**Г.Н. ФЕДОРОВА**

# КОМПЬЮТЕРЛІК ЖҮЙЕЛЕРГЕ АРНАЛҒАН БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУДІҢ МОДУЛЬДЕРІН ӘЗІРЛЕУ

ОҚУЛЫҚ

*«Білім беруді дамытудың федералды институты» федералды мемлекеттік автономды мекемесі «Компьютерлік жүйелерде бағдарламалау» мамандығы бойынша орта кәсіптік білім беру бағдарламасын іске асыратын білім беру ұйымдарының оқу үрдісінде пайдалану үшін оқулық ұсынған*

*2016 жылғы 31 наурыздағы пікірдің тіркеу нөмірі 84.*

ACADEMIA

Мәскеу, 2016
«Академия» баспа орталығы

ӘОЖ 004(075.32)
КБЖ 32.973.202-018.2(075.32)
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 -
*В.В.Кузьмин* - СМСҚУ ЖББ ФМББМ филиалының директоры доцент, тех.ғыл.канд., Белебей қ.

**Федорова Г. Н.**

Ф333

Компьютерлік жүйелерге арналған бағдарламалық қамтамасыз етудің модульдерін әзірлеу: орта кәсіп. білім беру мекемелерінің студенттеріне арналған оқулық /Г.Н. Федорова. —М.: «Академия» баспа орталығы, 2016. — 336 б.

ISBN 978-601-333-145-4 (каз.)
ISBN 978-5-4468-1585-2 (рус.)

Оқулық «Компьютерлік жүйелерге арналған бағдарламалық қамтамасыз етудің бағдарламалық модульдерін әзірлеу» 01 КМ бойынша «Компьютерлік жүйелерде бағдарламалау» мамандығы бойынша орта кәсіптік білім берудің Федералды мемлекеттік білім беру стандартының талаптарына толық сәйкестікте дайындалды.

Бағдарламалық қамтамасыз етуді әзірлеу кезеңдері, бағдарламалық өнімдерді ретке келтіру мен тестілеу әдістері, техникалық құжаттаманы әзірлеу түрлері мен құралдары жазылды. Жүйелі бағдарламалау технологиясы қаралды. Web-бағдарламалау және «1С» жүйесінде қолданбалы бағдарламалық қамтамасыз етуді құру мәселелеріне аса назар аударылды.

Орта кәсіптік білім беру мекемелерінің студенттеріне арналған. Бағдарламалық өнімдерді әзірлеумен байланысты, көп тұлғалар тобы үшін пайдалы болуы мүмкін.

ӘОЖ 004(075.32)
КБЖ 32.973.202-018.2(075.32)

ISBN 978-601-333-145-4 (каз.)
ISBN 978-5-4468-1585-2 (рус.)

# Құрметті оқырман!

Осы оқулық «Компьютерлік жүйелерде бағдарламалау» мамандығы бойынша оқу-әдістемелік жинақтамасының бір бөлігі болып табылады.

Оқулық «Компьютерлік жүйелерге арналған бағдарламалық қамтамасыз етудің модульдерін әзірлеу» кәсіптік модульді оқып білуге арналған.

Жаңа ұрпақтың оқу-әдістемелік жинақтамаларына жалпы білім беру және жалпы кәсіптік пәндер мен кәсіптік модульдерді оқып білуді қамтамасыз етуге мүмкіндік беретін дәстүрлі және инновациялық оқу материалдары кіреді. Әр жинақтамада оқулықтар мен оқу құралдары, жалпы және кәсіптік біліктілікті игеру үшін қажет, оның ішінде жұмыс беруші талаптарының есебімен оқыту және бақылау құралдары бар.

Оқу басылымдары электронды білім беру ресурстарымен толықтырылған. Электронды ресурстарға интерактивті жаттығулар мен жаттығу құрылғыларымен бірге нысандар, теориялық және тәжірибелік модульдер, мультимедиялық, интернеттегі қосымша материалдар мен ресурстарға сілтемелер кіреді. Оған терминологиялық сөздік пен электронды журнал енгізілді, онда оқу үрдісінің негізгі параметрлері: жұмыс уақыты, бақылау және практикалық тапсырмаларды орындау нәтижесі белгіленеді. Электронды ресурстар оқу үдерісін жеңілдетіп, оған жылдам бейімделуге мүмкіндік береді.

# Алғы сөз

Оқу басылымы «Компьютерлік жүйелерге арналған бағдарламалық қамтамасыз етудің модульдерін әзірлеу» кәсіптік модулі бойынша «Компьютерлік жүйелерде бағдарламалау» мамандығына арналған орта кәсіптік білім берудің Федералды мемлекеттік білім беру стандартының талаптарына толық сәйкестікте дайындалды.

Ұсынылып отырған оқу құралы көлемдік шектелуіне байланысты, көрсетілген тақырыптар бойынша материал толық қамтылмаған, себебі әр бөлім бағдарламалау және ақпараттық технологиялар саласындағы жеке курсқа сәйкес келеді. Бұл кітап бағдарламалық қамтамасыз етуді әзірлеу технологиялары, бағдарламалаудың кейбір заманауи тілдері, бағдарламалық жүйелердің компоненттерінің өзара әрекет етуінің процестері, түрлі бағдарламалық платформаларда сол немесе өзге де қолданбалы міндеттерді шешу ерекшеліктері туралы кең түсінік беретін сияқты. Қолданбалы және жүйелі бағдарламалаудың кейбір барынша маңызды аспектілері ғана тереңінен қаралады. Ұсынылып отырған материалға барынша егжей-тегжейлі түсінік алу үшін кітаптың әр бөлімінің соңында келтірілген әдебиеттер тізіміне жүгінген жөн.

Оқу құралы төрт бөлімнен, олардың әр қайсысына бірнеше тараудан тұрады.

*І-бөлімде* бағдарламалық қамтамасыз етуді жасау технологияларының жалпы мәселелері қаралады: тіршілік циклінің негізгі кезеңдері және бағдарламалық өнімді әзірлеу стратегиясы. Бағдарламалаудың құрылымдық және объектілік-бағдарланған технологиясының мәні ашылады. Бағдарламалық қамтамасыз етуді ретке келтірумен және тестілеумен байланысты мәселелер түсіндіріледі. Жеке тарау бағдарламалық қамтамасыз етуді құжаттау процесін сипаттауға арналған. Бағдарламалық құралдарды әзірлеу және қолдану бөлігінде мемлекеттік стандарттардың талаптары келтіріледі.

Оқу құралының екінші және үшінші бөлімдері қолданбалы бағдарламалық қамтамасыз етуді әзірлеу үрдістеріне арналған.

*II-бөлімде* интернет-қосымшаларды әзірлеудің негізгі қағидалары жазылады. Оқу басылымының бұл бөлімі Web-әзірлемеде пайдаланылатын технологиялар мен тәсілдердің жеткілікті кең спектрін қамтиды. Клиенттік белсенділікті іске асырудың тәсілдері сипатталады - JavaScript, VBScript клиенттік сценарийлері, сондай-ақ Java-қосымшаларын жасау аспаптары туралы айтылады. ActivX басқару элементтеріне түсінік беріледі, оларды жасау және қолдану, сондай-ақ Web-парақшасына енгізу технологиясы қаралады. Жеке тарау белсенді Web-серверлер ұйымдастыру, CGI көмегімен HTML динамикалық құжаттарды қалыптастыру тәсілдеріне, Perl, PHP тілдеріне арналған. ISAPI қосымшалары және оларды динамикалық құжаттарды жасақтау үшін пайдалану қаралады. ASP және ASP. NET ортасы туралы айтылады.

*III-бөлімде* қазіргі уақытта кеңінен пайдаланылып жүрген «1С: Кәсіпорын» жүйесі ұсынылды. Мұнда «1С» жүйесінде әзірлеу тұжырымдамасы ашылады, жүйенің түрлі объектілері құрылымдарының сипаттамасы, олардың мақсаты және пайдалану әдісі беріледі. «1С: Кәсіпорын» платформасындағы қолданбалы шешімдерді әзірлеу технологиясы, деректер сақтау және пайдалану қағидалары, сондай-ақ қосымшалардағы клиент-серверлік өзара іс-қимылдардың ерекшеліктері түсіндіріледі.

*IV-бөлімде* жүйелі бағдарламалаудың жеке мәселелері қаралады. Ресурстарды пайдалану, ағындар мен процестерді басқару, процестердің арасында деректерді беру, динамикалық тұрғыда қосылатын кітапханаларды құру, сервистерді әзірлеу қағидалары сипатталады.

Оқу құралының әр тарауына студентке оқыған материалды бекітуге мүмкіндік беретін өзіндік жұмыстарға арналған бақылау сұрақтары мен тапсырмалар енгізіледі.

Оқу құралын дайындау кезінде, бағдарламалық қамтамасыз етуді құру және енгізу бойынша оқыту және практикалық үрдіс кезінде автормен жинақтаған тәжірибе мен материалдар пайдаланылды. Оқу құралы оқытушылармен және «Компьютерлік жүйелерге арналған бағдарламалық қамтамасыз етудің бағдарламалық модульдерін әзірлеу» кәсіптік модульді оқыған кезде аталған мамандықтың студенттері пайдалана алады.

# Кіріспе

Кез келген дербес компьютерлер, есептеу жүйелері, желілер, мобильді құрылғы оларға орнатылған бағдарламалық қамтамасыз етудің басқаруында жұмыс істейді және адам қызметінің түрлі салаларында қолданбалы міндеттерді шешу үшін қызмет етеді. Бағдарламалық қамтамасыз ету – компьютерлік жүйенің ажырамас бөлігі. Ол техникалық құралдардың логикалық жалғасы болып табылады. Нақты компьютерді қолдану саласы үшін жасалған бағдарламалық қамтамасыз ету арқылы айқындалады. Компьютердің өзінің бірде-бір қолдану саласында білімге ие болмайды. Бұл білімдердің барлығы компьютерде орындалатын бағдарламаларда жинақталды.

*Бағдарламалық қамтамасыз ету* – бұл компьютерде қызметтің кез келген саласындағы түрлі мақсаттағы міндеттерді шешуге мүмкіндік беретін, сондай-ақ ЭЕМ аппараттық құралдарының қызмет етуін қамтамасыз ететін бағдарламалық құралдардың және оларға ілесетін құжаттаманың жиынтығы. Бағдарламалық қамтамасыз етудің құрамын бағдарламалық конфигурация деп атайды.

Күрделі және көп функциялы бағдарламалық қамтамасыз етудің өзара байланысты бағдарламалық модульдерден құрылған ішкі ұйымына немесе ішкі құрылымына ие болады. Осындай бағдарламалық *өнімдер бағдарламалық жүйелер* деп аталады.

Бағдарламалық қамтамасыз етудің мынадай санаттарын айқындайды: жүйелі, қолданбалы және аспаптық.

*Жүйелі бағдарламалық қамтамасыз ету* – бұл компьютерлік жүйенің компоненттерын (процессорды, жедел жадын, енгізу- шығару құрылғыларын, желілік жабдықтарды және өзгелерді) басқаруды қамтамасыз ететін бағдарламалардың кешені. Жүйелі бағдарламалық қамтамасыз етудің екі түрі бар: базалық және сервистік.

*Қолданбалы бағдарламалық қамтамасыз ету* – бұл белгілі пайдаланушылық міндеттерді орындауға арналған бағдарламалардың кешені. Көптеген операциялық жүйелерде қолданбалы бағдарламалар компьютердің ресурстарына тікелей жүгіне алмайды, ал жабдықтармен және өзгелермен операциялық жүйе арқылы өзара әрекет етеді.

*Аспапты бағдарламалық қамтамасыз ету* – бұл бағдарламаларды жобалау, әзірлеу және сүйемелдеу барысында пайдалануға арналған бағдарламалық қамтамасыз ету. Аспаптық бағдарламалық қамтамасыз ету жүйелі және қолданбалы бағдарламаларды қоса алғанда, кез келген саладағы бағдарламалық өнімдерді құру үшін пайдаланылады. Қазіргі уақытта бағдарламалық өнімдерді жасау үшін, көрнекі бағдарламалаудың қуатты жүйесі пайдаланылады, олардың стандартты бағдарламаларының кең кітапханасы, ретке келтіру мен тестілеудің арнайы құралдары болады. Бағдарламалау жүйелері нақты бағдарламалау тілінде жаңа бағдарламаларды әзірлеуге арналған және оған түсінік берушілер, түсіндірушілер, диалогтық орта, мәтіндердің редакторлары, стандартты қосымша бағдарламалардың кітапханалары, құрастырушылар, ретке келтірушілер, анықтама қызметтері, автоматтандырылған тестілеу құралдары, нұсқаларды басқару жүйелері, парсерлер мен басқалар кіреді.

*Қолданбалы бағдарламалауды* қолданбалы бағдарламалық қамтамасыз етуді әзірлеу үрдісі, ал *жүйелі бағдарламалауды* – жүйелі бағдарламалық қамтамасыз етудің әзірлемесі ретінде айқындауға болады. Жүйелі бағдарламалау қолданбалыдан елеулі ерекшеленеді, неге десең ол аппараттық бөліктен күшті тәуелділікті болжайды.

*Ерекшелік* бағдарламалық немесе аппараттық компоненттердің, олардың қызмет ету тәсілдерінің, басқа компоненттермен өзара әрекет етуінің, пайдалану талаптарының, ерекше сипаттамаларының нысандандырылған сипатын білдіреді. Бұл әзірленетін бағдарламалық өнім функциясы мен шектеулерінің дәл нысандандырылған сипаты.

Бағдарламалық қамтамасыз етуді жасаған кезде көп рет пайдаланылған модульдер көрсетіледі, оларға типтендіру және біріздендіру жүргізіледі, оның есебі арқылы тұтас алғанда бағдарламалық өнімді әзірлеуге мерзімдер мен еңбек шығындары қысқарады. Кейбір бағдарламалық өнімдер стандартты қосымша бағдарламалардың дайын кітапханаларынан деректерді өңдеу рәсімдерінен, функцияларынан, объектілерінен, әдістерінен модульдерді пайдаланады.

Бағдарламалық өнімдерді әзірлеудің заманауи құралдары көрнекі құралдардың ауқымды жиынына ие болады.

*Көрнекі бағдарламалау* — бұл бағдарламаны оның мәтінін жазу орнына графикалық объектілермен айла-шарғы жасау жолымен жасау тәсілі.

Кез келген бағдарламалық өнім ол үшін жасалған, функцияларды орындауға тиіс. Сапалы бағдарламалық өнім оны ұзақ уақыт ішінде табысты пайдалануға мүмкіндік беретін тағы бірқатар қасиеттерге ие болуға тиіс.

*Бағдарламалық қамтамасыз етудің сапасы* – бұл бағдарламалық қамтамасыз етудің пайдаланушының тағайындалған қажеттіліктерін қанағаттандыру қабілетін белгілейтін, оның сипаттамаларының жиынтығы. Бұл, алайда, түрлі бағдарламалық өнімдер сандық көрсеткіштердің бірдей мәндерімен сол бір қасиеттер жиынына ие болуға тиіс дегенді білдірмейді. Техникалық құрылғылар жағдайында сияқты, сапа көрсеткіштері қарама-қайшы болып табылады, ол мынаны білдіреді: бір сапа көрсеткіштерін жақсартуға басқаны нашарлату есебі арқылы қол жеткізуге болады. Бағдарламалық қамтамасыз етудің сапасы, егер қасиеттердің сандық көрсеткіштері оны табысты пайдаланылуға кепілдік берсе, қанағаттанарлық болып табылады.

Бағдарламалық өнім әзірлеушілерінің міндеті — талап етілген функционалдығы мен техникалық құжаттама жиыны бар жүйелі, қолданбалы немесе аспапты - сапалы бағдарламалық қамтамасыз ету жасау.

# БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУДІ ӘЗІРЛЕУДІҢ КЕЗЕҢДЕРІ

# I

## БӨЛІМ

# 1 тарау

# БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУДІ ӘЗІРЛЕУДІҢ ТІРШІЛІК ЦИКЛІ

## 1.1. БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУ ТІРШІЛІК ЦИКЛІНІҢ ҰҒЫМЫ

Кез келген жүйенің немесе бағдарламалық өнімнің тіршілік үрдісі қолдану саласына, мөлшеріне, өзгерістер мен мүмкіндіктердің қиындығына, қажеттілігіне сәйкес циклді тоқтатылуы және (немесе) қайталануы мүмкін, сатылардан тұратын тіршілік циклінің моделі арқылы сипатталуы мүмкін болады. *Бағдарламалық қамтамасыз етудің тіршілік циклі* бағдарламалық қамтамасыз етуді жасау туралы шешім қабылдау сәтінен басталатын және оны пайдаланудан толықтай алу сәтінде аяқталатын үздіксіз процесті білдіреді.

Бағдарламалық қамтамасыз етуді әзірлеу, әдетте, белгілі бір пәндік сала үшін орындалады. Пәндік саланың және онда бағдарламалық қамтамасыз етудің қызмет ету ерекшеліктері, сөзсіз, бағдарламалық қамтамасыз етудің құрамына әсер етеді. Жобаларды басқару мен жобаны әзірлеу фазаларының (тіршілік циклінің фазасы) көптеген ерекшеліктері пәндік саладан ғана емес, сонымен және жоба сипатынан да тәуелсіз ортақ болып табылады. Әр жобаның орындалуы үшін қажет жұмыстың қиындығы мен көлемінен тәуелсіз, ол өз дамуында белгілі бір күйден өтеді. Идея туындаудан жобаны толық аяқтауға дейін даму сатыларының жиынтығын сатыларға немесе кезеңдерге бөлу қолданылады.

Сатылар санын және олардың мазмұнын айқындауда, бұл сипаттамалар нақты жобаны іске асыру жағдайларынан және негізгі қатысушылардың тәжірибесінен көбінде тәуелді болғандықтан, кейбір айырмашылықтары бар. Дегенмен де, бағдарламалық қамтамасыз етуді әзірлеу процесінің логикасы мен негізгі мазмұны барлық жағдайда жалпы болып табылады дерлік.

Бағдарламалық қамтамасыз етуді, яғни оның тіршілік циклін әзірлеу мен дамытудың мынадай сатыларын атап өтуге болады:

1) пәндік саланы саралау және талаптарды (тұжырымдаманы) қалыптастыру;
2) жобалау;
3) іске асыру;
4) тестілеу;
5) пайдалануға енгізу;
6) жобаны сүйемелдеу (пайдалану).

Бағдарламалық қамтамасыз етудің тіршілік циклі оны пайдаланудан шығарумен аяқталады.

Әр саты үшін орындалған жұмыстардың құрамы мен жүйелілігі, алынған нәтижелері, жұмыстарды жүзеге асыру үшін қажет әдістер мен құралдары, қатысушылардың рөльдері мен жауапкершілігі және т.б. айқындалады.

Бағдарламалық қамтамасыз етудің *пәндік саласын талдау және талаптарды (тұжырымдаманы) қалыптастыру сатысы* бүкіл жобаның табысын айқындағандықтан, маңыздылардың бірі болып табылады. Бұл кезеңде жобаның мақсаттары мен міндеттері тұжырымдалады, бағдарламалық қамтамасыз етуді қолдану саласы белгіленеді, базалық мәндері және олардың арасындағы өзара байланыстар бөлінеді. Әзірленіп жатқан бағдарламалық қамтамасыз ету өзара әрекет етуге тиіс барлық сыртқы объектілерді дәлме-дәл келтіреді және бұл өзара әрекет етудің сипатын жоғары деңгейде айқындайды, яғни бағдарламалардың барлық функционалдық мүмкіндіктері дәлме-діл келтіріліп, олардың ең маңыздыларына сипаттама жүргізіледі. Бағдарламалық қамтамасыз етуді әзірлеудің мерзімдері мен құны айқындалады, бағдарламалық қамтамасыз етуді әзірлеуге техникалық тапсырма жасақталады және қол қойылады. Осылайша, бағдарламалық өнімді бұдан әрі жобалау үшін негіз құрылады.

*Жобалау сатысына*, әдетте, бағдарламалық жүйесі сәулетінің анықтамасы, оның функциялары, сыртқы қызмет ету талаптары, нинтерфейстері және пайдаланушылар мен жүйенің арасындағы функцияларды бөлу, бағдарламалық және ақпараттық компоненттерге қойылатын талаптар кіреді. Жүйені жобалау талаптарды қалыптастыру нәтижелерінің негізінде жүргізіледі. Жобалау әдістемесі жүйенің физикалық, логикалық, сондай-ақ динамикалық және статикалық модельдерін ұсыну тәсілдерін біріктіреді. Бағдарламалық қамтамасыз етудің функционалдық ерекшелігі әзірленеді, жүйенің сәулеті таңдап алынады,

деректер базасын басқарудың ең жарамды жүйесі (ДББЖ) айқындалады, деректерді сақтау құрылымы жобаланады, аппараттық қамтамасыз етуге талаптар ескеріледі, бағдарламалық қамтамасыз етуді енгізу үшін қажет ұйымдастырушылық іс-шаралар жиыны, сондай-ақ оны пайдалануды регламенттейтін құжаттар тізбесі айқындалады. Пайдалану құжаттамасы ресімделеді.

*Іске асыру сатысында* тұтас бағдарламалық жүйенің де, оның бөліктерінің де түптұлғасы қалыптасады, деректер құрылымдарын физикалық іске асыру жүзеге асырылады, бағдарламалық кодтар әзірленеді, ретке келтіру тестілеу орындалады, техникалық құжаттама жасалады. Іске асыру кезеңінің нәтижесінде өнімнің жұмыс нұсқасы пайда болады.

Бағдарламалық өнімді *тестілеу* жобалау және іске асыру сияқты, әзірлеу кезеңдерімен тығыз байланысты. Жүйеге арнаулы тетіктер кіріктіріледі, олар бағдарламалық қамтамасыз етуге талаптардың сәйкестігіне тестілеу жүргізуге, қажетті құжаттама пакетін ресімдеуді және оның болуын тексеруге мүмкіндік береді. Бағдарламалық өнімнің барлық кемшіліктерін жою және оның сапасы туралы тұжырымдама тестілеу нәтижесі болып табылады.

Бағдарламалық қамтамасыз етуді *пайдалануға қосу* (енгізу) әдетте бағдарламалық жүйені орнатуды, пайдаланушыларды оқытуды, құжаттандыруды (тиісті бұйрықтар, қабылдау актілерін және т.б. беруді) көздейді. Кез келген әзірлемеге құжаттаманың толық пакеті қоса беріледі, оған бағдарламалық өнімнің сипаты, пайдаланушылардың басшылықтары және жұмыс алгоритмдері кіреді.

Бағдарламалық қамтамасыз етудің қызмет етуін қолдау әзірлеушінің техникалық қолдау тобымен жүзеге асырылуға тиіс.

*Жобаны сүйемелдеу (пайдалану)* – жеткізілетін бағдарламалық қамтамасыз етудің жаңа талаптарға бейімделу, бағдарламалық қамтамасыз етуге және оның негізгі функцияларын өзгеріссіз сақтау кезінде түрлендіруде туындаған проблемалармен немесе қажеттіліктермен туындатылған тиісті құжаттамаға өзгерістер енгізу процесі.

Бағдарламалық қамтамасыз етуді пайдаланудан шығару оның моральдық ескіруінің, орнына неғұрлым жетілген өнім келуінің немесе өзге де объективті немесе субъективті себептер бойынша жүзеге асырылады.

Бағдарламалық қамтамасыз етудің тіршілік циклінің әр сатысының шекаралары кейбір уақыт шекараларымен белгіленді, онда белгілі бір сыни шешім қабылдау және демек, басты мақсаттарға жету қажет.

## 1.2. БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУДІ ӘЗІРЛЕУ СТРАТЕГИЯСЫ

*Каскадты стратегия* әзірленіп жатқан бағдарламалық құралға немесе әзірлеу үрдісінің басында жүйеге барлық талаптарды толық айқындауда негізделген және әзірлеу кезеңдерінің бер реттік өтуін білдіреді. Әзірлеудің әр кезеңі алдыңғы кезең аяқталғаннан кейін басталады. Орындалған кезеңдерге оралу көзделмейді. Әзірлеудің аралық өнімдері бағдарламалық өнімнің нұсқасы ретінде қолданылмайды.

Каскадты стратегия кез келген қолданбалы салаларда түрлі жүйелерді әзірлеуге сыныпсикалық тәсілді көрсетеді. Бағдарламалық қамтамасыз етуді әзірлеу үшін, бұл тәсіл 1970-інші жылдары және 1980-інші жылдардың бірінші жартысында кеңінен пайдаланылды. Каскадты стратегияның жетістіктеріне:

- әзірлеудің тіршілік циклі ішіндегі талаптардың тұрақтылығын;
- әзірлеу кезеңдерін бір рет ғана өту қажеттілігі, бұл стратегия қолданудың қарапайымдылығын қамтамасыз етуді;
- жобаны жоспарлау, бақылау және басқару қарапайымдылығын;
- тапсырыс берушілердің түсінуі үшін қол жетімділікті жатқызуға болады.

Каскадты стратегияны жобада пайдаланған кезде көрінетін негізгі кемшіліктері:

- әзірлеу үрдісінің басында талаптарды толық тұжырымдаудың күрделілігі және тіршілік циклінің бойында олардың динамикалық өзгеру мүмкіндігінің болмауы;
- әзірлеу үрдісі құрылымының түзулігі;
- аралық өнімдердің болмауы;
- талаптарды әзірлеу кезінде және қабылдау сынақтары уақытында ғана әзірлеу процесіне пайдаланушының аз қатысуы. Бұл пайдаланушының бағдарламалық қамтамасыз ету сапасын алдын ала бағалау мүмкін болмауына әкеліп келеді.

Осы тәсілді қолдану айқын, тіршілік циклінің ішінде өзгермейтін талаптармен және түсінікті іске асырумен және қиындығы жоғары емес жобаларды әзірлеген кезде барынша тиімді.

*Инкрементті стратегия* онда жүйенің әр бөліктері әртүрлі уақытта және түрлі қарқынмен әзірленетін және егер бір бөлігі дайын болса, онда оны жүйеге біріктіретін кезең-кезеңмен, уақытша кестелерді қолданатын стратегияны білдіреді.

Бұл стратегия әзірлеу үрдісінің басындағы әзірленіп жатқан бағдарламалық қамтамасыз етуге барлық талаптарды толықтай айқындауға негізделген. Инкрементті әзірлеу негізінде жатқан идея бағдарламалық жүйені әзірлеуші бағдарламалық қамтамасыз етудің бұрын жасалған нұсқаларын (релиздерін) әзірлеген кезде алынған деректерді пайдалануы үшін, өсім қағидасы бойынша әзірлеуді талап етуден тұрады.

Бағдарламалық қамтамасыз етудің ерекшелігі, жоба мен іске асыру бірінен кейін бірі әзірленетін бөліктерге (increment) бөлінеді. Осылайша, жүйенің бөліктерін қайта жасауға мұқтаж саны азаяды және клиенттер барынша ұзақ уақыт ішінде өз артықшылықтарын ойластыруға мүмкіндік алады. Әзірленетін өнімнің компоненттеры (бөліктері) пайдалануда болады, ол клиентке өнімге өзінің бұдан әрі қойылатын талаптарында үлкен айқындық алуға көмектеседі.

Инкрементті стратегияның негізгі құндылықтары:

- әр инкрементті іске асырудан кейін функционалдық өнімді алу мүмкіндігі;
- инкремент құрудың қысқа ұзақтығы; бұл бастапқы жеткізудің мерзімдерін қысқартуға алып келеді, бағдарламалық өнімді бастапқы және кейіннен жеткізуге шығындарды төмендетуге мүмкіндік береді;
- белгілі бір инкрементті құру уақытындағы талаптардың тұрақтылығы және өзгеріп жатқан талаптарды есепке алу мүмкіндігі;
- каскадты стратегиямен салыстырғанда тәуекелдердің төмендеуі болып табылады;
- процеске пайдаланушыларды қосу, бұл өнімнің функционалдық мүмкіндіктерін әзірлеудің неғұрлым ерте кезеңдерінде бағалауға мүмкіндік береді және осылайша бағдарламалық қамтамасыз етудің сапасын арттырады, сондай-ақ оны әзірлеуге уақыт пен шығындарды төмендетеді.

Инкрементті стратегияның негізгі кемшіліктеріне:

- инкременттерді жоспарлау мен жобаны басқаруды қамтамасыз ету үшін тіршілік циклінің басында жүйені немесе бағдарламалық құралды толықтай функционалдық айқындау қажеттілігін;
- жұмыстарды жоспарлау және бөлу қиындығын;
- қиын проблемаларды шешуді кеш инкременттерге созу беталысымен байланысты адам факторларының пайда болуын жатқызуға болады, ол жұмыс кестесін бұзуы немесе бағдарламалық өнімнің сапасын төмендетуі мүмкін.

Бұл стратегияны пайдалану мына жағдайларда барынша тиімді:

- талаптардың көбі алдын ала тұжырымдауға болатын жобаларды әзірлеу кезінде (алайда олардың бөлігі белгілі бір уақыт кезеңі арқылы нақтылануы мүмкін болады);

- нарыққа базалық функционалдық қасиеттері бар өнімді тез шығару қажет болғанда;
- төмен немесе орташа тәуекелдер дәрежесімен жобаларды әзірлеу кезінде.

Инкрементті стратегияны заманауи іске асыру экстремалды бағдарламалау болып табылады.

*Эволюциялық стратегия* әзірлеу кезеңдерін көп рет өтуді білдіреді. Осы стратегия әзірленетін бағдарламалық құралға немесе жүйеге әзірлеу процесінің басында талаптарды ішінара айқындауға негізделген. Талаптар әзірлеудің реттік циклдарында нақтыланады. Әзірлеудің әр циклінің нәтижесі әдетте бағдарламалық өнімнің кезекті жеткізілетін нұсқасын білдіреді.

Эволюциялық стратегияның негізгі құндылықтары:

- әзірлеу үрдісінде жаңа талаптарды нақтылау және енгізу мүмкіндігі;
- аралық өнімнің пайдалану үшін жарамдылығы;
- тәуекелдерді басқару мүмкіндігі;
- ерте кезеңдерден бастап жобада пайдаланушының кеңінен қатысуын қамтамасыз ету, бұл тапсырыс беруші мен әзірлеуші арасындағы келіспеушілік мүмкіндігін азайту және жоғары сапалы өнімді құруды қамтамасыз ету;
- каскадты және инкрементті стратегияның артықшылықтарын іске асыру болып табылады.

Оны тиісті таңдамаған кезде пайда болатын эволюциялық стратегияның кемшіліктеріне:

- қажетті итерацияның нақты санының белгісіздігі және келесі итерацияда әзірлеу үрдісін жалғастыру үшін критерийлерді айқындау күрделілігін; бұл жүйенің немесе бағдарламалық құралдың соңғы нұсқасын іске асыруды кідіруді тудыру мүмкін;
- жобаны жоспарлау және басқару күрделілігін;
- жобада пайдаланушылардың белсенді қатысу қажеттілігін, бұл нақты әрқашан жүзеге асырылмайды;
- қуатты аспаптық құралдар мен әдістерде түптұлғалау қажеттілігін;
- қиын проблемаларды шешуді кейінгі циклдарға жылжыту мүмкіндігін жатқызуға болады, бұл алынған өнімдердің тапсырыс берушілердің талаптарына сәйкес келмеуіне алып келуі мүмкін.

Эволюциялық стратегияның бірқатар кемшіліктері инкрементті стратегияға да сипатты екені айқын.

Осы стратегияны пайдалану талаптары тым күрделі, алдын ала белгісіз, тұрақсыз немесе нақтылауды талап ететін жобаларды әзірлеген кезде аса тиімді.

Эволюциялық стратегия тұжырымдаманы тексеру, техникалық жүзеге асушылығын немесе аралық өнімдерді көрсету қажет болғанда, орташа және жоғары тәуекел деңгейімен аналогтары жоқ үлкен ұзақ мерзімді жобаларды әзірлеген кезде, сондай-ақ жаңа технологияларды пайдаланатын жобаларды әзірлеген кезде ең жарамды.

## 1.3. ПӘНДІК САЛАНЫ САРАЛАУ ЖӘНЕ ЖОБАЛАУ

Пәндік саланы саралау бағдарламалық қамтамасыз етуді құруға дейін болды және оны әзірлеудің бөлігі болып табылады. *Пәндік сала моделі* дегенде зерттелетін пәндік саланың құрылымын немесе қызмет еуін имитациялайтын және осы салада барабар болу –и негізгі талапқа жауап беретін кейбір жүйе ұғынылады. Модель бағдарламалық қамтамасыз етудің қызмет етуінің барлық аспектілерін көрсетуі тиіс және бағдарламалық қамтамасыз етудің тіршілік циклінің барлық кезеңдерінде қажет. Бірақ пәндік сала моделінің айрықша ролі бағдарламалық өнімді оны құру кезінде талаптарды жасақтау сатысында ойнайды.

Пәндік саланы алдын ала модельдеу жобалау жұмыстарын жүргізудің уақыты мен мерзімдерін қысқартуға, аса тиімді және сапалы жоба алуға мүмкіндік береді. Пәндік салаға модельдеу жүргізусіз, өнімді кейін қайта жобалауға экономикалық ысыраптар мен жоғары шығындарға әкеліп соғатын, стратегиялық мәселелерді шешуде көптеген қателер болдырмау мүмкіндігі үлкен.

Пәндік сала моделіне мынадай талаптар ұсынылады:

- пәндік саланың құрылымын бір маңызды сипаттауды қамтамасыз ететін нысандандыру;
- моделді бейнелеудің графикалық құралдарын қолдану негізінде, тапсырыс берушілер мен әзірлеушілер үшін түсініктілік;
- пәндік сала моделін физикалық іске асыру құралдарының болуын болжайтын іске асушылық;
- белгілі бір әдістер мен есептелетін көрсеткіштердің негізінде пәндік саланың моделін іске асырудың тиімділігін бағалау мүмкіндігін қамтамасыз ету.

Аталған талаптарды іске асыру үшін моделдердің жүйесін құру қажет:

- пәндік саланың материалдық және ақпараттық объектілерінің процесінде өзара әрекет ететін құрамды көрсететін *объектілік*;
- үрдістерде нысандарды қайта құру бойынша функциялардың (іс-әрекеттердің) өзара байланысын көрсететін *функционалдық*;
- техникалық құралдар кешенінің орналасу топологиясын және коммуникация тәсілдерін сипаттайтын *техникалық*.

Бәлкім, үрдістердің орындалуына әсер ететін оқиғалар мен бизнес-ережелерді көрсететін *басқару моделі*, сондай-ақ ұйымдастырушылық бірліктердің өзара әрекет етуін көрсететін *ұйымдастыру моделі* қажет болады.

Пәндік сала моделінің барабарлығының басты критерийі әзірленетін бағдарламалық қамтамасыз етудің функционалдық толықтығынан тұрады.

Әдетте модельдер үш деңгейде құралады:

1) *сыртқы* (талаптарды айқындау);
2) *тұжырымдамалық* (талаптардың ерекшелігі);
3) *ішкі* (талаптарды іске асыру).

Сонымен, сыртқы деңгейде бағдарламалық қамтамасыз етудің негізгі компоненттерінің құрамы айқындалады: объектілер, функциялар, оқиғалар, ұйымдастырушылық бірліктері, техникалық құралдар. Тұжырымдамалық деңгейде жүйе компоненттеры өзара іс-әрекетінің сипаты айқындалады. Ішкі деңгейде модель көмегімен «жүйеге талаптар қандай бағдарламалық-техникалық құралдар көмегімен іске асырылады» деген сұраққа жауап беріледі.

Жобалау кезеңінің міндеті әзірленетін бағдарламалық қамтамасыз етудің егжей-тегжейлі ерекшелігін айқындау болып табылады. Бағдарламалық қамтамасыз етуді жобалау процесіне әдетте мыналар кіреді:

- жалпы құрылымды жобалау — негізгі бөліктерді (компоненттерді) және басқару және деректер бойынша өзара байланыстарды айқындау;
- блоктық-иерархиялық тәсілдің ұсынымдарына сәйкес компоненттерді бөлшектеп байланыстыру және құрылымдық иерархияны құру;
- компоненттерді жобалау.

Жобалау нәтижесі барлық деңгейдегі оның компоненттерінің ерекшеліктерімен бірге әзірленетін бағдарламалық қамтамасыз етудің егжей-тегжейлі моделі болып табылады. Модельдің түрі таңдалған немесе берілген тәсілден (құрылымдық, объектілік-бағдарланған) және нақты жобалау технологиясынан тәуелді.

Бірақ, кез келген жағдайда жобалау процесі бағдарламаларды (қосымша бағдарламаларды) жобалау және олардың арасындағы өзара байланысты анықтау, солай осы бағдарламалар немесе қосымша бағдарламалар өзара әрекет ететін деректерді жобалауды қамтиды.

Сондай-ақ, екі жобалау аспектісін айыру қолданылады:

1) келешек бағдарламалық өнімнің қызмет ету ортасын құрайтын қолданыстағы техникалық және бағдарламалық құралдарға тікелей тәуелді емес, сол жобалық операцияларды көздейтін *логикалық жобалау*;

2) қызмет ету ортасының нақты техникалық және бағдарламалық құралдарына бекітуді қамтитын *физикалық жобалау*;

Жобалау әдіснамалары, технологиялары мен аспапты құралдары кез келген бағдарламалық өнім жобасының негізін құрайды. Әдіснама нақты технологиялар және оларды қолдайтын стандарттар, әдістемелері және тіршілік циклінде процестерді орындауды қамтамасыз ететін аспапты құралдары арқылы іске асырылады.

*Жобалау технологиясы* жобаны әзірлеуге әкелетін олардың жүйелілігі мен өзара байланысында жобалаудың технологиялық операцияларының жиынтығы ретінде айқындалады. Жобалау технологиясын үш құрамның жиынтығы ретінде ұсынуға болады:

1) жобалаудың технологиялық операцияларының жүйелігін айқындайтын *қадамдық рәсім*;

2) технологиялық операцияларды орындау нәтижелерін бағалау үшін пайдаланылатын *критерийлер мен ережелер*;

3) жобаланатын бағдарламалық жүйесін сипаттау үшін пайдаланылатын *нотациялар* (графикалық және мәтіндік құралдар).

## БАҚЫЛАУ СҰРАҚТАРЫ

1. Бағдарламалық қамтамасыз етуді әзірлеудің базалық стратегияларын атаңыз.
2. Бағдарламалық қамтамасыз етуді әзірлеудің каскадты стратегиясының мәнін сипаттаңыз.
3. Каскадты стратегияның құндылықтарын, кемшіліктері мен қолдану саласын атаңыз.
4. Бағдарламалық қамтамасыз етуді әзірлеудің инкрементті стратегиясының мәнін сипаттаңыз.
5. Инкрементті стратегияның құндылығы, кемшіліктері мен қолдану саласы қандай?
6. Бағдарламалық қамтамасыз ету стратегиясының эволюциялық стратегиясының мәнін сипаттаңыз.

7. Эволюциялық стратегияның қандай құндылықтарын, кемшіліктері мен қолдану салаларын сіз білесіз?
8. Бағдарламалық қамтамасыз етуді әзірлеудің каскадты, инкрементті және эволюциялық стратегиясына салыстырмалы сипаттама беріңіз.
9. Каскадты стратегияны қандай жағдайларда аса тиімді пайдаланады?
10. Инкрементті стратегияны қандай жағдайда аса тиімді пайдаланады?
11. Эволюциялық стратегияны қандай жағдайда аса тиімді пайдаланады?
12. Бағдарламалық қамтамасыз етудің тіршілік циклінің әр кезеңдерін сипаттаңыз.
13. Бағдарламалық қамтамасыз етудің тіршілік циклінің аса маңызды кезеңдерін атап өтуге бола ма?

# 2 ТАРАУ

# ОБЪЕКТІЛІК-БАҒДАРЛАНҒАН БАҒДАРЛАМАЛАУДЫҢ НЕГІЗДЕРІ

## 2.1. БАҒДАРЛАМАЛАУДА ОЪЕКТІЛІК-БАҒДАРЛАНҒАН ӘДІСТІҢ МӘНІ

Бағдарламалауға объектілік тәсіл 1980-інші жылдардың ортасынан бастап 1990-ыншы жылдардың аяғына дейін қалыптасты. *Объектілік-бағдарланған бағдарламалау* олардың әрқайсысы белгілі типті (сыныпты) данасы болып табылатын, ал сыныптары қасиеттерді иеленумен иерархия құрайтын объектілердің жиынтығы түрінде бағдарламаларды беруге негізделген күрделі бағдарламалық қамтамасыз етуді құру технологиясы ретінде айқындалады. Бағдарламалық объектілердің өзара іс-әрекеті хабарламаларды беру арқылы жүзеге асырылады.

Модульдік бағдарламалаумен салыстырғанда объектілік-бағдарланған бағдарламалаудың негізгі құндылықтары – оның әзірлеуін елеулі жеңілдететін бағдарламалық қамтамасыз етуді неғұрлым табиғи бөлшектеп байланыстыру. Бұдан басқа, объектілік тәсіл иелену, полиморфизм, композиция тетіктеріне негізделген бағдарламаларды ұйымдастырудың негізгі тәсілдерін ұсынады. Бұл кодтарды қайталап пайдалану көрсеткіштерін маңызды ұлғайтуға және түрлі қолданулар үшін сыныптар кітапханаларын құруға мүмкіндік береді.

Бағдарламалау технологиясында объектілік тәсілдің дамуы көрнекі бағдарламалау ортасын құруға алып келді. Delphi, C+ + Builder, Visual C++, C# және т.б. сияқты көрнекі объектілік-бағдарланған бағдарламалау тілдері пайда болды. Алайда, объектілік-бағдарлаған бағдарламалау технологиясының кемшіліктері де бар. Олардың бастысы – бағдарламалық қамтамасыз ету модульдерінің деректердің экспортталатын өрістері мен әдістерінің, құрылымдары мен форматтарының мекенжайынан тәуелділігі. Бұл тәуелділік объективті, неге десең модульдер бір-бірінің ресурсына жүгініп, өз арасында өзара іс-қимыл жасауы тиіс.

Кез келген бағдарламалық жүйелер нақты жүйелерді модельдеуге арналған, сондықтан, біз бұл нақты жүйелерді қандай терминдерде сипаттауға тырысып отырғанымыз өте маңызды. Іс-қимылдардың жүйелілігі түрінде сипаттау (бағдарламалауға рәсімдік тәсіл) жеткілікті қиын болды. Объектілік-бағдарланған бағдарламалаудың негізгі ұғымы «объект» болып табылады. Негізінен, мұндай тәсіл жеткілікті табиғи болып көрінеді, өйткені шынайы өмірде біз бір-бірімен өзара әрекет ететін объектілермен (заттармен, құрылғылармен, адамдармен) істі боламыз. Пайдаланушылардың компьютерлік бағдарламамен өзара іс-қимылы – бұл да екі объектінің: бағдарламаның және бір-бірімен хабарламалармен алмасатын адамның өзара іс-қимылы.

Объектілік-бағдарланған бағдарламалау объектісі – бұл бағдарламалық жүйедегі нақты мәннің моделі немесе абстракциясы. Объектілік-бағдарланған бағдарлама – объектілер мен олардың өзара іс-қимыл тәсілдерінің жиынтығы.

Объектілік-бағдарланған бағдарламалау әдістемесін пайдаланумен шешілетін міндет объектілер мен олардағы операция терминдерінде сипатталады.

Объектілік-бағдарланған бағдарламалау тілінде жазылған кез келген бағдарлама өзінің деректерінде физикалық заттардың не абстракциялық түсініктердің – ол олармен жұмыс істеуге арналған бағдарламалау объектілерінің жай-күйін көрсетеді. Объектілік-бағдарланған бағдарламалауда әр объект өзінің жай-күйімен сипатталады. Объектінің жай-күйі атрибуттарының ағымдағы мәнімен (деректерімен) сипатталады. Объектілік-бағдарланған бағдарламалауда объектінің атрибуттары қарапайым маңыздары (сан, логикалық маңызы және т.б.), солай күрделі шамалар немесе басқа да объектілер болуы мүмкін. Бұны негізіне ала отырып, бағдарламалау объекті және оның басқа объектілермен байланысы туралы барлық деректерді объекті деп аталатын, бір құрылымдалған тұрақсыз шамаға біріктіруге болады. Деректерді (объекті сипаттамаларын) жазу алаңы ретінде қарауға болады. Бағдарламаның пайдаланушылары мен объектілерінің объектінің бұл деректерін оқуға, объектіге жаңа маңыздарды қандай да бір түрде өңдеп, жазу мүмкіндігі болуы тиіс.

Объектімен өзгеше *әдістер* деп аталатын іс-қимылдар жиыны байланысады. Бағдарламалау тілі тұрғысынан іс-қимылдар немесе әдістер жиыны – бұл объектіге міндетті параметр ретінде көрсеткіш алатын және бағдарламалау объектісінің деректерімен белгілі бір іс-қимылдарды орындайтын функциялар.

Объектілік-бағдарланған бағдарламалау технологиясы объектімен тек оның әдістері арқылы ғана және өзгеше ешқалай емес жұмыс істеуге мүмкіндік береді; осылайша, объектінің ішкі құрылымы сыртқы пайдаланушыдан жасырылған. Пайдаланушыға деректерге тікелей қол жетмділікке тыйым салынған және бұл мынадай себептер бойынша жасалады:

1. Объектінің сенімді қызмет етуі үшін оның деректерінің тұтастығы мен қайшылықсыздығын қолдау қажет. Егер мұны қолға алмаса, онда сыртқы объекті немесе пайдаланушы деректерді бұзуы немесе дұрыс емес енгізуі мүмкін және объект қателермен қызмет ететін болады.
2. Сыртқы объектілерді деректерді ішкі іске асыру ерекшеліктерінен оқшаулау қажет. Деректердің сыртқы тұтынушылары үшін, тек пайдаланушылық интерфейс - қандай деректер мен функциялар болуының, соның және оларды қалай пайдалануға болатынының сипаты қол жетімді, ал ішкі іске асыру – бұл объекті әзірлеушісінің ісі болуы тиіс. Мұндай жағдайда әзірлеуші объектіні әрқашан жаңғыртуы, сақтау құрылымы мен деректерді ұсыну нысанын өзгертуі мүмкін, алайда, бұл ретте, егер сыртқы интерфейстер туралы сөз болмаса, пайдаланушы мұны байқамай да қалады. Және, демек, бағдарламада және пайдаланушының іс-әрекеттерінде ештеңе өзгертілмейді.

