляция және жедел жадыға жүктеуге арналған.

Бағдарламалық модуль - бұл басқа бағдарламалық модульдерден дербес өңдеудің берілген функцияларын қамтамасыз ететін және оның белгілі тағайындалған мақсаты бар бағдарламаның жеке бөлігі. Көптеген модульдердің арасында ажыратады:

- *бас модуль* - бағдарламалық өнімді қосуды басқарады (жекеше түрінде болады);
- *басқарушы модуль* - басқа модульдердің өңдеуге шақырылуын қамтамасыз етеді;
- *жұмыс модульдері* - өңдеу функцияларын орындайды;
- *сервистік модульдер* және кітапхана, утилиттер - қызмет көрсету функцияларын жүзеге асырады.

Бағдарламалау тіліне және бағдарламашылардың жұмысын жеңілдетуге арналған пайдаланушы модульдеріне кіретін стандартты модульдерді ажыратады.

Модульдер күрделі міндеттерді модульділік принципіне сәйкес біршама ұсақ міндеттерге бөлуге мүмкіндік береді. Әдетте, осылайша, бағдарламашыларға бірнеше рет пайдалануға ыңғайлы функциялар, кластар, константа жиынтығы түріндегі функционалдылықты (интерфейсті) беретіндей жобаланады. Модульдер пакеттерге және кейін кітапханаларға бірігуі мүмкін. Модульді сәулетті пайдалану қолайлығы басқа жүйені өзгерту қажеттілігінсіз модульді жаңарту (ауыстыру) мүмкіндігінде болады. Көптеген жағдайларда түрлі модульдер бір серверде сияқты, басқа да серверлерде жүктемені тарату үшін және бөлінген сәулетті құру үшін қосылуы мүмкін.

Модульді бағдарламалау - бұл бағдарламаны белгілі бір ережелерге бағынатын модульдер, құрылымы деп аталатын азғантай тәуелсіз блоктардың жиынтығы ретінде бағдарламаны ұйымдастыру. Модульді

бағдарламалауды пайдалану бағдарламаны тестілеуді жеңілдетуге және қателерді анықтауға мүмкіндік береді. Аппаратты-тәуелді қосалқы міндеттер бір-бірінен қосалқы міндеттерден қатаң түрде бөлінуі мүмкін, құрылатын бағдарламалардың мобильділігін жақсартады.

Әрбір бағдарламалық модуль бағдарламаланады, құрастырылады және бағдарламаның басқа модульдерінен жеке қойылады және осы арқылы бағдарламаның өзге модульдерімен бөлінген. Сонымен қатар, әрбір әзірленген бағдарламалық модуль түрлі бағдарламалардың құрамына қосылуы мүмкін, егер оны пайдалану шарттары орындалса, онда осы модуль бойынша құжаттамада декларацияланған. Осылайша, бағдарламалық модуль бағдарламаның қиындығымен күресу құралы ретінде және бағдарламалауда қайталаумен күресу құралы ретінде қарастырылуы мүмкін (яғни, бағдарламалық кодты бірнеше рет қолдану және жинау құралы ретінде).

Модуль сипаттамаларының жиынтығын Майерс 1980 жылдардың өзінде ұсынған болатын. Ол төменде көрсетілген конструктивті сипаттамаларынан тұрады.

Модульдің өлшемі мұндағы операторлардың (жолдардың) құрамындағы сандармен өлшенеді. Модуль тым кішкентай немесе тым үлкен болмауы керек. Кішкентай модульдер бағдарламаның тым үлкен модульдік құрылымына әкеледі және оны ресімдеуге байланысты жөнелтпе құжаттарды ақтай алмайды. Үлкен модульдер зерделеу үшін және өзгертулерге қолайлы емес, олар бағдарламаны қосқанда бағдарламаны қайта трансляциялаудың қосынды уақытын біраз ұлғайтуы мүмкін. Әдетте, бірнеше ондықтан бірнеше жүз операторға дейін бағдарламалық модульдер ұсынылады.

Модульдің байланысуы (беріктілігі) - бұл оның ішкі байланыстарының шамасы. Модульдің беріктілігі қаншалықты жоғары болса, ол бағдарламаның бөліктеріне қатысты сыртқы бағдарламалардан жасырудың байланысы соншалықты артық, және де бағдарламаны жеңілдетуге ол қаншалықты көп үлес қоса алады.

Функционалды байланыс. Функционалды байланысы бар модуль қандай да бір белгілі функцияны іске асырады және осындай байланыс түріндегі екі модульге бөліне алмайды.

Бірізділік байланысы. Мұндай байланысы бар модуль тәуелсіз функцияларды орындайтын бірізді бөліктерге бөлінуі мүмкін, бірақ жалғыз функцияны бірлесе іске асыратын. Мысалы, бір және осы модуль бағалау үшін алдымен қолданылуы мүмкін, ал одан соң деректерді өңдеу үшін қолданылуы мүмкін.

Ақпараттық (коммуникативтік) байланыс. Ақпараттық байланысы бар модуль - бұл бір және осы деректер құрылымымен (ақпараттық объектімен) бірнеше операциялар немесе функцияларды орындайтын модуль, ол осы модульден тыс танымал емес болады. Бұл ақпараттық

байланыс деректердің дерексіз типтерін іске асыру үшін қолданылады.

Модульдің ілінісуі - бұл шара басқа модульдерден алынған деректер бойынша тәуелділігі. Деректерді беру тәсілімен сипатталады. Модульдің басқа модульмен ілінісуі қаншалықты әлсіз болса, олардың басқа модульдерге тәуелсіздігі соншалықты күшті. Модульдің ілінісуі деректердің беріліс тәсілімен сипатталады. Модульдің басқа модульдермен ілінісуі қаншалықты әлсіз болса, басқа модульдерден оның тәуелсіздігі соншалықты күшті.

Басқаша айтқанда, ілінісу - басқа модульдерден модульдің салыстырмалы тәуелсіздігінің шарасы. Тәуелсіз модульдер басқа модульдердің қайта жасалуынсыз түрлері өзгертілуі мүмкін. Модульдің ілінісуі қаншалықты әлсіз болса, ол соншалықты жақсы. Мұндай модульдердің өзара байланысын ұйымдастыру, олардың интерфейсін білу және тиісті түрде шығыс деректер бір модульден басқаның кірісіне қайта бағыттайды.

Деректер бойынша ілінісу (параметрлік) - бұл деректер модульге оның басқа модульге жүгіну нәтижесі ретіндегі мәні болып беріледі, не болмаса кейбір функцияларды есептеу үшін басқа модульге жүгіну нәтижесі ретінде беріледі. Бұл ілінісу түрі бағдарламалау тілдерінде функцияларға (процедураларға) жүгіну кезінде іске асырылады. Бұл ілінісудің екі түрі деректердің сипаты болып анықталады.

Модульдің ескішілдігі - бұл жүгінудің алдыңғы тарихынан тәуелсіздігі. Модуль ескішілдігі деп аталады, егер оған жүгіну нәтижесі (әсері) оның параметрлерінің мәніне ғана емес, сондай-ақ оған жүгінудің тарихына байланысты.

Модуль көп жағдайларда ескішіл болуы керек, бірақ тарихты сақтау керек болған жағдайларда. Алдыңғы тарихқа байланысты модульдерді деректер бойынша ілінісу керек болатын жағдайларда ғана қолдану керек. Модульдің алдыңғы тарихына байланысты сипаттамасында осы тәуелсіздік анық қалыптасуы керек, ол осындай модульдің әрекетін болжау мүмкіндігі пайдаланушыда болу үшін.

6.3. КОДТАУ ЖӘНЕ РЕТКЕ КЕЛТІРУ. БАҒДАРЛАМАНЫҢ ҚАТЕЛЕРІ

Кодтау - қандай да бір бағдарламалау тілінде белгілі бір алгоритмді іске асыру мақсатымен бағдарламалық кодты жазу процесі.

Кодтау талдаумен, жобалаумен, ретке келтірумен, тестілеумен, сүйемелдеумен бірге бағдарламалаудың бөлігі болып табылады. Басқаша айтқанда, кодтау кезінде тиісті тілдердің бірі болатын таңдап алынған бағдарламалау тілінің құралдарымен бағдарламалық қамсыздандырудың құрылған моделіне сипаттама беріледі. Тілді таңдау тапсырыс берушінің тілегімен, не болмаса шешілетін міндеттерді және әзірлеушілердің жеке тәжірибесін есепке ала отырып, жүзеге асырылады.

Ретке келтіру деп бағдарламалық қамсыздандыруды тестілеу кезінде анықталған қателерді оқшаулау мен жергілікті ету процесін атайды.

Оқшаулау есептеу процесін бұзуды тудырған бағдарлама операторын/операторларын анықтау болып табылады.

Қателерді түзету үшін оның себебін анықтау қажет, яғни қателеден тұратын оператор немесе үзіндіні анықтау керек. Қателердің себептері анық болуы мүмкін, сол сияқты өте жасырын болуы мүмкін.

Қателерді бағдарламаны өңдеу сатысы бойынша жіктеу:

- компиляция қателері (синтаксистік қателер) - бағдарламаны синтаксистік және жекелей семантикалық талдауды орындау кезінде компилятор (транслятор, интерпретатор) белгілейтін қателер;
- жинау қателері - бағдарлама модульдерін біріктіру кезінде жинаушы (байланыстар редакторы) анықтаған қателер;
- орындау қателері - бағдарламаны орындау кезінде операциялық жүйе, аппараттық құралдар немесе пайдаланушылар анықтаған қателер.

Компиляция қателерін ең қарапайым топтарға жатқызады, мысалы тіл синтаксисі, негізінен, ол қатаң формалданған және де қателердің жіберілген орны көрсетілген түсініктемелерімен қатар жүреді.

Тіл синтаксисінің ережелері қаншалықты жақсы қалыптастырылған болса, компилятор қателерді соншалықты көбірек анықтай алады. Сәйкесінше келесі сатыларда қателер аз анықталатын болады. Осыған байланысты синтаксисі қорғалған және қорғалмаған синтаксисі бар бағдарламалау тілдері туралы айтады.

Жинау қателері сыртқы сілтемелері рұқсат етілгенде анықталған проблемалармен байланысты. Мысалы, басқа модульдің қосалқы

бағдарламасына жүгінуі көзделген, ал модульдер біріктірілгенде бұл қосалқы бағдарлама табылмаған немесе параметрлер тізімімен түйіспейді. Көптеген жағдайда мұндай тектес қателерді оқшаулап, жоюға болады.

Орындау қателері ең болжап болмайтын болып табылады. Олардың табиғаты әр түрлі болуы мүмкін және тиісінше, әр түрлі шығады. Қателердің кейбіреуі анықталады және операциялық жүйемен құжатталады. Орындалу қателері келесі түрде шығуы мүмкін:

- машиналық командаларды орындалуын бақылау схемаларымен белгіленген қателер туралы хабарламалардың шығуы, мысалы, разрядтардың толтырылуы, нөлге бөлу, дербестенуді бұзу және т.с.с.;
- операциялық жүйемен анықталатын қате туралы хабарламаның пайда болуы, мысалы, жадыны қорғаудың бұзылуы, жазып алудан қорғалған құрылғыларға жазу талпынысы, берілген аты бар файлдың болмауы және т.с.с.;
- компьютердің «қатып қалуы» - кейде операциялық жүйені қайта жүктеусіз бағдарламаны аяқтау мүмкін, жұмысты жалғастыру үшін қайта жүктеу қажет болады;
- алынған нәтижелердің күтілген нәтижелермен сәйкес келуі.

Орындау қателерінің себептері тым әртүрлі, ал оқшаулау да тым күрделі болуы мүмкін. Қателердің барлық мүмкін себептерін келесі топтарға бөлуге болады:

- алғашқы деректерді дұрыс анықтамау;
- логикалық қателер;
- есептеу нәтижелерінің қателерінің жиналуы.

Алғашқы деректердің дұрыс анықталмауы, егер енгізу-шығару операцияларын орындаған кезде қандай да бір қателер шықса болады: беріліс қатесі, қайта түрлену қатесі, қайта жазу қатесі және деректер қатесі. Мұнда техникалық арнайы құралдарды қолдану және қателерден қорғап бағдарламалау осы қателердің тек бір бөлігін анықтауға және алдын алуға мүмкіндік береді, ол туралы ұмытуға болмайтындығы сөзсіз.

Логикалық қателердің табиғаты әртүрлі. Олар жобалау кезінде жіберілген қателерден болуы мүмкін, мысалы, әдістерді таңдау кезінде, алгоритмдерді әзірлеген кезде немесе класс құрылымын анықтау, ал модульді кодтау кезінде тікелей енгізілуі мүмкін.

Есептеу нәтижелерінің қателерін жинау қателеріне мыналар жатады:

- айнымалыларды дұрыс пайдаланбау қателері, мысалы, деректердің типтерін сәтсіз таңдау, айнымалыларды оларды бастауға дейін пайдалану, массивтерді анықтау шекарасынан шығатын индекстерді пайдалану, деректер типін анық немесе анық емес деректер типін

қолданған кездегі деректер типтерінің сәйкестілігінің бұзылуы, олар айнымалы, ашық массивтер, бірлестіктер, динамикалық жады, мекенжайлік арифметика және т.с.с. типтендірілгендерін қолданған кезде жадыда орналасқан деректер типін анықтау;

- есептеу қателері, мысалы, арифметикалық емес айнымалыларды дұрыс пайдаланбау, бүтін сандармен дұрыс жұмыс істемеу, есептеу процесінде деректер типтерін дұрыс қайта түрлендіру және т.с.с.;
- модульдердің өзара әрекет қателері, яғни, модульаралық интерфейс, мысалы, параметрдің берілісі кезінде типтер мен бірізділіктің бұзылуы, формалды және нақты параметрлерді өлшем бірлігінің бірлігін сақтамау, оқшаулар және өзекті айнымалылардың әрекеттерінің саласын бұзу;
- кодтаудың өзге қателері, мысалы, бағдарламалаудың нақты тілінің шектеулері немесе ерекшеліктерін ескермеу, кодтау кезіндегі бағдарлама логикасын дұрыс іске асырмау.

Ретке келтіру процесі бағдарламашыдан техникалық құралдар, операциялық жүйелер, бағдарламалау аясы мен тілі қолданатын басқару сипаттаманы, түрлі қателердің табиғаты мен сипаттамаларын, тиісті бағдарламалық құралдарды және ретке келтіру әдістемесін терең білуді талап ететін күрделі процесс болып табылады.

Ретке келтірудің күрделілігі келесі факторлардың әсерінің салдарынан артуы мүмкін:

- қателердің жанама білінуі;
- қателердің өзара әсерінің мүмкіндігі;
- түрлі қателердің сыртынан бірдей білінуін алу мүмкіндіктері;
- кейбір қателердің іске қосудан іске қосуға көрінуінің қайталануы (стохастикалық қателер);
- бағдарламаға кейбір өзгертулерді енгізу кезінде зерттеліп отырған жағдайда сыртқы қателерінің білінуін жою мүмкіндігі, мысалы, бағдарламаға диагностикалық үзінділер қосылғанда жойылуы мүмкін немесе қателердің сыртынан көрінуі өзгертілу мүмкін;
- әртүрлі бағдарламашылардың бағдарламаның жеке бөліктерін жазуы.

Бағдарламаны ретке келтіру кез келген жағдайда қате туралы барлық бар ақпараттың ойланып, логикалық түсіндірілуін болжайды. Қателер көбінесе қосымша ақпарат алусыз тестілеу нәтижелерін және бағдарлама мәтіндерін мұқият талдаудың жанама белгілері бойынша анықтауға болады. Бұл ретте түрлі әдістерді қолданады:

- қолмен тестілеу;
- индукция
- дедукция;
- кері бақылау.

Қолмен тестілеу әдісі - бағдарламаны ретке келтірудің ең

қарапайым және жалғыз тәсілі. Қателерді анықтаған кезде тестілендірілетін бағдарламаны қолмен орындау керек, тестілік жиынтықты қолдана отырып, олармен жұмыс кезінде қате табылғанда. Әдіс тиімді, бірақ, үлкен бағдарламалар үшін күрделі есептеуі бар бағдарламалар үшін қолданылмайды, сондай-ақ қате бағдарламалаушының операциялардың орындалуы туралы түсінігі дұрыс болмаған жағдаймен байланысты болғанда қолданылмайды. Бұл әдісті ретке келтірудің басқа әдістерінің құрамдас бөлшегі ретінде жиі қолданады.

Индукция әдісі қате туралы хабарлама ретінде немесе есептеудің қате нәтижелері ретінде пайда болатын қателердің симптомдарын мұқият талдауға негізделген. Егер компьютер жай ғана «қатып қалған» болса, онда қатенің пайда болу фрагменті соңғы алынған нәтижелерден және пайдаланушының әрекеттерінен есептейді. Осындай тәсілмен алынған ақпаратты зерделеуге болады, ол үшін бағдарламаның тиісті үзіндісін қарап шығады. Нәтижесінде кейіннен тексерілетін қателер туралы болжамды шығарады. Егер болжам дұрыс болса, онда қате туралы ақпаратты егжей-тегжейлейді, әйтпесе басқа болжамды шығарады.

Қате туралы деректерді жинай отырып, оның білінуі туралы белгілі болған кезде барлығын жазып алу керек, мұнда қатесі бар үзінді қалыпты орындалатыны сияқты, қате байқалатын жағдайларды да назарға алу қажет. Егер деректерді зерделеу нәтижесінде ешқандай болжам анықталмаған болса, онда қате туралы қосымша ақпарат қажет болады.

Дедукция әдісі келесідей болады. Алдымен қате туралы осындай көріністі болдыратын көптеген себептерді қалыптастырады. Содан кейін бар деректерге қарама-қайшы келетін себептерге талдау жасайды. Егер барлық себептері алынып тасталса, онда зерттелетін үзіндіні қосымша тестілеу қажет болады. Керісінше жағдайда біршама ықтимал болжамды дәлелдеуге талпынады. Егер болжам алынған қате белгілерін түсіндіретін болса, онда қате табылды, әйтпесе келесі себепті тексереді.

Кері бақылау әдісі шағын бағдарламалар үшін қолданылады. Тексеру дұрыс емес нәтижені шығару нүктесінен басталады. Бұл нүкте үшін қолдағы бар нәтижені алуға әкелетін негізгі айнымалылардың мәндері туралы болжам құрылады. Бұдан әрі, осы болжамға сүйене отырып, алдыңғы нүктелердегі айнымалылардың міндері туралы ұсыныс жасайды. Процесті қатенің себебін тапқанша жалғастырады.

Жиірек кездесетін *бағдарламалық қателердің* кейбір *санаттарын* қаратырамыз.

Функционалдық кемшіліктері. Бұл кемшіліктер өзі істеу қажетті нәрсені орындамай, өз функцияларының біреуін нашар орындайтын немесе толығымен нашар орындайтын бағдарламаға тән.

Бағдарлама функциялары оның сипаттамасында толық сипатталуы тиіс және бекітілген сипаттізімнің негізінде өз жұмысын құрады.

Пайдаланушы интерфейсінің кемшіліктері. Пайдаланушы интерфейсінің жұмыс істеу қолайлылығын және дұрыстығын бағалауды тек онымен жұмыс процесінде ғана мүмкін. Бұл жұмысқа пайдаланушының өзі қатысқаны дұрыс. Бұған БӨ прототипін әзірлеу көмегімен қол жеткізуге болады, мұнда талаптардың сипаттамасында пайдаланушы интерфейсінің одан әрі белгіленуіне барлық талаптарды келісу және ретке келтіру жүргізеді. Талаптардың сипаттамасын бекіткеннен кейін одан ауытқу немесе соңғыларының орындалмауы қате болып табылады. Бұл пайдаланушы интерфейсіне толық қатысты болады.

Жеткіліксіз өндірушілік. Кейбір БӨ әзірлеген кезде оның маңызды сипаттамасы жұмыс жылдамдығы болуы мүмкін, кейде бұл критерий БӨ тапсырыс берушінің талаптарында беріледі. Егер пайдаланушыға бағдарлама баяу жұмыс істейтін болып көрінетін болса, ол қиын, егер бәсекелес бағдарламалар талаптардың сипаттамасында берілген сипаттамаларын қанағаттандырмаған болса, ол тіптен қиын. Бұл жойылуы қажет қате болады.

Қателерді қате өңдеу. Қателерді өңдеу процедуралары - бағдарламаның өте маңызды бөлігі. Қатені дұрыс анықтап, бағдарлама ол туралы хабарлама беруі тиіс. Мұндай хабарламаның болмауы бағдарлама жұмысындағы қате болып табылады.

Шектік шарттарды қате өңдеу. Әртүрлі шектік жағдайлар болады. «Артық» немесе «кем», «бұрын» немесе «кейін», «бірінші» немесе «соңғы», «қысқарақ» немесе «ұзынырақ» ұғымдары қолданылатын бағдарлама жұмысының кез келген қыры диапазон шекараларында тексерілуі тиіс. Бағдарлама диапазондарының іші тамаша жұмыс істеуі мүмкін, ал шекараларында кездейсоқ жағдайлар болып жатуы мүмкін, олар өз кезегінде бағдарламалық қамсыздандырудың жұмысында қателерге әкеледі.

Есептеу қателері. Есептеу қателеріне қате формулалар, қате таңдап алынған есептеу алгоритмдеріне, өңделетін деректерге қолданылмайтын формулаларға қатысты қателер жатады. Қателердің арасында ең көп таралғандары дөңгелектеу қателері болып табылады.

Ағындарды басқару қателері. Бағдарламаның жұмыс істеу логикасы бойынша бірінші әрекеттің соңынан екіншісі орындалуы керек. Егер мұның орнына үшінші немесе төртінші әрекет орындалған болса, онда ағындарды басқаруда қате жіберілген.

Жарысу жағдайы. Мысалға, жүйеде екі оқиға: А және Б күтілуде. Егер бірінші оқиға *А* келсе, онда бағдарламаның орындалуы

жалғасады, ал егер *Б* келсе, онда бағдарлама жұмысында жаңылу болады. Әзірлеушілер әрдайым бірінші бірінші оқиға *А* оқиға болуы керек дейді де, *Б* жарыстан жеңіп, бірінші келуі мүмкін деп ойламайды. Классикалық жарыс жағдайы осылай.

Жарысу жағдайын тестілеу жеткілікті дәрежесінде күрделі. Олар өзара әрекет ететін процестер мен ағындар параллель орындалатын жүйелерге тән, сондай-ақ нақты уақыттың көп пайдаланушы жүйелеріне тән. Мұндай жүйелердегі қателерді қайта шығару қиын, және оларды анықтауға әдетте көп уақыт талап етіледі.

А с қ ы н ж ү к т е у. Бағдарлама жұмысындағы іркілістер жадының жетпеуінен немесе қажетті жүйелік ресурстардың болмауынан болады. Әрбір бағдарламаның өз шектері бар, бағдарлама шамадан тыс жүктememен жұмыс істей алмайды, мысалы тым үлкен көлемдегі деректермен. Бағдарламаның нақты мүмкіндіктері бағдарламаның сипаттаманы ресурстарына сәйкес келеді ме, және ол шамадан тыс жүктелгенде қалай әрекет ететіндігі туралы мәселе.

К о м п ь ю т е р а п п а р а т у р а с ы м е н д ұ р ы с ж ұ м ы с і с т е м е у. Бағдарламалар аппараттық құрылғыларға дұрыс емес деректер жіберуі мүмкін, олардың қате туралы хабарламаларын елемеуі мүмкін, бос емес немесе тіпті жоқ құрылғыларды пайдаланбақ болуы мүмкін. Тіпті, егер қажетті құрылғы сынған болса, бағдарлама бағдарлама мұны түсініп, ал оған жүгіну кезінде «іркіліп қалмауы» тиіс.

6.4. БАҒДАРЛАМА ҚҰРЫЛЫМЫН ӘЗІРЛЕУ ӘДІСТЕРІ

Бағдарламаның модульдік құрылымы ретінде салынатын бұтақтары бар ағашты қоса алғанда ағаш тәрізді құрылымды қолдану қабылданған. Мұндай ағаштың тораптарында бағдарламалық модульдер орналасады, ал бағытталған доғалар (нұсқарлар) модульдердің статикалық бағыныштылығын көрсетеді, яғни әрбір доға одан шығатын модуль мәтінінде оның кіретін модуліне сілтеме бар. Басқаша айтар болса, әрбір модуль оған қатысты модульдерге жүгіне алады, яғни осы модульдер арқылы өрнектеледі. Бұл ретте бағдарламаның модульдік құрылымы, соңында осы бағдарламаны түзетін модульдердің сипаттамаларының жиынтығынан тұруы тиіс.

Бағдарламалық модульдің сипаттамасына мыналардан тұрады:

- қолданылатын бағдарламалау тілінде синтаксистік жағынан оған жүгінуінің дұрыс бағытын құратын (оның кез келген кірісіне) кірістерінің синтаксистік сипаттамасынан;
- модульдің функционалды сипаттамасынан (осы модульдің оның әрбір шығысы бойынша функцияларының семантикасына сипаттама). Модульдің функционалды сипаттамасы бағдарламалық қамсыздандырудың функционалдық сипаттамасы сияқты құрылады.

Бағдарламаны әзірлеу процесінде оның модульдік құрылымы әртүрлі қалыптасуы мүмкін. Сол себепті бағдарлама құрылымын әзірлеудің түрлі әдістері туралы айтуға болады. Классикалық әдістерге төмен түсетін және жоғары көтерілетін әзірлеу әдістерін жатқызуға болады.

Жоғары көтерілетін әзірлеу әзірлеу әдісін қолданғанда бағдарламаның модульдік құрылымы ағаш түрінде құрылады. Содан кейін кезекпен ең төменгі модульден бастап модульдер бағдарламаланады, олар әрбір бағдарламалық модуль үшін олар жүгіне алатын барлық модульдерге бағдарламалану тәртібімен құрылады.

Барлық модульдер бағдарламаланып болған соң оларды кезекпен тестілеуден өткізеді және ретке келтіру жоғары көтерілетін тәртіптегіндей бағдарламалауда жүргізіледі. Осындай тәртіппен әрбір модуль үшін әзірлемелер, бас модульден басқасы жетекші бағдарламаны (модульді) құру керек болады, ол тестіленетін модуль үшін ақпараттық аяның күйін даярлап, оған қажет жүгінуді жүргізеді. Бұл «ретке келтіріп» бағдарламалаудың үлкен көлеміне әкеледі және де ешбір кепілдік бермейді, ол жұмыс бағдарламасында орындалған жағдайларға модульдерді тестілеудің кепілдігін бермейді.

Өрлейтін тестілеу қателердің орнын анықтаудың жақсы тәсілі болып табылады. Егер қате бір модульді тестілеу кезінде анықталған болса,

онда оның көзіне анықтау үшін бүкіл жүйенің кодына талдау жасаудың қажеті жоқ. Егер қате екі тестіленген модельдердің бірлескен жұмысында байқалатын болса, онда қате олардың интерфейстерінде болуы мүмкін. Өрлеу тестілеуін орындау кезінде бағдарламашы жалғыз модульге, екі модуль арасындағы параметрлердің берілуіне назарын қояды, яғни аз ғана бөлігіне, оның көмегімен тестілеу қатенің анықтаудың үлкен ықтималдылығымен мұқият жүргізіледі.

Төмендеуші әзірлеуде кезекпен бағдарлама модульдері бағдарланады, жоғарғы (бас) деңгейдің модулінен бастап, жүгінетін модуль бағдарламаланған болса ғана қандай да бір басқа модульді бағдарламалауға өтеді. Бағдарламаның барлық модульдері бағдарламаланып болғанша оларды кезекпен тестілеу жүргізіледі және ретке келтіру төмендеу тәртібімен де жүргізіледі. Бағдарламаның бас модулі ақпараттық аяның «табиғи» күйінде тестіленеді. Ол жүгіне алатын модульдер имитаторлармен ауыстырылады. Модуль имитаторы қарапайым бағдарламалық үзіндіні білдіреді, және ол өз кезегінде қажетті нәтижені береді.

Осыған ұқсас басқа модульдердің тестіленуі де өтеді. Бұл ретте әрбір модуль осы модульге жүгінген сәттегі ақпараттық аяның «табиғи» жағдайларында тестіленеді. Мұндай тәртіпте бағдарламаның әзірленуі, барлық қажетті өзекті ақпарат уақытылы қалыптасады.

Әрбір тәсілдің артықшылықтары мен кемшіліктері бар. Мамандардың пікірлері жоғарыда сипатталған екеуінің жоғарғысы біршама тиімді және қандай әзірлеу әдісі тиімді екендігі туралы пікірлері де айырмашылықтарымен болады. Кейбір мамандардың пікірінше, ең жақсы өрлеу әзірлемесі мен тестілеу; ал басқалары жалпы төмендеуші тестілеу жақсы дейді.

Әрбір практикада стратегияны таңдау мәселесі келесі түрде шешіледі. Әрбір модуль мүмкіндігіне қарай оны жазғаннан кейін тестіленеді, нәтижесінже бағдарлама бөліктерінің кейбіреуін тестілеу бірізділігі өрлеуші, ал басқаларынікі - төмендеуші болуы мүмкін.

6.5. ПАЙДАЛАНУШЫ ИНТЕРФЕЙСІН ӘЗІРЛЕУ

Мамандар интерфейстің ыңғайлылығын бағалауға мүмкіндік беретін принциптер мен ережелердің кейбір жиынтығын қалыптастырады, осылай оның ыңғайлылығын арттыратын шешімдерін ұсынады.

Қолжетімділік ережесі. Жүйе тақырыптық аяны жақсы білетін, оны бұрын көрмеген пайдаланушыға, ешбір оқусыз оны пайдалануды бастап кете алатындай түсінікті болуы керек. Бұл ереже ұмтылу қажет болатын идеал болады, себебі практикада мұндай түсініктілік деңгейіне жету ешқашан мүмкін бола бермейді. Сонда да, осы идеалға қол жеткізу үшін не қажет болса, соны жасау керек.

Тиімділік ережесі. Жүйе онымен ұзақ уақыт жұмыс істейтін тәжірибелі пайдаланушылардың тиімді жұмысына кедергі болмауы керек. Ережелерді бұзудың айқын мысалы жүйенің тәжірибесі жоқ пайдаланушылар үшін жақсы келетін құралдарын пайдалануға мақсатқа сай болатындығы болып табылады, оның бірдеңені басқаша жасап қою мүмкіндіктерін шектейді, ол сарапшы үшін де тиімді емес, нені және қайда жасау керек.

Үздіксіз даму ережесі. Жүйе пайдаланушының білімі, дағдысы мен біліктерінің үздіксіз өсуіне ықпал етуі және өзгеріп отыратын оның тәжірибесіне бейімделуі керек. Нашар нәтижелер тек негізгі мүмкіндіктерін меңгеруді береді немесе бастаушы пайдаланушыны сарапшылар сенімді пайдаланатын күрделі интерфейспен оңаша қалуы. Бір мүмкіндіктер жиынтығынан басқасына өту кезінде үздіксіздікті бұзу қолайсыздық тудырады, себебі пайдаланушы жаңа мәнмәтінде қосылған мүмкіндіктермен айналысуға мәжбүр.

Контексті қадағалау ережесі. Жүйе жұмыс істейтін мәнмәтінмен келісілген болуы тиіс. Бұл ереже жүйенің жұмысқа қабілеттілігін «жалпы ғана емес», атап айтар болсақ мұны қолданатын орта үшін қабілетті болуын талап етеді. Мәнмәтінге кіріс және шығыс деректерінің көлемі мен сипаттамасы, осы жүйе жұмыс істейтін ұйымның түрі мен мақсаттары, пайдаланушылардың деңгейі, үй-жайдағы шу болуы кіреді.

Жоғарыда көрсетілген ережелер қолайлы интерфейсті қанағаттантандыруы қажет жалпы талаптарды айқындайды. Келесі принциптер пайдаланушы интерфейсінің қолайлы болуын арттыратын шешімдерді табуға мүмкіндік береді.

Құрылымдау принципі. Пайдаланушы интерфейсі мақсатқа сай құрылымдалуы керек. Мағынасы жағынан жақын оның тектес бөлшектері көрінетін түрде байланысуы керек, ал тәуелсіз бөлшектері - ажыратылып көрсетілуі керек; ұқсас элементтері ұқсас, ұқсас емес - ерекше болуы керек.

Қарапайымдылық принципі. Ең көп таралған операциялар барынша орындалуы керек. Бұл ретте біршама күрделі процедураларға сілтемелер болуы керек.

Көрінушілік принципі. Белгілі бір міндеттерді шешуге қажетті барлық функциялар мен деректер пайдаланушы шешу үшін көрінетін болуы тиіс.

Кері байланыс принципі. Пайдаланушы жүйенің қызметі және оның ішіндегі маңызды оқиғалар туралы хабарламаларды алуы тиіс. Хабарламалар ақпараттық, қысқа, бір мағыналы және пайдаланушы үшін түсінікті тілде болуы керек.

Толеранттылық принципі. Интерфейс пайдаланушының қателеріне икемді және шыдамды болуы керек. Қателерден болатын залал әрекеттерді болдырмау және қайталау мүмкіндіктері есебінен және пайдаланушының әрекеттерінен және олар енгізген деректердің интерпретациясы есебінен азаюы керек. Мүмкіндігінше пайдаланушының еркіндігін шектеуге негізделген міндеттемелі өзара әрекеттерден (модальды диалогтар) алшақ болу қажет.

Қайта пайдалану принципі. Интерфейстің жеңілдігін және оның ұқсас элементтер арасындағы ұқсастықты қамтамасыз ете отырып, ішкі және сыртқы компоненттерді бірнеше рет пайдалануға тырысу қажет.

Адам-машиналық интерфейс пайдаланушы мен компьютердің арасындағы байланысты қамтамасыз етеді - ол қойылған мақсатқа табыспен жетуге мүмкіндік береді. Өзара байланыс - компьютер мен пайдаланушының арасындағы осы әрекеттерге реакциялар және әрекеттермен алмасу.

Эргономикалық интерфейсті құру мақсаты адамның қабылдауы үшін қаншалықты мүмкін болса, ақпарат соншалықты тиімді бейнелеу үшін, және дисплейде ең маңызды ақпарат бірліктеріне құрылымдайды. Негізгі мақсат жалпы ақпаратты минималды етіп, пайдаланушы үшін қажетті болып табылатындарды ғана азайтуға болады.

Экранда бейнеленетін ақпараттың саны *экранның толықтығы* деп аталады. Зерттеулер экран тығыздығы қаншалықты аз болса, бейнеленетін ақпарат да қолжетімді және пайдаланушыға түсінікті болады, және керісінше, егер экран тығыздығы үлкен болса, бұл ақпаратты меңгеруде қиындық тудыруы мүмкін және анық түсінуге мүмкіндік береді. Тәжірибелі пайдаланушылар үлкен экранды тығыздығы бар интерфейстерді артық көреді. Экранда ақпарат топтастырылып, маңызды бөлшектерге жеңіл бөлінуі мүмкін. Бұл түстік кодтау кадрларын (фреймдерді), әдістерді қолданумен қол жеткізілуі мүмкін, назар аудару үшін кері бейнелеу немесе басқа әдістермен қол жеткізілуі мүмкін.

Интерфейстің қандай да бір элементтеріне назар аудару үшін басқалардың аясында - біршама сұр түстің аясында ашық болуының осы

элементтермен бөлінуін пайдалануға болады. Бірақ, бұл әдісті шамадан тыс қолданудың қажеті жоқ, себебі ашық элементтердің көп болуы пайдаланушыларға қолайсыздық тудырады. Осылайша, кері әсерге жетуге болады - интерфейстің шамадан тыс жүктелуі. Бұл әдісті тек қажет болғанда қолдану керек.

Түс пайдаланушының интерфейсін жақсартуы мүмкін, бірақ көптеген жүйелер үшін түстерді пайдалану пайдаланушының жұмысының тиімділігіне әсерін тигізбейді. Түстердің негізгі маөсаты - пайдаланушылар үшін біршама қызықты интерфейстерді құру. Сонда да, түстер пайдаланушы интерфейстерін жобалаушыларға көмегін тигізген жағдайлар да бар. Бұл түстер ақпараттарды топтастырғанда; ақпараттардың арасындағы айырмашылықтарды бөліп көрсеткен жағдайда; қарапайым хабарламаларды (қателерді, күйін және т.б.) бөліп көрсеткен кезде біршама ұтымды.

Түс - көзбен көрсететін қуатты құрал, бірақ оны абайлап қолдану қажет, ол пайдаланушының түстерді қате үйлестіруін болдырмас үшін керек.

Эргономикалық интерфейсті жобалаған кезде басшылыққа алу керек болатын түстерді пайдаланудың кейбір принциптері:

- экрандардың жүйелі болуы үшін 7-ге дейін және экранда 4-ке дейін түстердің санын шектеу керек;
- актив емес элементтер үшін өңсіз түстерді пайдалану керек;
- егер түстер ақпаратты кодтау үшін қолданылса, пайдаланушы кодты дұрыс түсінетіндігіне көз жеткізу керек, мысалы, төленбеген шоттар қызыл түспен белгіленеді;
- түстерді пайдаланушының түсінігіне орай қолдану керек, мысалы, картограф үшін жасыл - орман, сары - шөл, көк - су; химик үшін қызыл - ыстық, көк - суық және т.б.;
- күйін бейнелеу үшін: қызыл түс қауіпті білдіреді / тоқта, жасыл түс жұмыстың жалғасатындығын, сары - ескерту белгісін білдіреді;
- назар аударту үшін ең тиімді ақ, сары және қызыл түстер;
- деректерді жеңіл түсіну үшін 7 түс (кемпірқосақ) спектрі қолданылады;
- деректерді бөлу үшін түстердің түрлі спектрінен таңдап алу керек (қызыл/жасыл, көк/сары, кез келген түс/ақ;
- деректерді топтастыру, біріктіру және осыған ұқсас топтастыру үшін спектрдегі көршілес болатын түстерді қолдану қажет (қызғылт сары түс/сары, көк/күлгін).

Экрандағы ақпараттың орналасуына назар аудару керек. Деректерді пайдаланушы қажетті ақпаратты қайдан және қалай тауып алу керектігін білдіретіндей орналастыру керек:

- назар аудару қажет болатын ақпарат пайдаланушының назарын алатындай көрінетін орында әрдайым «мысалы, ескертуші

хабарламалар мен қателер туралы хабарламалар көрсетілуі керек);

- жиі көру қажет емес ақпарат (мысалы, анықтама құралдары) көрсетілмеуі керек, бірақ ол қажет кезде қолжетімді болуы керек. Мысалы, Анықтама немесе тиісті мәзір опциясы әрбір экранда қолжетімді болуы керек;
- тым шұғыл емес немесе тым қажетті емес ақпарат пайдаланушының алдында үнемі тұрмауы керек, бірақ қажет болған кезде қолжетімді болуы керек.

Мәтіндік диалогтар мен бейнелерді құрған кезде басшылыққа алу керек болатын кейбір принциптерді келтірейік:

- төменгі регистрдегі мәтін жоғарғы регистрде толық басылған мәтінге қарағанда шамамен 13% жылдамырақ оқылады;
- жоғарғы регистрдің символдары назар аударатын ақпарат үшін біршама тиімді. Қандай да бір ақпаратты бөліп көрсеткіңіз келмесе, онда жоғарғы регистрді қолданбаңыз;
- оң жақ шеті бойында түзетілген мәтінді оң жақ өріспен түзетілмеген тең бөлінген мәтінге қарағанда оқу қиынырақ;
- жолдардың арасындағы оңтайлы аралық тең немесе символдардың биіктігіне қарағанда үлкенірек.

Автоматтандырылған жүйенің қажетті элементі - мәзір, ол пайдаланушыға қосымшалардың ішіндегі міндеттерді орындауға және шешім қабылдау процестерін басқаруға мүмкіндік береді.

Мәзір - пайдаланушылар әрекеттерін таңдай алатын және орындай алатын, осы арқылы интерфейстің күйіне өзгертулер жүргізе алатын экранда көрсетілетін опциялардың жиынтығы.

Мәзірдің артықшылығы пайдаланушылардың элементтің аты немесе әрекеттердің атын есте сақтауы міндетті емес, олар орындауы керек - олар оны мәзір тармақтарының арасынан тануы керек. Осылайша, мәзірді тіпті тәжірибелі емес пайдаланушы да қолдана алады. Алайда, мәзір жобасы мұқият ойланып жасалуы керек - мәзір тиімді болуы үшін мәзір тармақтарының атаулары да анық болуы керек.

Мәзір экран орнының біразын алуы мүмкін, алайда бұл проблеманың шешімі бар - қалқыма немесе төмен түсетін мәзір. Иконканы басқан кезде мәзір жолы немесе өзге объекті қалқымалы немесе төмен түсетін мәзірді шақыртады.

Жүйені жобалау процесінде қосымшалар мәзірін түсінікті және қолдану кезінде жеңіл болатындай ең жақсы тәсілмен мәзірді бейнелеуді қабылдау керек. Әдетте, мәзір командалары кейбір иерархиялық тәсілмен жеңілдендіріліп жасалған. Негізгі проблема мәзір тармақтарын түрлі деңгейлер бойынша дұрыс тарату және оларды дұрыс топтастыруда болады. Мәзірді жобалау принциптері:

- мәзір құрылымы шешілетін жүйелік міндеттің құрылымына сәйкес болуы керек, мәзірді ұйымдастыру алға қойылған міндеттерге қол

жеткізу үшін қадамдардың ұтымды бірізділігін беруі керек;

- мәзір тармақтары қысқа, грамматикалық дұрыс және мәзірдегі өзінің тақырыбына сәйкес келуі тиіс. Мәзір тармақтарының тәртібі келісімге, пайдалану жиілігіне, пайдалану тәртібіне пайдаланушының немесе қойылған міндеттердің қажеттілігіне қарай таңдап алынады;
- мәзір тармақтарын таңдау бірнеше тәсілмен қамтамасыз етілуі тиіс - пернетақтаның немесе тінтуірдің көмегімен, сондай-ақ пайдаланушы интерфейсінің өзге объектілері арқылы. Мәзір тармақтарына жылдам қатынау үшін пернелердің есте сақталатын жеңіл үйлесімін қолдануға болады, себебі бұл уақытты жақсы үнемдейді.

Формалар - интерфейстің негізгі элементі. Формалардың тағайындалуы - деректерді, күйлерін, автоматтандырылған жүйенің хабарламаларын ыңғайлы түрде енгізу және қарап шығу.

Формаларды жобалаудың негізгі принциптері:

- форма алға қойылған мақсатты біршама ыңғайлы, біршама түсінікті және жылдамдатып шешуге қол жеткізу үшін жобаланады. Егер форма қағаз формасынан ауыстырылатын болса, онда аралас өрістер бойынша жылжып қозғалуы пайдаланушы үшін қиындықтар туғызбауы керек;
- ақпараттық бірліктерді форма кеңістігіне орналастыру оны болашақта пайдалану логикасына сәйкес келуі керек: бұл ақпараттық бірліктерде қажетті қатынау бірізділігімен, оларды пайдаланудың жиілілігіне, сондай-ақ, элементтердің салыстырмалы түрдегі маңыздылығына байланысты болады;
- толтырылмаған кеңістікті пайдалану маңызды, ақпараттық формалардың теңдігі мен симметриясын құру үшін, пайдаланушының назарын қажетті бағытта белгілеу үшін;
- элементтердің логикалық топтарын бос орындармен, жолдармен, түстердің немесе өзге де көзге көрінетін құралдармен бөлу қажет;
- өзара тәуелді немесе байланысты элементтер бір формада бейнеленуі керек.

Формаларды әзірлеген кезде жүйелік мәзірдің жолағында қандай батырмалар қалай және қай терезеде қолжетімді болуы керектігі, не болмаса пайдаланушының терезелердің тақырыптарын қалай, қандай өлшемде өзгерте алатындығын ойлап, көрсету қажет.

Айрықша қажеттіліктерсіз терезелерді өзгертілген өлшемде жасау. Өлшемдер өзгерген кезде, егер оларға арнайы тәсілдер қолданылмаса, терезелердің жинақтылықтары бұзылады да, пайдаланушы өз операцияларынан ештеңе ұтпайды. Терезенің мәні оның өлшемдерін өзгеретіндей ету, егер бұл пайдаланушыға онда орналасқан бейнелеу компоненттерінің пайдалы алаңы және ақпараттарды: мәтіндерді,

бейнелерді, тізімдерді және т.с.с. редакциялауға мүмкіндік беретін болса.

Егер өзгеретін терезе жобаланып отырса, онда оның компоненттері терезеде де өз өлшемдерін немесе орналасқан жерлерін өзгерте алатындай, терезелерді алаңда бірдей орналастыру қажет және бос орындарды қалдырмайтындай шара қабылдау керек.

Басқарушы элементтер мен онымен функционалды байланысқан экран компоненттері көзбен қарағанда топтарға біріктірілуі тиіс, олардың тақырыптары анық көрініп, олардың тағайындалған мақсатын нақты түсіндіруі керек.

Әрбір терезенің оған бағынышты үйлесіммен орталық тақырыбы болуы керек. Пайдаланушы бұл терезенің не үшін арналғандығын және мұнда не маңыздырақ екендігін түсінуі керек. Терезені ақпараттарды енгізу мен бейнелеуді басқару элементтерінің көптеген сандарымен артық жүктеуге болмайды. Функциялары жағынан ұқсас басқару органдары әр терезеде әртүрлі аталуына немесе терезелердің әртүрлі орындарында аталуына болмайды. Қосымша жүйенің жұмыс кеңістігінің жалпы ұйымдасуында қалай жазылса, ол басқа қосымшалармен қалай өзара байланысатыны туралы да ойланған дұрыс.

Ақпарат саны көп формаларда тарауларды атауларын пайдалану керек, олар оған тиесілі ақпараттың сипатына сөзсіз куәлік етеді.

Тақырыптар мен енгізу өрістерінің тікелей бейнеленуін анық бөліп алу керек, себебі мұндай шатасу пайдаланушы үшін жайсыздық тудырады, тақырыптары қысқа, таныс және мазмұнды болуы керек. Толтыруға міндетті емес өрістер, не болмаса маңыздылығы жоқ өрістер маңызды және толтыруға міндетті өрістерден көзбен көргенде ажырата алатындай (түстермен немесе басқа әсерлермен) бөлініп көрсетілуі керек. Енгізу мүмкін болатын барлық өрістерге келісім бойынша мәндердің енгізілуін қамтамасыз ету керек және мұндай функция пайдаланушы үшін жағымсыз болмауы керек. Қайталанатын мәндерді енгізуге арналған пернелерді немесе кодтарды тағайындауға болады. Деректерді енгізу үшін жоғарғы және төменгі регистрлер арасында жиі ажыратып қосуды алып тастау керек. Пайдаланушыдан маңызды емес цифрларды (мысалы, 00000010 орнына пайдаланушы тек 10 енгізуі керек) енгізуін талап етуге болмайды. Осыған ұқсас пайдаланушының жүйеден автоматты түрде алуға болатын немесе алдын-ала енгізілген ақпаратты енгізуін талап етуге болмайды. Келісім бойынша мәндерді қолданған дұрыс, мұнда олар ақпараттарды енгізу процесін азайтуға мүмкін болады.

Навигация жүйесін және күйлерді көрсету жүйесін ұйымдастыру. Навигация пайдаланушыға түрлі экрандар арасында, ақпараттық бірліктер мен автоматтандырылған жүйедегі қосалқы бағдарламалар арасында жылжу қабілетін қамтамасыз етеді. Толыққанды жүйеде

пайдаланушы жүйенің күйі, қосалқы бағдарлама немесе орындалу процесі туралы ақпаратты ала алады.

Пайдаланушыға жүйеде бағдарлап жүру үшін көмек болатын навигациялық құралдар мен тәсілдердің қатары бар:

- әрбір экран үшін беттердің тақырыптарын қолдану;
- беттердің нөмерлерін: жолдардың және бағандардың нөмірлерін қолдану;
- экранның жоғарғы жағында файлдың ағымдағы атын көрсету.

Навигация жүйесінің типі қабылданған интерфейс стиліне байланысты болады. Командалар тілінің интерфейсі үшін толыққанды навигацияны қамтамасыз етудің тәсілдері өте аз. Мәзірі бар интерфейстерде иерархиялық-құрылымдалған мәзірді қолдануға болады. Диалогты интерфейстер олар пайдаланушыны қате әрекеттерден қорғайды. Күй ақпараты әдетте экранның төменгі жағынан көрсетіледі де, жазбалардың саны, іріктеліп алынған бірліктердің саны, басып шығару процесін, басып шығару кезегі деректерінен және т.б. тұрады.

Хабарламалар пайдаланушының әрекетін қажетті жолға бағыттау, міндетті шешу жолында қажетті әрекеттерді орындауға арналған көмек ескертпелер үшін қажетті. Олар сондай-ақ, пайдаланушы тарапынан әрекеттерді растауды және жүйенің сәтті орындауы, не болмаса қандай да бір себептермен орындалмағандығын растауды қамтиды. Хабарламалар диалог, экрандық бет басы түрінде және т.с.с. шығарылуы мүмкін.

Хабарламалар пайдаланушыға мынаны ұсына алады: ұсынылған баламалардан бір опцияны немесе опциялар жиынтығын таңдау; бір ақпаратты енгізу; кез келген өзге әрекетті қабылдамас бұрын пайдаланушыға сауалдарға жауап беруге мәжбүрлеу. Бұл жүйе пайдаланушының жұмысты жалғастырмас бұрын шешім қабылдауына мәжбүрлеген кезде пайдалы болуы мүмкін. Модальды емес диалогты терезелер интерфейстің басқа элементтерімен жұмыс істеуге мүмкіндік береді, ал осы уақытта ол өзі елемеуі мүмкін.

Пайдаланушы интерфейсін қолданудың ыңғайлылығы - осы интерфейстің көмегімен, оны қолданумен, кіріс деректерін даярлаумен және шығыс деректерінің интерпретациясы көмегімен бағдарламалық жүйемен жұмыс істеу принциптерін зерделеуге қажетті күш-жігерді айқындауға мүмкіндік беретін сапа көрсеткіші. Әйтпесе, басқаша айтқанда, пайдаланушының адам-машина (пайдаланушы) интерфейсі арқылы берілетін жүйенің функцияларына қарапайым қатынау дәрежесін айқындайтын пайдалану ыңғайлылығы.

БАҚЫЛАУ СҰРАҚТАРЫ МЕН ТАПСЫРМАЛАРЫ

1. Сәулет ұғымы мен оның сипаттамасының міндеттерін түсіндіріп беріңіз.
2. Бағдарламалық құралдардың сәулетінің негізгі кластарын атаңыз.
3. Бағдарламаларды әзірлеудің өрлеуші және төмендеуші әдістерінің артықшылықтары мен кемшіліктері қандай?
4. Орындау қателері қалай шығады?
5. Қателер бағдарламаны өңдеу сатылары бойынша қалай жіктеледі?
6. Есептеу нәтижелері қателерінің жиналуына не жатады?
7. Индукция әдісін пайдалана отырып ретке келтіру қалай өтеді?
8. Қандай жағдайларда бағдарламаларды ретке келтіргенде кері қадағалау әдісі қолданылады?
9. Дедукция әдісін пайдалана отырып ретке келтіру қалай өтеді?
10. Бағдарламалық қамсыздандырудың пайдаланушы интерфейсіне қандай негізгі талаптар қойылады?

7-бөлім

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ САПАСЫ

7.1. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУ САПАСЫНЫҢ СИПАТТАМАЛАРЫ

Бағдарламалық қамсыздандыруды әзірлеу – бұл, ең алдымен, сапалы бағдарламалық өнім алу тәсілдерінің бірі.

БҚ сапасын бағалау жөніндегі әзірлеу процесі мен қызметіне келесі БҚ келесі жалпы көрсеткіштері ықпал етеді:

- БҚ қолдану және тағайындау саласы;
- шешілетін міндеттердің типі;
- БҚ көлемі және күрделілігі;
- БҚ сапасының сипаттамаларына қажетті құрамы мен талап етіетін мәндері;
- шешілетін міндеттердің нақты уақыт масштабы немесе шешілетін міндетті күту нәтижелерінің ықтимал ұзындығымен байланыс дәрежесі;
- БҚ көптеген нұсқаларын пайдаланудың ұзақтылық және құру перспективалары;
- БҚ қажетті құжаттау дәрежесі.