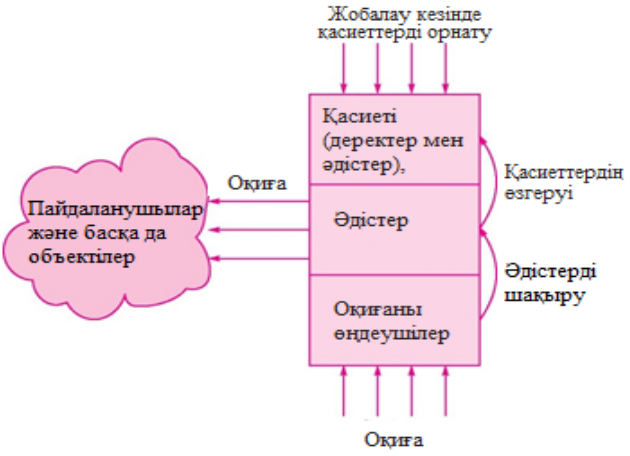
Деректерді жасыруды қамтамасыз ету үшін, объектіде деректермен барлық қажетті операцияларды (оқу, өзгерту, жазу) іске асыратын рәсімдер мен функциялар айқындалады. Осы рәсімдер мен функциялар (әдістер) арқылы объектінің деректерімен қатынас болады. Құрастырушы бағдарламаның қажетті жерлеріне осы әдістерді шақыруларды қояды. Осылайша, объектілердің өзара іс-қимылы (объектілер арасында хабарлама алмасу) негізінде сол немесе өзге объекті әдісін шақыруды қамтиды. Деректер жиынтығы мен оларды оқу және жазу әдістерін *қасиет* деп атайды. Қасиетті жобалау процесінде орнатуға, бағдарламаны орындау уақытында өзгертуге болады. Жекелеген деректерде жұмыс істейтін әдістерден басқа объектіде барлық олардың жиынтығымен жұмыс істейтін, олардың құрылымын өзгертетін әдістер бар. Осылайша, объектті қасиеттер мен әдістер жиынтығы ретінде айқындауға болады.

Объектілердің өзара іс-қимыл ортасы түрлі оқиғалар нәтижесінде жасалатын хабарлама болып табылады. *Оқиға* пайдаланушының іс-әрекетінің салдарынан (тінтуір немесе перне тақтаны басу, тінтуір курсорын орналастыру) басталады. Сондай-ақ оқиға объектілердің жұмысы нәтижесінде басталуы да мүмкін. Әр объектіде ол ден қоя алатын көптеген оқиғалар айқындалған. Объектілердің нақты даналарында осы оқиғаларды өңдеушілер айқындалуы мүмкін, олар объектінің осы данасының реакциясын қамтамасыз етеді. Бұдан бұрын айтылғандарды ескеріп, объектіге мынадай айқындама беруге болады.

*Объект* — қасиеттер мен әдістер, сондай-ақ ол ден қоя алатын оқиғалар жиынтығы. Объектінің жағдайы осы объектінің барлық ықтимал статикалық қасиеттерінің тізбесімен және әр осы қасиеттердің ағымдағы маңыздарымен (динамикалық) сипатталады. Мінез-құлық объектінің басқа объектілерге ықпалын және керісінше, осы объектілердің жай-күйінің өзгеруіне қатысты сипаттайды. Басқаша айтқанда, объектінің мінез-құлқы толықтай оның әрекеттерімен айқындалады.

Бұл сипаттау *объектінің интерфейсі* деп аталады. Объектілік-бағдарланған бағдарламалауда объектінің маңызды сипаттамасы ол қоршаған ортамен қалай өзара әрекет ететінін сипаттау болып табылады.

Объектіні ұйымдастырудың шартты схемасын мынадай үлгіде (2.1-сурет) ұсынуға болады. Объектіні сырттай басқару оқиғаны өңдеушілер арқылы жүзеге асырылады. Бұл өңдеушілер объектінің қасиеті мен әдістеріне жүгінеді.




2.1- сурет. Объектіні ұйымдастыру схемасы

Бұл объектінің бастапқы мәнінде жобалау процесінде түрлі қасиеттер орнату ұсынылуы мүмкін. Объектінің әдістерін орындау нәтижесінде бағдарламаның басқа объектілері немесе пайдаланушы қабылдайтын жаға оқиғалар өндірілуі мүмкін.

*Даралық* – бұл барлық басқа объектілерден оны ерекшелейтін объектінің қасиеті. Кейде объектілердің объективтік-бағдарланған бағдарламалауға біріздендіру қажет. Егер екі объекті болса, онда осы объектілер әр түрлі екенін қалай да айқындау қажет. Әдетте біріздендіру үшін объектілердің арнайы атрибуттары – *сәйкестендіргіш* қолданылады.

Кез келген жүйеде объекті құрылады, жұмыс істейді және жойылады бағдарламалауда объектілерді жоюдың екі тәсілі бар.

1. Объектілер арнайы шақырулардың көмегімен анық жойылу қажет.
2. Объектілер олар ешкімге қажет болмаған кезде жойылады (жүйеде осы объектіге сілтемелер болмайды) мұндай жоюды кейде қол жеткізушілікке қарай жою деп атайды.

Сонымен қатар бағдарламаны жаңадан іске қосқанда олардың алдыңғы жай-күйін қалпына келтіру қажет болатын объектілер болуы мүмкін. Мұндай объектілермен қызмет ету үшін барлық атрибуттар мәні жазылатын және қажет болғанда объектіні қалпына келтіру оқылатын *сериалдау әдісі* қолданылады.

Барлық объектілерге оларды құру және жою әдістері – конструкторлар және тиісінше деструкторлар салынды.

*Конструктор* — объектіні құрғаннан кейін бірден орындалатын арнайы әдіс. Конструктор объекті алаңын жүктейді – объектіні бастапқы жағдайға келтіреді. Конструкторлар параметрмен және параметрсіз де болады. Параметрсіз конструкторды тек қана біреу болатын әдеттегі конструктор деп атайды.

*Деструктор* —объекті жедел жадыдан жойылған кезде бағдарламаны орындау ортасымен шақырылатын арнайы әдіс.

Бастапқыдан қолданбалы бағдарламаға қатысуы тиіс объектілер конструкторлары бағдарламаны қосқан кезде іске қосылады. Деструкторлар жұмысты аяқтаған кезде іске қосылады.

Бағдарламаны аяқтаған кезде жойылатын тұрақты объектілер де бар және оны қосқан кезде қайта құрылмайды (объектілік-бағдарланған деректер базасының объектілері). Бағдарлама бірінші және екінші қосылғанда тұрақты жадыда сақталған сол бір объектіге ғана жүгінеді.

Объектілік-бағдарланған бағдарламалауда объекті басқа объектілерден де тұруы мүмкін. Бұл *объектілерді композициялау* деп аталады.

Объектілерді бағдарламаға тиісті операторлар, не компоненттерді пайдаланып, көрнекі бағдарламалау көмегімен қосуға болады.

Кез келген жүйеде көптеген объектілер бар. Олардың кейбіреуі «ұқсас» және бір типті. Бір типті объектілер *сыныптарға* бірігеді. Бір және сол сыныптың барлық объектілерінің бірдей интерфейсі болады және бұл интерфейсті бір және сол тәсілмен іске асырады. Объектілік-бағдарланған бағдарламалауда бір сыныптың екі объектісі тек ағымдағы жай-күйімен ғана ерекшелене алады.

Бір сыныпты барлық объектілерде әдістерді іске асыру бірдей. Объектілер *сынып данасы* деп аталады. Ал объектілік-бағдарланған бағдарламалауда сынып - объекті құрылатын шаблон, яғни, сынып – бұл объекті құрылымын және олармен жұмыс әдістерін сипаттау деп айтуға болады.

*Интерфейс* — бұл сыныптың сыртқы бөлігі. Интерфейс осы сыныптың объектілері осы немесе басқа сыныптармен қалай өзара жұмыс істейтінін айқындайды. Алайда егер екі объектіде интерфейстер сәйкес келсе, онда олар бір немесе сол сыныпқа тиесілі дегенді білдірмейді. Интерфейстердің сәйкес келуінен басқа оларды іске асыру және мінез-құлқы бірдей болуы қажет. Тип —қандай да бір шаманы, яғни оның ықтимал мәндерінің мөлшері және қолданылатын операциялар жиынын айқындау саласы. Тип сынып болып ұсынылуы мүмкін.

## 2.2. ОБЪЕКТІЛІК-БАҒДАРЛАНҒАН БАҒДАРЛАМАЛАУДЫҢ ҚАҒИДАЛАРЫ

Объектілік-бағдарланған бағдарламалау үш маңызды қағидаларға негізделген. Бұл қағидалар қапшықтану, мұралау және полиморфизм деп аталады.

*Инкапсуляция* —объектіні максималды дәрежеде сыртқы ортадан оқшаулауға мүмкіндік беретін осы деректерді өңдеу алгоритмдері мен объектілер деректерін бірыңғай тұтас біріктіру.

Бұдан бұрын объектілік-бағдарланған бағдарламалау аясында деректер объекті алаңы немесе атрибуты, ал алгоритмдер – объекті нысаны аталатынын айтылған.

Әзірленіп жатқан бағдарламалардың сенімділігін арттырады, себебі алгоритм объектісінде шектелген алгоритмдер салыстырмалы түрде деректері шағын көлемді бағдарламамен алмасады. Әрі бұл деректердің мөлшері мен типі әдетте тыңғылықты бақыланады.
Объектіге қапшықтанған алгоритмдер мен деректерді ауыстыру немесе үлгілеу нәтижесінде, әдетте, тұтастай бағдарлама үшін нашар бақыланатын салдарды қамтымайды (объектілік-бағдарланған бағдарламалауда бағдарламалардың қорғанысын арттыру мақсатында жаһандық өзгеріс пайдаланылмайды).

Инкапсуляция пайдаланушыға пайдаланып жатқан компонентті іске асырудың күрделілігі туралы ойланбауға, онымен ұсынылған интерфейс арқылы өзара әрекет етуге, сондай-ақ компонент үшін өмірлік маңызды деректерді біріктіруге және қорғауға мүмкіндік береді. Бұл ретте пайдаланушыға объектінің ерекшелігі (интерфейс) қана беріледі. Пайдаланушы объектімен осы интерфейс арқылы ғана өзара әрекет ете алады және жабық деректер мен әдістерді пайдалана алмайды.

Жабық кодтар немесе деректер осы объектінің басқа бөліктері үшін ғана қолжетімді және объектіден тыс бағдарламалардың бөліктері үшін қол жетімсіз. Егер кодтар мен деректер ашық болып табылса, онда олардың объектінің ішінде берілгеніне қарамастан, олар бағдарламаның басқа бөліктері үшін де қолжетімді. Объектінің ашық бөлігі объектінің жабық элементтерінің бақыланатын интерфейсін қамтамасыз ету үшін пайдаланылса, жағдай сипатты болып табылады.

Инкапсуляцияның басқа да маңызды салдары объектімен алмасу, олардың бір бағдарламадан екіншісіне ауысу жеңілдігі болып табылады.

Объектілік-бағдарланған бағдарламалау тұжырымдамасы бар сыныптарға алаңдар, қасиеттер мен әдістер қосу арқылы жаңа сыныптарды айқындауға мүмкіндікті болжайды. Жаңа сыныптарды алудың мұндай тетігі туындатқыш деп аталады. Бұл ретте жаңа, туындатқыш сынып (ұрпақ) өзінің базалық, аталық сыныптарының қасиеті мен әдістерін мұралайды.

*Мұралану* – бар (аталық) сынып негізінде жаңа сыныпты сипаттауға мүмкіндік беретін тетік, бұл ретте аталық сыныптың қасиеті мен функционалдығы жаңа сыныппен ауысады.

Басқа сөзбен айтқанда, мұрагер сынып қолданыстағы сыныптың ерекшелігін іске асырады. Бұл мұрагер сыныптың объектілерімен базалық сынып объектілерімен сияқты араласуға мүмкіндік береді. Мұралау жүргізілген сынып базалық немесе аталық деп аталады. Базалықтан өткен сыныптар ұрпақ, мұрагер немесе туынды сынып аталады.

Мұралау – өзінің ұрпақтарын туындататын объектілердің қасиеті. Ұрпақ-объекті аталықтан өзінің алаңдары мен әдістерін автоматты түрде мұраға алады, объектілерді жаңа алаңдар мен аталық әдістерді ауыстырып немесе оларды толықтыра алады.

Мұралау қағидаты объекті қасиеттерін түрлендіру проблемасын шешеді және объектілік-бағдарланған бағдарламалауға тұтастай ерекше икемділік береді. Объектілермен жұмыс кезінде бағдарламашы нақты міндетті шешу үшін өз қасиеттеріне барынша жақын объектіні таңдайды және одан аталықта іске асырылмағанды «жасай алатын» бір немесе бірнеше ұрпақты құрады.

*Полиморфизм* — түрлі сыныптарға кіретін әдістер үшін бірдей атауды пайдалану мүмкіндігі (яғни, бірдей ерекшелікпен түрлі іске асыру жағдайының болуымен объектілердің мүмкіндігі). Полиморфизм тұжырымдамасы объектіге әдіс қолданған жағдайда, объект сыныбына сәйкес келетін объектіні пайдалануды қамтамасыз етеді.

Мысалы, сыныпты іске асыру мұралау процесінде өзгеруі мүмкін. Полиморфизм неғұрлым абстрактылы бағдарламаларды жазуға және кодты қайта пайдалану коэффициентін арттыруға мүмкіндік береді. Объектілік-бағдарланған бағдарламалау аясында объектінің мінез-құлық қасиеті оған кіретін әдістер жиынымен айқындалады. Объекті ұрпағында сол немесе өзге алгоритмді өзгерте отырып, бағдарламашы бұл ұрпақтарға аталықта жоқ ерекше қасиетті бере алады. Әдісті өзгерту үшін оны ұрпақта бөгеп, яғни ұрпақта бір атаулы әдісті жариялап, онда қажетті іс-әрекеттің іске асыру қажет.

Нәтижесінде аталық объектіде және ұрпақ объектіде түрлі алгоритмдік негізі бар және ізінше, объектіге түрлі қасиет беретін екі бір атаулы әдіс әрекет ететін болады. Объектілердің полиморфизмі деген осы.

Осылайша, объектілік-бағдарланған бағдарламалауға сәйкес полиморфизмнің мақсаты сынып үшін ортақ іс-қимыл беру үшін бір атауды пайдалану болып табылады. Әрбір нақты іс-қимылды орындау деректер типімен айқындалады.

## 2.3. DELPHI ОБЪЕКТІЛІК-БАҒДАРЛАНҒАН БАҒДАРЛАМАЛАУДЫҢ МЫСАЛЫ

Delphi-ге объектілік-бағдарланған бағдарламалау мүмкіндігі Object Pascal тілі қасиеттеріне негізделеді. Қосымшаны бағдарламалау модульдік қағидатқа негізделген. *Модуль*

(unit) — бұл константт, типтер, айнымалы, рәсімдер мен функциялардың атаулы жиынтығы. Қосымша бағдарламадан (рәсімдер немесе функциялар) ерекшелігі модуль негізгі бағдарламадан жеке сақталады және одан тәуелсіз құрастырылады. Негізінен модуль орындалатын бағдарлама болып табылмайды – оның объектілерін басқа бағдарламалық бірліктер пайдаланады.

Бас бағдарлама шекті қарапайым және қысқа болады. Ол қосымшада бар нысандағы объектілерді құратын пайдаланылатын модульдер мен бірнеше операторлар тізімін жариялаудан тұрады және қосымшаны орындалуға іске қосады.

Компоненттердің барлық объектілері нысан-объектілерде орналастырылады. Delphi әр нысаны үшін жеке модуль құрады. Сол модульде бағдарламалау жүзеге асырылады. Одан әрі бос нысанмен модуль мәтіні келтірілген: unit Unitl;

```
   interface // модульдің ашық интерфейсі
   {қосылатын модульдердің тізімі}
   uses
    Windows, Messages, SysUtils, Variants, Classes,
Graphics,
   Controls, Forms, Dialogs;
   {хабарландыру сыныбы нысаны} type
    TForm1 = class (TForm)
    Private // сыныптың жабық бөлімі
    { Private declarations }
   {Мұнда нысан сыныбына кіретін, бірақ басқа
модульдерге қол жетімсіз айнымалы, функциялар мен
рәсімдер жарнамасы орналастырылады} public
    { Public declarations }
   { Мұнда нысан сыныбына және басқа да модульдер
үшін қол жетімді айнымалы, функциялар мен рәсімдер
жарнамасы орналастырылады} end; var
    Form1: TForm1;
   { Мұнда басқа модульдерден қол жетімділік
болатын, бірақ нысан сыныбына қосылмайтын типтер,
айнымалы, функциялар мен рәсімдер орналастырылады}
Implementation //Нысанды іске асыру
```

```
{$R *.dfm}
{ Мұнда uses қосымшасы, типтер, айнымалы, функциялар мен басқа модульдерден қолжетімділік болмайтын рәсімдер жарнамасы орналастырылады} end.
```

Модуль мәтіні екі негізгі бөлімнен тұрады: interface –модульдің ашық интерфейсі және implementation — модульді іске асыру. Interface бөліміне тікелей орналастырылатынның барлығын бағдарламаның басқа да модульдері пайдалана алады. Implementation бөліміне тікелей орналастырылатынның барлығы – модульдің ішкі ісі. Сыртқы модульдер іске асыру бөліміне орналастырылған типтер, айнымалы, функциялар мен рәсімдердің көре алмайды.

Object Pascal сыныбы деп өзінің құрамында алаңы, әдістері мен қасиеттері бар ерекше құрылымды атайды. Мұндай тип объектілік тип деп те аталады:

```
type
TMyObject = class(TObject) MyField: Integer;
Function MyMethod: Integer;
end;
```

Кез келген деректер типін бағдарламада пайдалану үшін осы типтік кем дегенде айнымалысын жариялау қажет. Объектілік типтің айнымалысы сынып данасы немесе объекті деп аталады:

```
var
AMyOb j ect:  TMyObj ect;
```

`TmyObject` сипатталған сыныбының `MyField` – бір алаңы бар. Алаңнан ерекшелігі бір сыныптың объектілерінің әдістері ортақ. Әдістер - сынып ішінде сипатталған және оның алаңдарындағы операцияларға арналған рәсімдер мен функциялар.

Сынып құрамына әдістерді шақыру үшін қажетті барлық ақпарат бар арнайы кестеге көрсеткіш кіреді. Қалыпты рәсімдер мен функциялардан әдістер оларға шақырған кезде оларды шақырған объектіге көрсеткіш берілетіндігімен ерекшеленеді.

Сондықтан әдіс шақырған объектінің алаңы пысықталады. Әдістің ішінде оны шақырған объектіге `self` резервтелген атаумен қолжетімді.

Одан әрі келтірілген мысалда `TPerson` қарапайым сыныбының жарнамасы — бұл сынып атауы, `fname` және `faddress` — алаңдар атауы, show — әдіс атауы:

```
TPerson = class
private
```

```
fname: string[15]; faddress: string[35];
public
procedure Show;
end;
```

Сыныпты сипаттауды бағдарламаға типтерді (type) сипаттау бөліміне орналастырылады. Объектілер сынып өкілдері ретінде var бөлімінде жарияланады, мысалы

```
var
student: TPerson; professor: TPerson;
```

Сыныптар модуль интерфейсінің (interface) секциясында немесе іске асыру (implementation) секциясы тіркемесінің жоғарғы деңгейінде сипатталуы мүмкін.

Объект объект-конструкторды жүктейтін арнайы әдісті шақыру көмегімен құрылады. Құрылған дана басқа әдіс – деструктормен жойылады:

```
AMyObject:= TMyObject. Create;
{құрылған объектімен іс-қимыл}
AMyObject.Destroy;
```

Объект құрылғанға дейін сәтті жұмыс істейтін әдістер (оның ішінде конструктор) бар. Олар сыныптың әдістері болып табылады.

В Object Pascal экземпляры объектов могут быть только динамическими. Это означает, что в приведенном ранее фрагменте переменная AMyObject на самом деле является указателем, содержащим адрес объекта.

Object Pascal-да сыныпта конструкторлар бірнеше болуы мүмкін. Көпшілік конструкторды Create, ал деструкторды — Destroy деп қабылдаған:

```
type
TMyObject = class(TObject)
MyField: Integer;
Constructor Create;
Destructor Destroy;
Function MyMethod: Integer;
end;
```

Объектінің данасын жою үшін бастапқыда көрсеткішті (ол nil-ге тең бе), содан кейін ғана Destroy-ді шақыратын Free әдісін пайдалану ұсынылады:

```
AMyObject.Free;
```

Конструктор егер оның алдында сынып атауы көрсетілген жағдайда ғана объектінің жаңа данасын құрады. Егер қолданыстағы объектінің атауын көрсетсе, ол өзін басқаша ұстайды, жаңа дананы құрамайды, тек конструктор денесінде бар кодты орындайды).

Сонымен бірге мысал ретінде құрамына конструктор енгізілген TPerson сыныбы сипаттамасын келтіруге болады:

```
TPerson = class private
fname: string [ 15 ] ;
faddress: string[35];
constructor Create; // конструктор
public
procedure show; // әдісі
end;
```

Енді объектілік-бағдарланған бағдарламалау мәнін құрайтын қағидаларды (инкапсуляция, мұралау және полиморфизм) қолдануды қарастырамыз.

Объектілік-бағдарланған бағдарламалаудың классикалыққ қағидасы сенімділікті қамтамасыз ету үшін объектінің алаңына тікелей қол жетімділік қажетті емес: оның ішіндегіні оқу мен жаңарту тиісті әдістерді (инкапсуляция қағидасы) шақыру арқылы жүргізіледі.

Әдетте қасиет үш элементпен: алаңмен және оны оқуды/жазуды жүзеге асыратын екі әдіспен айқындалады.

```
TAnObject = class(TObject)
function GetAProperty: TSomeType;
procedure SetAProperty (ANewValue: TSomeType);
property AProperty: TSomeType read GetAProperty
write SetAProperty;
end;
```

Бұл мысалда AProperty қасиетінің мәніне қол жетімділік GetAProperty және SetAProperty әдістерін шақыру арқылы жүзеге асырылады. Алайда бұл әдістерге жүгінудің аса қажеттілігі жоқ, жазса жеткілікті:

```
AnObject.AProperty := AValue;
AVariable := AnObject. AProperty;
```

Сырттай қасиет нақты қалыпты алаң сияқты көрінеді, бірақ оған әр қалай жүгінгендіктен қажетті іс-қимыл тұруы мүмкін. Мысалы, егер сізде экранда тік бұрышты қамтитын объект болса, оның «түс» қасиетіне «ақ» мәнін берсеңіз, онда экрандағы нақты түсті қасиет мәніне сәйкестікке келтіретін дереу жаңа сурет пайда болады.

Мұралау деген егер сіз бұрынғыдан сәл ерекшеленетін жаңа сынып құрғыңыз келсе, онда бар алаңдар мен әдістерді қайта жазу қажеттігі мүлдем жоқ екендігін білдіреді. Жаңа сынып жарнамасында ол баба немесе аталық сынып аталатын ескі сыныптың ұрпақ немесе еншілес сыныбы болып табылатынын көрсетеді:

```
TNewObject = class(TOldObject);
```

Оған жаңа алаңдар, әдістер мен қасиеттер қосылады. Объектілік-бағдарланған бағдарламалау тұжырымдамасы қолданыстағы сыныптарға алаңдарды, қасиеттерді және әдістерді қосу арқылы жаңа сыныптар айқындау мүмкіндігін бағамдайды. Жаңа сынып алудың мұндай тетігі туындатқыш деп аталады. Бұл ретте жаңа, туынды сынып (ұрпақ) өзінің базалық, аталық сыныптың қасиеттері мен әдістерін мұралайды. Ұрпақ сыныбының жарнамасында аталық сыныбы көрсетіледі.

Object Pascal-да барлық сыныптар Tobject сыныбының ұрпағы болып табылады. Сондықтан егер сіз TObject тікелей еншілес сынып құрсаңыз, онда айқындамада оны ескертпеуге болады. Келесі екі мән бірдей дұрыс:

```
TMyObject = class(TObject);
TMyObject = class;
```

Бірінші нұсқа ол неғұрлым ұзын болса да ықтимал бір мәнді еместікті жоюға ыңғайлырақ.

Баба сыныптан мұраланған алаң мен әдістерге еншілес сыныпта қолжетімді. Егер әдістердің атаулары сәйкес келген орын болса, онда олар жабылады. Аталық сыныпта жарияланған қасиеттер мен әдістерді жоюға болмайды.

Мысалы `TEmployee` (қызметкер) сыныбы `FDepartment` (бөлім) алаңын қосу арқылы бұрын қаралған `TPerson` туындаған.

`TEmployee` сыныбының жарнамасы бұл жағдайда мынадай болып көрінеді:

```
TEmployee = class(TPerson)
FDepartment: integer; // бөлім нөмірі
constructor Create(Name:TName; Dep:integer);
end;
```

Жақшаға алынған `TPerson` сыныбының атауы `TEmployee` сыныбы `TPerson` сыныбының туындысы болып табылатынын көрсетеді. Өз кезегінде, `TPerson` сыныбы `TEmployee` сыныбы үшін базалық болып табылады.

TEmployee сыныбы аталық сынып пен өз алаңдарын жүктеуді қамтамасыз ететін өзінің жекеменшік конструкторы болуы тиіс. Міне TEmployee сыныбының конструкторын іске асыру мысалы:

```
   constructor TEmployee.Create (Name: TName; Dep:
integer);
    begin
   inherited Create(Name);
   FDepartment:=Dep;
   end;
```

Келтірілген мысалда inherited директивімен аталық сыныптың конструкторы шақырылады. Осыдан кейін ұрпақ сынып алағына мін беріледі.

Полиморфизм — түрлі сыныпқа кіретін әдістер үшін бірдеу атауды пайдалану мүмкіндігі. Полиморфизм тұжырымдамасы объектіге әдісті қолданған жағдайда объектінің сыныбына сәйкес келетін әдісті пайдалануды қамтамасыз етеді.

Үш сынып айқындалсын, олардың біреуі басқа екуі үшін базалық болып табылады:

```
type
// TPerson базалық сыныбы = class
fname: string; // атауы
constructor Create(name:string) ;
function info: string;
virtual;
end;
// туындысы TPerson TStud = class(TPerson)
fgr:integer; // оқу тобының нөмірі constructor
Create(name:string;gr:integer); function info:
string; override; end;
// туынды TPerson TProf = class(TPerson)
fdep:string; // кафедра атауы
constructor Create(name:string;dep:string);
function info: string;
override;
end;
```

Осы сыныптардың әрқайсысында info әдісі айқындалды. Virtual директиві көмегімен базалық сыныпта info әдісі виртуалдық деп жарияланды. Әдісті виртуалдық деп жариялау еншілес сыныпқа өзінің меншікті әдісін виртуалдықпен ауыстыруға мүмкіндік береді.

Әр еншілес сыныпта аталық сыныптың (аталық сыныптың виртуалдық әдісін алмастыратын туынды сыныптың әдісі override директивімен белгіленеді) тиісті әдісін алмастыратын өзінің info әдісін айқындады.

Одан әрі әр сынып үшін info әдісінің айқындамасы келтірілген.

```
function TPerson.info:string;
begin
result : =     '' ;
end;
function TStud.info:string;
begin
result  := fname +  ' rp.' + IntTostr(fgr);
end;
function TProf.info:string;
begin
result  := fname +  ' каф.' + fdep;
end;
```

Себебі екі сынып та сол базалықтан туындады, студенттер мен оқытушылардың тізімін былай жариялауға болады (мұнда объект – бұл көрсеткіш екенін еске түсіру қажет):

list:  array[l.. SZL] of TPerson; // SZL — массив-тізімнің мөлшері

Мұндай үлгіде тізімді жариялауға болады, өйткені Delphi көрсеткішке аталық сыныпқа көрсеткіштің мәнін еншілес сыныпқа беруге мүмкіндік береді. Сондықтан list массивінің элементтері TStud сыныбы объектісі ретінде де, Tprof сыныбы объектісі ретінде де болуы мүмкін.

## 2.4. КОМПОНЕНТТІК ТӘСІЛ

Компоненттік тәсіл мен CASE-технологиясы (1990-шы жылдардың ортасынан біздің уақытқа дейін) жекелеген компоненттерден – стандартталған еселенген интерфейстер арқылы өзара әрекет ететін бағдарламалық қамтамасыз етудің физикалық түрде жеке қолданыста болатын бөліктерінен бағдарламалық қамтамасыз етудің құрылымын қамтиды.

Қалыпты объектілерге қарағанда, компонент объектілерді серпінді шақырылған кітапханалар немесе орындалатын файлдарға жинауға, екілік түрінде (бастапқы мәтінсіз) таратуға және тиісті технологияны қолдайтын бағдарламалаудың кез келген тілінде пайдалануға болады. Қазіргі уақытта компоненттер нарығы – Интернет және жаппай жарнама және жарияланым қолдайтын нақтылық.

Компоненттік тәсілдің негіздерін құрамдық құжаттарды құру үшін Windows-тың бұрынғы нұсқаларында қолданылған OLE (Object Linking and Embedding — объектілерді байланыстыру және енгізу) технологиясынан бастап *Microsoft* компаниясы әзірледі. Ол COM-технологиясының (Component Object Model — объектілердің компоненттік моделі) пайда болуы, сосын негізінде компонентті технологиялар әзірленген, бағдарламалық қамтамасыз етуді әзірлеудің түрлі міндеттері шешілетін оның DCOM – үлестірілген нұсқасының пайда болуымен дамыды.

Олардың ішінде OLE-automation – осы қосымшалардың ішкі қызметтеріне қол жетімділікті қамтамасыз ететін бағдарламаланған қосымшаларды құру технологиясын атап өткен жөн. OLE- automation негізінде шоғырландырылған да, үлестірілген де бағдарламалық қамтамасыз етуді құруға арналған ActiveX технологиясы құрылды.

Үлестірілген қосымшалардың қауіпсіздігі мен тұрақты жұмысы COM-ға салынған тағы екі технологиямен қамтамасыз етіледі. Бұл MDS (Multitier Distributed Application Sever) — көп буынды үлестірілген қосымшалар сервері және MTS (Microsoft Transaction Server) — транзакцияларды басқару сервері.

Компонентті тәсіл CORBA (Common Object Request Bracer Architecture — объектілер сауалдарын өңдеу делдалымен жалпы сәулет) технологиясы негізінде де жатыр. COM-ға ұқсас тәсілді іске асыратын бұл технологияны OMC (Object Management Group — бағдарламалаудың объектілік технологиясын енгізу тобы) компаниясының тобы әзірледі.

CORBA бағдарламалық өзегі барлық негізгі аппараттық және бағдарламалық платформалар үшін іске асырылды және гетерогендік есептеу ортасында бағдарламалық қамтамасыз етуді құруды қамтамасыз етеді.

Бағдарламалау технологиясының заманауи кезеңінің маңызды ерекшелігі – компьютерлік технологияларды кеңінен пайдалану және олардың тіршілік циклінің барлық кезеңдерінде бағдарламалық жүйелерді құру және сүйемелдеу.

Бұл технологиялар CASE-технология (ComputerAided Software/System engineering – компьютерлік қолдауды пайдалана отырып, бағдарламалық қамтамасыз ету/бағдарламалық жүйелерді әзірлеу) атауын алды. Бүгінде құрылымдықты да, объектілікті де, оның ішінде бағдарламалауға компонентті тәсілдерді қолдайтын CASE-технологиясы бар.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. Бағдарламалық модульдің ерекшелігі қалай сипатталады?
2. Модульдік бағдарламалаудың ерекшелігі неден тұрады?
3. Модульдің сипаттамаларын атаңыз.
4. Өршімелі және бәсеңдейтін әзірлеме әдістерінің айырмашылықтары неден тұрады?
5. Модульдік бағдарламалаумен салыстырғанда объектілік-бағдарланған бағдарламалаудың негізгі құндылықтары қандай?
6. Объектілік-бағдарланған бағдарламалауда объект дегеніміз не?
7. Объектілік-бағдарланған бағдарламалауда әдіс дегеніміз не?
8. Пайдаланушыға деректерге тікелей қол жетімділікке неліктен тыйым салынады?
9. Объектілік-бағдарланған бағдарламалауда қасиет дегеніміз не?
10. Объектінің жай-күйі немен сипатталады?
11. Объектінің интерфейсі дегеніміз не?
12. Сынып және сынып данасы дегеніміз не?
13. Инкапсуляция деген не. Осы қағидатты пайдалануға мысал келтіріңіз.
14. Мұралау дегеніміз не? Осы қағидатты пайдалануға мысал келтіріңіз.
15. Полиморфизм дегеніміз не Осы қағидатты пайдалануға мысал келтіріңіз.
16. Бағдарламалық қамтамасыз етуді әзірлеуде компонентті тәсілдің мәні неде?
17. Базалық және туынды сыныптарды қалай жариялау қажет?
18. Сынып туындатқышы деген не?
19. Сынып конструкторы қандай міндеттерді шешеді?
20. Әдісті виртуалды деп хабарлау не береді?

# 3 тарау

# БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУДІ РЕТКЕ КЕЛТІРУ ЖӘНЕ ТЕСТІЛЕУ

## 3.1. БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУДІ ВЕРИФИКАЦИЯЛАУ ПРОЦЕСІНІҢ БӨЛІГІ РЕТІНДЕ ТЕСТІЛЕУ

Верификация берілген функцияларды орындау дұрыстығын және бағдарламалық қамтамасыз етудің тапсырыс берушінің талаптарына, сондай-ақ берілген ерекшелікке сәйкестігін тексеруді қамтамасыз етеді. Верификация стандарттарда тіршілік циклінің дербес процесі ретінде ұсынылды және талаптарды талдау кезеңінен басталып, қорытынды кезеңде, атап айтқанда тестілеуде бағдарламалық кодтың қызмет ету дұрыстығын тексерумен аяқтап, пайдаланылады.

*Бағдарламалық қамтамасыз етуді тестілеу* – екі түрлі мақсатты қамтитын бағдарламалық қамтамасыз етуді (бағдарламалық код және құжаттама) зерттеу және тексеру процесі:

1) тапсырыс берушілер мен әзірлеушілерге бағдарламалық өнімнің талаптарға сәйкес келетінін көрсету;

2) бағдарламалық қамтамасыз етудің мінез-құлқы дұрыс емес, орынсыз немесе ерекшелікке сәйкес келмейтін жағдайды анықтау.

*Тестілеу* – бағдарламалық қамтамасыз етудің арнайы, жасанды қалыптасқан жағдайларда оның жұмысын қадағалау арқылы жүзеге асырылатын талаптарға сәйкестігін тексеру

Тестілеудің негізгі міндеті – барынша көрнекі және жалпы бағдарламалық қамтамасыз етудің дұрыстығына сенімділіктің жеткілікті дәрежесімен тестілеуді аяқтауға мүмкіндік беретін және бағдарламалық қамтамасыз етудің нақты жағдайының талаптарға сәйкес дұрыс жұмыс істеп тұрғанына көз жеткізетіндей жағдайлар жиынын қалыптастыру. Тестілеу алдын ала жоспарланып, арнайы тәуелсіз мамандар жүйелі түрде жүргізіп отыруы тиіс.

*Тестілеуші маман* — тестілеумен айналысатын маман. Тестілеуші маман бағдарламалық қамтамасыз етуде ықтимал қателер мен ақауларды іздеуді жүзеге асырады, бағдарламалық өнімді пайдалану процесінде туындауы мүмкін түрлі жағдайларды үлгілейді. Тестілеу бағдарламалық қамтамасыз ету сапасын бақылаудың неғұрлым кеңінен қолданылып жүрген әдісі болып табылады.

Сапаның көптеген атрибуттарын бағалау үшін тестілеуден басқа тиімді тәсілдер жоқ. Тестілеу жиыны үшін тестілерді іріктеу және құру әдістерінің жүйесі *тестілеу стратегиясы* деп аталады.

Тестілеушілер жұмысы талаптар ерекшеліктерін бекіткенге дейін басталады, өйткені ол бағдарламалық қамтамасыз етуге талаптардың толықтығын және тестілеу мүмкіндігін тексереді. Тестілеу әдістерін айқындайды. Тестілеу ерекшеліктерінің жоспарлау және құру кезеңі басталған сәттен бір уақытта тестілеу стратегиясын әзірлейді.

Талаптардың ерекшеліктерін бекіткеннен кейін тестілеуші тестілеудің егжей-тегжейлі жоспарын әзірлейді, бағдарламалық қамтамасыз етудің дұрыстығын тексеру үшін тесттер жиынын құрады. Тестілеу оның нәтижелері туралы есептер жасаумен аяқталады.

Тестілеу келтіру, бақылау және сынақ болып табылады.

*Ретке келтіру* — бағдарламалық қамтамасыз етуді әзірлеу кезеңінде бағдарламалық кодты тестілеу.

*Бақылау* —тестілеу және үлгілеу ортасында бағдарламаларды орындау кезінде қателерді іздеу.

*Сынақ* — нақты ортада бағдарламаларды орындаған кезде қатені іздеуге тырысу.

Тестілер белгілі бір талаптарды қанағаттандыруы тиіс.

1. Ең алдымен, тестпен болуы мүмкін қателерді анықтау мүмкіндігі жоғары болуы тиіс. Тестілеу сценарийін әзірлей отырып, бағдарлама ақауларының барлық ықтимал нұсқаларын талдау қажет.
2. Тестер жиыны артық болмауы тиіс. Сол бір қатені анықтау үшін бірнеше тест орындаудың қажеттілігі жоқ. Бірақ бұл тест өз санатында барынша таңдамалысы болған жағдайда солардың бірін орындаса жеткілікті. Ұқсас тесттер тобында барын тиімді және тиімділігі төмендеу тесттер де бар. Сондықтан қатені неғұрлым дәл анықтайтын тесті таңдаған жөн.
3. Тест тым қарапайым не тыс күрделі болмауы тиіс. Ауқымды, күрделі тестті түсінік қиын, орындау қиын және ұзақ құруды қажет етеді. Сондықтан ортаны ұстаған абзалырақ.

Бағдарламалық қамтамасыз етуде барлық қателерді табуға қабілетті тестердің болуы екіталай. Көрсетілген талаптарға жауап беретін жақсы тест көп қателерді анықтау мүмкіндігін береді.

Тестілеу процесі верификацияның компоненті болып табылады деп аталып кеткен. Верификацияның мақсаты тексеріліп отырған объект талаптарға сәйкестігіне, көзделмеген функцияларсыз іске асырылғанына, жобалау ерекшеліктері мен стандарттарды қанағаттандыратынына кепілдікке қол жеткізу болып табылады. Верификация процесі инспекцияны, кодты тестілеуді, тестілеу нәтижелерін талдауды, проблемалар туралы есептерді қалыптастыру және талдауды қамтиды.

Егер бұл процестерге олар жауап беретін сұрақ тұрғысынан қараса, онда тестілеу «Бұл қалай жасалды?» немесе «Әзірленген бағдарламалар мінез-құлқы талаптарға сәйкес келе ме?», верификация – «Не жасалды?» немесе «Әзірленген жүйе талаптарға сәйкес келе ме?» деген сұрақтарға жауап береді.

Верификация процесі жүйеде барлық іркіліс, бас тарту немесе авария тудыруы мүмкін барлық ақаулардың болмауына кепілдік бермейді – әңгіме осы ақаулардың болмауының белгілі бір деңгейі туралы ғана.

## 3.2. РЕТКЕ КЕЛТІРУ ӘДІСТЕРІ

Бағдарламалық қамтамасыз етуді тестілеу кезінде табылған қателерді оқшаулау және түзету процесі *ретке келтіру* деп аталады.

Орындалуы есептеу процесінің бұзылуына алып келген бағдарламалар операторы/операторларын айқындау *оқшаулау* болып табылады.

Қатені түзету үшін оның себебін айқындау қажет, яғни қатесі бар операторлы немесе фрагментті анықтау. Қателер себебі анық және өте терең жасырын болуы да мүмкін.

Өңдеу кезеңіне сәйкес қателерді жіктеу:

- *компиляция қателері* (синтаксистік қателер) —бағдарламаға синтаксистік және ішінара семантикалық талдау орындаған кезде компилятормен (транслятормен, интерпретатормен) тіркелген қателер;
- *құрастыру қателері* —бағдарлама модульдерін біріктірген кезде құрастырушы (байланыс редакторы) тапқан қателер;

■ *орындау қателері* — бағдарламаны орындаған кезде операциялық жүйе, аппараттық құрал немесе пайдаланушы тапқан қателер.

*Компиляция қатесін* ең қарапайым топқа жатқызады, өйткені тіл синтаксисі, әдетте, қатаң нысандандырылған және қателер оның орын көрсете отырып жазылған пікірлермен сүйемелденеді. Тіл синтаксисінің қағидасы жақсы нысандандырылған сайын компилятор барынша көп қате табады, тиісінше келесі кезеңдерде қате аз табылады. Осыған байланысты қорғалған синтаксиспен және қорғалмаған синтаксиспен бағдарламалау тілдері туралы айтылады.

*Құрастыру қателері* сыртқы сілтемеге рұқсат беру кезінде табылған проблемаларға байланысты. Мысалы, басқа модульдің қосымша бағдарламасына жүгіну көзделген, ал модульдерді біріктірген кезде бұл қосымша бағдарлама табылмады немесе параметрлер тізімі түйіспейді. Көп жағдайда мұндай қателерді жылдам оқшаулап, жоюға болады.

*Орындау қателері* ең болжауға келмейтіні болып табылады. Олардың табиғаты әртүрлі және тиісінше әртүрлі жолмен пайда болады. Қателердің бөлігін операциялық жүйе тауып, құжаттайды. Орындау қателері мынадай үлгіде пайда болуы мүмкін:

■ машиналық командаларды орындауды бақылау схемалары тіркеген қателер туралы хабарламаның пайда болуы, мысалы разрядтардың толып кетуі, нөлге бөлу, адрестеудің бұзылуы;

■ Операциялық жүйе тапқан қате туралы хабарламаның пайда болуы, мысалы жады қорғанысының бұзылуы, жазбадан қорғалған құрылғыға жазуға тырысу, берілген атаумен файлдың болмауы;

■ Компьютердің «қатып қалуы» - кейде бағдарламаны операциялық жүйені қайта жүктемей аяқтауға болады, ал кейде жұмысты жалғастыру үшін қайта жүктеу қажет болады;

■ алынған нәтиженің күтілген нәтижемен сәйкес келмеуі. Қателердің себептері алуан түрлі. Сондықтан оқшаулау да өте күрделі болуы мүмкін. Барлық қателердің ықтимал себептерін мына топтарға бөлуге болады:

■ бастапқы деректерді дұрыс айқындамау;

■ логикалық қателер;

■ есептеу нәтижелерінің олқылықтарының жинақталуы.

Егер енгізу-шығару операцияларын орындаған кезде кез келген қате туындаса, *бастапқы деректер дұрыс айқындалмаған* болады, атап айтқанда, беру қатесі, қайта құру қатесі, қайта жау қатесі, деректер қатесі.

Арнайы техникалық құралдарды пайдалану және қатеден қорғаумен бағдарламалау сөзсіз ұмытуға болмайтын осы қателердің бөлігін ғана тауып, алдын алуға мүмкіндік береді.

*Логикалық қателердің* табиғаты да әртүрлі болады. Сонымен, олар жобалау кезінде жол берілген қателерден тұруы мүмкін, мысалы әдістерді таңдау, алгроитмдерді әзірлеу немесе сыныптар құрылымдарын айқындау кезінде және модульді кодтау кезінде тікелей енгізілуі мүмкін.

*Кодтау қателеріне мыналар* жатады:

- Айнымалыны дұрыс пайдаланбау қателері, деректер типін сәтсіз таңдау, айнымалыны оларды жүктегенге дейін пайдалану, массивтерді айқындау шекарасынан шығатын индекстерді пайдалану, түрлендірілмеген айнымалыны, ашық массивтерді, бірлестіктерді, серпінді жадыны, адрестік арифметиканы анық немесе анық емес қайта айқындағанда пайдаланған кезде деректер типтері сәйкестігінің бұзылуы;
- есептеу қателері, мысалы, арифметикалық емес айнымалыларды дұрыс пайдаланбау, бүтін сандық айнымалылармен дұрыс жұмыс жасамау, есептеу процесінде деректер типтерін дұрыс түрлендірмеу;
- модульдердің өзара іс-әрекетінің, яғни, модульаралық интерфейстің қателері, мысалы, параметрлерді берген кезде типтер мен бәйектіліктің бұзылуы, формальдық және нақты параметрлердің өзгеру бірлігінің бірлігін сақтамау, жергілікті және жаһандық айнымалылар іс-қимылы саласының бұзылуы;
- Кодтаудың басқа да қателері, мысалы, кодтау кезінде бағдарлама логикасын дұрыс іске асырмау, бағдарламалаудың нақты тілінің ерекшеліктері немесе шектеулерін ескермеу.

Ретке келтіру процесі бағдарламашы маманның пайдаланудағы техникалық құралдарды, операциялық жүйелерді, бағдарламалау ортасы мен тілін, іске асырылып жатқан процестерді, түрлі қателердің табиғаты мен ерекшелігін, реттеу әдістемесін және тиісті бағдарламалық құралдарды басқару ерекшеліктерін терең білуді талап ететін күрделі процес болып табылады.

Ретке келтірудің күрделілігі мына факторлардың ықпалы салдарынан үдеуі мүмкін:

- қателердің жанама пайда болуы;
- қателердің өзара ықпалының мүмкіндігі;

- түрлі қателердің сырттай біркелкі көрінуінің алу мүмкіндігі;
- іске қосудан іске қосудағы кей қателердің (стохастикалық қате) көрінуінің қайталанбауы;
- бағдарламаға кейбір өзгерістерді енгізу кезінде зерттелетін жағдайда қателердің сырттай көрінуін жою мүмкіндігі, мысалы, бағдарламаға диагностикалық фрагменттердің енгізген кезде сырттай көрінген қателер жойылып немесе өзгеруі мүмкін;
- бағдарламаның жекелеген бөліктерін түрлі бағдарламашылардың жазуы.

Бағдарламаны ретке келтіру кез келген жағдайда қате туралы барлық бар ақпаратты ойластыруды және логикалық ой елегінен өткізуді болжайды. Қателердің басым бөлігін бағдарламалар мәтінін және қосымша ақпараттар алмай тестілеу нәтижелерін тыңғылықты талдау арқылы жанама белгілер бойынша табуға болады:

- қолмен тестілеу;
- индукция;
- дедукция;
- кері бақылау.

*Қолмен тестілеу әдісі* – бағдарламаны ретке келтірудің ең қарапайым да табиғи тәсілі. Қатені тапқан кезде тестіленіп жатқан бағдарламаны қолмен орындау қажет, мұнда жұмыс кезінде қате табылған тестілеу жиынын пайдаланады. Әдіс тиімді, алайда күрделі есептеулермен бағдарламаларға, ауқымды бағдарламаларға, сондай-ақ бағдарламашының операцияны орындау туралы дұрыс емес түсінігіне байланысты қате болған жағдайда қолданылмайды. Бұл әдіс ретке келтірудің басқа да әдістерінің компоненті ретінде жиі пайдаланылады.