МемСТ Р ИСО/МЭК 9126-9 анықтамаларына сәйкес келесі негізгі терминдерді қолданатын боламыз.

Атрибут - өнімнің өлшенетін нақты немесе дерексіз қасиеті. Атрибуттары сыртқы немесе ішкі болуы мүмкін.

Сыртқы қасиеті - өнім берілген жағдайларда қолданудың белгіленген және ойлап алынған қажеттіліктерін қанағаттандыратын дәреже.

Сыртқы шамасы - жүйенің бөлігі болып табылатын әрекет шараларынан алынған өнімнің жанама шамасы.

Ішкі қасиеті - өнімнің берілген жағдайларда қолданылған кезде белгіленген және ойлап алынған қажеттіліктерін айқындайтын атрибуттарының толық жиынтығы.

Ішкі шамасы - тура немесе жанама өнімнің меншік шамасы.

Өлшем - өнімнің атрибутына шкаладан мәнін (саны немесе санатын)

беру үшін метриканы пайдалану.

Шама - өлшем арқылы өнімнің атрибутына берілетін саны немесе санаты.

Метрика - белгілі бір әдіс және өлшеу шамасы. Метрикалар ішкі, сыртқы немесе пайдаланудағы метрикалар, тура немесе жанама болуы мүмкін.

Сапа моделі - сапаны бағалауға және сапаға қойылатын талаптарды анықтау үшін негізді қамтамасыз ететін, олардың арасындағы сипаттамалар мен байланыстардың жиынтығы.

Сапаны бағалау - өнім аталған талаптарды орындауға қабілетті болатын дәреженің жүйелік зерттелуі.

БҚ сапасының сипаттамасы - БҚ қасиеттерінің жиынтығы, оның көмегімен сапасы сипатталады және бағаланады.

Бірқатар метрика көмегімен бағалануы мүмкін атрибуттар немесе сипаттамалардың жиынтығы ретіндегі БҚ сапасын анықтау кеңінен қолданылады. Мұндай тәсіл жалпы БҚ сапасын құрылымдық түрде бағалауға, барлық қажетті қырларда бағалауға мүмкіндік береді. Ол стандартты сапа моделі ISO 9126 негізінде қабылданған болатын. Бұдан әрі қазіргі таңда осы стандарттың қолданыстағы нұсқасында сипатталған БҚ сапа моделі көрсетілетін болады.

Бағдарламалық қамсыздандырудың сапасы - бұл бағдарламалық қамсыздандыру сипаттамаларының жиынтығы, оның талаптарға сәйкестілік дәрежесін көрсетеді. Бұл ретте талаптар кең ауқымда түсіндіріліп берілуі мүмкін.

ISO 9126 стандарты БҚ сапасын қарастырғанда кезде үш қырынан алып қарауды ұсынады:

- БҚ ішкі сапасын қабылдайтын әзірлеушілердің көзқарасынан;
- БҚ сыртқы сапасы айқындалатын барыста оған қойылатын талаптарға қалыптастырылған сәйкестікте БҚ басшылығы мен аттестация көзқарасынан;
- Пайдалану кезінде БҚ сапасын сезінетін пайдаланушылардың көзқарасынан.

Барлық үш жағдайларда сапаны сипаттау үшін көп деңгейлі модель қолданылады, ол сапа мақсаттарынан (факторларынан), атрибуттарынан (белгілерінен) және метрикаларынан тұрады. Мақсаттар (факторлар) жоғарғы деңгейде БҚ бар немесе болуы керек негізгі сипаттамаларын анықтауға мүмкіндік береді. Әрбір фактор күтілетін немесе алынған сипаттамаларын біршама егжей-тегжейлі сипаттамаларын сапалы түрде сипаттауға мүмкіндік беретін атрибуттар (белгілер) жиынтығынан тұрады. Әрбір атрибут тиісті сипаттамаларының болуын сандық мағынада бағалауға мүмкіндік беретін метрика жиынтықтарын қолдайды (7.1-сурет).

1. ***Функционалдық мүмкіндіктері (функционалдық***

жарамдылығы) - техникалық тапсырмаға және тапсырыс берушінің талаптарының сипаттамасына немесе әлеуетті пайдаланушыға сәйкес келетін бағдарламалық құралдың тағайындалу мақсатын, номенклатурасын, негізгі, қажетті және жеткілікті функцияларын айқындайтын сипаттамаларының жиынтығы:

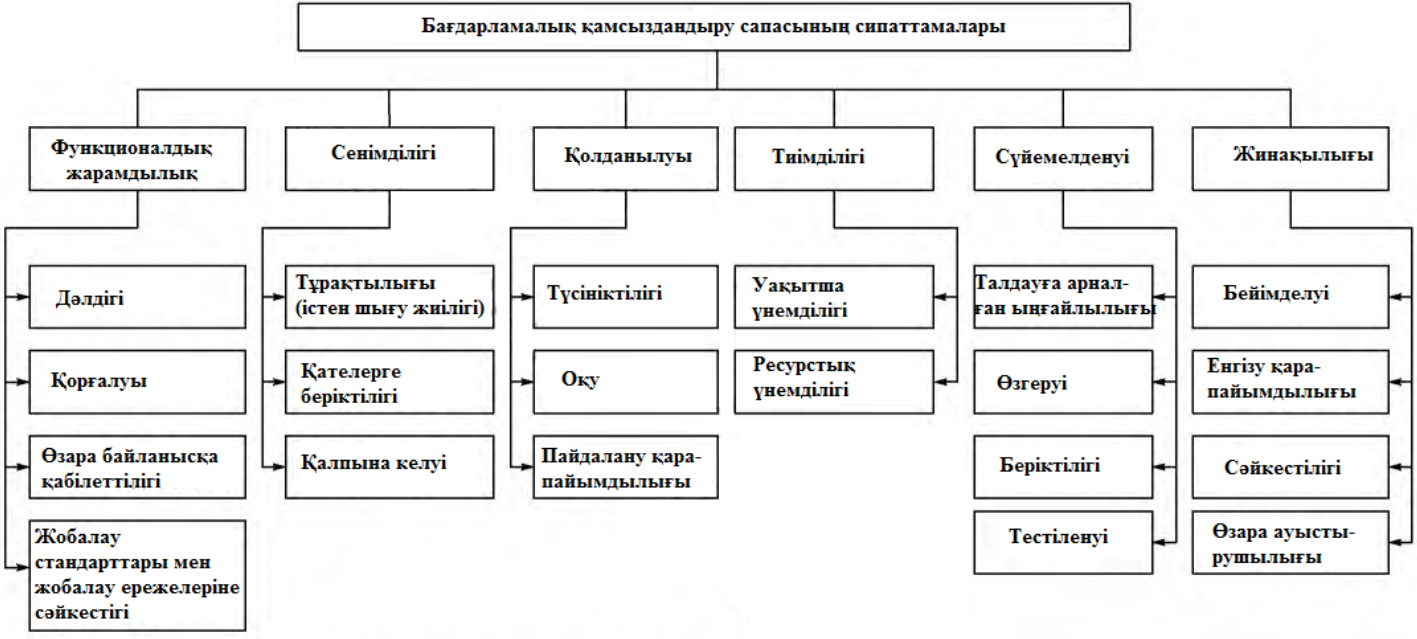
- қорғалу - бағдарламалық құралдың компоненттерінің бағдарламаны және ақпаратты кез келген теріс әсерден қорғау қабілеті;
- дәлдік - пайдаланушы үшін дұрыс немесе қолайлы нәтижелері мен сыртқы әсерлерін қамтамасыз етуге бағдарламалық құралдың қабілеті;
- өзара әрекет етуге қабілеттілігі - ішкі және сыртқы ортаның бір немесе бірнеше компоненттерімен өзара әрекеттесетін бағдарламалық құралдар мен олардың компоненттерінің қасиеті;
- келісушілік - жобалау стандарттары мен ережелеріне сәйкестілік.

2. ***Сенімділік*** - бұл берілген режимдерде және қолдану шарттары мен техникалық қызмет көрсетуде талап етілген функцияларды істен шықпай үздіксіз орындау қажеттілігі бар бағдарламалық қамсыздандырудың қабілеті. Сенімділік бағдарламалық қамсыздандырудың тірек көрсеткіші болып табылады:

- тұрақтылық - бағдарламалық қамсыздандыруда қателер болғанда істен шығудың жиілігі;
- қателерге беріктілік - бағдарламалық қателер немесе белгілі бір интерфейс бұзылғанда жұмыс істеу сапасының белгілі бір деңгейін ұстау қабілеті;
- қалпына келушілік - істен шыққаннан кейін қажетті ресурстар мен уақыт үшін деректердің тұтастығы мен жұмысқа қабілеттілігінің белгілі бір деңгейін қалпына келтіру қабілеті.

3. ***Қолданылушылық (қолдану ыңғайлылығы)*** - бағдарламалық құралды түсіну. Зерделеу және пайдалану, сондай-ақ білікті пайдаланушылар үшін аталған жағдайларда қолданғанда тартымдылығын негіздейтін бағдарламалық құралдың қасиеттері:

- түсініктілік - пайдаланушының жалпы логикалық концепциясы мен оның қолданылуын түсіну дәрежесі;



7.1-сурет.Бағдарламалық қамсыздандырудың сапа көрсеткіштері

- оқытылу - пайдаланушының бағдарламалық қамсыздандыруды қолдануды оқыту жөніндегі күш-жігерінің дәрежесі (мысалы, шұғыл басқару, енгізу, шығару);
- қолдану қарапайымдылығы - пайдаланушының пайдалану және шұғыл басқару жөніндегі күш-жігерінің дәрежесі.

4. ***Тиімділігі (өнімділігі)*** - белгіленген жағдайларда қолданылатын есептеу ресурстарының санын есепке ала отырып, функционалдық міндеттерді шешудің талап етілген өнімділігін қамтамасыз ететін бағдарламалық құралдың қасиеттері:

- уақытша үнемділігі - шақыру және өңдеу уақыты, сондай-ақ бағдарламалық қамсыздандыру функцияларын орындау жылдамдығы;
- ресурстарды пайдалану тиімділігі - белгілі түрдегі ресурстардың берілген көлемін пайдалана отырып, қажетті міндеттерді шешу қабілеті. Шұғыл және ұзақ уақытқа жады сияқты ресурстармен бірге желілік қосылу, енгізу және шығару құрылғылары бар және басқалары.

5. ***Сүйемелденуі*** - бағдарламалық құралдың конфигурацияларын және функцияларын өзгертуге және түрін өзгертуге қабілеттілігі:

- талдануы (талдауды өткізу ыңғайлылығы) - қателерге, ақаулар мен кемшіліктерге талдау жүргізу ыңғайлылығы, сондай-ақ, өзгерту қажеттілігі мен ықтимал салдарын талдау қажеттілігі;
- өзгертулерді енгізу қолайлылығы - қажетті өзгертулерді орындауға жұмсалатын көрсеткіш, кері еңбек шығындары;
- тұрақтылық (беріктілік) - қажетті өзгертулерді енгізу кезінде кездейсоқ тиімділіктердің пайда болуының кері тәуекелі, көрсеткіші;
- тестіленуі (тексеру қолайлылығы) - енгізілген өзгертулер қажетті нәтижелерге тексеру түрлері мен тестілеуді жүргізуге кері еңбек шығындары, көрсеткіші.

6. ***Жинақылық*** - бағдарламалық қамсыздандырудың бір шеңберден басқасына ауысуы мүмкін қабілеті:

- бейімделгіштігі - бұл нақты пайдалану жағдайларына бағдарламалық қамсыздандырудың бейімделу қабілеті, осы қарастырылып отырған бағдарламалық қамсыздандыру үшін арналған өзге әрекеттер немесе тәсілдерді қолданусыз;
- енгізу қарапайымдылығы - бұл нақты айналада бағдарламалық қамсыздандыруды енгізуге қажетті күш-жігердің дәрежесі;
- сәйкестілігі - бағдарламаның жинақтылыққа қатысты стандарттар немесе келісімдерге бағынуын тудыратын бағдарламалық қамсыздандырудың атрибуттары;

- өзара алмасушылық - белгілі бір қоршаған айналада міндеттерді шешу үшін нақты бағдарламалық құралдың орнына басқа бағдарламалық қамсыздандырудың қолданылуының еңбек сыйымдылығы дәрежесі сипатталады.

Берілген сапа атрибуттары стандартта бекітілген, бірақ, бұл бағдарламалық қамсыздандыру сапасының ұғымын толық алып тастайды. Аталған сипаттамалар кейбір мамандар үшін осы стандарттарда сипатталған біршама күрделі болып көрінеді.

7.2. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУ САПАСЫНЫҢ МЕТРИКАЛАРЫ

Бағдарламалық өнімнің сапасын бақылау ең түрлі деңгейлерде - әдіснамалық деңгейден басталып, технологиялық деңгеймен аяқтап, сапаны бақылау процестері автоматты режимде өтсе, мысалы, жобаны автоматты жинау кезінде жүзеге асырылуы тиіс. Алайда, кез келген бақылау өлшеудің бар екендігін болжайды, олар бағдарламалық жобаның осы не басқа сапасына қол жеткізілгендігін бағалауға мүмкіндік береді.

БҚ әзірлеу процесінде өлшеулер ұйымда қабылданған әзірлеу процесінің өзін, жобаны орындау барысы мен бағдарламалық өнімнің сапасын бағалау үшін орындалады. Процесс өлшемдері бұдан әрі жетілдіруге ықпал етеді, жобаны өлшеу жұмыстарын ұйымдастыруды жақсартады, ал бағдарламалық өнімді өлшеу оның сапасын арттыруға көмектеседі. Өлшеу нәтижесінде өлшеу объектісінің қандай да бір қасиетінің сандық сипаттамасы анықталады.

Өлшеу жүйесі бағдарламалық қамсыздандыру сапасын сандық бағалау үшін қолданылатын өлшем модельдері мен метриканы қамтиды.

Метрика сандық мәні бар қасиетті игеру дәрежесінің шарасы болып табылады. *Метрикалар* деп өлшеу немесе болжам орындалатын негізде тікелей қолданылатын бағдарламалық өнімді, процесті немесе жобаны сандық бағалауды түсінеді.

Метрикалардың маңыздылығы шешім қабылдаудың қандай шамада болуымен анықталады. Егер бағдарлама жобасының басшысы бұл туралы ұмытпаса, онда ол маңызды және пайдалы метрикаларға сүйене алады, оларды кездейсоқ жинамайды, пайдаланылуы қиын ақпараттың үлкен көлемін жинайды.

Тікелей өлшеу арқылы объектінің тірек қасиеттері - тірек метрикалары ғана анықталуы мүмкін. Басқа барлық метрикаларды тірек

метрикалардың міндерінен осы немесе басқа функцияларды есептеу нәтижесінде бағаланады. Бұл есептеулер тиісті формулалар бойынша өткізіледі.

Барлық метрикаларды үш негізгі топқа ажыратуға болады:

1) процесс метрикасы;
2) жоба метрикасы;
3) өнім метрикасы.

Әрбір топтың ішінде келесі метрика типтері болады:

- тікелей қадағаланатын (өлшенетін);
- болжанатын;
- есептелетін.

Қандай да бір нысан атрибутының *тікелей қадағалануы* өзге атрибуттардың немесе нысандардың өлшеу процесінде пайдалануды талап етпейді. Тікелей қадағалау немесе өлшеу қолданыстағы нысанды бағалау кезінде қолданылады.

Болжау кезінде белгісіз параметрлерді және нәтижелелердің интерпретациясын анықтау үшін қолданылатын болжау процедураларын жиынтығымен қатар таңдап алынған атрибуттың математикалық моделі қолданылады.

Есептеу немесе жанама өлшеу, өлшеу процесіне өзге атрибуттар мен нысандардың белгілі бір математикалық моделінің көмегімен өлшеу процесіне тартуды білдіреді (әрдайым пайдаланумен есептеуді қосады, ең кем дегенде, екі өзге метрика). Метрикалар келесі типтер бойынша сыныпталады:

- оның сипаттамалары - қасиеттері өлшенген кезде қолданылатын бағдарламалық өнім метрикалары;
- процесс метрикалары (өнімді құру арқылы ӨЦ процесінің қасиеттерін өлшеу);
- пайдалану метрикасы.

Бағдарламалық өнім метрикасы мыналарды қамтиды:

- пайдаланушыға көрінетін өнімнің қасиеттерін белгілейтін сыртқы метрикалары;
- әзірлеушілер командасына ғана көрінетін қасиеттерді көрсететін ішкі метрикалары.

Сыртқы метрикалар (атрибуттар) сыртқы ортамен нысанның байланысын есепке ала отырып, бағаланады. Өнімнің сыртқы метрикалары - мына метрикалар:

- ақау санын анықтауға қызмет ететін өнімнің сенімділігі;
- өнімдегі функциялардың болуы мен дұрыс іске асуы анықталатын функционалдылық;
- өнімнің ресурстары (жылдамдығы, жады, аясы) өлшенетін сүйемелдеу;

- өнімнің қолжетімділік дәрежесін зерделеу және пайдалану үшін анықтауға ықпал ететін өнімнің қолданылуы;
- жасалған өнімнің құны анықталатын құн.

*Ішкі метрикалар (*атрибуттары) нысанның өзінің терминдерінде, оның әрекетінен тыс өлшенуі мүмкін. Ішкі метрикалардың мысалдары бағдарламалық өнімдегі код жолының санының көрсеткіші болып табылады, әрекеттерді орындау ұзақтығы, еңбек шығындарының шамасы, сәтсіз тестілік сынақтардың саны, модульділік дәрежесі мен күрделілік деңгейі. Өнімнің ішкі метрикалары мыналарды қамтиды:

- өнімді оның ішкі сипаттамаларының көмегімен өлшеу үшін қажетті өлшем метрикалары;
- өнімнің күрделілігін анықтауға қажетті күрделілік метрикалары;
- өнімнің жеке компоненттерін құру тәсілдері мен технологияларын және оның құжаттарын анықтауға қызмет ететін стиль метрикалары.

Сыртқы және ішкі метрикалар БҚ қойылатын талаптарды қалыптастыру сатысында беріледі және соңғы бағдарламалық өнімнің сапасына қол жеткізуді жоспарлау мен басқарудың тақырыбы болып табылады. ИСО/МЭК 9126 стандарты келесі шама типтерін анықтайды:

- бағдарламалық қамсыздандырудың түрлі өлшем бірліктеріндегі өлшем шамасы (функцияларының саны, бағдарламадағы жолдардың саны, дискілік жады өлшемі және басқалары);
- уақыт шамасы (жүйенің жұмыс істеуі, компоненттің орындалуы және басқалары);
- күш шамасы (еңбек өнімділігі, еңбек сыйымдылығы және басқалары);
- есепке алу шамасы (қателердің саны, істен шығу саны, жүйелердің жауап саны және басқалары).

Арнайы шама жалпы жүйе өлшеміне дайын компоненттерден жасалған өнімнің қатынасы сияқты қайталанатын компоненттерді пайдалану деңгейі ретінде қызмет етедің. Бұл шама бағдарламалық қамсыздандырудың құны мен сапасын анықтауда қолданылады. Метрикалардың мысалы ретінде мыналарды келтіруге болады:

- нысандардың жалпы саны және қайталанып қолданылатындардың саны;
- қайта қолданылатын және жаңа операциялардың жалпы саны;
- сипаттамасы тән операцияларды алатын класс саны;
- осы класс тәуелді болатын класс саны;
- класс немесе операциялардың пайдаланушылар саны және басқалары.

Кейбір шамалардың жалпы санын бағалау кезінде орташа статистикалық метрикалар жиі қолданылады (кластағы операциялардың орташа саны, класс немесе класс операцияларының саны және

басқалары).

Бағдарламалардың ішкі метрикаларын кеңінен қолданылатын үлгілері Холстед метрикасы - нақты пайдалану тілінде бағдарламаның статикалық құрылымы негізінде анықталатын бағдарлама сипаттамасы: біршама жиі кездесетін операндтар мен операторлардың кірген саны; бағдарламаны сипаттау ұзындығы барлық операндтар мен операторлардың кірген санының сомасы ретінде бағдарламаны сипаттау және басқалары. Сыртқы метрикалар ретінде операциялар арасындағы өзара байланыстың орындалу уақытын (бағдарлама мен компьютер талап етіледі), көрсету пайдалылығы мен ыңғайлылығы (қосымша және пайдаланушы талап етіледі), сенімділігі, тиімділігі, тестіленуі, қайта қолданылуы, ауыстырылуы мен өзара байланысын қарастыруға болады.

Процесс метрикасы ретінде әзірлеу уақыты, тестілеу сатысында табылған қателердің саны және басқалары болуы мүмкін. Келесі процесс метрикалары қолданылады деуге болады:

- жалпы әзірлеу уақыты және әрбір саты үшін жеке уақыт;
- модельдерді модификациялау уақыты;
- процесте жұмысты орындау уақыты;
- тексеру кезінде табылған қателердің саны;
- сапасын тексеру құны;
- әзірлеу процесінің құны.

Қолдану метрикалары пайдаланушының міндеттерін шешу кезінде талаптарына қанағаттану дәрежесін өлшеуге арналған. Олар бағдарламаның қасиеттерін ғана емес, оны пайдалану нәтижелері - пайдалану сапасын бағалауға көмектеседі. Мысалы, пайдаланушының міндеттерін шешудің толықтығы мен дәлдігі болуы мүмкін, сондай-ақ жұмсалған ресурстар (еңбек шығындары, өнімділігі және басқалары) пайдаланушының міндеттерін тиімді шешуге жұмсалған. Пайдаланушының талаптарын бағалау сыртқы метрикалардың көмегімен жүргізіледі.

Метрикалар объективті және субъективті болады. Субъективті өлшемдер жеке субъективтік тәсілдің болуын болжайды, мысалы, қандай да бір салмақ коэффициентін қолдану.

7.3. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ СЕНІМДІЛІГІ

Сенімділік бағдарламалық өнім сапасының маңызды сипаттамасы болып табылады. Бағдарламалық қамсыздандырудың сенімділігін азайту талаптарында, жобалауда және орындаудағы қателердің салдарынан болады. Бұзылу мен қателер өнім өндірісінің тәсіліне байланысты және бірқатар уақыт аралығында олар орындалған кезде бағдарламаларда шығады. Осылайша, бағдарламалық қамсыздандырудың сенімділігін бағалау бағдарламада қалған және жойылмаған қателердің санына байланысты.

Бағдарламалық қамсыздандыруды пайдалану барысында қателер табылады да, жойылады. Егер қателерді түзеткен кезде жаңалары енгізілмесе, немесе кем дегенде жаңа қателер жойылғанына қарамастан, аз енгізіледі, онда пайдалану барысында бағдарламалық қамсыздандырудың сенімділігі үздіксіз артады. Пайдалану қаншалықты қарқынды жүргізілсе, қателер де қарқынды анықталады және жүйенің сенімділігі мен тиісінше оның сапасы жылдамырақ артады.

Істен шығу және бұзылулар

Сенімді бағдарламалық қамсыздандыру онда ақаулардың немесе қателердің болуын жоққа шығармайды, тек осы қателер бағдарламалық құралды практикалық қолдану барысында жеткілікті түрде сирек шыққаны маңызды. Бағдарламалық қамсыздандыру оны тестілеу кезінде сынау барысында, сондай-ақ практикалық қолданысында ие болатын қасиеті бар екендігіне көз жеткізу керек.

Жұмыс істеп тұрған бағдарламалық жүйеде ақаулардың болуы түрлі салдардың пайда болуына және шекті қиын жағдайларға себеп болуы мүмкін (мысалы, қаншалықты жүйе жұмысқа қабілетті емес күйінде болуы мүмкін). Бағдарламалық қамсыздандырудағы кейбір қателер оны қолданған кезде кейбір ыңғайсыздықтарды тудырады, ал басқа қателердің апатты салдары болуы мүмкін. Сол себепті де бағдарламалық қамсыздандырудың сенімділігін бағалау үшін кейде әрбір іркіліске немесе бұзылуға пайдаланушы үшін құнын есепке алатын, қосымша көрсеткіштерді қолданады.

Істен шығу - бұл бүкіл жүйе немесе оның бір бөлігі жұмысқа қабілетті күйден шығатын оқиға. Жүйе оның жұмысқа қабілеттілік күйін қалпына келтіру мүмкін болмайтын, не болмаса біршама уақыт

алатындай істен шығуы. Бағдарламалық жүйелер үшін істен шығудың себебі жасырын ақау болуы мүмкін, ол уақыт өте келе ғана көрінеді (мысалы, уақыттың ішкі санауышының шамадан тыс толуы, деректердің шамадан тыс толуы және т.с.с). Істен шығу жүйенің жұмыс істеуін ұзақ уақытқа бұзады немесе оны ең шекті күйіне жеткізеді.

Шекті күйі - бұл жүйені бұдан әрі пайдалану мүмкін болмайтын немесе оның жұмысқа қабілетті күйін қалпына келтіру мүмкін болмайтын, не болмаса мақсатқа сай болмайтын күйі.

Бұзылу - бұл өздігінен жөнделетін немесе бір ретті бұзылу, ол оператордың кірісуімен жойылады (МемСТ 27.002-89 «Техникадағы сенімділік. Негізгі ұғымдар. Терминдер және анықтамалар»).

Бұзылу уақыт жағынан ұзақтығы шағын және қалпына келтірудің қосымша процедураларынсыз жойылуы мүмкін. Негізінде, бұзылу пайдаланушылардың бүкіл жүйенің тұтастай жұмысының тоқтауынсыз деректердің біразын жоғалтуын, не болмаса қысқа уақытқа тоқтауын болдырады. Бұзылу салдары пайдаланушының көзқарасымен алғанда маңызды болуы мүмкін, әсересе, егер олар нағыз маңыз болса, алайда жүйенің үздіксіз жұмысы бұзылмайды.

Бұзылу жүйенің жұмысқа қабілеттілік күйі уақытша бұзылатын жағдайлардың себептері болып табылады. Істен шығулар жүйенің жұмысқа қабілеттілігі біржола немесе ұзақ уақытқа бұзылатын күйлерінің себептері болып табылады.

Келесі іркіліс түрлерін ажыратады:

- *жүйелік бағдарламалық қамсыздандырудағы іркілістер.* Негізінен, бұл іркілістердің салдары ең ауыры болады. Кейбір жағдайларда жүйенің деректері толық жоғалатыны сияқты, жүйенің күйі туралы деректер іркіліс кезінде толық жоғалуы мүмкін. Мұндай жағдайлар диагностика мен түзету үшін ең күрделі;
- *қолданбалы бағдарламалық қамсыздандырудағы іркілістер* қолданбалы жүйені жеткілікті дәрежеде тестілемегенде, не болмаса оны штаттан тыс қолданған кезде пайда болады. Негізінде, мұндай іркілістер туралы ақпараттарды жинау қосымшалардың өзінің құралдарымен мүмкін. Қиын жағдайларда қосымша жұмыс істейтін басқарудағы операциялық ортаның құралдарымен ақпараттық ортасы туралы мәліметтерді жинау мүмкін;
- *жүйемен жұмыс істегенде пайдаланушының көзделмеген әрекеттерінің салдарынан пайда болатын іркілістер.* Мұндай іркілістердің себептерін жою бірнеше бағыттар бойынша жүргізілуі мүмкін. Бұл, пайдаланушы нұсқаулығын мүмкін толықтыру болмақ немесе жүйенің пайдаланушы интерфейсін өзгерту (оны біршама ыңғайлы және түсінікті ету).

Бағдарламалық қамсыздандырудың істен шығуының пайда болуы әртүрлі себептермен болуы мүмкін:

- *ресурстың тоқтап қалуы* - нәтижесінде жүйе шекті күйіне жететін тоқтап қалу. Мұндай тоқтап қалу ең алдымен жүйенің жұмысына арналған ресурстардың жетіспеушілігінен (мысалы, диск кеңістігі) болады. Мұндай тоқтап қалуды болдыратын жағдайлар жүктеу тестілеуінде модельденуі мүмкін;
- *конструктивті тоқтап қалу* - белгіленген жобалау және конструкциялау қағидалары және/немесе нормаларының бұзылуымен немесе жетілдірілмеуіне байланысты себептер бойынша туындаған тоқтап қалу. Анықтау және тестілеу процесі ең алдымен конструктивтік тоқтап қалудан болған ақауларды анықтауға бағытталған;
- *өндірістік тоқтап қалу* - жүйенің өндіріс процесін немесе сүйемелдеу процесін бұзуға байланысты тоқтап қалу. Өндірістік тоқтап қалу пайда болуы мүмкін, мысалы, жүйені баптау фильтрлері жоғалғанда, оның нәтижесінде жүйе жабдықтың ағымдағы баптауымен үйлеспейтін режимге өтеді. Өндірістік тоқтап қалуды болдырмау үшін пайдалану құжаттамасын және сүйемелдеу құжаттамасын дұрыс құру қажет;
- *пайдалануда тоқтап қалу* - пайдалану ережелерін бұзуға байланысты тоқтап қалу. Бұл тоқтап қалудың түрлерін шығаратын себептер ең алдымен, адами факторларға байланысты болады. Сондықтан мұндай тоқтап қалуды анықтаудың негізгі тәсілдері- пайдалану құжаттамасын тексеру, пайдаланушының басым қателерін блоктайтын қорғаныс механзидмрені жүйеге енгізу.

Уақытша сипаттамалары бойынша тоқтап қалу келесі түрде жіктеледі:

- *кенеттен тоқтап қалу* - бағдарламалық қамсыздандырудың немесе өңделіп жатқан деректердің параметрлерінің бірін өзгертуден кенеттен қатты өзгеруіне байланысты болатын тоқтап қалу. Мұндай тоқтап қалуды тудыратын жағдайлар жүктеме тестілеу барысында жүйеге жүктеме деңгейін кенеттен арттыру арқылы келесіде жүктемені жылдам тұрақтандырумен модельдендіріледі (мысалы, бір уақытта қосылған пайдаланушылардың саны);
- *біртіндеп тоқтап қалу* - жүйенің немесе өңделетін деректердің параметрлерінің бірін біртіндеп өзгертуге байланысты туындаған тоқтап қалу. Мұндай тоқтап қалу жүйенің күйі туралы ақпаратты сақтайтын буфер шамадан тыс болғанда пайда болуы мүмкін;
- *кезектесіп тоқтап қалу* - бір және осы сипаттағы істен шығудың бірнеше рет өздігінен жойылуы. Бұл жағдайда бағдарламалық

қамсыздандырудың жүйелі түрде пайда болатын ақаулары туралы сөз болғандықтан, онда істен шығу туралы айту қажет.

Істен шығу анық және жасырын болуы мүмкін. Көзбен көріп байқалатын немесе штаттық әдістермен және диагностикалау, бақылау құралдарымен БҚ қолдануға даярлау барысында анықталатын істен шығу немесе оны мақсатына қарай қолдану процесінде *анық істен шығу* деп аталады.

Жасырын істен шығу - бұл көзбен көріп байқалмайтын немесе штаттық әдістермен және бақылау, диагностикалық құралдарымен және бақылау құралдарымен анықталмайтын, бірақ техникалық қызмет көрсетуді жүргізгенде немесе арнайы диагностика әдістерімен анықталатын істен шығу.

Болашақта істен шығуды болдау үшін бағдарламалық қамсыздандыруды пайдалану уақыты ішіндегі ақауларды шығу жиілігі туралы нақты деректер қажет.

Бағдарламалық қамсыздандырудың сенімділігін бағалау үшін статистикалық көрсеткіштер қолданылады, мысалға үздіксіз жұмыс уақыты мен ықтималдылығы, істен шығу мүмкіндігі мен жиілігі (қарқындылығы). Істен шығу себептері ретінде тек өздігінен жойылмайтын бағдарламадағы қателер ғана қарастырылатындықтан, онда бағдарламалық қамсыздандыруды жүйенің қалпына келтірілмейтін класына жатқызу керек.

Түрлі әзірлеу сатыларында бағдарламалық қамсыздандырудың сенімділігін қамтамасыз ету

Практикада бағдарлама сапасын тек жетілдірілген түрде ғана емес, сонымен қатар оларды әзірлеу процесінде бағалау маңызды және әзірлеу нысандарының сапасы оларды құрғанда және басқаруда технологиялық процестердің жоғарғы сапасы есебінен қол жеткізіледі. Бағдарламалық өнімнің сенімділігін жобаны әзірлеудің түрлі сатыларында жоспарлау керек. Әзірленетін бағдарламалық қамсыздандыру түрлі сенімділік дәрежесіне ие болуы мүмкін. Сенімділікті зерделеу үшін күрделі аналитикалық әдістемелерді қолдану жеткілікті, үлкен көлемде ақпарат жинау қажет болады, себебі бағдарламалық қамсыздандырудың өмірлік циклы кезінде сенімділікті өлшеу әдістері жасалады.

Сенімділігі жоғары бағдарламалық өнімді жасау үшін қажетті:

- қателерді болжау (сенімділік модельдерін құру);
- қателердің алдын алу (бағдарламаларды қайта қолдану, бағдарламаларды, формалды әдістерді конструкциялаудың аспаптық құралдарын қолдану);

- қателерді жою (формалды тексеру, анықтау және аттестаттау);
- істен шығу беріктілікті қамтамасыз ету (мониторинг, артық және айрықша жағдайларға талдау).

Қателерді болжау талаптарды жоспарлау және қалыптастыру сатысында орындалады.

Қателердің алдын алу - талаптарды қалыптастыру, жобалау және іске асыру сатыларында.

Қателерді жою - жобалауды, іске асыру және тестілеу сатыларында.

Істен шығуға беріктілікті қамтамасыз ету жобалау сатысында басталады және бағдарламалық өнімнің өмірлік циклының аяқталғанына дейін жалғасады (7.1-сурет).

Бағдарламалық қамсыздандыру сапасының қажетті деңгейін қамтамасыз ететін іс-шараларды әкімшілік және технологиялық деп шартты түрде ажыратуға болады.

Әкімшілік іс-шараларға мыналар жатады:

- қызметкерлерді оқыту, қайта даярлау және біліктіліктерін арттыру;

7.1-кесте. Бағдарламалық қамсыздандырудың өмірлік циклы сатыларымен сенімділікті қамтамасыз ету әдістерінің байланысы

БҚ өмірлік циклының сатылары	Қателерді болжау	Қателердің алдын алу	Қателерді жою	Істен шығуға беріктілігін қамтамасыз ету
Талаптарды талдау және қалыптастыру	+	+		
Жобалау және іске асыру		+	+	+
Тестілеу			+	+
Пайдалану және сүйемелдеу				+

- бағдарламалық қамсыздандыру құрылымындағы барлық өзгертулерді құжаттау;
- бағдарламалық қамсыздандыруды әзірлеудің және модификациялаудың әртүрлі учаскелеріне жауапты тұлғаларды тағайындау;
- бағдарламалық қамсыздандырудың сапасын бақылауды ұйымдастыру. Сапа мониторингін қамтамасыз ету (қателерді, бағдарламалық қамсыздандыру пайдаланушыларынан келіп түсетін апаттық хабарламаларды жазып алу);
- БҚ тексеруді жүргізу үшін тәуелсіз сараптама комиссиясын құру;
- Пайдаланушылармен бірге бірлескен аттестацияны жүргізу.

Технологиялық іс-шараларға келесі іс-шараларды жатқызады:

- әзірлеудің барлық сатыларында сапа стандарты мен дәл сақтауды таңдау;
- деректер қорын басқару жүйелерін таңдау;
- бірыңғай әзірлеу ортасын пайдалану, яғни әзірлеудің бағдарламалық өнімдері, олар бағдарламалық қамсыздандырудың бір немесе барлық өмірлік циклы сатыларын сүйемелдейді;
- жеңілдетілген модельдеу тілін пайдалану;
- тестілеуді автоматтандыру құралдарын қолдана отырып, бағдарламалық қамсыздандыруды тестілеу;
- заманауи сенімді бағдарламалау құралдарын, қателерді енгізу тәуекелін азайтатын құралдарды пайдалану;
- сынау нәтижелері мен қателерге талдау негізінде бағдарламалық қамсыздандыруды тұрақты жетілдіру.

Бағдарламалық қамсыздандыруды әзірлеу кезіндегі тәуекелді бағалау

Сенімділік - бұл белгілі бір пайдалану жағдайларында белгілі бір уақыт аралығы ішінде бағдарламалық қамсыздандырудың өз қасиеттерін (істен шықпау, беріктілік және басқалар) сақтау қабілеті. Сенімділік кепілінің факторларына жүйенің немесе ортаның залалына және қолайсыз салдарға әкелетін тәуекел жиынтығы ретінде тәуекелдер жиынтығы жатады. Тәуекел сенімділік қасиеттерін өзгертеді және азайтады.

Әзірлеу процесінде пайда болатын тәуекелдер ресурстардың шектелуіне, қаржылық және ұйымдастырушылық қамсыздандыруды шектеуге, бағдарламалық өнімнің үлкен көлемі мен күрделілігіне байланысты болуы мүмкін.

Ресурстық тәуекел ресурстарды пайдалануды жоспарлау

процесінде талаптарға талдау сатысында шығады. Оларға техникалық қамтамасыз ету өнімділігінің жеткіліксіздігін, бағдарламалық аспаптық құралдардың сәйкессіздігін, қажетті біліктілігі бар мамандардың болмауы және т.с.с. жатқызуға болады.

Қаржылық тәуекел, негізінен, ресурстарды дұрыс жоспарламаудан жобаның бюджеттен асып кетуіне байланысты болады. Осылайша, қаржылық тәуекел бағдарламалық қамсыздандырудың күрделілігі мен үлкен көлемімен негізделген тәуекелге және ресурс қаупіне байланысты. Қаржыландырудың жеткіліксіздігін әзірлеудің бастапқы сатысында белгіленбеген кейбір қосымша қажеттіліктер тудыруы мүмкін.

Ұйымдастырушылық тәуекелі әзірлеу процесін дұрыс орындамауға, орындаушылар арасында міндеттемелерді дұрыс жоспарламаумен және қате бөлумен, міндеттемелерін тиісті орындамауға байланысты.

Бағдарламалық өнімнің күрделілігіне және үлкен ауқымына байланысты тәуекел көлемі мен күрделілігін алдын-ала дұрыс анықтамауға немесе дәл анықтамауға негізделген. Жоспарлау процесінде алдын-ала бағалаудың түсірілуі қажетті ресурстарды дұрыс анықтамауға әкеледі және де оның салдарында жұмыс кестесі бұзылады және жобаларды орындау мерзімдерін үзеді.

Ықтимал тәуекелді анықтағаннан кейін сараптамалық бағалау олардың пайда болу ықтималдылығына жүргізіледі және оларды жеңу тәсілдері айқындалады.

7.4. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ САПАСЫН БАСҚАРУ

Бағдарламалық қамсыздандырудың сапасы, басқа өнімнің сапасы сияқты - бұл оның тапсырыс берушінің талаптарына сәйкестігі. БҚ сапасын арттыру үшін өнімді құру және сүйемелдеу процесін жақсарту керек.

Бағдарламалық қамсыздандыруды өнеркәсіптік әзірлеу - жоғары технологиялық процесс, мұнда түрлі міндеттері мен түрлі біліктілігі бар әзірлеушілердің ұжымы тартылады.

Көптеген әзірлеушілердің үйлескен жұмысы кезіндегі негізгі міндеттердің бірі - бірыңғай жұмыс схемасын қамтамасыз ету. Бұл жүйенің түрлі тораптарының тұтастығы мен қарама-қайшы болмауын қамтамасыз ететін жұмысты жоспарлауға мүмкіндік береді, сондай-ақ тапсырыс берушінің үмітіне БҚ сәйкес келетіндігіне кепілдік беруді

қамтамасыз етеді. Бұл міндетті орындау үшін БҚ әзірлеу процесін бір регламенттерге - әзірлеудің түрлі қырларын анықтайтын технологиялық талаптарға бағынады.

Әртүрлі типтегі бағдарламалық жүйелерді әзірлеген кезде технологиялық талаптар ажыратылуы мүмкін. Сонда да, әдетте технологияларды әзірлеуде енгізу бір негізгі мақсатты ұстанады - әзірленетін бағдарламалық өнімнің сапасын қамтамасыз ету және кепілдік беру. Жүйелерді әзірлеу технологияларына қойылатын талаптарды анықтайтын құжаттарды құруға - *сапа стандарттарына* алғышарт болады.

Сапа жүйесінің стандарттары тәсіліне сәйкес, *сапа* - бұл тұтынушының белгіленген және болжанған талаптарын қанағаттандыруға қабілеттілігіне қатысты нысанның сипаттамаларының жиынтығы. *Сапа объектісі* деп жеке бағдарламалық өнім немесе оны шығару процесі түсіндіріледі, сонымен өндіруші - ұйым немесе жеке қызметкер.

Бұл стандарттарда өнімнің өмірлік циклының сатыларының әрқайсысы үшін тән болатын сапасының белгілері айқындалады. Сапа стандарттарының негізгі мақсаты - технологиялық процестерге қойылатын талаптарды анықтау, оның ішінде әзірлеу процестері, құрылатын жүйелер сапасының деңгейінің таңдап алынған белгілерін анықтау көзқарасымен үнемі кепілдік беруге мүмкіндік береді.

Сапа стандарттарын қолдану саласы қаншалықты кең болса (жеке саладан жалпы қолданылуға дейін), оларды қарастырудың негізгі нысаны нақты әзірлеу әдістемелері емес, жалпы технологиялық процестер болып табылады.

Стандарттардың көбі жүйенің нақты өмірлік циклына тікелей тәуелді емес процестерге бағдарланған. Әрбір процесс үшін осы мақсаттарға жету құралдары сипатталып, мақсаттары анықталады. Өмірлік циклдың деректері үшін стандарттарда мақсат қанағаттандырылғандығын көрсететін сипаттама беріледі.

Стандарттарда қарастырылатын негізгі процесс - бағдарламалық қамсыздандыруды әзірлеу процесі, ол бағдарламалық қамсыздандырудың өмірлік циклының моделіне байланысты таңдалған жоқ. Әзірлеу процесі әдетте, ұсақ қосалқы процестерге ажыратылады. Қосалқы процестер келесі үш санаттарға жатқызылуы мүмкін:

1) БҚ жоспарлау процесі;
2) БҚ әзірлеу процестері, олар әдетте БҚ қойылатын талаптарды әзірлеуді, БҚ жобалау мен кодтауды қамтиды;
3) Тұтастылықты қамтамасыз ету процестері, олар БҚ анықтауды, БҚ сапасына кепілдік беруді, БҚ конфигурациялар басқаруды және

сертификаттау кезінде өзара байланысты қамтиды.

ISO 9000 стандарттар тобы - өнімдерді шығару кезінде сапа менеджменті ережелерін белгілейтін халықаралық стандарттардың тобы. Халықаралық ISO 9000 сәйкес келетін стандарттардың отандық тобы МемСТ Р ИСО 9000 деген атау алды. «Өнім шығару» ұғымы деп бағдарламалық қамсыздандыруды әзірлеу де түсіндіріледі.

БАҚЫЛАУ СҰРАҚТАРЫ МЕН ТАПСЫРМАЛАРЫ

1. Бағдарламалық қамсыздандырудың сапасы деген не?
2. Бағдарламалық қамсыздандыру сапасының белгілерін атаңыз.
3. Сапа факторы - сенімділік атрибуттарының жиынтығы қандай?
4. Сапа факторы - функционалдылық жарамдылықтың атрибуттар жиынтығы қандай?
5. Сапа факторы - сүйемелденудің атрибуттар жиынтығы қандай?
6. Сапа факторы - жинақылықтың атрибуттар жиынтығы қандай?
7. Бағдарламалық қамсыздандырудағы тоқтап қалу істен шығудан қалай ажыратылады?
8. Бағдарламалық жүйенің шекті күйі деп нені атайды?
9. Бағдарламалық қамсыздандыруда қандай тоқтап қалу түрлері бар?
10. Бағдарламалық қамсыздандырудың тоқтап қалуы неге байланысты туындайды?
11. Пайдалануда тоқтап қалу өндірістік тоқтап қалудан қалай ажыратылады?
12. Ресурстық істен шығу деген не?
13. Істен шығу уақыт сипаттамалары жағынан қалай жіктеледі?
14. Жасырын істен шығу деген не?
15. Бағдарламалық қамсыздандыру сапасының қажетті деңгейін қамтамасыз ететін іс-шараларды әкімшілік және технологиялық деп шартты түрде ажыратуға болады ма?
16. БҚ сапасының қажетті деңгейін қамтамасыз ететін қандай іс-шаралар әкімшілік іс-шараларға жатады?
17. БҚ сапасының қажетті деңгейін қамтамасыз ететін қандай іс-шаралар технологиялық іс-шараларға жатады?
18. БҚ әзірлеген кезде қандай тәуекел болады?
19. БҚ сапасын басқару не үшін қажет?

8-бөлім

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ТЕСТІЛЕУ

8.1. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ТЕКСЕРУ ПРОЦЕСІ РЕТІНДЕ ТЕСТІЛЕУ

Тексеру БҚ берілген функциялары мен тапсырыс берушінің талаптарына, сондай-ақ берілген сипаттамаларға сәйкес орындалуының дұрыстығын тексеруді қамтамасыз етеді.

Тексеру стандарттарда ӨЦ дербес процесі ретінде берілген және талаптарға талдау сатысынан бастап бағдарламалық кодтың қорытынды сатыда жұмыс істеуінің дұрыстығын тексерумен анықталуда, атап айтқанда тестілеуде, қолданылады.

Бағдарламалық қамсыздандыруды тестілеу - бағдарламалық қамсыздандыруды (бағдарламалық код пен құжаттаманы) тексеру және зерделеу процесі, ол екі түрлі мақсатты көздейді:

1) тапсырыс берушілерге, сондай-ақ әзірлеушілерге бағдарламалық өнімнің талаптарға сәйкестігін көрсету;
2) бағдарламалық қамсыздандыру әрекеті дұрыс емес, қажетсіз немесе сипаттамасына сәйкес келмейтін жағдайларын анықтау.

Тестілеу - бұл бағдарламалық қамсыздандырудың арнайы, жасанды құрылған жағдайларда жұмысын қадағалау арқылы жүзеге асырылатын талаптарға сәйкестігін тексеру.

Тестілеудің негізгі мақсаты - бағдарламалық қамсыздандырудың жалпы дұрыстығына жеткілікті дәрежеде тестілеуге және көрсетуге жеткілікті болатын оқиғалардың жиынтығын құру, және нақты осы жағдайда бағдарламалық қамсыздандыру дұрыс жұмыс істеп, талаптарға сәйкес болатындығына көз жеткізу. Тестілеуді арнайы тәуелсіз тестілеушілер алдын-ала жоспарлап, жүйелі түрде өткізіп тұруы тиіс.

Тестілеуші - тестілеумен айналысатын маман. Тестілеуші бағдарламалық қамсыздандыруды жұмысында болатын ықтимал қателер мен істен шығуларды іздеуді жүзеге асырады, бағдарламалық

өнімді пайдалану процесінде пайда болуы мүмкін түрлі жағдайларды модельдейді.

Тестілеу бағдарламалық қамсыздандыру сапасын бақылауда кеңінен қолданылатын әдіс болып табылады. Сапаның көптеген атрибуттарын бағалау үшін тестілеуден басқа, тиімді тәсілдер жоқ. Тестілік жинақ үшін тестілерді іріктеу әдістері мен құру жүйесі *тестілеу стратегиясы* деп аталады.

Тестілеушінің жұмысы талаптардың сипаттамасын бекіткенге дейін басталады, себебі бағдарламалық қамсыздандыруға тестілеудің толық болуы мен тестілену мүмкіндіктеріне қойылатын талаптарды анықтайды, тестілеу әдістерін айқындайды. Талаптардың сипаттамаларын жоспарлау және құру сатысының басталуымен бір уақытта тестілеуші тестілеу стратегиясын әзірлейді. Талаптардың сипаттамасын бекіткеннен кейін тестілеуші тестілеудің егжей-тегжейлі жоспарын әзірлейді, бағдарламалық қамсыздандырудың дұрыстығын тексеруге арналған тестілердің жинағын құрады. Тестілеу оның нәтижелері туралы есептерді құрумен аяқталады.

Тестілеу ретке келтіру, бақылау және сынау болып табылады.

Ретке келтіру - бағдарламалық қамсыздандыруды әзірлеу сатысында бағдарламалық кодты тестілеу.

Бақылау - тестіленетін немесе модельденетін ортада бағдарламаларды орындау кезіндегі қателерді іздеу.

Сынау - бағдарламаны нақты ортада орындау кезінде қателерді табу талпынысы.

Тестілер белгілі талаптарды қанағаттандыруы тиіс:

- ең алдымен тестінің ықтимал қателерді анықтау ықтималдылығы жоғары болуы мүмкін. Тестілік сценарийлерді әзірлей отырып, бағдарламаның барлық мүмкін істен шығу нұсқаларын талдау қажет немесе оның дұрыс жұмыс істемеуіне талдау жасау қажет;
- тестілер жиынтығы артық болмауы тиіс. Бір қатені анықтау үшін бірнеше тестілерді орындауда қажеттілік жоқ. Олардың бірін орындау жеткілікті, бірақ бұл тесті өз санатындағы ең үздігі болып табылса. Ұқсас тестілер тобында бірі едәуір тиімді, кейбіреулері аз тиімді, сол себепті қатені анықтау ықтималдылығы бар тестіні таңдау керек;
- тест тым қарапайым немесе тыс күрделі болмауы керек. Үлкен және күрделі тестті түсіну қиын, орындау қиын және ұзақ құру керек. Сондықтан ортасын ұстанған дұрыс.

Бағдарламалық қамсыздандырудағы барлық қателерді анықтауға қабілетті тестілер бар болуы екіталай. Аталған талаптарға жауап беретін жақсы тест қателердің көбін анықтауға мүмкіндік береді.

Тестілеу процесі тексеру процесінің құрамдас бөлігі болып табылады деп санау қабылданған. Тексерудің мақсаты тексерілетін нысанның талаптарға сәйкестігі кепілдігіне қол жеткізу болып табылады, яғни көзделмеген функцияларды қараусыз іске асырған және жобалық сипаттамалар мен стандарттарға қанағаттандырылуы. Тексеру процесі кодты тексеруді, тестілеуді, тестілеу нәтижелерін талдауды, проблемалар туралы есептерді қалыптастыру мен талдауды қамтиды.

Егер бұл процестерге жауап берілетін мәселеге қарау көзқарасынан алсақ, онда тестілеу «Бұл қалай жасалды?» немесе «Әзірленген бағдарлама талаптарға сәйкес келеді ме?» сауалдарына жауап береді, анықталатыны - «Не жасалды» немесе «Әзірленген жүйе талаптарға сәйкес пе?».

Тексеру процесі жүйеде істен шығуды, тоқтап қалуды немесе аппаттардың шығуын тудыруы мүмкін барлық ақаулардың болмауына кепілдік бере алмайды, осы ақаулардың белгілі бір деңгейде болмауы туралы сөз болып отыр.

8.2. ТЕСТІЛЕУ ӘДІСТЕРІ

Тестілеудің жүйелі әдістері бағдарламада «қара жәшік» ретінде қарастырылатын әдістерге және бағдарлама «ақ жәшік» ретінде қарастырылатын әдістерге бөлінеді.

«Қара жәшікті» тестілеу

«Қара жәшікті» тестілеу - бұл сыртқы әлем көзқарасынан нысанның (бағдарламалық жүйенің) функционалдық әрекетін тестілеу әдісі.

«Қара жәшік» әдісін тестілеу кезінде ішкі құрылымы белгісіз бағдарлама, нысан ретінде қарастырылады. Тестілеуші деректерді енгізеді де, нәтижені талдайды, бірақ ол бағдарламаның қалай жұмыс істейтіндігін білмейді. Тестілерді іріктей отырып, маман өз көзқарасы жағынан стандартты емес нәтижелерге әкелуі мүмкін қызықты кіріс деректері мен шарттарын іздейді. Тестіленетін бағдарламаның қателері анықталуы мүмкін үлкен ықтималдылықпен шығатын кіріс деректерінің әрбір кластарының өкілдері қолданылады.