*Индукция әдісі* есептеудің дұрыс емес нәтижелері ретінде немесе қате туралы хабарлама ретінде пайда болатын қателердің белгілерін мұқият талдауда негізделген. Егер компьютер жай ғана «қатып қалса», онда қатенің көріну фрагменті пайдаланушының соңғы алынған нәтижелері мен іс-қимылдарын ескеріп, есептеледі.

Осындай тәсілде алынға ақпаратты бағдарламаның тиісті фрагментін қарап отырып, зерделеуге болады. Нәтижесінде артынша тексерілетін қате туралы гипотеза жылжиды. Егер гипотеза дұрыс болса, онда қате туралы ақпарат талдап тексеріледі, олай болмағанда – басқа гипотезаны ұсынады.

Қате туралы деректерді жинай келе оның пайда болуы туралы белгілі болғанның барлығын тіркеу қажет. Қатемен фрагмент қалай қалыпты орындалатын жағдайды да, қате пайда болған жағдайды да назарға алады. Егер деректердің зерделеу нәтижесінде гипотеза пайда болмаса, онда қате туралы қосымша ақпарат қажет.

*Дедукция әдісі* мынаны қамтиды. Алдымен қатенің пайда болуын тудырған көптеген себептерді қалыптастырады, сосын себептерді талдай отырып, бар деректерге қарама-қайшы келетіндерді алып тастайды. Егер барлық себеп алып тасталса, онда зерттелетін фрагменттің қосымша тестілеуін орындау қажет. Қарсы жағдайда неғұрлым ықтимал гипотезаны дәлелдеуге тырысады. Егер гипотеза алынған қате белгілерін түсіндірсе, онда қате табылғаны, басқаша жағдайда – келесі себебін тексереді.

*Кері бақылау әдісі* шағын бағдарламалар үшін пайдаланылады. Тексеру дұрыс нәтиже шығармау нүктесінен басталады. Осы нүкте үшін бар нәтижені көруге алып келетін негізгі айнымалының мәні туралы гипотеза қалыптасады. Одан әрі, осы гипотезаны ескеріп, алдыңғы нүктеде айнымалының мәні туралы ұсыныс жасайды. Процес қатенің себебін таппайынша жалғасады.

Ендігі ретте кейбір жиі кездесетін бағдарламалық қателерді қарастырсақ:

*Функционалдық кемшіліктері.* Бұл кемшіліктер егер ол жасауы қажетті жасамаса, өзінің функцияларының бірін нашар орындаса немесе толықтай орындамаған бағдарламаға тән. Бағдарламаның функциясы оның ерекшелігінде егжей-тегжейлі жазылуы тиіс және сол бекітілген ерекшелік негізінде тестілеуші өзінің жұмысын қалыптастырады.

*Пайдаланушылық интерфейстің кемшіліктері.* Пайдаланушылық интерфейс жұмысының қолайлығы мен дұрыстығын онымен қызмет ету процесінде ғана бағалауға болады. Бұл жұмысқа пайдаланушының өзінің қатысқаны абзал. Бұған бағдарламалық өнім прототипін әзірлеу көмегімен қол жеткізуге болады, онда оларды талаптар ерекшелігіне одан әрі тіркей отырып, пайдаланушылық интерфейске барлық талаптарды келісу және сынау жүргізіледі.

Талаптар ерекшелігін бекіткеннен кейін одан кез келген ауытқу немесе соңғыны орындамау қате болып табылады. Бұл пайдаланушылық интерфейске де толықтай қатысты.

*Жеткіліксіз өнімділік.* Кейбір бағдарламалық өнімді әзірлеген кезде жұмыс жылдамдығы оның аса маңызды сипаттамасы болады, кейде бұл критерий тапсырыс берушінің бағдарламалық өнімге талаптарында беріледі.

Егер пайдаланушыда бағдарлама баяу жұмыс істейді деген әсер туындаса, әсіресе егер бәсекелесуші бағдарламалар жылдамырақ жұмыс істесе қиын, егер бағдарлама талаптар ерекшеліктерінде берілген сипаттаманы қанағаттандырмаса тіпті қиын. Бұл енді жою қажет болатын қатеге жатады.

*Қатені дұрыс өңдемеу.* Қателерді өңдеу рәсімдері – бағдарламаның өте маңызды бөлігі. Қатені дұрыс анықтаған бағдарлама ол туралы хабарлама беруі тиіс. Мұндай хабарламаның болмауы бағдарлама жұмысындағы қате болып табылады.

*Шектік жағдайларды дұрыс өңдемеу*. Көптеген түрлі шектік жағдайлар бар. «Көбірек», «азырақ», «ертерек», немесе «кешірек», «бірінші» немесе «соңғы», «қысқарақ» немесе «ұзынырақ» ұғымдары қолданылатын бағдарлама жұмыстарының кез келген аспектісі диапазон шекарасында міндетті түрде тексерілуі тиіс.

Диапазон ішінде бағдарлама керемет жұмыс істейтін болады, ал оның шекараларында өз кезегінде бағдарламалық қамтамасыз етудің жұмысында қате жіберуге алып келетін күтпеген жағдайлар орын алуы мүмкін.

*Есептеу қателері.* Есептеу қателеріне есептеу алгоритмін дұрыс таңдамағаннан, дұрыс емес формулалар, өңделетін деректерге қолданылмаған формулалардан болған қателіктер жатады. Есептеу қателерінің арасында ең кеңінен таралғаны дөңгелектеу қателері.

*Ағынды басқару қателері.* Бағдарлама жұмысының логикасы бойынша бірінші-іс-қимылдың соңынан екіншісі орындалуы тиіс. Егер мұның орнына үшінші немесе төртінші іс-қимылдар орындалса, демек, ағындарды басқаруда қатеге жол берілді.

*Жарыс жағдайы.* Жүйеде екі оқиға күтіледі деп болжайық: «А» және «Б». Егер «А» оқиғасы бірінші басталса, онда бағдарламаны орындау жалғасады, ал егер «Б» оқиғасы болса, онда бағдарлама жұмысында іркіліс болады. Әзірлеушілер «А» оқиғасы үнемі бірінші болуы қажет деп болжайды және «Б» жарысты жеңіп, бірінші шығады деп күтпейді де. Классикалық жарыс жағдайы осындай.

Жарыс жағдайын тестілеу барынша күрделі. Олар өзара іс-қимыл жасайтын процестер мен ағындар қатар орындалатын жүйелер, сондай-ақ нақты уақыттың пайдаланушылары көп жүйелері үшін барынша типті. Мұндай жүйелерде қатені табу қиын, оларды анықтау үшін өте көп уақыт талап етіледі.

*Артық жүктеу.* Бағдарлама жұмысында іркіліс жадының жетіспеуінен немесе басқа да қажетті жүйелі ресурстардың болмауынан орын алады. Әр бағдарламаның өз шегі бар, бағдарлама артық жүктемені көтере алмауы мүмкін.

Мысалы, аса көлемді деректер. Бағдарламаның нақты мүмкіндіктері, оның талаптары бағдарлама ерекшелігінің ресурстарына сәйкес келе ме, ол артық жүктемеде өзін қалай ұстайды, мәселе осында.

*Компьютердің аппаратурасымен дұрыс емес жұмыс істемеу.* Бағдарламалар аппараттық құрылғыға дұрыс емес деректер жіберіп, олардың қате туралы хабарламасын ескермей, бос емес немесе мүлдем жоқ құрылғыны пайдалануға тырысуы мүмкін. Тіпті егер қажетті құрылғы жай ғана істен шықса, оған жүгінген кезде «қатып қалмай» бағдарлама мұны түсінуі тиіс.

## 3.3. ТЕСТІЛЕУ ӘДІСТЕРІ

Тестілеудің жүйелі әдістері бағдарлама «қара жәшік» ретінде қарайтын әдістерге және бағдарлама «ақ жәшік» ретінде қаралатын әдістерге бөлінеді.

**«Қара жәшікті» тестілеу.** «Қара жәшікті» тестілеу – бұл сыртқы ілем тұрғысынан объектінің (бағдарламалық жүйенің) функционалдық мінез-құлқын тестілеу әдісі.

«Қара жәшік» әдісімен тестілеген кезде бағдарлама ішкі құрылымы белгісіз объект ретінде қаралады.

Тестілеуші деректер енгізеді және нәтижені талдайды. Бірақ бағдарлама қалай жұмыс істейтінін білмейді. Тесті таңдай отырып маман оның көзқарасы тұрғысынан қызықты стандартты емес нәтижелерге әкелуі мүмкін келген деректерді іздейді. Тестіленетін бағдарламаның қателері шығуының барынша ықтималдығымен кіретін деректер әр сыныбының өкілдері пайдаланылады.

Тестілеудің бұл әдісі үшін тестілеушінің негізгі міндеті дәйекті тексеруде жүйе мінез-құлқының талаптарға сәйкестігінен тұрады. Бұдан басқа, тестілеуші күрделі жағдайларда – дұрыс емес кіретін мәндер берген жағдайда жүйе жұмысын тексеруі қажет.

Мінсіз жағдайларда күрделі жағдайлардың барлық нұсқалары жүйеге қойылатын талаптарда сипатталуы тиіс және тестілеушіге осы талаптардың нақты тексерісін ойластыру ғана қалады. Алайда шындығында тестілеу нәтижесінде әдетте жүйе проблемасының екі типі анықталады:

1) жүйе мінез-құлқының талаптарға сәйкес келмеуі;
2) талаптарда көзделмеген жағдайлардағы жүйенің лайықсыз мінез-құлқы.

Проблеманың екі түрі туралы есеп құжатталып, әзірлеушіге беріледі.

Бұл ретте бірінші типтің проблемалары әдетте бағдарламалық кодтың өзгерісін, сирек болса да талаптардың өзгеруін тудырады.
Мұндай жағдайларда талаптардың өзгеруі олардың қарама-қайшылығынан (бірнеше түрлі талаптар бір және сол жағдайда жүйе мінез-құлқының әртүрлі үлгілерін сипаттайды) немесе дұрыс еместігінен (талап шындыққа сәйкес келмейді) талап етілуі мүмкін.

Екінші типтің проблемасы оның толық еместігінен талаптарды өзгертуді біржақты талап етеді – талаптарда жүйенің лайықсыз мінез-құлқына алып келетін жағдай жоқ. Бұл ретте лайықсыз мінез-құлық деп жүйенің толықтай күйреуі аталады және бұған талапта жазылмаған кез келген мінез-құлық жатады.

«Қара жәшік» әдістері мынаны қамтамасыз етеді:

- баламалы бөлу;
- шекті мәндерді талдау;
- резервтік талдаумен қосылғанда тестіленетін бағдарламаның қызмет етуі туралы жеткілікті толық ақпаратты беретін функционалдық диаграммаларды қолдану.

Баламалы бөлу бағдарлама деректерінің кіретін саласын кей сыныптың өкілі болып табылатын әр тест осы сыныптың кез келген басқа тестке баламалы болатындай баламалылық сыныптарының соңғы санына бөлуден тұрады.

Баламалылық сыныптары кіретін жағдайларды іріктеу және оларды екі немесе одан көп топқа бөлу арқылы анықталады. Бұл ретте баламалылық сыныптарының екі типін айырады: бағдарламалар үшін кіретін деректерді беретіндер дұрыс және қате кіретін маңыздар беруге негізделген дұрыс емес.

Баламалы бөлу әдісімен тесттерді әзірлеу екі кезеңде жүзеге асырылады: баламалылық сыныбын бөлу және тесттер қалыптастыру. Кіретін деректер таңдауға негізделген тесттерді қалыптастыру кезінде бағдарламаны таңбаикалық орындау жүргізіледі. Осылайша, «қара жәшік» қағидаты бойынша, тестілеу әдістері бағдарламада іске асырылған функцияларды тестілеу үшін пайдаланылады. Бұл үшін функцияның нақты мінез-құлқы мен талаптар ерекшеліктерін ескеріп, күтілетін мінез-құлық арасындағы сәйкессіздік тексеріледі.

Осы тестілеуге дайындық уақытында жағдай кестелері, себеп-салдарлық бағандар мен бөліс саласы қалыптастырылады. Бұдан басқа, функцияның мінез-құлқына әскер ететін орта шарты мен параметрлерді ескеретін тестілеу жиыны дайындалады.

**«Ақ жәшікті» тестілеу.** «Ақ жәшік» әдісі бағдарламаның ішкі құрылымын зерттеуге мүмкіндік береді.

«Ақ жәшік» қағидаты бойынша тестілеу жолдық және еліктемелік тестілеуді қолдану арқылы бағдарламалардың барлық жолдарын өтуді тексеруге бағдарланған.

«Ақ жәшікті» тестілеу модульдер және тесттік жағдайларды таңдау арқылы бағдарламаның графтық моделінің деңгейінде қолданылады және мына элементтерді тестілеуді қамтиды:

- бағдарламада логикалық мөлшердің аса көптігінен және осы жолдардың қосалқы жиынтығын өткеру қажеттілігінен қалып қоюы мүмкін қателерді есепке алмай, тым болмағанда бір рет орындалуы тиіс операторлар;
- Есептеу үшін барлық жолдарды өтуге кепілдік беретін тестілеу деректерінің жиынын құратын жол предикаттарының көмегімен басқаруды берудің түрлі маршруттарын анықтау үшін басқару ағындарының берілген графы бойынша жолдар. Алайда барлық жолды тестілеу мүмкін болмауы мүмкін. Сондықтан пайдалану процесінде кезігетін анықталмаған қателер қалып қоюы мүмкін.
- бағдарламаны бір функцияны іске асыру блоктарының жиынтығын қамтитын бағдарламадағы жолдарды табу не көрсетілген жолды орындау үшін пайдаланылатын кіретін көптеген деректерді табу кезінде бір рет немесе көп рет орындалатын жекелеген блок-бөлімдерге бөлетін блоктар.

«Ақ жәшікті» тестілеу – әзірлеу кезеңінде қолданылатын тестілеу технологиясы (кейде оны «шыны жәшік» тестілеуі деп атайды).

Тестілеуші маман ол толық қол жетімділігі бар бастапқы кодты білуін негізге алып тесттерді әзірлейді. Нәтижесінде ол мынадай артықшылықтар алады:

1. *Тестілеудің бағыттылығы.* Бағдарламашы маман бағдарламаны бөлімдер бойынша тестілейді, тестіленетін модульді шақыратын арнайы тестілеу сценарийін әзірлейді де оған қызықтыратын деректерді береді.
2. *Кодты толық қамту.* Бағдарламашы маман әр тестте кодтың қай фрагменттері жұмыс істейтінін үнемі айқындай алады.
3. *Командалар ағынын басқару мүмкіндігі.* Бағдарламашы маман бағдарламаға оның орындалу барысы туралы ақпаратты көрсететін реттеу командасын енгізе немесе осы үшін арнайы бағдарламалық құрал – реттеуішті пайдалана алады.
4. *Деректердің тұтастығы, бақылау мүмкіндігі.*

деректердің жай-күйін бақылай отырып (реттеуіштің көмегімен), бағдарламашы маман басқа модульдермен деректердің өзгеруі, олардың дұрыс түсінік бермеу немесе сәтсіз ұйымдастыру және т.б. сияқты қателерді анықтай алады.

5. *Ішкі шектік нүктелерді көру.* Мысалы, белгілі бір іс-қимылды орындау үшін бірнеше түрлі алгоритмдер пайдаланылуы мүмкін, бағдарламалық кодқа қолжетімділік болмаса, бағдарламаны әзірлеген кезде олардың қайсысы таңдалғанын айқындау мүмкін емес.

«Ақ жәшікті» тестілеу - бағдарламалау процесінің бөлігі. Бағдарламашы мамандар бұл жұмысты тұрақты орындайды, олар оны жазғаннан кейін әр модульді тестілеуден өткізеді, сосын оны жүйеге интеграциялағаннан кейін тағы да тестілеуден өткізеді. «Ақ жәшікті» тестілеудің неғұрлым қуатты теориялық негізі болғанына қарамастан, тестілеуші мамандардың басым бөлігі «қара жәшікті» тестілегенді жөн санайды. «Ақ жәшікті» тестілеу математикалық үлгілеуге жақсы келеді. Бірақ бұл оның тиімдірек екенін білдірмейді.

Технологиялардың әрқайсысы басқасын пайдаланған жағдайда жіберілетін қателерді анықтауға мүмкіндік береді. Бұл тұрғыдан қарағанда, оларды бірдей тиімді деп атауға болады.

## 3.4. ДЕҢГЕЙЛЕР БОЙЫНША ТЕСТІЛЕУДІ ЖІКТЕУ

Бағдарламалық өнімді толықтай тексеруге және қатені (модульдік, интеграциялық, жүйелі, шығу, қабылдау) көбірек анықтауға мүмкіндік беретін тестілеудің бірнеше деңгейі бар. Әр деңгейдің өз мақсаттары мен компоненттері болады.

*Модульдік тестілеу деп* – жеке алынған модуль, функция немесе сыныптар деңгейінде бағдарламаларды тестілеуді атайды. Модульдік тестілеудің мақсаты алгоритмдерді іске асыруда қателерді анықтаудан, сондай-ақ әзірлеу мен тестілеудің келесі деңгейіне өтуге жүйенің әзірлік дәрежесін айқындаудан тұрады. Модульдік тестілеу «ақ жәшік» қағидаты бойынша жүргізіледі, яғни бағдарламаның ішкі құрылымын білуге негізделеді және жиі кодты жабуды талдаудың сол немесе өзге тәсілдерін қамтиды.

Модульдік тестілеуді бағдарламалық қамтамасыз етуді әзірлеуші тікелей жүргізеді және әр модульдегі деректер ағыны мен барлық қосымша құрылымдарды тексеруге мүмкіндік береді. Тестілеудің бұл түрі әзірлеу кезегінің бөлігі болып табылады.

модульдік тестілеу әдетте әр модульдің айналасында тестіленетін модульдің барлық интерфейстері үшін бітеуішті қамтитын белгілі бір ортаны құруды көздейді. Олардың кейбірі кіретін мәндерді беру үшін, басқасы – нәтижелерді талдау және т.б. үшін пайдаланылады.

Модульдік тестілеу деңгейінде алгоритмдік қателерге және алгоритмді кодтау қателеріне байланысты ақауды, циклдардың жағдайымен, есептеуішімен, сондай-ақ жергілікті айнымалыны және ресурстарды пайдалана отырып жұмыс типін табу жеңілірек.

Деректерді дұрыс түсіндірмеуге, интерфейстерді дұрыс іске асырмауға, үйлесімділікке, өнімділікке байланысты қателер әдетте модульдік тестілеу деңгейінде жіберіледі және тестілеудің барынша кеш сатыларында анықталады.

Модульдік тестілеу кезінде тестілеумен әр модульдің 75 пайыздан астамы қамтылатындай әзірлеуші айқындаған тесттер жиыны орындалады.

Модульдік тестілеу мыналарды қамтиды:

- бағдарламалық кодта синтаксистік қателерді (синтаксистік тексеру) анықтау үшін кейбір аспапты құралдарды пайдалана отырып бағдарламалық кодты тексеру;
- кодты кодтау стандартына сәйкестікке тексеру (мысалы, бағдарламалық кодты ресімдеудің әзірлеуші ұйымның талаптарына сәйкестігі);
- Бағдарламалық кодты техникалық шолу.

Модульдік тестілеуді орындау кезінде технологияны не құрылымдық не функционалдық тестілеуді немесе соны және басқаны пайдалануға болады.

Құрылымдық тестілеу «Ақ жәшікті» тестілеу түрлерінің бірі болып табылады. Оның басты идеясы тестіленетін бағдарламалық жолды дұрыс таңдау болып табылады. Оған қарама-қарсылық ретінде функционалдық тестілеу «қара жәшікті» тестілеу санатына жатады.

Бағдарламаның әр функциясы оның кіретін деректерін енгізу және шығатын деректерді талдау арқылы тестіленеді. Бұл ретте бағдарламаның ішкі құрылымы өте сирек ескеріледі.

Модульдік тестілеуді сәтті аяқтағаннан кейін модульдің барлық өзгерістері мен тесттер жиыны жобаның деректер базасында сақталады.

*Интеграциялық тестілеу* жекелеген модульдердің бірлескен жұмысын тексеру үшін жүргізіледі және барлық жүйені біртұтас ретінде тестілеуге жол ашады.

Интеграциялық тестілеу —бұл екі және одан көп модульдерден тұратын жүйенің бөлімдерін тестілеу. Интеграциялық тестілеудің негізгі міндеті —модульдер арасындағы интерфейстік өзара іс-қимылды іске асыруда және интерпретациялауда қате іздеу.

Интеграциялық тестілеу элементтері мыналар болып табылады:

- функционалдылықты тексеру, яғни модульмен байланысты орындалған жекелеген функциялардың талаптардың ерекшеліктерінде берілген функцияларға сәйкестігін тексеру;
- аралық нәтижелердің болуы мен дұрыстығын тексеру;
- модульдер арасында ақпарат берудің дұрыстығын тексеру (интеграцияны тексеру).

Технологиялық көзқарас тұрғысынан интеграциялық тестілеу модульдікті сандық дамыту болып табылады, өйткені модульдік тестілеу сияқты модульдердің интерфейстері және қосымша жүйелермен операция жүргізеді және модульдер жоқ жерде бітеуішті қоса алғанда тестілеу ортасын құруды талап етеді.

Модульдік және интеграциялық тестілеу арасындағы негізгі айырма табылған ақаулар түрінен тұрады. Бұл кіретін деректер мен талдау әдістерін таңдау стратегиясын айқындайды. Интеграциялық тестілеу модульдер мен қосымша жүйелерді біртіндеп қосумен интеграциялық жүргізіледі. Оны тәуелсіз тестілеуші жүзеге асырады.

Интеграциялық естілеу барысында анықталған қателер жобаның деректер базасына енгізіледі. Интеграциялық тестілеудің нәтижелері тестілеу циклін аяқтаған кезде тестілеу барысы туралы есепке қосылады.

*Жүйелі тестілеуді* интеграциялық тестілеуді сәтті аяқтаған жағдайда тәуелсіз тестілеуші жүргізеді. Жүйелі тестілеу интеграциялық және модульдік деңгейден сапалы ерекшеленеді. Тестіленетін жүйені тұтастай қарайды және модульдер интерфейсі деңгейінде операция жүргізілетін интеграциялық тестілеудің соңғы фазасына қарағанда, пайдаланушылық интерфейстер дәрежесінде операция жүргізеді. Тиісінше тестілеудің бұл деңгейлерінің мақсаттары да ерекшеленеді. Тестілеу жолдарының бағдарламаның ішінен өтуін талдау немесе жүйе деңгейінде нақты функциялардың дұрыстығын бақылау күрделі.

Жүйелі тестілеудің негізгі міндеті – тұтастай жүйенің жұмысына байланысты проблемаларды анықтау (жүйе ресурстарын дұрыс пайдаланбау, айналамен үйлесімсіздігі, көзделмеген пайдалану сценарийі, жоқ немесе жөнсіз функционалдық, қолданудағы қолайсыздық және т.б.).

Жүйелі тестілеу «қара жәшік» әдісінің көмегімен тұтастай жобада жүргізіледі. Яғни, бағдарлама құрылымының еш мәні жоқ. Тексеру үшін пайдаланушыға көрінетін кіріс пен шығыс қана қолжетімді. Тестілеуге кодтар мен пайдаланушының құжаттамасы да жатады.

Жүйелі тестілеудің келесі санаттары мыналарды анықтайды:

- функционалдық міндеттерді шешу толықтығы;
- ресурстарды пайдалану дұрыстығы (жадының кемуі, ресурстардың қайтымы);
- өнімділікті бағалау;
- деректерді бұрмалаудан және лайықсыз іс-қимылдардан қорғау тиімділігі;
- түрлі платформаларда инсталяция мен конфигурацияны тексеру;
- құжаттаманың дұрыстығы

Тестілеудің осы деңгейде пайдаланылып жатқан деректер көлемі осындай, неғұрлым тиімді тәсіл тестілеуді толықтай немесе ішінара автоматтандыру болып табылады. Бұл модульдер немесе олардың комбинацияларын тестілеу деңгейінде қолданылатын тестілеу жүйесіне қарағанда, барынша күрделі тестілеу жүйесін құру қажеттігіне алып келеді.

Жүйелі тестілеу кезінде анықталған қателер жобаның деректер базасына енгізіледі. Жүйелі тестілеу нәтижелері тестілеу тарысы туралы есепке енгізіледі.

*Шығыс тестілеуі* тапсырыс берушіге/пайдаланушыға жеткізу үшін бағдарламалық қамтамасыз етудің әзірлігін тексеру мақсатында жүзеге асырылады. Бұл инсталляция бойынша нұсқаулықтардың дұрыстығын тексеруді, сондай-ақ құжаттаманың жинақтылығын тексеруді қамтитын тәуелсіз тестілеуші жүргізген тестілеудің аяқталу кезеңі.

Шығыс тестілеуі кезінде анықталған қателер жобаның деректер базасына енгізіледі. Бағдарламалық қамтамасыз етуді шығыс тестілеуді сәтті аяқтаған кезде тапсырыс берушіге тестілеу нәтижесі туралы есеппен бірге бағдарламалық өнім жеткізіледі.

*Қабылдау тестілеуін* бағдарламалық жүйені орнатуға, сүйемелдеуге және соңғы пайдаланушыны оқытуға жауап беретін ұйым жүргізеді. Бұл артынша өнім пайдалануға енгізілетін тестілеудің соңғы деңгейі.

Бірінші төрт деңгейді тестілеу әзірлеуші ұйымның ішінде жүргізіледі, ал қабылдау тестілеуі тапсырыс берушінің өкілімен бірге орындалады. Бірінші деңгейдегі тестілеуді әзірлеу кезеңінде бағдарламалық қамтамасыз етудің әзірлеушісі жүргізеді. Ал қалған деңгейдегі тестілеуді тәуелсіз тестілеушілер жүзеге асырады.

Бір-бірінен орындау уақытымен ерекшеленетін тестілеудің тағы екі түрі бар — альфа- және бета- тестілеу.

*Альфа-тестілеу* — әлеуетті пайдаланушылардың немесе тапсырыс берушілердің бағдарламалық қамтамасыз етуімен нақты жұмысы не әзірлеушілердің нақты жұмысының еліктемесі. Альфа-тестілеу көбінесе өнімді әзірлеудің ерте сатысында, алайда барлық немесе шамамен барлық функционалдық іске асырылғанда жүргізіледі.

Кей жағдайларда альфа-тестілеу ішкі қабылдау тестілеуі ретінде аяқталған өнім үшін қолданылады. Альфа-тестілеу реттеушті немесе қатені тез анықтауға көмектесетін қандай да бір аспапты пайдалана отырып орындалуы мүмкін. Табылған қателер қосымша зерттеу үшін тестілеуші маманға берілуі мүмкін.

*Бета-тестілеу* — жоспарланған функциялардың толық жиынымен бағдарламалық қамтамасыз етудің дайын нұсқасын қарқынды пайдалану. Оның мақсаты өнімнің нарыққа, жаппай тұтынушыларға соңғы шығуының алдынды оларды кейіннен жою үшін бағдарламалық қамтамасыз ету жұмысындағы қателердің барынша көбін анықтау болып табылады.

Әзірлеушілердің немесе тестілеуші маманның күшімен жүргізілген альфа-тестілеуге қарағанда, бета-тестілеу өнімнің жаңа алдын ала нұсқасы (бета-нұсқа) қолжетімді басқа да пайдаланушыларды тартуды көздейді.

Бұдан басқа, бета-тестілеу өнімді нарыққа жылжыту стратегиясының бөлігі (мысалы, бета-нұсқаларды тегін тарату өнімнің соңғы қымбат нұсқасына пайдаланушылардың назарын кеңінен тартуға мүмкіндік береді) ретінде сондай-ақ жаңа пайдаланушылардың олар туралы алдын ала пікірлерін алу үшін жарнамалық мақсатта пайдалана алады.

Бета-нұсқа өнімнің финалдық нұсқасы болып табылмайды, сондықтан әзірлеуші компьютердің жұмысын бұзатын және (немесе) деректердің жоғалуына алып келетін қателердің толықтай болмауына кепілдік бермейді.

## 3.5. БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУДІҢ ӨНІМДІЛІГІН ТЕСТІЛЕУ

Бағдарламалық қамтамасыз ету өнімділігін тестілеу бағдарлама жүйесі қаншалықты жылдам жұмыс істейтінін немесе оның бөлігі белгілі бір жүктемеде екенін айқындау үшін жүргізіледі.

Бағдарламалық қамтамасыз етудің өнімділігін тестілеу ресурстардың масштабталатындығы, сенімділігі, оларды тұтыну сияқты жүйелер сапасының басқа да атрибуттарын тексеру және растау үшін жүргізіледі.

Бағдарламалық қамтамсыз етудің өнімділігін тестілеудің екі тәсілі бар:

1) бағдарламалық қамтамасыз ету пайдаланудың түрлі сценарийіне сәйкес келетін жағдайларда тестілеуге бейім;

2) жүйе нақты соңғы пайданалушылармен сыналғанда, бета-тестілеу аясында.

өнімділік тестілеуінде мынадай бағыттардан тұрады:

- жүктемелік тестілеу;
- стресс-тестілеу;
- тұрақтылық тестілеуі;
- конфигурациялық тестілеу.

**Жүктемелік тестілеу.** Жүктемелік тестілеу өнімділікті айқындау және аталған жүйеге қойылатын талаптарға сәйкестігін анықтау мақсатында сыртқы сауалға бағдарламалық қамтамасыз етудің жауап уақыты көрсеткіштерінің жиынын қамтиды. Оны пайдаланудың қалыпты сценарийін арттыратын барынша жоғары жүктеме кезінде жүйенің жауап уақытын зерттеу үшін стресс-тестілеу жүргізіледі. Жүктемелік пен стресс-тестілеу арасында нақты шекара жоқ, алайда тестілеудің бұл түрлері түрлі сұрақтарға жауап береді және түрлі әдіснаманы пайдаланады.

Жүктемелік тестілеу оның құжаттамасында айқындалған бағдарламалық өнімдердің мүмкіндіктері шектеулерін тексеру болып табылады. Бағдарламалық қамтамасыз ету жұмыс істейтін, логикалық қолжетімді қандай да бір параметр мәндерін тексере алатын максималды файлдар саны немесе басқа да деректер құрылымын ашуға болады.

Түрлі аппараттық ресурстар аяқталып жатқанда бағдарлама өзін қалай ұстайтынын тексеру қажет. Жүктемелік тестілеу – шектік жағдайларды тестілеу түрлерінің бірі. Оны жүргізу схемасы қатты ұқсас. Алдымен бағдарламаны ол қызмет етуі тиіс жағдайда, сосын ол жұмыс істей алмайтын жағдайда іске қосады.

Жағдайлардың түрлі комбинацияларын тексерудің мәні бар. Жекежеке түрлі артық жүктемені көтерсе де, бағдарлама олардың барлығын бірге көтере алмауы әбден мүмкін.

Көбінесе жүктемелік тестілеу бірнеше пайдаланушылардың жұмыс эмуляциясын бір уақытта болжайды. Бұл әдетте, клиент-серверлік сәулетті (мысалы, Web-серверлер) пайдаланатын пайдаланушылары көп жүйе үшін қажет.

Алайда бағдарламалық қамтамасыз ету жүйесінің басқа да түрлері осы тәсілімен тестілеуден өтуі мүмкін. Мысалы, мәтіндік немесе графикалық редакторға өте үлкен құжатты жүктеуге болады: бухгалтерлік бағдарламалық жүйеге бірнеше жылғы деректер негізінде есепті түрлендіреді. Неғұрлым барабар жобаланған жүктемелік тест барынша нақты нәтиже береді.

Жүктемелік тестілеудің негізгі мақсаты жүйеге белгілі бір күтілетін жүктемені құрып, барабар бағдарламалық және аппараттық қамтамасыз етуді пайдалана отырып бағдарламалық жүйенің өнімділігінің көрсеткіштерін бақылауды қамтиды. Мінсіз жағдайда жүктемелік тестілеудің сәттілігінің критерийі ретінде негізгі сәулет шешімдерін бағдарламалауды бастағанға дейін жүйеге қойылатын функционалдық талаптарды әзірлеу сатысында қалыптасатын және құжатталатын жүйенің өнімділігіне қойылатын талаптар қаралады.

Алайда мұндай талаптар нақты немесе мүлдем қалыптастырылмаған болып шығады. Мұндай жағдайда бірінші жүктемелік тестілеу сынама болып табылады және күтілетін жүктеме және ресурстардың аппараттық бөлігін тұтыну туралы ақылға қонымды болжамдарға негізделеді.

Жүйенің түрлі тораптарында проблемаларды табу және зерттеу үшін түрлі аспаптар бар.

Жүктемелік тестілеудің нәтижесі талдау үшін одан әрі пайдаланылатын қосымша өнімділігінің көрсеткіштері болып табылады.

1. *Орталық процессордың ресурстарын тұтыну,* %, — берілген белгілі интервалдан таңдап алынған процесті есептеуге процессормен жұмсалған уақыт қанша екендігін көрсететін метрика. Заманауи жүйелерде процессор қатар есептеуді жүргізе алу үшін бірнеше ағындарда қызмет ету процесінің қабілеті маңызды фактор болып табылады. Процессордың ресурстарды тұтынуын талдауы жүйенің жалпы өнімділігіне түрлі факторлардың ықпалын түсіндіре алады.
2. *Жедел жадыны тұтыну,* Мбайт, — қосымша пайдаланатын жады мөлшерін көрсететін метрика. Пайдаланылған жады бірнеше санатқа бөлінеді:

■ процес пайдаланатын виртуалдық адрестік кеңістік көлемі. Бұл көлем тиісті дискі кеңістігін де, жедел жадыны да пайдалануды көздейді.

виртуалдық жады жүйесі бір процестің ағыны басқа процеске тиесілі жадыға қолжетімділік алмайтынына кепілдік береді;

- процеспен қамтылған және басқа процессорлардан бөлінбейтін адрестік кеңістіктің көлемі;
- процеспен таяуда пайдаланылған жады беттерінің жиыны; бос жады жеткілікті болған жағдайда, егер беттер пайдаланылмаса да, ол жиында қалады. Егер бос жады аз қалса, жадының басқа белсенді беттерін жүктеу үшін жедел есте сақтау құрылғысын босатып, пайдаланылған беттер жедел есте сақтайтын құрылғыдан қатаң дискіге ауыстырылады.
- басқа процеспен бірлесіп пайдалануға болатын физикалық жадының процеспен пайдаланылған көлемі.

3. *Желілік ресурстарды тұтыну* — тұтастай жүйенің өнімділік шегіне көрсетеді.

4. *Дискілік қосымша жүйемен жұмыс (енгізу-шығаруды күту уақыты).* оқулар мен жазбалар көлемінің көптігі процессордың дискіден деректерді өңдеуде күтіп тұрып қалуына әкеледі. Нәтижесінде жауап уақыты ұлғаяды.

5. *Сауалды орындау уақыты,* мс, —бағдарламалық қамтамасыз етудің өнімділігінің басты көрсеткіштерінің бірі болып табылады. Бұл уақыт сервер жақта сауал өңдеу үшін серверлік бөліктің талап еткен уақытының көрсетқосымша ретінде де, клиент жағында сауалды жіберу және өңдеуге талап етілетін толық уақыт көрсетқосымша ретінде де өлшенуі мүмкін.

**Стресс-тестілеу.** Стресс-тестілеу — қалыпты қызмет ету шектерін арттыру жағдайында жүйенің сенімділігі мен орнықтылығын бағалайтын бағдарламалық қамтамасыз етуді тестілеу түрлерінің бірі. Стресс-тестілеу әсіресе «күрделі маңызды» бағдарламалық қамтамасыз ету үшін қажет. Әдетте стресс-тестілеу жүйемен үлкен жүктемеде ерекшеліктерді өңдеуді және орнықтылық пен сенімділікті жақсы табады.

Жалпы жағдайда стресс-тестілеудің әдіснамасы сүйемелдеу сатысында күтілгеннен едәуір асатын жүктеме кезінде бағдарламалық қамтамасыз етудің өнімділік көрсеткіштерін талдауға негізделген және оны пайдалану бойынша белсенділік байқалған жағдайда қосымшаның төзімділігі немесе орнықтылығын айқындау мақсатын атқарады.

Стресс-тестілеудің қажеттілігі экстремальды жағдайларда жүйенің бас тарту құны өте жоғары болатын фактормен түсіндіріледі.

Стресс-тестілеуді қолданудың негізгі бағыттары мынадай:

- шыңдық жүктеме кезіндегі жүйелердің мінез-құлқын жалпы зерттеу;
- шыңдық жүктеме кезінде жүйенің қателерін және ерекше жағдайларын өңдеуді зерттеу;
- диспропорциялық жүктеме кезінде жүйенің шағын орындары немесе жекелеген компоненттерді зерттеу;
- жүйе сыйымдылығын тестілеу.

Стресс-тестілеу жүктемелік тестілеу сияқты ұзақ кезеңге жүйенің мінез-құлқын өзгерту серпінін алу мақсатында өнімділік өзгерістерін талдау үшін пайдаланылды. Стресс-тестілеу жекелеген бағдарламалар үшін де, клиент-серверлік сәулетпен бөлінген жүйелер үшін де қолданылады. Бағдарламалық қамтамасыз етудің стресс-тестілеу жағдайы әдетте талаптарды әзірлеу мен тәуекелдерді талдау сатысында айқындалған оның функционалдығының күрделі процестерін ескеріп қалыптасады.

Жалпы жағдайда стресс-тестілеудің жағдайы ретінде желілік ұлғайтылған күтілетін жүктемені пайдалануға болады. Көп буынды бөлінген жүйелерді тестілеу жағдайында көптеген элементтерден тұратын жүктеменің нақты көлемін ғана емес, сонымен бірге жалпы көлемдегі олардың сәйкестігін де ескеру қажет. Стресс-тесттерде диспропорциялық жүктемені пайдалану жүйенің жекелеген компоненттерінің шағын орындарын анықтау үшін де қолданылуы мүмкін.

Сыйымдылықты тестілеу стресс-тестілеудің ең маңызды және мінез-құлық пен талдау тұрғысынан ең күрделі бағыттарының бірі болып табылады. Сыйымдылықты тестілеу өнімділікке талаптарға толық сәйкес келген кезде жүйенің беріктігінің запасын айқындау мақсатында жүргізіледі.

Мұндай жағдайда жүйеге бір уақытта келіп түскен сауалдардың мөлшері мен пропорциясы түрінде ағымдағы жүктеме ретінде де, перспективада күтілетін жүктеме ретінде де ескеріледі. Жүйенің сыйымдылығын тестілеу нәтижесі жүйе сәулетті жобалау кезеңінде әзірленіп, құжатталған өнімділікке қойылатын талаптарға жауап беретін жүктеменің максималды қолжетімді сипаттамасының жиыны болып табылады.

**Тұрақтылықты тестілеу.** Тұрақтылықты тестілеу мақсаты жүктеменің күтілетін деңгейімен ұзақ тестілеу кезінде бағдарламалық қамтамасыз етудің жұмыс қабілетін тексеру болып табылады.

Бағдарламалық қамтамасыз етуді экстремальды жүктемеге бейімдеу алдында болжанған жұмыс жағдайында тұрақтылықты тексеру, яғни өнімді нақтыға жақын жағдайда сынақтан өткізу қажет.

Тестілеуді өткізудің ұзақтығының бірінші дәрежелі мәні жоқ. Негізгі міндет – ресурстарды тұтынуды бақылай отырып, деректерді өңдеу жылдамдығы және (немесе) қосымшаның жауап уақыты тесттің басында және уақыт ағынымен азаймау үшін бақылап отыру. Қарсы жағдайда өнімнің жұмысында іркіліс болуы және жүйенің жүктемесі артуы ықтимал.

Тұрақтылық тестілеуін жиі стресс-тестілеумен қиюластырады. Яғни тұрақтылықты ғана емес, сонымен бірге бағдарламалық қамтамасыз етудің ұзақ уақыт қатаң жағдай мен үлкен жүктемені көтеру қабілетін тексеріледі.

**Конфигурациялық тестілеу.** Конфигурациялық тестілеу – жүйенің өнімділігіне конфигурациядағы өзгерістердің ықпалын тестілеу (тестілеудің алдыңғы түрлерімен салыстырғанда берілетін жүктемені ұлғайту тұрғысынан жүйе). Конфигурациялық тестілеу – бағдарламалық қамтамасыз ету жұмысын түрлі бағдарламалық және аппараттық ортасына тексеру. Тестілеудің бұл түрі егер ақпараттық өнім, мысалы, түрлі платформаларда, түрлі браузерлерде драйверлердің түрлі нұсқасын қолдайтыны белгілі болса, қолданылады.

жоба түріне қарай конфигурациялық тестілеудің түрлі мақсаттары болуы мүмкін:

- өнімділіктің талап етілген сипаттамасын және тестілейтін жүйенің реакция уақытын қамтамасыз ететін жабдықтың оңтайлы конфигурациясын айқындау;
- Тестілеу объектісін ерекшелікте жарияланған жабдыққа,

операциялық жүйелерге және үшінші фирмалардың бағдарламалық өнімдеріне үйлесімділігіне тестілеу объектісін тексеру.

Конфигурациялық тестілеу жүктемелік, стресс- немесе тұрақтылық естілеуімен бірлесуі мүмкін.

## 3.6. КЕМІМЕЛДІК ТЕСТІЛЕУ

*Кемімелдік тестілеу* — бастапқы кодтың тестіленіп қойған учаскелерінен қателерді табуға бағытталған бағдарламалық қамтамасыз етуді тестілеу. Бағдарламаға өзгерістер енгізілгеннен кейін бұрын жұмыс істеп тұрған қызмет етуін тоқтатса, онда мұндай қателер *кемімелдік қателер* деп аталды.

Кемімелдік тестілеу мыналарды көздейді:

- қайта табылған ақауларды түзетуді тексеру;

■ бұрын түзетілген және тексерілген ақау жүйеде қайта қосылмаған кезде тексеру;

■ егер жұмыс істеп тұрған функционалдықтың коды басқа функционалдықта кейбір ақауларды түзеткен кезде қозғалса, оның жұмыс қабілетінің бұзылған-бұзылмағанын тексеру. Әдетте кемімелдік тестілеудің пайдаланылатын әдістері алдыңғы тесттердің қайталама өткізуін, сондай-ақ кемімелдік қателер кодтың қосылуы нәтижесінде кезекті нұсқаға түскен-түспегенін тексеруді қамтиды.

Бір және сол қатенің қайталанып пайда болуы нұсқаларды басқару техникасының әлсіздігінен немесе адамның қателесуінен болады. Бұдан басқа, қандай да бір кодтың бөлігін жазған кезде алдыңғы іске асыруда болған қате жиі шығып отырады.

Сондықтан қатені түзеткен кезде оған тест құрып, бағдарламаның келесі өзгерістері кезінде оны тұрақты жойып отыру дұрыс практика саналады. Кемімелдік практика қолмен орындалса да, көбінесе ол барлық кемімелдік тестті автоматты орындауға мүмкіндік беретін мамандандырылған бағдарламалардың көмегімен жасалады.

Кейбір жобаларда тіпті берілген уақыт аралығы арқылы кемімелдік тесттерді автоматты жүргізу үшін аспаптарды пайдаланады. Автоматты тесттерді пайдалану тексеру рәсімінің жұмысын жеңілдетеді.

Бағдарламалық қамтамасыз етуді тестілеу процесінде *кемімелдік тестің кітапханасы* қалыптасады. Бұл барлық бағдарламаны қамтитын және бағдарламалар оның кезекті нұсқасын тапсырғанда әр кез орындалатын тестердің толық жиыны. Бастапқыда кітапханада тесттер қажет санынан көбірек болады.

Оған шекті жағдайлар мен уақытша сипаттамаларды тексеру үшін мысалдар кіруі тиіс. Кейбір тесттер тестілеуді бірінші жүргізген кезде қатені анықтауы мүмкін, сосын оларды соңғы түзеткеннен кейін сол тестіні қайта орындау пайдасыз. Мұндай тесттерді қате шығарғандарын ғана қалдырып, қалғанын кітапханадан алып тастауға болады.

Кемімелдік тестердің кітапханаларын қалыптастырудың кейбір қағидаларын құруға болады.

1. кітапхананың басқа тестілерімен баламалы тесттерді жою қажет. Дұрысында мұндай тесттер, әрине, кітапханаға түспеуі тиіс. Кітапхананы бірнеше қызметкер құрған жағдайда мұндай жағдайлар орын алуы әбден мүмкін.
2. Объектісі түзетілген қате болып табылатын тесттер санын азайту қажет.

Егер қате немесе оның түрлері тестілеудің тұтастай бірнеше циклінде пайда болса, онда кітапханаға оларды анықтау үшін тесттердің жеткілікті санын қосу қажет. Алайда бағдарламалық қамтамасыз етудің тиісті бөлігі ретінде түзетілген қатені іздеуге бағыталған тесттердің басым бөлігі тестіленсе (қателер жойылса) кітапханадан жоюға болады.

3. Келесі тестілеуде тестілерді құрамдастырған жөн. Тестілеудің тасында мұны жасауға болмайды, алайда одан әрі тесттерді біріктіру жұмысты едәуір жеделдетуге мүмкіндік береді.

4. Тестілеу автоматтандырылған болуы тиіс. Егер тесттердің белгілі бір тобы тестілеудің бірнеше келесі циклдерінің ішінде орындалса, онда бұл процес автоматтандыруды қажет етеді.

5. Кемімелдік кітапхананың барлық тесттерін бағдарламаның ір өзгерісінен кейін орындау міндетті емес. Мұны сирек жасауға болады – әр екінші немесе әр үшінші циклда. Тестілеудің соңғы сатысында бағдарлама шығаруға әзір екеніне көз жеткізу үшін тесттердің максималды көп санын орындаған жөн, басқа циклдердің жартысын немесе барлық тесттің үштен бірін орындаған жеткілікті.

6. Кемімелдік кітапхана әзірленген тесттердің жақсысын қамтуы тиіс, егер ол тым үлкен болса, жаңа тесттердің әзірлеуге уақыт қалмайды. Ал мұнда сол жаңа тесттер әлі табылмаған қателерді барынша таба алады.. Сондықтан кемімелдік кітапхана жұмысты тежегіш емес, тестілеудің тиімділігін арттыру құралы болатындай жоспарлау қажет.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. Ретке келтіру дегеніміз не?
2. Орындау қателері қалай пайда болады?
3. Бағдарламаны өңдеу кезеңі бойынша қателер қалай жіктеледі? Индукция әдісінің мәні қандай?
4. Есептеу нәтижелерінің олқылықтарын жинақтау қателеріне не жатады?
5. Индукция әдісін пайдаланумен ретке келтіру қалай болатынын түсіндіріңіз.
6. Бағдарламаны ретке келтіруде қандай жағдайда кері бақылау әдісі қолданылады?
7. Дедукция әдісін пайдаланумен ретке келтіру қалай болады?
8. Тестілеуге анықтама беріңіз.

9. «Қара жәшікті» тестілеуімен салыстырғанда, «ақ жәшікті» тестілеудің қандай артықшылықтары бар?
10. Бағдарламалық қамтамасыз етуді түрлі деңгейде тестілеу қалай болады?
11. Интеграциялық тестілеудің жүйелі тестілеуден айырмашылығы неде?
12. Модульдік тестілеудің ерекшелігі неден тұрады?
13. Стресс-тестілеудің жүктемелік тестілеуден өзгешелігі неде?
14. Жүйелі тестілеу тесттерінің санатын атаңыз.
15. Шығыс тестілеудің қабылдау тестілеуден айырмашылығы неде?
16. Бета-тестілеудің ерекшеліктері қандай?
17. Альфа-тестілеу қалай жүргізіледі?
18. Бағдарламалық қамтамасыз етудің жүктемелік тестілеуі нәтижесінде алынатын өнімділік көрсеткіштерін атаңыз.
19. Стресс-тестілеу әдіснамасы неге негізделген?
20. Стресс-тестілеу жүргізу неден туындайды?
21. Тұрақтылық тестілеуінің негізгі міндеті қандай?
22. Конфигурациялық тестілеудің мақсаттары қандай?
23. Кемімелдік тестілеу нені қамтиды?
24. Кемімелдік тестілеу кітапханасы дегеніміз не?

# 4 Тарау

# БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУДІ ҚҰЖАТТАНДЫРУ

## 4.1. БҚБЖ ЖӘНЕ Р МЕМСТ ЖАЛПЫ МӘЛІМЕТТЕР

*Бағдарламалық қамтамасыз етудің бірыңғай жүйесі* (БҚБЖ) – бұл бағдарламаларды және бағдарламалық құжаттаманы әзірлеу, ресімдеу және жүгінудің өзара байланысты қағидаларын белгілейтін РФ мемлекеттік стандарттар кешені.

БҚБЖ стандарттарында бағдарламаларды әзірлеу, сүйемелдеу, дайындау және пайдалануды регламенттейтін талаптар белгіленеді, олар:

- бағдарламалармен өзара алмасу үшін және бұрын әзірленген бағдарламаларды жаңа әзірлемелерге қолдану үшін бағдарламалық бұйымдарды біріздендіру;
- еңбек сыйымдылығын азайту және бағдарламалық бұйымдарды әзірлеу, сүйемелдеу, дайындау және пайдалану тиімділігін арттыру;
- бағдарламалық құжаттаманы дайындауды және сақтауды автоматтандыру.

БҚБЖ стандарттары 1970-ші жылдары қабылданып, бізге бастапқыға жақын түрде жетті. Онда ауқымды ЭЕМ пайдаланылған ведомстволық есептеу орталықтары жұмысының практикасы көрсетілген. Компьютерлік жүйемен адамдардың байланысы ол кезде қазіргідей жолға қойылмаған, қазіргі уақытта бұл стандарттардың кейбірі ескірген деп танылды.

19.001 — 77 МЕМСТсәйкес БҚБЖ стандарттары 4.1-кестеде келтірілген топтарға бөлінеді.

4.1-кесте. БҚБЖ стандарттарының тобы

| Топтар коды | Топтың атауы |
|---|---|
| 0 | Жалпы ережелер |
| 1 | Негізін қалайтын стандарттар |
| 2 | Әзірлеу құжаттамасын орындау қағидасы |

*4.1-кестенің соңы*

| Топтар коды | Топтардың атауы |
|---|---|
| 3 | Дайындау құжаттамасын орындау қағидасы |
| 4 | Сүйемелдеу құжаттамасын орындау қағидасы |
| 5 | Пайдалану құжаттамасын орындау қағидасы |
| 6 | Бағдарламалық құжаттаманың айналым қағидасы |
| 7 | Резервтік топтар |
| 8 | |
| 9 | Басқа да стандарттар |

БҚБЖ нормативтік-техникалық құжаттар құрамы 4.2-кестесінде ұсынылған

**4.2-кесте. БҚБЖ құрамы**

| Белгілеу | Атауы |
|---|---|
| МЕМСТ19.001 — 77 | БҚБЖ. Жалпы ережелер |
| МЕМСТ 19.002 — 80 | БҚБЖ. Алгоритмдер мен бағдарламалар схемасы. Орындау қағидасы |
| МЕМСТ19.004 — 80 | БҚБЖ. Терминдер мен айқындама |
| МЕМСТ 19.005 — 85 | БҚБЖ. Алгоритмдер мен бағдарламалардың Р-схемасы Шартты графикалық және орындау қағидасын белгілеу |
| МЕМСТ 19.101 — 77 | БҚБЖ. Бағдарламалар және бағдарламалық құжаттар түрлері |
| МЕМСТ19.102 — 77 | БҚБЖ. Әзірлеу сатысы |
| МЕМСТ19.103 — 77 | БҚБЖ. Бағдарламалар мен бағдарламалық құжаттарды белгілеу |
| МЕМСТ19.104 — 78 | БҚБЖ. Негізгі жазбалар |
| МЕМСТ19.105 — 78 | БҚБЖ. Бағдарламалық құжаттарға ортақ талаптар |
| МЕМСТ19.106 — 78 | БҚБЖ. Баспа тәсілімен орындалған бағдарламалық құжаттарға қойылатын талаптар |
| МЕМСТ19.201 — 78 | БҚБЖ. Техникалық тапсырма. Мазмұнға және ресімдеуге қойылатын талаптар |

*4.2-кестенің соңы*

| белгілеу | Атауы |
|---|---|
| МЕМСТ19.202 — 78 | БҚБЖ. Ерекшелік. Мазмұны мен ресімдеуге қойылатын талаптар |
| МЕМСТ 19.301 — 79 | БҚБЖ. Сынақтар бағдарламасы және әдістемесі. Мазмұны мен ресімдеуге қойылатын талаптар |
| МЕМСТ 19.401 — 78 | БҚБЖ. Бағдарлама мәтіні . Мазмұны мен ресімдеуге қойылатын талаптар |
| МЕМСТ 19.402 — 78 | БҚБЖ. Бағдарлама сипаты |
| МЕМСТ 19.403 — 79 | БҚБЖ. Түпнұсқаны ұстаушылар ведомості |
| МЕМСТ 19.404 — 79 | БҚБЖ. Түсіндірме жазба. Мазмұны мен ресімдеуге қойылатын талаптар |
| МЕМСТ19.501 — 78 | БҚБЖ. Формуляр. Мазмұны мен ресімдеуге қойылатын талаптар |
| МЕМСТ19.502 — 78 | БҚБЖ. Қолдануды сипаттау. Мазмұны мен ресімдеуге қойылатын талаптар |
| МЕМСТ 19.503 — 79 | БҚБЖ. Жүйелі бағдарламашыға басшылық. Мазмұны мен ресімдеуге қойылатын талаптар |
| МЕМСТ 19.504 — 79 | БҚБЖ. Бағдарламашы басшылығы. Мазмұны мен ресімдеуге қойылатын талаптар |
| МЕМСТ19.505 — 79 | БҚБЖ. Оператордың басшылығы. Мазмұны мен ресімдеуге қойылатын талаптар |
| МЕМСТ 19.506 — 79 | БҚБЖ. Тілді сипаттау. Мазмұны мен ресімдеуге қойылатын талаптар |
| МЕМСТ 19.507 — 79 | БҚБЖ. Пайдалану құжаттарының ведомості |
| МЕМСТ 19.508 — 79 | БҚБЖ. Техникалық қызмет көрсету бойынша басшылық. Мазмұны мен ресімдеуге қойылатын талаптар |
| МЕМСТ 19.601 — 78 | БҚБЖ. Қайталау, есепке алу және сақтаудың жалпы қағидалары |
| МЕМСТ19.602 — 78 | БҚБЖ. Баспа тәсілімен орындалған бағдарламалық құжаттарды қайталау, есепке алу және сақтау қағидасы |
| МЕМСТ 19.603 — 78 | БҚБЖ. Өзгерістерді енгізудің жалпы қағидасы |
| МЕМСТ19.604 — 78 | БҚБЖ. Баспа тәсілімен орындалған бағдарламалық құжатқа өзгерістер енгізу қағидасы |

*МЕМСТ19.101 — 77 (1626 — 79) «БҚБЖ. Бағдарламалар мен бағдарламалық құжаттар түрлері» (өзгерістермен 1987 жылғы қарашада қайта шығарылды).* Бұл стандарт олардың мақсаты мен қолдану саласынан тыс есептеу машиналары, кешендер мен жүйелер үшін бағдарламалар мен бағдарламалық құжаттар түрлерін анықтайды. Бұл стандартта бағдарламалық компонентке және бағдарламалар кешеніне мынадай айқындамалар беріледі.

*Компонент* — аяқталған функцияны орындайтын және дербес немесе кешен құрамында қолданылатын біртұтас ретінде қаралатын бағдарлама.

*Кешен* — өзара байланысты функцияларды орындайтын және дербес немесе басқа кешен құрамында қолданылатын екі немесе одан көп компоненттер және (немесе) кешендерден тұратын бағдарлама.

4.3-кестеде келтірілген бағдарламалық өнімге құжаттама бөледі.

Пайдалану құжаттамасының түрлері мен оған қойылатын талаптар 4.4-кестесінде ұсынылған.

4.3-кесте. Бағдарламалық құжаттама түрлері

| | |
|---|---|
| Ерекшелігі | Бағдарлама құрамы және оған құжаттама |
| Түпнұсқаны ұстаушылар ведомості | Бағдарламалық құжаттардың түпнұсқасы сақталатын кәсіпорындар тізбесі |
| Бағдарлама мәтіні | Қажетті пікірлермен бағдарлама жазбасы |
| Бағдарлама сипаттамасы | Бағдарламаның логикалық құрылымы және қызмет етуі туралы мәлімет |
| Сынақтар бағдарламасы мен әдістемесі | Бағдарламаны зерттеу кезінде тексеруге жататын талаптар, сондай-ақ оларды бақылау тәртібі мен әдістері |
| Техникалық тапсырма | Бағдарламаның мақсаты және саласы, бағдарламаға қойылатын техникалық, техникалық-экономикалық және арнайы талаптар, әзірлеудің қажетті сатылары мен мерзімдері, сынақ түрлері |
| Түсіндірме жазба | Алгоритм схемасы. Бағдарламаның алгоритмі және (немесе) қызмет етуінің жалпы сипаттамасы, сондай-ақ қабылданған техникалық және техникалық-экономикалық шешімдер негіздемесі |
| Пайдалану құжаттары | Бағдарламаның қызмет ету және пайдалануын қамтамасыз ету үшін мәліметтер |

4.4-кесте. Пайдалану құжаттамасының түрлері

| | |
|---|---|
| Пайдалану құжаттамасының ведомості | Бағдарламалық өнімге пайдалану құжаттарының тізбесі |
| Формуляр | Бағдарламаның негізгі сипаттамасы, бағдарламаны пайдалану туралы мәліметтер және комплектілік |
| Қолдануды сипаттау | Бағдарламаның мақсаты, қолдану саласы, қолданылатын әдістер, шешілетін міндеттер сыныбы, қолдану үшін шектеулер, техникалық құралдардың минималды конфигурациясы |
| Жүйелі бағдарламашының басшылығы | Нақты қолдану жағдайында бағдарламаны тексеру, қызмет етуін қамтамасыз ету және баптау үшін мәліметтер |
| Бағдарламашының басшылығы | Бағдарламаны пайдалану үшін мәліметтер |
| Оператор басшылығы | Бағдарламаны орындау процесінде есептеу жүйесімен оператордың араласу рәсімдерін қамтамасыз ету үшін мәліметтер |
| Тіл сипаттамасы | Тілдің синтаксисі мен семантикасын сипаттау |
| Техникалық қызмет көрсетуге басшылық | Техникалық құралдарға қызмет көрсеткен кезде тестілік және диагностикалық бағдарламаларды қолдану үшін мәлімет |

БҚДЖ-ның негізгі кемшіліктеріне мыналарды жатқызуға болады:

- Бағдарламалық қамтамасыз етудің тіршілік циклінің жалғыз, «каскадты» моделіне бағдарлау;
- бағдарламалық қамтамасыз ету сапасының сипаттамасын құжаттау бойынша нақты ұсыныстардың болмауы;
- тіршілік циклі және тұтас алғанда, өнімді, мысалы БҚБЖ құжаттандыру бойынша стандарттардың басқа да қолданыстағы отандық жүйелерімен жүйелі үйлесуінің болмауы;
- тауарлық өнім ретінде бағдарламалық қамтамасыз етуді қанаттандыруға айқын көрсетілмеген тәсіл;
- халықаралық және өңірлік стандарттар ұсыныстарымен келісілген бағдарламалық қамтамасыз етуге перспективалық құжаттардың құрамы, оларды ұстау және ресімдеу бойынша ұсыныстардың болмауы.

РФ аумағында БҚБЖ қолданудың тек қана ұсынымдық сипаты бар,

яғни БҚБЖ  егер шартта, келісімшартта, жекелеген заңдарда, сот шешімінде және т.б. өзгеше көзделмес, ерікті негізде қолданылады.

Заманауи баламалы БҚБЖ орыс тіліне аударылған және Ресейде ұлттық құқықта қабылданған жүйелі және бағдарламалық инженерия саласындағы кейбір ИСО/ХЭК стандарттары болып табылады. Одан әрі осы стандарттардың кейбірінің мақсаты келтіріледі.

*Р ИСО/ХЭК 9294 МЕМСТ «Ақпараттық технология. Бағдарламалық қамтамасыз етуді құжаттауды басқару бойынша басшылық*». Стандарт олардың құрылуына жауап беретін басшылар үшін бағдарламалық қамтамасыз ету құжаттауын тиімді басқару бойынша ұсынысты белгілейді. Стандарттың мақсаты мыналарда көмек көрсету болып табылады:

- бағдарламалық қамтамасыз етуді құжаттау стратегиясын айқындау;
- құжаттау бойынша стандарттар таңдау;
- құжаттау рәсімдерін таңдау, қажетті ресурстарды айқындау;
- құжаттау жоспарларын жасау.

*Р ИСО/ХЭК 9126 МЕМСТ «Ақпараттық технология. Бағдарламалық өнімді бағалау. Оларды қолдану бойынша сапа мен басшылық сипаттамасы».* ИСО/ХЭК 9126 стандарты халықаралық стандартқа толықтай сәйкес келеді. Оның мәнмәтінінде сапа сипаттамасы дегеніміз «оның сапасы сипатталып, бағаланатын бағдарламалық өнім қасиеттерінің (атрибуттарының) жиыны».

*Р ИСО 9127—94 МЕМСТ «Ақпараттарды өңдеу жүйесі. Пайдаланушының құжаттамасы және тұтынушылық бағдарламалық пакеттер үшін орамада ақпарат».* ИСО 9127:1989 стандарты халықаралық стандартына толықтай сәйкес келеді. Бұл стандарттың мәнмәтінінде орындау үшін жобаланатын және сатылатын бағдарламалық өнім, сату үшін оралған бағдарлама мен оған тиісті құжаттама біртұтас ретінде саналады»

Пайдаланушының құжаттамасы дегеніміз – соңғы тұтынушыны бағдарламалық пакетті орнату және пайдалану бойынша ақпаратпен қамтамасыз ететін құжаттама. Орамадағы ақпарат дегеніміз бағдарламалық пакеттің сыртқы орамасында көрсетілетін ақпарат. Оның мақсаты әлеуетті сатып алушыға бағдарламалық пакет туралы бастапқы мәліметтерді беру болып табылады.

*Р ИСО/ХЭК 8631 МЕМСТ «Ақпараттық технология. Бағдарламалық құрамалар және оларды көрсету үшін шартты белгілер».* Рәсімдік алгоритмдер ұсынымын сипаттайды.

*Р ИСО/ХЭК 12119 МЕМСТ «Ақпараттық технология. Бағдарламалық құралдар пакеті. Сапа мен сынаққа қойылатын талаптар».*

Бұл стандартта бағдарламалар пакетінінің сапасына қойылатын талаптар және оларды берілген талаптар сәйкестігіне сынақтан өткізу бойынша нұсқаулықтар белгіленген.

«Бағдарламалық құралдар пакеті» ұғымы жалпы қолдану немесе жұмыс істеу үшін бірнеше пайдаланушы жеткізген бағдарламалар, рәсімдер және қағидалар жиынтығы ретінде қаралатын «бағдарламалық өнім» барынша жалпы ұғыммен нақты теңдестіріледі. Әр бағдарламалар пакетінің өнімге сипаты және пайдаланушылық құжаттамасы болуы тиіс.

*Р ИСО/ХЭК 9127 МЕМСТ «Пайдаланушының құжаты және тұтынушылық бағдарламалық пакеттер үшін орамадағы ақпарат».*

Стандарт пайдаланушының сипаттамасын және тұтынушылық бағдарламалық пакеттер жарақталуы тиіс орамадағы ақпаратты сипаттайды. Пайдаланушының құжаттамасы пайдаланушыларды бағдарламалық құралды орнату және өткізу үшін қажетті ақпаратпен қамтамасыз етеді. Әдетте бұл құжаттаманы ораманың ішіне бағдарламалық құралмен бірге салатын бір немесе бірнеше басшылық түрінде ұсынады. Нәтижесінде пайдаланушылар олар пакетті сатып алмайынша басшылықты қолдана алмайды. Пайдаланушы құжаттамасының мақсаты соңғы тұтынушыны бағдарламалық құралдың мақсаттары, функциясы және сипаттамасын анық түсіну үшін, оны қолданысқа қалай енгізіп, бағдарламалық құралды қалай пайдаланытыны туралы жеткілікті ақпаратпен қамтамасыз ету болып табылады.

Оқу құжаттамасының мақсаты жаңа немесе тәжірибесі жоқ пайдаланушылар үшін пакет жұмысына біртіндеп енгізу мүмкіндігін беру болып табылады.

Пакеттің сыртқы орамасындағы ақпараттың мақсаты әлеуетті сатып алушыға олардың сұраныстарына сәйкес осы бағдарламалық қамтамасыз етудің жарамдылығы туралы шешім қабылдау мүмкіндігін беру болып табылады.

## 4.2. БАҒДАРЛАМАЛЫҚ ҚҰРАЛДАРДЫҢ ТІРШІЛІК ЦИКЛІНІҢ ПРОЦЕСТЕРІ

Р ИСО/ХЭК 12207 МЕМСТ стандарты бағдарламалық индустрияда бағдарлануға болатын бағдарламалық құралдардың тіршілік циклінің процестерінің жалпы құрылымын белгілейді.

Бұл стандарт бағдарламалық өнімді немесе қызметтерді сатып алу кезінде, сондай-ақ бағдарламалық өнімді жеткізу, әзірлеу, мақсаты бойынша қолдану, сүйемелдеу және қолдануды тоқтату кезінде пайдаланылатын процестерді, қызмет түрлері мен міндеттерді айқындайды.

Стандарт жүйелерді, бағдарламалық өнімдерді сатып алған кезде, сондай-ақ сол ұйымнан не басқа жерден бағдарламалық-аппараттық құралдардың бағдарламалық өнімдері мен бағдарламалық компоненттерін жеткізу, әзірлеу, пайдалану және сүйемелдеу кезінде қолданылады. Стандарт бағдарламалық өнімдер мен қызметтердің мәнін түсінуді қамтамасыз ету үшін қажетті жүйелерді сипаттау аспектілерін де қамтиды.

Стандарт тіпті егер екі тарап та бір ұйымға тиесілі, сондай-ақ өзін-өзі бақылау үшін тараптардың бірі болса да, тараптардың екіжақты қатынасы кезінде қолданыла береді.

Стандарт:

- жүйелерге, бағдарламалық өнімдерге және қызметтерге тапсырыс берушілерге;
- жеткізушілер, әзірлеушілер, операторларға;
- сүйемелдеу персоналына;
- жобалар әкімшілеріне;
- сапасына жауап беретін әкімшілерге;
- бағдарламалық өнімдерді пайдаланушыларға арналған.

Стандарт шығарылатын құжаттаманың атауын, форматтары немесе егжей-тегжейлі мазмұнын айқындауға арналмаған. Бұл мәселелерді шешу осы стандартты пайдаланушылардың қарауына қалдырылды.

Стандарт тіршілік циклінің нақты моделін немесе бағдарламалық құрал әзірлеудің әдісін айқындамайды. Бұл стандартты қолданатын пайдаланушылар өзінің бағдарламалық жобасына қатысты тіршілік циклі моделін өздері таңдайды және осы моделде осы стандарттан таңдалған процестерді, жұмыстар мен міндеттерді бөледі; бағдарламалық құралдар әзірлеу әдістерін таңдайды және қолданады, нақты бағдарламалық жобаға тиісті жұмыстар мен міндеттерді орындайды.

Өзінің мақсаттарына қарай нақты ұйым өзінің нақты міндеттерін орындау үшін тиісті көптеген процестерді таңдай алады, сондықтан бұл стандартты нақты ұйымға, жоба немесе қосымша үшін бейімдеген жөн.

Р ИСО/ХЭК 12207 МЕМСТтүпкі ойдан кәдеге жаратқанға дейін бағдарламалық құралдың тіршілік циклінің жоғарғы деңгейінің сәулетін белгілейді. Сәулет процес деректері арасындағы көптеген процестер мен өзара байланыстардан тұрады.

Негізінен, әр процес тіршілік циклінің бірегей функцияларды іске

Р ИСО/ХЭК 12207 МЕМСТ әр процесі тараптардың жауаптылығы (міндеттілігі) тұрғысынан қаралған. Ұйым бір немесе бірнеше процестерді орындай алады. Процесті бір немесе бірнеше ұйымдар орындай алады, бұл ретте ұйымдардың бірі жауапты тарап ретінде айқындалуы тиіс.

Тіршілік циклі сәулетіндегі жауаптылық қағидаты көптеген тұлға тартылуы мүмкін нақты жоба үшін Р ИСО/ХЭК 12207 МЕМСТ қолданбалы қолдануды жеңілдетеді.

Процес үш жалпы сыныпқа топтастырылған:

1) негізгі;
2) қосалқы;
3) ұйымдастырушылық.

Практикада әр процес оның жұмысын құрайтын терминдерде айқындалуы тиіс, әрқайсысы оның міндеттерін құрайтын терминдерде айқындалуы тиіс. Процесте жұмыс байланысты міндеттер жиынынан тұрады. Р ИСО/ХЭК 12207 МЕМСТ стандартында процестер, жұмыстар мен міндеттер белгіленген. Ондағы процестер, жұмыстар мен міндеттер неғұрлым жалпы табиғи ұстанымдық дәйектілікте сипатталған.

Бұл дәйектілік тіршілік циклінің моделін іске асыру дәйектілігін айқындамайды. Сипатталған дәйектілік жобада жобаға тән таңдау, реттеу, қолдану және қайталау бағдарламалық құралын немесе оған сай процестер, жұмыстар (қызмет түрлері) және міндеттер (тапсырмалар) құруға арналған.

Р ИСО/ХЭК 12207 бағдарламалық қамтамасыз етудің тіршілік циклын қамтитын процестердің жалпыға бірдей интеграцияланған жиынына қойылатын талаптарды белгілейді. Бұл стандарт әр процес үшін жетілу процесі арқылы «жоспар – іске асыру – тексеру - акт» циклына қолжетімділікті қамтамасыз етеді.

Бұл ретте сапасына байланысты және бағдарламалық қамтамасыз ету тіршілік циклының ажырамас бөлігі ретінде тіршілік циклының тиісті процесіне кіреді. Осылайша, оның іске асырылуына жауап беретін әр процес пен персоналға сапаға байланысты осы процес аясында жұмыс бекітілген.

Р ИСО/ХЭК 12207 МЕМСТ:

- тіршілік циклінің кез келген моделіне (мысалы, каскадты, инкрементті немесе эволюциялық);
- Бағдарламалық инженерия әдісі немесе технологиясына (мысалы, объектілік-бағдарланған жобалау, құрылымдық бағдарламалау, төмендемелі тестілеу немес макеттеу)

■ бағдарламалау тілдеріне жарамды болып табылады.

Бұл мәселелерді шешу жобаның өзіне және технологияның заманауи жай-күйіне байланысты, ал бұл элементтерді таңдауды Р ИСО/ХЭК 12207 МЕМСТ пайдаланушы жүзеге асырады.

Стандарт жалпы көзқарас тұрғысынан икемді болып табылады, яғни тіршілік циклі процесінің жұмыстары (қызмет түрлері) мен міндеттері «Қалай істеу керек?» деген емес, «Не істеу керек?» деген сұрақтарға жауап береді. Басқаша айтқанда, міндеті «төмендемелі функционалдық жобалау әдісін пайдалана отырып сәулет жобасын әзірлеу немесе құжаттамалық ресімдеу» емес, «сәулет жобасын әзірлеу және құжаттамалық ресімдеу» болуы тиіс. Бұл схема тапсырыс берушіге соңғы тұтынушы немесе қызметке қойылатын талаптарды белгілеу үшін кең мүмкіндіктер береді және сонымен қатар сатушыға өнімді құру немесе қызмет көрсету үшін тиісті әдістер, тәсілдер мен аспаптар әзірлеу мен қолдануға мүмкіндік береді.

Р ИСО/ХЭК 12207 МЕМСТ құжаттандыру саласындағы стандартқа жатпайды, яғни тіпті егер көрсетілген стандартта процестердің кейбір шығыс нәтижелерін құжаттандыруға қойылатын талаптар белгілесе де, ол құжаттардың форматы немесе мазмұнын айқындамайды. Бұл стандарт жоспар, ерекшелік (техникалық тапсырма) немесе тестілеуге қойылатын талаптар сияқты ұқсас шығыс нәтижелерді қалай біріктіретінін белгілемейді.

Р ИСО/ХЭК 12207 МЕМСТ бағдарламалық құрал (сүйемелдеу сенімділігі немесе қолайлылығы сияқты) және көрсеткіштер (метрика) қасиетін (атрибуттар) айқындамайды немесе бермейді.

Бұл стандарт бағдарламалық құралдың осындай қасиеттері, айқындау үшін тәсілдер сипаттайды, бірақ оларды Р ИСО/ХЭК 12207 МЕМСТпайдаланушылары нақтылауы тиіс.

Р ИСО/ХЭК 12207 МЕМСТ жүйелерді бағдарламалық қамтамасыз ету жобалауын қатаң жүйелендірген басқаруды алмастырмайды.

Көрсетілген стандарт бағдарламалық құралдармен байланысты процестер, жұмыстар мен міндеттердің тиісті дәрежеде айқындалуы, жоспарлануы және орындалуы мүмкін құрылымды айқындайды.

Р ИСО/ХЭК 12207 МЕМСТ нақты айқындалған сындарлы блоктар (процестер) жиынын қамтиды. Стандартты пайдаланушы бұл блоктарды тиісінше өз ұйымының және жобасының мақсаттары мен міндеттеріне таңдауы, іс жүзінде қолдануы және құрастыруы қажет.

Бағдарламалық құралдарды құру жобаларын іске асыру жағдайларында Р ИСО/ХЭК 12207 МЕМСТ практикалық қолдану бойынша ұсынымдар «Ақпараттық технология. Р ИСО/ХЭК 12207 МЕМСТ қолдану бойынша басшылық» Р ИСО/ХЭК ТО 15271 МЕМСТ стандартында қамтылған.

## 4.3. ТЕХНИКАЛЫҚ ТАПСЫРМА. МАЗМҰНЫНА ҚОЙЫЛАТЫН ТАЛАПТАР

Техникалық тапсырма бағдарламалық қамтамасыз етуге қойылатын талаптар жиынтығы және ол әзірленетін бағдарламаны тексеру және қабылдау критерийі ретінде пайдаланылуы мүмкін, сондықтан тапсырыс беруші мен әзірлеуші толық жасап (қосымша бөлімдер енгізу мүмкіндігін ескеріп), қабылдаған техникалық тапсырма жобанің негізгі құжаттарының бірі болып табылады.

Бағдарламалық өнімді әзірлеуге техникалық тапсырманы сауатты жасай білу бағдарламашы маманның кәсіби деңгейін айқындайды және оны тапсырыс беруші тарапынан негізсіз талаптардан қорғайды.

*Техникалық тапсырма* — бағдарламалық өнімді әзірлеудің негізгі мақсаттары мен оларға қойылатын талаптарды қалыптастыратын құжат. Әзірлеу мерзімдері мен кезеңдерін айқындайды және қабылдау-тапсыру сынақтарының процесін регламенттейді. Осы құжаттың негізінде тапсырыс берушінің бастапқы талаптары, жоба алды зерттеуді орындау нәтижелері және т.б. жатыр.

Техникалық тапсырманы әзірлеу мынадай дәйектілікпен орындалады.

Алдымен орындалатын функция жиыны, тізбе және бастапқы деректердің сипаттамасы орнатылады.

Сосын нәтижелер тізбесі, олардың сипаттамасы және оларды ұсыну тәсілдері айқындалады.

Одан әрі бағдарламалық қамтамасыз етудің жұмыс істеу ортасын нақтылайды (нақты жинақталым, техникалық құралдар параметрлері, пайдаланылатын операциялық жүйенің нұсқасы және келешек бағдарламалық өніммен өзара іс-қимылда болатын басқа орнатылған бағдарламалық қамтамасыз ету нұсқасы мен параметрлері).

Әзірленіп жатқан бағдарламаны қамтамасыз ету кей ақпаратты жинап және сақтап немесе басқаруға қандай да бір техникалық процеспен қосылса, жабдықтар мен энергия жабдықтарының ақауы орын алғанда бағдарламаның іс-қимылын нақты регламенттеу қажет.

Әзірленіп жатқан бағдарламалық қамтамасыз етудің сипаттамасын айқындайтын негізгі факторлар мыналар болып табылады:

- бағдарлама немесе жүйенің функцияларын айқындайтын бастапқы деректер және талап етілетін нәтижелер;
- әзірленіп жатқан бағдарламалық қамтамасыз ету жұмыс істейтін орта (бағдарламалық және аппараттық); ол берілуі және техникалық тапсырмада көрсетілген параметрлерді қамтамасыз ету үшін таңдалуы мүмкін.
- басқа бағдарламалық қамтамасыз етумен және (немесе) нақты техникалық құралдармен ықтимал өзара іс-қимылда болуы да мүмкін, сондай-ақ орындалатын функциялар жиынын ескеріп, таңдалуы мүмкін.

*МЕМСТ19.106 — 78 «Техникалық тапсырма. Ұстауға және ресімдеуге қойылатын талаптар»* олардың мақсаты мен қолдану саласынан тыс есептеу машиналары, кешендері мен жүйелері үшін бағдарламаны немесе бағдарламалық қамтамасыз етуді әзірлеуге техникалық тапсырманы қалыптастыру және ресімдеу тәртібін анықтайды.

Стандартқа сәйкес «Техникалық тапсырма» құжаты мынадай бөлімдерді қамтуы тиіс.

1. Кіріспе.
2. Әзірлеу үшін негіз.
3. Әзірлеу мақсаты.
4. Бағдарламаға және бағдарламалық бұйымға қойылатын талаптар.
5. Бағдарламалық құжаттарға қойылатын талаптар.
6. Техникалық-экономикалық көрсеткіштер.
7. Әзірлеу сатылары мен кезеңдері.
8. Бақылау және қабылдау тәртібі.
9. Қосымша (қажетіне қарай).

Бағдарламалық қамтамасыз етудің ерекшеліктеріне қарай бөлімдердің мазмұнын нақтылауға, жаңа бөлімдерді енгізуге немесе олардан бөлектерді біріктіруге жол беріледі.

«Енгізу» бөлімінде әзірленіп жатқан бағдарламалық өнімнің мақсаты, бағдарламалық қамтамасыз етуді және бағдарламалық қамтамасыз етуді пайдаланатын объектіні қолдану саласындағы қысқаша сипаттаманы көрсетеді. «Әзірлеу үшін негіз» бөлімінде мыналар көрсетілуі тиіс:

- негізінде әзірлеу жүргізілетін құжат (құжаттар);

■ әзірлеме тақырыбының атауы және (немесе) шартты мәні.

«Әзірлеменің мақсаты» бөлімінде бағдарламалық қамтамасыз етудің функционалдық және пайдалану мақсаты көрсетілген.

«Бағдарламаға немесебағдарламалық бұйымға қойылатын талаптар» бөлімінде мынадай қосымша бөлімдер болуы тиіс:

1. Функционалдық сипаттамаларға қойылатын талаптар.
2. Сенімділікке қойылатын талаптар.
3. Пайдалану шарты.
4. Техникалық құралдар құрамына және параметрлеріне қойылатын талаптары.
5. Ақпараттық және бағдарламалық үйлесімділікке қойылатын талаптар.
6. Таңбалауға және орамаға қойылатын талаптар.
7. Тасымалдауға және сақтауға қойылатын талаптар.
8. Арнайы талаптар.

«Функционалдық сипаттамаларға қойылатын талаптар» қосымша бөлімінде орындалатын функциялар, кіретін және шығатын деректерді ұйымдастыру құрамына және уақытша сипаттамаға, т.б. қойылатын талаптар көрсетілуі тиіс. Кіретін деректерге қойылатын талаптарды сипаттаған кезде кіретін деректердңсипаты, ұйымдастырылуы мен алдын ала дайындығы, кіретін деректерді кодтау форматы, сипаттамасы, тәсілдері көрсетілуі тиіс.

Бағдарламаның кіретін ақпараты бастапқы құжаттарды (жүккұжат, есептер және т.б.), нормативтік-анықтамалық ақпараттар (анықтамалар, жіктеушілер, кодификаторлар және т.б.), электрондық құжаттар, кіретін дабылдар және т.б. болуы мүмкін. Бағдарламаның кіретін ақпараты құжаттар (электрондық немесе қағаз), деректер файлы, кіретін дабылдар және т.б. болуы мүмкін. Кіретін деректерге қойылатын талаптарды сипаттаған кезде кіретін деректердің сипаты, ұйымдастыруы, кіретін деректерді кодтаудың форматы, сипаттамасы және тәсілі көрсетіледі.

Негізгі функциялардан басқа техникалық тапсырмада жүйенің баптауышын (конфигурациясын) түзету мүмкіндігі, деректерді резервтік сақтау мүмкіндігі, жүйеге кіру паролін өзгерту мүмкіндігі, күнтізбені, калькуляторды, редакторды және т.б. бағдарламадан шықпай шақыру мүмкіндігі сияқты бағдарламаның сервистік функцияларына қойылатын талаптарды сипаттайды.

«Сенімділікке қойылатын талаптар» қосымша бөлімінде сенімді жұмыс істеуді (орнықты жұмыс істеуді, кіретін, шығатын ақпаратты бақылауды, бас тартқаннан кейін қалпына келтіру уақытын қамтамасыз ету және т.б. ) қамтамасыз етуге қойылатын талаптар көрсетілген.

«Пайдалану шарты» қосымша бөлімінде берілген сипаттамалар, сондай-ақ өызмет көрсету түрі, персоналдың қажетт саны мен біліктілігі қамтамасыз етілуі тиіс пайдалану шарты (қоршаған ауа температурасы, деректер тасығыштың таңдалған түрлері үшін салыстырмалы ылғалдылық) көрсетілуі тиіс.

«Техникалық құралдар құрамына және параметрлеріне қойылатын талаптар» қосымша бөліміне олардың негізгі техникалық сипаттамаларын көрсете отырып, техникалық құралдардың қажетті құрамын келтіреді.

«Ақпараттық және бағдарламалық үйлесімділікке қойылатын талаптар» қосымша бөлімінде кірген, шыққан кезде ақпараттық құрылымға, шешу әдістеріне, бастапқы кодқа, бағдарлама қолданатын бағдарламалау жіне бағдарламалық құралдар тіліне қойылатын талаптар, әзірленіп жатқан бағдарламалық өнім жұмыс істейтін операциялық жүйелер мен орталарға қойылатын талаптар компьютерге бағдарламалар пакеті – қосымшалар (осы бағдарламалық өнімді әзірлеу, жаңғырту немесе пайдалану үшін) әзірлеу құралдарын орнату қажеттілігі, түрлі графикалық компоненттер инсталляциясына сұраныс және т.б. көрсетілуі тиіс.

Қажет болған жағдайда ақпарат және бағдарламаларды қорғау қамтамасыз етілуі тиіс.

«Таңбалауға және орауға қойылатын талаптар» қосымша бөлімінде жалпы жағдайда бағдарламалық бұйымды таңбалауға, талаптар, орау нұсқалары мен тәсілдері көрсетіледі.

«Тасымалдауға және сақтауға қойылатын талаптар» қосымша бөлімінде бағдарламалық бұйым үшін тасымалдау шарты, сақтау орны, сақтау шарты, қоймалау шарты, түрлі жағдайда сақтау мерзімі көрсетілуі тиіс.

«Бағдарламалық құжаттамаға қойылатын талаптар» бөлімінде бағдарламалық құжаттаманың алдын ала құрамы және қажет болған жағдайда, оларға арнайы талаптар келтірілуі тиіс.

«Техникалық-экономикалық көрсеткіштер» бөлімінде бағдарланған экономикалық тиімділік, болжалды жылдық сұраныс, үздік отандық және шетелдік үлгілер және оның аналогтарымен салыстырғанда, әзірлеудің экономикалық артықшылықтары көрсетілуі тиіс.

«Әзірлеу сатылары мен кезеңдері» бөлімінде жұмыс (әзірленуі, келісілуі және бекітілуі тиіс бағдарламалық құжаттар тізбесі) әзірлеудің қажетті сатысы, кезеңдері, мазмұны, сондай-ақ әдетте, әзірлеу мерзімдері белгіленеді және орындаушыларды айқындайды.

«Бақылау және қабылдау тәртібі» бөлімінде сынақтар түрлері мен жұмысты қабылдауға қойлатын жалпы талаптар көрсетілуі тиіс.

Техникалық тапсырмаға қосымшаларда, қажет болған жағдайда:

- әзірлеуді негіздейтін ғылыми-зерттеу және басқа да жұмыстар тізбесі;
- әзірлеу кезінде пайдаланылатын алгоритмдер, кестелер, сипаттамалар, негіздемелер, есептер және тасқа да құжаттар тізбесі келтіріледі.

## 4.4. БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУ ТАЛАПТАРЫНЫҢ ЕРЕКШЕЛІГІ

Кез-келген бағдарламалық қамтамасыз етуді әзірлеу болашақ бағдарламалық өнімге қойылатын талаптарды талдаудан басталады. Талдау нәтижесінде әзірленетін бағдарламалық өнім таталпатырының ерекшіліктері алынуы тиіс, ол үшін шешілетін міндеттерді бөлшектеп байланыстыру және мазмұнын қою орындалады, олардың өзара іс-қимыл жасасуын және пайдаланушылық шектеулері нақтыланады. Жалпы ерекшелікті айқындау процесінде шынайы әлемнің кейбір бөлігі ретінде пәндік аймақтың жалпы моделін құрады, онымен әзірленетін бағдарламалық қамтамасыз ету қандай да бір тәсілмен өзара іс-қимыл жасасатын болады және оның негізгі функцияларын нақтылайды.

Бағдарламалық қамтамасыз етуге қойлатын талаптардың ерекшелігі — әзірленуі қажет бағдарлама әрекетін соңғы сипаттау.

*«Бағдарламалық қамтамасыз етуге қойылатын талаптардың ерекшеліктерін құру әдістемесі» IEEE 830* стандартында (Электротехника және радиоэлетроника жөніндегі инженерлер институты ұсынған) бағдарламалық қамтамасыз етуге қойылатын талаптардың дұрыс құрылған ерекшеліктерінің мазмұны және сапалы сипаттамасы сипатталады (Software Requirements Specification — SRS) және бірнеше үлгілі SRS келтіріледі. Осы ұсынылатын әдістеме мақсаты әзірленетін бағдарламалық қамтамасыз етуге талаптар белгілеу, бірақ меншік және коммерциялық бағдарлама өнімдерін таңдауда көмек көрсете алады.

Бұл әдістеме бағдарламалық қамтамасыз етуге қойылатын талаптардың ерекшеліктерін құруға ұсынылатын қағидаттарды сипаттайды. Ол бағдарламалық қамтамасыз ету ерекшелігі процесінің нәтижесі бір қатарлы және толығымен ерекше құжат болып табылатын үлгіге негізделген. Ол бағдарламалық қамтамасыз етуге тапсырыс берушіге не алғысы келетінін нақты сипаттауға, ал бағдарламалық қамтамасыз етуді жеткізушіге тапсырыс беруші не қалайтынын нақты білуге көмектесуге арналған.

Бұл әдіс бағдарламалық қамтамасыз етудің меншік ұйымдары үшін оның стандартты ерекшелігінің макетін әзірлеуге, бағдарламалық қамтамасыз етуге қойылатын талаптардың нақты ерекшеліктерінің форматы мен мазмұнын айқындауға мүмкіндік береді.

Сапалы құрылған ерекшелік тапсырыс берушіге, жеткізушіге және басқа адамдарға мынадай мүмкіндіктер береді:

1. Бағдарламалық бұйым қандай функция орындауға тиіс екендігі туралы мәселе жөнінде тапсырыс беруші мен жеткізушінің арасындағы келісім үшін негіз құру. SRS келтірілген бағдарламалық қамтамасыз етудің толық сипаттамасы әлеуетті пайдаланушыларға бағдарламалық қамтамасыз ету олардың қажеттіліктеріне жауап береді ме немесе олардың қажеттілігін қанағаттандыру үшін, бағдарламалық қамтамасыз етуді қалай өзгерту қажет екенін айқындауға көмек көрсетеді.

2. Әзірлеу бойынша жұмыс көлемін азайту. SRS дайындау тапсырыс беруші ұйымның әртүрлі қатысушы топтарын жобаны орындауға кіріспес бұрын, барлық талаптарды қатаң қарастыруға мәжбүрлейді және кейін қайталанатын жобалауды, кодтауды және тестілеуді қысқартады. SRS көрсетілген талаптарды толық талдау түзетуге болатын әзірлеу сатысында жол берілген олқылықтарды, дұрыс түсінбеуді және қарама-қайшылықтарды ашуы мүмкін.

3. Шығыстар мен жоспарларды бағалау үшін негізді қамтамасыз ету. SRS сәйкес әзірленген бағдарламаны сипаттау жобаға шығынды бағалау үшін практикалық негіз болып табылады және тапсырыс беруші ұйымымен жобаның бюджетін келісу үшін және жеткізуші бағасын бағалау үшін пайдаланылуы мүмкін.

4. Дұрыстығы мен верификациясын тексеру үшін негіз қамтамасыз ету. Ұйымдар сапалы әзірленген SRS пайдалану кезінде дұрыстығы мен верификациясын тексеру жоспарын тиімді құра алады. Әзірлемеге келісім-шарттың бір бөлігі ретінде SRS сәйкес келуін айқындау үшін негіз қамтамасыз етеді.

5. Беруді жеңілдету. SRS жаңа пайдаланушыларға бағдарламалық өнімды беруді немесе оны жаңа машинаға орнатуды қарапайым етеді. Осылайша тапсырыс берушілер бағдарламалық қамтамасыз етуді олардың ұйымдарының басқа бөліністеріне қарапайым бере алады, ал жеткізушілер үшін оны жаңа тапсырыс берушілерге беру жеңіл болады.

SRS-та жоба емес, ол әзірленген өнім талқыланатындықтан SRS дайын өнімды одан әрі кеңейту үшін негіз болып табылады. SRS кейбір өзгерістерді қажет етуі мүмкін, алайда өнімді үздіксіз бағалау үшін негіз болып табылады.

Дұрыс жасалған ерекшелік (SRS):

- тура;
- бір мәнді;
- толық;
- қайшылықсыз;
- оның мәні және (немесе) тұрақтылығы бойынша реттелген;
- тексерілетін;
- жетілдірілетін;
- қадағаланатын болуы тиіс.

SRS егер онда баяндалған әрбір талап бағдарламалық қамтамасыз ету қанағаттандыратын талап болып табылса *тура* болады. Туралыққа кепіл беретін қандай да бір құрал немесе рәсім жоқ. SRS жүйеге қойылатын талап ерекшелігі сияқты едәуір жоғары деңгейлі кез-келген қолданылатын ерекшелікпен, жобаның басқа құжаттамасымен және оның олармен келісілгенін қамтамасыз ету үшін басқа қолданылатын стандарттармен салыстырылуы тиіс. Балама ретінде тапсырыс беруші немесе пайдаланушы SRS нақты қажеттілікті дұрыс көрсетеме екенін анықтай алады. Қадағалану бұл рәсімді қарапайым және қатеге аз ұшырағыш етеді.

SRS онда баяндалған әрбір талаптың бір ғана түсіндірмесі бар болған жағдайда *бір мәнді* болады. Ол үшін ең аз дегенде соңғы өнімның әрбір қасиеті бір бірегей сөзбен сипатталғаны қажет. Ерекше мәнмәтінде пайдаланылатын терминің көп мәні бар болған жағдайда ол термин глоссарийге қосылуы тиіс, онда оның мәні неғұрлым дұрыс сипатталады. SRS бағдарламалық қамтамасыз етудің өмірлік қайталымына қойылатын талапты құру процесінің маңызды бөлігі болып табылады және жобаны жобалау, іске асыру, мониторинглеу кезінде, дұрыстығын верификациялау және тексеру, сондай-ақ оқыту кезінде пайдаланылады. SRS құратын және пайдаланатын адамдар үшін бір мәнді болуы тиіс.

SRS ол мына элементтерден тұрса ғана *толық* болып табылады:

- барлық бар талаптар, олар функционалдық мүмкіндіктерге, жұмыс сипаттамасына, жобалық шектеулерге, ерекше бөлігіне немесе сыртқы интерфейстарға жатуына қарамастан. Атап айтқанда жүйе ерекшелігі (оның бір бөлігі бағдарлама болады) салатын кез-келген сыртқы талаптар расталған және өнделген болуы тиіс;

- бағдарламалық қамтамасыз етудің барлық ықтимал жағдайларда іске асырылуы мүмкін кіріс деректердің барлық кластарына дыбыс беруін айқындау. Жол берілетін және жол берілмейтін кіріс мәндеріне дауыс беруді айқындау маңызды;
- SRS-тағы барлық суреттерге, кестелерге және схемаларға толық белгілеулер және сәлтемелер және барлық терминдер мен өлшем бірліктерінің анықтамасы.

*Қайшылықсыздық* ішкі қайшылықсыздықты көрсетеді. Егер SRS деңгейі едәуір жоғары, мысалы жүйелі талаптардың ерекшелігі (System Requirements Specification — SyRS) сияқты құжатпен келісілмеген болса ол тура емес болып табылады. SRS тек егер онда сипатталған қандай да бір жеке талаптардың көп болуы қайшлықта болмаса ішкі қайшылықсыз болып табылады.