Берілген тестілеу әдісі үшін тестілеушінің негізгі міндеті жүйенің талаптарға сәйкестігін тексеру бірізділігінде болады. Бұдан басқа, тестілеуші жүйенің апатты жағдайлардағы жұмысын тексеруі қажет,

мысалы, кіріс мәндері қате берілген кезде. Өте тамаша жағдайда апатты жағдайлардың барлық нұсқалары жүйеге арналған талаптарда сипатталуы керек және тестілеуші осы талаптардың нақты тексерілуін ойлап табуы керек. Алайда, шын мәнінде тестілеу нәтижесінде әдетте жүйенің екі түрлі проблемасы анықталады:

1) жүйенің талаптарға әрекет етуінің сәйкес болмауы;
2) жүйенің талаптармен көзделмеген жағдайларда лайықсыз әрекет етуі.

Бұл проблемалар түрлері туралы есептер құжатталады және әзірлеушілерге беріледі. Бұл ретте бірінші түрдегі проблемалар бағдарлама кодын өзгертеді, талаптардың өзгеруі тым сирек болады. Бұл жағдайда талаптарды өзгерту қарама-қайшылық (бірнеше түрлі талаптар осы жағдайда жүйенің әрекет етуінің түрлі модельдерін сипаттайды) немесе дұрыс болмауы (талаптар шынайылыққа сәйкес келмейді) салдарынан керек болуы мүмкін.

Екінші проблемалардың түрлері олардың толық болмауына байланысты талаптардың өзгеруін талап етеді - талаптарда жүйенің лайықсыз әрекет етуіне әкелетін жағдайлар анық кетіп қалған. Бұл әрекетті лайықсыз әрекеті деп жүйенің толығымен құруы сияқты, талаптарда сипатталмаған кез келген әрекеті түсіндірілуі мүмкін.

«Қара жәшік» принципі бойынша тестілеу әдістері қамтамасыз етеді:

- эквивалентті бөлу;
- шектік мәндерін талдау;
- реверсивтік талдаумен қосылғанда тестіленетін бағдарламаның жұмыс істеуі туралы жеткілікті түрде толық ақпарат беретін функционалдық диаграммаларды қолдану.

Эквивалентті бөлу бағдарлама деректерінің кіру ортасында эквиваленттіліктің класс сандарына бөлуде болады, мұнда әрбір тест кейбір кластың өкілі бола отырып, басқа кластың кез келген тестіне эквивалент бола алады.

Эвиваленттілік кластары кіріс шарттары мен бөлуді іріктеу арқылы екі немесе одан көп топтарға бөлінеді. Бұл ретте эквиваленттіліктің екі класс типтерін ажыратады: бағдарлама үшін кіріс деректерін беретін дұрыс және қате кіріс мәндерін беруге негізделген дұрыс емес. Эквивалентті бөлу әдісімен тестілерді әзірлеу екі сатыда жүзеге асырылады: эквиваленттілік кластарын бөлу және тестілерді құру. Кіріс деректерін таңдауға негізделген тестілерде құру кезінде бағдарламалық символикалық түрде орындалуы жүргізіледі.

Осылайша, «қара жәшік» принципі бойынша тестілеу әдістері бағдарламада әзірленген функцияларды тестілеу үшін қолданылады.

Бұл үшін функцияның нақты әрекеті мен талаптардың сипаттамасын есепке ала отырып, күтілетін әрекеттерінің арасындағы сәйкессіздік тексеріледі. Бұл тестілеуге дайындалу барысында шарттардың кестелері, себеп-салдар бағандары мен бөлу аясы құрылады. Сонымен қатар, функцияның әрекет етуіне ықпал ететін ортаның параметрлері мен жағдайларын есепке алатын тестілік жинақтар даярланады.

«Ақ жәшікті» тестілеу

«Ақ жәшік» әдісі бағдарламаның ішкі құрылымын зерделеуге мүмкіндік береді. ***«Ақ жәшік» принципі бойынша тестілеу*** бағдарламаның жолдық және имитациялық тестілеу арқылы бағдарламалардың барлық жолдарын тексеруге бағдарланған.

«Ақ жәшікті» тестілеу тестілік жағдайларды таңдау, деректерді дайындауды таңдау арқылы бағдарлама модельдерінің және графтық моделінің деңгейлерінде қолданылады және тестілеу келесі элементтерді қамтиды:

- кем дегенде бір рет орындалуы тиіс, есепке алу қатесіз операторлар бағдарламада логикалық жолдардың санына байланысты қала беруі мүмкін және осы мақсаттарға өтудің қажеттілігі;
- жол предикаттарының көмегімен басқаруды беріліс бағыттарын анықтауға арналған басқару ағынының берілген жолы бойынша, есептеу үшін осы жолдардың өтуіне кепілдік беретін тестілік деректердің жиынтығы құрылады. Алайда, барлық жолдарды тестілеу мүмкін емес, сондықтан пайдалану процесінде пайда болуы мүмкін анықталмаған қателер қалады;
- бағдарламаларды жеке бөлік-блоктарға бөлетін блоктарға, олар бағдарламадағы жолдардан бір немесе бірнеше рет өткенде орындалады, олар аталған жолдың орындалуы үшін қолданылатын көптеген кіріс деректерінің орналасқан жері, бір функцияның іске асу блоктарының жиынтығын қамтиды.

«Ақ жәшікті» тестілеу - бұл әзірлеу сатысында қолданылатын тестілеу технологиясы (кейде оны «шыны жәшікті» тестілеу деп те атайды).

Тестілеуші толық қатынау мүмкіндігі бар бастапқы кодты білуге негізделген тестілерді әзірлейді. Нәтижесінде ол келесі артықшылықтарды алады:

- *тестілеудің бағытталуы*. Бағдарламалаушы бағдарламаны бөліктері бойынша тестілей алады, тестіленетін модульді шақыратын арнайы мәтіндік сценарийлерді әзірлей алады және оған

қажетті деректерді береді;

- *кодты толық қамту*. Бағдарламалаушы әрбір мәтінде кодтың қандай фрагменттері жұмыс істейтіндігін анықтай алады;
- *командалар ағынын басқару мүмкіндігі.* Бағдарламашы оның орындалу барысы туралы ақпаратты көрсететін ретке келтіру командаларын қоса алады немесе арнайы бағдарламалық құрал - ретке келтірушімен қолдана алады;
- *деректердің тұтастығын қадағалау мүмкіндігі.* Деректердің күйін қадағалау (ретке келтірушінің көмегімен) бағдарламашы деректердің модельдерге өзгеруі, олардың қате интерпретациясы немесе сәтсіз ұйымдастырылуы сияқты қателерді анықтай алады;
- *ішкі шектеу нүктелерін көру.* Мысалы, белгілі бір әрекеттерді орындау үшін бірнеше түрлі алгоритмдер қолданылуы мүмкін, алайда бағдарламалық кодқа қатынау мүмкіндігі болмай, бағдарламаларды әзірлеу барысында қайсысы таңдалған анықтау мүмкін емес.

«Ақ жәшікті» тестілеу - бағдарламалау процесінің бөлігі.

Бағдарламалаушылар бұл жұмысты үнемі орындап отырады, олар әрбір модульді ол жазылғаннан кейін тестілейді, ал содан кейін жүйеге оның интеграциясынан кейін тестілейді.

«Ақ жәшікті» тестілеудің теориялық негіздемесі мықты болғанымен, көптеген тестілеушілер «қара жәшікті» тестілеуді мақұлдайды. «Ақ жәшікті» тестілеу математикалық модельдеуге жақсы келеді, бірақ мұның барлығы оның тиімді екендігін білдірмейді. Технологиялардың әрқайсысы басқасы қолданылған жағдайда өткізілетін қателерді анықтауға мүмкіндік береді. Бұл жағынан алғанда олардың тиімділігі бірдей деп айтуға болады.

8.3. ТЕСТІЛЕУДІ ДЕҢГЕЙЛЕР БОЙЫНША ЖІКТЕУ

Бағдарламалық өнімді толық тексеруге мүмкіндік беретін және қателерді барынша көп табуға мүмкіндік беретін тестілеудің бірнеше деңгейі бар: модульді, интеграциялық, жүйелік, шығу, қабылдау. Әрбір деңгейдің өз мақсаттары мен компоненттері бар.

Модульдік тестілеу - бұл бағдарламаның жеке алынған модульдері, функциялары немесе кластары деңгейінде бағдарламаны тестілеу.

Модульді тестілеудің мақсаты алгоритмдерді іске асыруда қателерді табудан, сондай-ақ әзірлеу мен тестілеудің келесі деңгейіне өтуге жүйенің дайындық дәрежесін анықтауда болады.

Модульді тестілеу «ақ жәшік» принципі бойынша жүргізіледі, яғни бағдарламаның ішкі құрылымын білуге негізделеді, және кодты жабудың талдау әдістерін жиі қамтиды. Модульді тестілеуді бағдарламалық қамсыздандырудың әзірлеушісі тікелей жүргізеді және әрбір модельдегі деректердің ішкі құрылымы мен ағындарын тексеруге мүмкіндік береді. Бұл тестілеу түрі әзірлеу сатысының бөлігі болып табылады.

Модульді тестілеу әдетте тестіленетін модульдің барлық интерфейстері үшін бітеуіштерді қамтитын әрбір модулінің шеңберінде белгілі бір ортаны құруды білдіреді. Олардың ішінен кейбіреулері кіріс мәндерін беру үшін, ал басқалары нәтижелерді талдау үшін қолданылуы мүмкін және т.б.

Модульді тестілеу деңгейінде алгоритмдік қателермен байланысқан қателерді және алгоритмдерді кодтау қателерімен байланысты ақауларды, циклдердің шарттары мен санауыштарына байланысты жұмыс түрлерімен байланысты ақауларды анықтайды, сондай-ақ жергілікті айнымалыларды және ресурстарды қолданумен. Деректердің дұрыс берілмеуіне, интерфейстерді дұрыс іске асырмауына, үйлесімділіген, өнімділігіне байланысты қателер және т.с.с, әдетте, модульді тестілеу деңгейінде өткізіледі және тестілеудің ең кеш сатыларында анықталады.

Модульді тестілеу кезінде әзірлеушілер әрбір модульдің 75% артық тестілеуді қамтитындай тестілер жиынтығы орындалады.

Модульді тестілеу қамтиды:

- бағдарламалық кодта синтаксистік қателерді анықтау үшін (синтаксистік тексеру) кейбір аспаптық құралдарды қолдана отырып,

бағдарламалық кодты тексеру;

- кодтың кодтау стандарттарына сәйкестігін тексеру (мысалы, бағдарламалық кодтың әзірлеуші-ұйым талаптарына сәйкестігін ресімдеу);
- бағдарламалық кодқа техникалық шолу.

Модульді тестілеуді орындау кезінде не құрылымдық, не функционалдық тестілеудің технологияларын пайдалануға болады.

Құрылымдық тестілеу «ақ жәшікті» тестілеу түрлерінің бірі болып табылады. Оның басты идеясы тестіленетін бағдарламалық жолды дұрыс таңдау болып табылады.

Оған қарама-қарсы *функционалдық тестілеу* «қара жәшік» тестілеу санаттарына жатады. Бағдарламаның әрбір функциясы оның кіріс деректерін енгізу және шығыс деректерін талдау арқылы тестіленеді. Бұл ретте бағдарламаның ішкі құрылымы өте сирек ескеріледі.

Модульді тестілеу сәтті аяқталғаннан кейін барлық өзгертілген модульдер мен тестілер жиынтығы жобаның деректер қорында сақталады.

Интеграциялық тестілеу жеке модульдердің бірлеске жұмыстарын тексеру үшін жүргізіледі және біртұтас ретінде бүкіл жүйені тестілеудің алдында болады. Интеграциялық тестілеу - бұл екі және одан артық модульдерден тұратын жүйенің бөлігін тестілеу. Интеграциялық тестілеудің негізгі міндеті - модульдер арасындағы интерфейстік өзара байланысты іске асыруда және интерпретациядағы қателерді іздеу.

Интеграциялық тестілеу элементтері болып табылады:

- функционалдылықты тексеру, яғни талаптардың сипаттамасында берілген модульдер, функциялармен орындалатын жеке функциялардың сәйкестілігін тексеру;
- аралық нәтижелердің бар екендігін және дұрыстығын тексеру;
- модульдер арасындағы ақпараттың дұрыс берілуін тексеру (интеграцияны тексеру).

Технологиялық жағынан алғанда, интеграциялық тестілеу модульді тестілеудің сандық дамуы болып табылады, себебі модульді тестілеу де модульдер мен қосалқы жүйелердің арасындағы интерфейстерді, тестілік аяның құрылуын талап етеді, жоқ модульдердің орнында тұйықтағышты қоса алғанда. Модульді және интеграциялық тестілеу арасындағы негізгі айырмашылық мақсаттарында болады, яғни анықталатын ақаулардың типтерінде. Бұл кіріс деректері мен талдау әдістерін таңдау стратегиясын анықтайды. Интеграциялық тестілеу интерациялық түрде жүргізіледі, модульдер мен қосалқы жүйелер біртіндеп қосылады. Оны тәуелсіз тестілеуші жүзеге асырады.

Интерациялық тестілеу барысында анықталған қателер жобаның деректер қорына енгізіледі. Интеграциялық тестілеу нәтижелері тестілеу циклы аяқталғанда тестілеу барысы туралы есепке қосылады.

Жүйелік тестілеуді интеграциялық тестілеу сәтті аяқталған жағдайда тәуелсіз тестілеуші жүргізеді. Жүйелік тестілеу сапасы жағынан интеграциялық және модульдік деңгейлерден ерекшеленеді, тестіленетін жүйені тұтастай қарастырады және пайдаланушылар интерфейсі деңгейінде пайдаланады, интеграциялық тестілеу фазаларының соңғысына қарағанда модульдер интерфейсі деңгейінде пайдаланылғанда.

Тиісінше, осы тестілеу деңгейлерінің мақсаттары да ажыратылады. Бағдарлама ішіндегі тест жолдарының өтуін талдау қиын немесе жүйе деңгейіндегі нақты функциялардың жұмысының дұрыстығын қадағалау қиын.

Жүйелік тестілеудің негізгі міндеті - жалпы жүйенің жұмысымен байланысты проблемаларды анықтау (мысалы, жүйенің ресурстарын дұрыс қолданбау, ортамен үйлеспеушілігі, пайдаланудың көзделмеген сценарийлері, жоқ немесе дұрыс емес функционалдылық, қолданудағы ыңғайсыздық және т.с.с.).

Жүйелі тестілеу «қара жәшік» әдісінің көмегімен жалпы жобамен жұмыс істегенде жүргізіледі, яғни бағдарлама құрылымының пайдаланушыға көрінетін кірістері мен шығыстары қолжетімді болатын ешбір мәні болмайды. Тестілеуге кодтар пайдаланушы құжаттамасы жатады.

Жүйелік тестілеу тестілерінің санаттары:

- функционалдық міндеттерді шешудің толықтығы;
- ресурстарды пайдаланудың дұрыстығы (жадының жылыстауы, ресурстарды қайтару және т.с.с.);
- өнімділікті бағалау;
- дұрыс емес әрекеттер мен деректерді қисық беруден қорғау тиімділігі;
- әр түрлі платформаларда инсталляция мен конфигурацияны тексеру;
- құжаттаманың дұрыстығы.

Тестілеудің осы деңгейінде қолданылатын деректердің көлемі ұтымды тәсіл тестілеудің толық немесе жеке автоматтандыру болып табылатындай, ол модульдерді тестілеу деңгейінде немесе олардың үйлесімін қолданатын тестілеу жүйесіне қарай біршама күрделі тестілеу жүйесін құру қажеттілігіне әкеледі.

Жүйелі тестілеу кезінде анықталған қателер жобаның деректер қорына енгізіледі. Жүйелі тестілеу нәтижелері тестілеу барысы туралы есепке қосылады.

Шығу тестілеуі бағдарламалық қамсыздандыруды тапсырыс берушіге/пайдаланушыға жеткізу үшін даярлығын тексеру мақсатымен жүзеге асырылады. Бұл тестілеудің аяқтаушы сатысы, тәуелсіз тестілеуші жүргізетін, баптау бойынша нұсқамалардың дұрыстығына тексеруді қамтиды, сондай-ақ, құжаттаманың жиынтығын тексеруді қамтиды.

Шығу тестілеуі кезінде анықталған қателер жобаның деректер қорына енгізіледі. Бағдарламалық қамсыздандырудың шығу тестілеуі сәтті аяқталғанда тапсырыс берушіге тестілеу нәтижелері туралы есеппен бірге бағдарламалық өнім жеткізіледі.

Қабылдау тестілеуді бағдарламалық жүйені қосуға, сүйемелдеуге және соңғы пайдаланушыларды оқытуға жауапты ұйым жүргізеді. Бұл тестілеудің соңғы деңгейі бұдан кейін өнім пайдалануға енгізіледі.

Алғашқы төрт деңгейлі тестілеу әзірлеуші-ұйымның ішінде, ал қабылдау тестілеу тапсырыс берушінің өкілімен бірлесе орындалады. Бірінші деңгейлі тестілеуді бағдарламалық қамсыздандыру әзірлеушісі әзірлеу сатысында жүргізеді, ал басқа деңгейлерді тестілеуді тәуелсіз тестілеушілер жүзеге асырады.

Бір-бірінен орындалу уақыты бойынша ерекшеленетін екі тестілеу түрі бар - альфа-тестілеу және бета-тестілеу.

Альфа-тестілеу - бұл әлеуетті пайдаланушылардың немесе тапсырыс берушілердің бағдарламалық қамсыздандырумен нақты жұмысы немесе әзірлеушілердің нақты жұмысының имитациясы.

Көбінесе альфа-тестілеу өнімді әзірлеудің ең алғашқы сатысында, бірақ функционалдылығы толық немесе шамамен тұтастай іске асырылғанда жүргізіледі.

Кейбір жағдайларда альфа-тестілеу ішкі қабылдау тестілеуі ретінде аяқталған өнім үшін қолданылуы мүмкін.

Альфа-тестілеу қателерді анықтауға көмектесетін қандай да бір аспаптардың немесе ретке келтірушіні қолданумен орындалуы мүмкін. Анықталған қателер қосымша зерттеусіз тестілеушілерге берілуі мүмкін.

Бета-тестілеу - дайын дерлік бағдарламалық қамсыздандыруды жоспарланған функцияларының толық жиынтығымен қарқынды пайдалану.

Бета-нұсқа өнімнің қорытынды нұсқасы болып табылмайды, сол себепті әзірлеуші компьютердің жұмыс бұзуы мүмкін және/немесе деректердің жоғалуына әкелетін қателердің толық жоқтығына кепілдік бермейді.

Оның мақсаты бағдарламалық қамсыздандырудың жұмысындағы қателердің барынша санын анықтау, кейіннен жою үшін бағдарламалық

қамсыздандырудың жұмысында оны нарыққа, көпшілік пайдалануға қорытынды шығарудың алдында анықтау болып табылады.

Әзірлеушілердің немесе тестілеушілердің күшімен өткізілетін альфа-тестілеуге қарағанда беста тестілеу өнімнің алдында аталған нұсқасының (бета-нұсқасы) қатынау мүмкін тараптық пайдаланушыларды тартуды болжайды.

Сонымен қатар, бета-тестілеу жарнама мақсаттарында өнімді нарыққа шығару стратегиясының бөлігі ретінде (мысалы, бета-нұсқаларды тегін тарату пайдаланушылардың өнім нұсқасын алуына кең тану үшін мүмкіндік береді), сонымен қатар болашақ пайдаланушылардың кең ауқымынан ол туралы алдын-ала пікір алу үшін қолданылуы мүмкін.

8.4. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ ӨНІМДІЛІГІН ТЕСТІЛЕУ

БҚ өнімділігін тестілеу бағдарламалық жүйе немесе оның бір бөлігі белгілі бір жүктememен қаншалықты жылдам жұмыс істейтіндігін анықтауға болатындай жүргізіледі. БҚ өнімділігін тестілеу ресурстардың масштабталуы, сенімділігі, тұтынылуы сияқты жүйенің өзге сапа атрибуттарын тексеру және растау үшін қызмет ете алады.

Бағдарламалық қамсыздандырудың өнімділігін тестілеудің екі тәсілі ықтимал:

1) бағдарламалық қамсыздандыру түрлі пайдалану сценарийіне сәйкес жағдайларда тестілеуге түседі;
2) жүйе нақты соңғы пайдаланушылармен сыналатын бета-тестілеу шеңберінде.

Өнімділікті тестілеуде келесі бағыттарды ажыратады:

- жүктеме тестілеу;
- күйзеліс тестілеу;
- тұрақтылықты тестілеу;
- конфигурациялық тестілеу.

Жүктеме тестілеу

Жүктеме тестілеу бағдарламалық қамсыздандырудың берілген жүйеге қойылатын талаптарға сәйкестігін белгілеу мен өнімділігін анықтау мақсатымен сыртқы сұранысқа жауап уақытының көрсеткіштерін жинауды білдіреді.

Жүйенің барынша жоғары жүктемелерінде жүйенің қосылу уақытын зерттеу үшін, олар оны пайдаланудың қалыпты сценарийлерінен асқан болса, күйзеліс-тестілеу жүргізіледі. Жүктеу және күйзеліс-тестілеу жүргізіледі. Жүктеу және күйзеліс-тестілеу арасында анық шектері жоқ, алайда бұл тестілеу түрлері түрлі сауалдарға жауап береді және түрлі әдістемені қолданады.

Жүктеу тестілеуі бағдарламалық өнімнің мүмкіндіктерін шектеуді тексеру болып табылады, олар құжаттамада айқындалады. Файлдардың барынша көп санын ашуға болады немесе бағдарламалық қамсыздандыру жұмыс істейтін, қандай да бір параметрлердің рауалы мәндері логикалық тексеретін өзге деректерінің құрылымын ашуға болады. Сондай-ақ, бағдарлама түрлі аппараттық ресурстар алынып тасталғанда қалай болатындығын тексеру қажет.

Жүктеу тестілеуі - бұл шектік шарттарды тестілеу түрлерінің бірі. Оның әрекет сұлбасы тым ұқсас. Алдымен бағдарламаны ол жұмыс істейтін жағдайларда қосады, ал содан кейін оның арналған жағдайларында жұмыс істеуге арналмаған. Шарттардың үйлесу тексерісі мен үйлесімдеріне мәні бар. Жеке шамадан тыс түрлі жүктемелерді жұмыс істей отырып, бағдарлама олардың барлығына шыдай алмайды.

Жүктеу тестілеуі, ең алдымен, бірнеше пайдаланушының жұмысын бір уақытта эмуляцияны болжайды, көп пайдаланушы жүйелер үшін, негізінен, ол клиент-серверлік сәулетті (мысалы, веб-серверлерді) қолданады. Алайда бағдарламалық қамсыздандыру жүйелерінің типтері де осы тәсілмен тестіленуі мүмкін. Мысалы, мәтіндік немесе графикалық редакторға өте үлкен құжатты жүктеуге болады; бухгалтерлік бағдарламалық жүйеде бірнеше жыл ішіндегі деректердің негізінде есепті қосады. Ең лайықты жобаланған жүктеу тесті біршама дәл нәтижелерді береді.

Жүктеу тестілеудің негізгі мақсаты жүйеде күтілетін белгілі бір жүктеуді құра отырып, ұқсас бағдарламалық және аппараттық қамсыздандыруды қолдана отырып, бағдарламалық жүйенің өнімділігінің көрсеткіштерін қадағалауда болады.

Керемет жағдайда жүктеп тестілеудің сәтті болу критерийлері ретінде жүйенің өнімділігіне қойылатын талаптар шығады, олар негізгі сәулеттік шешімдерді бағдарламалауды бастағанға дейін жүйеге қойылатын функционалдық талаптарды әзірлеу сатысында құжатталады. Алайда мұндай талаптар анық қалыптасқан немесе мүде қалыптаспаған болуы жиі. Бұл жағдайда алғашқы жүктеп тестілеу сынама болып табылады және күтілетін жүктеме туралы саналы ұсыныстарға және ресурстардың аппараттық бөліктерін тұтынуға негізделеді.

Жүйенің түрлі тораптарында проблемаларды анықтау мен тексеру үшін түрлі құралдар болады.

Жүктеп тестілеудің нәтижесі талдау үшін бұдан әрі қолданылатын қосымшалардың өнімділік көрсеткіштері болып табылады.

1. ***Орталық процесс ресурстарын тұтыну,*** (%) - берілген белгіленген аралықтан қанша уақыт таңдап алынған процесті есептеуге процессордың жұмсағандығын көрсететін метрика. Заманауи жүйелерде маңызды фактор бірнеше ағындарда жұмыс істеу қабілеті болып табылады, процессор есептеуді параллель жүргізу керек. Ресурстарды тұтынуды талдау жүйенің жалпы өнімділігіне түрлі факторлардың ықпалын түсіндіре алады.

2. *Жедел жадыны тұтыну,* (Мб) - қосымша қолданған жадының санын көрсететін метрика. Пайдаланылған жады бірнеше санаттарға бөлінеді:

- процесс қолданатын адрестік виртуалды кеңістіктің көлемі. Бұл көлемі дискілік кеңістіктің пайдаланылғандығы сияқты, жедел жадыны пайдаланылуын білдіреді. Виртуалды жады жүйесі бір процестің ағындары басқа процеске тиесілі жадыға қатынаса алмайтындығына кепілдік береді;
- процесс алған адрестік кеңістік көлемі және басқа процестермен бөлінбеген болса;
- жуырда пайдаланылған процестердің жады беттерінің жиынтығы. Бос жады жеткілікті болғанда, парақтар жиында қала береді, тіптен олар қолданылмаса да. Бос жады аз қалған кезде пайдаланылған парақтар ОЗУ қатты дискіге ауысады, ол жадының актив басқа парақтарын жүктеу үшін ОЗУ босатады;
- процессор қолданатын физикалық жады көлемі, ол басқа процестермен бірге пайдаланылуы мүмкін.

3. ***Желілік ресурстарды тұтыну*** - жалпы жүйенің өнімділігі шектерін көрсетеді.

4. ***Дискілі қосалқы жүйемен жұмыс*** (енгізу-шығаруды күту уақыты). Оқу немесе жазудың көп көлемі процессордың деректерді дискіден өңдеуін күтуде тұрып қалуына және шақырту уақытын арттыруға әкеледі.

5. ***Сұратуды орындау уақыты,*** (мс) - бағдарламалық қамсыздандырудың өнімділігінің басты көрсеткіштерінің бірі. Бұл уақыт сервер жағында өлшенгені сияқты, сұратуды өңдеу үшін серверлік бөлігінде талап етілетін; сол сияқты сұратуды қайта жіберу мен өңдеуге жұмсалатын толық уақыт көрсеткіші ретінде клиент жағында өлшенуі мүмкін.

Күйзеліс-тестілеу

Күйзеліс-тестілеу - бағдарламалық қамсыздандыруды тестілеу түрлерінің бірі, ол қалыпты жұмыс істеу шектерінің арту жағдайларында жүйенің сенімділігі мен беріктілігін бағалайды.

Күйзеліс тестілеу бағдарламалық қамсыздандырудың «нағыз маңыздысы» үшін әсіресе қажет болады. Әдетте, күйзеліс тестілеу жоғары жүктемедегі жүйенің беріктілігін, қолжетімділігін және өңделуін жақсы анықтайды.

Жалпы жағдайда күйзеліс тестілеу әдістемесі бағдарламалық қамсыздандырудың жүктемелерде өнімділігінің көрсеткіштерін талдауға негізделген, ол сүйемелдеу сатысында күтетіннен біршама артық, және мақсаты қосымшаның қолданылу жөніндегі белсенділікті артқан жағдайда беріктілігі немесе шыдамдылығын анықтау болып табылады.

Күйзеліс тестілеудің қажеттілігі, ең алдымен, жүйенің істен шығу құны тым шұғыл жағдайларда өте жоғары болуы мүмкін фактормен беріледі.

Күйзеліс тестілеуді қолданудың негізгі бағыттары:

- шекті жүктемелерде жүйе әрекетінің жалпы зерттелуі;
- шекті жүктемелерде айрықша жағдайларды және қателерді өңдеуді зерттеу;
- жүйенің тар орындарын немесе диспропорционал жүктемедегі жеке компоненттерді зерттеу;
- жүйенің сыйымдылығын тестілеу.

Күйзеліс-тестілеу жүктеме тестілеу сияқты жүйенің ұзақ уақытқа әрекет етуіндегі өзгерістер динамикасы алу мақсатында өнімділіктің

өзгерістер талдау үшін қолданылуы мүмкін.

Күйзеліс тестілеу жеке бағдарламаларға сияқты, клиент-серверлік архитектурасы бар таратылған жүйелер үшін қолданылуы мүмкін. Күйзеліс тестілеу шарттары әдетте бағдарламалық қамсыздандырудың функционалдылығының апатты процестерінен қалыптасады, олар тәуекелге талдау мен талаптарды әзірлеу сатыларында айқындалған.

Жалпы жағдайда шарт ретінде күйзеліс тестілеу үшін сызықтық ұлғайған күтілетін жүктеме қолданылуы мүмкін. Көп буынды таратылған жүйелерді тестілеген жағдайда көптеген элементтерден тұратын жүктеменің нақты көлемі ғана емес, сондай-ақ жалпы көлемдегі оның пропорциясынан тұратын жүктеме көлемін ескеру керек. Күйзеліс тестілерінде диспропорционал жүктемені пайдалану да жүйенің жеке компоненттерінің тар орындарын анықтау үшін қолданылуы мүмкін.

Сыйымдылықты тестілеу – күйзеліс-тестілеудің маңызды бағыттарының бірі және талдау жүргізу жағынан алғанда ең күрделі болып табылады. Сыйымдылықты тестілеу өнімділікке қойылатын талаптар толық сәйкес болғанда жүйенің беріктілік қорын анықтау мақсатымен жүргізіледі. Бұл жағдайда ағымдағы жүктеме жүйеге бір уақытта келіп түсетін сұратулардың саны мен пропорциясы сияқты, перспективада күтілудегі жүктеме есепке алынады. Сыйымдылықты тестілеу нәтижесі максималды мүмкін жүктеме сипаттамаларындағы жиыны болып табылады, бұл ретте жүйе сәулетті жобалау сатысында әзірленген және жобаланған өнімділікке қойылатын талаптарға жауап береді.

Тұрақтылықты тестілеу

Тұрақтылықты тестілеу мақсаты бағдарламалық қамсыздандырудың жұмысқа қабілеттілігін ұзақ тестілеу кезінде күтілетін жүктеме деңгейімен тексеру болып табылады.

Бағдарламалық қамсыздандыруды экстремалды жүктемелерге түсірмес бұрын, болжанған жұмыс ағдайларына тұрақтылығына тексеру жүргізу керек, яғни өнімді ақиқатқа жақын жағдайларда сынау. Тестілеуді өткізу ұзақтығы алғашқы мәнді емес.

Негізгі міндеті - ресурстардың тұтынылуын қадағалай отырып, деректерді өңдеу жылдамдығы және/немесе қосымшаның шақырылу уақыты тестінің басында және уақыт өте келе азайған жоқ. Керісінше жағдайда өнімнің жұмысында іркілістер болуы мүмкін және жүйенің шамадан тыс жүктелуі мүмкін.

Тұрақтылықты тестілеуді күйзеліс тестілеумен жиі үйлестіреді, яғни тек тұрақтылықты ғана емес, сонымен қатар бағдарламалық қамсыздандыруды қатаң жағдайлардан шығуы мен ұзақ уақытқа күшті жүктемелерге қабілеттілігін тексереді.

Конфигурациялық тестілеу

Конфигурациялық тестілеу - бұл жүйеге конфигурациядағы өзгерістердің өнімділікке ықпалы (алдыңғы тестілеу түрлеріне қарағанда берілетін жүктеменің артуы жағынан қарағанда).

Конфигурациялық тестілеу - БҚ түрлі бағдарламалық және аппараттық орталарда жұмысын тексеру.

Бұл тестілеу түрі, егер ақпараттық өнім қолданылатын болса, мысалы, әртүрлі платформаларда, әртүрлі браузерлерде, драйверлердің түрлі нұсқаларын қолдайтын жағдайда қолданылады.

Жоба типіне қарай конфигурациялық тестілеу түрлі мақсаттарда болады:

- өнімділіктің талап етілетін сипаттамалары мен тестіленетін жүйенің жауап беру уақытын қамтамасыз ететін жабдықтың оңтайлы конфигурациясын анықтау;
- тестілеу нысанының сипаттамада жарияланған жабдықтармен, операциялық жүйелермен және үшінші фирмалардың бағдарламалық өнімдерімен үйлесімділігіне тестілеу.

Конфигурациялық тестілеу жүктеме, күйзелік немесе тұрақтылыққа тестілеумен де үйлестірілуі мүмкін.

8.5. РЕГРЕССИОНДЫ ТЕСТІЛЕУ

Регрессионды тестілеу - бағдарламалық қамсыздандыру алғашқы кодтың тестіленген учаскелерінде қателерді анықтауға бағытталған тестілеу.

Бағдарламаға өзгертулер енгізгеннен кейін ол жұмыс істеуін тоқтататын болса, онда мұндай қателер *регрессионды қателер* деп аталады.

Регрессионды тестілеу мыналарды қамтиды:

- жаңадан табылған ақаудың түзетілуін тексеруді;
- бұрын түзетілген және анықталған ақау жүйеде қайта туындамауын тексеруді;

- бұрыннан жұмыс істеп тұрған функционалдылықтың жұмысқа қабілеттілінің бұзылмауын тексеру, егер оның коды қозғалған болса немесе кейбір ақауларды функционалдылықта өзгерткен кезде.

Әдетте регрессивті тестілеудің қолданылған әдістері алдыңғы тестілердің өткен жолдарын қамтыған болса, ал тексерулер өз кезегінде кодтарды біріктіру нәтижесінде регрессионды қателерге түскен жоқ па екендігін тексереді.

Сол бір қателердің қайтадан шығуы нұсқаларды басқарудың әлсіз техникасынан немесе адами қателердің себебінен болады. Сонымен қатар, кодтың бір бөлігін жазып алғанда алдыңғы іске асыруда болған сол қателер жиі қалқып шығады. Сол себепті қателерді түзеткен кезде оған тесті құру және оны бағдарламаның келесі өлшеулерінде жиі өткізіп тұру жақсы практика болып саналады. Регрессионды тестілеу қолмен де орындалуына қарамастан, бірақ бұл көбінесе регрессионды тестілердің барлығын автоматты түрде орындауға мүмкіндік беретін арнайы мамандандырылған бағдарламалардың көмегімен жасалады. Кейбір жобаларда белгіленген уақыт аралығы арқылы регрессионды тестілерді автоматты түрде өткізу үшін құралдар қолданылады. Автоматтандырылған тестілерді қолдану тексеру процесінің еңбек сыйымдылығын жеңілдетеді.

Бағдарламалық қамсыздандыруды тестілеу бұл бүкіл бағдарламаны қамтитын және кезекті жұмыс нұсқасын бағдарламалаушы мамандар тапсырғанда әрбір рет орындалатын тестілердің толық жинағы. Алдымен кітапханада шынымен қажетті тестілерден көп тестілер болуы мүмкін. Оған шекті минимум жағдайлар мен уақытша сипаттамаларын тексеруге арналған мысалдар кіруі керек. Тестілердің кейбіреуі тестілеуді алғаш өткізгенде қателерді шығаруы мүмкін. Бірақ олардың толық түзетуінен кейін бұл тестілерді қайта орындаудың пайдасы жоқ. Мұндай тестілерді кітапханадан алып тастайды, тек қателер анықталғандары ғана қалады.

Регрессивті тестілердің кітапханасын құрудың кейбір ережелерін қалыптастыруға болады.

Кітапхананың басқа тестілеріне эквивалентті тестілерді жою керек. Ең дегенде, бұл тестілер, әрине, кітапханаға түсуі керек. Кітапхананы бірнеше қызметкер құрған жағдайда мұндай жағдайлардың болуы мүмкін.

Нысаны түзетілген қате болып табылатын тестілердің санын азайту керек. Егер қате немесе оның кейбір түрлері тестілеудің циклдарының бірқатарында байқалатын болса, онда кітапханаға оларды анықтау үшін жеткілікті тестілерді қосу керек. Бірақ, бағдарламалық қамсыздандырудың тиісті бөлігі тестіленгеннен кейін (қателер

түзетілген), түзетілген қателерді іздеуге бағытталған тестілердің көптеген бөлігін кітапханадан жойып тастауға болады.

Келесі тестілеу кезінде тестілерді үйлестіру керек. Тестілеудің басында мұны жасаудың қажеті жоқ, бірақ кейіннен тестілерді біріктіру жұмысты біршама жеделдетеді.

Тестілеу автоматтандырылған болуы керек. Егер тестілердің белгілі бір тобы тестілеудің келесі бірнеше циклдары ішінде орындалатын болса, онда бұл процесс автоматтандырылуды қажет етеді.

Регрессионды кітапхананың барлық тестілерін бағдарламаның әрбір өзгеруінен кейін орындау міндетті емес. Мұны сирек жасау керек - циклдың әрбір екінші немесе үшінші цикында. Тестілеудің соңғы сатысында барынша ықтимал тестілер санын орындаған жақсы, ол бағдарламаның шығаруға дайындығына көз жеткізу үшін, басқа циклдарда тестілердің жартысы немесе тіптен үштен бір бөлігі жеткілікті болады.

Регрессионды кітапхана ең үздік тестілерді қамту керек, олар әзірленген болуы керек, бірақ ол тым үлкен болса, жаңа тестілерді әзірлеуге уақыт қалмайды. Ал, жаңа тестілер ғана үлкен ықтималдылықпен табылған қателерді ғана анықтайды. Сол себепті жұмысты регрессионды кітапхана тестілеу тиімділігін арттыратын құрал болып қызмет ететіндей жоспарлау керек, ол оны тежемейтін болуы керек.

БАҚЫЛАУ СҰРАҚТАРЫ ЖӘНЕ ТАПСЫРМАЛАРЫ

1. Тестілеу деп нені айтады?
2. «Ақ жәшік» тестілеуінің «қара жәшік» тестілеуімен салыстырғанда қандай артықшылықтары бар?
3. БҚ түрлі деңгейлерде тестілеу қалай өтеді?
4. Интеграциялық тестілеу жүйелік тестілеуден қалай ерекшеленеді?
5. Модульді тестілеудің ерекшелігі неде?
6. Күйзеліс тестілеу жүктеме тестілеуден қалай ерекшеленеді?
7. Жүйелік тестілеу тестілерінің санатын атаңыз.
8. Шығу тестілерінің қабылдау тестілерінен қалай ерекшеленеді?
9. Бета-тестілеудің айырмашылықтары қандай?
10. Альфа-тестілеу қалай жүргізіледі?
11. БҚ жүктеме тестілеуі нәтижесіндегі өнімділік көрсеткіштері қандай?

12. Күйзеліс тестілеудің әдістемесі неге негізделген?
13. Күйзеліс тестілеуді жүргізу қажеттілігі неден болады?
14. Тұрақтылық тестілеуінің негізгі міндеттері қандай?
15. Конфигурациялық тестілеудің мақсаты қандай?
16. Регрессионды тестілеу нені қамтиды?
17. Регрессионды тестілеудің кітапханасы нені білдіреді?

9-бөлім

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ЕНГІЗУ ЖӘНЕ ПАЙДАЛАНУ

9.1. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУ НҰСҚАЛАРЫН ЖӘНЕ ЖЕТКІЗІЛУІН БАСҚАРУ

Бағдарламалық қамсыздандыруды жеткізуді басқару үшін оның әзірлеумен айналысатын ұйымда бағдарламалық қамсыздандыруды жеткізу процедурасы айқындалуы керек. Бағдарламалық қамсыздандыруды жеткізу процедурасы бағдарламалық қамсыздандыруды әзірлеумен айналысатын ұйымның ішінде әзірленеді және бекітіледі, бірақ тапсырыс берушінің талаптарына байланысты ол нақты жоба үшін түрі өзгертілуі мүмкін.

Бағдарламалық өнімді жеткізу процедурасы жеткізу процесінде орындалуы керек, кезектілігімен көрсетілген негізгі әрекеттер мен міндеттердің тізбесін қамтиды.

Бағдарламалық өнімнің нұсқасы *(version)* деп бағдарламалық осы өнімнің басқа даналардан ерекшелігі бар бағдарламалық өнім данасы деп атайды. Жаңа нұсқалар функционалдылығымен, тиімділігімен немесе ескі нұсқалардағы қателерінің болмауымен ерекшеленуі мүмкін. Кейбір нұсқалары бағдарламалық немесе аппараттық қамсыздандырудың конфигурациясымен ажыратылуы мүмкін, олар әзірленген, бұл ретте бірдей функционалдылыққа ие болады.

Шығу нұсқасы *(release)* - бұл БҚ жеткізу болып табылатын, яғни тапсырыс берушіге жеткізілетін бағдарламалық қамсыздандырудың нұсқасы. Әрбір жаңа шығатын нұсқада жаңа функционалдық мүмкіндіктер болады, не болмаса бұл нұсқа жаңа аппараттық платформа үшін әзірленген. Нұсқаларының саны, негізінде жеткізу санынан артық болады, себебі барлық нұсқалары тапсырыс берушіге жеткізіледі, ал бағдарламалық қамсыздандыруды әзірлеу процесінде ішкі пайдалану үшін құрылады.

Жеткізілетін бағдарламалық қамсыздандырудың әрбір нұсқасының өзінің белгілі бір нөмірі бар. Бағдарламалық қамсыздандырудың

нұсқасы A.B.C.D схемасына сәйкес нөмірленеді, мұнда:

- A - бағдарламалық қамсыздандырудың мажор нұсқасы (major version);
- B - бағдарламалық қамсыздандырудың минор нұсқасы (minor subversion, аралық нұсқасы);
- C - бағдарламалық қамсыздандыру «релизі» (release);
- D - бағдарламалық қамсыздандыруды жинау (build).

Сондай-ақ, бағдарламалық қамсыздандырудың қарапайым нөмірі қолданылуы мүмкін - A.B (мысалы, пайдалану, жарнамалық және маркетинг құжаттарында көрсеткенде, веб-сайтта және т.б.).

Бағдарламалық қамсыздандырудың мажор нұсқасының нөмірінің өзгеруі өнімнің функционалы ауқымды өзгергенде болады.

Бағдарламалық қамсыздандырудың минор нұсқасының нөмірінің өзгеруі мүмкін:

- ескі нұсқамен бағдарламалық үйлесімділікке әкелетін жаңа функционалдылық өнімге енгізу (деректер деңгейіндегі үйлесімсіздік);
- өнімнің функционалдылық схемасындағы өзгерістер;
- жаңа өнімде жаңа бәсекелестік артықшылықтардың пайда болуымен жаңа функционалдылықты ұлғайту, қосу. Бағдарламалық қамсыздандыру релизінің нөмірін өзгерту жоғарыда аталмаған бағдарламалық қамсыздандыруды көпшілік шығаруда болады. Релиздердің нөмірлері қателердің түзетілу шығыстарын белгілейді, олар өнімнің функционалдылық схемасына өзгерту енгізбейтін және деректердің файлы деңгейінде үйлесімсіздікті енгізеді. Өнім релиздерінің нөмірленуі 0-ден басталады (1.0.0-нұсқа – өнімнің нарыққа алғаш шығуы). Өнімнің жаңа аралық нұсқасы шыққан кезде релиз нөмірленуі нөлдік мәннен басталады. Бұл ретте өнімнің алдыңғы аралық нұсқаларының релиздерін шығару мүмкін (осы немесе өзге техникалық себептер бойынша, пайдаланушыларды қолдау үшін).

Бағдарламалық қамсыздандыруды жинау нөмірлерін өзгерту өнімді жаңа жинауда болады (бағдарламалық қамсыздандыру компиляциясы ішкі мақсаттар үшін). Өнімнің жиналу нөмірленуі 1-ден басталады (0.0.0.1 - өнімнің прототипін алғашқы жинау). Жинау нөмірі өнімнің жаңа нұсқасы шыққанда тасталуы мүмкін (әзірлеме бөлімінің шешімі бойынша).

Қазіргі уақытта нұсқаларды басқаруды қолдау үшін көптеген CASE-құралдар әзірленген, олардың көмегімен әрбір нұсқаны сақтауды басқаруды жүзеге асырады және БҚ компоненттеріне қатынауды бақылау.

БҚ жаңа нұсқаларын шығару мен енгізудің себептері болуы мүмкін:

- алдыңғы нұсқаларды пайдалану процесінде анықталған қателерді түзету қажеттілігі;
- алдыңғы нұсқаларды жетілдіру қажеттілігі, мысалы, интерфейсті жақсарту немесе орындалатын функциялардың құрамын ұлғайту;
- жұмыс істеу ортасының өзгеруі, мысалы, жаңа техникалық құралдардың және/немесе бағдарламалық өнімдердің пайда болуы.

Жеткізуге дайын бағдарламалық қамсыздандыру келесі түрде жіктелуі мүмкін:

- ES-жеткізу - прототипті жеткізу;
- PA-жеткізу - альфа-нұсқаларды жеткізу;
- PB-жеткізу - бета-нұсқаларды жеткізу;
- RP-жеткізу - қорытынды жеткізу.

Прототип бағдарламалық өнімнің бастапқы нұсқасы болып табылады, ол жүйеде салынған концепцияларды көрсету, талаптар нұсқасын тексеру үшін қолданылады, сондай-ақ БӨ әзірлеу барысында да, пайдалану кезінде де туындауы мүмкін проблемаларды іздеу және осы шешімдердің мүмкін нұсқаларын көрсетуге қолданылады.

Бағдарламалық қамсыздандырудың альфа-нұсқасы БҚ барлық немесе барлық дерлік негізгі функцияларын іске асырады, алайда олардың кейбіреулері болмауы немесе тыс тұрақты емес қателермен орындалуы мүмкін. Альфа-нұсқаны әдетте тапсырыс берушіге өнімді әзірлеудің алғашқы сатыларында анықтау мақсатымен ғана береді, бірақ ол толығымен немесе барлық дерлік функциялары жүзеге асырылған болуы керек.

Бета-нұсқада бағдарламалық қамсыздандырудың жоспарланған функцияларының толық жинағы іске асырылған. Альфа-нұсқаларға қарағанда бета-нұсқалар тараптық пайдаланушыларды тартуды болжайды.

Бағдарламалық қамсыздандыруда керемет қиын қателер жоқ, маңызды қателер өте аз. Барлық жобалық құжаттар дайын және бағдарламалық қамсыздандыру тапсырыс берушінің талаптарына сәйкес. Анықтамалық жүйе мен пайдаланушының құжаттамасы да толық дайын. Бұл қарқынды пайдалану да бағдарламалық қамсыздандырудың жоспарланған функцияларының толық жиынымен дайын деуге болады.

Оның мақсаты бағдарламалық қамсыздандырудың жұмысындағы қателердің барынша көп санын анықтау болып табылады, оларды нарыққа, көпшілік тұтынушыларға өнімді шығару алдында қорытынды жою үшін керек.

Бұдан басқа, бета-нұсқа жарнамалық мақсаттарда нарықта өнімді алға жылжыту стратегиясының бөлігі ретінде қолданылуы мүмкін

(мысалы, бета-нұсқалардың тегін таратылуы пайдаланушылардың өнімнің қымбат нұсқасына тартуға мүмкіндік береді), сондай-ақ болашақ пайдаланушылардың кең ауқымынан ол туралы алдын-ала пікір алу үшін қолданылуы мүмкін.

Бета-нұсқа өнімнің қорытынды нұсқасы болып табылмайды, сол себепті әзірлеуші қателердің толық болмауына кепілдік бермейді, олар деректердің жоғалуына және/немесе компьютердің жұмысын бұзуы мүмкін.

Қорытынды жеткізу БҚ әзірлеу жөніндегі жобамен жұмысты аяқтауды және сүйемелдеу сатысына өтуді білдіреді. Өнім барлық жолдама құжаттамасымен тапсырыс берушіге беріледі.

9.2. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ ӨМІРЛІК ЦИКЛЫН СҮЙЕМЕЛДЕУ САТЫСЫ

БҚ сүйемелдеу бағдарламалық өнімді өзгертуді басқаруды, оның ағымдағы күйі мен функционалдылық жарамдылығын басқаруды, сондай-ақ жаңа жағдайларға жеткізілетін бағдарламалық қамсыздандыруды бейімдеуді білдіреді. Негізінен, сүйемелдеу процесінде енгізілетін өзгертулер бағдарламалық қамсыздандырудың негізгі функцияларына қатысты емес, алайда өмірлік циклдың әрбір сатысында қабылданатын жобалық шешімдерді қайта қарауды талап етуі мүмкін.

Сүйемелдеу сатысының мақсаты БҚ пайдаланушыларын қолдау, БҚ жақсарту, оңтайландыру мен ақауларын жою оны пайдалануға бергеннен кейін қолдау болып табылады.

БҚ сүйемелденуі - пайдаланушының өзгеретін қажеттіліктеріне сәйкес қателерді жою үшін және түрін өзгерту үшін бағдарламалық өнімге өзгертулерді енгізу бойынша күш-жігерді азайтуға мүмкіндік беретін бағдарламалық өнімнің сипаттамаларының жиынтығы.

Сүйемелдеудің үш типі ажыратылады.

1. *Түзететін сүйемелдеу,* әдетте талаптардың сипаттізімін өзгертуді талап етпейтін қателерді анықтауға жауап ретінде орындалатын қателерді түзету деп аталады. Түзетіп сүйемелдеу кезінде бағдарламалық қамсыздандырудағы ақауларды диагностикалау мен түзету жүйені жұмысқа қабілетті түрінде сақтау үшін жүргізіледі.

2. *Бейімделген сүйемелдеу* орындау ортасы немесе деректерді өзгерту талаптарына жауап ретінде жүзеге асырылады. Ол

қолданылады, қолданыстағы жүйе жақсарғанда немесе ұлғайғанда, ал талаптардың сипаттамасы жаңа функцияларды іске асыру мақсатымен өзгертіледі.

3. *Жетілдіруші (прогрессивті) сүйемелдеу* жүйенің жұмыс тиімділігін арттыру немесе оны сүйемелдеу тиімділігін арттыру мақсатымен кез келген өңдеуді қамтиды.

Бірнеше сүйемелдеу желілерін ажырату қажет:

- 0-желі (call-center, ақпараттық орталық, жедел желі) - клиенттерден телефон хабарларын өңдеу, хабарламаларды техникалық мамандарға беру (1-сүйемелдеу желісі);
- 1-желі (сүйемелдеу жөніндегі инженер, техникалық қолдау инженері, support engineer) - бағдарламалық қамсыздандыру жұмысындағы қателерді жою, кеңес беру, баптау, білім базасын толықтыру, пайдаланушылар нұсқаулығын құру;
- 2-желі (сүйемелдеу жөніндегі инженер, техникалық қолдау инженері, support engineer) - функционалды сүйемелдеу, тапсырыс берушінің жабдығына бағдарламалық қамсыздандыруды іске қосу сатысындағы жобалық қызмет;
- 3-желі (сүйемелдеу жөніндегі инженер, техникалық қолдау инженері, support engineer) - тапсырыс берушінің жабдығында бағдарламалық қамсыздандыруды іске қосу сатысындағы жүйелік сүйемелдеу, жобалық қызмет.

Сүйемелдеу жөніндегі инженердің жұмысын ақпараттық орталық жұмысымен қате салыстырады. Алайда, функционалы бойынша бұл мамандарды ажыратады - егер call-center пайдаланушылардың жүгінуін нақты аккумуляциялайтын болса, онда сүйемелдеу БҚ әзірлеу мен толықтыру тізбегінде орталық буын болып табылады, БҚ пайдалану кезеңінде (жүйе, сервис) пайда болатын проблемаларды шешеді.

Сүйемелдеу жөніндегі инженер (техникалық қолдау инженері, support) - әзірлеудің барлық сатыларында процестер, сервистердің дұрыс жұмыс істеуін қолдаумен айналысатын маман.