Егер SRS-тағы әрбір талапта осы нақты талаптың не мәнін не тұрақтылығын көрсететін сәйкестендіргіш бар болса *мәні және (немесе) тұрақтылығы бойынша реттелген* болып табылады. Бұл ретте бағдарлама өніміне қатысы бар барлық талаптар тең мәнінде маңызды болмайды. Кейбңр талаптар ажырамас болуы мүмкін, әсіресе адамның өмірі байланысты қосымшалар үшін, ал басқалар тек қаланатын болуы мүмкін. Әрбір талап, бұл айырмашылықтарды нақты және анық етуі үшін SRS-те сәйкестендірілген болуы тиіс. Талаптарды сәйкестендіру:

а) тапсырыс берушіге әрбір талапты толық қарап шығуға көмектеседі, ол оған негізделген жасырын жол беруішіліктерді түсіндіруге мүмкіндік береді;

б) әзірлеушілерге дұрыс жобалық шешім қабылдауға және бағдарламалық өнімның әр түрлі компонентіне тиісті күш салуға көмектеседі.

SRS онда баяндалған әрбір талап тексерілетін болған кезде ғана *тексерілетін* болып табылады. Пайдаланушы немесе машина қандай да бір соңғы экономикалық тұрғыдан тиімді (пайдалы) процессті пайдалана отырып, бағдарламалық қамтамасыз ету осы талапты қанағаттандыратынына көз жеткізе алса ол тексерілетін болады. Жалпы кез-келген бір мағыналы емес талап тексерілмейді. Тексерілмейтін талаптар «жақсы жұмыс істейді», «жақсы пайдаланушы интерфейсі» және «әдетте орындалады» деген формулировкадан тұрады. Бұл талаптар тексерілмейді, себебі «жақсы», «әдетте» деген терминдерді анықтау мүмкін емес. «Бағдарлама ешқашан тоқтап қалмауға тиіс» деген пікір тексерілмейтін болып табылады, себебі бұл қасиетті тестілеу теориялық тұрғыдан мүмкін емес.

SRS егер оның құрылымы мен стилі кез-келген талаптың өзгеруі құрылымы мен стилін сақтай отырып жеңіл, толық және қайшылықсыз орындалатындай болған кезде *жетілдірілетін* болып табылады. Бұл ретте жетілдіру SRS:

- мазмұны, алфавитті көрсеткосымша және анық көрінетін айқыш сілтемелері бар байланысты және пайдалануға жеңіл құрылымының бар болуын;
- артық болмауын (яғни бір талап бір жерде ғана SRS түрінде пайда болмауы тиіс);
- әрбір талапты басқа талаптармен араластырмай бейнелеуін қажет етеді.

Артық болу қате болып саналмайды, бірақ ол қатенің пайда болуына әкеп соғуы мүмкін. Кейбір жағдайда артықтық SRS оқылатын болуына көмектеседі, бірақ артық құжатты жаңарту кезінде проблемалар туындауы мүмкін. Мысалы, талап ол пайда болатын үш жердің бір жерінде ғана өзгертілуі мүмкін. Онда SRS қайшылықты болады. Артықшылық қажет болған кезде SRS оны жетілдіретіндей ету үшін көрінетін айқыш сілтемелер қосуы тиіс.

SRS егер оның талаптарының әрқайсысы анық көрінсе және ол құжаттаманы одан әрі әзірлеу және жетілдіру кезінде әрбір талапқа сілтеме жасай алатын болса ол *қадағаланатын* болып табылады. Қадағалаудың мынадай екі типі ұсынылады:

1) *кері* (яғни әзірлеудің алдыңғы сатысына). Әрбір талап бұдан бұрынғы құжаттарға сілтеме жасайтынына байланысты;

2) *түзу* (яғни SRS туындайтын барлық құжаттарға). SRS-тан әрбір талаптың анық атауы немесе сілтеме нөмірі бар-жоғына байланысты. SRS тікелей қадағалау бағдарламалық өнім пайдалану және ілесіп жүру сатысында болған кезде әсіресе маңызды. Кодты және сәулет құжаттарын ауыстырған сайын осы өзгерістер әсер етуі мүмкін талаптардың толық жиынын айқындау мүмкіндігі болуы тиіс.

IEEE 830 стандарты ұсынатын SRS құрылымы мынадай.

1. Кіріспе:
   1.1. Мақсаты.
   1.2. Іс-әрекет аясы.
   1.3. Анықтамалар, акронимдар және қысқарған сөздер.

1.4. Жарияланымдар.
1.5. Қысқаша шолу.
2. Толық сипаттамасы:
2.1. Өнім мәнмәтіні.
2.2. Өнім функциясы.
2.3. Пайдаланушы сипаттамасы.
2.4. Шектеулер.
2.5. Рұқсаттар және байланыстар.
3. Ерекше талаптар.
Қосымшалар.
Алфавитті көрсеткіш.

SRS кіріспесі бүкіл SRS қысқаша шолуынана тұруы тиіс. «Толық» сипаттау бөлімі бағдарламалық қамтамасыз етуге және оған қойылатын талаптарға әсер ететін жалпы факторларды сипаттауы тиіс. Бұл бөлім нақты талаптар белгілемейді. Оның орнына SRS-тың 3 бөлімінде толық анықталатын талаптар туралы алдын ала мәліметтермен қамтамасыз етеді және оларды түсіну үшін қарапайым етеді.

«Өнім мәнмәтіні» қосымша бөлімі басқа өніммен байланысты өнімді мәнмәтінінде сипатталуы тиіс. Егер өнім тәуелсіз және толығымен автономды болып табылса, онда құжатта анық көрсетілуі тиіс. Егер SRS үлкен жүйенің компоненті болып табылатын өнімді айқындайтын болса онда осы қосымша бөлім лсы үлкен жүйенің бағдарламалық қамтамасыз етудің функционалдық мүмкіндіктерімен байланысын орнатуы және осы жүйе мен баңдарламалық қамтамасыз ету арасындағы итрефейстерді сәйкестендіруі қажет. Үлкен жүйенің, қосылыстардың және сыртқы интерфейстердің басты құрамдас бөліктерін көрсететін блок-схемаларды пайдалану тиімді болуы мүмкін. Осы қосымша бөлім юағдарламалық қамтамасыз ету әртүрлі шектеулер кезінде қалай жұмыс істейтінін сипаттауы тиіс.

«Өнім функциялары» қосымша бөлімі бағдарламалық қамтамасыз ету орындайтын негізгі функциялардың ақпаратын көрсетуі тиіс. Айқындықты қамтамасыз ету мақсатында:

- функциялар тапсырыс берушіге немесе құжатты бірінші рет оқып тұрған кез келген адамға функциялар тізімі түсінікті болатындай ұйымдастырылған болуы тиіс;
- әртүрлі функцияларды және оның байланыстарын көрсету үшін мәтіндік және графикалық әдістер қолданылуы мүмкін.

«Пайдаланушының сипаттамасы» қосымша бөлімі білім деңгейінен, дағдысынан және арнайы техникалық білімінен тұратын өнімді пайдаланушыны жалпы сипаттай алуы тиіс.

«Шектеулер» қосымша бөлімі әзірлеушінің опцияларын шектейтін кез-келген басқа позицияларды жалпы сипаттауды қамтамасыз етуі тиіс. Олар аппараттық шектеулерді, басқа қосымшалармен интерфейстерді, параллель операциялардың бақылау функцияларын, басқару функцияларын және т.б. көздейді.

«Рұқсаттар және байланыстар» қосымша бөлімінде SRS қалыптасқан талаптарға әсер ететін барлық факторлар берілуі тиіс. Бұл факторлар бағдарламалық қамтамасыз етудің сәулеттік шектеулері болып табылмайды, ол SRS талабына әсер етуі мүмкін кез-келген олардың өзгерісі.

«Талаптарды бөлу» қосымша бөлімі жүйенің болашақ нұсқалары пайда болғанша кейінге қалтырылатын талаптарды сәйкестендіруі тиіс.

«Ерекше талаптар» бөлімі жобалаушыларға осы талаптарды қанағаттандыратын жүйені әзірлеу мүмкіндігін беру үшін, ал сынақшыларға жүйе осы талаптарды қанағаттанырдатынына көз жеткізу үшін жеткілікті нақтылай деңгейінде бағдарламалық қамтамасыз етуге қойылатын барлық талаптардан тұруы тиіс. Бұл бөлімде әрбір қалыптасқан талапты пайдаланушы, операторлар немесе басқа сыртқы жүйелер қабылдай алуы тиіс. Бұл талаптар жүйеге әрбір кіріс ықпалды (стимулды), жүйе мен барлық функциялардың шығу ықпалының (дыбысталуын) сипаттамасынан тұруы тиіс. Ерекше талаптар барлық сипаттамаларға сәйкес қалыптастырылуы тиіс. Барлық талаптар бірдей сәйкестендірілетін болуы тиіс.

Бұл бөлімде сыртқы интерфейстер сипатталкы тиіс — бағдарламалық қамтамасыз етудің барлық кіріс және шығыс деректерін толық сипаттау. Функционалдық талаптар негізгі операцияларды айқындауы қажет, оларды бағдарламалық қамтамасыз ету кіріс деректерді қабылдау және өндеу кезінде және шығыс деректерді өндеу және генерация кезінде орындауы тиіс. Сондай-ақ бағдарламалық қамтамасыз етуге немесе пайдаланушының бағдарламалық қамтамасыз етумен өзара іс-қимыл жасасуына қойылатын статистикалық және динамикалық сандық сипаттамаларға қойылатын талаптар сипатталауы тиіс. Жеке қосымша бөлімде деректер базасында орналасуы тиіс кез-келген ақпарат логикасына қойылатын талаптарды айқындау қажет. Сондай-ақ барлық жүйе үшін берілген қол жетімділік деңгейін қамтамасыз ету үшін қажетті, бақылау нүктесі, қайта қалпына келтіру немесе қайта іске қосу сияқты фактор, сондай-ақ бағдарламалық қамтамасыз етуді кездейсоқ немесе қасақана енуден, пайдаланудан, өзгертуден, бұзудан немесе ашудан қорғайтын факторлар белгіленуі тиіс.

Мобильдікке қойылатын талаптарлы сипаттау қажет — бағдарламалық қамтамасыз етудің атрибуттары, олар бағдарламалық қамтамасыз етуді басқа машиналарға және (немесе) операциондық жүйелергеа ауыстыру оңайлығына жатады.

Қосымшалар нақты SRS бір бөлігі ретінде үнемі қаралмайды және үнемі қажет емес. Олар мынадан тұруы мүмкін:

- кіру-шығудың үлгілік форматтары, өзіндік құн калькуляциясы немесе пайдаланушылардан сұрау нәтижелері;
- SRS оқырмандарына көмектесе алатын қосымша немесе алдын ала ақпарат;
- бағдарламалық қамтамасыз ету шешуге тиіс проблеманы сипаттау;
- қорғау, экспорт, бастапқы жүктеу немесе басқа талаптарға сәйкес келуді қамтамасыз ететін кодтар мен тасымалдаушылар үшін арнайы әмірлер.

Қосымшаны SRS-та қосу кезінде осы қосымшалар талаптардың бір бөлігі болып саналу керек пе екенін нақты қалыптастыру қажет.

## 4.5. БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУДІ ҚҰЖАТТАНДЫРУДЫ БАСҚАРУ

*Р ИСО/ХЭК 9294 МЕМСТ «Ақпараттық технология. Бағдарламалық қамтамасыз етуді құжаттандыруды басқару бойынша басшылық»* бағдарламалық қамтамасыз етудің немесе бағдарламалық өнімнің өндірісі үшін жауап беретін басшылар үшін бағдарламалық қамтамасыз етуді құжаттандыру бойынша басшылықты білдіреді. Осы стандарт басшылар бағдарламалық қамтамасыз етуді құжаттандыруды тиімді басқару үшін айналысуға тиіс стратегияларды, стандарттарды, рәсімдерді, ресурстар мен жоспарларды анықтауға бағытталған.

Бағдарламалық қамтамасыз етудің барлық типіне қолдануға арналған басшылық — қарапайым бағдарламалардан едәуір күрделі бағдарламалық жиынтыққа немесе бағдарламалық қамтамасыз ету жүйесіне дейін. Бағдарламалық қамтамасыз етудің тіршілік циклінің барлық сатысына қатысы бар бағдарламалық құжаттаманың барлық типі қамтылған.

Бағдарламалық қамтамасыз етуді басқару қағидасы кез-келген жоба көлеміне бірдей. Шағын жобалар үшін осы стандартта келтірілген ережелердің басым бөлігін қолданбауға болады, бірақ қағидалар сол қалпы қалады.

Басшылар осы ұсынымдарды өздерінің нақты қажеттіліктеріне бағыттауы мүмкін.

Бағдарламалық құралдарды қолданудың көбеюші ауқымынан және олардың күрделілігі осы құралдарға пайдаланушыларға қол жетімді толық, нақты және түсінікті құжаттаманың болу қажеттілігі туындайды. Құжаттама көбінесе нақты бағдарламалық қамтамасыз етуді құрғаннан кейін әзірленетін зат сияқты қаралады. Алайда бағдарламалық құжаттаманы әзірлеу сапасының тұрғысынан ол осы бағдарламалық қамтамасыз етуді құру процесінің ажырамас бөлігі ретінде қаралуы тиіс. Осы проблемаға тиісті қараған кезде, құжаттаманы жасау және басқару процесінде едәуір күрделі жұмыс қажет.

## 4.6. ПАЙДАЛАНУШЫНЫҢ ҚҰЖАТТАМАСЫН ЖАСАУ ПРОЦЕСІ

Пайдаланушы интерфейсі бар бағдарламалық құрал үшін пайдаланушы құжаттамасының барлық түрін жасаудың минималды қажетті процесі *Р ИСО/ ХЭК 15910 МЕМСТ «Бағдарламалық құралдың пайдаланушысының құжаттамасын жасау процесі»* стандартымен анықталады. Пайдаланушы құжаттамасына баспа құжаттама (мысалы, пайдаланушының басшылығы және қысқаша анықтамалық карталар), диалогты (жедел) құжаттама, анықтамалық мәтін және диалогты құжаттаманың жүйесі, сондай-ақ жеткізуді қамтамасыз етудің анықтамалық жүйесі, анықтамалық ақпарат және т.б. жатқызылады.

Егер бағдарламалық құралды әзірлеу сапаны басқару стандартына сәйкес құжаттандырылса, осы стандарттың ережелерін әзірлеменің өзіне, солай тиісті құжаттамаға тең қолданады.

Тапсырыс беруші құжаттама әзірлеушісіне:

- барлық тиісті ерекшеліктерге, жазба форматтарына, экрандар мен есептерді құрастыруға, бағдарламаландыруды автоматтандыру құралдары жұмысының шығыс нәтижелеріне және құжаттаманы дайындау үшін қажет басқа да ақпаратқа;
- бағдарламалық құралдың жұмыс көшірмесіне (қажет болған жағдайда);
- құжаттама әзірлеушілерінің персоналында туындайтын мәселелерді уақтылы дұрыс шешуді қоса алып, талдаушылар мен бағдарламалаушыларға;

■ аудиторияны талдау және қолайлылыққе тестілеу үшін тұрпатты пайдаланушыларға (мүмкіндігі бойынша) қолжетімділікті қамтамасыз етуі тиіс.

Құжаттама әзірлеушісінің міндетіне кем дегенде шығарылатын өнім мазмұнын түсінуге және оған сәйкес келетін аудиторияға кепіл беретін тапсырыс берушінің бағдарламалық құралдарыдың әзірлеушілерімен байланыстарының жемістілігін қамтамасыз ету кіреді.

Құжаттаманың әзірлеушісі бір уақытта бағдарламалық құралдың әзірлеуші болып табылуынан тәуелсіз, тапсырыс беруші оны барлық қолданылатын стандарттардың, стиль және формат бойынша нұсқаулықтың, сондай-ақ тиісті материалдардың (егер олар жалпыға қол жетімді болып табылмаса) көшірмелерімен қамтамасыз етуі тиіс.

Құжаттаманың әзірлеушісі құжаттандыру жоспарын құруы тиіс, онда нақты құжаттаманы құру кезінде орындалатын міндеттер анықталуы тиіс. Осы жоспар тапсырыс берушімен ресми келісілуі тиіс, ол бұл жоспарда тапсырыс берушінің барлық талаптарының толық есепке алынуын растайды.

## 4.7. БАҒДАРЛАМАЛЫҚ ӨНІМДІ БАҒАЛАУ

**Р ИСО/ХЭК 9126 МЕМСТ стандартының мақсаты.** *Р ИСО/ХЭК 9126 МЕМСТ* стандарты *«Бағдарламалық өнімді бағалау. Сапаның және оларды қолдану жөніндегі басшылықтың сипаттамасы»* минималды көшіру арқылы бағдарламалық қамтамасыз етудің сапасын сипаттайтын алты сипаттаманы анықтайды. Осы сипаттамалар бағдарламалық қамтамасыз етуді одан әрі нақтылау және сипаттау үшін негіз құрады. Басшылық бағдарламалық қамтамасыз етудің сапасын бағалау үшін сапа сипаттамаларын пайдалануды сипаттайды.

Стандарт өзгерту, саралау және бағалау көрсеткіштері мен әдістерін анықтамайды.

Сипаттамалардың анықтамалары және бұл стандартта келтірілген сапаны бағалау процесінің тиісті моделі бағдарламалық өнім үшін талаптар анықталған және тіршілік циклі кезінде оның сапасы бағаланған кезде қолданылады. Бұл сипаттамалар ЭЕМ бағдарламаларын және бағдарламалық-техникалық құралдарға (кіріктірілген бағдарламаларға) кіретін деректерді қоса алғанда бағдарламалық қамтамасыз етудің кез-келген түріне қолданылуы мүмкін.

Стандарт бағдарламалық қамтамасыз етуді сатып алумен, әзірлеумен, пайдаланумен, қолдаумен, сүйемелдеумен немесе тексерумен байланысты сипатталамарға арналған.

**Р ИСО/ХЭК 9126 МЕМСТ сәйкес бағдарламалық қамтамасыз ету сапасын бағалау.** Р ИСО/ХЭК 9126 МЕМСТ стандартына сәйкес бағдарламалық қамтамасыз ету сапасы мынадай сипаттамалар бойынша бағалануы мүмкін.

*Функционалдық мүмкіндіктері* — функция жинағына және оның дұрыс қасиеттеріне жататын атрибуттардың жиыны. Орнатылған немесе болжанатын қажеттіліктерді іске асыратындан функциялар болып табылады. Осы атрибуттар жиынтығы бағдарламалық қамтамасыз ету қажеттілікті қанағаттандыру үшін орындайтынын сипаттайды, ал басқа жиынтықтар ең бастысы қашан және қалай оны орындау керек екенін сипаттайды.

*Сенімділік* — бағдарламалық қамтамасыз етудің белгіленген уақыт аралығында белгіленген жағдайда қызмет көрсету сапасын сақтап қалу қабілетіне жататын атрибуттардың жиыны. Бағдарламалық қамтамасыз ету тозбайды немесе ескірмейді. Сенімділікті шектеу талаптардағы, жобадағы және іске асырудағы ақауларға байланысты байқалады. Осы ақаулардан бас тарту бағдарламалық қамтамасыз етуді пайдалану тәсіліна және бұрын тандалған бағдарлама нұсқасына байланысты.

*Қолайлылық* — пайдалану және мұндай пайдалануды белгіленген немесе пайдаланушылар ортасы ұсынатын жеке бағалау үшін қажетті жұмыс көлеміне жататын атрибуттардың жиыны. Қолайлылық пайдалануға дайындықты және нәтижелерді бағалауды қоса алғанда бағдарламалық қамтамасыз етуге әсер ететін пайдаланушының пайдалану жағдайының барлық түрінде қаралуы тиіс. Қолайлылық осы стандартта бағдарламалық өнімнің атрибуттар жиынтығы ретінде анықталған, эргономика көзқарасы жағынан анықтамадан айырмашылығы бар, онда тиімділік және тиімділік емес сияқты басқа сипаттамалар қолайлылықтың компоненті ретінде қаралады.

*Тиімділік* — белгіленген жағдайда бағдарламалық қамтамасыз ету сапасының деңгейі мен пайдаланылатын ресурстар көлемінің арасындағыарасалмаққа жататын атрибуттардың жиыны.

Ресурстарға басқа да бағдарламалық өнімдер, техникалық құралдар, материалдар (мысалы басып шығаруға арналған қағаз, иілгіш дисктер) және пайдаланылатын, сүйемелденетін немесе қызмет көрсетілетін персоналдың қызметтері кіруі мүмкін.

*Сүйемелдеу* — нақты өзгерістер (түрлендіру) жүргізу үшін талап етілетін жұмыстардың көлеміне жататын атрибуттардың жиыны. Өзгеріс бағдарламалық қамтамасыз етудің қоршаған ортадағы, қызмет ету талаптары мен жағдайларындағы өзгерістерге түзетулер, жетілдірулер немесе бейімделулер кіруі мүмкін.

*Мобильдік* — бағдарламалық қамтамасыз етудің бір ортадан басқа ортаға ауыстырылатын болу қабылетіне жататын атрибуттардың жиыны. Қоршаған орта ұйымдастырушылық, техникалық және бағдарламалық ортадан тұруы мүмкін.

**Р ИСО/ХЭК 9126 МЕМСТ сәйкес бағдарламалық қамтамасыз ету сапасының сипаттамаларын қолдану.** Р ИСО/ХЭК 9126 МЕМСТ стандарты бағдарламалық қамтамасыз етудің сапасына қойылатын талаптарды белгілеу және бағдарламалық өнімдерді бағалау (өлшеу, саралау және бағалау) үшін қолданылады, оның ішінде:

- бағдарламалық өнімнің сапасына қойылатын талаптарды анықтау;
- сапа талаптары әзірлеу процесінде қанағаттандырылуына бақылау жасау кезінде бағдарламалық қамтамасыз етуге қойылатын техникалық талаптарды бағалау;
- енгізілген бағдарламалық қамтамасыз етудің белгілері мен қасиеттерін (атрибуттарын) сипаттау (мысалы пайдаланушылардың басшылықтарында);
- әзірленген бағдарламалық қамтамасыз етуді оны жеткізер алдында бағалау;
- бағдарламалық қамтамасыз ету қабылдар алдында бағалау.

Осы стандартта сипатталған сипаттамалар үшін бірнеше жалпы қабылданған метрикалар бар. Стандарттау жөніндегі ұйымдар мен топтар бағалау процесінің өз меншік үлгілерін белгілеуі мүмкін. Сондай-ақ, бағдарламалық қамтамасыз етуді әртүрлі қолдану салаларында және тіршілік циклінің сатыларында қамту үшін, осы сипаттамалармен байланысты метрикаларды қалыптастыру және тексерудің өз әдістерін қолдануы мүмкін. Тиісті метрикалар болмаған және әзірленуі мүмкін болмаған кездегі жағдайларда, кейбір кезде сөзбен сипаттауды немесе «болжалды әдістер» пайдаланады.

Аталған сапа сипаттамаларын пайдалану кезінде, соннымен қатар, нақты осы ұйым үшін немесе осы қолдану үшін немесе сол немесе басқа үшін критерийлер белгілеу қажет.

Бағалау нәтижелерімен алмасу кезінде, сапаны бағалауға қолданылатын метрикалар мен критерийлер белгіленуі тиіс.

Бағдарламалық қамтамасыз етуді сыныптаудың жалпыға бірдей жүйесі болмағанымен, бағдарламалық қамтамасыз етудің бірнеше жалпыға бірдей кластары бар. Сапаның әр сипаттамасының маңыздылығы бағдарламалық қамтамасыз етудің класына байланысты өзгереді. Мысалы, сенімділік сыни жүйелердің бағдарламалық қамтамасыз етуіне өте маңызды, ал тиімділік – нақты уақыт жүйелерінің бағдарламалық қамтамасыз ету үшін , ал қолайлылық - соңғы пайдаланушы диалогының бағдарламалық

қамтамасыз ету үшін.

Сондай-ақ, сапаның әр сипатаммасының маңыздылығы қабылданған көзқарасқа байланысты да өзгереді.

Сапа туралы бірнеше көзқарас бар:

- пайдаланушының көзқарасы;
- әзірлеушінің көзқарасы;
- басшының көзқарасы.

Пайдаланушының көзқарасы бойынша сапа — бұл объектінің белгіленген және болжамдалатын қажеттіліктерді қанағаттандыру қасиетіне жататын сипаттамаларының жиынтығы.

Құру процесі пайдаланушыдан және әзірлеушіден бағдарламалық қамтамасыз етудің сол бір сипаттамасын пайдалануды талап етеді, себебі олар талаптар мен қабылдауды белгілеу үшін қолданылады. Бағдарламалық қамтамасыз ету сатуға белгіленген кезде, сапаға қойылатын талаптарда болжамдалатын қажеттіліктер көрсетілуі тиіс. Әзірлеушілер сапаның талаптарын қанағаттандыруы тиіс бағдарламалық қамтамасыз етуді құру үшін жауап беретіндіктен, оларға аралық өнім сапасы да және соңғы өнім сапасы да қызығушылық тудырады. Әзірлеу циклінің әрбір фазасында аралық өнімнің сапасын бағалау үшін, әзірлеушілер сол бір сипаттамалар үшін әртүрлі метрика пайдалануға тиіс, неге десең сол бір метрикалар тіршілік циклінің барлық фазасында қолданбалы емес. Мысалы, пайдаланушы реакиця уақыты терминінде тиімділікті түсінеді, ал әзірлеуші жобалық ерекшелікте бағыт ұзақтығы және күту уақыты мен қолжеткізу терминдерін пайдаланады. Өнімнің сыртқы интерфейсіне қолданылатын метрикалар оның құрылымы үшін қолданылатын метрикамен алмастырылады.

Пайдаланушының түсінігі, сонымен қатар бағдарламалық қамтамасыз етуді сүйемелдейтіндерден талап етілетін сапаның сипаттамасы туралы түсінігі кіруі тиіс.

Басшының сапаның нақты сипаттамасына қарағанда, жалпы сапаға қызығушылығы болуы мүмкін және бұл себеп бойынша дара сипаттамаларға арналған коммерциялық талаптарды көрсететін мәндерінің маңыздлығын белгілеуде мұқтаж болады. Сонымен қатар, басшыға сапаны арттыруды жоспарлы кідіріс немесе құнының артық шығыны сияқты, басқарушылық критерийлерімен салыстыруы қажет болады, неге десең ол шектелген құнының, еңбек ресурстары мен белгіленген уақытының шегінде сапаны оңтайландыруды қалайды.

**Бағалау процесінің моделі.** Р ИСО/ ХЭК 9126 МЕМСТ стандартына келісімді бағдарламалық қамтамасыз ету сапасын бағалау процесі үш сатыдан тұрады:

1) сапаға қойылатын талаптарды белгілеу (айқындау);
2) бағалауға дайындық;
3) бағалау рәсімі.

Осы процесс бағдарламалық өнімнің әр компоненті үшін тіршілік циклінің кез-келген сәйкес келетін фазасында қолданылуы мүмкін.

Бастапқы сатының мақсаты сапа сипаттамалары терминдерінде және ықтимал кешендік көрсеткіштерде қойылатын талаптарды белгілеу болып табылады. Талаптар қарастырылатын бағдарламалық өнім үшін сыртқы ортаның қажеттілігін білдіреді және әзірлеме басталғанға дейін айқындалуы тиіс. Бағдарламалық өнім негізгі компоненттерге бөлінетіндіктен, өнімге қойылатын талаптардың жалпы жеке компоненттерге қойылатын талаптардан айырмашылығы болуы мүмкін.

Екінші сатының мақсаты бағалау үшін негіздерді дайындау болып табылады. Бұл саты метриканы таңдаудан, саралау деңгейін белгілеуден және бағалау критерийлерін айқындаудан тұрады.

*Сапа метрикасын (көрсеткіштерін) таңдау*. Сапа сипаттамасы анықталған амал оларды тікелей өлшеуге жол бермейді. Бағдарламалық өнім сипаттамасына сәйкес келетін метрикаларды (көрсеткіштерді) белгілеу қажеттілігі болады. Әрбір сандық белгі және бағдарламалық қамтамасыз етудің сипаттамамен сәйкес келетін оның ортасымен әрбір сандық бағаланатын өзара іс-қимыл жасасуы метрика (көрсеткіш) ретінде қабылдануы мүмкін.

Метрикалар әзірлеу процесінің онда олар пайдаланылатын ортасы мен фазасынан түрлі тәуелді болуы мүмкін. Әзірлеу процесінде пайдаланылатын метрикалар пайдаланушының тиісті метрикаларына сәйкес болуы тиіс, себебі метрикалар пайдаланушының түсінігі бойынша шешілетін болып табылады.

*Саралау деңгейін анықтау.* Сандық белгілер сапа метрикасын пайдаланып өлшенуі мүмкін. Нәтижесі, яғни өлшенген мәні талаптардың қанағаттандыру деңгейін көрсетпейді. Осы мақсатта берілген шкалалар талаптарды қанағаттандырудың әртүрлі деңгейіне сәйкес келетін диапазондарға бөлінуі тиіс. Сапаның нақты қажеттілікке жатуына байланысты, саралаудың жалпы деңгейі мүмкін емес. Олар әрбір нақты бағалау үшін анықталуы тиіс.

*Бағалау критерийін анықтау*. Өнім сапасын анықтау үшін әртүрлі сипаттамаларды бағалаудың нәтижелері алынуы тиіс. Бағалайтын адам ол үшін рәсім дайындауы керек, мысалы шешім кестесін немесе орташа сараланған шешім кестесін пайдалану.

Рәсімге нақты пайдалану жағдайларында бағдарламалық өнімнің сапасын бағалауға ықпал ететін уақыт пен құны сияқты, басқа да аспектілер кіреді.

Бағалау процесі (бағалау рәсімі) моделінің соңғы сатысы «өзгерту», «саралау» және «бағалау» деп аталатын үш кезең бойынша нақтыланады.

*Өлшеу* үшін таңдалған метрикалар бағдарламалық өнімге қолданылады. Нәтижесі метрика масштабындағы мәні болып табылады.

*Саралау* сатысында өлшенген мән үшін саралау деңгейі белгіленеді.

*Бағалау* бағдарламалық қамтамасыз етуді бағалау процесінің соңғы сатысы болып табылады, онда көптеген белгіленген деңгейлер жинақталады. Бағдарламылық өнімнің сапасы туралы қорытынды нәтиже болып табылады. Одан әрі жинақталған сапа уақыт және құны сияқты басқа факторлармен салыстырылады. Басшылықтың соңғы шешімі басқару критерийлері негізінде қабылданады. Бағдарламалық өнімді қабылдау немесе бракка шығарып тастау немесе шығару немесе шығармау бойынша басшылық шешімі нәтижесі болып табылады.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. ББҚЖ стандарттарының топтарын атаңыз.
2. ББҚЖ нормативтік-техникалық құжаттарының құрамы қандай?
3. Бағдарламалық құжаттаманың түрлерін атаңыз.
4. Пайдаланушылық құжаттаманың түрлерін атаңыз.
5. ББҚЖ негізгі кемшіліктері қандай?
6. Р ИСО/ХЭК 12207 МЕМСТ стандартының мақсаты неде?
7. Р ИСО/ХЭК 12207 МЕМСТ стандарты тіршілік циклінің қандай мқделіне бағдарланған?
8. Р ИСО/ХЭК 12207 МЕМСТ стандарты қай бағдарламалау тіліне бағдарланған?
9. «Техникалық тапсырма» құжаты не үшін жасалды?
10. МЕМСТ 19.106-78 «Техникалық тапсырма» құжаты қандай бөлімдерден тұрады?
11. МЕМСТ 19.106-78 келісімді, «Техникалық тапсырма» құжатындағы «Бағдарламаға немесе бағдарламалық бұйымға қойылатын талаптар» бөлімі қандай қосымша бөлімдерден тұрады?
12. МЕМСТ 19.106-78 келісімді, техникалық тапсырмадағы «Ақпараттық және бағдарламалық үйлесімділікке қойылатын талаптар» бөлімінде не көрсетіледі?

13. МЕМСТ 19.106—78 келісімді техникалық тапсырмадағы «Техникалық-экономикалқ көрсеткіштер» бөлімінде не бар?
14. Бағдарламалық қамтамасыз етуге қойылатын талаптардың ерекшелігі деген не?
15. «Бағдарламалық қамтамасыз етуге қойылатын талаптардың ерекшеліктерін жасау әдістемесі» IEEE 830 стандарты нені сипаттайды?
16. Сапалы жасалған ерекшелік тапсырыс берушіге, жеткізушіге және басқа адамдарға қандай мүмкіндік береді?
17. Жасалған ерекшеліктің дұрыстығы нені білдіреді?
18. Қандай жағдайда (SRS) ерекшелік толық болып табылады?
19. Қандай жағдайда (SRS) ерекшелік жетілдірілетін болып табылады?
20. IEEE 830 стандартымен ұсынылған SRS құрылымына не кіреді? Оның бөлімдерінің мазмұны қандай?
21. Р ИСО/ХЭК 15910 МЕМСТ стандарты нені сипаттайды?
22. Р ИСО/ХЭК 9126 МЕМСТ стандартының мақсаты қандай?
23. Бағдарламалық қамтамасыз етудің сапасы қандай сипаттамалармен бағалануы мүмкін?
24. Р ИСО/ХЭК 9126 МЕМСТ стандарты қалай қолданылады?
25. Пайдаланушының көз қарасынан бағдарламалық қамтамасыз етудің сапасы нені білдіреді?
26. Бағдарламалық қамтамасыз ету сапасын бағалау процесі қандай сатылардан тұрады? Әр сатының мақсаты қандай?

# WEB-БАҒДАРЛАМАЛАУ НЕГІЗДЕРІ

# II

## БӨЛІМ

5 тарау

# ИНТЕРНЕТТІҢ ҚҰРЫЛЫМЫ ЖӘНЕ ОНЫҢ НЕГІЗГІ ҚАҒИДАЛАРЫ

## 5.1. ЖАЛПЫ ТҮСІНІКТЕР МЕН АНЫҚТАМАЛАР

Интернет (Internet, «желілердің желісі») — бұл бірыңғай басқару орталығы жоқ, бірақ бірыңғай қағида бойынша жұмыс істейтін және өз пайдаланушыларына қызметтердің бірыңғай жиынтығын ұсынатын жаһандық ақпараттық желі. Осы желінің бір бөлігі бірыңғай мекенжайы кеңістігі арқылы бір бірімен логикалық тұрғыдан өзара байланысты. Интернет көптеген өзара байланысты компьютер желісінен тұрады, әрқайсысын тәуелсіз оператор — 0Интернет қызметін жеткізуші (Internet Service Provider — ISP) басқарады деп айтуға болады. Интернет сервистері — ол Желіде пайдаланушыларға, бағдарламаларға, жүйелерге, қызмет блоктарына және т.б. ұсынылатын сервистер. Интернетке сервистер желілік қызметтер түрінде іске асырылған, оларға қол жетімділік жергілікті және жаһандық желіден іске асырылады.

World Wide Web (WWW, Дүниежүзілік тор немесе қарапайым Web-желі) сервисі — желіге қосылған кез-келген сервердегі ақпаратқа қол жетімділік алуға мүмкіндік беретін Интернеттегі негізгі қызмет. WWW — бұл Интернеттің физикалық инфрақұрылымына және HTTP деректер беру хаттамасына негізделген жаһандық ақпараттық кеңістік. Көбінесе Интернет туралы сөйлеген кезде Web-желіні көздейді.

Интернет орталықсыздандырылған желі болып табылады, оның өз артықшылықтары мен кемшіліктері бар. Негізгі артықшылығы деп ол желінің кез-келген ресурсына жүгіну мүмкіндігін, сондай-ақ ISP арасында келісім жасасу жолымен Интернетті өсіру жеңілдігін айтуға болады.

Кемшіліктерге мыналарды жатқызуға болады:

а) Интернет технологиялары мен қызметтерін жаңғырту күрделілігі, себебі барлық қызмет жеткізушілердің келісілген күштері қажет;

б) бұл желінің жеке сегменттерінің жұмыс істеу қабілеттілігі үшін жауапкершілік Интернет қызметтерінің жеткізушілеріне жүктеледі, ол жұмысқа кейбір тұрлаусыздық енгізеді.

Барлық бұрын жазылғандардан Интернет өте күрделі желі болып таблады және желі тораптарының арасындағы өзара іс-қимылды ұйымдастыру міндеті де күрделі болып табылады деп есептеледі.

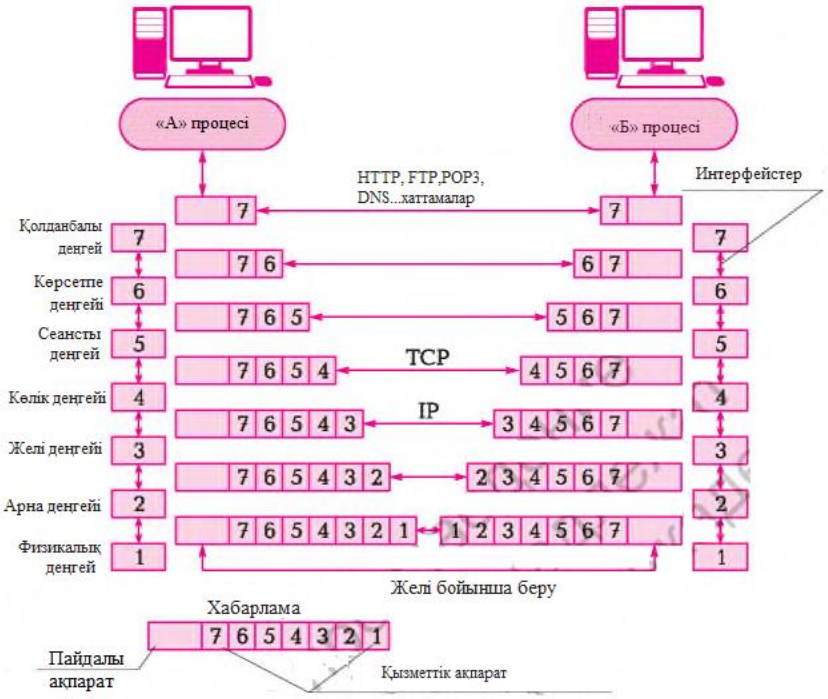
*Ашық жүйе* — бұл жалпыға қол жетімді және жалпыға бірдей қабылданған халықаралық стандарттарға сәйкес әзірленген аппараттық және бағдарламалық өнімдер мен технологиялардан тұратын қандай да бір есептеу ортасы. Компьютер, есептеу желісі, операциялық жоспар, бағдарлама пакеті, басқа аппараттық және бағдарламалық өнімдер ашық жүйе болып табылады.

Ашық ерекшеліктерге сәйкес құрылған жүйе ашық болып табылады. Ашық ерекшеліктер дегеніміз стандарттарға сәйкес келетін және барлық мүдделі тараптар жан-жақты талқылағаннан кейін, келісімге келу нәтижесінде қабылданып жарияланған, барлығына қол жетімді ерекшеліктер. Интернет ашық жүйелердің қағидаттарына толық сәйкестікте құрылған. Интернеттің ішкі құрылғысын сипаттайтын стандарттар жұмыс қосымшалары деп аталатын құжаттарда жарияланады (RFK — Requests for Comments).

Компьютерлік желіде стандарттау негізі желілік өзара іс-қимыл құралдарын әзірлеуге көп деңгейлі жолы болып табылады. Дәл, осы жолдың базасында өзінше желілік мамандардың әмбебап тілі болған, ашық жүйелердің өзара іс-қимылының стандартты моделі құрылған. Желілік модель желіде орындалатын барлық рәсімдерді стандарттау, оларды өзара іс-қимыл жасасатын деңгейлер мен қосымша деңгейлерге бөлу мүмкіндігін қамтамасыз етеді. Олар өзара іс-қимыл жасасудың барлық сатысында деректерді беру және қабылдау сияқты келісім орнатады. Бұл ретте желілік модульдер бір желі ішінде абоненттерге де әртүрлі деңгейде әртүрлі желілерде ақпарат алмасуды дұрыс ұйымдастыруға мүмкіндік береді.

*Хаттамалар* — бұл желінің тораптарындағы бір деңгейді білдіретін, бірақ түрлі торабында орналасқан желілік компоненттермен алмасатын хабарламалардың тәртібі мен форматын анықтайтын нысандандырылған қағидалар. Басқа сөзбен айтқанда, хаттамалар — бұл желіде байланысты қамтамасыз ету тәртібін реттейтін рәсімдер мен қағидалардың жиыны.

Сонымен қатар, көрші деңгейлердің хаттамаларын іске асыратын және бір торапта орналасқан модульдер хабарламалардың стандартталған форматының көмегімен нақты белгіленген қағидаларға сәйкес бір-бірімен өзара іс-қимыл жасасады. Бұл қағидалар интерфейс деп атау қолданылады.




5.1-сурет. Ашық жүйелердің өзара іс-қимыл жасасу үлгісі

*Интерфейс* бір тарапта көрші деңгейлердегі желілік компоненттер алмасатын хабарламалар реттілігі мен форматын анықтайды. Интерфейс осы деңгеймен көрші деңгейге ұсынылатын қызметтердің жиынын анықтайды.

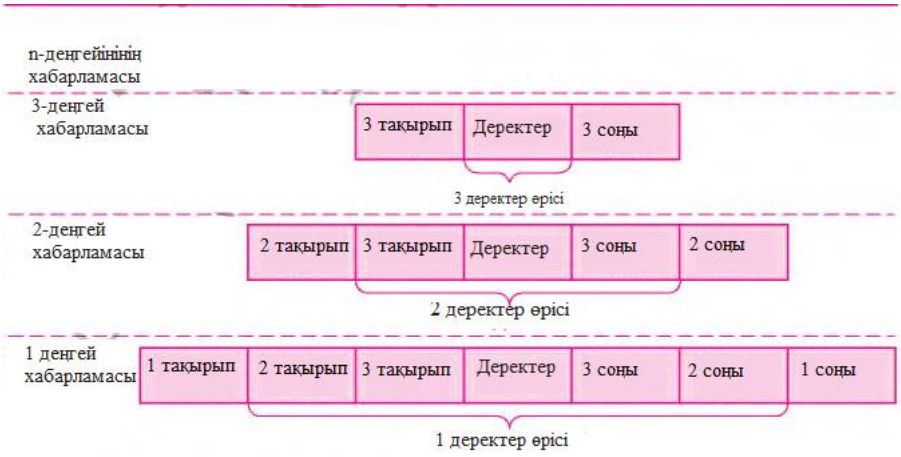
Бір қарағанда хаттама мен интерфейс бір түсінікті білдіреді, алайда олардың міндеттері әр түрлі: хаттамалар бір деңгейлі модульдердің әртүрлі тораптарда өзара іс-қимыл жасасу қағидаларын, ал интерфейстер бір тораптағы көрші деңгей модульдерінің өзара іс-қимыл жасасу қағидаларын анықтайды (5.1-сурет).

Қосымша қолданбалы деңгейге сұрау салған кезде, мысалы файл сервисіне, қолданбалы деңгейдің бағдарламалық қамтамасыз етуі осы сұрау салу негізінде стандартты форматты хабарлама қалыптастырады, оған қызметтік ақпарат (тақырыбы) және берілетін деректерді орналастырады. Осы хабарламалар өрісі бос болуы немесе қандай да бір деректерден тұруы мүмкін, мысалы қашықтықтағы файлға жазу қажет деректер. Бұл ақпаратты мақсаты бойынша жеткізу үшін тағы көптеген міндеттерді шешу қажет.

Қолданбалы деңгей тақырыбынан алынған ақпарат негізінде ұсыну деңгейінінің хаттамасы талап етілетін іс-әрекеттер жасайды және хабарламаға өзінің қызметтік ақпаратын қосады – ұсыну деңгейінің тақырыбы, онда мекенжай машинасының ұсыну деңгейінің хаттамасы үшін нұсқаулар болады. Нәтижесінде алынған хабарлама төмен қарай сеанс деңгейіне беріледі, ол өз кезегінде өз тақырыбын қосады және т.б. кейбір хаттамалар қызметтік ақпаратты тақырып түрінде тек хабарлама басына ғана емес, соңғы жазба түрінде немесе ол кейде «терминатор» деп аталады және соңына орналастырады. Хабарлама төменгі физикалық деңгейге жетеді, ол байланс желісі бойынша алушы компьютерге береді. Бұл сәтте хабарлама барлық деңгей тақырыптарымен «толып» кетеді. Хабарлама желі бойынша басқа машинаға келіп түскен кезде ол жоғары қарай ретпен деңгейден деңгейге қарай орын ауыстырады. Әрбір деңгей өз деңгей тақырыбын талдайды, өндейді және жояды, осы деңгейге сәйкес функция орындайды да жоғарыда тұрған деңгейге хабарламаны береді (5.2-сурет).

Хаттамалардың сатылы ұйымдастырылған жиынтығы желіде тораптардың өзара іс-қимыл жасасуын ұйымдастыру үшін жеткілікті, *коммуникация хаттамаларының құйлымы* деп аталады. Желі — бұл әр текті жабдықтың қосылысы, қоса атқарушылық проблемасы өзекті болып табылады, ол өз кезегінде жалпыға бірдей стандарттарды шығарушылармен келісуді қажет етеді.

Интернет хаттамасының негізгі құйылымы — TCP/IP құйылымы. Бұл хаттамалар басынан бастап жаһандық желіге бағытталған, онда қосылатын арналардың сапасы мінсіз емес.




5.2-сурет. Түрлі деңгейдегі хабарламалардың құрылымы

Ол жаһандық желі құру мүмкіндігін жасайды, ондағы компьютерлер бір біріне әртүрлі амалдармен қосылған — жылдамдығы жоғары оптоталшықты кабель мен серік арналарынан коммутацияланатын телефон желісіне дейін. TCP/IP хаттамалары құйылымының атауы екі түрлі хаттама атауынан тұрады. IP (Internet Protocol) хаттамасы төменгі (желілік) деңгей хаттамасы түрінде болады және желіде деректер пакетін беру үшін жауап береді. IP хаттамасымен жоғары (көлік) деңгейлі деректер беру хаттамасы жұмыс істейді — TCP (Transmission Control Protocol). IP хаттамасы арқылы пакеттерді жібере және қабылдай отырып, TCP хаттамасы барлық деректердің берілген пакеттерін дұрыс реттілікпен жеткізуге кепіл береді.

Интернет хаттамалары сатысындағы ең жоғарғы деңгейді мынадай қолданбалы деңгей хаттамалары алады:

- DNS — доменді атауларды бөлу жүйесі, ол хосттың доменді атауы бар сұрау салу бойынша IP-мекенжайын хабарлайды;
- HTTP — Интернетте гипермәтінді беру хаттамасы;
- HTTPS — шифрлауды қолдайтын HTTP хаттамасын кеңейту;
- FTP (File Transfer Protocol — RFC 959) — компьютер желілерінде файлдарды беруге арналған хаттама;
- Telnet (TELecommunication NETwork — RFC 854) — желі бойынша мәтіндік интерфейсті іске асыруға арналған желілік хаттама;
- SSH (Secure Shell — RFC 4251) — операциялық жүйені қашықтықтан басқаруға және файлдарды беруге мүмкіндік беретін қолданбалы деңгей хаттамасы. Telnet қарағанда бүкіл трафикті шифрлайды;
- POP3 — пошта клиентінің хаттамасы, ол серверден электронды пошта хабарламасын алу үшін қолданылады (пайдаланушы компьютеріне аусытыру);
- IMAP — электронды поштаға қол жетімділік хаттамасы. POP3 қолдану кезінде клиент жаңа хабарламаларды жүктеу үшін қажетті уақыт аралығына ғана қосылады. IMAP қолдану кезінде пайдаланушы интерфейсі белсенді болғанша байланыс үзілмейді, ал хабарламалар клиенттің талабы бойынша ғана жүктеледі. Бұл жәшігінде көлемі үлкен көп хабарлама бар пайдаланушы үшін дыбыс беру уақытын азайтуға мүмкіндік береді;
- SMTP — поштаны пайдаланушыдан серверлерге және одан әрі алушыға жіберу үшін серверлер арасында жіберу үшін пайдаланылатын хаттама;

- LDAP — X.500 каталогы қызметіне қол жетімділіке арналған хаттама каталог қызметтеріне қол жетімділіктің кеңінен қолданылатын стандарты болып табылады;
- XMPP (Jabber) — шынайы уақытта хабарламалармен тез алмасу үшін кеңейтілген хаттама XML негізделген;
- SNMP — Интернетті басқарудың базалы хаттамасы.

*HTTP (HyperText Transfer Protocol)* — гипермәтінді беру үшін қолданбалы деңгей хаттамасы. Қарапайым түрінде Web-сервер HTTP-сұрау салу желісі бойынша белгілі бір ресурс алады, жергілікті қатты дискте тиісті файлды табады да оны сұрау салған компьютерге желі бойынша жібереді. Аса күрделі Web-серверлер HTTP-сұрау салуға жауап ретінде ресурстарды динамикалық қалыптастыра алады.

WWW қызметі гиперорта қағидасында ұйымдастырылған — бір бірімен ұқсастық негізде байланысқан шамалы блоктар түрінде ақпарат ұсыну технологиясы.

*Гипермәтін* — ақпараттық массивтерді ұйымдастыру қағидасы, онда жеке ақпараттық элементтер қажетті ақпаратты тез іздеуді және (немесе) өзара байланысты деректерді қарап шығуды қамтамасыз ететін ассоциативті қатынастармен өзара байланысқан.

*Ассоциативті байланыс* — реттелген бірізді тізбек құратын деректер белгілерінен берілген үйлесімділігінен белгіленетін байланыс түрі. Байланысқан деректерге көрсеткіштер деректердің өзінде немесе деректер базасын басқарудың бағдарламалық құралдарында орналасуы мүмкін.

Гипермәтінде материал оның бірліктері тізбек реттілігінде емес, олар арасындағы ықтимал ауысу мен байланыстарды анық көрсететін жүйе түрінде ұйымдастырылған. Осы байланыстарға негізделе отырып, әртүрлі тізбекті мәтін құра отырып материалды кез келген тәртіпте оқуға болады. Көбінесе гипермәтінді ақпарат байланысты тораптар жиынтығы түрінде болады. Оқырмандар ақпаратты, бір тораптан екіншіге көшіп әр түрлі тәсілмен ала алады. Гипермәтін мысалы ретінде HTML (белгілеудің гипермәтінді тілі) құжаттары бола алады.

WWW «клиент-сервер» схемасы бойынша құрылған.

*Web-сервер* — клиенттерден көбінесе, HTML-парақпен, бейнемен, файлмен, медиаағыммен немесе басқа деректермен бірге HTTP-сұрау салу қабылдайтын сервер. Web-сервер дегеніміз Web-сервер функциясын атқаратын бағдарламалық қамтамасыз ету, сондай-ақ осы бағдарламалық қамтамасыз ету жұмыс істейтін тікелей компьютер.

Web-браузер болып табылатын клиент Web-серверге URL-мекенжайлармен белгіленген ресурстарды алуға сұрау салады. Ресурстар — бұл HTML-парақтар, бейнелер, файлы, медиаағындар немесе клиентке қажетті басқа деректер. Жауап ретінде Web-сервер клиентке сұратқан деректерін береді. Осы алмасу HTTP хаттамасы бойынша жүргізіледі. Мәтін мен бейнеден басқа Web-парақтар басқа парақтарға сілтемеден, графикалық бейнелерге сілтемеден, аудиу және бейне ақпараттан, деректер енгізудің әртүрлі объектілерінен (өрістер, пернелер, нысандер жіне т.б.), соедай-ақ басқа объектілерден тұруы мүмкін. Web парақтары әртүрлі тип объектілері арасындағы байланыстыратын буын болуы мүмкін.

*Web-сайт* — жалпы мекенжайдағы (доменді атауы немесе IP-мекенжайы) компьютер желісіндегі жеке адамның немесе ұйымның электронды құжаттардың жүйесі (деректер мен код файлы).

*Браузер* — ол Web-парақтың әртүрлі құрамдас бөліктерін өндеу және шғару үшін, сондай-ақ Web-сайт пен пайдаланушы арасындағы интерфейсті ұсыну үшін кешенді қосымша. Браузер серверге жолданатын деректерді алдын ала өндей алады, сондай-ақ серверден алынған нәтижелерді пайдаланушыға ыңғайлы түрде өндей және ұсына алады. Браузер көбінесе «жіңішке клиент» деп аталады, яғни бизнес-логиканың минималды аз саны бар клиент.

*Web-қосымша* — ол серверлі қосымша клиент, онда клиент рөлінде браузер болады, ал сервер Web- сервер болады. Web-қосымша логикасы сервер мен клиент арасында бөлінген, деректерді сақтау серверде жүзеге асырылады, ақпарат алмасу желі бойынша болады. Мұндай жолдың артықшылықтарының бірі клиенттер пайдаланушының нақты операциялық жүйесіне байланысты емес, сондықтан Web-қосымша платформа аралық сервис болып табылады. Web-қосымшалар клиент жақта қосымша бағдарламалық қамтамасыз етуді орнатуды қажет етпейді, ал барлық логика негізінен сервер жақта орындалады. Web-бағдарламалау клиент-серверлі қосымшаны бағдарламалаудың жеке жағдайы болып табылады.

## 5.2. IP- АДРЕСАЦИЯЛАУ

IP хаттамасы желілік деңгейге жатуына байланысты, IP-мекенжайларды желілік мекенжайлар деп атайды.

*IP-мекенжайы* (Internet Protocol Address) — IP хаттамасы бойынша құрылған компьютер желісіндегі бірегей желілік мекенжайы.

Интернетте мекенжайдың жаһындық бірегейлігі қажет, жергілікті желіде жұмыс істеген жағдайда желі шегінде мекенжай бірегейлігі қажет.

*Жергілікті IP-мекенджай* жергілікті желідегі белгілі ресурстарға қол жетімділік алу үшін құрылғыға белгіленеді.

*Сыртқы IP-мекенжай*— ол бірегейлігі жергілікті желіде емес Интернет бойынша жаһандық анықталатын IP-мекенжай.

IP-мекенжайлар — мекенжайлардың негізгі типі, оның негізінде желілік деңгей *IP-пакеттер* деп аталатын хабарламалар жібереді. IP-мекенжай екі мәнді сан түрінде болады. IPv4 хаттамасы нұсқауында IP-мекенжайы ұзындығы 4 байт немесе 32 бит, ықтимал бірегей мекенжайымен 4 294 967 296 ($2^{32}$) шектелген мекенжай кеңістігі. IPv4 (Internet Protocol version 4) — IP-хаттаманың төртінші нұсқасы, бірінші кеңінен қолданылатын нұсқасы. 6-шы нұсқада IP-мекенжайы 128-бит болады. Pv6 (Internet Protocol version 6) — IP хаттамасының жаңа нұсқасы, ол алдыңғы нұсқасы (IPv4) Интернетте оны пайдалану кезінде мекенжай ұзындығы 32 бит орнына 128 бит қолдануы есебінен проблеманы шешуге арналған. Қазіргі уақытта IPv6 хаттамасы бүкіл әлем бойынша бірнеше мың желіде (2013 жылғы күзде 14 000 желіден астам) пайдаланылады, бірақ әлі IPv4 сияқты Интернетте кеңінен тараған жоқ.

IP-мекенжайды (IPv4) ыңғайлы жазу нысаны нүктемен бөлінген төрт ондық сан (0-ден 255) түрінде жазу болып табылады (мысалы, 192.168.0.1). Бұл мекенжай ұсынудың дәстүрлі ондық формасы. Бұл ретте әрбір төрт ондық сан 8 битті пайдаланып есептеудің екілік жүйесінде жазуға болады. Бұл жағдайда сегіз екілік бірліктен тұратын сан максималды болады — 11111111, ол ондық жүйеге қайта санағанда 255-ке тең болады.

Осылайша IP-мекенжайдың әртір төрт саны ондық түрінде 255-тен көп болмайды.

IP-мекенжай екі бөлімнен тұрады: желі мекенжайы және торап нөмірі.

*Желі мекенжайы (желі сәйкестендіргіш)* — ол TCP/IP хаттамасын пайдаланатын аса ірі біріккен желідегі (желі желісі) бір желілік сегмент. Бір жүйеге қосылған барлық жүйелердің IP-мекенжайының бір ғана желі сәйкестендіргіші болады.

*Торап нөмірі (сәйкестендіргіш, торап мекенжайы)* әрбір желі шегінде TCP/ IP (жұмыс станциясын, сервер, бағдарлауыш немесе басқа TCP/IP- құрылғы) торабын айқындайды. Торап сәйкестендіргіші ол қосылған желі сегментінде жүйені белгілейді.

IP-мекенжайы желідегі жеке торапты емес, жеке желілік интерфейсті сәйкестендіретінін ұмытпай қажет. Торап бірнеше желіге кіруі мүмкін, ол жағдайда саны бойынша онымен байланысты желілердің бірнеше IP-мекенжайы болуы тиіс. Осылайша анықтамасы бойынша бағдарлауыш бірнеше желіге кіреді, сондықтан бағдарлауыштың әрбір портының жеке IP-мекенжайы болады. Соңғы торап бірнеше IP-желіге кіруі мүмкін. Бұл жағдайда компьютерде желі байланыстарының саны бойынша бірнеше IP-мекенжайы болуы тиіс.

Арнайы резервтелген IP-мекенжайға мыналар жатады:

- екі нөлден тұратын мекенжай. Мұндай IP-мекенжайынан тұратын пакетті өндірген желі торабын белгілейді;
- желі нөмірі барлық нөлдерден тұратын мекенжай. Осы желіні белгілейді, яғни осы мекенжаймен пакетті өндірген компьютер бар желі;
- торап нөмірі барлық нөлдерден тұратын мекенжай. Пакеттік өндірген торап жататын желіге жататын торапты белгілейді;
- мекенжай екі бірліктен ғана тұрады — осындай мекенжайы бар пакет пакеттің дереккөзі сияқты желінің барлық тораптарына арналғанын білдіреді. Бұл ретте пакет осы желі шегінен шықпайды, сондықтан мұндай жіберілім шектелген кеңінен тарату деп аталады;
- 127.0.0.1 мекенжайымен пакет желіге жіберілмейді, жаңа ғана қабылданған ретінде хаттама стегінің жоғарғы деңгейіне қайтып келеді. 127 басталатын мекенжайлар бағдарламалық қамтамасыз етуді тестілеу және желілік процестердің жеке торап шегінде өзара іс-қимыл жасасуы үшін пайдаланылады. Сондықтан желіде компьютерлерге 127 басталатын мекенжай орнатуға тыйым салынады.

Мекенжай жүйесін иілгіш ету үшін IP-мекенжайлардың бүркемелері қолданылады. *Қосымша жүйе бүркемесі* немесе желі бүркемесі TCP/IP желісі терминологиясында желі торабының IP-мекенжайының қай бөлігі желі мекенжайына, ал қай бөлігі осы желідегі тораптың өзіне жататынын анықтайтын бит бүркемесі. Ол үшін бір желі мекенжай кеңістігін бір бірімен қиылыспайтын диапазондарға бөле отырып бірнеше қосымша желіге бөлінеді. Бүркеменің екі рет жазылуы IP-мекенжайда желі нөмірі ретінде түсіндірілуі тиіс разрядтарда бірліктерден тұрады. Бүркеме битінің мәні бірге тең болған жерде тораптарды дербестендіруге тыйым салынады, ал мәні нөлге тең жерде рұқсат етіледі. Бұл ретте «бүркемелену» аға биттен қосымшаға қарай жүреді. Желінің стандартты класы үшін бүркемелер мынандай мәнге ие:

- A класы — 11111111.00000000.00000000.00000000 (255.0.0.0);
- *B* класы — 11111111. 11111111.00000000.00000000 (255. 255.0.0);
- *C* класы — 11111111. 11111111. 11111111.00000000 (255. 255. 255.0).

Бүркемелер механизмі IP-бағдарлауда кеңінен тараған, бұл ретте бүркемелер әртүрлі мақсатта пайдаланылуы мүмкін. Бүркемелердің көмегімен администратор қызметтерді жеткізушілерден қосымша желі нөмірлерін қажет етпей өз желісін құрылымдай алады.

Қызметті жеткізушілер префикстер енгізе отырып бірнеше желінің мекенжай кеңістігін біріктіру үшін бүркемелер механизмін қолданады. Осылайша бағдарлау кестесінің көлемі азаяды және бағдарлаудың өнімділігі артады.

Егер желі Интернеттің бір бөлігі болса, онда желі нөмірлері орталықтандырылып белгіленеді. Егер желі оңаша жұмыс істейтін болса, онда желі нөмірлері еркін белгіленеді.

## 5.3. DNS ДОМЕНДІ АТАУЛАРДЫҢ ЖҮЙЕСІ

Компьютерлерді сәйкестендіру үшін аппараттық және бағдарламалық қамтамасыз ету TCP/IP желілерінде бағдарлама қандай хостқа жүгіну керек екенін дұрыс түсіну үшін, IP-мекенжайды қолданады. Мысалы, http://81.19.70.3 әмірі «Рамблер» интернет-порталының бастапқы бетін ашады. Бірақ, пайдаланушыларға компьютерлердің таңбалы атауларымен жұмыс істеу ыңғайлы, мысалы браузердің мекенжайы жолында rambler.ru енгізген жеңіл. Адамға сандарды есте сақтауға қарағанда, ойланған атауларды есте сақтау жеңіл. Қазіргі уақытта Ш-мекенжайлар өте көп, олардың тіптен аз бөлігін жаттап алу қажет. Сондықтан адресациялаудың сандық схемасынан басқа мекенжайларды таңбалармен көрсету қолданылады. Яғни TCP/IP желілерінде хосттардың таңбалы атаулары және таңбалы атаулары мен IP-мекенжайы арасындағы сәйкестікті анықтау үшін механизм болуы тиіс.

*DNS (Domain Name System — доменді атаулар жүйесі)* — TCP/IP желілерінде IP-мекенжайларының домендерінің таңбалы атауларын түрлендіруге және керісінше мүмкіндік беретін жүйе.

*Домен (domain)* — қандай да бір елге, ұйымға және т.б. бөлінген Интернеттің доменді атаулары жүйесіндегі (DNS) белгілі бір аймақ.

Доменнді атау бір бірінен нүктемен ажыратылған бірнеше бөліктен тұруы мүмкін, мысалы mail. rambler. ru. Мұндай бөліктердің әрқайсысын домен деп те атайды.


5.3-сурет. Домен атының компоненттері

Доменде сол жағында сатының ең төменгі деңгейі бар доменге кіретін торап аты, ал оң жағына — ең жоғары сатылы деңгейі бар домен жазылады. Сондықтан, оң жағындағы шеткі домен *жоғарғы* немесе *бірінші деңгейлі домені* деп аталады. Келесі нүктемен ажыратылған сол жақтағы домен бірінші деңгейлі доменіне қарағанда, еншілес домен болып табылады, яғни оның компоненттері ретінде оған кіреді. Бұл домен *екінші деңгей домені* деп аталады. Екінші деңгей домені үшін еншілес болып табылатын домендер *үшінші деңгейлі домені* деп аталады және т.б. Мысалы, mail.rambler.ru мекенжайында «ru» домені жоғарғы (бірінші) деңгейлі домені болып табылады, екінші деңгейлі домені — «rambler», «mail» сөзі үшінші деңгей — хост атауы болып табылады (5.3-сурет).

Доменді атаулардың сатысы файлдық жүйеде қабылданған файл аттарының ұқсас. Бірінші деңгейлі доменінің аты халықаралық стандарттарға сәйкес орталықтандырылып белгіленеді. Бірінші деңгейлі домендерінің аттары елдерді немесе ұйым типтерін белгілеуі мүмкін және екі немесе үш әріптен тұратын аббревиатура түрінде болады (5.1-кесте).

5.1-кесте. Бірінші деңгей домендерінің мысалы

| Домендер | | Белгіленуі |
|---|---|---|
| Жалпы (ұйым типтері) | com | Коммерциялық |
| | edu | Білім беру |
| | gov | Үкіметтік |
| | int | Халықаралық |
| | mil | Әскери |
| | info | Ақпараттық |
| | net | Желілік |
| | org | Коммерциялық емес |

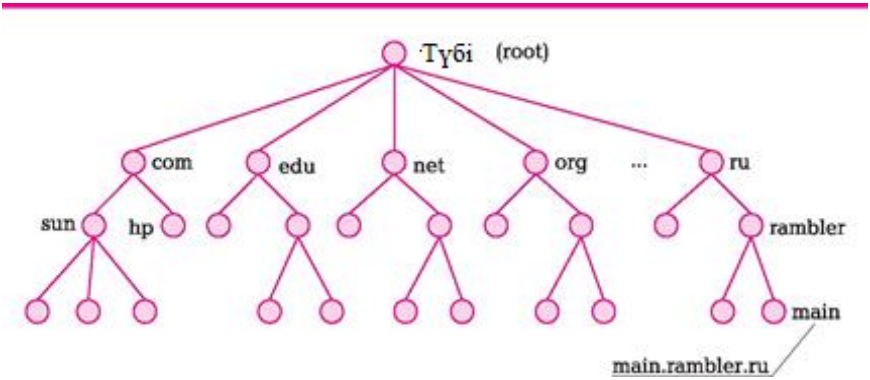
5.1-кесте соңы

| Домендер | | Белгіленуі |
|---|---|---|
| Аймақтық (елдер және аймақтар) | ru | Ресей Федерациясы |
| | ua | Украина |
| | us | АҚШ |
| | jp | Жапония |
| | de | Германия |
| | gb | Ұлыбритания |
| | au | Аустралия |
| | za | Оңтүстік Африка |

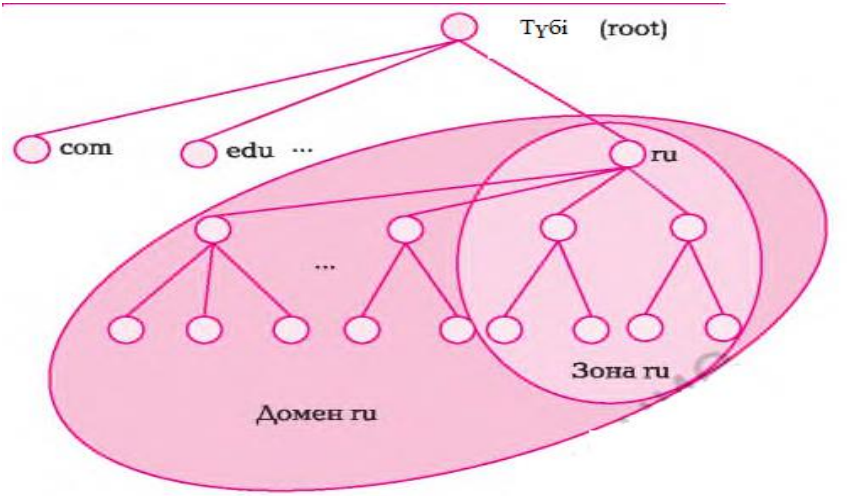
Егер бір домен екінші доменге оның компоненті ретінде кіретін болса, онда ондай домен *қосымша домен* немесе *қосалқы домен* (subdomain) деп аталуы мүмкін.

Атын бөліктерге бөлу әртүрлі адамдар немесе ұйымдар арасында өзінің сатылығы деңгейінінің шегінде бірегей аттар белгілеу үшін әкімшілік жауапкершілікті бөлуге мүмкіндік береді. Ол ұйымдардың өзара келісімінсіз бірегей аттарды түзу проблемасын шешуге мүмкіндік береді. Сатының жоғарғы деңгейі атын белгілеу үшін, жауап беретін бір ұйым болуы керек екені айқын.

DNS реттелген деректер базасы болып табылады. DNS деректер базасы *аттардың доменді кеңістігі* деп аталатын ағаш құрылымы болады (5.4-сурет). Доменнің аты аталық доменге қатысы бойынша осы деректер базасында оның орналасуын сәйкестендіреді, бұл ретте атаудағы нүктелер домен торапына сәйкес келетін бөліктерді бөледі.




5.4-сурет. DNS деректер базасының құрылымы




5.5-сурет. «ru» доменнің «ru» аймағының мысалы

DNS сервисінің көмегімен орталықтандырылған доменді аттарды желі мекенжайларына орнату жүзеге асырылады. DNS-сервер бақылайтын доменді атау кеңістігінің бір бөлігі *аймақ* деп аталады (5.5-сурет).

DNS сервисінің жұмыс істеу қағидасы «клиент-сервер» сәулетін пайдалануға негізделген, онда DNS-серверлер мен DNS- клиенттер анықталған. DNS-сервер (аттар сервері) — доменді аттар кеңістігі туралы ақпарат сақтайтын бағдарламалық қамтамасыз ету. DNS-клиенттер IP-мекенжайға доменді атауды көрсету туралы сұрау салумен серверге жүгінеді. Кез-келген торап ізделетін желінің кез-келген торапының IP-мекенжайы туралы мәлімет ала алады. Ол үшін, торап іздеуді жүзеге асыратын осы торап және ізделетін торап үшін, жалпы доменде орналасқан серверге жеткенше реттілікпен саты бойынша жоғары орналасқан барлық DNS-серверлерге жүгінеді. Одан әрі ізделетін торабы бар домен табылмағанша, доменді саты бойынша төмен орналасқан серверлерге ретпен жүгіну болады. Іс-жүзінде ізделетін DNS-серверге жақыннан іздеу басталады. Егер олар туралы ақпарат кэште болса және ескірмеген болса, сервер DNS- сервелерге сұрау салмайды.

Бұрын атап өткендей DNS әзірлеудің негізгі мақсаттарының бірі орталықтандырылған басқаруы бар реттелген деректер базасын құру болып табылған.

Бұған өкілеттіктерді табыстау арқылы қол жеткізіледі. Доменді басқаратын ұйым оны қосымша домендерге бөле алады және осы қосымша домендерді басқару құқығын басқа ұйымдарға бере алады. Ол басқару құқығы берілген (өкілеттік табысталған) ұйым осы қосымша доменнің ұсталуына және барлық деректерін басқаруына жауапты болады деген сөз. Ол деректерді ерікті түрде өзгерте алады, өз қосымша домендерін одан төмен деңгейлі қосымша домендерге бөле алады және оларды басқа ұйымдарға табыс ете алады. Аталық доменде сұрау салуларлы осы қосымша домендердің ресурстарына жолдай алатындай, өзінің қосымша домендерінің дереккөздеріне көрсеткіштер (көбінесе атаулары серверіне көрсеткіштер) болады.

Бір IP-мекенжайдың бірнеше аты болуы мүмкін, ол компьютерде бірнеше Web-сайт ұстауға мүмкіндік береді (виртуалды хостинг). Кері де болады — бір атқа көптеген IP-мекенжай сәйкестендірілуі мүмкін, ол жүктемені теңгеруге мүмкіндік береді.

Жүйенің тұрақтылығын арттыру үшін, осыған ұқсас ақпарат бар көптеген сервер пайдаланылады, ал хаттамада әртүрлі серверде орналасқан ақпараттың үйлесімділігін ұстауға мүмкіндік беретін құралдар болады.

Мынадай жолмен DNS сипаттамасын анықтауға болады:

- *ақпаратты сақтау реттілігі.* Желінің әрбір торабы міндетті түрде оның жауапкершілік аймағына ғана кіретін деректерді және түпкілікті DNS-серверлер мекенжайын сақтауға тиіс;
- *сатылы құрылымы,* онда барлық тораптар ағашқа біріктірілген және әрбір торап төмен тұрған тораптардың жұмысын өздігінен анықтай немесе оларды басқа тораптарға табыстай (бере) алады;
- *қауіпсіздік және резервтеу.* Өз тораптарын (аймақтарын) сақтау және қызмет көрсету үшін (көбінесе) физикалық және логикалық тұрғыдан бөлінген бірнеше сервер жауап береді, ол деректердің сақталуын жәгне тораптардың біреуі істен шыққан кезде де жұмыстың жалғастырылуын қамтамасыз етеді.

DNS барлық бағдарламалық шешімдері қорғауды қажет етеді. Егер хакер DNS-серверге басып кірсе пайдаланушылар ол туралы білмей тұзаққа түседі. Шауылдың осы қосымша түрімен мынадай қауіптер байланысты:

- DNS-шабуылы нәтижесінде пайдаланушы қажетті бетке кіре алмай қалады. Сайт мекенжайын енгізген кезде, шабуылданған DNS сұрау салуды басқа бетке жіберетін болады;

- пайдаланушы жалған IP-мекенжайға кіру нәтижесінде хакер оның жеке ақпаратына кіре алады. Бұл ретте пайдаланушы оның ақпараты құпиясыздандырылып қалған білмей де қалады;
- DDoS-шабуылдар DNS-сервердің жұмыс істей алмай қалуына әкеп соғады. DDoS-шабуыл — ол есептеу жүйесі істен шыққанға дейін жеткізу мақсатымен шабуыл жасау, яғни мұндай жағдайлар құрғаннан, онда жүйенің заңды пайдаланушылары ұсынылатын жүйелі ресурстарға кіре алмайды не бұған кіру қиын болады. DNS-сервер қол жетімді болмағанда пайдаланушы оған қажетті бетке кіре алмайды, себебі оның браузері енгізілген сайт мекенжайына сәйкес келетін IP-мекенжайды таба алмайды. DNS-серверлерге DDoS-шабуылдар DNS-сервердің төмен өнімділігі есебінен де, байланыс арнасының жеткіліксіз еңі есебінен жүзеге асырылуы мүмкін.

Доменді атау алу рәсімі *доменді тіркеу* деп аталады және DNS деректер базасында домен әкімшісіне көрсететін жазба құруға негізделеді. Тіркеу тәртібі және талаптар тағдалған доменді аймаққа байланысты. Доменді тіркеуді тіркейтін ұйым да, егер таңдалған доменді аймақ қағидалары мүмкіндік беретін болса жеке тұлға да орындай алады.

## 5.4. WEB-РЕСУРСТАРДЫҢ СӘЙКЕСТЕНДІРГІШТЕРІ

Web желісінде ресустарды бірдей анықтау үшін, бірегей URI сәйкестендіргіштер пайдаланылады.

*Ресурсты бір қалыпты сәйкестендіргіш* (Uniform Resource Identifier — URI) дерексіз немесе физикалық ресурсты анықтайтын таңбалардың қысқаша реттілігі түрінде болады. URI ресурсты қалай алу керек екенін көрсетпейді, тек оны анықтайды. Ол Интернет арқылы алынбайтын ресурстарды (атаулары, аттары және т.б.) RDF (Resource Description Framework) көмегімен сипаттау мүмкіндігін береді. URI ең танымал мысалдары:

- *URL* (Uniform Resource Locator) — бұл ресурсты сәйкестендіруден басқа, бұл ресурстың орналасқан жері туралы ақпаратты да ұсынатын URI;
- *URN* (Uniform Resource Name) — бұл аттардың белгілі кеңістігінде ресурсты сәйкестендіретін URI, бірақ URL URN қарағанда, бұл ресурстың орналасқан жерін көрсетпейді.

URL — бұл Интернетте ресурстың мекенжайын жазудың стандартталған тәсілі. URL Тим Бернер-Ли 1990 жылы Женевада Ядролық зерттеулер жөніндегі еуропалық кеңесте ойлап шығарған. URL Интернеттегі негізгі жаңалық болды. Алғашында URL Дүниежүзілік торда ресурстар (көбінесе файлдар) орналасқан жерді белгілеуге арналған. Қазір URL Интернеттің барлық ресурстарының мекенжайын белгілеу үшін қолданылады және URI (Uniform Resource Identifier — таңбалар реттілігі, анықталатын абстрактылы немесе физикалық ресурс) ресурстарды анықтаудың жалпы жүйесінің бір бөлігі ретінде жайғастырылады. URL стандартын IETF ұйымы және оның бөлімшелері реттейді.

Алғашында URL ресурстардың желіде орналасуын максималды шынайы көрсету жүйесі ретінде әзірленген. Мынадай дәстүрлі URL жазу формасы бар:

```
<схема>://<логин>:<пароль>@<хост>:<порт>/<URL-жолы>
?<параметрлер>#<зәкір>
```

Бұл жазбада:

- `схема` — ресурсқа жүгіну схемасы, көбінесе желілік хаттама;
- `логин` — ресурсқа кіру үшін пайдаланылатын пайдаланушының аты;
- `пароль` — көрсетілген пайдаланушы паролі;
- `хост` — DNS жүйесінде хосттың толық жазылған доменді атауы немесе хосттың IP-мекенжайы;
- `порт` — қосылу үшін хост порты;
- `URL-жолы` — ресурстың орналсқан жері туралы нақтылайтын ақпарат, хаттамаға байланысты;
- параметрлер — серверге берілетін параметрлері бар мсұрау салу жолы. Парамерлерді бөлгіш — & белгісі;
- `зәкір` — ашылатын құжаттың кейбір бөлігіне сілтеме жасайтын «зәкір» сәйкестендіргіші. Бет көрсетілген зәкірга байланысты браузерде әртүрлі көрінуі мүмкін. URL жалпыға бірдей схемалары (хаттамалар) мыналардан тұрады:
- `ftp` — FTP файлын беру хаттамасы;
- `http` — HTTP гипермәтінін беру хаттамасы;
- `rtmp` — Real Time Messaging Protocol — деректерді үздіксіз беру үшін проприетарлы хаттама, негізінен ағымдық бейне жіне аудиоағымдарын Интернет арқылы Web-камерадан беру үшін пайдаланылады;
- `rtsp` — шынайы уақыттың ағымдық хаттамасы;
- `https` — шифрлауды пайдаланатын HTTP хаттамасын арнайы іске асыру (әдетте SSL немесе TLS, олар туралы кейін айтылады);

- `gopher` —Gopher хаттамасы;
- `mailto` — электронды пошта мекенжайы;
- `news` —Usenet жаналықтары;
- `nntp` — NNTP хаттамасы арқылы Usenet жаналықтары;
- `telnet` — Telnet интерактивті сессияға сілтеме;
- `file` — жергілікті фацл атауы;
- `data` — тікелей деректер (Data: URL);
- `tel` — көрсетілген телефон бойынша қоңырау шалу.

Қарапайым URL мекенжайына мысал келтірейік:
http://academia-moscow. ru/about/elektronnyy_kontent/index. php#sel

Осы жағдайда жол машинаның доменді мекенжайынан тұрады, оған HTTP сервері және сервер ағашының түбінен index. php файлына жол салынған. HTTP схемасы — WWW үшін негізделген — сәйкестендіргіштен, машина мекенжайынан, TCP-порттан, сервер директориясының жолынан, критерий іздеуден және белгіден тұрады. Ол URL кеңінен таралған түрі, WWW құжаттарында қолданылады. Машина мекенжайы ретінде IP-мекенжайды пайдалануға болады.

## 5.5. HTTP ХАТТАМАСЫ

HTTP хаттамасы компьютерлердің қолданбалы деңгейдегі өзара іс-қимылын белгілейді. HTTP гипермәтіннің блоктары болып табылатын хабарламаларды беруге арналған және жаһандық бірлік қызметінде пайдаланылады. HTTP үшін көлік хаттамасы ТОҚ хаттамасы шығады, бұл ретте HTTP (Web сервері) сервері клиент жақтан 80 ТОҚ порты бойынша стандартты қосылуды күтуде, ал HTTP (Web-браузер) клиенті қосылу бастамашысы болып табылады.

Web серверінің маңызды функцияларының бірі жергілікті файл жүйесіне рұқсат беруден тұрады. Ол үшін сервер түзетулерінде берілген Web сервері үшін түпкілікті болып табылатын кейбір директория көрсетіледі. Құжатты жариялау үшін, яғни осы серверге кіретін қол жетімді пайдаланушы етіп жасау (HTTP хаттамасы бойынша онымен қосылатын), бұл құжатты Web-серверінің немесе оның қосымша директорияларының біріне көшіру керек. HTTP хаттамасы бойынша қосу кезінде серверде пайдаланушы құқығы бар процесс құрылады, оның шынайы пайдаланушысы болмайды, ол сервер ресурсын көру үшін арнайы құрылған болады. Осы пайдаланушының құқықтары мен рұқсаттарын баптап, Web-сервердің ресурстарына қолжетімділікті басқаруға болады.

Клиент пен Web-сервер арасындағы өзара іс-қимыл хабарламалармен алмасу жолымен жүзеге асырылады. HTTP хабарламалары клиенттің серверге сұрағы және сервердің клиентке жауабы деп бөлінеді. HTTP негізі «клиент-сервер» технологиясы болып табылады. HTTP-дағы орталық объект — ресурс, оған клиенттің сұрау салуында URL көрсетеді. Көбінесе, мұндай ресурс ретінде серверде сақталатын файлдар болады. HTTP хаттамасының ерекшелігі сұрау салуда және жауапта да бір ресурсты әртүрлі параметрлермен көрсету тәсілін көрсету мүмкіндігіне негізделген: форматтар, кодтау, тілі және т.б. Хабарламаны кодтау тәсілін көрсету мүмкіндігінің арқасында, басында осы хаттама таңбалы ақпарат беруге арналған болса да, клиент пен сервер екілік деректермен алмаса алады. Бірінші рет қарағанда, бұл ресурстарды текке шығару болып көрінеді. Таңбалы белгі түріндегі деректер көп орын алады, хабарламалар байланыс арналарына қосымша жүктеме жасайды. Желі бойынша берілетін хабарламалар жеңіл оқылады және жүйелі әкімгер алынған деректерді талдап, қатені тез тауып алуы және оны жоюы мүмкін. Қажет болған кезде, өзар іс-қимыл жасайтын бір қосымшаның ролін талап етілетін форматта хабарламаны қолмен енгізетін адам орындауы мүмкін.

Көптеген басқа хаттамалардан айырмашлығы, HTTP клиенттардың алдынғы сұрау салулары және сервер жауаптары туралы ақпаратты сақтамайды. HTTP қолданатын компоненттер соңғы сұрау салулармен және жауаптармен байланысты жағдай туралы ақпаратты сақтауды өздері жасай алады. Мысалы, сұрау салуды жіберетін клиенттік Web-қосымшасы жауаптарды кідіруді қадағалауы мүмкін, ал Web-сервер соңғы клинеттердің IP-мекенжайларын және сұрау салу тақырыптарын сақтауы мүмкін.

HTTP хаттамасымен жұмыс істеуге арналған барлық бағдарламалық өнімдер үш негізгі санатқа бөлінеді:

1) серверлер — ақпаратты сақтау және қндеу үшін қыщметтер жтекізуші (сұрау салуларды өндеу);

2) клиенттер — сервер қызметтерін соңғы пайдаланушылар (сұрау салу жолдау);

3) көлік қызметтерінің жұмысын қолдауға арналған прокси-серверлер.

Браузерлер, мысалы: Internet Explorer, Opera, Mozilla Firefox, Netscape Navigator және т.б. негізгі клиенттер болып табылады. Web-серверлердің кеңінен тараған іске асырулары — Internet Information Services (IIS), Apache және т.б.

Apache — еркін Web-сервер. Apache кроссплатформалы бағдарламалық қамтамасыз ету болып табылады, ол GNU/Linux, BSD, Mac OS, Microsoft Windows, Novell NetWare, BeOS операциялық жүйелерін ұстайды.

Apache жүйеде Web-серверлердің бірі ретінде пайдаланылады, оның көмегімен қашықтықтан пайдаланушылар Интернет арқылы ақпараттық базамен жұмыс істей алады.

Internet Information Services (IIS) — *Microsoft* компаниясының атынан Интернеттің бірнеше қызметтері үшін серверлер жиыны. IIS Windows NT тобының операциялық жүйелерімен таралады. IIS жүйеде Web-серверлердің бірі ретінде пайдаланылады, оның көмегімен қашықтықтан пайдаланушылар Интернет арқылы ақпараттық базамен жұмыс істей алады.

Прокси-серверлерінің кеңінен тараған іске асырылуы: Squid, UserGate, Multiproxy, Naviscope.

HTTP-сеансының «классикалық» схемасы мынадай:

- ТОҚ-қосылысын орнату;
- клиенттің сұрау салуы;
- сервердің жауабы;
- ТОҚ-қосылысын ажырату.

Осылайша, клиент серверге сұрау салу жолдайды, одан жауап алады, одан кейін өзара іс-қимыл жасасу тоқтайды. Әдетте клинеттің сұрау салуы HTML-құжатты немесе басқа ресурсты беруді талап ету түрінде болады, ал сервер жауабы осы ресурс кодынан тұратын.

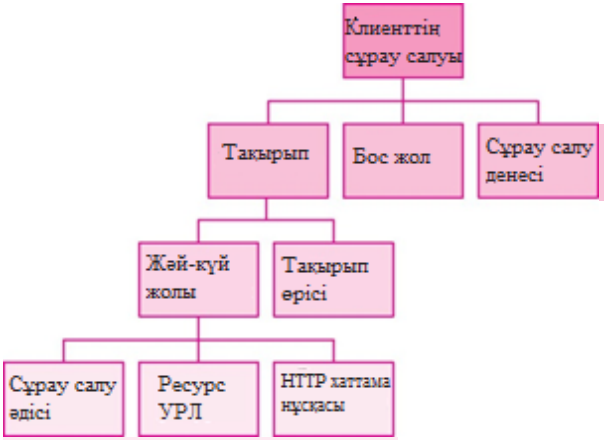
Клиент серверге беретін HTTP-сұрау салу құрамына мына компоненттер кіреді (5.6-сурет):

- күй жолынан және тақырып өрісінен тұратын сұрау салу тақырыбы (кейде оны белгілеу үшін статус-жол немесе сұрау салу жолы деген терминдер қолданылады);
- бос жол;
- сұрау салу денесі.

Күй жолы мынандай форматта болады:

- сұрау салу әдісі;
- URL_ресурс;
- HTTP хаттама нұсқасы.

Күй жолы компоненттерін қарап шығайық, бұл ретте сұрау салу әдістеріне ерекше көңіл бөлейік. Күй жолында көрсетілген әдіс URL сол жолда берілген ресурсқа ықпал ету амалымен анықталады. Әдіс GET, POST, HEAD, PUT, DELETE және т.б. мәндерін қабылдай алады. Әдістердің көп болуына қарамастан, Web-бағдарлама жасаушыға оның екеуі ғана маңызды: GET және POST.




5.6-сурет. Клиенттің сұрау салуының құрылымы

GET көрсетілген URL көрсетілген ресурсты алуға арналған. GET сұрау салуын алып, сервер көрсетілген ресурсты оқуы тиіс және ресурс кодын клиентке жауап құрамына қосуы тиіс. URL сұрау салу құрамында берілетін ресурс HTML-бет, бейнесі бар файл немесе басқа деректер түрінде болуы міндетті емес. Ресурс URL бағдарламасының орындалатын кодын көрсетуі мүмкін, ол белгілі бір жағдайды сақтаған кезде серверге жіберілуі тиіс. Бұл жағдайда клиентке бағдарлама коды емес, оны орындау процесінде туындаған деректер қайтып келеді. Анықтамасы бойынша GET әдісі ақпарат алуға арналғанына қарамастан, оны басқа мақсаттарда да пайдалануға болады. GET әдісі серверге шағын деректер үзіндісін беруге сәйкес келеді.

Егер сұратылатын ресурстың сәйкестендіргіші құжатқа көрсетсе, онда сервер осы құжатты (файлды) қайтарады. Егер сұратылатын ресурс өз жұмысы процесінде кейбір деректерді қалыптастыратын қосымша (бағдарлама) болса, онда жауап хабарламасында осы деректер қайтарылады. Егер сұратылатын ресурстың сәйкестендіргіші директорияға көрсететін (каталог, папка) болса, онда сервер баптауына байланысты, не директория ішіндегілер (файлдар тізімі) қайтарылады не осы директориядағы бір файлдың ішіндегі (әдетте, index.) қайтарылады. Папка сұратылған жағдайда оның атауы соңында «/» таңбасымен көрсетілуі және онсыз да болуы мүмкін. Соңында осы таңба ресурсының сәйкестендіргіші болмаған жағдайда сервер қайта жолдап, жауаптардың біреуін шығарады.

GET командасының түрлері шартты GET (conditional GET) және жартылай GET (partial GET) болып табылады. Шартты GET келтірілген тақырыпта сипатталған жағдайларды объект қанағаттандыратын болса, ол объектіні беруді сұратады. Жартылай GET объектісінің бір бөлігін ғана беруді сұратады.

POST әдісінің негізгі мақсаты — деректерді серверге беру. Алайда GET әдісі сияқты, POST әдісі әртүрлі пайдаланылуы мүмкін және серверден ақпарат алу үшін пайдаланылады. Қалып күйінде берілген GET, URL әдісі сияқты, нақты ресурсты көрсетеді. Жіберілетін сервер сұрау салу хабарламасына (объектіге) қосылған деректерді қабылдайды және сұратылатын ресурс ретінде көрсетілген қосымшаны өндеуге жібереді. POST әдісі процесті іске қосу үшін де қолданылуы мүмкін.

HEAD және PUT әдістері GET және POST әдістерін түрлендіру болып табылады.

HEAD — GET әміріне ұқсас, тек мұнда сервер жауапта хабарлама денесін қайтармайды.

PUT — хабарлама денесі ол сұрау салуда жіберіледі, серверде сақталады, бұл ретте сұратылатын ресурс сәйкестендіргіші сақталған құжаттың сәйкестендіргіші болады.

OPTIONS — сұратылатын ресурс үшін сервер ұстайтын қосылыс опциялары туралы ақпаратты сұрату (мысалы, әдістер, құжат типтері, кодировка), оны сервер сұратылатын ресурс үшін қолдайды. Егер сұратылатын ресурстың сәйкестендіргіші — жұлдызша («*») болса, онда сұрау салу жалпы серверге қатынасуға арналған.

DELETE — сұратылатын сәйкестендіргіші бар ресурсты жоюға сұрау салу.

TRACE тестілеу немесе диагностика үшін қолданылады. Сұрау салуды алушы (Web сервері) алған хабарламаны жауап хабарламасының денесі ретінде клиентке қайта жібереді.

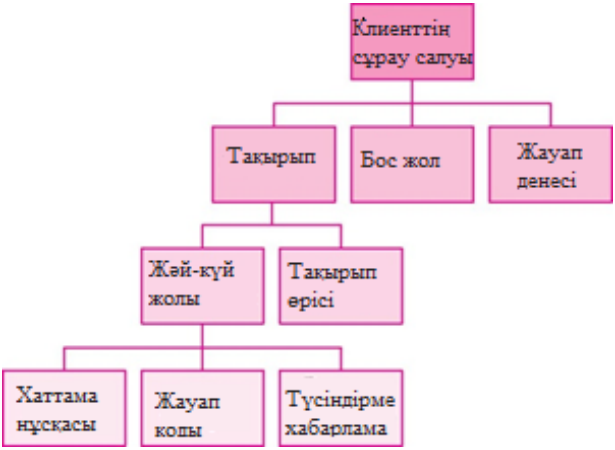
HTTP хаттамасының нұсқасы, әдетте мынадай форматта болады:

```
HTTP/ түрлендіру нұсқасы
```

Күй жолынан кейінгі тақырып өрісі сұрау салуды нақтылауға, яғни серверге қосымша ақпарат беруге мүмкіндік береді. Тақырыптан кейін мынадай форматта болады:

Өріс_аты: Мәні

Өрістің мақсаты мәнінен екі нүктемен ажыратылатын оның атынан белгіленеді.




5.7-сурет. Сервер жауабының құрылымы

Клиенттен сұрау салу алғаннан кейін, сервер оған жауап беруге тиіс. Сервер жауабы құрылымын білу Web-қосымшасын әзірлеушіге қажет, себебі серверде орындалатын бағдарламалар клиентке жауапты өздігінен қалыптастыруы тиіс.

Клиент сұрау салуы сияқты сервер жауабы мына компоненттерден тұрады (5.7-сурет):

- жай-күй жолы және тақырып өрісі;
- бос жол;
- жауаптың денесі.

Хаттама нұсқасы клиент сұрау салуындағы сияқты форматта беріледі және мағынасы сол.

*Жауап коды (жай-күй коды)* — ол сұрау салуды түсіну және қанағаттану нәтижесінің толық санды үш разрядты коды. Жай-күй кодының бірінші саны жауап класын анықтайды. Соңғы екі саны саралауда белгілі рөл атқармайды. Бірінші санның бес мәні бар:

1xx: Ақпараттық кодтар — сұрау салу алынды, өңдеу жалғасуда.

2xx: Сәтті кодтар — іс-қимыл сәтті қабылданды, түсінікті және өңделді.

3xx: Қайта жолдау коды — сұрау салуды орындау үшін, бұдан әрі іс-қимылдар қолданылуы тиіс.

4xx: Клиент қателерінің кодтары — сұрау салудың синтаксис қатесі бар немесе орындалуы мүмкін емес.

5xx: Сервер қателерінің коды — сервер рұқсат етілетін сұрау салуды орындай алмайды.

Жауап коды бағдарлмалық қамтамсыз етумен өндеуге, ал түсіндірме хабарлама – пайдаланушыларға арналған. Түсіндірме хабарлама жауаптың таңба түріндегі кодын көшіреді. Бұл клиентпен өнделмейтін таңбалардың жолы. Ол жүйеге қызмет көрсетумен айналысатын жүйелік әкімгерге немесе операторға арналған және жауап кодын ашып айтады.

## 5.6. ДЕРЕКТЕРДІ БЕРУ ҚАУІПСІЗДІГІН ҚАМТАМАСЫЗ ЕТУ. HTTPS ХАТТАМАСЫ

HTTP хаттамасы ашық шифрланбаған түрде таңба деректерін беруге арналған. Демек, клиент пен сервер арасында деректерді беру арнасына қолжетімді адамдар қиындықсыз бүкіл трафикті қарап шыға алады және оны санкцияланбаған іс-әрекеттер жасау үшін қолдана алады. Осыған байланысты, интернет-трафикті санкцияланбаған қолжетімділіктен қорғалуын арттыруға бағытталған базалық хаттаманың бірқатар кеңейтулері ұсынылды.