Сүйемелдеу екі түрге бөлінеді: ішкі және сыртқы. *Ішкі ақпараттық-технологиялық сүйемелдеу* ұйымның ішкі міндеттерін шешуге бағытталған: маман бағдарламалық қамсыздандыру пайдаланушыларымен өзара әрекет етеді, ішкі ақпараттық ресурстармен (бағдарламалық қамсыздандыру, портал, жұмыс деректер қоры және басқалары).

Сыртқы ақпараттық-технологиялық сүйемелдеу тапсырыс беруші (клиент) пайдаланатын сервистерді, бағдарламалық қамсыздандыру мен қызметтерді сүйемелдеуге бағытталған.

Сүйемелдеу жөніндегі инженер жоспарлау дағдыларын меңгеруі

керек және ұйымдасқан, көптеген көп міндеттілік режимінде жұмыс істеуге қабілетті болуы керек.

Инженердің міндеті - қысқа мерзімде проблеманы жою, ол сүйемелденетін бағдарламалық қамсыздандырудың, сервистің тақырыптық саласында жақсы түсінік білуі керек және аралас бөлімшелердің техникалық тапсырмасын құра білуі керек. Бұл бағдарламалау тілдерін, деректер қорын, бағдарламалық қамсыздандыруға әкімшілік ету дағдыларын, аналитикалық ойлау мен зерделілікті білусіз мүмкін емес.

9.3. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ ЭКОНОМИКАЛЫҚ ТИІМДІЛІГІН БАҒАЛАУ

Бағдарламалық қамсыздандырудың құнын бағалау процесі біршама күрделі міндет болып табылады. Ол бағдарламалық өнімге қойылатын талаптарды талдау сатысынан басталады да, іске асыру сатысында жалғаса береді. Осы міндетті шешуге көптеген тәсілдер бар, бірақ кепілді нәтиже беретін әмбебап әдістеме жоқ.

Пайда болған жағдайларда бағдарламалық қамсыздандыруға салымдардың тиімділігін бағалау проблемасы тым өзекті болады. Ақпараттық технологияларға инвестициялардың қажеттілігін қалай негіздеуге болады? Бағдарламалық өнімдерді пайдаланудан түсетін пайда мен осы инвестициялардың арасындағы өлшемдердің саналы өрнегіне қалай жетуге болады? Осы сауалдарға жауап беріп көрейік.

Бағдарламалық қамсыздандырудың тиімділігі - бұл өз мақсатына сәйкестілік дәрежесі. Экономикалық тиімділікті бағалау ақшалай түрде өрнектелген шығындар мен нәтижелерді салғастыруға негізделген. Экономикалық тиімділікке қойылатын негізгі талаптар талап етілген тұрақтылыққа нәтижелер мен шығындардың арақатынасын және олардан асатын талаптар болып табылады.

Қазіргі таңда *бағдарламалық жобаларды енгізудің экономикалық тиімділігінің келесі көрсеткіштерін* бөліп көрсетеді:

- ішкі табыс нормасы (IRR - Internal Rate of Return);
- таза берілген құны (NPV - Net Present Value);
- ақталу мерзімі (PB - Payback Period);
- иелік етудің бірлескен құны (TCO - Total Cost of Ownership);
- инвестицияларды қайтару нормасы (ROI - Return of Investment);

- бизнес үшін мүмкіндіктер құндылығы (TVO - Total Value of Opportunity).

Көбінесе ұйымдар бағдарламалық қамсыздандыруды сатып алуға жұмсалатын тура шығындарды ғана есепке алады, ал жанама шығындар сүйемелдеуге және қолдауға (жаңарту және басқалары) байланысты шығындар негізінен, ескерілмейді.

Бағдарламалық өнімдерге инвестициялар салудың тиімділігін бағалау міндетін шешудегі шетелдік тәжірибе *ақпараттық технологияларға иелік етудің бірлес құнын бағалау* (Total Cost of Ownership - TCO) тұжырымдамасы болып табылатын кеңінен таралған әдістерінің бірі екендігін көрсетеді.

TCO деп бағдарламалық өнімді оның пайдаланудан шыққан сәтіне дейін жұмыс істеуін қамтамасыз етуге және енгізуге жұмсалған барлық шығындарды сомасы түсіндіріледі. Иелік етудің бірлескен құнын есептеудің екі негізгі моделі бар: Gartner Group ұсынылған тұжырымдама және Microsoft және Interpose бірлескен күшінің нәтижесі. Gartner ұсынған әдістеме бойынша барлық шығындар белгіленген және ағымдағы болып бөлінеді.

Белгіленген шығындар жүйені енгізу сатысында бір рет жүргізіледі. Оларға мыналар жатады: жобаны әзірлеу және енгізу құны, аппараттық және бағдарламалық қамсыздандыруды енгізуге қажетті бастапқы сатып алулар, сырттан кеңесшілерді тарту.

Ағымдағы шығындар - жүйенің жұмыс істеуін қамтамасыз ететін шығындар. Бұл жүйе жұмыс істеп тұрғанда талап етілетін шығындар. Оларға мыналар жатады:

- жүйені жаңарту мен жаңғырту;
- жалпы жүйені басқару - әкімшілік ету, әкімшілікті және соңғы пайдаланушыларды оқыту, қызметкерлердің жалақысы, сыртқы ресурстарды тарту;
- «пайдаланушының белсенділігі» - бағдарламалық қамсыздандыруды толықтырып жасау, бағдарламалық қамсыздандыруды қосымша баптау, деректермен жұмыс істеу, пайдаланушының құзыретті емес әрекеттерінің салдары.

Барлығы қарапайым сияқты - тек жоғарыда аталған баптардың әрбірі бойынша шығындарды ғана санау керек. Бірақ барлық шығындарды санау жеңіл емес - олардың біршама бөлігі алдын-ала салынбайды да, ешбір жерде есептелмейді де. Мұнда TCO құрамына кіретін шығыдардың 75% соңғы пайдаланушылардың проблемаларына негізделген. Microsoft және Interpose әзірлеген TCO моделінде әсіресе осы шығындар есепке алынады. Олардың әдістемесіне сәйкес шығындар *тура және жанама* деп бөлінеді.

Иелік етудің бірлескен құнын қарапайым формула бойынша есептеуге болады

$$ТСО = Тш + Жш,$$

мұндағы Тш - тура шығындар; Жш - жанама шығындар.

Тура шығындар - бюджетпен қарастырылатын және жоспарланатын шығындар, жанама шығындар жоспарлауға түспейтін және тіпті тіркелмейтін орта шығындардың 50%-дан артығын құрайды. Оларға, ең алдымен, барлық пайдаланушы шығындар (формалды емес оқыту, дербес қолдау, қателер мен есептер) және істен шыққан немесе жоспарлық сақтандыру тоқтауларының есебінен жатады. Көбінесе жанама шығындар бағдарламалық өнімнің жұмысындағы кемшіліктермен туындаған уақытты жоғалтуға байланысты.

Салымдардың тиімділігі (инвестицияларды қайтару, ROI) - бұл бағдарламалық өнімдерді енгізу есебінен сипаттайтын көрсеткіш. Ол инвестициялардың бастапқы құнына берілген бағдарламалық өнімді енгізуден күтілетін дисконтты түсімдердің өрнегі ретінде есептеледі:

$$ROI = Эф / А,$$

мұндағы Эф - ақша бірліктерімен өрнектелетін енгізу тиімділігі; А - ақпараттық жүйелерге инвестициялар (ИТ).

Модель ROI де Gartner Group тиесілі. Табыс бөлігін бағалау үшін, негізінен, бағдарламалық жобаны енгізу арқылы немесе қандай да бір бағдарламалық жаңа өнімдердің пайда болуы арқылы қол жеткізілетін мақсат бизнесіне талдау жасалады. Бизнестің өлшеуші көрсеткіштерін алады, мысалы, операциялық шығындарды қысқарту, бәсекеге қабілеттілік күйін қолдау, ішкі бақылауды жақсарту, және тиімділікті бағалайды. Шығын бөлігі ретінде көбінесе ТСО көрсеткішін жиі қолданады.

БАҚЫЛАУ СҰРАҚТАРЫ ЖӘНЕ ТАПСЫРМАЛАРЫ

1. Бағдарламалық өнімнің нұсқасы деп нені атайды?
2. Бағдарламалық өнімнің шығыс нұсқасының ерекшелігі неде?
3. Бағдарламалық өнімнің мажор нұсқасы деген не?
4. Бағдарламалық өнімнің минор нұсқасының ерекшелігі неде?
5. Бағдарламалық қамсыздандырудың нұсқалары қандай ережелер бойынша нөмірленеді?
6. Бағдарламалық қамсыздандырудың минор нұсқасы нөмірінің өзгеруі неден болады?
7. Бағдарламалық қамсыздандыру релизі нөмірінің өзгеруі қандай

жағдайларда болады?

8. Жаңа бағдарламалық қамсыздандыру нұсқаларын шығару мен енгізудің себептері қандай болады?
9. Бағдарламалық өнімнің альфа-нұсқасы мен бета-нұсқасы қалай ерекшеленеді?
10. Бағдарламалық қамсыздандырудың сүйемелденуі деген не?
11. Сүйемелдеудің қандай типтері болады?
12. Жетілдірілетін сүйемелдеу бейімделуден қалай ажыратылады?
13. Сүйемелдеудің қандай желілері бар?
14. Сыртқы сүйемелдеу сыртқы сүйемелдеуден қалай ерекшеленеді?

Бағдарламалық қамсыздандырудың экономикалық тиімділігін есептеген кезде қандай шығындар ескеріледі?

II

ТАРАУ

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ӘЗІРЛЕУДІҢ АСПАПТЫҚ ҚҰРАЛДАРЫ

10-тарау

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ӘЗІРЛЕУ ҚҰРАЛДАРЫ

10.1. АСПАПТЫҚ БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУ

Аспаптық бағдарламалық қамсыздандыру - бұл, бағдарламаны жобалау, әзірлеу және сүйемелдеу барысында пайдалануға арналған бағдарламалық қамсыздандыру.

Аспаптық бағдарламалық қамсыздандыру кез келген салада, желілік және қолданбалы бағдарламаларды қоса алғанда бағдарламалық өнімдерді жасау үшін қолданылады. Бүгінгі уақытта бағдарламалық өнімдерді жасау үшін көзбен шолу бағдарламалаудың қуатты жүйелері қолданылады, олар өзіне стандартты бағдарламалардың ауқымды кітапханасын, дұрыстау және тестілеудің арнайы құралдарын қамтиды.

Бағдарламалау жүйелері нақты бағдарламалау тілінде жаңа бағдарламаларды әзірлеуге арналған және өз құрамына компиляторларды, интерпретаторларды, диалогтік ортаны, мәтін редакторларын, стандартты ішкі бағдарламалар кітапханаларын, құрастырушыларды, ретке келтірушілерді, анықтама қызметтерін, автоматты тестілеу құралдарын, нұсқаларды басқару жүйелерін, парсерлерді және т.б. қамтиды.

Компиляторлар - жоғары деңгейлі тілде жазылған бағдарламаны машиналық тілдегі эквивалентті бағдарламаға немесе нысанды модульге ауыстыратын бағдарламалар. Компилятор бағдарламаның машиналық тілдегі аяқталған нұсқасын құрады, ол содай кейін ЭЕМ орындалады. Компилятордың жұмыс істеу үрдісі компиляция деп аталады.

Трансляторлар - бағдарламаның трансляциясын орындайтын бағдарламалар немесе техникалық құралдар.

Интерпретаторлар - командаларды талдайтын және оларды сол сәтте орындайтын бағдарламалар (кейде аппараттық құралдар) немесе бағдарлама операторлары.

Құрастырушылар - құрастыру жасайтын бағдарламалар, яғни кірісіне бір немесе бірнеше нысанды модульдерді қабылдайды және олар бойынша орындалатын модуль жинайды.

Бастапқы мәтіндердің препроцессорлары - бұл, кірісіне деректерді қабылдап алатын және деректерді беретін, компилятор сияқты басқа бағдарламаның кірісіне арналған компьютерлік бағдарлама.

Ретке келтіруші (debugger) әзірлеу ортасыны модулі немесе бағдарламадағы қателерді іздеуге арналған жеке қосымша болып табылады.

Мәтіндік редакторлар - мәтіндік файлдарды құруға және түзетуге, сондай-ақ оларды экранда қарап шығуға, басып шығаруға, үзінділер ідеуге және т.с.с. арналған компьютерлік бағдарламалар.

Бастапқы мәтіндердің арнаулы редакторлары - бағдарламаның бастапқы кодын құруға және редакциялауға арналған мәтіндік редакторлар. Бастапқы мәтіндердің арнаулы редакторы жеке қосымша болуы мүмкін немесе талдаманың кіріктірілген ортасына (IDE) кіріктіріле алады.

Ішкі бағдарлама кітапханалары - бағдарламалық қамсыздандыруды әзірлеуге пайдаланылатын ішкі бағдарламалар мен нысандардың жинақтары.

Бұдан басқа, аспаптық бағдарламалық қамсыздандыруға SDK, ассемблер, құжаттама генераторы жатады.

SDK (Software Development Kit) - бағдарламалық қамсыздандыру бойынша мамандарға белгілі бір бағдарламалар пакетіне, негізгі өңдеу құралдарының бағдарламалық қамсыздандыруына, аппараттық платформа, компьютерлік жүйеге, бейнеойын консолдеріне, операциялық жүйелерге және өзге платформаларға арналған қосымшалар жасауға мүмкіндік береді. SDK жеткізушілері кейде Software Development Kit сөз тіркесіндегі Software терминін дәлірек сөзге алмастырады. Мысалы, Microsoft құрылғылардың драйверлерін әзірлеу үшін Driver Development Kits (DDK) ұсынады.

Ассемблер - компьютерлік бағдарлама, машиналық кодтағы бағдарламаға ассемблер тілінде жазылған бағдарламаның бастапқы мәтіннің компиляторы. Тілдің өзі секілді, ассемблерлер, әдетте нақты сәулетке, операциялық жүйеге және тіл ситаксисінің нұсқасына тән болып табылады. Сонымен бірге, мультиплатформалық немесе әмбебап (нақтырақ айтқанда, шектеулі-әмбебап, себебі төменгі деңгейлі тілде аппаратты-тәуелсіз бағдарламаларды жазуға болмайды) ассемблерлер, олар түрлі платформаларда және операциялық жүйелерде жұмыс жасай алады.

Құжаттама генераторы - бастапқы кодқа ерекше түрде

түсіндірілген және, кейбір жағдайларда орындалатын модульдер бойынша (компилятордың шығысында алынған), бағдарламашыларға және/немесе жүйенің ақырғы пайдаланушыларына арналған құжаттарды алуға мүмкіндік беретін бағдарлама немесе бағдарламалар пакеті.

10.2. ҚОСЫМШАЛАРДЫ ӘЗІРЛЕУ ЗАМАНАУИ КІРІКТІРІЛГЕН ОРТАСЫНЫҢ ТҰЖЫРЫМДАМАСЫ

Бағдарламалық қамсыздандыруды әзірлеудің кіріктірілген ортасы (Integrated Development Environment, IDE) - бұл бағдарламашылармен бағдарламалық қамсыздандыруды әзірлеуде қолданылатын бағдарламалық құралдар жүйесі.

Әдетте, құрастыру ортасы мәтіндік редактор, компилятор және/немесе интерпретатор, құрастыруды автоматтандыру құралдарын және ретке келтірушіні қамтиды. Кейде, сондай-ақ құрамына нұсқаларды басқару жүйесі және пайдаланушының графикалық интерфейсін құрастыруды жеңілдетуге арналған әртүрлі құралдар кіреді. Көптеген заманауи әзірлеу орталары да кластар браузерін, нысандар инспекторын және бағдарламалық қамсыздандыруды нысанға бағытталған әзірлеуде пайдалану үшін кластар иерархиясының диаграммасын қамтиды.

Құрастырудың кіріктірілген ортасының (ҚКО) - құрастырманың өмірлік кезеңінің барлық негізгі фунцияларының орындалуын қолдайтын, жалпы интерактивті графикалық қабығы бар.

Құрастырудың интерактивті орталарының әрқайсысы үшін келесі компоненттердің болуы тән:

- функционалдық пернелерді кеңінен пайдалана отырып, басқа компоненттерді ортадан шықпастан шақыруды қамтамасыз ететін бірыңғай интерактивті қабық;
- бағдарламалардың бастапқы мәтіндерін теруге және редакциялауға арналған мәтіндік редактор. Жуырда өткен уақытта отандық дәстүрде дәл бастапқы мәтін термині ақырында бастапқы код (source code) термині ретінде қолданыла бастады;
- құрастыруды (build) сүйемелдеу жүйесі, яғни, бастапқы іске асырылатын тілінен компиляторды және нысанды бинарлы кодтарды бірыңғай атқарылатын кодтарға (толтыру модулі) құрастырушыны (linker) қосатын бастапқы кодтардан жобалар

компиляциясы; не операциялық жүйенің құрамына кіретін штаттық немесе аталмыш ортаға арналған құрастырушы қолданылады;

- ретке келтіруші (debugger) - әдеттегі командалар жиынтығы: тоқтау нүктесінің, рәсімдерді (әдестерді) қадамдық орындаудың, айнымалының мәндерін көзбен шолудың және т.б. көмегімен ортадағы бағдарламаны ретке келтіру үшін.

ҚКО әдетте IntelliJ IDEA, NetBeans, Eclipse, Qt Creator, Geany, Embarcadero RAD Studio, Xcode или Microsoft Visual Studio сияқты бағдарламалаудың бірнеше тілдеріне арналған, бірақ белгілі бір бағдарламау тіліне арналған ҚКО бар, мысалы, Visual Basic, Delphi, Dev-C++.

ҚКО жекелеген оқиғасы - бағдарламаның интерфейсін көзбен шолу редакциялауды қамтитын көзбен шолуды әзірлеу ортасы.

Кіріктірілген орталарда, заманауи мәтіндік редакторлар кодты автоматты түрде аяқтау режимін (code completion) қамтамасыз етеді, ол оларға әдепкі қалпы бойынша қосылған және орта редакторы құрастырушыға болжалды және оның синтаксистік дұрыс жалғасыны ойға салады, мысалы, жабушы жақшаның жоқтығы, нүктелі үтірдің болмауы; қайсыбір белгілі кластың нысанынан әдісті шақыру кезінде әдіс атауларының болжалды нұсқалары және т.б.

Кіріктірілген орталардың заманауи нұсқаларында келесі компоненттер бар.

Профильдеуші (profiler) - пайдаланушылардың белсендігін, кіріктіріген ортаның басқаруымен бағдарламаны орындау нәтижесінде алынған статистикалық деректерді жиынтықтау және талдауға арналған құралы: рәсімдерді (әдістерді) шақыру саны, бағдарламаны орындау кезінде қолданылатын жады көлемі және т.б.

Рефакторинг (refactoring) - кодты - бағдарламаның, оның сыртқы тәртібін қозғамайтын және оның жұмысын түсінуді жеңілдету мақсаты бар, ішкі құрылымын өзгертуін қайта ұйымдастыруға арналған аспаптар. Рефакторинг негізінде шағын баламалы (яғни, тәртіпті сақтайтын) өзгерістердің жүйелілігі жатыр. Әрбір өзгеріс кішкене болғандықтан, бағдарламашыға оның дұрыстығын бақылап отыру оңай, және сол мезетте барлық жүйелілік бағдарламаның едәуір өзгертілуіне және оның келісімділігі мен дәлдігінің жақсаруына алып келуі мүмкін. Әдеттегі осындай әрекеттерге, мысалы, оның анықтамасындағы және оған жасалған сілтемелердегі әдіс атауының, оның дәлелін қосу және т.с.с.

Тестілер генераторы (unit test generator) - модульдерді, әдістерді немесе дәлел мәндерінің әртүрлі ықтимал үйлесімдері бар процедураларды тестілеуге арналған типтік тестілерді іздестіру құралы.

Бастапқы кодтардың нұсқаларын бақылау жүйесі (source code control system) өзгермелі ақпаратпен жұмыс істеуді жеңілдетуге арналған. Нұсқаларды басқару жүйесі бағдарламаны сүйемелдеу барысында ортадағы жобалардың бастапқы файлдар кодтарының бірнеше нұсқаларын сақтауға мүмкіндік береді.

Бағдарламаны командалық құрастыруды сүйемелдеу құралдары (teamwork) - бағдарламаның өмірлік цикл кезеңдерін (талаптар және сипаттамалар, жобалау, өткізу, тестілеу), бағдарламашылар команда қатысушаларының ішіндегі құрастыру бойынша тапсырмаларды үлестіру, жоба менеджерімен тапсырманың орындалуын бақылау.

Кодты талдау құралдары (code analysis) - жоба кодын статистикалық және динамикалық талдауға арналған бағдарламалақ қамсыздандыру. Кодты статикалық талдау (static code analysis) - динамикалық талдаудан айырмашылығы зерттелетін бағдарламаларды шынайы орындалуынсыз жасалынатын бағдарламалық қамсыздандыру талдауы. Қолданылатын құралға байланысы талдаудың тереңдігі, барлық бастапқы кодты қамтитын, талдауға дейінгі жекелеген операторлардың мінез-құлқын анықтаудан өзгеруі мүмкін. Талдау барысында алынған ақпараттарды пайдалану тәсілдері де әртүрлі - қате болуы мүмкін жерлерді анықтаудан бастап, бағдарламаның қайсыбір қасиеттерін математикалық тұрғыдан дәлелдеуге мүмкіндік беретін ресми әдістерге дейін. Кодтың динамикалық талдауы (dynamic program analysis) - бағдарламаларды нақты немесе көзбен шолу процессорда (статикалық талдаудан айырмасы) орындаудың көмегімен өткізілетін бағдарламалық қамсыздандыру талдауы. Динамикалық талдаудың утилиталары арнайы кітапханаларды жүктеуді, бағдарламалық кодтың қайта компиляциясын талап етуі мүмкін. Динамикалық талдаудың үлкен тиімділігі үшін, кодты толығымен жабылуын алу үшін, тестіленетін бағдарламаға кіріс деректерінің жеткілікті мөлшерін табыстау талап етіледі.

Өндірілген бинарлық кодты көзбен шолу құралдары - әдістерді, айнымалыларды, олардың атауларын және т.б.

Кодты «шатастыру» құралдары (obfuscation), дәл осы мақсатта код элементтерінің аттарын - кластарды, әдістерді, өрістерді және т.б. түсініксіз, «кездейсоқ», «шатастырылған» аттарға алмастыруды орындайды, қайта қарастырылған бинарлық кодты зерттеуді қиындату мақсатында, рұқсат етілмеген жолмен өзіне кодтағы жаңа идеяларды иемденгісі келген қаскүнемдердің кодты «бұзуын» қорғау үшін немесе оны шабуыл ұйымдастыруды көздеген арам ниетті оймен оны зерттеу үшін.

Кодтың типтік үлгілерінің негізінде әртүрлі бағдарламалық

жоба және шешім түрлерін құруды сүйемелдеу; кеңейтулерді құру механизмі. Заманауи бағдарламалар құру барысында жиі әртүрлі қосымшалар құру талап етіледі - консолды, веб-қосымшалар және веб-сервистер, мобилді қосымшалар, бұлттық қосымшалар және басқалар. Осы түрлердің әрқайсысына бастапқы код файлдарының, сондай-ақ конфигурациялық файлдардың өзіне тән құрылымын жасау талап етіледі. Заманауи кіріктірілген орталар, бастапқы код үлгісін ұсына отырып және жоба үшін автоматты түрде қажетті конфигурациялық файлдарды түрлендіре отырып, әртүрлі жоба түрлерінің құрылуын автоматтандырады. Бүгінгі уақытта кодтың дайын үлгілерін пайдаланбастан бағдарламалауды елестету қиын, бұл қателікке алып келетіндігі сөзсіз, мысалы, қолмен қайсыбір файлды, жобаның ажырамас бөлігін құруды ұмытып кету және кодтың қайсыбір маңызды үзіндісін назардан шығарып алу өте оңай. Сол себептен әртүрлі жоба түрлерін кіріктірілген орталармен қолдау ерекше маңызды. Мұнан басқа, заманауи кіріктірілген орталарда жобалардың болжалды түрлерін жиынтықтау кеңейтілетін болып табылады.

UML біркелкіленген модельдеу тілінде бағдарламаның құрылымын модельдеуді сүйемелдеу. Заманауи кіріктірілген орталар екі бағыт бойынша UML тілін пайдалануды қолданады: бастапқы код бойынша модельді және сәйкес бағдарламаны іздестіру және, керісінше, бастапқы кодты (үлгісін) дайындалған үлгі бойынша іздестіру.

Әуелгі кезде кіріктірілген орталар қайсыбір бастапқы тілде бағдарламалауға дайындалды. Дегенмен, бара-бара осындай монотілдік кіріктірілген орталардың көптілдіктерге айналу беталысы байқалды, себебі әртүрлі тілдердегі жобаларды құру үшін ұқсас қағидалар мен механизмдер қолданылады және, сонымен бірге кейде үлкен жобада әртүрлі тілде жазылған бағдарламалардың үзінділерін пайдалану қолайлы. Мысалы, өте ертеректегі тілде жазылған (мысалы, Си), дайын мұрагерлік кодты, қайтадан жазбас үшін, пайдалануды қалаймыз, мысалы, C# тілінде, жобаға енгізу жалғыз мақсатымен.

10.3. COM КОМПОНЕНТІК ҚҰРАСТЫРУ ТЕХНОЛОГИЯСЫ

COM компоненттік құрастыру технологиясы стандартты механизмді анықтайды, оның көмегімен бағдарламалық қамсыздандырудың бір бөлігі, іске асыру амалдарына қарамастан, өзінің сервистерін басқасына ұсынады.

COM (Component Object Model, компоеттердің нысанды моделі) - бұл өзара әрекеттесуші клмпоненттердің негізінде бағдарламалық

қамсыздандыру құруға арналған Microsoft компаниясы ұсынған технологиялық, олардың әрқайсысы көптеген бағдарламаларды бір мезетте қолданыла алады.

Стандарт нысанға бағытталған бағдарламалаудың қапшықтануы және полиморфизм идеясін өзіне іске асырады. Windows COM заманауи нұсқаларында кең түрде пайдаланылады. COM негізінде Microsoft OLE Automation, ActiveX, DCOM, COM+, DirectX, XPCOM технологиялары іске асырылған.

COM технологиясында қосымша пайдалануға, ол үшін COM нысандарын қолдана отырып, өз қызметтерін ұсынады. Бір қосымша кем дегенде бір нысанды қамтиды. Әрбір нысанның бір немесе бірнеше интерфейсі бар. Әрбір интерфейс нысандар әдісін біріктіреді, олар ерекшеліктерге (деректерге) қолжетімділікті және операцияның орындалуын қамтамасыз етеді. Әдетте интерфейсте, бір типті операцияларды орындайтын немесе біркелкі ерекшеліктермен жұмыс істейтін барлық әдістер біріктіріледі.

Клиент нысанның қызметтеріне тек интерфейс және оның әдістері арқылы рұқсат алады. Бұл механизм негізгі болып табылады. Клиентке нысан қасиеттерінің және әдістерінің құрамы туралы түпкілікті ақпарат алу үшін бірнеше негізгі интерфейсті білген жеткілікті. Сол себептен, кез келген клиент олардың құрастыру ортасына қарамастан жұмыс жасай алады.

COM сипаттамасына сәйкес, әлдеқашан құрылған интерфейс ешбір жағдайларға қарамастан өзгертілуі мүмкін. Бұл COM негізіндегі қосымшаның, кез келген түрлендіруге қарамастан, тұрақты жұмыс қабілеттілігіне кепілдік береді.

Нысан үнемі COM серверінің құрамында жұмыс жасайды. Сервер динамикалық кітапхана немесе орындалатын файл бола алады. Нысанның өз қасиеттері мен әдістері болуы мүмкін немесе сервердің деректері мен қызметтерін қолдана алады.

Нысанның әдістеріне рұқсат алу үшін, клиент тиісті интерфейске нұсқаушы алуы тиіс. Әрбір интерфейс үшін өзінің жеке интерфейсі болады. Осыдан кейін клиент, жай ғана оның әдістерін шақыра отырып, нысанның қызметтерін пайдалана алады. Нысандардың қасиеттеріне рұқсат тек оның әдістері арқылы жүзеге асырылады.

COM технологиясын пайдаланатын қосымшаның құрастыру кезінде, құрылымдық элементтер қолданылады:

- *интерфейс* COM (COM Interface) - оның көмегімен COM нысаны өзінің сыртқы клиенттерге арналған функционалдық (қызметтің) мүмкіншіліктерін ұсынатын құрал. COM нысаны әдістер мен қасиеттердің әрбір жиынтығын интерфейспен қамтиды. COM кез

келген нысанның бір немесе бірнеше интерфейсі болады;

- COM (COM-server) *сервері* - COM нысанына арналған кодты қамтитын кейбір модуль (exe немесе dll);
- COM (COM-client) *клиенті* - COM нысанының интерфейс(тер)і арқылы серверден қажетті қызметтерді алатын бағдарламалық код. Клиент серверден не алғысы келетіндігін біледі, бірақ оның сауалы сервердің ішінде қалай орындалатындығын білмейді. Көптеген жағдайларда клиент Automation controller (дәл ActiveX-клиент секілді) іске асырылады;
- *типтер кітапханасы* (Type Library) - нысанның, оның интерфейстерінің және әдістерінің COM сипаттамасын, сондай-ақ олардың GUID-идентификаторларын қамтиды. Кітапханадан ақпарат желілік тізілімге жазылады және клиенттік қосымшамен қолданылады;
- *кластар фабрикасы* (Class Factory) - COM нысан құрайтын нысанның данасы. Кластар фабрикасының өзінің нысаны, бірінші интерфейстің клиенттік қосымшасы сұратқан жағдайда, COM сервермен құрылады.

Нысан - COM орталық элементі болып табылады. COM қолдайтын қосымшалардың, өз құрамында бір немесе бірнеше COM нысаны бар. Әрбір нысан тиісті кластың данасын білдіреді және бір немесе бірнеше интерфейсті қамтиды.

Кез келген нысан белгілі бір кластың данасы болып табылады, яғни нысан типтегі айнымалыны білдіреді. Сол себептен нысан осы ерекшеліктермен жұмыс істейтін қасиеттер мен әдістер жиынтығын қамтиды. Нысандарға қолданылатын үш негізгі сипаттамалар тән: қапшықтандыру, мұралану және полиморфизм. COM нысандары барлық осы талаптарды қанағаттандырады.

Нысандарға негізі нысанның интерфейсі түсінігін қолдану тән, жоғарыда анықталғандай, қолданылмайды. Нысанның барлық әдістері оның жалғыз интерфейсін құрайды деп санауға болады, ал интерфейстің нұсқаушысы - нысаның нұсқаушысы болып табылады. COM нысанының кез келген интерфейстер саны бола алады, сонымен бірге әрбір интерфейстің жеке нұсқаушысы бар. Бұл COM нысандарының әдеттегілерінен алғашқы айырмашылығы. COM нысандары тек интерфейсті мұралануды ғана қолдайды, мұнысымен аталық қапшықтануының болжалды бүлінуін болдармайды. Осылайша, COM нысаны нысанға бағытталған бағдарламалаудың көзқарасынан, сөзсіз нысаны болып табылады. Дегенмен, COM технологиясының басты элементі ретінде ол негізгі механизмдерді іске асырудың бірқатар ерекшеліктеріне ие.

Егер COM нысаны COM іске асырудың негізгі элементі болып табылса, онда интерфейстер COM әдістемесінің орталық буыны болып табылады. Интерфейс клиентке COM нысанына дұрыс жүгінуіне, ал нысанға оны клиент «түсінетіндей» мүмкіншілік беретін құрал болып табылады.

Сәйкестендіру үшін әрбір интерфейстің екі атрибуты болады:

1) қолданылатын бағдарламалау тілінің ережелеріне сәйкес символдардан құрылатын оның аты. Әрбір атау «I» символынан басталуы тиіс. Бұл атау бағдарламалық кодта қолданылады;
2) жаһандық бірегеі идентификатор (Globally Unique IDentifier, GUID), ол әлемдік ешбір компьютерде қайталанбайтын, символдардың кепілдендірілген бірегей қиысуын білдіреді. Интерфейстер үшін, осындай идентификатордың атауы IID (Interface Identifier).

Жалпы жағдайда, клиент нысанның қандай интерфейстері бар екендігін білмеуі мүмкін. Олардың тізімін алу үшін, кез келген COM нысанында бар IUnknown негізгі интерфейсі қолданылады.

COM сервері орындалатын файлды білдіреді: бір немесе әртүрлі кластардың бір немесе бірнеше нысандарын қамтуы мүмкін қосымша немесе динамикалық кітапхана. Сервердің үш типі бар:

1) *ішкі сервер* (in-process server) динамикалық кітапханалармен іске асырылады, олар қосымша-клиентке қосылады және онымен бірге бір адрестік кеңістікте жұмыс істейді;
2) *жергілікті сервер* (local server) клиентпен бірге бір компьютерде жұмыс істейтін бөлек процессормен құрылады;
3) *қашықтықтағы сервер* (remote server) клиентке қатысты басқа компьютерде жұмыс істейтін процессормен құрылады.

Негізгі функциялар мен интерфейстердің орындалуын қамтамасыз ету үшін операциялық жүйеде арнайы *COM кітапханасы* бар (нақты іске асыру әртүрлі болуы мүмкін). Кітапхана мүмкіншілктерін рұқсат стандартты жолмен жүзеге асырылады - функцияларды шақыру арқылы.

Қолдайтын COM қосымшаны орнату кезінде желілік тізілімге онымен іске асырылатын барлық COM нысандары туралы ақпарат жазылады:

- клас идентификаторы (Class Identifier, CLSID), ол бір мәнді нысанның класын анықтайды;
- нысан серверінің типі - ішкі, жергілікті немесе қашықтықтағы;
- жергілікті және ішкі серверлер үшін динамикалық кітапхананың немесе орындалатын файлдың толық аты сақталады;
- қашықтықтағы серверлер үшін толық желілілк адресі жазылады.

Кітапхана, қызметтерді басқару диспетчерлерінің көмегімен (Service Control Manager, SCM), желілік тізілімге жүгінеді, класс

идентификаторы бойынша сервер туралы ақпаратты табады және оны іске қосады. Сервер класс данасы - нысанды құрады және кітапханаға сұратылған интерфейске нұсқаушыны қайтарады.

COM кітапханасы клиентке нұсқаушыны табыстайды, ол ақырында тікелей нысанға жүгінуі мүмкін.

Класс данасын іске қосу үшін арнайы нысанды - класс фабрикасы қолданылады. Оның көмегімен бір нысанды, сондай-ақ оның бірнеше данасын құруға болады. Әрбір класс үшін өзінің жеке класс фабрикасы болуы тиіс.

2002 ж. Microsoft . NET плаатформасы ресми түрде шығарылды, оны Microsoft бүгінгі уақытта Windows астына қосымшаларды және компоненттерді құруға ұсынылатын негіз ретінде жарияланған. NET-ке .NET қосымшасынан және керісінше COM компоненттеріне жүгінуге болатын барлық құралдар енгізілген. Microsoft, COM өкілдерінің айтуы бойынша .NET өзара толықтырушы технологиялар болып табылады.

10.4. JAVA ТЕХНОЛОГИЯСЫ

Java технологиясы - бір мезгілде Sun Microsystems компаниясының (кейінірек Oracle компаниясымен сатып алынған) жасап шығарған нысанға бағытталған бағдарламалау тілі және платформасы болып табылады. Java технологиясы виртуалды Java-машинасының (Java virtual machine - JVM) тұжырымдамасына негізделеді.

Виртуалды Java-машинасы - Java Runtime Environment (JRE) деп аталатын Java атқарушы жүйесінің негізгі бөлігі болып табылады. Виртуалды Java-машинасы, алдын-ала Java компиляторымен Java-бағдарламаның бастапқы мәтінінен құрылған Java байт-кодын түсіндіреді.

Java байт-коды - виртуалды Java-машинасымен орындалатын нұсқаулар жиынтығы. Байт-код операциясының әрбір коды - бір байт.

JVM сондай-ақ басқа бағдарламалау тілдерінде жазылған бағдарламаларды орындау үшін де қолданыла алады. Мысалы, Ada тіліндегі бастапқы код Java байт-кодына компиляциялануы мүмкін, ол осыдан кейін JVM көмегімен орындала алады.

JVM - JVM платформасының басты компоненті болып табылады. Виртуалды Java-машиналары көптеген аппараттық және бағдарламалық платформаларға қолжетімді болғандықтан, Java байланыстырушы бағдарламалық қамсыздандыру және жеке платформа ретінде қарастырылаы алады. Бір байт-кодты пайдалану көптеген платформалар үшін Java-ны «бір рет компиляцияланған, барлық жерде іске қосылады»

(compile once, run anywhere) деп сипаттауға мүмкіндік береді. Бағдарламалау тілінің барлық іске асырулары JVM қамтуы тиіс, соның есебінен, Java тілінде жазылған бағдарламалар кез келген операциялық жүйеде жұмыс істейді.

Java бағдарламалау тілі өзгеше болып табылады, себебі бір жағынан, Java бағдарламалар компиляцияланады (Java байт-код аралық тіліне), ал екінші жағынан, олар түсіндіріледі (байт-код бөлшектеледі және JVM шеңберінде орындалады). Компиляция бір рет жасалады, ал түсіндіру бағдарламаны іске қосылған сайын орын алады. Компиляцияланған байт-код JVM арналған оңтайландырылған машиналық код формасы болып табылады; түсіндірушінің өзі JVM іске асыру болып табылады.

Әртүрлі нұсқалардағы бар Java платформасы, JVM-нен және Java қолданбалы бағдарламалық интерфейстен (Java API - Java Application Programming Interface) - апплеттер (applets) мен қосымшалардың құрастырылауын және өрістетілуін жеңілдететін дайын бағдарламалық компоненттердің үлкен жиынтығынан тұрады, кәсіпорын масштабындағы сенімді, қауіпсіз қосымшаларды қоса алғанда. Java API кластар мен интерфейстер кітапханасына топтастырылған; кітапханаларды жиі пакеттер (packages) деп те атайды.

Java тілінде дайын компоненттер ретінде, компоненттердің жаңа бағдарламалық құралдарға кіріктіру ретінде өрістетудің функционалдығы, интерфейсі және үлгісінің сиптаммасын беретін компоненттер ретінде қолданылады. Ол функцияларды өз бетінше орындау үшін әртүрлі орталарда немесе басқа компоненттердің құрамында қайтадан қолданыла алады.

Интерфейс - қайтадан пайдалану элементі ретінде компонентті ортаға кіріктіруге арналған компонент сипаттамасының көзге көрінетін бөлігі. Интерфейстің сипаттамасы жұп ретінде беріледі - параметрдің атауы және параметрдің мәні, олар компонент кодына кедергі келтірместен, автоматты түрде өзгертіле алады. Бұл сипаттау Inspector Components аспабын іске асырады. Ол көзбен шолу кестенің көмегімен интерфейстің қажетті параметрлерін өзгертуге және ұсынылған параметрлердің өгермейтін бөлігін қамтуға мүмкіндік береді, ол сипаттаманың инвариантына енгізілуі мүмкін. Оған компоненттің типі, компоненттің атауы, кіріс және шығыс деректері, компонент әдістерінің атрибуттар мен параметрлер түрлері жатады.

Компоненттердің әртүрлі түрлерін сипаттау және инициализациялау және оларды жаңа жобаға кіріктіру үшін арнайы үлгілер қолданылады. Компонет түрінің функционалдығы бар және Java әдістерінің компоненттерді іске қосуға, жұмыс істеуіне және жоюына арналған стандартты жиынтығын қолдайды.

Java құрастырушыларға арналған таптырмайтын құрал болды және олар үшін келесі мүмкіншіліктерді ашты:

- бір платформада бағдарламалық қамсыздандыру жазу және оны кез келген басқа бағдарламаға іске қосу;
- веб-браузерде жұмыс істейтін және веб-қызметтерге рұқсаты бар бағдарлама құру;
- Интернеттегі форумдарға, дүкендерге, сауалдарға, HTML нысандарына және т.б. арналған сервер жағында қосымша құрастыру;
- жоғары мамандандырылған қосымшалар немесе қызметтер құру үшін Java тіліп пайдалана отырып қосымшалар немесе қызметтерді біріктіру;
- ұялы телефондарға арналған көп функциялық және тиімді қосымшаларды, қашықтықтағы процессорларды, микроконтроллерлерді, сымсыз модульдерді, қадағаларды, шлюздерді, тұтынушылық өнімдерді және электронды құрылғылардың кез келген санаттарын құру.

10.5. .NET FRAMEWORK ПЛАТФОРМАСЫ

.NET амалының тұжырымдамасы

Microsoft .NETFramework - web-сервистер мен қосымшаларды құруға, өрістетуге және іске қосуға арналған платформа. Ол қолданыстағы қосымшаларды келесі буынның қосымшалары мен сервистеріне кіріктіруге мүмкіндік беретін, сонымен қатар Интернет қосымшаларды өрістету және пайдалану міндеттерін шешетін өнімділігі жоғары, стандарттарға негізделген көптілді ортаны білдіреді. .NET Framework сәйкестендірілген кітапханаларды орындаудың жалпы тілдік ортасынан және иерархиялық жиынтығынан тұрады.

Платформа қарапайым және серверлік қосымшаларды қолдайды. Microsoft, Java от Sun Microsystems бәсекелесі ретінде .Net Framework платформасын шығарды. Java сол сәтте ең үздік платформа деп танылған және Microsoft сапасы жоғарырақ платформа жасап шығаруды шешті.

. NET Framework платформасын құрастыру кезінде келесі мақсаттар ескерілді:

- нысанды кодты жергілікті сақтау және орындауға, Интернет

желісінде үлестірілген кодты жергілікті орындау, немесе қашықтықтан орындауға арналған келісілген нысанға бағытталған бағдарламалау ортасын қамтамасыз ету;

- бағдарламалық қамсыздандыруды және нұсқаларды басқаруды өрістету кезінде қақтығыстарды төмендететін, кодты орындау кодын қамтамасыз ету;
- кодты қауіпсіз орындауға кепілдік беретін коды орындау ортасын қамтамасыз ету, беймәлім немесе толық емес сенім білдірілген өндірушімен құрылған кодты қоса алғанда;
- сценарийлерді орындау ортасы немесе түсіндірілетін кодтың өнімділігінің проблемаларын болдырмайтын, кодты орындау ортасын қамтамасыз ету;
- Windows қосымшалары және веб-қосымшалар сияқты қосымшалардың әртүрлі типтеріне арналған жұмыс істеудің бірыңғай қағидаларын қамтамасыз ету;
- . NET Framework платформасының кодын кез келген басқа кодпен біріктірілуін қамтамасыз ететін өнеркәсіптік стандарттардың негізінде өзара әрекеттестікті құрастыру.

Платформа бағдарламалық өнімдердің төрт тобынан тұрады.

1. Тілдер жиынтығы, оған C# және Visual Basic .NET кіреді; құрастырудың аспаптық құралдары жиынтығы, соның ішінде Visual Studio .NET; Windows және Интернетте жұмыс істейтін Web-қызметтер мен қосымшаларды құруға арналған кластардың кең кітапханасы; сондай-ақ CLR (Common Language Runtime - орындаудың жалпы тілдік ортасы) бағдарламасының ортасы, онда осы платформада құрылған нысандар орындалады.

2. Реляциялық дерекқорларға жүгіну, электронды поштаны қолдану және т.б. арналған мамандандырылған функционалдық мүмкіншіліктерді ұсынатын .NET Enterprise Servers серверлер жиынтығы.

3. .Net MyServices деп аталытын коммерциялық веб-қызметтердің бай таңдауы. Мардымсыз ақыға құрастырушылар, тұлғаның жеке басын сәйкестендіруді және басқа деректерді талап ететін қосымшаларды құрастыру кезінде осы қызметтерді пайдалана алады.

4. .NET құралдарын қолдайтын жаңа компьютерлік емес қосымшалар - ұялы телефондардан бастап ойын приставкаларына дейін.

Microsoft .NET тек қана тілдік тәуелсіздікті ғана емес, сонымен бірге тілдік кіріктіруді де қолдайды. Бұл құрастырушының бірнеше тілдермен бір мезгілде жұмыс істеу кезінде, кластардан мұралануын, ерекшеліктерді өңдеуге және полиморфизм артықшылықтарын пайдалана алады дегенді білдіреді. .NET Framework платформасы CTS

(Common Type System - типтердің жалпы жүйесі) сипаттамасының көмегімен осындай мүмкіншілікті береді, ол орындау ортасымен қолданатын деректердің барлық типтерін толығымен сипаттайды, деректердің бір типтері басқалармен әрекеттесе алатындығын және олардың .NET форматында қалай ұсынылатындығын анықтайды. Мысалы, .NET-те кез келген болмыс белгілі бір кластың нысаны, System.Object түбірлі кластың туындысы болып табылады.

Барлық .NET бағдарламалау тілдерінде CTS анықталған деректердің барлық типтері міндетті түрде қолданылуы тиіс емес екендігін түсіну маңызды. CLS (Common Language Specification - жалпы тілдік сипаттамасы) сипаттамасы барлық тілдер ұстануы тиіс заңдарды анықтайтын негізгі ережелерді орнатады: басты сөздер, түрлер, қарабайыр типтер, әдістерді артық жүктеу және т.с.с. CLS сипаттамасы .NET платформасының тіліне ұсынылатын минималды талаптарды анықтайды. Осындай сипаттаманы қанағаттандыратын компиляторлар, бір-бірімен өзара әрекеттесуге қабілеті бар нысандарды құрады. CLS талаптарына сәйкес келетін кез келген тіл, FCL (Framework Class Library - платформа кластарының кітапханасы) кітапханасының барлық мүмкіншіліктерін пайдалана алады. CLS бағдарлама қамсыздандырусын құрастырушыларына да, жеткізушілерге де, өндірушілерге де тілдерге арналған ережелердің жалпы жиынтығының, компиляторлардың және дерек типтерінің шегінен шықпауға мүмкіндік береді.

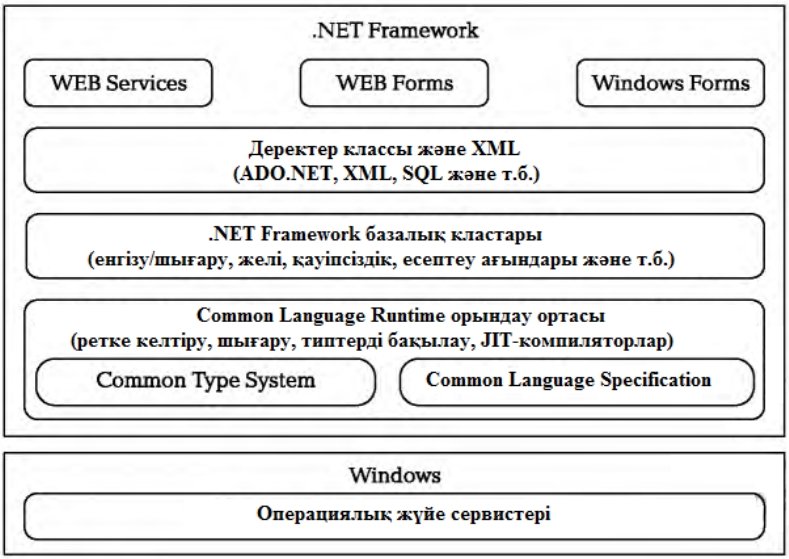
.NET Framework платформасы операциялық жүйенің үстіңгі құрылысы болып табылады, ол ретінде кез келген Windows нұсқасы бола алады. Бүгінгі күні .NET Framework платформасының құрамына кіретіндер:

- Төрт ресми тілдер: C#, VB.NET, Managed C++ (C++ басқарылатын) және JScript .NET;
- CLR (Common Language Runtime) нысанға бағытталған ортасы, Windows және Интернетке арналған қосымшаларды құру үшін бірігіп қолданылады;
- FCL (Framework Class Library) жалпы атауымен кластардың өзара байланыстырылған кітапханалар қатары.

.NET Framework платформасының сәулеттік компоненттерінің қатынасы тұжырымдамалық көзқарастан 10.1-суретте берілген.

CLR деңгейінің үстінде платформалардың негізгі кластырының жиынтығы бар, оның үстінде деректер мен XML кластарының қабаты, сондай-ақ веб-қызметтер (Web Services), Web- және Windows-қосымшалар (Web Forms және Windows Forms) құруға арналған кластар қабаты орналасқан. Бір жерге жиналған,бұл кластар FCL (Framework Class Library) - кластар кітапханасы атты жалпы атауымен танымал. Ол

ертеректе мүмкін болған тек API Windows арқылы, сондай-ақ веб-құрастырма (ASP.NET) арналған қолданбалы функцияларға, қолжетімді болған (ADO.NET) деректеріне рұқсатты, қауіпсіздікті қамтамасыз ету және қашықтықтан басқаруды қоса алғанда желілік функцияларға рұқсатты ашады.



10.1-сурет. Microsoft .NET Framework негізгі компоненттері

Платформаның негізгі кластар жиынтығы, файлдық енгізу/шығару, графиканы өңдеу және компьютер жабдығымен әрекеттесу секілді әдеттегі төменгі деңгейлі операцияларды жасырып қана қоймай, сонымен бірге заманауи қосымшаларда қолданылатын қызметтердің көп мөлшерін (қауіпсіздікті басқару, желілік байланысты сүйемелдеу, есептеу ағындарын басқару, бейнелеулермен және топтамалармен жұмыс істеу және т.б.) сүйемелдеумен қамтамасыз етеді.

Осы деңгейдің үстінде, деректерді және XML басқаруды қамтамасыз ету мақстымен, негізгі кластарды кеңейтетін кластар деңгейі бар. Деректер класы серверлік дерекқорда сақталатын ақпаратты басқаруды іске асыруға мүмкіндік береді. Осы кластардың құрамына SQL (Structured Query Language, құрылымдалған сауалдар тілі) кластары кіреді, олар бағдарламашыларға SQL стандартты интерфейсі арқылы ұзақ мерзімді сақтау орындарына жүгінуге мүмкіндік береді. Мұнан

басқа, ADO.NET деп аталатын кластар жиынтығы, тұрақты деректерді пайдалануға мүмкіндік береді. .NET Framework платформасы сондай-ақ, XML-деректермен күрделі әрекет жасауға және XML түрлендіру және іздеуді орындауға мүмкіндік беретін бірқатар кластарды қолдайды.

Web Forms және Windows Forms аспаптық құралдары Web- және Windows- қосымшаларды құру үшін RAD технологиясын қолдануға мүмкіндік береді.

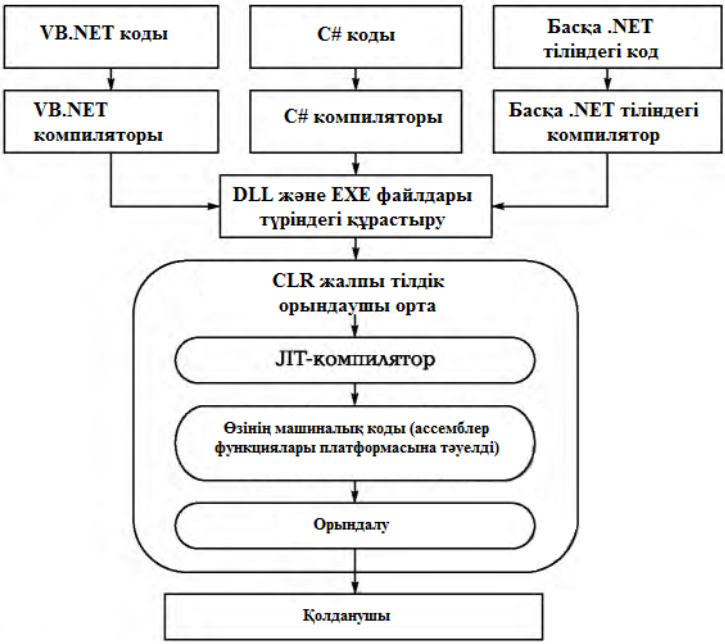
.NET Framework құрастыру кезіндегі негізгі идеясы құрылғылардың әртүрлі типтерінде және әртүрлі орталарда орындауға қабілетті, әртүрлі типті қосымшаларды құру мүмкіндігін беру арқылы құрастырушының еркіндігін қамтамасыз ету болатын.

.NET орындау ортасы

.NET Framework платформасының ең маңызды компоненті, бағдарлама орындалатын ортаны таныстыратын CLR (Common Language Runtime) болып табылады. Оның басты рөлі .NET типтерін анықтау және жүктеуді және алынған командаларға сәйкес оларды басқаруды жүзеге асыру. CLR құрамына, көбінесе виртуалды Java-машинасына ұқсас виртуалды машина кіреді. Жоғарғы деңгейде, орта нысандарды іске қосады, қауіпсіздігін тексереді, жадыға нысандарды орналастырады, оларды орындайды, сондай-ақ қоқыс жинағышты іске қосады (қоқыс жинау дегеніміз - пайдасыз және қосымшаның кейінгі жұмысында қолданылмайтын нысандарымен орын алынған жадыны босату түсіндіріледі).

.NET-қосымшалар, дәстүрлі Windows-қосымшаларға қарағанда басқаша орынлдалады. Мұндай бағдарламалар екі сатымен компиляцияланады (10.2-сурет).

Бірінші кезеңінде бастапқы код жобаны құру кезінде компиляцияланады және машиналық коды бар орындалатын файлдардың орнына бірінші құрастырма пайда болады (assembly), оның құрамына MSIL (Microsoft Intermediate Language - Microsoft аралық тілі) аралық тілінің командалары кіреді. IL коды дискідегі файлда сақталады. Сонымен бірге, компилятормен түрлендірілетін MSIL (қысқаша IL) файлдары, мысалы C#, басқа .NET тілдерінің компиляторларымен түрлендірілетін IL-файлдарға ұқсас. Осы мағынады, платформа тілге қатысты бейтаныс болып қалады. CLR ортасының ең маңызды сипаттамасы оның үлкендігінде болып табылады; бір орта C# жазылған бағдарламалармен бірге, VB.NET тіліндегі бағдарламаларды да орындайды.



10.2-сурет. Компиляцияның екі кезеңді процесі

Компиляцияның екінші кезеңі бетті тікелей нақты орындаудың алдында орын алады. Осы кезеңде CLR, процессормен орындалатын аралық IL кодын төменгі деңгейлі өзіндік машиналық кодта трансляциялайды. Процесс келесі түрде жүргізіледі: .NET-бағдарламаны орындау барысында CLR жүйелері JIT-компиляторды белсендіреді, ол кейінірек MSIL-ді процессордың ішкі кодына айналдырады. Бұл кезең «нақты қажетті сәтте» (Just-In-Time) жедел компиляциясы немесе JIT-компиляция түрінде танымал, және ол барлық .NET қосымшалар үшін бірдей өтеді (мысалы, Windows қосымшаларын қоса алғанда).

Осылайша, .NET компиляциясы құрастырушыларға қолайлы жағдай жасау мен жинақылық мақсатында екі кезеңге бөлінеді. Төменгі деңгейлі машиналық кодты құрудың алдында компиляторға қосымшаның қандай операциялық жүйеде және қандай негізгі жабдықта жұмыс істейтіндігін білу қажет. Компиляцияның екі кезеңінің арқасында .NET коды бар компиляцияланған құрастыруды құруға және оны бірден артық платформаған үлестіруге болады.

CLR орындау ортасы жадыны, ағындарды басқаруды, кодты орындауды, код қауіпсіздігін тексеруді, компиляцияны және басқа жүйелік қызметтерді басқарады. Бұл құралдар басқарылатын код үшін ішкі болып табылады, ол CLR ортасында орындалады. Орындау ортасына жүгінетін кодты *басқарылатын* код деп атайды, ал жүгінбейтінді - *басқарылмайтын* деп атайды.

CLR орындау ортасы кодқа арналған рұқсатты басқаруды қамтамасыз етеді. Мысалы, пайдаланушылар веб-парақшаға енгізілген орындалатын қосымшаға, оларға жеке деректерге, файлдық жүйеге немесе желіге рұқсат алуға жол берместен, экрандағы аниммацияны жаңғыртуға немесе дыбыс жазуға сенім білдіре алады.

CLR орындау ортасы да, жалпы типтер жүйесі (CTS) деп аталатын қатаң типтендіру және кодты тексеру инфрақұрылымын іске асыра отырып, кодтың сенімділігін қамтамасыз етеді. Microsoft корпорациясының және тәуелсіз өндірушілердің әртүрлі тілдік компиляторлары, жалпы типтер жүйесін қанағаттандыратын, басқарылатын кодты құрады.

Мұнан басқа, орындаудың басқарылатын ортасы бағдарламалық қамсыздандырумен жиі орын алатын проблемаларды алып тастайды. Мысалы, CLR нысандарды, олар пайдаланылмайтын кезде оларды босата отырып, орналастыруды және нысандарға сілтемелерді автоматты түрде басқарады. Жадыны автоматты түрде басқару қосымшалардың екі жиі кездесетін қателерін алып тастайды: жадының кемуі және жадыға жарамсыз сілтемелер.

CLR орындау ортасы да құрастырушылардың өнімділігін артырады. Мысалы, бағдарламашылар бағдарламаны қарастырудың үйреншікті тілінде жазады. Бұл кезде кластар кітапханасын және басқа құрастырушылардың басқа тілдерде жазған компоненттер сияқты орындау ортасының барлық артықшылықтарын пайдаланады. Орындау ортасы болашақ бағдарламалық қамсыздандыруға құрастырылғандығына қарамастан, сондай-ақ, ол ескірген бағдарламалық қамсыздандыруды қолдайды. Басқарылатын және басқарылмайтын кодтардың өзара әрекеттестігі құрастырушыларға COM қажетті компоненттерін және DLL кітапханасын пайдалануға мүмкіндік береді. .NET Framework платформасы орындаудың бірнеше негізгі ортасын беріп қана қоймай, сонымен қатар тәуелсіз өндірушілер орындаудың негізгі құралдарын құрастыруды қолданады.

.NET кластарының кітапханасы

.NET Framework кластарының кітапханасы жүйенің функционалдық

мүмкіншіліктерін қамтамасыз ететін кластар, интерфейстер кітапханасы және мәндер типтерін білдіреді.
Ол .NET Framework қосымшасын, компоненттерін және басқару элементтерін құру негізін құрайды.

Кластар кітапханасы қосымшаны құрастыру үшін қолданылады - командалық қатардан іске қосылатын қарапайым қосымшалардан және пайдаланушының графикалық интерфейсі бар қосымшасынан (GUI) бастап, XML веб-қызметі секілді соңғы технологиялық мүмкіншіліктерді пайдаланатын қосымшалармен аяқтайды.

.NET Framework құрастыру процесін жеңілдететін және оңтайландыратын кластарды, интерфейстерді және мән түрлерін қамтиды, сондай-ақ жүйенің функцияларына рұқсатты қамтамасыз етеді. Тілдер арасындағы өзара әрекеттестікті жаңілдету үшін .NET Framework платформасы түрлерінің көпшлігі CLS-үйлесімді болып келеді және сол себептен оларды кез келген бағдарламалау тілінде пайдалануға болады, оның компиляторы CLS сипаттамасына (циклдың негізгі функциялар жиынтығына) сәйкес келеді.

.NET Framework типтері басқару элементтерін, компоненттерді және қосымшаларды құруға арналған негізді білдіреді. .NET Framework құрамында келесі тапсырмаларға арналған типтер бар:

- деректер мен ерекшеліктердің негізгі типтерін таныстыру;
- құрылымдық деректерді қапшықтандыру;
- енгізу-шығару операциялары;
- жүктелген типтер туралы деректерге рұқсат;
- .NET Framework қауіпсіздік тексеруін шақыру;
- деректерге рұқсат, клиент жағындағы графикалық пайдаланушылық интерфейсті ұсыну және клиент жағындағы серверімен басқарылатын графикалық пайдаланушылық интерфейсін ұсыну.

.NET Framework интерфейстердің, сондай-ақ дерексіз және нақты (дерексіз емес) кластардың кең таңдауын ұсынады. Қолданыстағы нақты кластарды пайдалануға болады, мұнан басқа, көптеген жағдайларда олардың негізінде меншікті туынды кластарды құруға болады.

Интерфейстің мүмкіншіліктерін пайдалану үшін, интерфейсті іске асыратын клас құруға, немесе интерфейсті іске асыратын, .NET Framework кластарының негізіндегі туынды класты құруға болады.

NET платформасын қолдайтын барлық тілдер, барлық кластарға және .NET платформасы кластарының кітапханаларының барлық класты бірдей толық рұқсат бермейтіндігін немесе беруге міндетті еместігін айта кеткен жөн - бұл нақты компилятордың және тілдің іске асырылу ерекшелігіне байланысты.

Көптеген басқа кластар кітапханалыран айырмашылығы, .NET

платформасының кластар стандартты кітапханасы операциялық жүйе функцияларының «үстіңгі бөлігі» немесе белгілі бір API (қолданбалы бағдарламалық интерфейсінің) үстінде болып табылмайды. .NET платформасының кітапханасы .NET Framework платформасының өзінің органикалық бөлігі, оның «туған» API болып табылады. Оны виртуалды .NET машинасының API ретінде қарастыруға болады. .NET Framework платформасының кітапханасы әрбір нұсқасы бойы жаңартылып отырады.

.NET қауіпсіздікті қамтамасыз ету

Бүгінгі уақытта қауіпсіздік - бағдарламалаудың кез келген идеологиясының, технологиясының және аспаптық құралының ең маңызды элементі болып табылады. Бұл үшн .NET бағдарламалық *қамсыздандырудың өмірлік циклын автоматты түрде басқару* сияқты қауіпсіздік шарасы іске асырылған. Бағдарламашы үшін бұл, мысалы, «қоқысты құрастыру» жосығын автоматты түрде іске асыруда білінеді, сондай-ақ белгісіз мәндері («ілінбелі» сілтемелер) бар жады аясына нұсқаушыларды пайдалануға тыйым салу және өзіне көрсететін нұсқауышаларға сілтемемен (циклдық сілтемелер) білінеді.

Қауіпсіздіктің маңызды шектеуі *кодты синтасистік қамтамасыз етуді автоматтандыру* болып табылады. Бұл функциялар мен рәсімдерді қауіпсіз шақыру, бағдарламашымен мәлімделген жадының статистикалық таратылатын аясы өлшемінің шегінен шығуды бақылау арқылы, сонымен қатар, егер онымен әдепкі қалпы бойынша мәні (инициализация) берілмеген болса, айнымалыларды пайдалануға тыйым салу арқылы қол жеткізіледі.

.NET қауіпсіздікті кешенді қамтамасыз етудің маңызды қырының тағы біреуі, типтердің сәйкестігін кеңінен тексерудің іске асырылған стратегиясының шеңберінде жүзеге асырылатын, *аралық кодты типтендіру дұрыстығына міндетті тексеру* болып табылады.

Пайдаланушылардың ресурстарға рұқсат құқығы да елеулі жетілдірелген. Жеке алғанда, компонентті жобаға енгізу үшін, автордың сандық қолтаңбасымен расталған, *кодтың дереккөзін тексеру,* және жөнелтушінің түпнұсқалығына көз жеткізу қажет.

Пайдаланушылардың ресурстарға рұқсатының икемді және сенімді шектеуі, пайдаланушының профиліне сәйкес, динамикалық түзетілетін *рұқсат саясатының* кең ауқымының арқасында жүзеге асырылады.

Қауіпсіздікті қамтамасыз ету үшін, мысалға алсақ, Интернер-арналар арқылы берілетін құпия немес коммерциялық ақпаратты шифрлеу үшін

қажетті *криптографиялық әдістердің* маңызы зор.

БАҚЫЛАУ СҰРАҚТАРЫ МЕН ТАПСЫРМАЛАРЫ

1. Бағдарламаларды құрудың кіріктірілген ортасының анықтамасын тұжырымдаңыз.
2. Кіріктірілген ортаның негізгі компоненттері қандай?
3. Мәтіндік редактор дегеніміз не?
4. Заманауи редакторлардың кіріктірілген ортасына енгізілетін бастапқы кодты синтаксистік тексеру бойынша қандай қосымша функциялар кіріктірілген?
5. Ретке келтіруші дегеніміз не және оның типтік командалары қандай?
6. Бағдарламаларды ұжымдық құрастыруды сүйемелдеу қандай функционалдықты қамтамасыз етеді?
7. Рефакторинг дегеніміз не?
8. Аспаптық бағдарламалық қамсыздандыру неге арналған?
9. Келесі түсініктерге анықтама беріңіздер: компилятор, транслятор, құрастырушы, түсіндруші (интерпретатор).
10. SDK (Software Development Kit) дегеніміз не?
11. БҚ құрастырудың әрбір кіріктірілген ортасына қандай компоненттердің болуы тән?
12. Құрастыруды сүйемелдеу жүйесіне нелер кіреді?
13. Кодты «шатастыру» құралдары не үшін қолданылады?
14. COM технологиясының негізінде не жатыр?
15. COM-нысандар деп нені атаймыз?
16. COM технологияларындағы интерфейстердің рөлі қандай?
17. COM-технологияда кластар фабриаһкасы не үшін қолданылады?
18. COM-нысанның интерфейстері болмауы мүмкін бе?
19. COM-технологияларда типтер кітапханасы не үшін қолданылады?
20. Қолдайтын COM қосымша орнату кезінде жүйелік тізілімге қандай ақпарат жазылады?
21. Виртуалды Java-машина дегеніміз не?
22. Құрастырушылар үшін Java қандай мүмкіншіліктер береді?
23. .NET Framework кластар кітапханасы дегеніміз не?
24. CLR орындау ортасының ерекшеліктері қандай?
25. Байт-код дегеніміз не?
26. .NET платформасының тұжырымдамасы қандай?
27. .NET платформасының артықшылықтары қандай?

11-тарау

CASE-ҚҰРАЛДАР

11.1. CASE-ҚҰРАЛДАР ТУРАЛЫ ЖАЛПЫ МӘЛІМЕТТЕР

Жобаны қаржылық және уақыт ресурстарының шектеулері кезінде, құрастырылатын бағдарламалық қамсыздандырудың үлкен өлшемдері мен жоғары күрделілігі ақырғы бағдарламалық өнмнің және тұтастай жүйенің төменгі сапасына алып келуі мүмкін. Осыған байланысты соңғы уақытта, бағдарламалық қамсыздандырудың тіршілік циклы процестерінің автоматтандыруын қамтамасыз ететін аспаптық құралдарға (CASE-құралдарға) және заманауи технологияларға назар көбірек аударылады. Осындай аспаптық құралдарды пайдалану, құрастыру процесінің сапасын бір мезгілде арттыра отырып, жүйені құрастырудың ұзақтығын және құнын, және осының салдарынан құрастырылған бағдарламалық құралдардың сапасын едәуір қысқартуға мүмкіндік береді.

CASE (Computer Aided Software/System Engineering) - бағдарламалардың жоғары сапасын, қателердің болмауын және бағдарламалық өнімдерге қызмет көрсетудің қарапайымдылығын қамтамасыз етуге көмектесетін, бағдарламалық қамсыздандыруды жобалауға арналған бағдарламалық инженерияның құралдары мен әдістерінің жиынтығы. Сонымен қатар CASE дегенде, CASE-құралдарын пайдалан отырып, ақпараттық жүйелерді жобалау әдістері мен құраладарының жиынтығын түсінеді.

Бағдарланы құрастыруды автоматтандыру құралдары (CASE-құралдар) - БҚ құрастырушының жүйелік талдаушысы үшін бағдарламалық қамсыздандыруды жобалау және құрастыру процестерін автоматтандыру құралдары.

Бастапқыда CASE термині Computer Aided Software Engineering (бағдарламалық қамсыздандыруды жобалауды компьютерлік сүйемелдеу) ретінде түсіндірілген. Алайда ISO/IEC 14102 стандартының келуімен CASE-құралдар БҚ өмірлік циклының процесін сүйемелдеуге арналған бағдарламалық құралдар ретінде анықталатын болды. Бүгінгі

уақытта аталмыш терминге мұнан кең мағына беріледі, және ол Computer Aided System Engineering (жүйелерді жобалаудың компьютерлік сүйемелдеуі) ретінде шифрден ашылады. Заманауи CASE-құралдары кең тағайындалудағы күрделі жүйелердің мәндік аясын модельдеу, сипаттамаларын дайындау, жобалауға бағдарланады.

CASE-технология - бұл, өзара байланысқан автоматтандыру құралдарымен сүйемелденетін бағдарламалық құралдар мен күрделі жүйелерді құрастырудың және ілестірудің әдістемелер жиынтығы.

Бағдарламалық қамсыздандыруды дайындау кезінде CASE-технологияны пайдаланудың негізгі мақсаттары - құрастырушыға көзбен талдау және жобалаудың тиісті әдістемесін ұсына отырып, бағдарламалаудан және құрастыру үрдісінің кейінгі жұмыстарын талдау және жобалауын бөліп көрсету.

CASE-құралдары талдау және талаптарды қалыптастыру, қосымшалар және деректер қорын жобалау, кодты генерациялау, тестілеу, сапаны қамтамасыз ету, конфигурацияны және жобаны басқару сияқты ақпараттық қамсыздандырудың толық өмірлік циклын сүйемелдеу және құру үрдістерін қолдайтын бағдарламалық құралдар.

CASE-жүйені белгілі бір функционалдық тағайындамасы бар және бірыңғай бағдарламалық өнім шеңберінде орындалған CASE-құралдардың жиынтығы деп анықтауға болады.

Заманауи CASE-құралдары, қарапайым талдау құралдарынан және құжаттамалаудан бастап автоматтандырудың толық масштабты құралдарына дейінгі ақпараттық жүйелерді жобалаудың көптеген технологияларын қолдаудың кең ауқымын қамтиды. Олардың пайдалануы, тек қана жұмысты жылдамдатып және оларды пайдаланудың сапасын жоғарылатап қана қоймай, сонымен қатар бағдарламашылар тобының ұжымдық жұмысын ұйымдастыруға арналған құралдар береді.

11.2. CASE-ҚҰРАЛДАРМЕН ҚҰРУ қағидалары және ЖҰМЫС ТӘСІЛДЕРІ

Нақты түрде CASE-құралы - бұл, бағдарламалық қамсыздандырудың өмірлік циклының процестерін немесе процестерінің жекелеген кезеңдерін сүйемелдейтін графикалық бағдарланған аспаптық құралдарының жиынтығы.

CASE-құралдарға БҚ құрастыру кезінде, келесі негізін қалаушы қағидаларда негізделетін, автоматтандырылған көмекті, оларды ілестіруді немесе жобаны басқаруды қамтамасыз ететін кез келген

бағдарламалық құралды жатқызуға болады:

- құрастырушымен қолайлы интерфейсті қамтамасыз ететін және оның шығармашылық мүмкіншіліктерін қолданатын жүйені сипаттау және құжаттауға арналған қуатты графикалық құралдарының болуы;
- бағдарламалық қамсыздандыруды құрастыру процесін басқаруды қамтамасыз ететін, CASE-құралдардың жекелеген компоненттерін интеграциялау;
- жобалық метамәліметтердің - репозиторийдің, ұйымдастырған қоймасын арнайы қолдану.

Жоғарыда аталып өткен қағидалардан басқа, CASE-құралдарды тұжырымдамалық құрастырудың негізінде келесі ережелер жатыр:

- бағдарламалық қамсыздандырудың өмірлік циклын жеңіл, ыңғайлы және үнемді формаға алып келуге мүмкіндік беретін адами факторды ескеру;
- басқа қосымшаларды қолданылатын негізгі бағдарламалық құралдарды пайдалану (ДББҚ, әртүрлі бағдарламалау тілінің компиляторлары, ретке келтірушілер және басқалар);
- автоматтандырылған немесе автоматты кодты генерация;
- әзірленетін бағдарламалық қамсыздандыру компоненттерінің немесе жүйесінің күрделілігін түсінуге, пайдалануға және түрлендіруге қолжетімді, қолдауға мүмкіндік беретін күрделілікті шектеу;
- пайдаланушылардың әрқандай санаттарына, соның ішінде тапсырыс берушілердің, мәндік сала мамандарының, жобалаушылардың, бағдарламашылардың, тестілеушілердің, сапа инженерлерінің, жоба менеджерлерінің қолжетімділігі;
- CASE-құралдарды сатып алуға жұмсалған ақшалай қаражаттардың жылдам өтемділігін, жоба мерзімі мен құнын қысқарту арқылы қамтамасыз ететін рентабельділік;
- сүйемелденілуі. CASE-құралдар жобаның өзгермелі талаптарына және мақсаттарына бейімделу қабілеті бар. CASE-технологиялар бағдарламалық өнімді әзірлеу және сүйемелдеу процестерін барынша автоматтандыруға арналған. Олар қабылданатын техникалық шешімдердің сапасын және жобалық құжаттаманың дайындалуын қамтамасыз етеді. Графикалық құралдар әзірлеушілерге көрнекі түрде пән саласын модельдеуге, тиісті ақпараттық жүйені зерттеуге, оны қойылған мақсаттарға және қолданыстағы шектеулерге сәйкес өзгертуге мүмкіндік береді. Ақпаратты көзбен шолуды ұсыну әдістері өте маңызды рөл атқарады және пайдаланушының ақпараттық мұқтаждығына сәйкес таңдап алынады.

11.3. CASE-ҚҰРАЛДАРДЫҢ НЕГІЗГІ ФУНКЦИОНАЛДЫҚ МҮМКІНШІЛІКТЕРІ

CASE-құралдардың негізгі компоненттері және олардың функциялары

CASE-құралдары, бағдарламалық қамсыздандырудың өмірлік циклының әртүрлі негізгі, көмекші және ұйымдастырушылық процестерінің кезеңдерін қолдайтын әртүрлі функционалдық тағайындалудың аспаптарын қамтиды. CASE-құралдардың құрамына төрт негізгі компонент кіреді.

1. *Жоба туралы барлық ақпаратты орталықтандыра сақтау құралдары - репозиторий.* Әзірленетін бағдарламалық қамсыздандыру немесе жүйе туралы ақпаратты әзірлеудің бүкіл өмірлік циклының ішінде сақтауға арналған.

2. *Енгізу құралдары.* Репозиторийге, жоба қатысушылары CASE-құралымен өзара әрекеттесу ұйымдарына мәліметтерді енгізуге арналған. Талдау, жобалау, тестілеу, бақылаудың әртүрлі әдіснамаларын қолдау тиіс. Бағдарламалық қамсыздандырудың немесе жүйенің өмірлік циклы бойы жоба қатысушылырына (жүйелі талдаушыларға, жобалаушыларға, бағдарламашыларға, тестілеушілерге, менеджерлерге, сапа бойынша мамандарына және т.б.) әртүрлі санаттарымен қолдануға арналған.

3. *Талдау және әзірлеу құралдары.* Әзірлеу барысында графикалық және мәтіндік сипаттамалардың әртүрлі түрлеріне және олардың түрлендірулерін талдауға арналған.

4. *Шығару құралдары.* Кодты генерациялау, әртүрлі құжат түрлерін жасау, жобаны басқару үшін қызмет етеді.

CASE-құралдардың барлық компоненттері жиынтық түрінде келесі функционалдық мүмкіншіліктері бар:

- графикалық модельдерді сүйемелдеу;
- қателерді бақылау;
- репозиторийді қолдау;
- БҚ ӨЦ негізгі, көмекші және ұйымастырушылық процестерін қолдау.

CASE-құралдардағы графикалық модельдерді қолдау. Бағдарламалық қамсыздандыруды әзірлеудің ең бейнетті кезеңдері талаптарды қалыптастыру және жобалау кезеңдері болып табылады, олардың барысында CASE-құралдар қабылданатын техникалық

шешімдердің сапасын және жобалық құжаттаманы дайындауды қамтамасыз етеді. Сонымен бірге ақпаратты көзбен шолып таныстыру әдістері үлкен рөлін ойнайды. Талдау және жобалаудың графикалық құралдары, БҚ жүйесінің модельдерін құратын өзара байланысқан диаграммаларды құруды және редакциялауды қамтамасыз етеді. Нысандар тобына құру, ауыстыру және түзету, олардың өлшемдерін өзгерту, масштабтау, оларды ауыстыру және өлшемдерін сақтау кезінде нысандар арасындағы байланысты сақтау, қателерді автоматты түрде бақылау және басқа мүмкіншіліктері бар.

Пәндік саласын модельдеудің графикалық құралдары, берілген мақсаттар мен қолданыстағы шектеулерге сәйкес оны өзгерту әзірлеушілерге көрнекі түрде қолданыстағы ақпараттық жүйені зерттеуге мүмкіндік береді. Жобалаудың әртүрлі деңгейлерінде бағдарламалық қамсыздандырудың графикалық ұсынысының әртүрлі түрлерін және нотацияларын қолдана алады. Әдетте әртүрлі түрдегі диаграммалар қолданылады, соның ішінде талаптар иерархиялары. Диаграммаларды әзірлеу арнайы графикалық редакторлардың көмегімен жүзеге асырылады, олардың негізгі функциялары иерархиялық байланысқан бағдарламаларды, олардың нысандарын және нысандардың арасындағы байланысын құру және редакциялау, сондай-ақ қателерді автоматты түрде бақылау болып табылады.

CASE-құралдардағы қателерді бақылау. Талаптар мен жобалауды қалыптастыру кезеңдерінде қателерді бақылау өте маңызды. Бұл соңғы кезеңдерде оларды анықтау және жою анағұрлым қымбат болатындығымен байланысты. CASE-технология, әзірлеудің ерте кезеңдерінде, жобаның толықтығына және қисындылығына бақылуды және автоматты тексеруін қамтамасыз етеді, бұл әзірлеудің жалпы алғандағы сәттілігіне әсер етеді. CASE-құралдарда әдетте бақылаудың келесі түрлері іске асырылады:

- диаграммалардың синтаксисін және олардың элементтерінің түрлерін бақылау;
- диаграммалардың толықтығын және қисындылығын бақылау (диаграммалардың барлық элементтері сәйкестендірілуі және репозиторийларда көрсетілуі тиіс);
- бір немесе әртүрлі типтегі диаграммаларды деңгейлер бойынша олардың қисындылығына өтпелі бақылау - *диаграммаларды тігінен және көлденең теңгеру.* Бір типті диаграмманы тігінен теңгеру талдап тексерілетін және таңдап тексеріліп жатқан диаграммалардың арасындағы деректер ағындарының сәйкестіктерін айтады. Көлденең теңгеру деректер құрамы мен процестер сипаттамасының арасындағы сәйкессіздікті анықтайды;

- функция декомпозициясын бақылау. Бақылаудың аталмыш типінде әртүрлі метрикалар негізінде декомпозицияны бағалау жүргізіледі. Мысалы, модульдердің байланыстылығы мен тіркесу тарапынан декомпозицияның тиімділігі мен дұрыстығы бағалануы мүмкін.

CASE-құралдардағы репозиторийді қолдау. Репозиторий жоба нұсқаларын және оның жекелеме компоненттерін сақтауды, топтық әзірлеуде әртүрлі әзірленушіден түскен ақпаратты синхронизациялануды, метадеректерді толықтығына және қарама-қайшы еместігіне бақылауды қамтамасыз етеді. Репозиторийді құрамы тек қана әртүрлі ақпараттық нысандардың түрлерін ғана емес, сонымен бірге осы компоненттерді пайдалану немесе өңдеу ережелерін де қамтиды.

Репозиторий диаграммаларды, экрандар мен мәзір анықтамаларын, есептер жобаларын, деректердің сипаттарын, алғашқы кодтарды және т.б. сақтай алады. Репозиторийдегі әрбір ақпараттық нысан оның қасиеттерін (сәйкестендіргіш, синоним, тип, мәтіндік сипаттау, компоненттер, мәндер аясы және т.с.с.) атап өтумен сипатталады. Бұдан басқа, онда нысанды қалыптастыру және редакциялау ережелері, сондай-ақ нысанды құру уақыты, оның соңғы жаңару уақыты туралы бақылау ақпараты, нұсқа нөмірлері, жаңарту мүмкіндіктері және т.с.с. сақталады.

Репозиторий жоба бойынша құжаттаманы стандарттау және жобалық сипаттамаларды бақылауға арналған негіз болып табылады. Барлық есептер репозиторий құрамы бойынша автоматты түрде жасалынады. CASE-технологияның күмәнсіз құндылығы, құжаттама ісінің ағымдағы жай-күйін көрсетуінде, себебі жобадағы кез келген өзгерістер автоматты түрде репозиторийде көрсетіледі.

Репозиторий арқылы қауіпсіздік бақылауы (рұқсатты, рұқсат артықшылықтарын шектеу), нұсқа бақылауы, өзгерістер бақылауы және басқалар жүзеге асырыла алады.

Репозиторий жоба бойынша құжаттамалардың автоматты генерациясына арналған негіз болып табылады. Есептердің негізгі типтері болып табылатындар:

- *құрамы бойынша есептер* - деректер ағыны мен олардың компоненттері бойынша ақпаратты қамтиды; диаграммалардың функционалдық блоктарының және олардың кіріс және шығыс ағындарының тізімі; барлық ақпараттық нысандар мен олардың атрибуттарының тізімі; нысандарды өзгерту тарихын; олардың арасындағы модульдер мен интерфейстердің сипаттары; модельдерді тестілеу жоспарлары және т.с.с.;
- *тоғыспалы сілтемелер бойынша есептер* - барлық шақыратын және

шақыртылатын модульдердің байланыстары бойынша ақпаратты қамтиды; жобаның нақты орындаушысының рұқсаты бар, репозиторий нысандарының тізімі; диаграммалар мен нақты деректер арасындағы байланыс бойынша ақпаратты; деректердің кірістен шығысқа қозғалу бағдары;

- *талдау нәтижелері жөніндегі есептер* - диаграммалардың, белгісіз ақпараттық нысандар тізімі, толық емес диаграммалар тізімі, жоба құрылымын талдау нәтижелері бойынша деректердің және т.с.с. өзара дұрыстығы бойынша деректерді қамтиды;
- *нысандарды декомпозициялау жөніндегі есептер* - әрбір нысанның құрамына кіретін нысандардың жиынтығын, сондай-ақ құрамына әрбір нысан кіретін нысандарды қамтиды.

Бағдарламалық құралдар және жүйелердің өмірлік циклының процестерін қолдау. Әзірлеу процесін қолдаудың негізі ретінде төменде берілген заманауи CASE-құралдардың қасиеттері болып табылады.

1. *Жүйенің немесе бағдарламалық құралдардың барлық өмірлік циклын өтеу.* Заманауи CASE-құралдар бағдарламалық қамсыздандырудың толық өмірлік циклын қолдайды. Дегенмен, алдыңғы қатардағы назар әзірлеу процесінің бастапқы жұмыстарына аударылады - жүйеге қойылатын талаптар талдауына, жүйелік сәулетті жобалауға, бағдарламалық құралдарға қойылатын талаптарды талдау және бағдарламалық сәулетті жобалауға. БҚ қойылатын талаптарды сауатты әзірлеу тұтастай жобаның негізі болып табылады, олардың толықтығы және дұрыстығы тапсырыс берушінің талаптарына әзірлеу нәтижелерінің сәйкестік деңгейін анықтайды.

2. *Прототиптендіруді қолдау.* Күрделі немесе сыни бағдарламалық қамсыздандыруды әзірлеу үшін арналған өмірлік циклдың көптеген үлгілері прототиптендіру қолдануға негізделеді. Прототиптерді әзірлеу өмірлік циклдың ерте кезеңдерінде жүзеге асырылады және бағдарламалық қамсыздандыруға қойылатын талаптарды нақтылауға, сондай-ақ дайындалатын өнімнің жай-күйін болжалдауға мүмкіндік береді.

3. *Бағдарламалық қамсыздандыруды әзірлеудің заманауи әдістемесін қолдау.* CASE-құралдардың заманауи қатары, әдеттегідей, әзірлеу процесінің әртүрлі кезеңінде пайдалану үшін арналған әртүрлі әдістемелерді қолдайды. Сонымен бірге, әртүрлі диаграмма түрлерін құруды графикалық қолдау, жобалау қадамдарының дұрыстығын және құжаттама дайындауды бақылау орындалады.

4. *Автоматты түрде кодты генерациялау.* Кодты генерация бастапқы кодтардың 90% дейін автоматты түрде жоғары деңгейлі

тілдерде құруға мүмкіндік береді.

Әртүрлі CASE-құралдармен бағдарламалаудың барлық танымал тілдері қолданады.

CASE-құралдарды пайдалану құрылатын жобалардың сапасын жақсартуға мүмкіндік береді, ал ірі корпоративті жүйелер құру кезінде тіпті шарасыз болып табылады. Дегенмен, CASE-құралдарды пайдалану, бағдарламашыны логикалық жобалаудың тек қана жалпы мағынасын түсінуден ғана емес, сонымен қатар оның егжей-тегжейін түсінуден босатпайды.

Кодты генерациялау құралдары

Кодты генерациялау үшін пәндік саланың метадеректері, үлгілері және ережелері анықталуы тиіс. Пәндік сала ережелері үлгілердің, метадеректердің қандай болатындығын анықтайды. Генератор болса, метадеректер мен үлгілердің негізінде автоматы генерацияланған кодты тұрғызады.

Кодты генерациялау - бұл арнайы құралдармен (код генераторларымен), бағдарлама кодын автоматты немесе автоматтандырып жасау болып табылады, ол кезінде берілген шарттар бойынша бағдарламаның бастапқы коды толығымен немесе ішінара қалыптасады.

Кодты генерация құралдарын екі түрге бөлуге болады:

1) өнімнің басқарушы құрылымын генерациялау құралдары; аталмыш құралдар бағдарламалық құралдың логикалық құрылымын, дерекқорға арналған кодтарды, экрандарды, есептерді автоматты құруды орындайды. Бағдарламалық құралдың қалған үзінділері қолмен кодталанады;

2) толық өнімнің генерациялау құралдары; аталмыш құралдар әзірленген сипаттамалар немесе модельдердің негізінде бағдарламалық құралдың, оның пайдаланушылық және бағдарламалық құжаттамасының толық кодтарын генерациялауға мүмкіндік береді.

Генераторлар көптеген қысқа, ұқсас итерацияларды орындауды талап ететін проблемалық салада жақсы қолданылады, мысалы, дерекқорлар, мұнда жобалаудың бірдей үлгісі көп рет көптеген дерекқордың кестелеріне қолданылады. Әрбір кестенің құрылымы бірегей бола алады, бірақ кодты әзірлеу кезіндегі кестелерге кестенің жолына қойылатын шарттар мен ережелер бірдей.

Іс жүзінде ең қолданбалы салалары болып табылатындар:

- дерекқордың қосымшалары (логикасы жоқ нысандар, транзакцияларды қолдау);
- SQL-сауалдар;

- дерекқорлармен әрекеттесетін, оларды дерекқорларды енгізу-шығаруға пайдаланатын қосымшалар;
- бағдарламалаудың нысанға бағытталған тілдерінің кластары және құрылымы;
- бағдарламалық кодты тестілеу;
- бағдарламаның техникалық құжаттамасы.

Жоғарыда сипатталған жағдайлардан басқа, кодты генерациялау скриптерді, конфигурациялық, командалық файлдарды, макростарды және т.б. автоматты түрде құру үшін қолданылуы мүмкін.

Бағдарламалық жобаны әзірлеу кезіндегі автоматты түрде кодты генерациялауды қолданудың бірқатар артықшылықтары бар.

1. Машинамен генерацияланған *кодтың құрылымдылығы және кодтың келісімділігі*. Генератор символға дейінгі дәлдікпен кодты жазу ережелерін сақтауға мүмкіндік береді.

2. *Қосымшаның сәулеті деңгейіндегі келісімділік.* Генераторды қолдану құрылатын кодтың қосымшалардың бастапқыда жобаланған құрылымына және үлгілеріне сәйкес келеді деген сенімділік береді. Егер қайсыбір функционалдылықты генератордың көмегімен іске асыру мүмкін болмай жатса, бұл оның бастапқы сәулетіне сәйкес келмейді деген сигнал болып табылады.

3. *Кодтың жоғары сапасы* және *қателерді жеңіл жөндеу.* Бағдарлама қателіктерден бос және тұрақты.

4. *Өзгерістерге икемділігі.* Егер талаптар өзгеретін болса, тек үлгілерді жаңартып, кодтың жаңа нұсқасын шығару қажет. Басқа технологиялар мен платформаларға жеңіл және қиналмай өту мүмкіншілігі бар. Адами фактордың салдарынан пайда болатын қателерді мейлінше қысқартуға келеді.

5. *Әзірлеу жылдамдығы*. Егер метадеректер әлдеқашан жіберілсе және енгізілсе, онда үлкен көлемді кодты генерациялау нанғысыз қысқа мерзімде жүргізілу мүмкін.

6. *Қосымша туралы бірыңғай ақпарат көзі.* Қолмен жазылатын қосымшаларда, кестенің немесе баған атауының қарапайым өзгеруі кодтың көптеген жерлерінде өзгерістер топтамасын тартуы мүмкін. Генерациялау кезінде метадеректер бөлек дереккөзде болады, сол себептен сызба анықталатын жердегі кесте немесе бағанның атауын өзгерткен және кодты қайта құру жеткілікті.

7. *Бизнес-қисын көрнекілігі.* Қолмен жазылатын қосымшаларда, бизнес қисынның мағынасы бағдарламалық кодтың жүздеген және мыңдаған қатарларында жоғалып кетеді. Генераторлар метадеректер файлдарын пайдаланады, оларда бағдарламалатын пәндік саланың барлық құрылымы, оның жай-күйі, ерекшеліктері және т.б. көрінеді.

Қосымша дерексіздіктің жоғары деңгейінде не істейтіндігін түсіну оңай. Метадеректерді қосымшаның пәндік саласындағы сарапшылар пайдалана алады.

8. *Кодты дайындаудың жетілдірілген процесі.* Енді кодты құру процесі аз уақыт алатындықтан, қосымшаны жобалауға көп уақыт бөлу мүмкіндігі пайда болады. Жобаны орындау уақыты азайтылады, жобаны сүйемелдеу қолайлылығы артады. Сонымен қатар, жобалау кезінде жіберілген қате, тек әзірлеу процесінде ғана анықталған жағдайларды болдырмауды талдап және тестілеп отырып, генератордың көмегімен кодтың көптеген прототиптерін жылдам дайындауға болады.

9. *Бағдарламашылардың әзірленудің интеллектуалдық саласына зейін қою мүмкіншілігі,* бұл олардың кәсіби дағдыларын үздік пайдалануға, біліктілігін арттыруға мүмкіндік береді.

10. *Бағдарламашылардың жұмыс қолайлылығын жақсарту.* Дәл қазір қажет нәрсе генерацияланғандықтан, код таза әрі қарапайым бола түседі.

11. *Бағдарламашылардың жұмысқа деген ынтасын және қызығушылығын жақсарту.* Жоғарыда аталып өткендердің барлығы әзірлеушінің жұмысын қызық, маңызды әрі өнімді етеді, бұл ынталандыруға оң әсерін тигізбей қоймайды, ал бұл жобаның сәттілігін әсер ететін өте маңызды фактор болып табылады.

Код генераторын қолдану барлық жағдайларда тиімді бола бермейді. Егер жоба көлемі айтарлықтай үлкен болмаса, егер код генерацияға қиын берілсе, қосымшаны қолмен жазған оңайырақ болып көрінуі мүмкін. Практикада әрдайым қолмен құруды қажет ететін код бөлімі болады. Осындай кодтың көлемі әртүрлі жобаларда әрқилы болуы мүмкін.

Генерация басынан жүзеге асырылуы мүмкін, ол жағдайда код үлгілерден және метадеректерден генерацияланады, алдын-ала құрылған код болмайды. Кейде, белгілі бір жолмен қолданыстағы код өзгереді. Бұл жағдайда бастапқы код үлгі рөлін атқарады, сондай-ақ метадеректерді қамтуы мүмкін.

Метадеректер иерархиялық құрылымы бар арнайы файлдарда (мысалы XML-файлдар), жобалау тілдерінің файлдарында (мысалы, UML), басқа файлдық дереккөздерде, мәтіндік файлдарда, дерекқорда және басқаларда сақталуы (және алынуы) мүмкін.

Генерацияланған код аяқталған және аяқталмаған болуы мүмкін. Аяқталмаған генерацияланған код кейінгі қолмен түрлендіруді қажет етеді. Жиі бұл генератордың жасалып бітпегендігін, шикілігін немесе код генерациялануға нашар берілетіндігін көрсетеді. Аяқталған генерацияланған код түрлендіруді қажет етпейді және оны өндірістік

сызбаға шығаруға болады. Бұл бағдарламалық жобаның қолдан жасалған кодты қамтымайтындығын білдіреді, жай ғана генерацияланған код қолмен жасалған кодпен өзгеріссіз біріктіріледі.

Бағдарламалық кодтың генерациялау масштабы әртүрлі болуы мүмкін:

- белгілі бір модуль немесе файлдар тобы - генератордың жұмысын бастапқы сынамалы тексеру кезінде қолданыла алады;
- қосымша сәулетіндегі деңгей - қосымша сәулетінде тек бір деңгей генерацияланады. Көбісінде бұл дерекқорлар деңгейі, себебі ол генерацияға оңай беріледі және қалған басқа деңгейлердің жұмысы оған байланысты. Дерекқорлардың деңгейінсіз басқа деңгейлердің генерациясы өзінің тиімділігінде көптен айырылады;
- қосымша сәулетіндегі барлық деңгейлер - қосымшаның әрбір деңгейінде генерацияланған код қолданылады. Қосымшаның көп бөлігін генерациялауға мүмкіндік беретін үлгілер әзірленген жағдайда қолданылады.

Кодты генерациялау технологиясы әлеуетті шексіз мүмкіндіктері бар тиімді және қуатты техника болып табылады. Дегенмен, код генераторы, барлық проблемаларды шешетін сиқырлы таяқша болып табылмайтындығын ескерген жөн. Егер жобалау кезеңінде қателіктер жіберілсе, немесе жоба идеясы ең үздік болмаса, онда ешқандай генератор жобаның сәттілігіне кепілдік бере алмайды. Код генераторы, басқа аспаптық құралдар секілді, - бұл аспап, ал аспапты сауатты қолдану - әзірлеушінің міндеті.

Пайдаланушы интерфейсінің генерациясы

Пайдаланушы интерфейсінің генерациясы белгілі бір артықшылықтар береді.

1. *Пайдаланушы интерфейсін нақты іске асырудан бизнес-процестерді дерексіздендіру мүмкіндігі.* Салдарынан, бұл жаңа технологияларға және платформаларға жылдам өтуге, жаңа бизнес-процестерді өзгертуге немесе енгізуге мүмкіндік береді. Дерексіз пішін технологияға тәуелді емес және бұрынғы күйінде қала береді. Минималды өзгерістермен оны жаңа технологиялар үшін кодты генерациялауға пайдалануға болады. Бұл өте маңызды қасиет болып табылады, себебі пайдаланушы интерфейсін әзірлеу технологиясы дамып келеді және дерекқорлардың технологияларымен салыстырғанда өте жылдам өзгереді.

2. *Пайдаланушы интерфейсін жүйеге келтіру және оны бүкіл қосымшада біркелкі ету мүмкіндігі.* Бұл функционалдылықтың сыртқы түріне, сондай-ақ ішкі құрамына, кодтың құрамын ұйымдастыру және

оның стандарттарына сәйкестігіне де қатысты.

3. *Енгізу жылдамдығы.* Өзгерістер жай ғана бүкіл қосымша бойынша таратылуы мүмкін, үлгілерді өзгерту немесе анықтамалардың дерексіз файлын өзгерту жеткілікті.

4. *Өзгерістерге икемділігі.* Метадеректер мен үлгілердің бөлінуі, сондай-ақ генерациялау кезінде өзгерістерді енгізудің үлкен жылдамдығы қосымшаларды оңай әрі жылдам түрлендіруге мүмкіндік береді. Пайдаланушылық операциялар орындалуының бірнеше нұсқасын енгізуге болады, сонымен көп функционалдық интерфейсті қалыптастыра отырып.

5. *Қателер мөлшерінің едәуір төмендеуі.* Пайдаланушылық интерфейс кодымен жұмыс істеу барысында, қосымшаның басқа деңгейлеріне жататын, көптеген шақырулар мен командаларды қамтитын адами фактор минимумға жеткізілген.

Пайдаланушылық интерфейстің тиімді генерациясы үшін, ең алдымен, кодтың пайдаланушылық интерфейске қатысы жоқ көлемін барынша төмендету қажет. Оның көлемін оңтайландыру жолымен, сондай-ақ функциялар кітапханаларында тиісті рәсімдерді құру және шақыру арқылы төмендеткен жөн. Сондай-ақ, максималды құрылымдалған және иерархиялық жақсы ұйымдастырылған үлгілерді құру қажет. Осының барлығы генерациялауға қолайлы, стандартталған және қарапайым құрылымы бар интерфейсті алуға мүмкіндік береді.

Егер бастапқыда интерфейс нашар жобаланса және ойластырылмаса, нашар стандартталса, онда оның осындай қасиеттері генератормен автоматты түрде жаңғыртылып отыратындығын ескерген жөн. Жақсы ойластырылған интерфейс, қолмен жасалғандай генерацияланатын болады.

Құжатты генерациялау

Бағдарламалық қамсыздандыруды әзірлеу оған техникалық құжатама жасау және жүргізуді де білдіреді. Толық және өзекті құжаттаманы әзірлеу тым бейнетті процесс және көп уақыт алады.

Сондай-ақ қосымшаның бағдарламалық кодына техникалық құжаттама жүргізу кезінде маңызды проблема - қосымша кодының келісілмегендігі болып табылады. Ол қосымша кодына түрлендіру енгізу салдарынан пайда болады, оның нәтижесінде өзгерістер құжаттамада толығымен немесе ішінара көрсетілмейді. Ереженің болуына қарамастан - кодтағы әрбір өзгерістен кейін сәйкесінше құжаттаманы да жаңарту қажет, бірақ шынайылығында жұмыстың осындай барысын сақтау көп күшті қажет етеді. Мұнан басқа, егжей-

тегжейлерге келетін болсақ, бағдарламашылар дәл болмауы мүмкін құжаттаманы оқығанша, бағдарламалық кодты қарастырғанды қалайды. Осыдан, құжаттаманы қанағаттандырусыз қолданудың тағы бір себебі, олардың әртүрлі дереккөздерде сақталуында болып табылады және қосымшаны жаңарту кезінде әрдайым құжаттама сәйкесінше өзгертілмейді.

Мәселенің шешімі кодты және құжаттаманы бір дереккөзде сақтау болып табылады. Жақсы құжатталған код бағдарламаның жақсы сапасына кодты жазу кезінде және жоба логикасын іске асыру түсінігінде де септігін тигізеді. Құжаттама және код бірігіп бір дереккөзде болған жағдайда бұл кодтың анықтығын, бағдарламалық қамсыздандырудың барлық өмірінің барысында құжаттаманы үнемі сүйемелдеу мүмкіндігі мен ыңғайлылығын арттырады. Код, техникалық құжаттама және пікірлер тұтастай бірыңғай түрінде басқарылады, олардың келіспеушілігі орын алмайды. Құжаттама генераторы екі тәсілмен қолданылады:

1) құжаттама метадеректердің анықтамаларынан, кодпен бірге бір мезгілде генерацияланады, сонымен бірге метадеректердің бір бөлігі кодтың өзіне және құжаттамаға енгізілетін пікірлер болуы тиіс;
2) бағдарламалық кодтан код және пікірлердің маңызды блоктары іріктеледі, осы ақпараттан құжаттама құралады. Бұл нұсқа өте ыңғайлы, себебі құжаттама код тұрған жерде орналасады.

Өзекті және толық құжаттаманы жүргізудің ең басты сәттері, бұл - пікірлер жүйесін сауатты ұйымдастыру болып табылады.

Құжаттаманы генерациялаудың артықшылықтары келесілер:

- құжаттама бағдарламалық кодпен келісілген болады, яғни нақтырақ және өзекті болады;
- құжаттаманы әзірлеуге қажетті уақыт пен күштің едәуір мөлшері үнемделеді. Босатылатын уақыт және ресурстарды құжаттаманың сапасын жақсарту үшін қолдануға болады;
- құжаттаманы берудің форматын немесе стилін өзгерту процесі қарапайымдылаурақ болады. Өз генераторы болған жағдайда, әрбір құжаттың форматын өзгерту қажет емес, үлгіні өзгерткен жеткілікті;
- құжаттама анағұрлым жылдам құрылады.

11.4. CASE-ҚҰРАЛДАРДЫҢ ЖІКТЕМЕСІ

Барлық заманауи CASE-құралдар негізінде типтер және санаттар бойынша жіктелуі мүмкін. Типтері бойынша жіктеу CASE-құралдардың функционалдық бағдарын бейнелейді. Санаттары бойынша жіктеу

атқарылған функциялар бойынша интеграцияланғандық дәрежесін анықтайды.

Типтер бойынша жіктеу бағдарламалық қамсыздандырудың өмірлік циклындегі CASE-құралдардың функционалдық тағайындалымын көрсетеді.

1. *Талдау және жобалау құралдары.* Осы типтің құралдары әзірлеу процесінің бастапқы кезеңдерін қолдау үшін пайдаланылады: пәндік саласын талдау, жүйеге қойылатын талаптарды әзірлеу, желілік сәулетті жобалау, бағдарламалық құралдарға қойылатын талаптарды әзірлеу, бағдарламалық сәулетті жобалау, бағдарламалық қамсыздандыруды техникалық жобалау. Осы типтің құралдары талдау мен жобалаудың танымал әдістемесін қолдайды. Шығысында жүйенің сипаттамалары, оның компоненттері мен осы компоненттерді байланыстыратын интерфейстері, бағдарламалық құралдардың сәулеті, деректер құрамының алгоритмдері мен анықтамаларын қоса алғанда, бағдарламалық құралдың техникалық жобасы генерацияланады.

2. *Деректер базасын және файлдарды жобалау құралдары.* Осындай типті құралдар деректерді логикалық модельдеуді, деректер моделдерін үшінші қалыпты формаға автоматтты түрде түрлендіруді, дерекқорлардың сызбаларын автоматты түрде генерациялауды және бағдарламалық код деңгейінде файлдардың форматтарын сипаттауды қамтамасыз етеді.

3. *Бағдарламалау және тестілеуге арналған құралдар.* Аталмыш құралдар сипаттамалар немесе модельдердің негізінде бағдарламалық қамсыздандырудың автоматтты түрде кодтың генерациясын орындайды. Графикалық редакторларды, репозиториймен жұмыс істеуді қолдау құралдарын, код генераторлары мен талдағыштарды, тест генераторларын, тестілермен қамту талдағыштарын, ретке келтірушілерді қамтиды.

4. *Сүйемелдеу құралдары.* Осындай типті құралдардың жалпы мақсаты қолданыстағы жүйені түзету, өзгерту, түрлендіру, жоба бойынша құжаттаманы қолдау болып табылады. Аталмыш құралдарға құжаттау құралдары, бағдарлама талдағыштары, БҚ өзгертулері және конфигурациясын басқару құралдары, қайта құрылымдау құралдары, әзірленген жүйені немесе бағдарламалық құралдарды жаңа операциялық немесе аппараттық құрылымға ауыстыруға мүмкіндік беретін жинақылықты қамтамасыз ету құралдарын қамтиды.

5. *Жобаны басқару құралдары.* Аталмыш типтегі құралдарға БҚ өмірлік циклын басқару процестін қолдау құралдары жатады. Олардың функциялары ретінде жоспарлау, бақлау, басшылық ету, ұйымдастыру, өзара әрекеттесу және т.с.с болып табылады.

Санаттар бойынша жіктелімі CASE-құралдардың атқарылатын қызметтері бойынша біріктірілуін бейнелейді.

1. *Tool санаты* (tool - жұмыс құралы) бірігудің ең төменгі деңгейінің құралдарын қамтиды. Құралдардың аталмыш санатсына бағдарламалық құралдар немесе жүйені әзірлеу кезінде шағын дербес міндетті шешетін аспаптық құралдар жатады. Әдетте аталмыш санатының құралдары CASE-құралдардың жоғары деңгейлі бірігетін компоненттері болып табылады.

2. *ToolKit санаты* (toolkit - құралдар жиынтығы, әзірлеуші пакеті) орта деңгейлі бірігетін CASE-құралдарды қамтиды. Аталмыш санаттың құралдары репозиторийді жоба туралы барлық ақпарат үшін қолданады және әзірлеу процесінің бір кезеңін немесе бір жұмысын қолдауға немесе бағдарламалық жүйенің өмірлік циклының көмекші немесе ұйымдастырушылық процестерінің бірін қолдауға бағдарланған. Аталмыш санатның CASE-құралдары, әдетте жалпы функционалдық бағдары бар аспаптық құралдардың біріктірілген жыинтығын білдіреді.

3. *Workbench* санаты (workbench - жұмыс орны) бірігудің ең жоғарғы деңгейіне ие. Құралдар БҚ өмірлік циклының барлық әзірлеу процесін және бірқатар көмекші және ұйымдастырушылық процестерін қолдайтын аспаптық құралдардың біріктірілген жиынтығын білдіреді. Репозиторийді жоба жөніндегі ақпаратты сақтауға қолданады, жоба бойынша ұжымдық жұмысты ұйымдастыруды қолдайды. Әдетте, Workbench санатын CASE-құралдардың қатары жатады, оларды біріктіріп пайдаланған жағдайда.

Деңгейлері бройынша жіктеу жүйенің және ұйымның бағдарламалық қамсыздандырудың өмірлік циклындегі CASE-құралдардың әсер ету саласымен байланысты.

1. *Жоғарғы* (Upper) CASE-құралдар - компьютерлік жоспарлау құралдары. Олардың негізгі мақсаты ұйымдар, кәсіпорындар және нақты жоба басшыларына жоба жоспарын құру және саясатын ұйымдастыруда көмек көрсету болып табылады. Аталмыш деңгейдің CASE-құралдары пәндік саласының моделін құрауға, әртүрлі сценарийлердің (қолданыстағы, ең үздік, ең нашар) талдауын жасауға, оңтайлы шешім қабылдау үшін ақпарат жинауға мүмкіндік береді.

2. *Орташа* (Middle) CASE-құралдар әзірлеу процесінің бастапқы кезеңдерін қолдайды (пәндік саласын талдау, жүйеге қойылатын талаптарды әзірлеу, жүйелік сәулетті жобалау, жобалауға қойылатын талаптарды әзірлеу, бағдарламалық сәулетті жобалау. Әдетте аталмыш құралдар жоба жөніндегі ақпаратты жинақтау және сақтау мүмкіндіктеріне ие. Бұл жинақталған деректерді ағымдық және басқа жобаларда қолдануға мүмкіндік береді. Жиі прототиптендіруді және

автоматты түрде құжаттамалауды қолдайды.

3. *Төменгі* (Lower) CASE-құралдар БҚ әзірлеу процесінің екінші жұмыс бөлігін қолдайды. Сипаттамалар әдетте, әзірленіп жатқан бағдарламалық құралдың немесе жүйенің бағдарламалық кодына тікелей түрлендірілетін модельдер түрінде беріледі. Автоматты түрде 90% дейінгі кодтар генерацияланады. Кодты генераторларға арналған кіріс ақпараты - аталмыш деңгейдің CASE-құралдарында, сондай-ақ орташа деңгейдегі CASE-құралдарда әзірленген сипаттамлар болып табылады. Төменгі деңгейлі CASE-құралдар, әдетте, прототиптендіруді, тестілеуді, конфигурацияны басқаруды, құжаттамаларды генерациялауды қолдайды, БҚ түрлендірілімін және сүйемелдеуін жеңілдетеді.

11.5. ЗАМАНАУИ CASE-ҚҰРАЛДАРҒА ШОЛУ

Oracle Desinger

Oracle фирмасының *Oracle Designer* CASE-құралдары, Oracle Developer және Oracle қосымшаларының құралдарымен бірігіп қамтамасыз ететін біріктірілген CASE-құрал болып табылады.

Oracle компаниясының аталмыш өнімі, деректерді өңдеу қосымшасын құрудың барлық кезеңдерін толық қолдауы мүмкін. Дегенмен, басқа құралдардан айырмашылығы, ол тәжірибе жүзінде бір мақсаттық ДББҚ - Oracle Server қолдайды.

Құрылымы және қызметі. Oracle Designer әдістер мен оларды қолдайтын бағдарламалық өнімдердің тұқымдасын білдіреді.

Oracle Designer пәндік саланың әртүрлі модельдерін (диаграммаларды) әзірлеу кезінде графикалық интерфейсті қамтамасыз етеді. Модельдерді құру барысында олар туралы ақпарат репозиторийге енгізіледі. Oracle Designer құрамына кеесі компоненттер кіреді:

- Repository Administrator - репозиторийді басқару құралдары (қосымшаларды құру және жою, әртүрлі пайдаланушылар тарапынан деректерге рұқсатын басқару, деректердің экспорты және импорты);
- Repository Object Navigator - репозиторийдің барлық элементтеріне көп терезелі нысанға бағытталған интерфейсті қамтамасыз ететін, репозиторийге рұқсат құралы;
- Process Modeler - бизнес-процестер реинжинирингі және сапаны

басқарудың жаһандық жүйесінің тұжырымдамаларында негізделген ұйымның қызметін талдау және ұйымдастыру құралы;

- Systems Modeler - «болмыс-байланыс» диаграммаларын, функционалдық иерархиялар диаграммалары, деректер ағын диаграммаларын құруға арналған құралдарды және әртүрлі типтегі репозиторийлердің байланыс нысандарын талдау және түрлендіру құралын қамтитын, жобаланып отырған ақпараттық жүйенің функционалдық және ақпараттық моделін құру құралдарының жиынтығы;
- Systems Designer - реляциялық дерекқорын құру құралын (Data Diagrammer), сондай-ақ деректермен өзара әрекеттестігін көрсететін диаграмманы құру құралдарын, сақталынатын рәсімдермен іске асырылатын, қосымшалардың иерархияларын, құрылымын және логикасын қамтитын, БҚ жобалау құралдарының жиынтығы;
- Server Generator - Oracle ДБ нысандарын сипаттау генераторы (кестелер, индекстер, кілттер, жүйеліліктер және т.б.). Oracle басқа генерация және ДБ реверсті инжинирингі (шектеулерімен) DB2, MS SQL Server, Sybase ДББҚ үшін орындала алады, сондай-ақ ANSI SQL DDL стандарты және ODBC арқылы рұқсат алынатын деректер базасы үшін орындалады;
- Forms Generator (Oracle Forms арналған қосымшалар генераторы). Генерацияланатын қосымшалар өзіне әртүрлі экранды формаларды, деректерді бақылау құралдарын, тұтастығын шектеуді тексеру және автоматты түрде еске салу құралдарын қамтиды. Қосымшамен кейінірек жұмыс істеу Oracle Developer ортасында орындалады;
- Repository Reports - Oracle Reports біріктірілген және есептерді орыстандыруға, сондай-ақ ақпараттың құрылымдық берілуін өзгертуге мүмкіндік беретін стандартты есептер генераторы. Oracle Designer репозиторийі барлық жобалау деректерінің қоймасын білдіреді және бірнеше әзірлеушілермен ақпаратты қатарлас жаңаруын қамтамасыз ететін көп пайдаланушылар режимінде жұмыс істей алады. Жобалау процесінде сөздіктің нысандары арасындағы тоғыспалы сілтемелер автоматты түрде сүйемелденеді және модельденетін пәндік сала туралы 70-тен астам стандартты есептер генерацияланады. Репозиторийді сақтаудың табиғи ортасы - Oracle дерекқоры.

Oracle Developer басқа қосымшалар генерациясы сондай-ақ Oracle Wfeb Application Server, C++ және Visual Basic үшін де орындалады.

Басқа құралдармен өзара әрекеттесу. API (Application Programming Interface) қосымшасының ашық интерфейсін қолдана отырып, Oracle Designer басқа құралдарымен біріктіруге болады. Бұдан

басқа, басқа CASE-құралдарымен ақпарат алмасу мақсатында, репозиторий нысандарының экспорты/импорты үшін Oracle CASE Exchange құралын пайдалануға болады.

Oracle Developer - Windows, Macintosh графикалық ортасында жұмыс істейтін тасымалданатын қосымшалардың әзірленуін қамтамасыз етеді. Windows ортасында Oracle Developer қосымшасының басқа құралдарымен бірігуі OLE (Object Linking and Embedding) механизмі және VBX (Visual Basic extension) басқару элементтері арқылы іске асырылады. Қосымшаның басқа ДББҚ өзара әрекеттесуі ODBC, Oracle Open Gateway және API арналған Oracle Client Adapter құралдарының көмегімен іске асырылады.

ERwin және Pbwin

Бағдарламалық жүйелердің өмірлік циклын тиімді сүйемелдеуін қамтамасыз ететін заманауи аспаптық құралдарға Computer Associates компаниясының ***AllFusion*** қатары жатады. Аталмыш қатар ұйымдағы ақпараттық жүйелерді әзірлеуге, енгізуге және басқаруға арналған біріктірілген құралдардың жиынтығын білдіреді.

Қатардың барлық құралдарының репозиторийлері бар және құжаттаманы автоматты түрде құруды қолдайды. AllFusion қатарының құралдарын жиынтық пайдалану әзірлеу және сүйемелдеуге жұмсалатын шығындарды қысқартуға мүмкіндік береді, әзірленімнің ұзақтығын және құнын төмендетеді, ӨЦ процестерінің сапасын, әзірленімнің аралық және ақырғы өнімдерінің сапасын арттырады.

Функционалдық тағайындалуы бойынша AllFusion қатарының негізгі компоненттерін келесі типтерге бөлуге болады:

- дерекқорды, бизнес-процестерді және БҚ компоненттерін модельдеу құралдары;
- өзгерістер мен конфигурацияны басқару құралдары;
- жобаларды және процестерді біріктіре басқарцу құралдары.

Қазір AllFusion қатарына ондаған өнімдер кіреді (CA ERwin Data Modeler, AllFusion Process Modeler, CA ERwin Data Profiler, CA ERwin Data Model Validator және бсқалар). Алайда барлығы Erwin және BPwin құралдарынан басталған, олар Logic Works фирмасымен әзірленген. 1998 ж. Logic Works компаниясы Platinum Technology фирмасымен құрылған болатын. Ол өз кезегінде, бір жылдан кейін, 1999 ж. Computer Associates сатып алынды, ол қазіргі уақытта осы өнімдерді қолдаумен және дамытумен айналысып келеді.

AllFusion ERwin Data Modeler. Еретеректе ERwin, *AllFusion ERwin Data Modeler* - дерекқорды жобалауға арналған CASE-

құрал, ол дерекқорды, қоймаларды және деректер витринасын құруға, құжаттауға және сүйемелдеуге мүмкіндік береді. AllFusion ERwin Data Modeler (ERwin) әкімшілерге арналған дерекқорды, желілік талдаушылардың, жобалаушылардың дерекқорын, әзірлеушілердің, жоба жетекшілерінің дерекқорын әзірлейтін және барлық компанияларға пайдалануға арналған. AllFusion ERwin Data Modeler корпоративті өзгерістер барысында, сондай-ақ қарқынды өзгерістегі технология жағдайларында деректерді басқаруға мүмкіндік береді.

ERwin өнімдердің тұқымдасы - IDEF1X әдісін қолданатын, деректерді тұжырымдамалық модельдеу құралдарының жиынтығын білдіреді. ERwin ДБ сызбасын жобалауды, мақсаттық ДББҚ (Oracle, Informix, Sybase, DB2, Microsoft SQL Server және басқалар) тілінде оны сипаттау генерациясын және қолданыстағы ДБ реверсті инжинирингін іске асырады. 4GL қосымшасын әзірлеудің ең таралған құралдарына бағдарланған бірнеше конфигурацияда шығарылады. PowerBuilder, Visual Basic, Delphi қосымшаларының клиенттік бөлімін әзірлеудің танымал құралдарымен біріктіріледі, бұл қосымшаның коды автоматты түрде генерациялауға мүмкіндік береді.

ERwin модельді көзбен шолудың қуатты құралдары бар, атап айтқанда әртүрлі шрифттерді, түстерді және бейнелерді әртүрлі деңгейлерде пайдалану, мысалы, болмысын сипаттау деңгейінде, болмыстың бастапқы кілттерінің деңгейінде және басқалар. Бұл ERwin құралдары жүйені әзірлеушілердің немесе бөгде тұлғаларға модельді таныстыру кезінде көмектеседі.

ERwin моделін деректерді логикалық жіне физикалық тұрғыдан таныстыру үшін бір мезгілде пайдалану мүмкіншілігі, жұмысты аяқтағаннан кейін толығымен құжатталған модельді алуға мүмкіндік береді. ERwin, бизнесс-процестерді модельдеу құралы BPwin ретінде Logic Works - RPTwin фирмасының есептер генераторымен біріктірілген. Бұл құрал, әртүрлі ракурстар мен қырларды жарықтандыра отырып, модель бойынша толық есептер алуға мүмкіндік береді. RPTwin құралы ERwin бірге жектізіледі және модель бойынша жан-жақты ақпарат алуға мүмкіндік беретін кіріктірілген есептердің көп жиынтығы бар. Деректер құрамын құжаттау модельдеудің ең маңызды бөлігі болып табылады, себебі бұл жүйені алып жүретін басқа әзірлеушілерге немесе тұлғаларға ішкі құрылымға жылдам бағдарлануға және компоненттердің тағайындалуын түсінуге мүмкіндік береді.

Топтық әзірленімді басқару үшін Model Mart құралы қолданылады, ол Erwin көмегімен құрылған модельдерге көп пайдаланушылық рұқсатты қамтамасыз етеді. Модельдер орталық серверде сақталады және жобалау тобының барлық қатысушыларына қолжетімді.

Model Mart, ірі ақпараттық жүйелердің әзірленімдерін басқару құралдарына қойылатын бірқатар талаптарды қанағаттандырады.

Бірігіп модельдеу. Жобаның әрбір қатысушысының кез келген уақытта қызығушылық танытқан моделін іздеу және рұқсат құралы бар. Біріккен жұмыс кезінде үш режим қолданылады: қорғалмаған, қорғалған және қарап шығу режимі. Қарап шығу режимінде модельді кез келген өзгертуге тыйым салынады. Қорғалған режимде, бір пайдаланушы жұмыс істейтін модель, басқа пайдаланушымен өзгертіле алмайды. Қорғалмаған режимде пайдаланушылар шынайы уақыт масштабында жалпы модельдермен жұмыс істей алады. Сонымен бірге пайда болатын қақтығыстар арнайы модуль Intelligent Conflict Resolution (ICR) көмегімен шешіледі.

Шешімдер кітапханасын құру. Model Mart, іске асырылған жобалардың ең сәтті үзінділерін қамтитын стандартты шешімдер кітапханасын құруға, типтік модельдерді қажет болған жағдайда үлкен жүйелер «жинағына» біріктіріп пайдалануға және жинақтауға мүмкіндік береді. Қолданыстағы дерекқордың негізінде Erwin көмегімен модельдерді қалпына келтіру мүмкін (реверсті инжиниринг), олар жаңа жүйеге жарамдылығын талдау барысында модельдер кітапханасынан алынған типтік модельдермен біріге алады.

Рұқсатты басқару. Жобаның әрбір қатысушысына рұқсат құқығы белгіленеді, оларға сәйкес олар белгілі бір модельдермен жұмыс істеу мүмкіндігін алады. Рұқсат құқықтары топтарға және жобаның жекелеген қатысушыларына анықталуы мүмкін. Әртүрлі жобаларға қатысатын мамандардың рөлі өзгеруі мүмкін, сол себептен Model Mart жоба қатысушыларының кітапханаларға рұқсат беру құқықтарын анықтауға және басқаруға болады.

Басқа құралдармен өзара әрекеттесу. Erwin Rational Rose-бен өзара әрекетестікті қолдайды: ERwin Translation Wizard модулі Rational Rose нысанды моделін ERwin моделіне айырбастауға (және кері) және содан кейін Erwin көмегімен Erwin-де кез келген қолданылатын ДББҚ үшін ДБ сызбасын генерациялауға мүмкіндік береді.

Сондай-ақ, деректерді BPwin және Oracle Designer репозиторийлерінен Erwin репозиторийіне(н) импорттау/экспорттау мүмкіншілігі бар.

AllFusion Process Modeler. Ертеректе BPwin, *AllFusion Process Modeler* деп аталған - қуатты бағдарламалық өнім, оның көмегімен бизнес-процестерді модельдеуге, талдауға, сипаттауға және кейінгі оңтайландыруын жүргізуге болады.

BPwin көмегімен бизнес-процестердің графикалық модельдерін құруға болады. Жұмыстарды орындау сызбасының графикалық бейнесі,

құжат айналымын ұйымдастыру, әртүрлі ақпарат түрлерімен алмасу бизнесті ұйымдастырудың қолданыстағы моделін көзбен шолуға мүмкіндік береді. Бұл ұйымды басқару міндеттерін шешу үшін алдыңғы қатарлы инженерлік технологияларды пайдалану мүмкіндігін береді.

BPwin функционалдық модельдеуді, жұмыс ағындарын және деректер ағындарын модельдеуді қолдайды. Тиісті диаграммалар IDEF0, IDEF3 және DFD стандарттарының негізінде іске асырылған. Функционалдық модельдеу, жүйелі түрде орындалатын міндеттерге (функцияларға) назар аудара отырып, бизнес процестердің жүйелендірілген талдауын жүзеге асыру мүмкіндігін береді. Жұмыс ағынын модельдеу процесті орындаудың логикасын талдауды қамтамасыз етеді. Дерекқорды модельдеу әртүрлі міндеттердің арасында деректерді алмасуға назар аударуға мүмкіндік береді. Мұнан басқа, BPwin жекелеген модельдер құрылатын болғандықтан, сондай-ақ аралас модельдер де құрыла алады.

Ұйымның жұмысын кешенді түрде талдау, және үлкен модельдерді құру үшін, BPwin-де талдап-тексеру қарастырылған. Модельдер топтарға бөлінеді. Әрбір модель талдап-тексерудің төменгі деңгейінде беріледі. Сонымен бірге, модельдер мен олардың элементтері арасындағы өзара байланыстылық сақталады. BPwin көмегімен модельді құрамдас бөліктерге бөлуге, олардың әрқайсысымен жұмыс жасауға болады, содан кейін қайтадан бірыңғай модельге біріктіруге болады.

BPwin модельдердің келесі түрлерін құруға мүмкіндік береді:

- IDEF0 стандартының негізінде құрылған функционалдық диаграммалар;
- IDEF3 стандартының негізінде құрылған жұмыс ағыны диаграммалары. Бұл диаграммалар, тапсырмаларды белгілі бір жүйелілікте беру арқылы процестің логикасын көрсетуге мүмкіндік береді. Кейінірек бұл модельдерді динамикалық модельдерді құруға арналған негіз ретінде пайдалануға болады;
- деректер ағынының диаграммалары;
- бағалық талдау модельдері. Бұл модельдер бағалық талдау ережелері бойынша тұрғызылады (Activity Base Costing - талдау). Модель, егер толығымен аяқталған және кереғар емес функционалдық модель бар болған жағдайда ғана тұрғызылауы мүмкін. Функционалдық модельдердің осы тапсырмаларының әрқайсысына шығындарды таныстыратын метрикалар тағайындалады. Модель үшін шығын орталықтары анықталады. Нәтижесінде бағалық талдау моделі пайда болады;
- динамикалық модельдер. Бұл модельдер жұмыс ағыны

диаграммасының негізінде тұрғызылуы мүмкін. BPwin процесі тапсырмалары жағдайларының дикретті өзгеруінің барысында әсерді зерттеуге мүмкіндік береді. Ол үшін процесс қылығының әртүрлі сценарийлері берілуі мүмкін.

BPwin-нің модельдер мен жобаларды құжаттау құралдарының кең жиынтығын бар екендігін айта кеткен жөн.

Rational Rose

Rational Rose - БҚ талдау және жобалау процесі автоматтандыруға қолданылатын, сондай-ақ әртүрлі тілдердегі кодтарды генерациялау үшін және жобалық құжаттаманы құруға арналған Rational Software Corporation фирмасының нысанға бағытталған CASE-құралдар тұқымдасы.

Rational Rose негізінде UML модельдеу тілінде негізделген, нысанға бағытталған талдау және жобалау әдісі жатыр. Rational Rose бағдарлама кодтарының генерациясын C++, Smalltalk, Java, PowerBuilder, CORBA Interface Definition Language (IDL) және басқалар үшін іске асырады, сондай-ақ диаграммалар мен сипаттамалар түріндегі жобалық құжаттама алуға мүмкіндік береді. Бұдан басқа, Rational Rose құрамында жаңа жобаларда бағдарламалық компоненттердің қайта қолдануды қамтамасыз ететін, бағдарламаны реверсті инжиниринг құралары бар.

Rational Rose жұмысының негізінде жүйенің сәулетін, оның статикалық және динамикалық қырларын анықтайтын UML-диаграммалар және сипаттамалар құрылады. Rational Rose құрамында келесі құрылымдық компоненттерді бөліп көрсетуге болады: репозиторий, пайдаланушының графикалық интерфейсі, жобаны қарап шығу құралдары (browser), жобаны бақылау құралдары, статистиканы жинау құралдары, құжаттар генераторы, кодтар генераторы (әрбір тіл үшін жеке) және реверсті инжинирингті қамтамасыз ететін бағдарламалау тілдеріне арналған талдағыш. Rational Rose - UML-диаграмм технологиясын қолдайды және Oracle және SQL сызбасын генерациялайды.

Репозиторий нысанға бағытталған дерекқорды білдіреді. Қарап шығу құралдары жоба бойынша навигацияны қамтамасыз етеді. Статистиканы бақылау және жинау құралдары жобаны сипаттауды аяқтағаннан кейін емес, оны дамыту барысында қателіктерді табуға және жоюға мүмкіндік береді. Есептер генераторы репозиторийдегі ақпараттың негізінде шығыс құжаттарын қалыптастырады.

Бағдарлама кодтарын автоматты түрде генерациялау құралдары, кластар диаграмасында қамтылған ақпаратты және компоненттерді

пайдала отырып, тақырыптар файлдарын және кластар мен нысандардың сипаттама файлдарын құрастырады. Жұмыс барысында талдағыш бастапқы мәтіндердің дұрыстығына және қателер диагностикасына бақылау жүргізеді. Оның жұмысының нәтижесінен алынған модель, әртүрлі жобаларда пайдаланыла алады. Талдағыштың кіру және шығу бойынша икемге келтіруінің кең мүмкіншілігі бар. Мысалы, бастапқы файлдардың типтерін, негізгі компиляторды анықтауға болады, құрастырылатын модельге қандай ақпарат енгізілуі тиіс екендігін тапсыруға және шығыс моделінің қандай ақпараты құрылатын модельге қосылуы тиіс және шығыс моделінің қандай элементтерін экранға шығаруға болатындығын анықтауға болады. Осылайша, Rational Rose бағдарламалық компоненттерді қайтадан пайдалану мүмкіндігімен қамтамасыз етеді.

Rational Rose CASE-құралдарының көмегімен жобаны құру нәтижесінде келесі құжаттар құрылады:

- жиынтығында әзірленіп жатқан бағдарламалық жүйенің моделін білдіретін UML диаграммалары;
- кластардың, нысандардың, атрибуттар мен операциялардың сипаттамасы;
- бағдарлама мәтіндерін дайындау.

Rational Rose келесі нұсқалары бар: Modeler Edition (UML тілінің тікелей сүйемелдеуін қамтамасыз етеді), Enterprise Edition (кәсіпорын масштабындағы жобаларды әзірлеуге арналған интеграциялау платформасын білдіреді), Professional Edition (Rational Rose Modeler Edition барлық мүмкіншіліктерін плюс бағдарламалық код генерациясын және реверсті инжинирингті қосады) және UNIX арналған Rose.

Жобамен командалық жұмысты қолдау үшін БҚ өмірлік циклының әрбір сатысында Rational Suite өнімдерінің кіріктірілген жиынтығы бар.

Rational Rose ақпараттық жүйенің кез келген тапсырмасын шешу кезінде пайдасы тиеді: бизнес процестерді талдаудан бастап белгілі бір бағдарламалау тіліндегі кодты генерацияға дейін. Осындай қор тек қана жаңа жүйені жобалап қана қоймай, сонымен бірге кері жобалау процесін жүргізіп, ескісін өзгертуге мүмкіндік береді.

Silverrun

Computer Systems Advisers, Inc. (CSA) американдық фирмасының ***Silverrun*** CASE-құралы бағдарламалық қамсыздандырудың өмірлік циклының шиыршықты моделіне бағдарланған. Ол функционалдық

және ақпараттық модельдердің (деректер ағыны диаграммалары және «болмыс-байланыс» диаграммалары) бөлек тұрғызылуына негізделген кез келген әдістемені қолдау үшін қолдануға келеді.

Нақты әдістемеге икемдеу модельдердің қажетті графикалық нотациясын және жобалық сипаттамларды тексеру ережелерінің жиынтығын таңдауымен қамтамасыз етіледі. Жүйеде кең тараған әдістемеге арналған дайын икемдеулер бар: DATARUN (Silverrun қодайтын негізгі әдістеме), Gane/Sarson, Yourdon/ DeMarco, Merise, Ward/Mellor, Information Engineering. Жобаға енгізілген әрбір түсінік үшін өзіндік сипаттаушыларды қосу мүмкіншілігі бар.

Silverrun сәулеті қажет болған жағдайда әзірлеу ортасын өсіруге мүмкіндік береді.

Silverrun модульдік құрылымы бар және төрт модульден тұрады, олардың әрқайсысы өзінше жеке өнім болып табылады және басқа модульдермен байланыссыз сатып алына алады және қолданыла алады.

Деректер ағынының диаграммасы түрінде бизнес-процестердің моделін тұрғызу модулі (BPM - Business Process Modeler) тексерілетін ұйымның немесе құрылып жатқан АЖ жұмыс істеуін модельдеуге мүмкіндік береді.

Деректерді тұжырымдамалық модельдеу модулі (ERX - Entity-Relationship eXpert) нақты іске асыруға байланыстырылмаған «болмыс-байланыс» деректер моделін тұрғызуды қамтамасыз етеді. Бұл модульдің, деректердің өзара байланысы туралы маңызды сұрақтарға жауаптардың көмегімен дұрыс қалыптастырылған деректер моделін құруға мүмкіндік беретін, кіріктірілген сарапшылық жүйесі бар.

Реляциялық моделдеу модулі (R D M - Relational Data Modeler) реляциялық дерекқорында іске асыру үшін арналған, «болмыс-байланыс» талдап тексерілген модельдерді құруға мүмкіндік береді. Бұл модульде дерекқорды тұрғызумен байланысты барлық конструкциялар құжатталады: индекстер, триггерлер, сақталатын рәсімдер және т.б.

Жұмыс тобы репозиторийінің менеджері (WRM - Workgroup Repository Manager) барлық модельдер үшін жалпы ақпаратты сақтауға арналған деректер сөздігі ретінде қолданылады, сондай-ақ Silverrun модульдерінің жобалаудың бірыңғай ортасына біріктірілуін қамтамасыз етеді.

Модельдерді тұрғызудың бейнелеу құралдарының жоғары икемділігіне және әртүрлілігіне ақы ретінде, Silverrun-ның әртүрлі модельдердің компоненттерінің арасындағы қатал өзара бақылаудың болмағаны болып табылады (мысалы, DFD әртүрлі декомпозиция деңгейлерінің арасындағы өзгерістерді автоматты түрде тарату мүмкіндігі). Дегенмен, бұл жетіспеушілік БҚ ӨЦ каскадты моделін

пайдаланған кезде ғана маңызды мәні болу мүмкіндігін ескерген жөн.

Silverrun-да дерекқордың сызбасын автоматтты түрде генерациялау үшін көп тараған ДББҚ-на көпірлер бар: Oracle, Informix, DB2, Ingres, Progress, SQL Server, SQLBase, Sybase. Деректерді қосымшаларды әзірлеу құралдарына табыстау үшін 4GL тілдеріне көпірлер бар: JAM, PowerBuilder, SQL Windows, Uniface, NewEra, Delphi. Барлық көпірлер Silverrun RDM-ге тиісті ДББҚ каталогтарынан ақпаратты немесе 4GL тілдерін жүктеуге мүмкіндік береді. Бұл жаңа платформаларға пайдаланылып келе жатқан дерекқорды және қолданбалы жүйелерді құжаттамалауға қайта жобалауға немесе тасымалдауға мүмкіндік береді.

Жобалауды автоматтандырудың басқа құралдарымен деректер алмасуы үшін, сондай-ақ әртүрлі стандарттарға сәйкес арнайы есептерді жасау үшін Silverrun жүйесінде жобалық ақпараттарды сыртқы файлдарға берудің келесі тәсілдері бар:

- есептер жүйесі есептің құрамындағысын репозиторийден файлға жүктеу және мәтіндік редакторға жүктеуді және басқа есепке қосуға мүмкіндік береді;
- экспорт/импорт жүйесі әртүрлі жүйелермен: басқа CASE-құралдармен, ДББҚ, мәтіндік редакторлармен және электронды кестелермен деректермен алмасу мүмкіншілігін береді;
- репозиторий деректеріне рұқсат алу үшін, дерекқорды басқарудың ең көп таралған жүйелерінен барлық жобалық ақпаратты тікелей осы ДББҚ форматында сақтау қамтамасыз етілген.

Silverrun-ның әртүрлі операциялық платформаларға арналған іске асырулары бар, олардың арасында жобалық деректермен алмасу мүмкіншілігімен.

БАҚЫЛАУ СҰРАҚТАРЫ МЕН ТАПСЫРМАЛАРЫ

1. Қандай бағдарламалық құрал CASE-құралы деп аталады?
2. CASE-технологиясы деп нені атаймыз?
3. CASE-құралдары негізделетін негізін қалаушы қағидаларды атап өтіңіз.
4. CASE-құралдарды тұжрымдамалық қалау негізінде қандай ережелер жатыр?
5. CASE-құралдардың негізгі компоненттерін атап өтіңіз және сипаттап беріңіз.
6. CASE-құралдарда әдетте қандай бақылау түрлері іске асырылады?
7. CASE-құралдарда жоба бойынша құжаттамаларды автоматты түрде генерациялау кезінде іске асырылатын негізгі есеп типтерін атап өтіңіз.

8. Бағдарламалық өнімдерді әзірлеу процесін қолдауды қамтамасыз ететін, заманауи CASE-құралдардың ерекшеліктерін атап өтіңіз.
9. Кодты генерациялау құралдары қандай белгілер бойынша бөлінеді?
10. CASE-құралдарды типтері бойынша жіктеу нені көрсетеді?
11. CASE-құралдардың типтерін атап өтіңіз және сипаттап беріңіз.
12. CASE-құралдарды санаттары бойынша жіктеу нені көрсетеді?
13. CASE-құралдардың санаттарын атап өтіңіз және сипаттап беріңіз.
14. CASE-құралдарды деңгейлер бойынша жіктеу нені көрсетеді?
15. CASE-құралдардың деңгейлерін атап өтіңіз және сипаттап беріңіз.

III

ТАРАУ

ҚҰЖАТТАУ ЖӘНЕ СЕРТИФИКАТАУ

12-тарау

СТАНДАРТТАУ ТУРАЛЫ ЖАЛПЫ МӘЛІМЕТТЕР

12.1. СТАНДАРТТАУДЫҢ МАҚСАТТАРЫ МЕН МІНДЕТТЕРІ. СТАНДАРТТАУДЫҢ ДЕҢГЕЙЛЕРІ

Бағдарламалық қамсыздандырудың сапасына қойылатын талаптардың үздіксіз өсуі халықаралық стандарттар мен регламенттелген технологияларды құруды және белсенді пайдалануды ынталандырады. Бүгінгі уақытта бағдарламалық құралдар әзірленімін стандарттау олардың өмірлік циклының барлық кезеңдерін қамтиды.

Стандарттау қоғамдағы әлеуметтік, өндірістік және экономикалық қатынастарды ұйымдастырудың тиімді құралдарының бірі болып табылады.

2002 ж. 27 желтоқсанындағы № 184-ФЗ «Техникалық реттеу туралы» Федералды заңында (22.12.2014 ж. бастап өзгерістер мен толықтыруларымен күшіне енеді) стандарт және стандарттаудың келесі анықтамалары беріледі.

Стандарт - ерікті түрде және көп мәрте пайдалану мақсатында, өнімнің сипаттамалары, өндіру, пайдалану, сақтау, тасымалдау, сату және кәдеге жарату, жұмыстарды орындау немесе қызмет көрсетудің процестерін жүзеге асыру ережелері және сипаттамалары белгіленетін құжат. Стандартта сондай-ақ, символикаға, орамға, таңбалауға немесе зат белгілерге және оларды басу ережелеріне қойылатын талаптар да қамтылуы мүмкін.

Стандарттау - көп мәрте және ерікті пайдалану үшін ережелер және сипаттамалар белгілеу арқылы өнім өндіру және айналымға шығару саласындағы және өнімнің, жұмыстың немесе көрсетілетін қызметтің бәсекеге қабілеттілігін арттыруды ретке келтірудің оңтайлы деңгейіне қол жеткізуге бағытталған қызмет.

Стандарттау бойынша халықаралық ұйымы (ИСО) және Халықаралық электртехникалық комиссия (ХЭК) стандарттауға келесідей анықтама береді.

Стандарттау - шынайы бар немесе әлеуетті міндеттерге қатысты жалпыға ортақ және көп мәртелі пайдалануға арналған ережелерді белгілеу арқылы белгілі бір салада ретке келтірудің оңтайлы дәрежесіне қол жеткізуге бағытталған.

Стандарттау келесілерді қамтамасыз ету мақсатында нормалар, ережелер және сипаттамалар белгілеуді білдіреді:

- өнімнің, жұмыстар мен қызметтердің қоршаған орта, өмірге, денсаулыққа және мүлік үшін қауіпсіздігі;
- техникалық және ақпараттық үйлесімділік, сондай-ақ өнімдердің өзара алмастырушылығы;
- ғылым, техника және технологияның даму деңгейіне сәйкес өнімнің, жұмыстың және қызметтің сапасы;
- өлшемдер тұтастығы;
- барлық ресурс түрлерінің үнемдеуі;
- табиғи және техногенді апаттар және басқа төтенше жағдайлардың пайда болу тәуекелін ескере отырып, шаруашылық нысандардың қауіпсіздігі;
- елдің қорғаныс қабілеті және жинақылауға әзірлігі.

Стандарттау, техникалық ғылым ретінде бастапқы ережелерге - стандарттау қағидадаларына негізденеді:

- барлық мүдделі тараптардың келісімінің негізінде стандарттау бойынша құжаттарды әзірлеу;
- әлеуметтік, техникалық және экономикалық қажеттілік көзқарасынан стандартты әзірлеудің мақсаттылығы;
- әзірлеудегі артықшылық - бұл, адамдардың өмірінің, денсаулығының және мүлкінің қауіпсіздігін, қоршаған ортаны қорғауды қамтамасыз етілуіне септігін тигізетін, өнімдердің үйлесімділігін және өзара алмастырушылығын қамтамасыз ететін стандарттар;
- өзара байланысқан нысандарды стандарттаудың кешенділігі;
- талаптар белгілеу және олардың нысан түрде тексерілуі мүмкін, стандарттау нысандарының негізгі қасиеттеріне бір мәнділігі;
- стандарттарды ерікті қолдану;
- стандарттарды әзірлеу кезіндегі мүдделі тұлғалардың мүдделерін барынша есепке алу;
- стандарттарды бірдей қолдануға арналған шарттарды қамтамасыз ету;
- халықаралық стандарттарды ұлттық стандарттарды әзірлеуге

арналған негіз ретінде пайдалану;

- қауіпсіз өнімді өндіру және айналымға шығаруға, халықаралық саудаға кедергілер жасауға жол берілмеушілік;
- техникалық регламенттеуге қайшы келетін стандарттарды белгілеуге жол берілмеушілік.

Стандарттау нысаны - стандарттауға жататын немесе стандартталған өнім, жұмыс (процесс), қызмет.

12.2. СТАНДАРТТАУ БОЙЫНША НОРМАТИВТІК ҚҰЖАТТАР

Өнімге қойылатын техникалық талаптар, оларды дайындау және тексеру, таңбалау және қаптау, сақтау және тасымалдау ережелері ***нормалар*** деп аталады.

Нормативтік құжат - белгілі бір қызмет түрлеріне немесе олардың нәтижелеріне қатысты ережелерді, жалпы қағидалар немесе сипаттамаларды белгілейтін құжат.

Стандарт - тараптардың келісімі негізінде әзірленген және уәкілетті органмен бекітілген, қызметтің әр түріне немесе олардың нәтижелеріне қатысты ұзақ мерзімге және тұрақты пайдалану үшін ережелер, жалпы принциптер немесе сипаттамалар белгіленетін құжат. Бұл құжаттың міндеті берілген салада ең үздік дәрежедегі реттілікке қол жеткізу.

Стандарттау бойынша нормативтік құжаттармен белгіленетін талаптар, заманауи ғылым, техника және технология жетістіктеріне, халықаралық (аймақтық) стандарттар, ережелер, нормалар және стандарттау бойынша ұсыныстар, басқа елдердің прогрессивті ұлттық стандарттарына негізделеді.

Ақпараттық технологиялар саласындағы стандарттау - есептеу техникасының аппараттық және бадарламалық құралдарын пайдалану және өндіру, сипаттамасы бойынша келісім қабылдау; стандарттар, нормалар, ережелер және т.с.с. белгілеу және қолдану.

Ақпараттық технология саласындағы стандарттау, ақпараттық технологиялар түрлерінің, олардың компоненттерін және халықаралық ақпараттық алмасудағы техникалық кедергілерді жоюға, өзінің функционалдық тағайындалуының сәйкестік дәрежесін арттыруға бағытталған.

Стандарттау деңгейі, әлемнің қандай географиялық, экономикалық, саяси аймағының қатысушылары стандартты қабылдағанына байланысты ерекшеленеді.

Ресей Федерациясының *Ұлттық стандарты* (Р МемСТ) - РФ

Ұлттық стандарттау органымен бекітілген стандарт.

Халықаралық (аймақтық) стандарт - стандарттау жөніндегі халықаралық (аймақтық) ұйыммен бекітілген стандарт.

Ұйым стандарттары (ҰСТ) ұйымдармен өнім сапасын, жұмыстың орындалуын және қызметтің көрсетілуін жетілдіру үшін өз бетінше әзірленеді (және бекітіледі). ҰСТ аталмыш ұйымда қолданылатын өнімге, процестерге және онда көрсетілетін қызметтерге, сондай-ақ ішкі және сыртқы нарыққа арналған өнімге, жұмыстарға, қызметтерге әзірленеді.

Техникалық-экономикалық және әлеуметтік ақпараттың *жалпы ресейлік жіктеуіштері* - мемлекеттік ақпараттық жүйелер және ақпараттық ресурстарды құру кезіндегі қолдану үшін міндетті болып табылатын және оның жіктеуішіне (класы, топтары, түрлері) сәйкес техникалық-экономикалық және әлеуметтік ақпаратты үлестіретін нормативті құжаттар.

Халықаралық стандарттау халықаралық қауымдастықтың барлық елдерінің тиісті органдарының қатысуы үшін ашық стандарттау жөніндегі халықаралық қызмет. Ол тек қана ИСО, ХЭК сияқты ұйымдардың шеңберінде ғана жүзеге асырылып қоймай, сонымен қатар көптеген басқа ұйымдарда (үкіметтік емес және үкіметаралық) жүзеге асырылады.

Ресей Федерациясының аумағында әрекет ететін стандарттау жөніндегі нормативтік құжаттарға келесілер жатады:

- Ресей Федерациясының мемлекеттік стандарттары, белгіленген тәртіп бойынша қолданылатын халықаралық (аймақтық) стандарттар, ережелер, нормалар және стандарттау бойынша ұсыныстар; жалпыресейлік техникалық-экономикалық ақпарат жіктеуіштері;
- сала стандарттары; кәсіпорын стандарттары; ғылыми-техникалық, инженерлік қоғамдардың және басқа қоғамдық бірлестіктердің стандарттары.

Мемлекеттік стандарттар, жалпыресейлік техникалық-экономикалық ақпарат жіктеуіштерінің құрамына кіретіндер:

- өнімдерге, жұмыстарға және қызметтерге олардың қоршаған ортаға, өмірге, денсаулық пен мүлікке қауіпсіздігі бойынша қойылатын талаптар, өрт қауіпсіздігі талаптары, қауіпсіздік техника және өндірістік санитария талаптары;
- техникалық және ақпараттық сәйкестік бойынша, сондай-ақ өнімдердің өзара алмастырушылығы жөніндегі талаптар;
- өнімдердің негізгі тұтынушылық (пайдаланушылық) сипаттамалары,

оларды бақылау әдістері, қаптамасына, таңбалануына, тасымалдануына, сақталуына, қолданылуы мен кідеге жаратылуына қойылатын талаптар;

- өнімді әзірлеу, өндіру, қолдану (пайдалану), жұмыстарды орындау және қызмет көрсету кезіндегі техникалық және ақпараттық тұтастықты қамтамасыз ететін ережелер мен нормалар, соның ішінде техникалық құжаттаманы рәсімдеу ережелері, рұқсаттар мен отырғызулар, өнімнің, жұмыстар мен қызметтердің сапасын қамтамасыз етудің, барлық ресурс түрлерін сақтау және оңтайлы пайдаланудың жалпы ережелері, терминдер және олардың анықтамалары, шартты белгілер, метрологиялық және басқа жалпы техникалық және ұйымдық-техникалық ережелер мен нормалар.

Өнімдер мен қызметтердің мемлекеттік стандарттардың талаптарына сәйкестігі өнімді және қызметтерді мемлекеттік стандарттарға сәйкестігі жөніндегі таңбалау жолымен расталады.

Кәсіпорын стандарттары, оларды талаптарды қамтамасыз ету талаптарына қарай қолдану қажеттілігінен, сондай-ақ ұйымды жетілдендіру және өндірісті басқару мақсатында, кәсіпорындармен өз бетінше әзірлене және растала алады. Кәсіпорын стандарттарының талаптары, егер өнімді әзірлеу, өндіру және жеткізу, жұмыстарды орындау және қызметтер көрсету шартында осы стандарттарға сілтеме көрсетеліген жағдайда, басқа шаруашылық қызметінің субъектілерімен міндетті түрде сақталуға жатады.

Ғылыми-техникалық, инженерлік және басқа қоғамдық бірлестіктердің стандарттары әртүрлі салалардан алынған білімді, нәтижелерді және әзірленімдерді пайдалану және серпімді тарату үшін осы қоғамдық бірлестіктер әзірлейді және қабылдайды. Осы стандарттарды қолдану қажеттілігін шаруашылық қызмет субъектілері өз бетінше анықтайды.

Әлемдік тәжірибедегідей, РФ-да стандарттардың бірнеше түрі қолданылады, олар стандарттау нысанының ерекшелігіне қарай ерекшеленеді. Стандарттардың келесі түрлерін ажыратады:

- негізгі стандарттар (ұйымдастыру-әдістемелік және жалпытехникалық);
- өнім, қызмет стандарттары;
- жұмыс/процесс стандарттары;
- бақылау әдістеріне стандарттар (сынақтар, өлшеулер, талдау).

Негізгі жалпытехникалық стандарттар орнатылады:

- стандарттаудың әртүрлі нысандарына арналған шартты белгілер (атаулары, кодтары, белгілері, символдары және т.б.), олардың

сандық, әріптік-цифрлі белгіленулері, соның ішінде физикалық шамалардың параметрлерінің белгіленуі (орыс, латын, грек әріптерімен), олардың өлшемділігі, алмастырушы жазбалары, символдары және т.с.с.;

- құжаттамалардың әртүрлі түрлерінің (нормативті, конструкторлық, жобалық, технологиялық, бағдарламалық және басқалар) құрылымына, баяндалуына, рәсімделуіне және мазмұнына қойылатын талаптар;
- жалпы техникалық шамалар, өндірістік процесстерді техникалық, соның ішінде метрологиялық қамсыздандыруға қажетті талаптар мен нормалар.

Стандарт бөлімдерінің құрамы, мазмұны және атауын стандартталатын өнімнің ерекшеліктеріне және оған қойылатын талаптардың сипатына қарай анықталады.

Жұмыс/процесске қойылатын стандарттар өнімнің техникалық бірлігін және оңтайлылығын қамтамасыз ететін, оларды әзірлеу, дайындау, сақтау, асымалдау, пайдалану, жөндеу және кәдеге жарату технологиялық процестердегі әртүрлі түрдегі жұмыстарды орындау әдістеріне (тәсілдеріне, амалдарына, режимдеріне, нормаларына) қойылатын негізгі талаптарды белгілейді.

Заманауи кезеңде өнімнің (қызметтің) сапасын қамтамасыз ету жүйесінің шеңберіндегі басқару процестерінің стандарттары үлкен маңыздылыққа ие.

Бақылау әдісінің стандарттары (сынау, өлшеу, талдау) өнімді құру, сертификаттау және пайдалану кезінде, өнімге сынақ, өлшеу, талдау жүргізу әдістерін (тәсілдерін, амалдарын) белгілейді. Осындай стандарттар өнімнің/қызметтің сапасына қойылатын міндетті талаптарды бағалау нәтижелерінің әділдігін, дәлдігін және жаңғыртылуын жоғары дәрежеде қамтамасыз етулері тиіс. Осы шарттарды орындау стандарттағы өлшеу қателіктері туралы мәліметтердің болуына байланысты болады.

Бақылау әдістерінің, тәсілдерінің және амалдарының көп түрлілігіне қарамастан, стандарттауға жататын жалпы ережелерді бөліп көрсетуге болады:

- бақылау құралдары және көмекші құрылғылар;
- бақылауды даярлау және өткізу тәртібі;
- бақылау нәтижелерін өңдеу және рәсімдеу ережелері; сынақтың рұқсат етілген қателігі.

Нәтижелер дұрыс әрі салыстыруға келетіндей, стандарттарда өткізілетін операциялардың және алынған нәтижелерді өңдеудің

реттілігін анықтайтын ережелерге қатысты ұсыныстар беріледі.

Стандарттау саласындағы ең ірі және танымал ұйымдардың бірі - ИСО, Стандарттау бойынша Халықаралық ұйымы (ISO, International Standards Organization). ИСО тауарлар мен қызметтермен алмасуды жеңілдету үшін, сондай-ақ интеллектуалдық, ғылыми, техникалық және экономикалық қызмет саласындағы ынтымақтастықты дамыту үшін бүкіл әлемдегі стандарттауды дамытуда көмек көрсетіп келеді. Кеңес Одағы осы ұйымды құруда белсенді ат салысқан. Ағылшын, орыс, француз тілдері ИСО ресми тілдері болып табылады. ИСО стандарттары құрағыш өнімдердің өзара алмасушылығын, өнім сапасын сынау және бағалаудың бірыңғай әдістерін қоса алғанда, өнімге қойылатын талаптардың бірыңғайлығын қамтамасыз етуге көмектеседі.

Электртехника, электроника, радиобайланыс және аспап жасау саласындағы халықаралық стандарттаумен Электроника жөніндегі Халықаралық комиссия, ХЭК (IEC, International Electrotechnical Commission) айналысады. Бүгінгі күнде ХЭК ИСО құрамындағы дербес ұйым болғандығына қарамастан, ХЭК қызмет саласы ИСО қызмет көрсету саласы болып табылмайды. ХЭК қызметінің мақсаты – халықаралық стандарттарды әзірлеу жолымен халықаралық ынтымақтастыққа көмек көрсету және электртехникалық өнеркәсіп, ядролық аспап жасау, лазерлік техника, байланыс құралдары, авиациялық және ғарыш аспабын жасау, кеме құрылысы және теңіз навигациясы, атом энергетикасы, информатика, акустика, медициналық техника саласында әзірленетін халықаралық стандарттардың негізінде сертификатауды жүргізу.

Соңғы уақытта, халықаралық стандарттау процесінің ең өнімді құрауышы болған, консорциум деңгейіндегі стандарттау ерекше қарқынмен дамып келеді. Осы стандарт әзірлеушілер тобының ең танымал өкілдеріне келесілер мысал болып табылады:

- IEEE (Institute of Electrical and Electronic Engineers - Электртехника және электроника Инженерлер институты) - кәсіби халықаралық ұйым - ақпараттық технологиялар саласындағы бірқатар маңызды халықаралық стандарттарды әзірлеуші;
- OMG (Object Management Group - Нысандарды басқару тобы) - біркелкіленген үлестірілген нысанды бағдарламалық қамсыздандыру құруға арналған стандарттарды әзірлеуді жүзеге асыратын халықаралық консорциум;
- W3C (World Wide Web Consortium) - WWW- технология (HTTP, HTML, URL, XML) стандарттарын және көптеген басқаларды әзірлеуге және дамытуға маманандырылған консорциум.

Стандарттар ақпараттық тезнология әзірлеушілеріне басқа әзірлеушілердің деректерін, бағдарламалық, коммуникациялық құралдарын пайдалану мүмкіншілігін, деректердің экспорт/импортын жүзеге асыруды, ақпараттық технологиялардың әртүрлі компоненттерінің біріктірілуін қамтамасыз етуге мүмкіндік береді.

БАҚЫЛАУ СҰРАҚТАРЫ ТАПСЫРМАЛАРЫ

1. Стандарт дегеніміз не?
2. Стандарттау деп нені атаймыз?
3. Стандарттаудың негізгі міндеттерін тұжырымдаңыз.
4. Стандарттаудың принциптері қандай?
5. Нормативтік құжат деп нені атаймыз?
6. Ресей Федерациясының аумағында қандай нормативтік құжаттар әрекет етеді?
7. Стандарттау деңгейлерін қалай айырады?
8. Стандарттау нысанының ерекшелігіне қарай стандарттардың қандай түрлері бар?
9. Мемлекеттік стандарттардың құрамына нелер кіреді?
10. Халықаралық стандарттардың ұлттық стандарттардан айырмашылығы неде?
11. Стандарттау саласындағы қандай халықаралық ұйым ірі және танымал болып табылады?
12. Қандай ұйым электртехника, электроника, радиобайланыс саласында халықаралық стандарттаумен айналысады?
13. Стандарттаумен айналысатын қандай халықаралық консорциумдар бар?

13-тарау

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ӘЗІРЛЕУДІ СТАНДАРТТАУ ЖӘНЕ ҚҰЖАТТАУ

13.1. БҚБЖ ЖӘНЕ Р МемСТ. ЖАЛПЫ МӘЛІМЕТТЕР

Бағдарламалық құжаттардың бірыңғай жүйесі (БҚБЖ) - бағдарлама және бағдарламалық қамсыздандыруларды әзірлеу, рәсімдеу және айналымының өзара байланысқан ережелерін белгілейтін Ресей Федерациясының мемлекеттік стандарттар жиынтығы.

БҚБЖ стандарттарында бағдарламаларды әзірлеу, сүйемелдеу, дайындау және пайдалануын регламенттейтін талаптар қойылады, бұл келесі мүмкіншіліктерді қамтамасыз етеді:

- бағдарламалармен өзара алмасу және ертеректе әзірленген бағдарламаларды жаңа әзірленімдерде пайдалану үшін бағдарламалық өнімдерді біріздендіру;
- еңбек сыйымдылығын төмендету және бағдарламалық өнімдердің әзірлену, сүйемелдеу, дайындау және пайдалану тиімділігін арттыру;
- бағдарламалық құжаттаманы дайындау және сақтауды автоматтандыру.

БҚБЖ стандарттары 1970 жылдың аяғында қабылданған және бізге бастапқы күйіндегі түрі жақын күйінде жетті. Оларда ведомстволық есептеу орталықтарының жұмыс тәжірибесі көрсетілген, онда үлкен ЭЕМ пайдаланылды. Адамның компьютерлік жүйемен өзара әрекеттесуі ол кезде қазіргідей құрылмаған еді, және бүгінгі уақытта осы стандарттардың кейбіреуі ескерген деп танылды.

МемСТ 19.001-77 сәйкес, БҚБЖ стандарттары 13.1-кестеде келтірілген топтарға бөлінеді.

БҚБЖ нормативтік-техникалық құжаттарының құрамы 13.2-кестеде берілген.

МемСТ 19.101-77 (1626-79). БҚБЖ. Бағдарламалар мен бағдарламалық құжаттардың түрлері (1987 жылдың қарашасында қайта басып шығарылған, № 1 өзгертулерімен, қаңтар 2010 ж. - соңғы ред.) есептеу машиналарына олардың тағайындалуы мен қолдану саласына қарамастан кешендер мен жүйелерге арналған бағдарлама мен бағдарламалық құжаттардың түрлерін белгілейді.

13.1.-кесте. БҚБЖ стандарт топтары	
Топтың коды	Топтың атауы
0	Жалпы ережелер
1	Негізге алынатын стандарттар
2	Әзірлеу құжаттамасын орындау ережелері
3	Дайындау құжаттамасын орындау ережелері
4	Сүйемелдеу құжаттамасын орындау ережелері
5	Пайдалану құжаттамасын орындау ережесі
6	Бағдарламалық құжаттаманың айналым ережесі
7	Резервті топтар
8	
9	Басқа стандарттар

13.2-кесте. БҚБЖ құрамы	
Белгіленуі	Атауы
МемСТ 19.001-77	БҚБЖ. Жалпы ережелер
МемСТ 19.002-80	БҚБЖ. Алгоритмдер мен бағдарламалар сызбалары. Орындау ережелері.
МемСТ 19.004-80	БҚБЖ. Терминдер және анықтамалар
МемСТ 19.005-85	БҚБЖ. Р - Алгоритмдер мен бағдарламалар сызбалары. Шартты графикалық белгіленуі және орындау ережелері
МемСТ 19.101-77	БҚБЖ . Бағдарлама және бағдарламалық құжаттар түрлері
МемСТ 19.102-77	БҚБЖ. Әзірлеу кезеңдері
МемСТ 19.103-77	БҚБЖ. Бағдарлама және бағдарламалық құжаттардың белгіленуі
МемСТ 19.104-78	БҚБЖ. Негізгі жазбалар
МемСТ 19.105-78	БҚБЖ. Бағдарламалық құжаттарға қойылатын жалпы талаптар
МемСТ 19.106-78	БҚБЖ. Баспа түрінде орындалған бағдарламалық құжаттарға қойылатын талаптар
МемСТ 19.201-78	БҚБЖ. Техникалық тапсырма. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.503-79	БҚБЖ. Жүйелі бағдарламашының нұсқаулығы. Мазмұны мен рәсімдеуге қойылатын талаптар

13.2-кестенің соңы

Белгіленуі	Атауы
МемСТ 19.202-78	БҚБЖ. Сипаттама. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.301-79	БҚБЖ. Бағдарлама және сынақ әдістемесі. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.401-78	БҚБЖ. Бағдарлама мәтіні. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.402-78	БҚБЖ. Бағдарламаның сипаттамасы
МемСТ 19.403-79	БҚБЖ. Түпнұсқа ұстаушылар ведомості
МемСТ 19.404-79	БҚБЖ. Түсіндірме жазбасы. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.501-78	БҚБЖ. Формуляр. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.502-78	БҚБЖ. Қолданылудың сипаттамасы. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.503-79	БҚБЖ. Жүйелі бағдарламашының нұсқаулығы. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.504-79	БҚБЖ. Бағдарламашының нұсқаулығы. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.505-79	БҚБЖ. Оператордың нұсқаулығы. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.506-79	БҚБЖ . Тілдің сипаттамасы. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.507-79	БҚБЖ . Пайдаланушылық құжаттардың ведомості
МемСТ 19.508-79	БҚБЖ. Техникалық қызмет көрсету жөніндегі нұсқаулық. Мазмұны мен рәсімдеуге қойылатын талаптар
МемСТ 19.601-78	БҚБЖ. Қосарлау, есепке алу және сақтаудың жалпы ережелері
МемСТ 19.602-78	БҚБЖ. Баспа түрінде орындалған бағдарламалық құжаттарды қосарлау, есепке алу және сақтаудың ережелері
МемСТ 19.603-78	БҚБЖ. Өзгерістер енгізудің жалпы ережелері
МемСТ 19.604-78	БҚБЖ. Баспа түрінде орындалған бағдарламалық құжаттарға өзгерістер енгізу ережелері

Аталмыш стандартта бағдарламалық компонентке және бағдарламалар кешеніне анықтама беріледі.

Компонент - өз бетінше немесе кешеннің құрамында қолданылатын және аяқталған қызметті орындайтын, тұтастай түрінде қарастырылатын бағдарлама.

Кешен - екі немесе одан көп компоненттерден және (немесе) кешеннен тұратын, өзара байланысқан қызметтер атқаратын және жеке түрде немесе басқа кешеннің құрамында қолданылатын бағдарлама.

Бағдарламалық өнімге құжаттамалар түрлері 13.3-кестеде берілген.

Пайдалану құжаттар мен оларға қойылатын талаптардың түрлері 13.4-кестеде келтірілген.

13.3-кесте. Бағдарламалық құжаттардың түрлері

Сипаттамасы	Бағдарламаның құрамы және оған арналған құжаттамалар
Түпнұсқа ұстаушыларының ведомосы	Бағдарламалық құжаттардың түпнұсқаларын сақтайтын кәсіпорындар тізімі
Бағдарлама мәтіні	Қажетті пікірлерімен бағдарламаны жазу
Бағдарламаны сипаттау	Бағдарламаның логикалық құрылымы және қызметі туралы мәліметтер
Бағдарлама және сынақ әдістемесі	Бағдарламаны сынау кезінде тексеруге жататын талаптар, сондай-ақ оларды бақылаудың тәртібі мен әдістері
Техникалық тапсырма	Бағдарламаның тағайындалуы және қолдану аясы, бағдарламаға қойылатын техникалық, техникалық-экономикалық және арнайы талаптар, әзірлеудің қажетті кезеңдері мен мерзімдері, сынақ түрлері
Түсіндірме жазба	Алгоритм сызбасы, алгоритмнің және (немесе) бағдарлама қызметінің жалпы сипаты, сондай-ақ қабылданған техникалық және техникалық-экономикалық шешімдерді негіздемесі
Пайдаланатын құжаттар	Бағдарламаның қызмет етуін және пайдаланылуын қамтамасыз етуге арналған мәліметтер

13.4-кесте. Пайдалану құжаттардың түрлері	
Пайдалану құжаттар ведомосы	Бағдарламалық өнімге пайдаланушылық құжаттар тізімі
Формуляр	Бағдарламаның негізіг сипаттамалары, жиынтықтылығы және бағдарламаны пайдалдану жөніндегі мәліметтер
Қолдануының сипаты	Бағдарламаны тағыйындауы, қолдану саласы, қолданылатын әдәстер, шешілетін тапсырмалар класы, қолдануға арналған шектеулер, техникалық құралдардың минималды конфигурациясы туралы мәліметтер
Желілік бағдарламашының нұсқаулығы	Нақты пайдалану шарттарында бағдарламаны тексеру, қызмет етуін қамтасыз ету және икемдеуге арналған мәліметтер
Бағдарламашының нұсқаулығы	Бағдарламаны пайдалануға арналған мәліметтер
Оператордың нұсқаулығы	Бағдарламаны орындау барысында оператордың есептеу жүйесімен қарым-қатынасқа түсу рәсімін қамтамасыз етуге арналған мәліметтер
Тілді сипаттау	Тілдік синтаксисін және семантикасын сипаттау
Техникалық қызмет көрсету жөніндегі нұсқаулық	Техникалық құралдарға қызмеет көрсету барысында тестілік және диагностикалық бағдарламаларды қолдануға арналған мәліметтер

БҚБЖ басты жетіспеушіліктерінің қатарына келесілерді жатқызуға болады:

- бағдарламалық қамсыздандырудың жалғыз «каскадты» өмірлік циклына бағытталу;
- бағдарламалық қамсыздандырудың сапасының сипаттамасын құжаттау бойынша нақты ұсыныстардың жоқтығы;
- өмірлік цикл және тұтастай алғанда өнімді құжаттау бойынша басқа қолданыстағы отандық стандартар жүйесімен желілік үйлесімділіктің жоқтығы, мысалы, БҚБЖ;
- бағдарламалық қамсыздандыруға тауарлық өнім ретінде құжаттауға нақты айқындалмаған тәсілі;
- халықаралық және аймақтық стандарттардың ұсыныстарымен келісілген, бағдарламалық қамсыздандыруға перспективалық құжаттардың құрамы, мазмұны және рәсімделуі бойынша ұсыныстардың жоқтығы.

БҚБЖ РФ аумағында қолдану тек ұсыныс жасау сипатында, яғни БҚБЖ ерікті негізде қолданылады, егерде өзге шартпен, келісім-шартпен, жеке заңдармен, сот шешімімен және т.с.с. қарастырылмаған жағдайда.

Орыс тіліне аударылған және Ресейде ұлттық құқықта қабылданған кейбір желілік және бағдарламалық инженерия саласындағы ИСО/ХЭК стандарттары БҚБЖ заманауи баламасы болып табылады. Бұдан әрі осы стандарттардың кейбіреуінің тағайындалуы келітірлген.

Р МемСТ ИСО/ХЭК 9294-93. Ақпараттық технология. Бағдарламалық қамсыздандыруды құжаттауды басқару бойынша нұсқаулық. Стандарт толығымен ИСО/ХЭК 9294 халықаралық стандартына сәйкес келеді және олардың құрылуына жауап беретін, басшыларға арналған бағдарламалық қамсыздандыруды құжаттауды тиімді басқару бойынша ұсыныстарды белгілейді. Стандарттың мақсаты келесілерге көмек көрсету болып табылады:

- бағдарламалық қамсыздандыруды құжаттау стратегиясын анықтауды;
- құжаттау бойынша стандарттарды таңдауда;
- құжаттау рәсімдерін таңдауда; қажетті ресурстарды анықтауда;
- құжаттау жоспарларын құруда.

Р МемСТ ИСО/ХЭК 9126-93. Ақпараттық технологиялар. Бағдарламалық өнімді бағалау. Сапа сипаттамалары және оларды пайдалану жөніндегі нұсқаулық. Стандарт толығымен ИСО/ХЭК 9126-93 халықаралық стандартына сәйкес келеді. Оның мәнмәтінінде сапа сипатамасы түсінігінде, «оның сапасы сипатталатын және бағаланатан, бағдарламалық өнімнің «қасиеттер (атрибуттар) жиынтығы» түсіндіріледі.

Р МемСТ ИСО 9127-94. Ақпаратты өңдеу жүйелері. Пайдаланушы құжаттары және тұтынушылар бағдарламалық топтамаларға арналған қаптамадағы ақпарат. Стандарт толығымен ИСО 9127:1989 халықаралық стандартқа сәйкес келеді. Осы стандарттың мәнмәтінінде тұтынушылар бағдарламалық топтамасы (БТ) «белгілі бір функцияларды орындауға жобаланған және сатылатын бағдарламалық өнім; бағдарлама және оған сәйкес біртұтас күйінде сатуға арнап қапталған құжаттама» ретінде түсіндіріледі. Пайдаланушы құжаттамасы ретінде соңғы пайдаланушыны БТ орнату және пайдалану бойынша ақпаратпен қамтамасыз ететін құжаттама түсіндіріледі. Қаптамадағы ақпарат ретінде БТ сыртқы қаптамасында берілген ақпарат түсіндіріледі. Оның мақсаты әлеуетті сатып алушыларға БТ туралы бастапқы мәліметтер беру болып табылады.

Р МемСТ ИСО/ХЭК 8631-94. Ақпараттық технология. Бағдарламалық конструктивтер және оларды таныстыруға арналған

шартты белгілер. Рәсімдік алгоритмдерді таныстыруды сипаттайды.

Р МемСТ ИСО/ХЭК 12119-2000. Ақпараттық технология. Бағдарламалық құралдар топтамасы. Сапаға қойылатын талаптар және сынақтар. Бұл стандартта бағдарламалар топтамасына қойылатын талаптар және оларды берілген талаптарға сәйкестігіне сынақ жөніндегі нұсқаулар белгіленген. «Бағдарламалық құралдар топтамасы» түсінігі, бірнеше пайдаланушыға жалпы пайдалануы немесе қызмет істеуі үшін жеткізілетін бағдарламалар, рәсімдер және ережелердің жиынтығы ретінде қарастырылатын, нақты «бағдарламалық өнім» жалпы түсінігімен теңдестіріледі. Әрбір бағдарламалық топтамасының өнім сипаттамасы және пайдаланушылар құжаттары болуы тиіс.

Р МемСТ ИСО/ХЭК 9127-94. Пайдаланушының құжаттары және тұтынушылар бағдарламалық топтамасына арналған қаптамадағы ақпарат. Стандарт пайдаланушының құжаттарын және тұтынушылар бағдарламалық топтамасы жасақталуы тиіс, қаптамадағы ақпаратты сипаттайды. Пайдаланушының құжаттары пайдаланушыларды бағдарламалық құралдарды орнатуға және өткізуге қажетті ақпаратпен қамтамасыз етеді. Әдетте, бұл құжаттарды, қаптаманың ішіне бағдарламалық құралдармен бірге салатын бір немесе бірнеше нұсқаулық ретінде ұсынады. Нәтижесінде пайдаланушылар топтаманы сатып алмағанша, нұсқаулықты қолдана алмайды.

Пайдаланушы құжаттарының тағайындалымы соңғы пайдаланушыны бағдарламалық құралдың мақсатын, қызметін және сипаттамасын, бағдарламалық құралды қалай іске қосу және пайдалану керектігін анық түсіну үшін жеткілікті ақпаратпен қамтамасыз ету болып табылады. Оқу құжаттарының тағайындалымы ретінде жаңа және тәжірибесіз пайдаланушылар үшін топтаманы біртіндеп жұмысқа енгізу мүмкіндігін беру борлып табылады. Топтаманың сыртындағы қаптамасындағы ақпараттың мақсаты әлеуетті сатып алушыларға аталмыш бағдарламалық құралды олардың қажеттіліктеріне сәйкес қолданулығы туралы шешім қабылдауға мүмкіндік беру болып табылады.

13.2. БАҒДАРЛАМАЛЫҚ ҚҰРАЛДАРДЫҢ ӨМІРЛІК ЦИКЛЫНЫҢ ПРОЦЕСТЕРІ

Стандарт Р МемСТ ИСО/ХЭК 12207-2010 бағдарламалық құралдардың, бағдарламалық индустрияда бейімделетін, өмірлік циклының процестерінің жалпы құрылымын белгілейді. Бұл стандарт бағдарламалық өнімді немесе қызметтерді сатып алу кезінде, сондай-ақ бағдарламылық өнімді жеткізуде, әзірлеуде, тағайындалуы бойынша пайдалануда, сүйемелдеу және пайдалануды тоқтатқан кезде,

пайдаланылатын процестерді, қызмет түрлері мен міндеттерді, анықтайды.

Стандарт жүйені, бағдарламалық өнімдерді сатып алу және тиісті қызметтерді көрсету кезінде қолданылады; сондай-ақ ұйымда, және одан тыс жерде бағдарламалық-аппараттық құралдардың бағдарламалық өнімдерін және бағдарламалық-аппараттық құралдарын жеткізуде, әзірлеуде, пайдалану және сүйемелдеуде қолданылады.

Стандарттың құрамында, сондай-ақ бағдарламалық өнімдер мен қызметтердің мәнін түсінуді қамтамасыз етуге қажетті жүйені сипаттаудың қырлары қамтылған.

Стандарт тараптардың екі жақты қатынасында қолданыла алады, тіпті егер екі тарап та бір ұйымға тиесілі, сонымен бірге бір тараптың өзін-өзі бақылауы үшін болғандығына қарамастан.

Стандарт келесілерге арналған:

- жүйелер, бағдарламалық өнімдер және қызметтерге тапсырыс берушілер үшін;
- жеткізушілерге, әзірлеушілерге, операторларға;
- сүйемелдеу қызметкерлеріне;
- жоба әкімшілеріне;
- сапаға жауап беретін әкімшілерге;
- бағдарламалық өнім пайдаланушыларына.

Стандарт, шығарылатын құжаттың атауын, форматтарын немесе толық мазмұнын анықтауға арналмаған. Бұл сұрақтардың шешімі осы стандартты пайдаланушылардың қарастыруына қалдырылған.

Стандарт бағдарламалық құралдың өмірлік циклының нақты моделін немесе әзірлеу әдісін алдын-ала анықтамайды. Осы стандартты қолданатын пайдаланушылар, өзінің бағдарламалық жобасына қолдануға келетін өмірлік циклының моделін өздері таңдайды және осы стандарттан алынған процестерді, жұмыстарды және міндеттерді аталмыш модельге үлестіреді; бағдарламалық құралдарды әзірлеудің әдістерін таңдай алады және қолданады және нақты бағдарламалық жобаға сәйкес келетін жұмыстарды және міндеттерді орындайды.

Нақты ұйым, өз мақсаттарына байланысты, өздерінің нақты міндеттерін орындау үшін тиісті процестердің ішкі жиынтығын таңдай алады. Сол себептен бұл стандартты нақты ұйымға, жобаға немесе қосымшаға бейімдеген жөн.

Р МемСТ ИСО/ХЭК 12207-2010 бағдарламалық құралдардың жоғарғы деңгейінің түпкі ойынан кәдеге жаратуға дейінгі сәулетін белгілейді. Сәулет аталмыш процестердің арасындағы көптеген процестер мен өзара байланыстардан тұрады. Негізінде, әрбір процесс өмірлік циклдағы бірегей функцияны іске асыруға арналған және мамандандырылған қызметті орындау үшін басқа процесті жұмылдыра

алады.

Р МемСТ ИСО/ХЭК 12207-2010 әрбір процесс тараптардың жауапкершілігі (міндеттемелері) көзқарасынан қарастырылған. Ұйым бір немесе бірнеше процестерді орындай алады. Процесс бір немесе бірнеше ұйымдармен орындала алады, сонымен бірге ұйымдардың бірі жауапты тарап ретінде анықталуы тиіс.

Өмірлік цикл сәулетіндегі жауапкершілік принципі, нақты жоба үшін Р МемСТ ИСО/ХЭК 12207-2010 қолданбалы пайдалануын жеңілдетеді, онда көптеген тұлғалар жұмылдырыла алады.

Процестер үш жалпы класқа топтастырылған (1.1-суретті қараңыз):

1) негізгілер;
2) көмекшілер;
3) ұйымдастырушылық.

Тәжірибеде әрбір процесс оны құрайтын жұмыстардың терминдерімен анықталуы тиіс, олардың әрқайсысы оларды құрайтын міндеттер терминдерімен анықталуы тиіс. Процестегі жұмыс байланысқан міндеттер жиынтығынан тұрады. Р МемСТ ИСО/ХЭК 12207-2010 көптеген процестер, жұмыстар және міндеттер белгіленген. Процестер, жұмыстар және міндеттер жалпы табиғи позициялық жүйелілікпен сипатталған. Бұл жүйелілік өмірлік циклының моделін іске асырудың жүйелілігін алдын-ала анықтамайды. Сипатталған жүйелілік, бағдарламалық құралдарды құру жобасында, жобаға тиесілі немесе оған лайықты процестерді, жұмыстарды (қызмет түрлерін) және мәндеттерді (тапсырмалар) таңдауға, ретке келтіруге қолдануға және қайталауға арналған.

Р МемСТ ИСО/ХЭК 12207-2010 бағдарламалық құралдың өмірлік циклын тегіс қамтитын біріктірілген процестер жиынтығына қойылатын талаптарды белгілейді. Аталмыш стандарт әрбір процесс үшін жетілдендіру процесі арқылы «жоспар - іске асыру - тексеру - акт» циклыне рұқсатты қамтамасыз етеді. Сонымен бірге, сапамен байланысты және бағдарламалық құралдың өмірлік циклының ажырамас бөлігі деп түсіндірілетін жұмыстар, өмірлік циклдың тиісті процестеріне кіреді. Осылайша, оны іске асыруға жауап беретін әрбір процесс және қызметшінің артында, осы процестің шеңберіндегі, сапамен байланысты жұмыстар бекітілген.

Р МемСТ ИСО/ХЭК 12207-2010 келесілерге қолдануға болады:

- өмірлік циклдың кез келген модель(дер)іне (мысалы, каскадты, инкрементті немесе эволюциялық);
- бағдарламалық инженерияның кез келген әдістеріне немесе технологияларына (мысалы, нысанға бағытталған жобалау, құрылымдық бағдарламалау, төменге бағытталған тестілеу немесе макеттеу);

■ кез келген бағдарламалау тілдеріне.

Аталмыш сұрақтардың шешімі жобаның өзіне және технологиялардың заманауи жағдайына байланысты, ал осы элементтерді таңдауды Р МемСТ ИСО/ХЭК 12207-2010 пайдаланушысы жүзеге асырады.

Стандарт жалпы көзқарас бойынша икемді болып табылады, яғни өмірлік циклы процесінің жұмыстары (қызмет түрлері) және міндеттері (тапсырмалары) «қалай істеу керек?» деген емес «не істеу керек?» деген сұраққа жауап береді. Басқаша айтқанда, міндеті «сәулеттік жобаны әзірлеу және құжат жүзінде рәсімдеу» болуы мүмкін, бірақ «сәулеттік жобаны бәсеңдейтін функционалдық жобалау әдісін қолдана отырып әзірлеу немесе құжат жүзінде рәсімдеу» емес. Аталмыш сызба тапсырыс берушіге ақырғы өнімге немесе қызметке талаптарды орнату үшін кең мүмкіндіктер береді және сол мезетте сатушыға өнімді құруға немесе қызмет көрсетуге арналған тиісті әдістерді, тәсілдерді және құралдарды әзірлеуге және қолдануға мүмкіндік береді.

Р МемСТ ИСО/ХЭК 12207-2010 құжатта саласындағы стандарт болып табылмайды, яғни, егер тіпті көрсетілген стандартта процестердің кейбір шығыс нәтижелерін құжаттауға талаптар белгіленсе де, ол құжаттардың форматын немесе мазмұнын анықтамайды. Аталмыш стандарт жоспар, сипаттамалар (техникалық тапсырмалар) немесе тестулеуге қойылатын талаптар сияқты ұқсас шығыс нәтижелерін қалай біріктіруді анықтамайды.

Р МемСТ ИСО/ХЭК 12207-2010 көрсеткіштер (метрикалар) мен нұсқаушылар нақты жүйесінің терминдерінде бағдарламалық құралдардың қасиеттерін (атрибуттарын) (сүйемелдеудің сенімділігі мен ыңғайлылығы секілді) анықтамайды және тапсырмайды. Бұл стандарт бағдарламалық құралдың ұқсас қасиеттерін анықтауға арналған тәсілдерді сипаттайды, бірақ олар Р МемСТ ИСО/ХЭК 12207-2010 пайдаланушыларымен нақтылануы тиіс.

Р МемСТ ИСО/ХЭК 12207-2010 қатаң түрде жүйенің бағдарламалық қамсыздандырусын жобалаудың жүйелендірілген басқаруын алмастырмайды. Көрсетілген стандарт құрылымды көрсетеді, онда бағдарламалық құралмен байланысты процестер, жұмыстар және міндеттер, тиісті түрде анықталуы, жоспарлануы және орындалуы мүмкін. Р МемСТ ИСО/ХЭК 12207-2010 нақты анықталған конструктивті блоктар (процестер) жиынтығын қамтиды. Стандарт пайдаланушысы, аталмыш блоктарды өз ұйымының және жобасының мақсаттарына және міндеттеріне сәйкес таңдауы, тәжірибе жүзінде қолдануы және жиынтықтауы тиіс.

Бағдарламалық құралдарды құрудың нақты жобаларын іске асыру жағдайларында, Р МемСТ ИСО/ МЭК 12207-2010 тәжірибе жүзінде

пайдалану бойынша ұсыныстар Р МемСТ ИСО/ХЭК ТО 15271 қамтылған. Апараттық технология. Р МемСТ ИСО/ХЭК 12207-2010 пайдалану бойынша нұсқаулық.

13.3. ТЕХНИКАЛЫҚ ТАПСЫРМА. МАЗМҰНЫНА ҚОЙЫЛАТЫН ТАЛАПТАР

Техникалық тапсырма бағдарламалық қамсыздандыруға қойылатын талаптар жиынтығын қамтиды және әзірленген бағдарламаны тексеру және қабылдау белгісі ретінде қолданыла алады, сол себептен жеткілікті түрде толық құрастырылған (қосымша бөлімдерді енгізу мүмкіншілігін ескере отырып) және тапсырыс берушімен және әзірлеушімен қабылданған техникалық тапсырма жобаның негізін қалаушы құжаттардың бірі болып табылады. Бағдарламалық өнімді әзірлеуге техникалық тапсырманы сауатты құру мүмкіншілігі бағдарламашының кәсіби деңгейін анықтайды және оны тапсырыс берушінің тарапынан негізсіз наразылықтардан арылтады.

Техникалық тапсырма - бұл әзірлеудің негізгі мақсаттары, бағдарламалық өнімге қойылатын талаптар қалыптастырылатын, әзірлеудің мерзімі мен кезеңдері анықталатын және қабылдау-тапсыру сынақтар процесі регламенттелетін құжат. Бұл құжаттың негізінде тапсырыс берушінің бастапқы талаптары, жобаалды зерттеулерді орындау нәтижелері және т.с.с. жатыр.

Техникалық тапсырманы әзірлеу келесі жүйелілікте орындалады.

Ең алдымен орындалатын қызметтер жиынтығы, сондай-ақ бастапқы деректердің тізімі мен сипаттамалары белгіленеді. Содан кейін нәтижелер тізімі, олардың сипаттамалары мен оларды таныстыру тәсілдерін белгілейді.

Бұдан әрі, бағдарламалық қамсыздандырудың қызмет ету ортасы, яғни: нақты жиынтығын және техникалық құралдар параметрлерін, қолданылатын операциялық жүйенің нұсқасын, және болашақ бағдарламалық өнімге өзара әрекеттесуі мүмкін, басқа орнатылған бағдарламалық қамсыздандырудың нұсқалары нақтыланады.

Әзірленіп отырған бағдарламалық қамсыздандыру қайсыбір ақпаратты жинаған және сақтаған немесе қайсыбір техникалық процесті басқаруға қосылған жағдайда, сондай-ақ жабдық немесе энергиямен жабдықтауда іркіліс болған жағдайда бағдарламаның әрекетін нақты регламенттеу қажет.

Әзірленіп отырған бағдарламалық қамсыздандырудың сипаттамаларын анықтайтын негізгі факторлар бар.

Оларды атап көрсетейік:

- бағдарламаның немесе жүйенің қызметін белгілейтін бастапқы деректер және талап етілетін нәтижелер;
- әзірленіп отырған бағдарламалық қамсыздандыру қызмет етуді мүмкін орта (бағдарламалық және аппараттық) берілуі мүмкін, немесе техникалық тапсырмада көрсетілген параметрлерді қамтамасыз ету үшін таңдап алынуы мүмкін;
- басқа бағдарламалық қамсыздандырумен және/немесе нақты техникалық құралдарымен болжалды әрекеттесу - сондай-ақ анықталуы мүмкін, немесе орындалатын қызметтер жиынтығынан таңдап алынуы мүмкін.

МемСТ 19.106-78. Бағдарламалық құжаттаманың бірыңғай жүйесі. *Баспа түрінде орындалған бағдарламалық құжаттарға қойылатын талаптар*, есептеу машиналарына, кешендер және жүйелерге арналған бағдарламаны немесе бағдарламалық өнімді әзірлеуге, олардың тағайындалуына және қолдану саласына қарамастан техникалық тапсырма құру және рәсімдеу тәртібін белгілейді.

Стандартқа сәйкес «Техникалық тапсырма» құжаты бөлімдерден тұруы тиіс. Оларды атап өтейік:

1) кіріспе;
2) әзрлеу негізі;
3) әзірлеудің тағайындалымы;
4) бағдарламаға немесе бағдарламалық өнімге қойылатын талаптар;
5) бағдарламалық құжаттамаға қойылатын талаптар;
6) техникалық-экономикалық көрсеткіштер;
7) әзірлеудің кезеңдері мен сатылары;
8) бақылау және қабылдау тәртібі;
9) қосымшалар (қажет болған жағдайда).

Бағдарламалық қамсыздандырудың ерекшелігіне қарай, бөлімдердің мазмұнын нақтылауға, жаңа бөлімдер енгізуге немесе олардың кейбіреулерін біріктіруге рұқсат беріледі.

«Кіріспе» бөлімінде әзірленетін бағдарламалық өнімнің мақсаты, бағдарламалық қамсыздандыруды және бағдарламалық қамсыздандыру қолданылатын нысанды қолдану саласының қысқаша сипаттамасы көрсетіледі.

«Әзірлеу негіздері» бөлімінде келесілер көрсетілуі тиіс:

- әзірлеу негізінде жүргізілетін құжат (құжаттар);
- осы құжатты бекіткен ұйым және оның бекітілген күні;
- әзірлеу тақырыбының атауы және/немесе шартты белгілері. «Әзірлеу тағайындалымы» бөлімінде бағдарламалық қамсыздандырудың функционалдық және пайдланушылық тағайындалуы көрсетілуі тиіс.

«Бағдарламаға немесе бағдарламалық қамсыздандыруға қойылатын талаптар» бөлімі келесі қосымша бөлімдерді қамтуы тиіс:

- функционалдық сипаттамаларға қойылатын талаптар;
- сенімділікке қойылатын талаптар;
- пайдалану шарттары;
- техникалық құралдардың құрамына және параметрлеріне қойылатын талаптар;
- ақпараттық және бағдарламалық сәйкестікке қойылатын талаптар;
- таңбалауға және қаптамаға қойылатын талаптар;
- тасымалдау және сақтауға қойылатын талаптар;
- арнайы талаптар.

«Функционалдық сипаттамаларға қойылатын талаптар» қосымша бөлімінде атқарылатын қызметтердің құрамына, кіріс және шығыс деректерді ұйымдастыруға, уақыт сипаттамаларына және т.б. қойылатын талаптар көрсетілуі тиіс. Кіріс деректеріне қойылатын талаптарды сипаттау кезінде кіріс деректерінің сипаты, ұйымдастыруы және алдын-ала дайындалуы, форматы кіріс деректерін кодтаудың сипаты және тәсілі көрсетілуі тиіс.

Бағдарламаның кіріс ақпараты бастапқы құжаттар (жүккұжаттар, есептер және т.б.), нормативтік-анықтамалық ақпарат (анықтамалар, жіктеуіштер, кодтамалар, және т.б.), электронды құжаттар, кіріс сигналдары дәне т.б.болуы мүмкін.

Бағдарламаның шығыс ақпараты құжаттар (электронды немесе қағаз), деректер файлдары, шығыс сигналдары және т.б. болуы мүмкін. Шығыс деректерге қойылатын талаптарды сипаттаған кезде шығыс деректердің сипаты, ұйымдастырылуы, шығыс деректерінің форматы, сипаттамасы және кодтау тәсілі көрсетіледі.

Негізгі функциялардан басқа, техникалық тапсырмада бағдарламаның сервистік қызметіне қойылатын талаптар сипатталады, айта кетсек, жүйені икемдеуді (конфигурациялау) түзету мүмкіншілігі, деректерді резервті сақтау мүмкіншілігі, жүйеге кіру құпиясөзінің өзгеру мүмкіндігі, бағдарламадан шықпастан, күнтізбені, калькуляторды, редакторды және т.б. шақырту мүмкіндігі.

«Сенімділікке қойылатын талаптар» қосымша бөлімінде сенімді қызмет көрсетуді қамтамасыз етуге қойылатын талаптар көрсетілуі тиіс (тұрақты қызмет көрсетуді қамтамасыз ету, кіріс және шығыс ақпаратты бақылау, істен шыққаннан кейін қалпына келтіру уақыты және т.с.с.).

«Пайдалану шарттары» қосымша бөлімінде пайдалану шарттары көрсетілуі тиіс (қоршаған ауа температурасы, деректер тасымалдағышының таңдап алынған түрлерінің салыстырмалы ылғалдылығы және т.с.с.), олар кезінде белгіленген сипаттамалар қамтамасыз етілуі тиіс, сондай-ақ қызмет көрсету түрі, қызметшілердің

қажетті мөлшері және біліктілігі.

«Техникалық құралдардың құрамына және параметрлеріне қойылатын талаптар» қосымша бөлімінде, техникалық құралдардың негізгі техникалық сипаттамаларын көрсете отырып, техникалық құралдардың қажетті құрамын көрсетеді.

«Ақпараттық және бағдарламалық сәйкестікке қойылатын талаптар» қосымша бөлімінде кіріс және шығысындағы және шешу әдістерінің ақпараттық құрылымдарына, бастапқы кодтарға, бағдарламамен қолданылатын бағдарламалау тіліне және бағдарламалық құралдарға қойылатын талаптар, әзірленетін бағдарламалық өнім қызмет етуі мүмкін операциялық жүйелер және орталарға қойылатын талаптар, компьютерге бағдарламалар топтамасын - қосымшаларды әзірлеу құралдарын (аталмыш бағдарламалық өнімды өзгерту, түрлендіру немесе пайдалануға арналған) орнату қажеттілігі, әртүрлі графикалық компоненттердің инсталляциялау қажеттілігі және т.б. көрсетілуі тиіс.

Қажет болған жағдайда ақпараттарды және бағдарламаларды қорғау қамтамасыз етілуі тиіс.

«Таңбалауға және қаптауға қойылатын талаптар» қосымша бөлімінде, жалпы жағдайда, бағдарламалық өнімді таңбалауға қойлатын талаптар, қаптау тәсілдері мен нұсқалары көрсетіледі.

«Тасымалдау және сақтауға қойылатын талаптар» қосымша бөлімінде, бағдарламалық өнім үшін тасымалдау шарттары, сақтайтын орындар, сақтау шарттары, қоймалау шарттары, сақтау мерзімдері және әртүрлі шарттар көрсетілуі тиіс.

«Бағдарламалық құжаттарға қойылатын талаптар» бөлімінде бағдарламалық құжаттаманың алдын-ала құрамы және қажет болған жағдайда оларға қойылатын арнайы талаптар көрсетілуі тиіс.

«Техникалық-экономикалық көрсеткіштер» бөлімінде: бағдарлы экономикалық тиімділігі, болжалды жылдық мұқтаждық, әзірленімнің үздік отандық және шетелдік үлгілерімен немесе аналогтарымен салыстырғанда, экономикалық артықшылықтары көрсетілуі тиіс.

«Әзірленімнің кезеңдерімен сатылары» бөлімінде жұмыстарды әзірлеудің, кезеңдерінің және мазмұнының қажетті кезеңдері (әзірленуі, келісілуі және бекітілуі тиіс бағдарламалық құжаттардың тізімі), сондай-ақ әдеттегідей, әзірленім мерзімі белгіленеді және орындаушыларды анықтайды.

«Бақылау және қабылдау тәртібі» бөлімінде сынақ түрлері және жұмысты қабылдауға қойылатын жалпы талаптар көрсетілуі тиіс.

Техникалық тапсырманың қосымшаларында қажет болған жағдайда беріледі:

- әзірленімді негіздейтін ғылыми-зерттеу және басқа жұмыстардың тізімі;

- алгоритмдер сызбасы, кестелер, сипаттамалар, негіздемелер, есептеулер және әзірлеу кезінде қолданылуы мүмкін басқа құжаттар;
- әзірлеудің басқа дереккөздері.

13.4. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ ТАЛАПТАР СИПАТТАМАСЫ

Кез келген бағдарламалық қамсыздандыруды әзірлеу болашақ бағдарламалық өнімге қойлатын талаптарды талдаудан басталады. Талдау нәтижесінде әзірленіп жатқан БҚ талаптар сипаттамасы алынуы тиіс, ол үшін шешілетін тапсырмалардың декомпозициясын және мазмұнды қойылымын орындайды, олардың өзара әрекеттесуін және пайдаланушылық шектеулерін нақтылайды. Тұтастай алғанда, сипаттамаларды анықтау барысында, шынайы өмірдің қайсыбір бөлігі ретінде пәндік саланың жалпы моделін құрады, онымен әзерленіп жатқан бағдарламалық қамсыздандыру қайсыбір жолмен өзара әрекеттесетін болады, және оның негізгі қызметтерін нақтылайды.

Бағдарламалық қамсыздандырудың талаптарының сипаттамасы - әзірленуі қажет бағдарламаның жай-күйінің аяқталған сипаттамасы.

IEEE 830. Бағдарламалық қамсыздандыруға қойылатын талаптардың сипатамаларын құру әдістемесі (Электротехника және электроника инженерлері институтымен ұсынылған) стандартында бағдарламаға дұрыс құрылған талаптар сипаттамасының сапалық сипаты және мазмұны сипатталады (Software Requirements Specification, SRS) және бірнеше SRS үлгілері беріледі. Аталмыш ұсынылатын әдістеменің әзірленіп жатқан бағдарламалық қамсыздандыруға талаптар қою мақсаты бар, бірақ сондай-ақ өзіндік және бағдарламалық өнімдерді таңдау кезінде көмектесуге де қолданыла алады.

Бұл әдістеме бағдарламалық қамсыздандыруға қойылатын талаптардың сипаттамасын құрудың ұсынылған принциптерін сипаттайды. Ол бағдарламалық қамсыздандырудың сипатамасы процесінің нәтижесі бір мағыналы және толық сипаттамалық құжат болып табылатын модельдерге негізделген. Ол бағдарламалық қамсыздандырудың тапсырыс берушілеріне олардың нақты не алғысы келетіндігін сипаттап беруге көмектесуге, ал бағдарламалық қамсыздандыруды жеткізушілерге тапсырыс берушінің ненің нақты қалайтындығын түсінуге арналған. Бұл әдістеме, олардың жеке ұйымдары үшін бағдарламалық қамсыздандырудың стандартты сипаттамасының макетін (SRS) әзірлеуге, бағдарламалық қамсыздандыруға қойылатын

талаптардың нақты сипаттамаларының форматын және мазмұнын белгілеуге мүмкіндік береді.

Сапалы құрылған сипаттама тапсырыс берушілерге, жеткізушілерге және басқа тұлғаларға келесі мүмкіндіктерді береді:

- тапсырыс берушілер мен жеткізушілер арасындағы бағдарламалық өнімдер қандай қызмет атқаруы керек деген мәселеге қатысты келісімге келуіне арналған негізді құру. SRS келтірілген бағдарламалық қамсыздандыру қызметінің толық сипаты, әлеуетті пайдаланушыларға бағдарламалық қамсыздандыру олардың қажеттіліктерін қанағаттандырады ма, әлде осы қажеттіліктерді қанағаттандыру үшін бағдарламалық қамсыздандыруды қалай өзгерту керектігін анықтауға көмектеседі;
- әзірлеу бойынша жұмыстар көлемін азайту. SRS дайындау тапсырыс берушінің әртүрлі қатысушы топтарын, жобаны орындауға кіріспестен бұрын барлық талаптарды қатаң түрде қарастыруға мәжбүрлейді, және кейінгі қайта жобалау, кодтау және тестілеуді қысқартады. SRS-те көрсетілген талаптарды мұқият талдау, әзірлеу кезеңінде жіберіп алған қателіктерді, бұрыс түсінулерді және қарама-қацшылықтарды, оларды түзету оңай болған кезде, аша алады;
- шығындар мен жоспарларды бағалауға арналған негізді қамтамасыз ету. SRS-пен сәйкес әзірленетін бағдарламаны сипаттау, жобаға жұмсалған шығындарды бағалауға арналған тәжірибелік негіз болып табылады және тапсырыс беруші ұйымдағы жобаның бюджетін келісу үшін немесе бағаны жеткізушінің бағалауына пайдаланыла алады ;
- дұрыстығын және анықтығын тексеруге арналған негізді қамтамасыз ету. Ұйымдар, сапалы әзірленген SRS пайдаланған кезде, дұрыстығын және анықтығын тексеру жоспарын тиімдірек құра алады. Әзірлеу келісім-шартының бір бөлігі ретінде, SRS сәйкестікті анықтау үшін негізді қамтамасыз етеді;
- табыстауды жеңілдету. SRS бағдарламалық өнімді жаңа пайдаланушыларға табыстауды немесе оның жаңа машиналарға орнатылуын қарапайым етеді. Осылайша, тапсырыс берушілер бағдарламалық қамсыздандыруды олардың ұйымындағы басқа бөлімшелерге қарапайым жолмен табыстау алады, ал жеткізушілер үшін оны жаңа тапсырыс берушілерге табыстау оңай болады;
- SRS-те ол әзірленген жоба емес, өнім талқыланатын болғандықтан, SRS дайын өнімді кейінгі өрістету үшін негіз қызметін атқарады. SRS бірқатар өзгерістерді талап етуі мүмкін, алайда дегенмен өнімді үздіксіз бағалауға негіз ұсына алады.

Дұрыс құралған сипаттама (SRS) келесідей болуы тиіс:

- дұрыс;
- бір мағыналы;
- толық;
- қайшылықсыз;
- өзінің мәнділігі және/немесе төзімділігі бойынша ретке келтірілген;
- тексерілетін;
- түрлендірілетін;
- қадағаланатын.

SRS, егер онда баяндалған әрбір талап, бағдарламалық қамсыздандыру қанағаттандыруы тиіс талап болып табылған жағдайда *дұрыс* болып табылады. Дұрыстыққа кепілдік беретін қайсыбір құрал немесе рәсім жоқ. SRS, оның басқалармен келісімділігін қамтамасыз ету үшін, жүйеге қойылатын талаптар сипаттамасы сияқты, жоғары дәрежелі кез келген қолданылатын сипаттамамен, жобаның басқа құжаттамасымен және басқа қолданылатын стандарттармен салыстырылуы тиіс. Балама ретінде тапсырыс беруші немесе пайдаланушы, SRS-тің нақты мұқтаждықтарды дұрыс көрсететіндігін анықтай алады. Қадағалаушылық бұл рәсімді қарапайым және қателерге аз бейімделгіш етеді.

SRS, тек егер онда баяндалған талаптардың бір ғана түсіндірмесі болған жағдайда *бір мағыналы* болып табылады. Бұл үшін кем дегенде, соңғы өнімнің әрбір қасиеті бірегей сөйлем арқылы сипатталуын талап етеді. Арнайы мәнмәтінде қолданылатын терминнің көп мағынасы болған жағдайда, бұл термин глоссарийге енгізілуі тиіс, онда оның мәні нақтырақ сипатталады. SRS бағдарламалық қамсыздандырудың өмірлік циклындегі талаптарды құру процесінің маңызды бөлігі болып табылады және жобаны жобалау, іске асыру, мониторингтеуде, дұрыстығын тексеру және анықтауда, сондай-ақ оқыту кезінде пайдаланылады. SRS оны құрушыларға және оны пайдаланушыларға бір мағыналы болуы тиіс.

SRS, егер оған келесі элементтер қосылса ғана *толық* болып табылады:

- барлық маңызды талаптар: олардың функционалдық мүмкіншіліктеріне, жұмыс сипаттамаларына, жобалық шектеулеріне, атрибуттарға немесе сыртқы интерфейстеріне жататындығына қарамастан. Жекелеген жағдайларда, жүйенің сипаттамасымен (бағдарлама оның бір бөлігі болады) салынатын кез келген сыртқы талаптары расталуы және өңделуі тиіс;
- кез келген ықтимал жағдайларда іске асырылуы мүмкін кіріс деректерінің барлық кластарына бағдарламалық қамсыздандырудың жауабын анықтау. Жауаптарды жол берілетін, және жол берілмейтін кіріс міндеті деп анықтау маңызды екендігін ескерген жөн;

■ SRS-гі барлық суреттер, кестелер және сызбаларға толық белгілеулер және сілтемелер және барлық теминдер мен өлшем бірліктерінің анықтамалары. *Қайшылықсыз* ішкі қайшылыққа келмеушілікті білдіреді. Егер SRS, желілік талаптар сипаттамасы (System Requirements Specification, SyRS) сияқты, белгілі бір басқа жоғары деңгейлі құжатпен келісілмесе, онда ол бұрыс болып табылады. SRS іштей қайшылықсыз болып табылады, тек егер онда сипатталған жекелеген талаптардың ешбір ішкі жиынтығы кереғарлықта болмаған жағдайда.

SRS, егер ондағы әрбір нақты талаптың мәнділігін немесе төзімділігін көрсететін идентификатор болған жағдайда, *өзінің мәнділігі және/немесе төзімділігі бойынша ретке келтірілген болып табылады*. Әдеттегідей, бағдарламалық өнімге қатысты барлық талаптар, бірдей дәрежеде маңызды болып табылмайды. Кейбір талаптар, әсіресе қосымшалар үшін, адамның өмірі тәуелді болатын ажырамас болып табылады, сол мезетте басқалары тек қалаулы бола алады. SRS-гі әрбір талап, осы айырмашылықтарды нақты және анықтау үшін сәйкестендірілуі тиіс. Талаптарды келесі түрде сәйкестендіру мына тұлғаларға көмектеседі:

■ тапсырыс берушілерге: әрбір талапты мұқият қарап шығып, мүмкін болатын жасырын қателіктерді анықтауға мүмкіндік береді;
■ әзірлеушілерге: дұрыс жобалық шешімдер қабылдауға және бағдарламалық өнімнің әртүрлі құрауыштарына тиісті күш жұмсауға мүмкіндік береді.

SRS, егер ондағы баяндалған әрбір талап тексерілуі мүмкін болған жағдайда *тексерілетін* болып табылады. Талап тексерілетін болып табылады, тек егер қайсыбір ақырғы экономикалық тиімді (рентабельді) процесс бар болған жағдайда, оны пайдалана отырып пайдаланушы немесе машина бағдарламалық қамсыздандыру осы талапты қанағаттандыратындығына көз жеткізе алады.

Жалпы алғанда, кез келген бір мағыналы емес талап тексеруге келмейді. Тексерілмейтін талаптар «жақсы жұмыс істейді», «жақсы пайдаланушы интерфейсі» және «әдетте орын алуы тиіс» түбіндегі тұжырымдарды қамтиды. Бұл талаптар тексеріле алмайды, себебі «жақсы», «тәуір» немесе «әдетте» терминдерін анықтау мүмкін емес. «Бағдарлама ешқашан шырғаланбауы тиіс» деген пікір тексерілмейтін болып табылады, себебі осы қасиетті тестілеу теориялық тұрғыдан мүмкін емес.

SRS *түрлендірілетін* болып табылады, егер оның құрылымы және стилі осындай болса, талаптардың кез келгені құрылымын және стилін сақтай отырып оңай, толық және қайшылықсыз түрде орындалуы мүмкін, SRS:

■ мазмұнымен, алфавитті көрсеткішпен және анық берілген тоғыспалы

сілтемелерімен байланысты және пайдалануға жеңіл құрылымды иеленуі үшін;

- артық болмауы үшін (яғни, бірдей талап SRS-те бір жерден артық жерде пайда болмауы тиіс);
- әрбір талапты басқа талаптармен араластырмай, жеке көрсететіндей.

Артықшылық өз бетінше қате болып табылмайды, бірақ ол қателіктің пайда болуына оңай алып келуі мүмкін. Кей кездері артықшылық SRS-ті оқылатындай етуі мүмкін, алайда артықшыл құжатты жаңарту кезінде проблема пайда болуы мүмкін. Мысалы, талап тек ол пайда болған жерлердің біреуінде ғана өзгертілуі мүмкін. Ол жағдайда SRS қайшылықты болады. Артықшылық қажет болған кез келген кезде, түрлендірмелі ету үшін SRS анық тоғыспалы сілтемелерді іске қосуы тиіс.

SRS, егер оның талаптарының әрқайсысы дереккөзі нақты бақыланып отырған кезде және егер ол кейінгі құжатты әзірлеуде немесе түрлендіруде талаптардың әрқайсысына сілтеме жасауға мүмкіндік беретін болса, *қадағаланатын* болып табылады. Қадағаланушылықтың келесі екі түрі ұсынылады:

1) *кері қадағаланушылық* (яғни, алдыңғы әзірлеу кезеңдеріне). Әрбір талап, оның ертеректегі құжаттардағы дереккөзіне сілтеме жасайтындығына байланысты.;
2) *тікелей қадағаланушылық* (яғни, SRS әзірленетін барлық құжаттарға). SRS-тен алынған әрбір талаптың бір мағыналы анықталған атауы немесе сілтеме нөмірлікі бар-жоғына байланысты. SRS тікелей қадағаланушылығы, бағдарламалық өнім пайдалану және сүйемелдеу кезеңіне енген кезде ерекше маңызды болады. Кодтары мен сәулеттік құжаттарының өзгеруіне қарай, осы өзгерістер әсер етуі мүмкін талаптардың толық жиынтығын анықтау мүмкіндігін иелену қажет.

IEEE 830 стандартымен ұсынылатын SRS құрылымы:

1. Кіріспе.
1.1. Тағайындалуы.
1.2. Әрекет ету саласы.
1.3. Анықтамалар, акронимдер және қысқартулар.
1.4. Жарияланымдар.
1.5. Қысқа шолу.
2. Толық сипаттамасы.
2.1. Өнімнің мәнмәтіні.
2.2. Өнімнің қызметі.
2.3. Пайдаланушының сипаттамасы.
2.4. Шектеулер.
2.5. Жорамалдар және тәуелделіктер.

2.6. Арнайы талаптар.

3. Талаптардың сипаттамалары.

Қосымшалар.

Алфавиттік көрсеткіш.

SRS-ке «кіріспе» тұтастай SRS-ке қысқаша шолу түрінде болуы тиіс. «Толық сипаттамасы» бөлімі бағдарламалық қамсыздандыру әсер ететін жалпы факторларды және оларға қойылатын талаптарды сипаттауы тиіс. Бұл бөлім нақты талаптарды белгілемейді. Оның орнына ол SRS-нің 3-бөлімінде толық анықталған және оларды түсінуге қарапайымдырақ ететін талаптар туралы алдын-ала мәліметтерді қамтамасыз етеді.

«Өнімнің мәнмәтіні» қосалқы бөлімі өнімді онымен байланысқан басқа өнімдермен мәнмәтінінде сипаттауы тиіс. Егер өнім тәуелсіз және толығымен дербес болып табылса, онда бұл құжатта анық түсіндірілуі тиіс. Егер SRS, әдетте жиі кездесетін, үлкен жүйенің компоненті болып табылатын өнімді анықтаса, онда бұл қосалқы бөлім үлкен жүйенің талаптарының бағдарламалық қамсыздандырудың функционалдық мүмкіндіктерімен байланысын орнатуы, және осы жүйе мен бағдарламалық қамсыздандыру арасындағы интерфейсті сәйкестендіруі тиіс. Үлкен жүйенің, байланыстардың және сыртқы интерфейстердің басты компоненттерін блок-сызбаларды пайдалану пайдалы болуы мүмкін. Бұл қосалқы бөлім сондай-ақ бағдарламалық қамсыздандырудың әртүрлі шектеулердің шеңберінде қалай қызмет ететіндігін сипаттауы тиіс.

«Өнімнің қызметі» - SRS-тің бұл қосалқы бөлімі, бағдарламалық қамсыздандыру орындайтын негізгі қызметтердің мәліметін білдіруі тиіс. Анықтықты қамтамасыз ету мақсатында:

- қызметтер, қызметтер тізімін тапсырыс берушіге немесе құжатты бірінші рет оқып отырған кез келген адамға түсінікті ететін тәсілмен ұйымдастырылуы тиіс;
- әртүрлі қызметтерді және олардың байланыстарын көрсету үшін мәтіндік немесе графикалық әдістер қолданыла алады.

SRS-тің «Пайдаланушының сипаттамасы» қосалқы бөлімі, білімділік деңгейін, тәжірибесін және арнайы техникалық білімдерді қамтып, өнімді пайдаланушылардың жалпы сипаттамаларын сипаттап беруі тиіс.

«Шектеулер» - SRS-тің бұл қосалқы бөлімі әзірлеушінің опцияларын шектейтін кез келген басқа позициялардың жалпы сипатын қамтамасыз етуі тиіс. Олар аппараттық шектеулерді, басқа қосымшалармен интерфейстерді, қатарлас операцияларды, бақылау функциясын, басқару функциясын және т.б. қамтиды.

«Жорамалдар және тәуелділіктер» - SRS-тің бұл қосалқы бөлімінде SRS-те құрастырылған талаптарға әсер ететін барлық факторлар атап өтілуі тиіс. Бұл факторлар бағдарламалық қамсыздандырудың сәулеттік

шектеулері болып табылмайды, бұл ең дұрысы, олардың SRS-гі талаптарға әсер ете алатын кез келген өзгерістері.

«Талаптарды үлестіру» қосалқы бөлімі жүйенің болашақ нұсқалары пайда болғанша кейінге қалдырылуы мүмкін талаптарды сәйкестендіруі тиіс.

«Арнайы талаптар» - SRS-тің бұл бөлімі бағдарламалық қамсыздандыруға қойылатын барлық талаптарды, жобалаушыларға осы талаптарды қанағаттандыратындай жүйені әзірлеуге мүмкіндік беретіндей, ал сынаушыларға - жүйе осы талаптарды қанағаттандыратындығына көз жеткізетіндей талдап тексеру деңгейінде қамтуы тиіс. Осы бөлімде әрбір тұжырымдалған талап пайдаланушылармен, операторлармен немесе басқа сыртқы жүйелермен қабылдануы тиіс. Бұл талаптар кем дегенде, жүйеге әрбір кіріс әсерін (ынтасын), жүйенің әрбір шығыс сигналын (пікірін) және жүйемен кіріс әсеріне жауап ретінде орындалатын немесе шығыс сигналын қолдау үшін барлық функцияларды қамтуы тиіс. Арнайы талаптар барлық сипаттамаларға сәйкес тұжырымдалуы тиіс. Барлық талаптар бірдей сәйкестендірілуі тиіс.

Бұл бөлімде сыртқы интерфейстері - бағдарламалық қамсыздандырудың барлық кіріс және шығыс деректері егжей-тегжейлі сипаты сипатталуы тиіс. Функционалдық талаптар, кіріс деректерін өңдеу кезінде және кіріс деректерін өңдеу және кіріс деректерін генерациялау кезінде бағдарламалық қамсыздандырумен орындалуы тиіс іргелі операцияларды анықтауы тиіс.

Сондай-ақ, бағдарламалық қамсыздандыруға немесе пайдаланушының бағдарламалық қамсыздандырумен толықтай әрекет етуіне қойылатын статистикалық және динамикалық сандық сипаттамаларға талаптар анықталуы тиіс. Жекелеген қосалқы бөлімінде дерекқорда орналастырылуы тиіс кез келген ақпараттың логикасына қойылатын талаптарды анықтаған жөн.

Барлық жүйеге арнап белгіленген қолжетімділіктің деңгейін қамтамасыз ету үшін қажетті факторлар анықталуы тиіс, атап айтқанда: бақылау нүктесі, қалыпқа келтіру және қайта іске қосу, сондай-ақ бағдарламалық қамсыздандыруды кездейсоқ немесе арам ниетті кіруден, пайдаланудан, бұзудан немесе ашудан қорғайтын факторлар. Жинақылыққа - бағдарламалық қамсыздандыруды басқа машиналарға және/немесе операциялық жүйеге ауыстырудың қарапайымдылығына жататын бағдарламалық қамсыздандырудың атрибуттарына қойылатын талаптарды сипаттау қажет.

«Қосымшалар» нақты SRS бөлігі ретінде үнемі қарастырылмайды және үнемі қажет емес. Олар мыналарды қамтуы мүмкін:

- кіріс/шығыс типтік форматтарын, өзіндік құн калькуляциясын

зерттеудің сипаты немесе пайдаланушылық сауалдардың нәтижелерін;
- SRS оқырмандарына көмектесуі мүмкін қосымша немесе алдын-ала ақпаратты;
- Бағдарламалық қамсыздандырумен шешілуі тиіс проблеманың сипаттамасын;
- Қорғаныс, экспорт, бастапқы жүктеу немесе басқа талаптарға сәйкестігін қамтамасыз ететін кодтар мен тасымалдағыштарға арналған арнайы командаларды.

SRS-те қосымшаны іске қосу кезінде, бұл қосымшалардың талаптардың бір бөлігі деп саналуы тиіс пе, әлде жоқ па анық тұжырымдау қажет.

13.5. БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫҢ ҚҰЖАТТАУЫН БАСҚАРУ

Р МемСТ ИСО/ХЭК 9294-93. Ақпараттық технология. Бағдарламалық қамсыздандыруды құжаттауды басқару бойынша нұсқаулық. Стандарт, бағдарламалық қамсыздандырудың немесе бағдарламалық өнімнің өндірісіне жауап беретін жетекшілерге арналған бағдарламалық қамсыздандыруды құжаттау бойынша нұсқаулықты білдіреді. Аталмыш стандарт, жетекшілердің өздері бағдарламалық қамсыздандыруды тиімді басқаруы үшін айналысуы тиіс стратегияларды, стандарттарды, рәсімдерді, ресурстар мен жоспарларды анықтауға бағытталған.

Нұсқаулық бағдарламалық қамсыздандырудың барлық типтеріне - қарапайым бағдарламалардан ең күрделі бағдарламаға немесе бағдарламалық қамсыздандыру жүйесіне дейін қолдануға арналған. Бағдарламалық қамсыздандырудың барлық өмірлік циклының кезеңдеріне жататын бағдарламалық қамсыздандырудың барлық типтері қамтылған.

Бағдарламалық қамсыздандыруды құжаттауды басқарудың принциптері кез келген жоба көлеміне бірдей. Шағын жобалар үшін осы стандартта келітірілген қосымшалардың маңызды бөлігін қолданбауға болады, бірақ принциптері сол күйінде қала береді. Жетекшілер өздерінің нақты тұтынушылары үшін аталмыш ұсыныстарды бейімдей алады.

Бағдарламалық құралдарды қолданудың өспелі масштабы және олардың күрделілігі, осы құралдарға толық, нақты және түсінікті

құжаттардың пайдаланушыларға қолжетімді болу қажеттілігін туындатады. Құжаттама жиі нақты бағдарламаны құрғаннан кейінгі қайсыбір ретінде қарастырылады. Дегенмен, бағдарламалық құжаттаманы әзірлеудің сапасы көзқарасынан, ол аталмыш бағдарламалық қамсыздандыруды құру процесінің ажырамас бөлігі ретінде қарастырылуы тиіс.

Осы проблеманы тиісті түрде қарастырған кезде, құжаттаманы құру және басқару процесінде жеткілікті күрделі жұмыс талап етіледі

13.6. ПАЙДАЛАНУШЫНЫҢ ҚҰЖАТТАМАСЫН ҚҰРУ ПРОЦЕСІ

Пайдаланушының интерфейсі бар бағдарламалық құралға, барлық түрдегі пайдаланушының құжаттамасын құрудың минималды қажетті процесі, *Р МемСТ ИСО/ХЭК 15910* стандартымен белгіленеді.. *Бағдарламалық құралды пайдаланушының құжаттамасын құру процесі.* Пайдаланушының құжаттамасына баспа құжаттаманы (мысалы, пайдаланушының нұсқаулығы және қысқаша анықтамалық карталар), диалогты (оперативті) құжаттаманы, анықтамалық мәтін және диалогты құжаттама жүйесін, сондай-ақ жеткізудің анықтамалық жүйелерін, анықтамалық ақпараттарды және т.б. жатады.

Егер бағдарламалық құралдың әзірленуін сапаны бақылау стандартына сәйкес құжаттайтын болса, аталмыш стандарттың ережелерін теңдей мөлшерде әзірленімнің өзіне, сондай-ақ тиісті құжаттамаға да қолданады.

Тапсырыс беруші құжаттаманы әзірлеушіге келесіге рұқсатты қамтамасыз етуі тиіс:

- барлық тиісті сипаттамаларға, жазба форматтарына, экрандар және есептер құрастырымына, бағдарламалауды автоматтандыру құралдарының жұмыстарының шығыс нәтижелеріне және құжаттаманы дайындауға қажетті басқа ақпаратқа;
- бағдарламалық құралдың жұмыс көшірмесіне (қажет болған жағдайда);
- талдаушыларға және бағдарламашыларға, құжаттаманы әзірлеушінің қызметкерінде пайда болатын мәселелерді уақытында дұрыс шешуді қоса алғанда;
- аудиторияны талдау және тиімділікке тестілеу үшін қарапайым пайдаланушыларға (мүмкіндігі бойынша).

Құжаттаманы әзірлеушінің міндетіне, тапсырыс берушінің кем дегенде шығарылатын өнімді және оған сәйкес келетін аудиторияны

түсінуге кепілдік беретін бағдарламалық құралдарын әзірлеушілермен жасалған келісім-шарттардың табыстылығын қамтамсыз ету кіреді.

Құжаттаманы әзірлеуші бір мезетте бағдарламалық құралдың әзірлеушісі болып табылатындығына қарамастан, тапсырыс беруші оны барлық қолданылатын стандарттардың көшірмелерімен, стильдері мен форматтары бойынша нұсқаулықтармен, сондай-ақ тиісті материалдармен (егер олар жалпыға қолжетімді болса) қамтамасыз етуі тиіс.

Құжаттаманы әзірлеуші құжаттау жоспарын дайындауы тиіс, онда нақты құжаттаманы құру кезінде орындалатын тапсырмалар анықталуы тиіс. Аталмыш жоспар тапсырыс берушімен ресми түрде келісілуі тиіс, бұл тапсырыс берушінің барлық талаптарының бұл жоспардағы толық есебін растайды.

13.7. БАҒДАРЛАМАЛЫҚ ӨНІМДІ БАҒАЛАУ

Р МемСТ ИСО/ХЭК 9126-93 СТАНДАРТЫНЫҢ ТАҒАЙЫНДАЛУЫ

Р МемСТ ИСО/ХЭК 9126-93 стандарты. Бағдарламалық өнімді бағалау. Сапа сипаттамалары және оларды пайдалану бойынша нұсқаулық алты сипаттаманы белгілейді, олар минималды қосарлануымен бағдарламалық қамсыздандырудың сапасын сипаттайды. Аталмыш сипаттамалар бағдарламалық қамсыздандыруды кейінгі нақтылау және сапасын сипаттау үшін негіз болады. Нұсқаулықтар бағдарламалық қамсыздандырудың сапасын бағалау үшін сапаның сипаттамасын пайдаланылуын сипаттайды.

Стандарт өлшеу, саралау және бағалаудың көрсеткіштерін және әдістерін белгілемейді.

Осы стандартта берілген сипаттамаларды анықтау және сапаны бағалау процесінің тиісті моделі, бағдарламалық өнімге арналған талаптар анықталған кезде қолдануға болады және өмірлік циклының барысындағы оның сапасы бағаланады. Бұл сипаттамалар, ЭЕМ бағдарламаларын және бағдарламалық-техникалық құралдарға кіретін деректерді (кіріктірілген бағдарламалар) қоса алғанда, бағдарламалық қамсыздандырудың кез келген түріне қолданыла алады.

Стандарт, бағдарламалық қамсыздандыруды сатып алу, әзірлеу, пайдалану, қолдау, сүйемелдеу немесе тексеруге байланысты сипаттамаларға арналған.

Р МемСТ ИСО/ХЭК 9126-93 сәйкес бағдарламалық қамсыздандырудың сапасын бағалау

Р МемСТ ИСО/ХЭК 9126-93 стандарытна сәйкес бағдарламалық қамсыздандырудың сапасы төменде берілген сипаттамалармен бағалануы мүмкін.

Функционалдық мүмкіншіліктер - функциялар жиынтығы және олардың нақты қасиеттеріне қатысты атрибуттар жиынтығы. Функциялары орнатылған немесе болжалды қажеттіліктерді іске асыратындар болып табылады.

Атрибуттардың аталмыш жиынтығы бағдарламалық қамсыздандыру қажеттіліктерін қанағаттандыру үшін орындайтынды сипаттайды, ал басқа жиынтықтар, ең бастысы, оның қалай және қашан орындалатындығын сипаттайды.

Сенімділік - бағдарламалық қамсыздандырудың, белгіленген уақыт кезеңіндегі белгіленген шарттарда өзінің қызмет ету сапасының деңгейін сақтау қабілетіне қатысты атрибуттар жиынтығы.

Бағдарламалық қамсыздандырудың тозуы немесе ескіруі болмайды. Сенімділікті шектеулер талаптардағы, жобадағы және іске асырудағы қателіктерден пайда болады. Қателіктерге байланысты бас тартулар бағдарламалық қамсыздандыруды пайдалану тәсіліне және бағдарламаның ертеректе таңдап алынған нұсқаларына байланысты болады.

Қолайлылық - пайдаланушылар ортасымен анықталған немесе жорамалданған, пайдалану үшін және осындай пайдалануды жеке бағалау үшін қажетті жұмыс көлеміне қатысты атрибуттар жиынтығы.

Қолайлылық бағдарламалық қамсыздандыруға әсер ете алатын, пайдаланушымен пайдалану шарттарының барлық әртүрлілігімен, пайдалануға даярлық пен нәтижелерді бағалауды қоса алғанда, қарастырылуы тиіс. Осы стандартта бағдарламалық өнімнің нақты атрибуты ретінде анықталған қолайлылық, анықтамадан эргономика көзқарасынан ерекшеленеді, онда тиімділік және тиімсіздік сияқты, басқа сипаттамалар қолайлылығынаң құрамдас бөліктері ретінде қарастырылады.

Тиімділік - бағдарламалық қамсыздандырудың қызмет ету сапасының деңгейі мен белгіленген шарттардағы пайдаланылатын ресурстар көлемінің арасындағы қатынасқа жататын атрибуттар жиынтығы.

Ресурстар басқа бағдарламалық өнімдерді, техникалық құралдарды, материалдарды (мысалы, басып шығаруға арналған қағаз, жұмсақ

дискілер) және пайдалануға енгізу, сүйемелдеуші және қызмет көрсетуші қызметкерлердің қызметтерін қамтуы мүмкін.

Сүйемелденушілік - нақты өлшемдерді (түрлендірулерді) жүргізуге қажетті жұмыстар көлеміне жатқызылатын атрибуттар жиынтығы.

Өзгертуге түзетулер, жетілдендірулер немесе бағдарламалық қамсыздандыруды қоршаған жағдайдың өзгерістеріне, талаптарға және қызмет ету шарттарына бейімдеулері кіреді.

Жинақылық - бағдарламалық қамсыздандырудың бір ортадан екіншісіне ауыстырылу қабілеттілігіне қатысты атрибуттар жиынтығы. Қоршаған жағдай ұйымдастырушылық, техникалық немесе бағдарламалық ортаны қоса алады.

Р МемСТ ИСО/ХЭК 9126-93 сәйкес бағдарламалық қамсыздандырудың сапасипаттамасын қолдану

Р МемСТ ИСО/МЭК 9126-93 стандарты бағдарламалық қамсыздандырудың сапасына қойылатын талаптарды белгілеу және бағдарламалық өнімдерді бағалау (өлшеу, саралау және бағалау) үшін қолданылады, келесілерді қоса алғанда:

- бағдарламалық өнімнің сапасына қойылатын талаптарды анықтау;
- сапа талаптары әзірлеу барысында қанағаттандырылуын бақылау кезінде, бағдарламалық қамсыздандыруға қойылатын техникалық талаптарды бағалау;
- енгізілген бағдарламалық қамсыздандырудың белгілері мен қасиеттерін (атрибуттарын) сипаттау (мысалы, пайдаланушының нұсқаулығында);
- жеткізудің алдында әзірленген бағдарламалық қамсыздандыруды бағалау;
- қабылдау алдында бағдарламалық қамсыздандыруды бағалау.

Осы стандартта сипатталған сипаттамаларға арналған бірнеше жалпы қабылданған метрикалар бар. Стандарттау жөніндегі ұйымдар және топтар өзінің жеке бағалау процесінің моделін белгілей алады. Сондай-ақ, БҚ қолдану саласы мен өмірлік циклының кезеңдерін қамту үшін осы сипаттамалармен байланысты метрикаларды құрастыру және тексерудің өзіндік әдістерін қолдана алады. Тиісті метрикалар болмаған және әзірлене алмаған жағдайда, кейде сөздік сипаттауларды және «шамалас әдістерді» пайдаланады.

Аталып өткен сапа сипаттамаларын пайдалану кезінде де, нақты аталмыш ұйымға немесе аталмыш қолдануға немесе біріншісіне және

екіншісіне де белгілер орнату қажет.

Бағалау нәтижелерімен алмасқан кезде, сапаны бағалауға қолданылатын метрикалар мен белгілер орнатылуы тиіс.

Бағдарламалық қамсыздандыруды жіктеудің жалпы қабылданған жүйесі болмаса да, бағдарламалық қамсыздандырудың жалпы қабылданған кластары бар. Әрбір сапа сипатамасының маңыздылығы бағдарламалық қамсыздандырудың класына қарай өзгереді. Мысалы, сенімділік - сыни жүйелердің бағдарламалық қамсыздандыруы үшін ең маңызды, тиімділік - нақты уақыт жүйесінің бағдарламалық қамсыздандыруына ең маңызды, ал қолайлылық - соңғы пайдаланушы диалогының бағдарламалық қамсыздандыруына ең маңызды болып табылады.

Әрбір сапа сипаттамасының маңыздылығы қабылданған көзқарастарға қарай өзгеріп отырады.

Сапа туралы бірнеше түсініктер бар:

- пайдаланушының ұсынысы;
- әзірлеушінің ұсынысы;
- бақарушының ұсынысы.

Пайдаланушының көзқарасы бойынша, сапа - бұл, белгіленген және жорамал қажеттіліктерді қанағаттандыру қабілеттілігіне қатысты, нысан сипаттамаларының жиынтығы.

Құру процесі пайдаланушыдан және әзірлеушіден бағдарламалық қамсыздандырудың бірдей сапа сипаттамаларын пайдалануды талап етеді, себебі олар талаптар мен қабылдауды орнату үшін қолданылады. Сатуға арналған бағдарламалық қамсыздандыру әзірлеген кезде, сапа талаптарында болжалданған қажеттіліктер көрсетілуі тиіс.

Әзірлеушілер, сапа талаптарын қанағаттандыруы тиіс бағдарламалық қамсыздандырудың құрылуына жауап беретін болғандықтан, олар ақырғы өнімнің сапасына сияқты, аралық өнімнің сапасына мүдделі. Әзірлеу циклының әрбір фазасындағы аралық өнімнің сапасын бағалау үшін, әзірлеушілер бірдей сипаттамаларға әртүрлі метрикаларды пайдаланулары тиіс, себебі бірдей метрикалар өмірлік циклдың барлық фазалары үшін қолдануға келмейді. Мысалы, әзірлеуші жобалық сипаттамада бағдар ұзындығы және күту және рұқсат уақыты терминдер пайаланған кезде, пайдаланушының реакция уақытының терминдеріндегі тиімділікті түсінуі. Өнімнің сыртқы интерфейсі үшін қолдануға болатын метрикалар, оның құрылымына қолдануға болатын метрикалармен алмастыруға болады. Пайдаланушының ұсынысы, бағдарламалық қамсыздандыруды сүйемелдеушімен талап етілетін сапа сипаттамалары туралы ұсынысты қамтуы тиіс.

Жетекші нақты сапа сипаттамасына қарағанда, жалпы сапаға

мүдделі болуы мүмкін, және сол себептен жеке сипаттамаларға арналған коммерциялық талаптарды бейнелейтін мәндердің маңыздылығын анықтауды қажет етеді. Жетекшіге, жоспарлы іркіліс немесе құнның артық шығындалуы сияқты, басқарушылық белгілері бар сапаны жоғарылатуды салыстыру қажет болуы мүмкін, себебі ол сапаны шектеулі құн, еңбек ресурстары және белгіленген уақыт шегінде оңтайландыруды қалайды.

Бағалау процесінің моделі

Р МемСТ ИСО/ХЭК 9126-93 стандартына сәйкес бағдарламалық қамсыздандырудың сапасын бағалау процесі үш кезеңнен тұрады:

1) сапаға талаптар орнату (белгілеу);
2) бағалауға даярлық;
3) бағалау процедурасы.

Аталмыш процесс, бағдарламалық қамсыздандырудың әрбір компоненті үшін өмірлік циклдың кез келген қолайлы фазасында қолданыла алады.

Бастапқы кезеңнің мақсаты *сапа сипаттамасы терминдерінде талаптар және болжалды кешенді көрсеткіштер белгілеу* болып табылады. Талаптар қарастырылып отырған бағдарламалық өнімге арналған сыртқы ортаның қажеттіліктерін көрсетеді және әзірлеуді бастауға дейін белгіленуі тиіс. Бағдарламалық өнім негізгі компоненттерге бөлінетін болғандықтан, өнімге қойылатын талаптар, жалпы алғанда, жекелеген компоненттерге қойылатын талаптардан өзгеше болуы мүмкін.

Екінші кезеңнің мақсаты *бағалауға арналған негізді дайындау* болып табылады. Бұл кезең метрикаларды таңдау, саралау деңгейін анықтау және бағалау белгілерін белгілеуден тұрады.

Сапа (көрсеткіштерінің) метрикаларын таңдау. Сапа сипаттамалары анықталған тәсіл, олардың тікелей өлшеуіне жол бермейді. Метрикаларды (көрсеткіштерді) орнату қажеттілігі болады, олар бағдарламалық өнімнің сипаттамаларымен салыстырылады. Әрбір сандық белгі және әрбір бағдарламалық қамсыздандырудың, сипаттамамен салысытырылатын оның ортасымен сандық бағаланатын өзара әрекеттесуі, метрика (көрсеткіш) ретінде қабылдануы мүмкін.

Метрикалар ортасына және олар пайдаланылатын, әзірлеу процесінің фазаларына әртүрлі байланысты болуы мүмкін. Әзірлеу барысында қолданылатын метрикалар, пайдаланушының сәйкес метрикаларымен салыстырылуы тиіс, себебін пайдаланушының ұсынысындағы метрикалар шешуші болып табылады.

Саралау деңгейлерін белгілеу. Сандық белгілері сапа метрикасын пайдалану арқылы өлшенуі мүмкін. Нәтижесі, яғни өлшенген мәні, талаптарды қанағаттандыру деңгейін көрсетпейді. Бұл мақсатта шкала мәліметтері, талаптарды қанағаттандырудың әртүрлі дәрежелеріне сәйкес келетін диапазондарға бөлінулері тиіс. Сапа нақты қажеттілікке жатқызылғандықтан, саралаудың жалпы деңгейлері мүмкін емес. Олар әрбір нақты бағалауға белгіленулері тиіс.

Бағалау белгілерін белгілеу. Өнімнің сапасын анықтау үшін әртүрлі сипаттамалардың бағалау нәтижелері алынуы тиіс. Бағалаушы ол үшін, мысалы, шешімдер кестесін немесе орта есеппен өлшенген кестелерді пайдалана отырып, процедураларды даярлау тиіс. Процедура әдетте, уақыт және құн сияқты басқа қырларды қамтиды, олар бағдарламалық өнімнің сапасын нақты пайдалану шарттарында бағалауға септігін тигізеді.

Бағалау процесі моделінің соңғы кезеңі *(бағалау процедурасы)* «өлшеу», «саралау» және «бағалау» деп аталатын үш кезеңдермен нақтыланады.

Өлшеу үшін таңдап алынған метрикалар бағдарламалық өнімге қолданылады. Нәтижесі метрика масштабындағы мәндер болып табылады.

Саралау кезеңінде өлшенген мәнге арналған саралау деңгейі белгіленеді.

Бағалау бағдарламалық қамсыздандыруды бағалау процесінің соңғы кезеңі болып табылады, онда белгіленген деңгейлердің көпшілігі қорытындыланады. Нәтижесі бағдарламалық өнімнің сапасы туралы қорытынды болып табылады. Содан кейін қорытылған сапа, уақыт және құн сияқты басқа факторлармен салыстырылады.

Басшылықтың ақырғы шешімі басқарушылық белгісінің негізінде қабылданады. Нәтижесі, қабылдау немесе ақауға шығару, немесе бағдарламалық өнімді шығару немесе шығармау бойынша басшылықтың шешімі болып табылады.

БАҚЫЛАУ СҰРАҚТАРЫ ЖӘНЕ ТАПСЫРМАЛАРЫ

1. БҚБЖ нормативтік-техникалық құжаттарының құрамы қандай?
2. БҚБЖ негізгі жетіспеушіліктері қандай?
3. Р МемСТ ИСО/ХЭК 12207 стандартының мағынасы неде?
4. МемСТ 19.106-78 «Техникалық тапсырма» құжатының қандай бөлімдері бар?
5. Бағдарламалық қамсыздандырудың талаптар сипаттамасы дегеніміз не?
6. IEEE 830 стандарты нені сипаттайды. Бағдарламалық қамсыздандыруға қойылатын талаптардың сипаттамасын құру әдістемесі?
7. Сапалы түрде құрылған сипаттама тапсырыс берушілерге, жеткізушілерге және басқа тұлғаларға қандай мүмкіншіліктер береді?
8. Құрылған сипаттаманың дұрыстығы нені білдіреді?
9. Қандай жағдайларда сипаттама (SRS) толық болып табылады?
10. Қандай жағдайларда сипаттама (SRS) түрлендірілетін болып табылады?
11. IEEE 830 стандартымен ұсынылған SRS құрамына не кіреді? Оның бөлімдерінің құрамы қандай?
12. Р МемСТ ИСО/ХЭК 15910 стандарты нені сипаттайды?
13. Р МемСТ ИСО/ХЭК 9126 стандартының мағынасы неде?
14. Бағдарламалық қамсыздандырудың сапасы қандай сипаттамалармен бағалана алады?
15. Пайдаланушының көзқарасынан бағдарламалық қамсыздандырудың сапасы дегеніміз нені білдіреді?
16. Бағдарламалық қамсыздандырудың сапасын бағалау процесі қандай кезеңдерден тұрады? Әрбір кезеңнің мақсаттары қандай?

14-тарау

БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ СЕРТИФИКАТТАУ

14.1. СЕРТИФИКАТТАУДЫҢ НОРМАТИВТІК-ҚҰҚЫҚТЫҚ НЕГІЗДЕРІ

«Сертификаттау» сөзі латын тілінен аударғанда (sertifico) «растаймын, куәландырамын» дегенді білдіреді. Өз мағынасы бойынша сертификаттау, қарастырылып отырған нысанның белгілі бір талаптарға сәйкестігін белгілеуге және растауға бағытталған қызметін білдіреді.

Жалпы айтқанда, ***сертификаттау*** дегенде тауарлар немесе қызметтердің белгілі бір талаптарға сәйкестігін тәуелсіз растауын түсіну қабылданған.

Сертификаттау рәсімін орындау нәтижесі ***сәйкестік сертификаты*** болып табылады.

Бүгінгі уақытта Ресей Федерациясындағы өнімдер мен тауарларды сертификаттаудың жалпы құқықтық негіздері № 184-ФЗ «Техникалық реттеу туралы» Федералды заңымен реттеледі, мұнда мемлекеттік басқару органдарының, сондай-ақ дайындаушылардың (сатушылардың, орындаушылардың) және сертификаттаудың басқа қатысушыларының сертификаттау саласындағы құқықтар мен жауапкершіліктер анықталған.

Жалпы жағдайда сертификаттау келесі мақсаттарда өткізіледі:

- Ресей Федерациясының бірыңғай тауарлы нарығындағы кәсіпорындардың, мекемелердің, ұйымдардың және кәсіпкерлердің қызметі үшін, сондай-ақ халықаралық экономикалық, ғылыми-техникалық ынтымақтастыққа және халықаралық саудаға қатысу үшін жағдайлар жасау;
- тұтынушыларға өнімді құзыретті таңдауда көмектесу;
- тұтынушыны дайындаушының (сатушының, орындаушының) арам ниеттілігінен қорғау;
- өнімнің қоршаған ортаға, өмірге, денсаулыққа және мүлікке қауіпсіздігін бақылау;
- дайындаушымен мәлімделген, өнім сапасының көрсеткіштерін

растау.

Құралдар мен ақпараттандыру жүйелерін сертификаттау Ресей Федерациясындағы жалпы сертификаттау жүйесінің элементі болып табылады.

Бағдарламалық қамсыздандыруды, ақпараттық технологияларды және қызметтерді сертификаттаудың басты мақсаттары:

- ақпараттандыру құралдары мен жүйелерінің пайдаланушыларын құралдар мен жүйелерді, соның ішінде өмірге, денсаулыққа, мүлікке, сондай-ақ қоршаған ортаға қауіп төндіретін импорттық сатып алудан қорғау;
- жүйе әзірлеушілерін және осы шынайы ақпарат жүйелерін пайдаланушылардың кең ортасын, отандық және шетелдік ақпараттандыру, телекоммуникация, ақпараттық технологиялар және қызметтер құралдар нарығының жай-күйі туралы шынайы ақпаратпен қамтамасыз ету;
- мемлекеттік ақпараттандыру жүйелерінің арасындағы ақпараттық алмасуды қамтамасыз ету (салық қызметі, құқық қорғау органдары, еңбек және жұмысты басқару қызметі, білім беру, денсаулық сақтау және т.б.);
- мемлекеттік емес керек-жарақтар субъектілерін мемлекеттік керек-жарақтар субъектілерімен ақпараттық өзара әрекеттесуі үшін жағдаймен қамтамасыз ету;
- ғылыми-техникалық деңгейдің және отандық ақпараттандыру жүйесінің, ақпараттық технологиялар мен қызметтердің бәсекеге қабілеттілігінің артуына көмек көрсету;
- Ресейдің әлемдік ақпарат кеңістігіне кіруі үшін жағдай жасауға көмектесу.

Ақпараттандыру құралдарын сертификаттау тек қана тұтынушының қажеттіліктерін қанағаттандыруды қамтамасыз етіп қана қоймай, сонымен қатар өнім дайындаушыға (жеткізушіге) белгілі бір пайдалар алып келетіндігін ескерген жөн. Осылайша, жеке алғанда, сертификаттау өткізу нарығының кеңеюіне көмектеседі және бәсекелестердің өнімдерімен салыстыра отырып, фирманың өнімінің сапасының расталуын қамтамасыз етеді. Сауда өзара байланыстығын ұйымдастыру көзқарасынан, сертификаттау өнім өндірушілері (жеткізушілері) мен тұтынушыларының арасында сенімгерлік қарым-қатынастың құрылуына көмектеседі. Отандық ақпараттық өнімнің нақты түрлерінің және ақпарат құралдарының нысанды расталған сапасы және қолданыстағы орны ғана оларды бәсекеге қабілетті ете алады және оларға ұсынысты шынайы қамтамасыз ететіндігін есте сақтаған жөн.

№ 184-ФЗ «Техникалық реттеу туралы» Федералды заңында,

стандарттау және сертификаттау саласындағы келесі негізгі терминдер мен олардың анықтамалары қабылданған.

Аккредиттеу - аккредиттеу органымен, жеке немесе заңды тұлғаның сәйкестікті бағалаудың белгілі саласында жұмыс атқаруға құзыреттілігін ресми түрде мойындау.

Қауіпсіздік - азаматтардың өміріне немесе денсаулығына, жеке немесе заңды тұлғалардың мүлкіне, мелекеттік немесе муниципалды мүлікке, қоршаған ортаға, жануарлар мен өсімдіктердің өмірі немесе денсаулығына нұқсан келтірумен байланысты жол берілмейтін тәуекел жоқ жағдай.

Сәйкестікті декларациялау - өнімнің техникалық регламенттердің талаптарына сәйкестігін растау нысаны .

Сәйкестік туралы декларация - айналымға шығарылатын өнімнің техникалық регламент талаптарына сәйкестігін куәландыратын құжат.

Өтініш беруші - сәйкестікті міндетті түрде растауды жүзеге асырушы жеке немесе заңды тұлға.

Сәйкестік белгісі - сатып алушыларды сертификаттау нысанының ерікті сертификаттау жүйесінің талаптарына және ұлттық стандартқа сәйкестігі ақпараттандыру үшін қызмет ететін белгі.

Өнімді сәйкестендіру - өнім сипаттамаларының оның маңызды белгілеріне біркелкілігін орнату.

Техникалық регламент талаптарын сақтауды бақылау (қадағалау) - заңды тұлғаның немесе жеке кәсіпкердің өнімге, өндіріс процестеріне, пайдалануға, сақтауға тасымалдауға, сатуға және кәдеге жаратуға техникалық регламенттің талаптарын орындауын тексеру және тексеру нәтижелері бойынша шаралар қолдану.

Сертификаттау органы - сертификаттау бойынша жұмыстарды орындауға белгіленген тәртіпте аккредиттелген заңды тұлға немесе жеке кәсіпкер.

Сәйкестікті бағалау - нысанға қойылатын талаптардың сақталуын тікелей немесе жанама анықтау.

Сәйкестікті растау - өнімнің немесе өзге нысандардың, өндіріс процестерінің, пайдалану, сақтау, тасымалдау, сату және кәдеге жарату, жұмыстарды орындау немесе қызметтер көрсетудің техникалық регламент талаптарына, стандарттардың ережелеріне немесе шарттардың талаптарына сәйкестігін құжат жүзінде куәландыру. Сәйкестікті растау жеткізушімен және тәуелсіз органдармен жүзеге асырыла алады.

Сертификаттау - сертификаттау органымен жүзеге асырылатын, нысандардың техникалық регламент талаптарына, стандарттардың ережелеріне немесе шарттардың талаптарына сәйкестігін растауының нысаны.

Сәйкестік сертификаты - нысанның техникалық регламент талаптарына, стандарттардың ережелеріне немесе шарттардың талаптарына сәйкестігін куәландыратын құжат.

Сертификаттау жүйесі - оның қатысушыларының сертификаттау бойынша жұмыстарды орындау ережелерінің және жалпы алғанда сертификаттау жүйесінің қызмет ету ережелерінің жиынтығы.

Стандарт - ерікті түрде және көп мәрте пайдалану мақсатында, өнімнің сипаттамалары, өндіру, пайдалану, сақтау, тасымалдау, сату және кәдеге жарату, жұмыстарды орындау немесе қызмет көрсетудің процестерін жүзеге асыру ережелері және сипаттамалары белгіленетін құжат. Сондай-ақ, стандартта символикаға, қаптамаға, таңбалауға немесе зат белгілерге және оларды басу ережелеріне қойылатын талаптар да қамтылуы мүмкін..

Стандарттау - көп мәрте және ерікті пайдалану үшін ережелер және сипаттамалар белгілеу арқылы өнім өндіру және айналымға шығару саласындағы және өнімнің, жұмыстың немесе көрсетілетін қызметтің бәсекеге қабілеттілігін арттыру бойынша ережелер мен сипаттмалар белгілеу жөніндегі қызмет.

Техникалық реттеу - өнімге, өндіру, салу, монтаждау, баптау, пайдалану, сақтау, тасымалдау, өткізу және кәдеге жарату процестеріне, қойылатын міндетті талаптарды белгілеу, қолдану және орындау саласындағы сондай-ақ өнімге, өндіру, салу, монтаждау, баптау, пайдалану, сақтау, тасымалдау, өткізу және кәдеге жарату процестеріне қойылатын талаптарды ерікті түрде белгілеу және қолдану саласындағы, сондай-ақ сәйкестікті бағалау саласындағы қатынастарды құқықтық реттеу;

Техникалық регламент - Ресей Федерациясының заңнамасымен, немесе Федералды заңмен, немесе Ресей Федерациясының Бұйрығымен, немесе Ресей Федерациясы Үкіметінің қаулысымен белгіленген тәртіп бойынша ратификацияланған Ресей Федерациясының халықаралық шартымен қабыданған құжат және техникалық реттеу нысандарына (өнімдерге, соның ішінде ғимараттарға, құрылыстарға және құрылымдарға, өндіріс, пайдалану, сақтау, тасымалдау, сату және кәдеге жарату процестеріне) қолдануға және орындауға міндетті талаптарды белгілейді.

Сәйкестікті растау нысаны - өнімнің немесе басқа нысандардың, жұмыстардың орындалуының немесе қызмет көрсетілудің техникалық регламент талаптарына, стандарттардың ережелеріне немесе шарттардың талаптарына сәйкестігін құжат жүзінде куәландырудың белгілі реті.

Сәйкестікті растау келесі принциптердің негізінде мыналар жүзеге асырылады:

- мүдделі тұлғаларға сәйкестікті растауды жүзеге асыру тәртібі туралы ақпараттың қолжетімділігі;
- техникалық регламенттің талаптары белгіленбеген нысандарға сәйкестікті міндетті растауды қолданудың жол берілмеушіліктері;
- тиісті техникалық регламенттегі белгілі өнім түрлеріне қатысты сәйкестікті міндетті растау сызбасы мен нысандар тізімін белгілеу;
- сәйкестікті және өтініш берушінің шығынын міндетті растау мерзімдерін қысқарту;
- сәйкестікті ерікті растауды жүзеге асыруға мәжбүрлеуге жол берілмеушіліктері, соның ішінде ерікті сертификаттаудың белгілі жүйесінде;
- өтініш берушілердің мүліктік мүдделерін қорғау, сәйкестікті растауды жүзеге асыру кезінде алынған мәліметтерге қатысты коммерциялық құпияны сақтау;
- сәйкестікті міндетті растауды ерікті сертификаттаумен алмастыруға жол бермеу.

Ресей Федерациясының аумағында сәйкестікті растау ерікті немесе міндетті сипатта болуы мүмкін.

Сәйкестікті ерікті растау ерікті сертификаттау түрінде жүзеге асырылады.

Сәйкестікті міндетті растау сәкестік туралы декларацияны қабылдау түрінде жүзеге асырылады.

Сертификаттау жөніндегі орган төменде келтірілген қызметтерді атқарады:

- сәйкестікті ерікті растау нысандарының сәйкестігін растауды жүзеге асырады;
- ерікті сертификаттаудан өткен нысандарға сәйкестік сертификаттарын береді;
- өтініш берушілерге сәйкестік белгісін қолдану құқығын ұсынады, егер сәйкестік белгісін қолдану ерікті сертификаттаудың тиісті жүйесімен қарастырылған болса;
- онымен берілген сәйкестік сертификаттарының әрекетін тоқтатады немесе жояды.

14.2. ӨНІМДІ СЕРТИФИКАТТАУДЫ ЖҮРГІЗУДІҢ ТӘРТІБІ

Кез келген сертификаттау сынақтарының принципшіл ерекшелігі - бұл, сынақ нәтижелеріне тәуелсіз бақылауды жүзеге асыратын, сертификаттау органының тәуелсіздігі.

Бағдарламалық өнімді сертификаттау келесі негізгі кезеңдерді қамтиды:

- өтініш берушінің сертификаттау жөніндегі органға сертификаттауға өтінім беруі;
- өтінім бойынша шешімді қарастыру және қабылдау, сынақ зертханасын таңдау және сертификаттау сызбасы;
- қажетті тексерулер жүргізу (құжаттарды талдау, сынақтар және т.б.), сыналатын БҚ шолу және нұсқасын сәйкестендіру;
- сынақ зертханасымен БҚ сертификаттау сынақтары;
- алынған нәтижелерді талдау және өтініш берушіге сәйкестік сертификатын беру мүмкіндігі туралы шешім қабылдау;
- бағдарламалық өнімнің сертификатталған нұсқасына сертификат беру;
- сертификаттау сызбасына сәйкес сертификатталған бағдарламалық қамсыздандыруды инспекциялық бақылау.

Сертификаттауға өтінім беру. Өтініш беруші өтінімді сертификаттау жөніндегі тиісті органға жібереді, ал ол болмаған жағдайда - РФ Мемлекеттік стандартына немесе басқа мемлекеттік басқару органына жібереді.

Өтінім бойынша шешім қарастыру және қабылдау. Сертификаттау жөніндегі органы өтінімді қарастырады және оны алғаннан кейін 15 күннен кешіктірмей өтініш берушіге өтінім бойынша шешімді жібереді.

Қажетті тексерулерді өткізу. РФ заңнамасына сәйкес, сертификаттауға тағайындалуы бойынша пайдалануға жарамды және қажетті таңбалауы және бағдарламалық өнім туралы ақпаратты қамтитын техникалық құжатамасы бар бағдарламалық өнім жіберіледі.

Сынақ хаттамалары өтініш берушіге және сертификаттау жөніндегі органға жіберіледі, олардың сақталуы сертификаттың әрекет ету мерзіміне сәйкес болады.

Егерде сынақ барысында мәлімделген талаптарға белгілі бір сәйкессіздіктер анықталса, онда олар өтініш берушімен жұмыс тәртібінде жойылады, көптеген жағдайларда дәл осылай болады немесе өнімге қойылатын талаптарды өзгерту туралы шешім қабылдануы мүмкін, мысалы, қорғалу класының төмендеуі туралы шешім.

Алынған нәтижелер талдауы және сәйкестік сертификатын беру

мүмкіндігі туралы шешім қабылдау. Сертификаттау жөніндегі органға келіп түскен сынақ хаттамалары және өнімнің сәйкестігі туралы басқа құжаттар, бағдарламалық өнімнің белгіленген талаптарға сәйкестігі туралы қорытынды алу үшін талдаудан өткізіледі. Осы деректердің негізінде талдаудың нәтижелері бойынша, сарапшының қорытындысы жасалады. Осы құжаттың негізінде сертификаттау жөніндегі орган сәйкестік сертификатын беру туралы шешім қабылдайды.

Сертификаттау сынақтары теріс нәтижемен аяқталуы да мүмкін. Сарапшының теріс қорытындысында, сертификаттау органы өтініш берушіге себептерін көрсете отырып, бас тарту туралы шешім шығарады. Бұл жағдайда сертификаттау жөніндегі орган, сондай-ақ өндірушінің, өнімді сатушының (жұыстарды орындау үшін) орналасқан жері бойынша, тиісті мемлекетті бақылау және қадағалау аумақтық органын, аталмыш өнімді сату (немесе жұмыстарды орындау, қызметтер көрсету) бойынша ескерту бойынша қажетті шараларды қабылдау үшін, ескертуі тиіс.

Сәйкестік белгісін қолдануға сертификат және лицензия (рұқсат) беру. Сертификаттау нәтижелері бойынша, бағдарламалық өнімге, нормативтік құжаттардың талаптарына сәйкестігі расталған жағдайда сәйкестік сертификаты беріледі. Бұл өнім белгіленген тәртіпке сәйкес, сәйкестік таңбасымен таңбаланады. Сәйкестік белгісі қаптамаға, тұтынушыға сату кезінде келіп түсетін ілестірілетін техникалық құжаттамаға басылады.

Өндіруші, сертификаттау органынан лицензия ала отырып, сертификатталған өнімді сәйкестік белгісімен таңбалау құқығын алады.

Әдетте әрбір жүйеде өз белгісі қабылданған. Сертификаттау жүйесіндегі сертификат та, оларды мемлекеттік тізілім де міндетті түрде тіркеуге қойғаннан бастап күшіне енеді.

Сертификаттау сызбасына сәйкес сертификатталған нысанды (сертификаталған өнімді) *инспекциялық бақылау,* сертификаттау сызбасымен қарастырылған жағдайда, сертификаттың және сәйкестік таңбасын қолдануға лицензияның әрекет ету мерзімінің ішінде жүргізледі. Жоспардан тыс тексерулер, тұтынушылардан және бақылау органдарынан өнімнің сапасына наразылықтар туралы ақпарат түскен жағдайда, сертификаттау жөніндегі органмен тағайындалады. Инспекциялық бақылаудың мақсаты сатылып жатқан өнімнің белгіленген талаптарға сәйкестігін растауда. Сертификатталған өнімнің нақты түріне инспекиялық бақылаудың жалпы ережелері, сертификаттау органдарын және сынақ зертханаларын аккредиттеу бойынша ережелерді және біртекті өнімді сертификаттау ережесімен анықтайтын құжаттармен белгіленеді.

Инспециялық бақылаудың нәтижелері актімен рәсімделеді, ол

сертификаттау жөніндегі органда сақталады.

Сертификаттау жөніндегі орган сертификаттың және сәйкестік белгісін қолдануға берілген лицензияның әсерін тоқтатуға немесе жоюға құқығы бар.

БАҚЫЛАУ СҰРАҚТАРЫ МЕН ТАПСЫРМАЛАРЫ

1. Сертификаттау дегеніміз не?
2. Бағдарламалық қамсыздандыруды сертификаттаудың мақсаттары?
3. Ресей Федерациясында, өнімдер мен қызметтерді сертификаттаудың құқықтық нормалары қандай заңмен реттеледі?
4. Ақпараттық технологиялар мен қызметтердің бағдарламалық қамсыздандыруды сертификаттаудың негізгі мақсаттары?
5. «Сәйкестік декларациясының» «сәйкестік белгісінен» айырмашылығы неде?
6. «Сәйкестікті бағалауды» «сәйкестікті растаудан» не ажыратады»?
7. Техникалық регламент дегеніміз не?
8. Өнімдердің/қызметтердің техникалық регламенттердің талаптарына сәйкестігін реттеудің принциптері қандай?
9. Техникалық реттеу дегеніміз не?
10. Сәйкестікті растау нысаны деп нені атайды?
11. Сертификаттау жөніндегі орган қандай қызметтер атқарады?
12. Бағдарламалық өнімді сертификаттаудың негізгі кезеңдерін атап өтіңіз.

Әдебиеттер тізімі

IEEE 830-1998. Бағдарламалық қамсыздандыруға қойылатын талаптар сипаттамасын құрудың ұсынылған әдістемесі.

МемСТ 19.001-77. Бағдарламалық құжаттаманың бірыңғай жүйесі. Жалпы ережелер.

МемСТ 19.502-78. Бағдарламалық құжаттаманың бірыңғай жүйесі. Жалпы сипаты. Мазмұнына және безендіруге қойылатын талаптар.

МемСТ 19.504-79. Бағдарламалық құжаттаманың бірыңғай жүйесі Бағдарламашының нұсқаулығы. Мазмұнына және безендіруге қойылатын талаптар.

МемСТ 27.002-89. Техникадағы сенімділік. Негізгі түсініктер. Терминдер мен анықтамалар.

МемСТ 34.602-89. Ақпараттық технология. Автоматтандырылған жүйедегі стандарттар кешені. Автоматтандырылған жүйені құруға техникалық тапсырма.

Р МемСТ ИСО/ХЭК 12207-2010. Ақпараттық технология. Бағдарламалық құралдардың өмірлік циклының процестері.

Р МемСТ ИСО/ХЭК 15910-93. Ақпараттық технология. Бағдарламалық құралдың пайдаланушысының құжаттамасын құру процесі.

Р МемСТ ИСО/ХЭК ТО 9294-2002. Ақпараттық технология. Бағдарламалық қамсыздандыруды құжаттауды басқару жөніндегі нұсқаулық.

Р МемСТ ИСО/ХЭК ТО 15271-2002. Ақпараттық технология. Р МемСТ ИСО/МЭК 12207-2010 (Бағдарламалық құралдардың өмірлік циклының процестері) қолдану жөніндегі нұсқаулық.

Р МемСТ ИСО/ХЭК ТО 16326-2002. Бағдарламалық инженерия. Жобаны басқару кезінде Р МемСТ ИСО/МЭК 12207-2010 қолдану жөніндегі нұсқаулық.

Р МемСТ ИСО/ХЭК 12119-2000. Ақпараттық технология. Бағдарламалар топтамасы. Сапасына қойылатын талаптар және тестілеу.

Р МемСТ ИСО/ХЭК 9126-93. Ақпараттық технология. Бағдарламалық өнімді бағалау. Сапа сипаттамалары және оларды қолдану жөніндегі нұсқаулық.

Р МемСТ ИСО/ХЭК 8631-94. Ақпараттық технология. Бағдарламалық конструктивтер және оларды таныстыруға арналған шартты белгілер.

Бахтизин В.В. Бағдарламалық жасақтама технологиясы: О.И. Владимир / В.В. Бахтизин, Л. А. Глухова. - Минск: BSUIR, 2010.

Благодатских В.А. бағдарламалық қамтамасыз етуді стандарттау: О.У. Владимир / Благодатских В.А., Вольнин В.А., Қ.Ф. Поскакалов; КСРО редакторлары О.Разумова - М .: Қаржы және статистика, 2003 ж.

Боггфс Уэнди. UML и Rational Rose / Уэнди Боггс, Майкл Боггс. - М.: Лори, 2001.

Гагарина, Л.Г. Бағдарламалық қамтамасыз етуді әзірлеу технологиясының негіздері: О.У. Владимир / LG Gagarina, B.D. Виснадуль, А.В. Игошин. - Мәскеу: FORUM: INFRA-M, 2006. - (Кіспптік білім). Иванова Г.С. Бағдарламалау технологиясы: LOE Arnal'an oyiyatitar / GS Иванова. - М .: Н.Э. Бауман атындағы АТУ, 2002 ж.

Леонников А.В. Оқытушы UML / A.V. Леонников. - 2-ши басс. - Санкт-Петербург: BHV-Петербург, 2004.

Орлов С.А. Бағдарламалық жасақтама әзірлеу технологиясы: О.У. Владимир / С.А. Орлов. - 2-ши басс. - Санкт-Петербург: Петр, 2003.

Пономарев В.А. және ActiveX в Delphi / VA Пономарев. - Санкт-Петербург: BHV. - Петербург, 2001.

Rambo J. UML 2.0. Объектілі-бағытталған модельдеу және даму / Дж. Рамбо, М. Блака. - 2-ши басс. - Санкт-Петербург: Петр, 2007.

Рудаков А.В. БҚ әзірлеу технологиясы: А.В. Рудаков. - Мәскеу: «Академия» баспа орталығы, 2010 ж.

Fowler M. UML қысқаша мәлімдемесінде. Стандартты объектілі модельдеу тілін қолдану / M. Fowler, K. Scott; Айылш. Ауд. - М .: Әлем, 1999.

Шураков В. В. бағдарламалық қамтамасыз етудің сенімділігі. - М .: Қаржы және статистика, 1987 ж.

Джекобсон А. Бағдарламалық қамтамасыз етудің бірыңғай процесі / А. Якобсон, Х.Бух, Дж. Рэмбо. - Санкт-Петербург: Питер, 2002.

2002 ж. «Бірінші рет, 2014 жылғы 22 желтоқсандағы Кошине ауылында« Rettue Tourals »әдістемесі №184-ФЗ 27 жалған жала жапты.

Мазмұны

II БӨЛІМ. **БАҒДАРЛАМАЛЫҚ ҚАМСЫЗДАНДЫРУДЫ ӘЗІРЛЕУДІҢ АСПАПТЫҚ ҚҰРАЛДАРЫ**

III-БӨЛІМ. **ҚҰЖАТТАУ ЖӘНЕ СЕРТИФИКАТТАУ**

Оқу баспасы

Федорова Галина Николаевна

Бағдарламалық модульдерді біріктіруге қатысу

Оқу құралы

Редактор *О.Ю.Румянцева,*
Компьютерлік беттеу: *Л. М. Беляева*
Корректор *Л.В.Гаврилина*

Баспа № 101116893. Басып шығаруға қол қойылды16.09.2015. Пішіні 60 х 90/16.
«Балтика» гарнитурасы. Офсеттік қағаз. Офсеттік баспа. Шартты баспа беті 19,0.
Таралымы 1500 дана. Тапсырыс №

«Академия» баспа орталығы» ЖШҚ www.academia-moscow.ru 129085, Мәскеу, Мир даңғылы, 101В, 1-құр.
Тел./факс: (495) 648-0507, 616-00-29.

Санитарлық-эпидемиологиялық қорытындысы № РОСС RU. АЕ51. Н 16679, 25.05.2015.

Баспаның электронды тасымалдауыштарынан басып шығарылды.
«Тверской полиграфический комбинат» ААҚ, 170024, Тверь қ., Ленин даңғылы, 5.
Телефон: (4822) 44-52-03, 44-50-34. Телефон/факс: (4822) 44-42-15
Home page - www.tverpk.ru. Электронды пошта (E-mail) - sales@tverpk.ru