Ең қарапайым HTTPS кеңейту болып табылады, онда HTTP хаттамасы бойынша берілетін деректер SSL немесе TLS криптографиялық хаттамасына «қапталады», ол осы деректердің қорғанысын қамтамсыз етеді. HTTP қарағанда HTTPS үшін TCP-порт 443 пайдаланылады. HTTPS-қосылыстарды өндеу үшін Web-сервер дайындау үшін администратор жүйеге осы Web-сервер үшін сертификат алуы және орнатуы қажет.

SSL (Secure Sockets Layer — қорғалған сокеттердің деңгейі) — Интернет бойынша деректерді қауіпсіз беруді қамтамасыз ететін криптографикалық хаттама. Оны пайдалану кезінде клиент пен сервер арасында қорғану байланысы құрылады. SSL алғашында *Netscape Communications* компаниясы әзірлеген. Кейін SSL 3.0 хаттамасы негізінде TLS атауын алған RFC стандарты әзірленді және қабылданды. TLS (Transport Layer Security, — көлік деңгейінінің қауіпсіздігі) — бұл хаттама жіберуші мен алушының шынайылығын растау үшін ашық кілтпен шифрлауды қолданады. TSL түзететін кодтарды және қауіпсіз хэш-функцияларды пайдалану есебінен деректер беру сенімділігін ұстайды. Көп деңгейлі көлік хаттамасының төменгі деңгейінде (мысалы, TCP) ол жазба хаттамасы болып табылады және әртүрлі хаттамаларды (POP3, IMAP, SMTP или HTTP) инкапсуляциялау үшін пайдаланылады.

Әрбір инкапсулданған хаттама үшін TSL сервер мен клиент бір біріне шынайылығын растай алатын, шифрлеу алгоритмін орындай алатын және қолданбалы бағдарлама хаттамасы деректерді бере және ала бастамас бұрын криптографиялық кілттермен алмаса алатын жағдай жасайды.

SSL хаттамасымен қорғалған Web-параққа ені үшін URL-да http схемасының орнына https схемасы беріледі, ол SSL-қосылуы пайдаланылатынын көрсетеді. Стандартты TCP-порт https — 443 хаттамасы бойынша қосылуға арналған. SSL жұмысы үшін серверде SSL-сертификаттың бар болуы қажет.

Белгілі болғандай шифрлаудың классиклық криптотөзімді алгоритмдері бар, олар деректерді қолда бар кілт негізінде шифрлайды. Деректерді шифрлау және шифрын ашу үшін бір кілт қолданылады. Кілт — ол белгілі ұзындықтағы биттардың күнделікті реттілігі. Кілт ұзындығы көп болған сайын шифрлау алгоритмін бұзу қиын. Егер осы кілтті ашық түрде берсе шифрлау мағынасы жоғалады, сондықтан қосымша басқа шифрлау түрін – ассиметриялық қолданады. Бұл жағдайда қос кілт – ашық және жабық бар. Ашық кілттің көмегімен ақпаратты шифрлауға ғана болады, ал жабық кілттің көмегімен шифрды ашуға болады. Көбінесе мұндай жолда жабық кілт құпия сақталады, ал ашық кілт бәріне қолжетімді болады. Алайда ассиметиялық алгорим симметриялыққа қарағанда баяу жұмыс істейді, сондықтан оны алғашқы симметриялық кілтпен алмасу үшін пайдаланады.

HTTP-хаттамасының тұқым қуалаушылық проблемасы клиент пен сервер арасындағы қосылыстың тұрақсыздығына негізделеді, яғни HTTP-хаттамасы бойынша беру кезінде әрбір құжат (файл) үшін жеке сұрау салу жолданады. Басқа сөзбен айтқанда, браузер сұрау салу, ал сервер тиісті жауап бергеннен кейін транзакция аяқталады. Осыдан кейін сервер пайдаланушы туралы «ұмытып кетеді» және сол пайдаланушының кейінгі әрбір сұрау салуы жаңа пайдаланушы болып табылады. HTTP-хаттамасына cookie енгізу бұл проблеманы жартылай шешті.

*Cookie* — ол сервер браузерге беретін мәтіндік ақпараттың шағын бөлшегі. Браузер осы ақпаратты сақтайды және әрбір сұрау салу сайын HTTP-тақырыбының бөлігі ретінде серверге жібереді. Cookie бір мәндері бір сессия кезінде ғана сақталады, олар браузер жабылған соң жойылады. Уақыттың белгілі бір кезеңіне белгіленген басқалары фйлға жазылады.

Cookie пайдалана отырып HTTP хаттамасы бойынша сессияны эмуляциялауға болады: бірінші сұрау салуда cookie тиісті мәні беріледі, ал әрбір келесі сұрау салуда бұл мән ауыспалы қоршаудан (HTTP_COOKIE) оқылады және тиісті түрде өңделеді.

Қарапайым мысал: пайдаланушы өз есімін көрсетуі ұсынылатын нысан бар, одан cookie мәнін пайдаланушы браузеріне жазатын шағн бағдарлама (скрипт/ сценарий) шақырылады. Әрбір келесі кірген сайын cookie мәнін талдау негізінде пайдаланушы браузерінен бетте не атаулы сәлемдесу (егер орнатылған cookie мәні бар болса) не пайдалнушы есімін сұраумен алғашқы нысаны (егер cookie мәні орнатылмаған болса) пайда болады.

Өздігінен cookies ештене істей алмайды, бұл тек бірнеше мәтіндік ақпарат. Алайда сервер cookies-тағы ақпаратты оқи алады және оны талдау негізінде қандай да бір іс-әрекет істей алады. Мысалы WWW арқылы кез-келген затқа авторланған рұқсат алған жағдайда сессия кезінде login және password сақталады, ол пайдалнушыға парольмен қорғалған әрбір құжатты сұрату кезінде қайтадан енгізбеуге мүмкіндік береді. Cookies пайдалануға онлай дүкендерде тапсырыстарды ресімдеу функциясы жиі құрылады. Cookies пайдаланудың тағы бір тараған аймағы — әрбір тірелген пайдаланушының жеке профилінен түзету енгізу кезінде.

## БАҚЫЛАУ СҰРАҚТАРЫ ЖӘНЕ ТАПСЫРМАЛАР

1. Web-қосымша дегеніміз не?
2. Web-сервер дегеніміз не?
3. Қолданбалы деңгей хаттамасын атаңыз.
4. Гипермәтін деген не?
5. Қандай IP-мекенжайлар арнайы резевтелген болады?
6. Қандай IP-мекенжай Web- қосымшаны тестілеу кезінде пайдаланылады?
7. Домен деген не?
8. Бір IP-мекенжайдың бірнеше аты болуы мүмкін бе?
9. HTTP және HTTPS хаттамаларының айырмашылығы неде?
10. Web-қосымшадан клиентке сұрау салу өндеу циклін сипатаңыз.
11. HTTP хаттама қалай жұмыс істейді және ол не үшін керек?
12. HTTP-хабарлама тақырыбы деген не және олар не үшін керек?
13. HTTP-хабарлама денесі деген не?

14. HTTP-хабарламада тақырыптар хабарлама денесіне қалай ажыратылады?
15. HTTP-сұрау салуының HTTP-тақырыбына және HTTP-жауабына мысал келтіріңіз.
16. Шифрлаудың симметриялық алгоритмі симметриялық еместен қалай ажыратылады?
17. Қандай интерфейстер негізінде Web- сервер мен Web-қосымша өзара іс-қимыл жасаса алады?
18. HTTP деректерін қауіпсіз беру қалай қамтамасыз етіледі?
19. Cookie дегеніміз не?

# 6 тарау

# WEB – БЕТТЕРДІ БЕЛГІЛЕУ ТІЛДЕРІ

## 6.1. HTML ГИПЕРМӘТІНДІК ТІЛДІ БЕЛГІЛЕУ

*Құжат белгілеу тілі* — құжаттарда қандай да болсын құрылымдылық қалыптастыру және сол құрылымның түрлі элементтері арасында қатынасты анықтауға арналған теги деп аталатын арнайы нұсқамалардың жиынтығы. Тілдің тегі, кейде оларды басқарушы дескриптолары деп те атайды, бұл құжаттарда қалай да кодталады, құжаттың негізгі мазмұнына қатысты бөлініп тұрады және клиент жағында құжат мазмұнын көрсетіп шығарушы бағдарлама үшін нұсқаулық ретінде қызмет етеді. Белгілеу тілін пайдаланып жазылған мәтінді құжатта тек мәтін ғана емес, сонымен оның түрлі бөліктері туралы қосымша ақпарат болады, мысалы, бөліп шығару, тізімдер. Ал күрделі жағдайларда белгілеу тілі құжатқа интерактивтік элементтер және басқа құжаттардың мазмұнын кірістіріп салуға мүмкіндік береді.

Ақпаратты жариялау және оны жаһандық тарату үшін әлеуетті барлық компьютерлерге түсінікті негізгі жалпыға ортақ тіл типті әмбебап тіл қажет. Сөйтіп, World Wide Web – те пайдаланылатын тіл - HTML (HyperText Markup Language болады, гипермәтінді белгілеу тілі.

*HTML*- бұл Web-бетінде элементтердің қандай және қалай орналасуын анықтайтын беттеу жүйесі. Сайттағы ақпарат, оны ұсыну және рәсімдеу тек қана оны дайындаушы мен оның алдына қойған мақсатына байланысты болады. Ресми HTML SGML (Standard Generalized Markup Language – стандартты жалпыланған белгілеу метатілі) – нің қосымшасы болып табылады және ISO 8879 халықаралық стандартқа сәйкес келеді.

HTML тілін шамамен 1991 — 1992 жж. британ ғалымы Тим Бернерс-Ли Женева (Швейцария) қаласында ядролық зерттеу бойынша Еуропалық кеңес қабырғасында зерттеп дайындаған. HTML тілі беттеу саласының маманы емес адамдар үшін пайдалануға жарамды ғылыми және техникалық құжаттама алмасуына арналған тіл болып жасалған. HTML тілі біркелкі қарапайым, алайда әдемі рәсімделген құжаттарды жасайтын құрылымдық және семантикалық элементтері нің терімін анықтау жолымен («тег» деп белгіленетін) SGML күрделілік мәселесін ойдағыдай орындады. Құжат құрылымын қысқартудан басқа HTML –ке гипермәтінді қолдау енгізілді. Мультимедиялық мүмкіндіктері кейінірек қосылды. Бастапқыда HTML тілі құжатты қайта жаңғырту (кейіптеу) құралдарына бекітусіз құжатты құрылымдау және нысандау құралы ретінде ойластырып құрылған.

Қалаған түрінде HTML белгілеуімен мәтін стилистикалық және құрылымдық қатесіз түрлі техникалық жабдықталған құралда (заманауи компьютердің түрлі – түсті экраны, органайзер экраны, өлшемі бойынша шектелген ұялы телефон немесе қондырғының және мәтінді дауысты жаңғырту бағдарламасы) жаңғыртылуы керек еді.

HTML құжатты белгілеудің бейнелеу тілі болады және жасаушыларға келесіні жасау үшін амал – құралдарды ұсынады:

- мәтін, кесте, тізім, мазмұн, суреттермен т.б. online-құжаттарды жариялау;
- батырмаға басып, гиперсілтеме бойынша ауысып online- ақпаратты сұрау және т.с.;
- қашықтағы сервиспен өзара әрекеттестік жасау, ақпарат іздеу, мұрағат құру, тауарлар сату үшін формалар құру және т.б.;
- құжатқа деректер кестесін, видео- және аудиоақпаратты енгізу және басқа.

HTML құжаты басқарушы HTML-кодтары (тегтері) қосылған жай ASCII-файл болып саналады. Тегтік моделі құжатты әрқайсысы тегпен басталып тегпен аяқталатын контейнерлер жиынтығы ретінде бейнелейді. HTML –де тек үш басқарушы таңбалы қолдануға рұхсат берілген: көлденең табуляция, күймешені ауыстыру және жаңа жолды ауыстыру. Бұл құжаттармен жұмысты, сонымен қатар, түрлі операциялық жүйелермен өзара ірекет жасауды жеңілдетеді. HTML-құжаттардың тегтері көбінде қарапайым және түсінікті, өйткені жалпы қолданылатын ағылшын тілі сөздері, түсінікті қысқартулар мен белгілеулер көмегімен жасалған.

HTML- тег атаудан тұрады, одан кейін тегтің міндетті емес атрибуттар тізімі жалғаса алады. Тег мәтіні бұрышты жақшаларға алынады. Тег атрибуттары (параметрлер) атауынан кейін жүреді және бір – бірінен бір немесе бірнеше табуляция белгілері, аралықтар және жол басына қайтару таңбалармен бөлінеді. Тегте атрибуттарды жазу реттілігінің мәні жоқ. Атрибут мәні егер ондайы болса атрибут атауынан кейін тұратын теңдік белгісінен кейін тұрады. Тег пен атрибуттар атауларында таңбалар регистрі ескерілмейді, атрибуттар мәні туралы оны айтуға болмайды. Әр тег үшін ұйғарынды параметрлер жинағы жекеленген. Параметрлерімен тег жазбасының үлгісін көрсетейік:

```
<table border align="left">
```

Бұл жерде <table> тегі үшін екі параметр берілген. Бірінші параметр border мәнсіз көрсетілген. Екінші параметр align «left» мәні бар.

Жиірек HTML (HTML-контейнерлері) белгілеу элементтері бастапқы жіне соңғы тегтен тұрады, олардың арасында құжаттың мәтіні және басқа элементтері орналасады. Соңғы тегтің атауы бастапқы тегтің атауына сәйкес, бірақ соңғы тегтің атауы алдында қисық сызықша қойылады (мысалы, <TITLE> үшін жабатын жұп </TITLE> болады). Соңғы тегтер ешқашан атрибут ұстамайды. Тегтер өз мәні бойынша бағдарламалық тілдеріндегі локалды өзгермелілер және т.с. атаулары әрекеті саласын реттейтін «begin... end» операторлық жақшалар түсініктеріне жақын. Тегтер мәтінді құжаттарының интерпретация ережелері әрекеті саласын анықтайды. HTML-элементінің мәні бастапқы және соңғы тегтер арасындағы мазмұнды нысандау үшін қолдануда тұрады. Қолданылатын нысандау элемент атауына тәуелді.

Ашатын және жабатын тегтер HTML-элементтердің барлығында болмайды, яғни барлық тегтер контейнерлік болмайды, жалғыз тегпен кодтала алады, мысалы, link тегі сыртқы құжатпен байланыс орнатады:

```
<head>
<link атрибуттар>
</head>
```

Браузерлердің тегтерді түсіндірудің ортақ ережелері бар. Қате операторлар бағдарламаның компиляция кезеңінде сәйкес хабарламаны беруге жеткізетін және қатені түзеуді талап ететін бағдарламалау тілінен айрықша браузерлер HTML –де тегтердің дұрыс емес жазбаларына назар аудармайды.

Дұрыс жазылмаған тег немесе оның параметрі браузермен жай ғана ескерілмеуі керек. Бұл дұрыс емес тегтер және аталған браузер версиясымен танылмайтын барлық браузерлер үшін ортақ ереже.

<title> Тег-контейнері жалғыз міндетті атау тегі болып табылады және құжатқа атау беру үшін қызмет етеді. Әдетте ол браузер терезесі атауында көрсетіледі. <title> және </title> белгілері арасындағының бәрі браузермен құжат атауы ретінде түсіндіріледі.

Егер де қандай болсын Web-бетті ашсақ, ол өзінде сайттың түрі мен бағытынан өзгермейтін бір үлгідегі элементтерді қамтиды. Әрі қарай, HTML 4.01 версиялы негізгң тегтері бар қарапайым құжат кодының үлгісі беріледі:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//RU">
<html>
<head>
<MMe>Бұл бет атауы </MMe>
<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
<link href="css/styles. css" rel="stylesheet" type="text/css">
</head>
<body>
<h1>1 деңгей атауы</h1>
<!-- Бұл комментарий -->
<p>Бұл менің бірінші бетім HTML
<Ь>Бұл мәтін қалың қаріппен жазылады </b>
</p>
</body>
</html>
```

Көрсетілген үлгіде <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"> жолы құжаттың түрі мен форматын білдіреді. Бұл файлды басқа бағдарламалардың тануын жеңілдетеді. Басында HTML стандарты нұсқасын және құжат тілін көрсетеді. Көпшілік браузерлердің бірдей көрсетуін қамтамасыз ету үшін өзгеріссіз қалдыру ұсынылады.

HTML-құжаты HTML құжатының басталғаны туралы браузерге хабарлайтын <html> тегінен басталады да, HTML құжатының аяғына жеткені туралы браузерге хабарлайтын </html> тегімен аяқталады.

</html> тегінен кейін ештеңе жазылмайды.

<head> и </head> тегтер арасындағы мәтін құжат атауының ақпараты болып табылады. <head> тегі басқа элементтерді сақтауға арналған, мақсаты – браузерге деректермен жұмыста көмек беру. Сонымен қатар, <head> контейнерінің ішінде браузерлер мен іздеу жүйелеріне арналған ақпаратты сақтау үшін пайдаланылатын метатегтер болады. Мысалы, іздеу жүйелерінің механизмдері сайт, негізгі сөз және басқа да деректер сипатын алу үшін метатегтерге жүгінеді. <head> тегінің ішіндегісі <title> тегінен басқа Web-бетте көрінбейді.

<title> и </title> тегтер арасындағы мәтін «Бұл бет атауы» құжат атауы болады. <title> элементі құжаттың бөлшегі болмайды және тура Web-бетте көрінбейді. Windows операциялық жүйесінде атау мәтіні браузер терезесінің жоғарғы сол бұрышында бейнеленеді. Құжатқа тек бір ғана <title> тегі пайдаланылып, <head> контейнеріне орналастыруға болады. <meta http-equiv="Content-Type" content="text/html; charset=windows-12 51"> — құжатта windows 1251 кодымен HTML-мәтінінің барына көрсетеді – бұл кириллица.

<link href="css/styles. css" rel = "stylesheet" type="text/css"> — құжатқа CSS стилінде кестені қосады (бұл туралы кейінірек айтылады).

<body> и </body> тегтері арасындағы мәтін браузер терезесінде шығарылатын мәтін болады. <body> элементі браузер терезесінде көрінетін Web-беттің (контент) мазмұнын сақтау үшін арналған. Құжатқа шығарылатын ақпаратты нақ <body> контейнерінің ішіне орналастыру керек. Бұндай ақпаратқа мәтін, суреттер, тегтер, JavaScript скриптері т.б. жатады.

<h1> және </h1> тегтер арасындағы «1 деңгейдегі аты» мәтіні атау стилі болып үлкен өлшемдегі қалың қаріппен бейнеленіп көрсетіледі. HTML түрлі деңгейдегі атауынан кейін орналасқан секцияның салыстырмалы маңыздылығын көрсететін алты атауды ұсынады. <h1> тегі бірінші деңгейдегі ең маңызды атау болса, <h6> тегі алтыншы деңгейлі атауды белгілеуге қызмет етеді және маңыздылығы анағұрлым аз. Үнсіз келісім бойынша, бірінші деңгейлі атау ең ірі қаріппен қалың жазылады, келесі деңгедегі атаулар өлшемі бойынша кішірейеді.

<h1>,...,<h6> тегтері блокты элементтерге жатады, олар әрдайым жаңа жолдан басталады, ал одан кейінгі басқа элементтер келесі жолда белгіленеді. Бұдан басқа, атауы алдында және одан кейін бос орын қосылады.

<p> тегі мәтінді абзацты анықтайды. <p> тегі блокты элемент болып әрдайым жаңа жолдан басталады, бірінің артынан бірі келетін мәтін абзацтары аралық айырумен бөлінеді. Айыру көлемін стилдер көмегімен басқаруға болады. Егер жабушы тег болмаса, абзац соңы келесі блокты элементтің басымен сәйкес келеді деп есептеленеді.

 және  тегтері арасындағы мәтін «Бұл мәтін қалың қаріппен жазылады» қалың қаріппен жазылады.  тегін пайдалану HTML –де айыпталмағанмен, мәтін қалыңдығын басқаруда стилдерді қолдану көбірек мүмкіндік ұсынады. Басқа жағынан, іздеу жүйелері қалың жазылумен сөздерді «жақсы көреді» және олардың рейтингін көтереді.

Қандай да болсын HTML – құжаттың спецификацияға сәйкестілігі W3C белгілеуді тексеру онлайн Қызметі (http://validator.w3.org/) көмегімен тексеріледі.

HTML тілінің жоғары құндылық санына қарамастан, кемшіліктерін де бөліп шығаруға болады.

1. HTML –де бекітілген тегтер жинағы бар. Басқа пайдаланушыларға түсінікті өз тегтеріңді жасауға болмайды.
2. HTML — бұл тек қана деректер ұсынатын технология. HTML тегтерде қамтылған мазмұн мәні туралы ақпарат бермейді.
3. Қосымша үшін платформа ретінде браузерлер пайдаланылады. HTML –дің қазіргі уақытта Web-дайындаушылар ұмытылатын Web-қосымшалар деңгейлеріне жететін қуаты жоқ.
4. Тор трафик көлемі үлкен. Қосымша ретінде пайдаланылатын бар HTML-құжаттар үлкен көлемдерімен «клиент-сервер» жүйелерінде Ғаламторды ауырлатады. Бұған үлгі болып тек құжаттың шағын ғана бөлігі қажет болсада, тор бойынша құжаттың үлкен көлемінің салып жіберілуі болады.
5. Кемшіліктердің кейбіреулері XML белгілеу тілінде жойылуда.

Web-беттерді бағдарламалаудың оңтайлы стилін (HTML және XHTML көмегімен), ұстау үшін нысандау тегтері орнына CSS стиліндегі каскадты кестені пайдалану керек.

## 6.2. CSS СТИЛІНДЕГІ КАСКАДТЫ КЕСТЕЛЕР

HTML құжаттарды физикалық нысандау құралдары ретінде арнайы тегтер (мысалы, `<font>`, `<b>`, `<i>` тегтері) және көптеген әр түрлі атрибуттар (`size`, `color`, `height`, `width` және т.с.) ұсынады. Web-нысандаудің ерекшеліктері дайындаушыны бұл тегтер мен атрибуттарды сөзсіз әр жаңа абзацқа тіркеуге мәжбүрлейді, бұл бет көлемін қатты көбейтеді.

Нысандаудің бұл тәсілінде адам және іздеу машиналары үшін құжат құрылымын талдау қиындатылады. Логикалық талдаудың қиындылығынан бұл нысандаудің тәсілі *физикалық нысандау* деп аталады.

HTML спецификациясында құжаттың құрылымы мен рәсімделуі анық түрде бөлінген *логикалық нысандау* пайдаланылады. Аталған нысандау тәсілі Консорциумом WWW – дің қолдауымен ұсынылған, себебі Торда ақпарат іздеудің кеңейген мүмкіндіктерін ашады, іздеу машиналары арқылы ақпаратты нақты құрылымдау мен талдауға мүмкіндік береді, сонымен қатар, бет көлемін және толық жүктелу уақытын едәуір азайтады. Құжаттың құрылымын бөлу және рәсімдеу CSS көмегімен жүзеге асады (Cascade Style Sheets — стилдердің каскадты кестелері).

*Стиль* дегеніміз браузер терезесінде бейнеленгендегі HTML құжатының сыртқы түрін анықтайтын параметрлер жинағы: қаріп және түрлі деңгейлі атаулардың түстері, абзац тегінде берілетін негізгі мәтін қаріпі және т.б. Стиль белгілі ережелер бойынша жасалады.

*Стилдер кестесі* — сәйкес стиль кестесі бекітіліп қосылған құжатта қолданылатын бейнелеу ережесінің жинағы. Стил кестесі – бұл Web-құжаттағы HTML тегтерін нысандауді басқаратын шаблон.

«Каскадты» термины бір HTML құжатын нысандауді басқару үшін бірнеше стил кестелерін қолдану мүмкіндігіне ие болғасын пайдаланылады ал браузер белгілі ережелер бойынша бұл кестелердің қолдану басымдылығын құрады.

CSS синтаксисі үш бөлімнен тұрады: селектор, ерекшелік және мәні:

```
селектор {ерекшелік:      мәні}
```

Селектор — HTML –дің анықтау қажетті элементі/тегі. Ерекшелігі – өзгерту керек атрибуты. Бұл нақты бір стилдегі эффекті белгілейтін сөз немесе сөз тіркесі.

Әр ерекшеліктің өз мәні бар. Есте сақтау қажеттілігі бар ережелер мен ұсыныстар қатары бар.

Ерекшелік пен мәні қоснүктемен бөлінеді және фигуралы жақшаға салынады:

```
{font-size:   75%}
```

Егер мәні бірнеше сөзден тұрса, онда мәнін тырнақшаға енгізу қажет:

```
h1 {font-family: "lucida calligraphy"}
```

Егер біреуден артық ерекшелік санын анықтау қажет болса, бұл жерде ерекшелікті нүктелі үтірмен бөлу қажет:

```
table { font-family: arial,     "sans     serif";
border-
style: dotted}
```

Стилдер анықтамаларын оқуға ыңғайлы болу үшін, әр ерекшелікті бөлек жолға жазуға болады:

```
h2
{
font-family: arial;
margin-right: 20pt;
color:#ffffff }
```

Ережені анықтауда селекторларды топтауға болады, бұл жерде селекторлар бөлгіші ретінде үтір қолданылады. Келесі үлгіде топқа абзацтар, кесте және тізімдер элементтері біріктірілген. Бұл барлық элементтер sans serif қаріпімен шығарылады:

```
p,table,li
{
font-family: "sans serif";
}
```

Web-дайындауда кейбір бір HTML элемент типі үшін түрлі стилдегі тапсырмалар қажеттілігі туындайды. Бұл мәселені шешу үшін класс селекторын пайдаланады.

Егер құжат екі типті атауды талап етеді деп айтсақ: аса ірі атауы 10 пунктті сыртқы шегініс, ал екіншісі – 20 пункт болуы керек. Бұны осылай стилдер көмегімен жасауға болады:

```
stepleft {margin-left: 10pt}
stepright {margin-left:   20pt}
```

HTML құжатында бұл стилдерді қолдану үшін «класс» атрибутын пайдалану қажет:

```
<h1 class="stepleft">

10 пунктті сыртқы шегініспен атау.
</h1>
<h2 class="stepright">
10 пунктті сыртқы шегініспен атау.
</h2>
```

HTML элементтерінің стилдерін # таңбаымен анықталатын идентификатор селекторы көмегімен де анықтауға болады.

Стилдің келесі ережесі «fontsz» мәнді id атрибуты бар элементке қолданылады:

```
#fontsz{font-size: 50%}
```

Стилдің келесі ережесі «first» мәнді ul атрибуты бар элементке қолданылады:

```
ul#first
{
list-style: disc;
color: #ffffff
}
```

Стилдер кестесі құжаттың сыртқы келбетіне әсерін беру үшін, браузер оның бар екенін білуі керек. Бұл үшін оны HTML-құжатпен байланыстыру қажет. Байланыстыру стилдер кестесін бөлек файлда сақтау және <head> тарауында берілетін <link>тегі көмегімен құжаттарға қосу мүмкіндігін береді.

```
<link rel="stylesheet" type="text/css"
ref="mystyles. css"/>
```

Бұл жол кез келген html-файлда көрсетіледі. Сонымен, бірыңғай рәсімдеу бірнеше беттер үшін пайдаланылады. Бұл ретте стилдер кестесінің барлығы бір файлда сақталады (файлды кеңейту стандарттық болуы керек — css).

Қазіргі уақытта белсенді түрде CSS3 спецификациясы – үшінші буынды стилдердің каскадты кестесі жабдықталуда. CSS3 белгілеу тілінің көмегімен жүзеге асатын формалды тіл болып табылады. Бұл алдыңғы версиялармен (CSS1, CSS2 и CSS2.1) салыстырғанда ең ауқымды редакциясы. CSS3 негізгі ерекшелігі - JavaScript - ті пайдалабай анимациялық элементтерді жасау, желілік және радиалдық градиенттерді, іздерін қолдау, тегістеу және көптеген басқа мүмкіндіктері.

CSS3 айырықша HTML и XHTML белгілеу тілдері көмегімен жазылған Web-беттерін бейнелеу және рәсімдеу құралы ретінде пайдаланылып қана қоймай, сонымен қатар, кез келген XML-құжатында да қолданылады.

## 6.3. XML БЕЛГІЛЕРІНІҢ КЕҢЕЙТІЛЕТІН ТІЛІ

XML (eXtensible Markup Language) — құрылымдалынған деректерді сақтау және ақпаратты өңдеуде түрлі жүйелердің арасында деректермен алмасуға арналған мәтінді формат.

XML *W3C* Дүниежүзілік ғаламтор Консорциумымен ұсынылған. Бұл белгілеу тілі және өзі ортақ синтаксикалық ережелер жинағы болып есептеледі. XML SGML тілінің қысқартылған қосалқы жиынтығы болады.

XML ең маңызды сипаттары келесі:

- XML — құрылымдалынған деректерді бейнелеуге арналған гипермәтінді белгілеу тілі;
- XML анықталу алдындағы тегтерді пайдалануға мүмкіндік беріп қана қоймай, сонымен өзінің меншікті тегтерін де енгізуге жол береді;
- XML платформаға тәуелді емес.

XML тілі, басынан Web-беттерді құру үшін стандартты тіл болған HTML гипермәтінді белгілеу тіліне аналогия құрайтын World Wide Web –те арнайы ақпаратты орналастыру үшін *XML Working Group* консорциумы *W3C* жұмыс тобымен дайындалды. Бұл кеңейтілетін деректерді белгілеу тілі, алайда нақты бір деректерді емес, ал кез келген мәтінді түрде ұсынылатын құрылымдалынған деректерді белгілеуге арналған тіл. XML бағдарламалық өнімдерді дайындау жүйесінде үлкен ролге ие болды және ие болуды жалғастырады. XML негізінде түрлі кеңейтулерді жасақтауға болады – арнаулы белгілеу тілдері. XML стандарты құжаттың құрылымдық элементтері (тегтер) және онымен байланысқан атрибуттар қалай бейнеленуі керектігін ғана регламенттейді. Өзге сөзбен айтқанда, XML құжатты белгілеудің жалпыланған синтаксисін анықтайды. XML ерікті нұсқаулық жиынтығын пайдалана отырып, түрлі типті ақпаратты құрылымдауға мүмкіндік береді. Бүгінгі таңда XML құрылымдалынған ақпаратты қажет ететін кез келген қосымшаларда қолданыла алады – берілетін ақпараттың гигантық көлемі бар күрделі геоақпараттық жүйелерден бастап қызметті ақпаратты бейнелеу үшін пайдаланатын қарапайым бағдарламаларға дейін.

XML пайдалану арқылы шешуге болатын құрымдалынған ақпаратты құру және өңдеумен байланысты көптеген мәселелерді белгілеуге болады.

1. Бұл технология түрлі құрылымдағы ақпараттық ағындармен байланысты қосымшалардың үлкен санымен күрделі ақпараттық жүйелерді дайындаушылар үшін пайдалы бола алады. Бұл жағдайда XML – құжаттары үлкен бағдарламаның бөлек компоненттері арасындағы ақпараттардың алмасуы үшін әмбебап форматы ролін атқарады.
2. XML қажетті ақпаратты іздеу, желілік ресурстар мазмұнын бақылауды қамтамасыз ету, электронды кітапханаларды құрумен және т.б. байланысты Web – тегі көптеген мәселелерді қысқартуға мүмкіндік беретін ресурстарды бейнелеу, RDF (Resource Description Framework — ресурсты бейнелеу ортасы) жаңа тіл үшін базалық стандарт болады. RDF — бұл Дүниежүзілік ғаламтор Консорциумының дайындаған деректерді ұсыну үшін, соның ішінде метадеректерді, абстракталы моделі.
3. XML тілі ерікті типті деректерді бейнелеуге мүмкіндік береді және арнаулы ақпараттарды, мысалы химикалық, математикалық, физикалық формулалар, медициналық рецепттер, ноталық жазбалар, ұсынуға пайдаланылады. Бұл XML –дің Web-те «стандартты емес» ақпаратты тарату үшін HTML –ге толыққанды қосымша ретінде қызмет етуге болатынын білдіреді.
4. XML- құжаттарын үшдеңгейлі жүйелерде деректердің аралық форматы ретінде пайдалануға болады. Әдетте қосымша серверлары және деректер базасы арасында әркеттестік сызбасы нақты СУБД және деректерге қол жеткізу үшін пайдаланылатын SQL диалектісіне тәуелді. Егер де сұраныс нәтижесі кейбір әмбебап мәтінді форматта ұсынылса, онда СУБД буыны қосымша үшін «мөлдір» болады.
5. XML – құжатында қамтылған ақпарат өзгеріледі, клиент машинасына беріледі және бөліктеп жаңғыртылады.
6. XSL стиль кестелерін (eXtensible Stylesheet Language — XML-құжаттарының өзгертумен визуализациялау тілін бейнелеу) пайдалану нақты шығару құралына тәуелсіз XML – құжаттарын бейнелеуге мүмкіндік береді.
7. XML қарапайым қосымшаларда бірыңғай форматта құрылымдалынған деректерді сақтау және өңдеуде пайдаланылады.
8. XML-құжаты құжаттың құрылымы мен оның мазмұнының реттілігі мен тіркемелігі анықтайтын арнайы маркерлер көмегімен деректер элементтері жасалатын жай мәтінді файл.

XML құжатының негізгі қасиеті болып құру және өңдеудің қарапайым тәсілі (жай мәтін кез келген тест процессорымен редакцияланады және стандартты XML-анализаторлармен өңделеді) бола тұрып, компьютерлер жақсы «түсінетін» құрымдалынған ақпаратты құруға мүмкіндік береді. XML – құжатты құру үшін қарапайым жағдайда жай мәтіндеу редакторынан басқа ештеңе қажет болмайды.

XML нақты белгілі типтегі және белгілі қосымшаға қажетті құрылымды құжаттарды құру үшін пайдаланылады. Алайда тілді қолдану сферасы жеткілікті кең болса және ол жасаушылардың көпшілігі тарапынан қызығушылық оятса, онда оның спецификациясы толығымен *W3C* –де қаралуына ұсынылуы және барлық мүдделі жақтардың келісімінен кейін ресми ұсыныс ретінде консорциуммен бекітілуі мүмкін. Тілдің белгілі кемшіліктеріне келесіні жатқызуға болады:

1. XML синтаксисі көп. XML құжатының өлшемі сол деректердің бинарлық түсініктерінен біршама үлкен. Деректер берудің альтернативтік мәтінді форматындағы құжатына қарағанда XML – құжатының өлшемі біршама үлкен, соның ішінде нақты пайдалану жағдайына оптимизацияланған деректер форматындағы құжаттарынан үлкен болады. XML-дің артықтығы қосымшаның тиімділігіне әсерін береді. Деректерді сақтау, өңдеу және беріп отыру құны өседі.
2. Көптеген міндет санын орындау үшін XML синтаксисінің барлық қуаты қажет емес және де едәуір қарапайым өндіруші шешімдерді пайдалануға болады.
3. XML атау кеңістігін пайдалану қиын және оларды XML-парсерларда жүзеге асыру қиын. XML-парсер — XML спецификациясына сәйкес мәтінді құжаттың мазмұнын талдауға арналған бағдарлама.
4. XML тіл ішіне енгізілген деректер типтерін қолдаумен қамтылмаған. Онда «бүтін сандар», «жолдар», «нөлдік мәндер» және т.б. деген түсініктер жоқ.

XML –дің HTML –ден маңызды айырмашылығы – құжатты белгілеуде қаншалықты нақты тіл ережесінің сақталуын бақылауға бөлінетін үлкен зейін. XML құжаты дұрыс құрылған деп саналады, егер ол XML –дің барлық синтаксикалық ережелеріне сәйкес келеді, соның ішінде:

- XML құжаты ішіне барлық қалған элементтерді қамтитын бір жалғыз түпті элементі болады;

■ түпті элементтегі туынды элементтер дұрыс салынуы қажет.

Элемент аттары қарапайым ережелерге бағынады: атауы әріптен, астын сызу белгісі немесе қос нүктеден; бірінші таңбадан кейін атауда әріптер, цифрлар, тасымал белгілері, астын сызу, нүкте немесе қос нүкте; Имена элементов подчиняются простым правилам: имя начинается с буквы, знака подчеркивания или двоеточия; после первого таңбаа в имени могут быть буквы, цифры, знаки переноса, подчеркивания, точка или двоеточие; атауы XML әріп үйлесімінен басталмайды

XML құжатының бірінші жолы XML –ді хабарлау деп аталады. Бұл XML стандарт нұсқасына нұсқайтын міндетті емес жол. Сонымен қатар, бұл жерде таңбалардың кодтары және сыртқы тәуелділіктер көрсетіледі. Комментарий ағаштың кез келген жерінде орналасуы мүмкін. XML-комментарийлар <!—тегтер жұбының ішінде орналасады да --> тегтерінде аяқталады. Дефис екі белгісі (--) комментарий ішіндегі ешқандай бөлімдерде қолданыла алмайды. XML – құжаттың қалған бөлімі кейбіреуі атрибут және мазмұны бар салынған элементтермен қамтылған. Элемент әлбетте мәтін мен басқа элементтерді қоршаған ашатын және жабатын тегтерден тұрады. Элемент мазмұны болып мәтін және басқа енгізілген элементтерді қосқанда ашатын және жабатын тегтердің арасындағының бәрі аталады.

Мазмұннан басқа элементте ашатын тегтің ішіне элементтің атынан кейін қосылатын жұптар «аты =мәні» - атрибуттар да болады. Атрибуттар мәні әрдайым тырнақшаның ішіне алынады (жалаң немесе қос), бір элементте атрибуттың бір аты екі рет кездестірілмейді. Бір тегтің атрибут мәндері үшін тырнақшаның түрлі типтерін пайдалану ұсынылмайды.

Бос элемент деп аталатын мазмұнсыз элементті белгілеу үшін элемент атынан кейін қисық сызықша «/» салынатын бір тегтен тұратын жазбаның ерекше формасын қолдану қажет.

Сонымен, XML тілінде бейнелеу белгілі синтаксистің сақталуымен жазылған операторлар болып табылады. XML-құжатын жасау үшін шектелген белгілі элементтер жинағын пайдаланып қоймайсың, сонымен өзіңнің жеке кез келген атпен атауға болатын элементтеріңді пайдалануға болады.

Демек, XML іс жүзінде кез келген құжатты бейнелеу үшін пайдалануға болады, музыкалық партитурадан бастап деректер базасына дейін. Мысалы, ұсынылған XML – құжатта оқу әдебиетінің тізімі берілген:

```
<?xml   version="1.0"   encoding="Windows-1251"?>
<INVENTORY>
  <BOOK><T1^E>Деректер жинағы</T1^E>
  <AUTHOR^. С. Федорова</AUTHOR>
```

```
  <BINDING> оқу құралы </BINDING>
  <PAGES>298</PAGES>
  <PRICE>250 ру6.</PRICE>
  </BOOK>
 <BOOK>
  <TITLE>Бағдарламалау негізі</TITLE>
  <AUTHOR>Ф. И.Иванов</AUTHOR>
  <BINDING>оқулық</BINDING>
  <PAGES>380</PAGES>
  <PRICE>350 ру6.</PRICE>
  </BOOK>
 <BOOK>
  <TITLE>Ақпараттық 6e3onacHOCTb</TITLE>
  <AUTHOR>E. E.Дремина</AUTHOR>
  <BINDING>y4e6Hое құрал</BINDING>
  <PAGES>253</PAGES>
  <PRICE>400 ру6.</PRICE>
  </BOOK>
 </INVENTORY>
```

Берілген үлгінің бірінші жолында XML версиясы мен пайдаланған кодтауды бейнелейтін XML – декларациясы орналасқан. Келесі жолда құжаттың түпті элементі бейнеленген. Одан кейінгі төрт жолда түпті элементке қатысты (осындай үш топ) төрт туынды элементтер бейнеленген. Соңғы жолда түпті элементтің соңы көрсетіледі.

Алдында суреттелген ережелер XML – құжаттың тек формалды дұрыстығын бақылауға мүмкіндік береді, алайда мазмұнын тексермейді. Екінші мәселені шешу үшін сызбалар пайдаланылады.

Сызба түпкі элементтің оның барлық туынды элементтерінің спецификацияларын қоса аты мен құрылымын айқын анықтайды. Бағдарлама жасаушы қандай элементтер және қандай санда міндетті, ал қандайы міндетті емес екенін тапсыра алады. Сызба, сонымен қатар, қандай элементтердің атрибуттары бар, сол атрибуттардың ұйғарынды мәндері, соның ішінде үнсіз келісім бойынша мәндерін анықтайды.

Сызбаны бейнелеу үшін жиі келесі спецификациялар пайдаланылады:

- DTD (Document Type Definition) —құжат типін анықтау тілі;
- XDR (XML Data Reduced) — диалект XML, *Microsoft* дайындалған*;*
- XSD (XML сызбаларын анықтау тілі) *W3C* консорциумымен ұсынылған.

XML-құжаты HTML-құжатынан Web-браузерде бейнеленуімен де ерекшеленеді. CSS (стилдердің каскадты кестесі) және XSL (XML өзгерту және визуализациялау тілі) пайдаланусыз XML-құжат көптеген Web-браузерлерде қарапайым мәтін болып бейнеленеді. Кейбір Web-браузерлер, Internet Explorer, Mozilla и Firefox сияқты құжат құрылымын тінтуір клавишасын басу көмегімен тораптарды жинақтау және ашу мүмкіндігін беретін ағаш түрінде бейнелейді.

XML-құжатын пайдаланушы үшін түсінікті түрге келесі тәсілдер бар.

1. *Стилдер кестесін XML –құжатымен байластыру* . Стилдер кестесі жеке XML-элементтерді нысандау үшін нұсқаулықтармен қамтылған бөлек файл болады. Я HTML-беттері үшін қолданылатын CSS стилдердің каскадты кестесін, я CSS –ке қарағанда біршама кең мүмкіндіктерге ие, XML-құжаттары үшін арнайы дайындалған XSL стилді кестелер тілі форматындағы кеңейтілетін кестені пайдалануға болады.

2. *Деректерді байланыстыру.* Бұл әдіс HTML- беттерін құруды, онымен XML-құжатты байланыстыру және бетте XML элементтерімен SPAN или TABLE сияқты стандартты HTML-элементтерінің өзара әрекеттестігін орнатуды талап етеді. Алдағы уақытта HTML – элементтері онымен байланысқан XML-элементтерінен автоматты түрде ақпаратты бейнелейді.

3. *Сценарийді жазу* — жүйе орындайтын іс – әрекеттің қысқа сипаттамасы. Құрылған HTML-бет XML – құжатымен сценарийдің (JavaScript, VBScript) арнайы жазылған коды арқылы байланысады. Браузер XML-құжатты үлкен объекті жинағы, ерекшелігі мен командалардан тұратын құжаттың объектті моделі (Document Object Model — DOM) ретінде қабылдайды. Жазылған код XML – элементтеріне қол жеткізу, бейнелеу және манипуляциялауға мүмкіндік береді.

XML өз қолданысын HTML және XML XHTML комбинациясын білдіретін (Extensible Hypertext Markup Language) тілінде тапты. XHTML қолданыста XML –тің қатал синтаксисімен үйлесімді HTML тегтерін пайдаланылады. Кейде XHTML-ді HTML –дің қаталдау версиясы.

XHTML өз қатарында HTML сияқты SGML тілінің қосалқы бөлімі, алайда XHTML алдыңғысына қарағанда XML спецификациясына сәйкес келеді. XHTML HTML-дің ізін басушы. HTML –дің біраз қатал версия қажеттілігі бүгінгі күні Web-беттердің мазмұны жиі ресурстары шектелген, соның ішінде оралымды, көп талап қоймайтын HTML-ды өңдейтін (тіл синтаксисі еркін болған сайын, оны талдау қиындайды) құралдардың дәстүрден тыс түрлеріне (мысалы, ұялы телефондар) бағытталғанына байланысты пайда болды.

Қазіргі уақытта XHTML –дің екі спецификациясы бар: XHTML 1.0 және XHTML 1.1. Іс жүзінде барлық заманауи браузерлер XHTML –ді қолдайды. Ол ескі браузерлермен де үйлесімді, өйткені XHTML негізінде HTML жатыр. XHTML –дің дамуы тоқтатылды, XHTML –дің жаңа версиялары шықпайды. XHTML 2.0 соңғы версиясы болып саналады. Бұл версия ұсыну статусына жеткен жоқ: XHTML 2.0 жұмыс тобының қызметі 2010 жылдың аяғында тоқтатылды, ал барлық ресурстар HTML5 жұмыс тобына ауыстырылды.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. Құжаттарды белгілеу тілінің ерекшелігі неде?
2. Дайындаушыларға HTML қандай құралдарды ұсынады?
3. Белгілеу тілінде тег не анықтайды?
4. Браузерлер HTML-де қате жазбаға орай қалай әрекеттенеді?
5. HTML-құжаты қандай тегтен басталады?
6. HTML –дің кемшіліктерін атап беріңіз.
7. Стилдердің каскадты кестелері не үшін пайдаланылады?
8. CSS –те стиль болып не аталады?
9. Стилдер кестесі деп не аталады?
10. Селектор дегеніміз не?
11. Қандай тәсілдермен стилдерді анықтауға болады, атап беріңіз.
12. Стилдер кестесін HTML – құжатымен қалай байланыстыруға болады?
13. CSS3-тің ерекшелігі неде?
14. XML не үшін арналған?
15. XML –дың ең маңызды белгілерін атап беріңіз.
16. Қандай міндеттерді шешуде XML –ді пайдалануға болады?
17. XML тілінің кемшілігіне нені жатқызуға болады?
18. XML-құжатты құру үшін өзіңнің жеке элементтеріңді пайдалануға бола ма?
19. XML-құжатты пайдаланушыға түсінікті түрге ауыстыру тәсілдерін атаңыз.
20. Қандай жағдайда XML –құжат дұрыс құрылды деп саналады?
21. Неліктен XHTML тілінің дамуы тоқталды, түсіндіріңіз.

# 7 тарау

# КЛИЕНТТІК БЕЛСЕНДІЛІКТІ ЖҮЗЕГЕ АСЫРУ

## 7.1. ҚҰЖАТТЫҢ НЫСАНДЫҚ МОДЕЛІ

Web-бағдарламалауды шартты екі бөлімге бөлуге болады: клиенттік және серверлік. Клиенттік Web-бағдарламалау клиенттік қосымша – браузерлерде беттерді бейнелеу міндеттерін шешеді. Клиенттік бағдарламалаудың негізгі аспаптары HTML, JavaScript, VBScript тілдері, CSS стилдері, сонымен қатар, Java-апплеттері және ActiveX-компоненттері. Заманауи браузерлерге басынан енгізіліп салынған қолжетімді құралдар жинағы саны әрдайым көбейіп отыр. Қалау бойынша клиент жағына XSL-стилдерін пайдаланумен XML-құжатты өңдеуді ауыстырып салуға болады.

Беттерді генерациялауда Web –те «клиент-сервер» архитектурасымен байланысты дилемма пайда болады. Беттерді клиент жағында да, солай сервер жағында дда генерациялауға болады. Клиенттік белсенділікті жүзеге асыру үшін JavaScript, VB Script сценарийін, Java апплетін, ActiveX басқару элементтерін және басқа технологияларды қолдану мүмкін болады.

*Құжаттың объектік моделі* (Document Object Model — DOM) Web-консорциумы ұсынған стандарт болады және құжат ішіндегі мазмұнын (соның ішінде, Web-беттерді) объектілер жинағы түрінде көрсету тәсілі.Ішіндегі мазмұны деп Web-беттегі барлығы: суреттер, сілтемелер, абзацтар, мәтін және т.б. түсініледі.

Әр браузер үшін бірегей ерекше болатын браузердің объектік моделінен айырмашылығы құжаттың объектік моделі стандарт болып, барлық браузерлермен қолданылуы керек. Тәжірибеде DOM –ды қолдау толыққанды жүзеге аспаса да, бұл стандарт талаптарын браузер дайындаушылар да, Web-сайт дайындаушылары да ұстануға ынталану қажет.

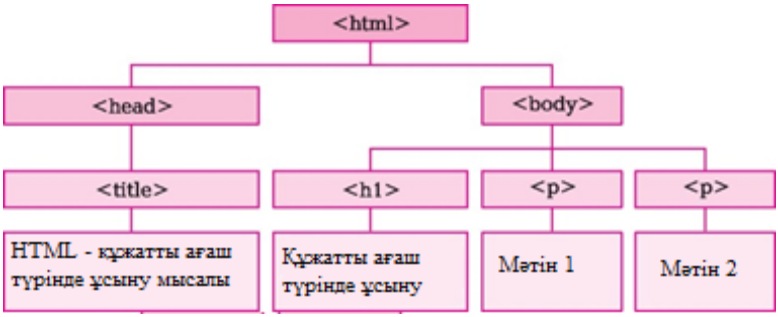
Атап кетерлік, DOM тек Web- беттерде ғана емес, сонымен кез келген басқа құжаттарда қолданыла алады. Соның ішінде, ол XML кез келген сөздігімен, және де осындай сөздіктердің бірі XHTML болады.

DOM-да құжат бағдарламалау жүйесінде жиі қолданылатын құрылым ағаш келбеті түрінде ұсынылады (7.1 суреті). Бұл құжат бойынша бірыңғайланған навигация тәсілін қамтамасыз етеді. DOM құрылымы кез келген объектік –бағытталған тіл иерархиясына ұқсаған объектілер иерархиясын ұсынады. DOM бағдарламалау ешбір нақты тіліне байланбаған өзінің пайдалы API ұсынады. Әрі қарай 7.1 суретінде ұсынылған ағаш бейнелі құрылым сәйкес келетін HTML-код беріледі:

```
<html>
<head>
^y^e> HTML-құжатын ағаш түрінде бейнелеу үлгісі
</tytle>
</head>
<body>
^1> құжатты ағаш түрінде бейнелеу үлгісі <^1>
<p>1 мәтін</p>
^>2 мәтін<^>
</body>
</html>
```

DOM моделіне сәйкес:

- Барлық құжат құжаттың буыны болып ұсынылады;
- Әр HTML-тегі элемент буыны болады;




7.1-сурет. HTML-құжаттың ағашбейнелі құрылым түріндегі ұсынылуы

- HTML-элементтері ішіндегі мәтін мәтінді буындар болып ұсынылады;
- әр HTML-атрибутқа атрибут буыны сәйкес келеді;
- комментариилер комментарий буыны болады.

Алдында берілген үлгіде түпкі буын <html>. Тегі болды. Қалған барлық буындар <html> ішіне салынған. Бұл буынның <head> и <body>. екі туынды буыны бар. <head> буынында <title> буыны, ал <body> буынында – <h1> и <p> буындары бар.

HTML-құжаттың барлық буындары ағаш құрылымы арқылы қолжетімді болады, сонымен бірге оның ішіндегі өзгертілуі және жойылуы мүмкін және жаңа элементтерді қосуға болады. Ағаштың барлық буындары өзара иерархиялық қатынаста болады. Бұл қатынастарды суреттеу үшін терминдер пайдаланылады: «ата-ана», «туынды элемент» және «жаңа буын». Аналық буындардың туынды буындары бар, ал бір деңгейдің туынды элементтерін «жаңа буын» деп атайды (ағалар немесе сіңілдері).

Ағаш буындарына қатысты келесі қағидалар сақталады:

- ағаштың ең жоғары буыны түпті деп аталады;
- әр буынның, түпті буыннан басқа, тура бір аналық буыны бар;
- буын қанша болса да туынды буынды болуы мүмкін;
- ағаштың соңғы буынының туындысы болмайды;
- жаңа буындар ортақ аналы болады.

DOM — бұл құжат ішіндегі мазмұнға әмбебап жету жолының (платформалар мен бағдарламаларға тәуелсіз) спецификациясы және оларды өңдеушілер үшін жай ғана өзіндік API болады. DOM — қажетті фрагменттерді іздеу, жасау, оның элементтерін жою мен модификациялау көмегін беретін кез келген HTML объектік моделі немесе XML –құжатты құрудың стандартты тәсілі.

DOM –ның бағдарламалық интерфейсі өз бойына стандартты қабілеттер мен әдістер жинағын енгізген. Қабілеттер кейбір мәндерді, ал әдістер – олармен әрекеттерді ұсынады. DOM спецификациясында құжат ішіндегі мазмұнға жету интерфейсын бейнелеу үшін IDL тілі қолданылады және пайдалану үшін оларды қандай да болсын бағдарламалаудың нақты бір тіліне «аудару» қажет. Алайда бұнымен анализаторлардың өздерін құрушылар айналысады, қолданушы бағдарламашылар интерфейстерді жүзеге асыру тәсілдері туралы ештеңе білмеуіне болады - DOM қолданушы бағдарламаларды дайындаушылар тұрғысынан белгілі әдістер мен қабілеттеріне ие объектілер жинағы болып көрінеді.

Құжаттың объектік моделі беттің барлық элементтерін бағдарланған объектілерімен жасайды. Оның көмегімен сценарий тілдері арқылы құжатқа қолжетімділік алу және барлық элементтерін басқаруға болады. HTML-дің әр элементі жеке объект ретінде қолжетімді, демек, HTML-бетінің кез келген тегі мәнін өзгертуге болады, және құжат серпінді бола бастайды. Пайдаланушының кез келген әрекеті сценарий процедурасымен ұстап алынуы және өңделуі мүмкін болатын оқиға ретінде түсіндіріледі.

DOM HTML- құжаты үшін бетті қайта жүктеусіз өз мазмұнын серпінді өзгерту мүмкіндігін қамтамасыз етеді. Серпінді HTML (Dynamic HTML или DHTML) бетті белгілеу тілі болмайды. Бұл мазмұны серпінді түрде өзгеретін HTML-бетті белгілеу үшін қолданылатын термин. Реализация DHTML жүзеге асыру HTML, стилдердің каскадты кестелері (Cascade Style Sheets — CSS) және сценарийлер тіліне (JavaScript или VBScript) негізделеді. Бұл DHTML-дің үш компоненті өзара DOM құжатының объектік моделімен байланысқан.

## 7.2. HTML5

HTML –дің бесінші версия стандарты 2014 жылы аяқталған, сол уақытта пайдалануға ұсынылған версиясы шықты. HTML –дің бесінші нұсқасын дайындау мақсаты мультимедиатехнологияларын қолдау деңгейін жақсарту болды. HTML5 — бұл Web-индустрияны дамытудағы жаңа версия деп айтуға болады. HTML5 тілінің спецификациясын құруда ерекше назарда болған негізгі моменттерін қарастырайық.

HTML5 құраушының маңызды моментінің бірі семантика болады, әр тег өзінің мәндік жүктемесіне ие. HTML5-де жасалған сайттарды талдаушы бағдарламалар тегтер арасында қандай деректер орналасқан, олардың мәнділігі қандай екенін түсінуі қажет. Жаңа тегтер пайда болды, мысалы, семантикалық бөлінуі үшін сайт бетін жоғарғы жаққа, тқменгі, жанындағы, негізгі контент (<section>, <article>, <header>, <nav>). HTML 4.01- те қолдануға болатын кейбір ескірген элементтер, таза рәсімдеу элементтерін қоса, <font> и  сияқты, қолданыстан шықты, өйткені эффекттер стилдердің каскадты кестелері (CSS) көмегімен жасалатын болды. Кейбір элементтер, мысалы, <a>, <menu> ЖӘНЕ <cite>, өзгертілді, қайта орналастырылды немесе стандартталды. Бірақ та HTML5 алдыңғы қатарлы XHTML, HTML4 и 3 белгілеу тілдерімен толық үйлесімділікті сақтайды.

Тілдің плагиннен (серпінді қосылатын модулдер) тәуелсіз болуына үлкен көңіл бөлінеді. Музыка, видео, анимация, ойындар, осының бәрімен компьютер және браузерге өзге бағдарламаларды жүктемей жұмыс жасауға болады. <video>, <audio> и <canvas> элементтері, сонымен қатар, SVG және математикалық формулаларды пайдалану мүмкіндігі өзге API –ді пайдалану қажеттілігін жойып, Торда графикалық және мультимедиялық объектілерді құру мен басқаруды қысқарту үшін жасақталған. Енді кодта бұрын XML бөлігі болған сол спецификацияларды пайдалануға болады, мысалы HTML-құжатының кодында түрлі векторлық фигуралар жасайтын SVG спецификациясын пайдалануға болады. Соның ішінде, <canvas> тегі векторлық графиканы жасау, оны анимациялау, тіпті ойын жасауға мүмкіндік береді. API (қосалқы бағдарламаның интерфейсі) және DOM (құжаттың объектік моделі) HTML5 спецификациясының фундаменталдық бөліктері болып табылады. HTML5 спецификациясында DOM технологиясын пайдалана отырып, беттің элементтері JavaScript бағдарламалау тілімен өзара қалай әрекеттену керектігі белгіленген, өйткені HTML5-де тегтер толыққанды объектілер болып саналады. Сонымен қатар, спецификацияға браузерлердің HTML-кодта кездестірілетін қателерді қалай өңдеу қажеттілігі сипаттамасы да қосылған (браузер дайындаушылар үшін бұл маңызды). Бұған дейін браузер дайындаушыларда қатесіз жасалған дұрыс кодты қалай өңдеу сипаттамасы ғана болған.

HTML-құжатының код құрылымы аздап өзгерген, келесі жолдың орнына

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//RU">
```

```
//
қалғаны
```

```
<!DOCTYPE html>.
```

HTML5 белгілеу анықтамасына қосымша қосалқы бағдарламалаудың скриптік интерфейсін (API) бекітеді. DOM құжатының объектік моделі (Web-беттерін объект түрінде ұсыну тәсілі) бар интерфейсі кеңейтілген, сонымен ерекшеліктері факт түрінде тіркелген. Cookies файлдарына қарағанда локалды деректермен манипуляциялар көбірек жасау мүмкіндігін беретін оларды сақтау бойынша жаңа технологиялар пайда болды, өйткені DOM-сақтау қоймалары - cookieке қарағанда артық функционалдық альтернативаны құрайды.

Сонымен қатар, жаңа API бар, мысалы: 2D-да ерікті сурет салу әдісі үшін элемент-кенеп; медиафайлдардың ойнауын бақылауда,соның ішінде видеодан субтитрларды синхронизациялау үшін пайдалану; құжатты бетке таңдау арқылы жүктеу жолымен (тег <input type="file">) немесе тасып алу жолымен редакциялау; MIME типі (электрондық пошта арқылы деректерді беру) және хаттама өңдеушіні тіркеу және басқа.

Барлық заманауи браузерлер HTML5 тілін қолдауда. *W3C* ұйымдастырушылары Web-дайындаушыларын CSS3-бен бірге HTML5-ді пайдалануға шақырады. CSS3 спецификациясы HTML5 спецификациясының бөлігі болып табылмайды. Бұл екі стандарт бір – бірінен бөлек, әр түрлі уақытта түрлі жерде жұмыс жүргізген бөлек адамдармен жасақталған. Алайда HTML5 және CSS3 заманауи Web-дизайнның бір жаңа толқынының бөліктері.

## 7.3. JAVASCRIPT КЛИЕНТТІК СЦЕНАРИЙЛЕРІ

HTML мәтінді оқу, суреттерді тамашалау, басқа Web-беттерге ауысу және т.б. мүмкіндіктерін береді. Бірақ HTML –де келесі мүмкіндік қатары қарастырылмаған: ол пайдаланушының форманы дұрыс толтырғанын талдай алмайды, сайтқа кірушімен диалог негізінде шешім қаббылдауға қабілетсіз. 1995 жылы *Netscape* компания мамандары JavaScript бағдарламалау тілін жасақтап, клиенттік жағында беттерді басқару механизмін құрды. Бұл клиент жағында да, сервер жағында да орындалатын енгізілетін қосымшалардың сценариялық, объектік – бағытталған тілі. JavaScript кең қолданысын браузерлерде сценарий тілі ретінде Web-беттерге интербелсенділік беру үшін табады.

Бағдарламалау тілдері үлкен және күрделі бағдарламалық кешендерді жасау үшін пайдаланылады. Сценарийді жазу тілдері Web-серверде немесе браузерде Web-буынның функциясын орындаушы сценарий немесе скрипт деп аталатын шектеулі мүмкіндіктерге ие бағдарламаларын құру үшін арналған. Күрделі бағдарламалау тілінен сценарийді жазу тілдерінің айырмашылығы келесімен түсіндіріледі: нұсқамаларды ретімен команда интерпретаторы деп аталатын аралық бағдарламалар орындайды. Интерпретация орындау тиімділігін азайтса да, сценарий жазу тілдері игеру үшін қарапайым және көптеген мүмкіндіктерді қамтамасыз етеді. Сценариилер HTML-бетіне ішіндегі мазмұнды нысандау үшін енгізілген болады немесе өзінде бизнес – логикамен қамтылған COM компоненттерін жүзеге асыра алады.

Клиенттік қосымшалар браузермен пайдаланушының машинасында орындалады, серверлік қосымшалар – серверде. Серверлік те, клиенттік қосымшаларды да дайындауда ядро деп аталатын және өзінде стандартты объектілер мен конструкциялар анықтамаларын, және тіл қосымшаларына сәйкес компоненттер, әр қосымша типі үшін объектілердің спецификалық анықтамаларын енгізген тілдің ортақ компоненті пайдаланылады. JavaScript сценарийлер тілі конструкциясы браузерлер басқарумен беттерді жүктеген кезде, сонымен қатар, пайдаланушының сол беттерде орналасқан объектілермен белгілі әрекеттерді орындағанда HTML беттеріне енеді және орындалады (түсіндіріледі).

JavaScript сценарийлерін пайдалана отырып, серпінді түрде өзгеріп отыратын объектілер жасауға болады. JavaScript сценарийлері пайдаланушының HTML формалары аясын толтыру нақтылығын олар серверге берілгенше жедел тексеруді қамтамасыз етеді. JavaScript сценариилері пайдаланушының форма аясына енгізген деректерін, сонымен қатар, пайдаланушының тінтуірмен манипуляцилары үдерісінде пайда болатын оқиғаларды өңдеуге, браузер терезесіне HTML-дің басқа беттерін көшіруге немесе жүктелген беттердің ішіндегі мазмұнын өзгертуге қабілетті.

JavaScript сценарийлері көмегімен локалды міндеттерді шешуде пайдаланушымен өзара әрекеттестік жүзеге асады. Соның ішінде, JavaScript сценарийлері түрлі визуалдық эффекттерді жасауда кең ауқымда қолданылады. Мысалы, үстінде тінтуір курсоры орналасқан басқару элементтерінің сыртқы түрін өзгерту, графикалық суреттерді анимациялау, дыбыстық эффекттер жасау. Cookie локалды естің механизмі JavaScript сценарийлеріне компьютерде пайдаланушының енгізген локалды ақпаратын (мысалы, Cookie-де сатып алуға таңдалған тауарлар тізімін сақтауға болады) сақтауға мүмкіндік береді.

JavaScript қосымшасы браузерге енгізілген интерпретатормен ретімен өңделетін тіл операторлар жинағы болып ұсынылады. JavaScript тілінде
C + + немесе Java-да жүзеге асқан түрінде класстармен жұмыстану үшін құралдар қарастырылмаған. Сценарий дайындаушы бар болатын JavaScript класы негізінде қосымша класс құру, әдісті қайта анықтау немесе класпен қандай да болсын басқа операциялар орындай алмайды. JavaScript тілінде жазылған сценарийге негізінде тек дайын объектілер қолжетімді. Жаңа объектіні құруды орындау сирек жағдайда болады. Объекті браузер терезесінде бейнеленген HTML – құжат, браузер терезесі, формалар, суреттер, гипермәтінді сілтемелер және басқа да Web-беттің компоненттері болады.

JavaScript жұмысы үдерісінде сценарий осы объектілерге қарасады, ақпарат алады және оларды басқарады.

Бұдан басқа JavaScript тілінде сценарий дайындаушыға HTML – құжатымен тура байланыспаған объектілер қолжетімді. Оларды алдын ала анықталған немесе тәуелсіз объектілер деп атайды. Осы объектілер көмегімен ауқымды жүзеге асыру, күні мен уақытын суреттеу, математикалық есептеулерді орындау және кейбір басқа есептерді шешуге болады.

JavaScript сценарийін HTML- бетіне келесі тәсілдермен енгізуге болады.

1. <script>...</script> HTML тегі-контейнері ішінде тіл операторының тапсырмасы. HTML –дің қандай да болсын тегтерін қолдамайтын браузерлер оларды ескермейді, өткізіп жіберетін тегтер ішіндегі мазмұнды HTML синтаксисі тұрғысынан талдайды. Бұл бетті бейнелеуде қателерге әкелуі мүмкін. Сондықтан JavaScript тілі операторлары комментарий контейнеріне орналасады <!— ... —>:

```
<script type =
"text/javascript"
language="javascript">
<! —
JavaScript операторлары
//-->
</script>
```

// таңбалары --> комментарийдің жабатын тегі алдында JavaScript комментарий операторы болады. Ол интерпретатордың дұрыс жұмысы үшін қажет. Құжатта кез келген жерде орналасқан бірнеше <script> тегтері болады. Олар бәрі ретімен браузер терезесінде құжат бөліктерінің бейнеленуі бойынша JavaScript интерпретаторымен өңделеді. Сондықтан глобалды функциялары мен өзгермелімен сценарийлер құжаттың <head> бөлімінде орналастыру ұсынылады. Бұл жағдайда барлық анықтамалар интерпретатормен құжатты жүктеу басында өңделеді және құжаттың браузер терезесінде бейнеленуінің бастапқы моментінен бастап есте сақталады.

2. <script>тегінде src параметрінде JavaScript кодымен файлды көрсету. Параметр мәні ретінде JavaScript тілінде бағдарламалық коды бар сыртқы файлдың толық немесе қатыстық URL –адресі беріледі. </ script> жабатын тегтің тапсырмасы тег ішінде операторлар берілді ме әлде жоқ па тәуелсіз міндетті.

Байланысатын сыртқы файлда HTML тегтері болмайды және ол js кеңейту болуы керек:

```
   < s c r i p t
t y p e = " t e x t / j a v a s c r i p t "
language="javascript"
   src="http:www. my site. ru/func/jsfunc. js">
   операторлар JavaScript
   </script>
```

3. JavaScript өрнегін HTML тегтері параметрлер мәні ретінде пайдалану. JavaScript элементтері солай & амперсанд пен нүктелі үтір арасында орналасады, бірақ фигуралық жақшалар { } ішіне салынып, тек HTML тегтері параметрлер мәні ретінде пайдалануы керек. JavaScript элементтерін HTML мәтінінде пайдалануға болмайды. Олар параметрден оң жақта орналасып, оның мәнін бергенде ғана түзетіледі. Мейлі brw айнымалы шама анықталған, және оған 50 мәні берілген. Келесі тег браузер терезесінің көлденең өлшемінен 50 % ұзындықтағы көлденең сызықты салады:

```
   hr width="&{brw};%"/>
```

HTML тегінде оқиға өңдеушілерін анықтау. Сценарий тілдерімен үйлесімділігі үшін HTMLтегтерінің кейбіреуіне пайда болатын оқиғаларды өңдеудің арнайы параметрлері енгізілді. Бұл параметрлер мәні JavaScript тілінің операторлары болуы мүмкін. Әдетте мәні ретінде шақырылатын оқиға өңдеудің параметрлерімен анықталатын сәйкес оқиға болған кезде функция аты беріледі. Параметр аты on приставкасынан басталады, одан кейін оқиғаның өзінің аты жүреді. Мысалы, Click оқиғасының өңдеу параметрі onClick атын иемденеді. Оқиғалар негізінде пайдаланушының HTML форма элементтерімен жасалған іс – әрекетіне байланысты. Сондықтан, бәрінен жиі оқиғаларды ұстап алу және өңдеу форма элементтері параметрлерінде беріледі, бұл енгізілген ақпаратты өңдеуге жіберер алдында тексеруге мүмкіндік береді.

Міне, сценарийден функцияны айқын шақыру үлгісі:

```
   DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN"
   "http://www.w3.org/TR/html4/loose. dtd">
   <html>
   <head>
    <title>Demo</title>
```

```
    < s c r i p t t y p e = " t e x t  /
j a v a s c r i p t "  language="javascript">
    <!-- // JavaScript  қолдамайтын браузерлерден
жасыру function proverka(form) { if(form. value
>=18)
    {alert("Bbi — кәмелетке толған");}
    else {alert ("Сіз – кәмелетке толмаған");}
    }
    //-->
    </script>
    </head>
  <body>
  <form name="form l">
  Сіздің жасыңыз: <input type="text" size="5"
name="age" /> <hr/>
  <input type="button" value="Проверить" onclick="
  proverka(this. form.age)"/>
    </form>
  </body>
  </html>
```

Бұл үлгіде onclick батырма формасы оқиғаны өңдеуші age мәтінді алаңға енгізілген мәні берілетін validate функциясын шақырумен байланысты. Алаң аты name параметрімен беріледі. Батырмаға басқан кезде енгізілген мәнге байланысты хабарламаны бейнелейтін validate функциясы шақырылады.

Форманың name элементі параметрі сценарий операторларында сәйкес элементке сілтеу үшін пайдалануға болатын элементтің таңбаикалық атын береді. Берілетін this. form. age функция параметрі форманың age (мәтінді алаң) атаулы элементін білдіретін объектіге бағытталған тілдер синтаксисін пайдаланады. JavaScript тілінің this негізгі сөзі осы жағдайда ағымдағы формаға сілтемені білдіреді.

## 7.4. AJAX ТЕХНОЛОГИЯСЫ

Барлық Web – қосымшалар HTTP хаттамасы негізінде жұмыс жасағандықтан, клиент пен сервер арасындағы өзара қатынас «сауал — жауап» үдерісі негізінде жүзеге асады. Және бұл жерде беттің ішіндегі мазмұны браузерде осындай үдеріс жүрген сайын толығымен қайта жүктеледі, бұл клиенттік бағдарламалық қамтамасыз етілуіне қосымша жүктеме болады. Бұл мәселені шешу үшін AJAX тәсілдемесін пайдалануға болады.

Бұл жағдайда бет толығымен жүктелмейді, тек оның бөлігі жүктеледі. AJAX — бұл HTML, CSS, JavaScript және DOM –ның бірлескен қызметінің тиімді тәсілі. AJAX пайдалану көмегімен интерфейс реакциясының жылдамдығы біраз көбейтуге және серверге түсетін жүктемені біраз азайтуға болады. Бұл ақпаратпен асинхронды алмасуына және бетті толығымен жүктемей, тек «жаңғыртылған» бөлігін қайта жүктеу қабілетіне байланысты мүмкін болады. AJAX тәсілдемесін пайдалануда бет толығымен тек бірінші рет жүгінгенде ғана жүктеледі. Одан кейінгі әрекеттестік операциялары асинхронно JavaScript клиенттік сценарийлерден Web-серверге жүгіну арқылы өндіріледі. AJAX келесі көптеген белгілі Web-қосымшаларымен пайдаланылады: Facebook, Flickr, Gmail, Google Maps и Youtube.

*AJAX (Asynchronous JavaScript and XML)* — бұл интерактивтік қосымшаларды дайындауға бағытталған, пайдаланушы әрекетіне орай шұғыл әрекет жасайтын, клиент жағында жұмыстың қомақты бөлігін атқаратын және бетті ауыстырудың орнына шұғыл жаңғыртатын сұраныс арқылы сервермен өзара әрекеттестіктегі бірнеше шектес технологияларын пайдалану концепциясы.

Асинхронды сұраныс — бұл серверден жауапты күтусіз берілген сұраныс. Аталған сұраныстарды өңдеу нәтижесінде клиент пен сервер арасында деректер XML форматында беріле алады.

Сондықтан да бұл тәсілдеме AJAX деп аталды — Asynchronous Javascript and XML.

AJAX тәсілдеме базасында қосымшалар жұмысы атақты браузерлер көпшілігінде JavaScript бағдарламалық интерфейстер құрамында XMLHttpRequest объектісінің пайда болу нәтижесінде мүмкін болды. Дәл осы объекті арқасында JavaScript клиент коды клиент кодынан асинхронды сұранысты орындауға қабілетті. Серверден жауап алғаннан кейін JavaScript клиенттік коды HTML белгілеудің ішіндегі мазмұнды нәтижені бейнелеу мақсатында серпінді түрде өзгертеді.

Web-қосымшаларды дайындаудың заманауи әдістері AJAX-әрекеттестігі нәтижесінде серверден клиентке берілетін деректердің түрі әртүрлі болуына әкелді. Аталған факт бұл деректердің беттегі бейнелеу әдісін анықтайды.

*XMLHttpRequest (XMLHTTP, XHR)* — браузерлердің скриптік тілдеріне қолжетімді, Web-серверге тікелей HTTP немесе HTTPS сұраныстарын пайдаланушы және сервер жауабы деректерді шақырылған скриптке жүктейтін қосымша пайдаланушылардың интерфейсі (API).

Атауында XML аббревиатурасы болса да технология берілетін деректердің форматына шектеулер қоймайды. Ақпарат кез келген мәтінді форматта беріледі, мысалы XML, HTML немесе жай құрылымдалмаған мәтінмен беріледі. XMLHTTP AJAX технологиясының маңызды құрамды бөлігі болып табылады, көптеген сайттармен серпінді, пайдаланушының сұранысына тез әрекет ететін қосымша ретінде пайдаланылады. XMLHTTP тек XMLHTTP беті пайдаланатын сол доменде орналасқан файлдармен жұмыс жүргізеді, алайда шектеуді бұзу мүмкіндігі бар.

Келесі AJAX-әрекеттестік түрлері бар:

- сервер клиентке бейнеленуге қажет беттің HTML-кодының фрагментін береді. Бұл жағдайда клиентік сценариилер JavaScript сервермен әрекеттестік үдерісі аяқталғаннан кейін бар құжат құрылымына сервермен өзара әрекеттестікте алынған сол HTML-кодты салады;
- сервер клиентке XML форматында деректер береді. Бұл жағдайда клиентік код JavaScript бар құжат құрылымына салынатын, HTML белгілеуінде сол деректерді өзгертуге мүмкіндік беретін логикаға ие болу керек;
- сервер деректерді JSON форматында береді (JavaScript Object Notation — JavaScript негізделген деректер алмасуының мәтінді форматы) JSON форматы AJAX –пен XML орнына пайдаланылады. Жалпы алғанда бұл тәсілдеме деректерді беру форматының айырмашылығы қоспағанда алдыңғыға аналогия құрайды.

HTML-белгілеу емес, тек деректер берілетін тәсіл ақпаратты беру жылдамдығы көзінен тиімдірек, алайда оны жүзеге асыру JavaScript тілінде кейбір клиенттік кодты дайындауды талап етеді

AJAX тәсілдемесін пайдалану келесі артықшылықтарды ұсынады:

- трафикті үнемдеу — AJAX-әрекеттестігі үдерісінде беттер тұтасымен берілмейді, тек оның қажетті бөлімдері ғана беріледі, бұл Торға түсетін жүктемені төмендетеді және трафикті үнемдейді;
- серверге түсетін жүктемені азайту — әр AJAX-әрекеттестікте серверге беттің тұтас HTML-кодын генерациялау қажет емес, тек беттің бөлек бөлігін генерациялау жеткілікті;
- интерактивтік әрекеттестік мүмкіншілігі — жүктелгеннен кейін бет толығымен қайта жүктелмейді, бұл ыңғайлы және интерактивті пайдаланушының интерфейсін құрады.

Алайда AJAX тәсілдемесін пайдалануда кемшіліктер де орын алады:

- браузерлер, әдеттегідей, деректердің жүктелуін AJAX-әрекеттестік арқылы қадағаламайды. Бұл ерекшелік қосымшамен өзара әрекеттестікте қосымша ыңғайсыздық жасауы мүмкін;
- серпінді түрде жүктелетін ішіндегі мазмұн іздеу жүйелерімен индексацияланбайды. Бұл іздеу машиналарының JavaScript сценарийлерін қоспау себебінен болады. Сондықтан, іздеу машиналарының бет ішіндегісін индексациялауы талап етілсе, бұл туралы бөлек ойластыру керек.

Жалпы алғанда AJAX негізіндегі тәсілдеме – бүгінгі таңда көптеген Web-қосымшалар пайдаланатын жеткілікті перспективалық бағыт. AJAX- қосымшаны құру үшін көптеген дайын кітапханалар жабдықталған, оларды қосымша дайындау барысында пайдалануға болады. ASP. NET технологиясында да AJAX-қосымшаларын жасау үшін бағдарламалық компоненттер қатары бар.

## 7.5. VBSCRIPT КЛИЕНТТІК СЦЕНАРИЙЛЕРІ

Visual Basic Scripting Edition (немесе жай VBScript) — бұл сценарийлер (скрипттер) жасау үшін арналған *Microsoft* компаниясынан бағдарламалау тілі. Ол Visual Basic тілінің қосалқы жиынтығы және Windows жүйесінде административтік сценарийлерді құруда кең пайдаланылады. VBScript үнсіз келісім бойынша Windows Script Host (WSH) та қолданылады, ал ол өз кезеңінде үнсіз келісім бойынша кез келген Windows версиясымен қондырылады.

VBScript синтаксисі Visual Basic стандартты синтаксисінің біркелкі қысқартылған версиясы болып табылады. Мысалы, VBScript – те типизациялау қолданылмайды: барлық өзгермелілер Variant типінде болады.

Сценарий құруда оны қолданудың артықшылығы болып оны Visual Basic және Visual Basic for Application тілдерінде бұрын жазылған процедураларды шағын түзетулермен қолдану мүмкіндігі.

VBScript клиент сценарийлерін жазу үшін пайдаланылады (бұл жағдайда браузерде сол тілдің енгізілген интерпретаторы болуы керек), сонымен қатар, сервер сценарийін жазу үшін (бұл жағдайда сервер VBScript тілін қолдауы қажет). Клиент сценарийлерін жазу үшін JavaScript объектілер жинағына аналогия құрайтын объектілер жинағы пайдаланылады.

Web-серверлер беттерін құруүшін Интернетте орналасқан VBScript қолдану орынсыз, өйткені IE –ні қолданатын пайдаланушылар барлығы емес. Алайда егер Интернет технологиясы корпоративтік интраторда қолданылса және IE –ні барлық пайдаланушылардың компьютеріне қондырса, онда бағдарлама жасаушылардың JavaScript –тің орнына VBScript қолдануы дайындаудың мерзімі мен құнын біраз қысқартады.

## 7.6. JAVA ТЕХНОЛОГИЯСЫ

Java технологиясы өзінде бағдарламалау тілі мен платформаны қамтиды.

*Java — Sun Microsystems* (соңынан *Oracle* компаниясы иелеген) компаниясы дайындаған бағдарламалаудың объектік – бағытталған тілі. Java тілі сәулеттік тұрғыдан тәуелсіз, жоғары өнімді, жоғарғы деңгейлі бағдарламлаудың серпінді тілі болып табылады. Java –да бағдарламалар Java виртуалды машиналары орындайтын байт-кодта трансляцияланады. Java (JVM) виртуалды машинасы Java-бағдарламасының түпкі мәтінінен алдын ала жасалған Java компиляторымен Java байт кодын өңдейді де құралға нұсқаманы береді. Осылай, Java виртуалды машинасы интерпретатор сияқты қызмет атқарады. JVM, Java Runtime Environment (JRE) деп аталатын, Java жүйесінің негізгі орындаушы бөлігі болады. Байт-код Java (сокр. J-код) — бұл Java виртуалды машинасы орындайтын нұсқаулықтар жиынтығы. Байт-коды операциясының әр коды – бір байт. JVM бағдарламалаудың басқа тілдерінде жазылған бағдарламаларды орындау үшін де пайдаланылады. Java тілі қатар құрастырылатын және түсіндірілетін тіл болып табылады. Java интерпретаторы — бұл нақты аппараттық – бағдарламалық платформасы: Windows, Linux, Mac, UNIX- машинасына арналған қосымша. Java-код платформаға тәуелді емес. Бұл сәулеттік тәуелсіздікті және Java тілінде бағдарламаларды тасымалдауды қамтамасыз етеді. Java байт коды Java виртуалды машинасы орындалатын кез келген ортада орындалады. JVM Java платформасының негізгі компоненті болады. Java виртуалды машиналары көптеген аппараттық және бағдарламалық платформалар үшін қолжетімді болғандықтан, Java байланыстыратын бағдарламалық қамтамасыз ету болып та, өзіндік платформа болып та қарастырылады. Көптеген платформалар үшін бір байт-кодты пайдалану Java –ны «бір-ақ рет құрастырып, жер – жерде жіберіледі» деп сипаттауға жол береді. Байт-кодтың операциялық жүйеден және құралдан толық тәуелсіздігі Java-қосымшаны ол үшін сәйкес виртуалды машинасы бар кез келген қондырғыда орындауға мүмкіндік береді.

Java технологиясының маңызды ерекшелігі бағдарламаның орындауын виртуалды машина толығымен бақылау шеңберіндегі қауіпсіздіктің икемді жүйесі болып табылады. Бағдарламаның орнатылған өкілеттігінен асатын кез келген операциялар (мысалы, деректерге рұхсатсыз жету әрекеті немесе басқа компьютермен қосылу) тез арада үзіледі.

Виртуалды машинаның концепция кемшілігіне өндірістің төменділігін жатқызады. Жетілдіру қатары Java –да бағдарламаның орындалу жылдамдығын біраз көбейтті:

- байт-кодты машиналық кодқа трансляциялау технологиясын машиналық кодта класс нұсқасын сақтау мүмкіндігімен бағдарламаның (JIT-технология) тура жұмысы уақытында қолдану;
- стандартты кітапханаларда платформалық – бағытталған кодты (native-код) кең пайдалану;
- байт-кодты жылдамдатып өңдеуді қамтамасыз ететін аппараттық құралдар (мысалы, *ARM* фирмасының кейбір процессорларымен қолданатын Jazelle технологиясы).

Концепция негізіне салынған идеялар және Java виртуалды машиналардың ортасын түрлі жүзеге асыру көпшілік дайындаушылар қатарын виртуалды машинада орындалатын бағдарламаларды құру үшін пайдалануға болатын тілдер тізімдемесін кеңейтуге шабыттандырды. Бұл идеялар сонымен *Microsoft* компаниясы NET платформасы негізіне салынған CLI инфрақұрылымның жалпытілдік спецификациясында көрінісін тапты.

Java бағдарламасын құратын негізгі бірліктер кластар болады. Кластар тілдің объектік – бағытталған моделі негізінде жатқан иерархиялық ағашбейнелі құрылымын құрайды. Java-бағдарламаны жазу келесіде қамтылады: жаңа класты немесе өзара байланысқан кластар жиынтығын құру, яғни, шешілетін міндетті бейнелейтін деректер жинағын анықтау, және осы деректерді өңдеуді жүзеге асыратын әдістерді құрастыру. Бұл ретте жаңа кластарды туындау үшін бар болатын базалық кластар коллекциясын пайдалануға болады. Java тілінде класты бейнесін құрайтын түпкі мәтін файлда сақталады, оның аты .java кеңейтуіне ие болады. Бұл файл J-код –қа компиляцияланады және файлда сол атымен және кеңейтілумен .class сақталады. Java-ның құрастырылған кластары компьютерде локалды немесе Торда бөлініп сақталып, қажеттілік бойынша орындалып жатқан қосымшалармен серпінді жүктеле алады.

J-код форматындағы кластар өзінің арнауларына сәйкес бөлектенген пакеттерге топтастырылған. Класс коллекциясымен жұмыс жүргізу үшін DNS домен жүйесін еске салатын пакеттер мен кластарға атау беру арнайы жүйесі пайдаланылады.

Java –да белгілі сілтегіш сияқты төмендеңгейлі конструкциялар жоқ, сонымен қатар, Java –да өте қарапайым жад моделі, бұл жерде әр объект үймеде болады және объектік типіндегі барлық өзгермелілер сілтеме болады. Жадын басқару JVM орындайтын интеграцияланған автоматикалық қалдықты жинау көмегімен жүзеге асады.

Java ортасында бағдарламаның екі негізгі типін анықтауға болады: қосымшалар (application) және апплеттер (applets). Қосымша болып оны орындау үшін тек Java виртуалдық машинаның болуы талап етілетін өзіндік бағдарлама. Апплет — бұл Web-браузер немесе апплетті қараудың арнайы бағдарламасы құрамында орындалуы үшін арналған қосалқы бағдарлама. Java апплет тілі тұрғысынан – бұл Applet класының кеңейтуі.

*Апплет* (applet, application — қосымша және let — қосымшарейткіш жұрнақ) — бұл басқа, толық қосымшаның мәнмәтінінде қызмет ететін, тек бір ғана кішкентай міндетті орындауға арналған және базалық қосымшадан алшақ кетсе құндылығы болмайтын дербессіз бағдарламалық қамтамасыз ету компоненті. Java-апплеттер — Web-беттердің браузері терезесінде орындалатын екілік кодтағы бағдарламалық компоненттер және қауіпсіздік мақсатында негізгі жүйеден бөлектенген.

Java-апплетер көмегімен динамикалық HTML-құжаттары құрылады. Апплеттер HTML-мен беріле алмайтын Web-қосымшалардың интерактивтік мүмкіндіктерін ұсыну үшін пайдаланылады. Так как байт-код Java платформадан тәуелсіз болғандықтан, Java-апплеттер көптеген браузерлермен плагиндер көмегімен орындала алады, соның ішінде Windows, UNIX, Apple Mac OS, Linux . Плагин (plug-in — қосу) — тәуелсіз компиляцияланатын бағдарламалық модуль, негізгі бағдарламаға серпінді қосылатын және оның мүмкіндіктерін кеңейту және пайдалану үшін арналған. Плагиндер әдетте жалпы пайдалануға арналған динамикалық кітапханалар түрінде орындалады. Негізгі қосымшаны плагин пайдалана алатын сервистер ұсынады. Оларға плагинге ұсынылатын өзін негізгі қосымшада тіркеу мүмкіндігі, сонымен қатар, басқа плагиндермен деректермен алмасу хаттамасы жатады. Плагиндер негізгі қосымша ұсынатын сервистерге тәуелді болады, және бөлек пайдаланылмайды.

Соңғы пайдаланушыларға негізгі қосымшаға өзгертулер енгізу қажеттілігісіз плагиндерді динамикалық қосу және жаңғырту мүмкіндігі беріледі.

Апплеттер үшін интерпретатор браузер болады. Апплет Web-серверде жасалады, құрастырылады және сақталады. Серверде жарияланатын HTML- құжатта арнайы тег көмегімен апплеттің орналасу орны туралы сілтеме салынады. Серверден құжат алынғаннан кейін браузер апплетті жүктейді де оны орындай бастайды. Өзге сөзбен айтқанда, апплеттер HTML-бетке құрастырылады да клиент машинасына жіберіліп, браузер қолдауымен орындалады. Апплеттерді браузер ортасына енгізу үшін апплетті шақырушы HTML-тегті жасау қажет.

Апплетті жүктеу келесі жолмен орындалады:

- HTML-файл жүктеледі;
- тег `<applet>` табылады;
- серверден, `applet` –те көрсетілген класс файлы жүктеледі;
- applet класы сілтейтін кластар танылады және жүктеледі;
- applet класы `init()` и `start()` әдістерін шақырады;
- олар жақсы аяқталғанда апплеттің интерфейс бөлімі браузер терезесінде көрінеді.

JavaScript сценарийлері сияқты, Java апплеттері деректерді серверге жіберу алдында оларды енгізу және алдын ала өңдеу, визуалды және дыбыстық эффектер ұйымдастыру, сонымен қатар, серверден алынған деректерді графикалық, кесте немесе қандай да болсын басқа түрде ұсыну үшін қолданылады.

Апплеттер браузердің басқаруымен орындалады және компьютердің локалдық ресурстарына қолжетімділігі жоқ. Алайда олар сервер кеңейтулерімен өзара әрекеттесуге және өздері жүктелген сол Web-серверден файлдарды оқуға қабілетті. Апплет локалдық файлдық жүйеде оқу – жазу операцияларын жүзеге асыра алмайды, яғни файлдарды оқу және өзгерту, каталог ішін қарап шығу, файл және каталогтарды құру, өшіру, қайта атау. Апплет өзі жүктелген компьютерден басқа компьютерлермен жүйелік байланыс жасай алмайды, өзі орындалып жатқан компьютерде орындау үшін басқа бағдарламаларды қоса алмайды. Апплеттерде бағдарламалаудың басқа тілдегі кітапханаларын пайдалану мүмкіндігі жоқ, сонымен қатар, жүйелік параметрлерді өзгертуге болмайды.

Компьютерлік бағдарламаларды қауіпсіз орындау үшін арнайы бөлінген ортаны «құмсалғыш» деп кейде атайды. «Құмсалғыш» әдетте қонақ бағдарламасын орындау үшін қатаң бақыланатын ресурстар жинағы боп есепке алынады, мысалы, дискте немесе жадында орын.

Әдетте, «құмсалғышты» зиян келтіретін кодтан қорғану үшін құрал ретінде белгісіз дереккөзден алынған тексерілмеген кодты қосу, сонымен қатар, зиян келтіретін бағдарламаларды табу және талдау үшін пайдаланады. Және де жиі «құмсалғыштар» бағдарламалық қамтамасыз етуді дайындау үдерісінде байқаусызда жүйеге зиян келтіріп немесе күрделі конфигурацияны бүлдіріп қоятын «шикі» кодты қосу үшін пайдаланады.

Java ішінде технологиялардың бірнеше негізгі тұқымдастығы бар:

- Java SE — Java Standard Edition, Java негізгі баспа, компиляторлары, API, Java Runtime Environment бар; пайдаланушылар қосымшаларын құру үшін жарамды, бірінші кезекте — үстел үсті жүйелерге;
- Java EE — Java Enterprise Edition, кәсіпорын деңгейінде бағдарламалық қамтамасыз етуді құру үшін спецификация жинағы болып табылады;
- Java ME — Java Micro Edition, есептеу қуаттылығы шектеулі қондырғыда пайдалану үшін жасалған, мысалы, ұялы телефонда, КПК, енгізіліп салынған жүйелерде;
- JavaFX — технология, являющаяся следующим шагом в эволюции Java как Rich Client Platform; предназначена для создания графических интерфейсов корпоративных приложений и бизнеса;
- Java Card — смарт – карталары мен өте шектеулі жады көлемі және өңдеу мүмкіндіктерімен басқа қондырғыларда жұмыс жасайтын қосымшалар үшін қауіпсіз ортаны ұсынатын технология.

Java-ның бағдарламалық платформасы - Қосалқы бағдарламалық қамтамасыз етуді дайындау және оны кез келген кросс-платформалық бағдарламалық қамтамасыз етуге енгізу жүйесін бірлесіп ұсынатын *Sun Microsystems* компаниясының бағдарламалық өнімдері мен спецификациялар қатары. Java құрастырылып енгізілетін қондырғылар мен ұялы телефондардан бастап корпоративтік серверлер мен суперкомпьютерлерге дейін әр түрлі компьютерлік платформаларда пайдаланылады. Java-апплеттер үстел үсті компьютерлерде сирек қолданылғанмен, алайда олар кейде Әлемдік ғаламторды шарлауда функционалдық қабілеті және қауіпсіздікті арттыру үшін пайдаланылады.

Java-ның бағдарламалық платформасы қандай болсын бір процессор немесе операциялық жүйе үшін спецификалық болмайды, бірақ орындау механизмі (виртуалды машина аталатын) және кітапхана жинағымен компилятор Java-бағдарламалары барлық жерде бірдей жұмыс жасауы себебінде түрлі аппараттық қамтамасыз ету мен операциялық жүйелер үшін жүзеге асқан.

## 7.7. ТЕХНОЛОГИЯ ACTIVEX

ActiveX басқару элементтері. ActiveX технологиясы басқа қосымшалар, компоненттер, операциялық жүйеге түрлі сервистер ұсынатын бағдарламалық компоненттерді құру мен пайдалануға мүмкіндік беретін COM (Component Object Model) технологиясына негізделеді. COM технологиясы бағдарламалаудың қандай да болсын объектік – бағытталған тілінде жазылған бағдарламаның орындалуы кезінде емес, ал бөлек бағдарламалық бірліктер – компоненттер түрінде болатын өзінің сапалары, әдістері мен оқиғаларымен объектілерді пайдалануға мүмкіндік береді. Дайындалудағы бағдарламаның тұлғасына осындай объект – компонент қосылғанда ол алу және орнату, сонымен қатар, орындауға болатын өзінің сапалары мен әдістерін ашады.

*ActiveX басқару элементтері* — бұл дайындалудағы бағдарламада қажетті функционалдық мүмкіндіктерді жүзеге асыру үшін пайдалануға болатын өзіндік бағдарламалық компоненттер.

ActiveX элементтері Web-беттерді жасақтауда пайдаланылады. Бұл бағдарламалар Web-беттің функционалын кеңейтуге мүмкіндік береді, мысалы, ActiveX көмегімен браузерде видео және музыка онлайн орындату үшін плеерді жүктеуге немесе тура браузер терезесінде басқа форматтағы файлдарды ашуға болады. Сонымен қатар, ActiveX –тің басқару элементтері көмегімен Web-бетте түрлі диалогтық терезелер жасалады, анимация орындатылады. Көптеген компаниялар ActiveX басқару элементтерін өз бағдарламаларын өздерінің интернет – сайтынан пайдаланушы компьютеріне орнату үшін пайдаланады.

Егер сайтты жасақтауда ActiveX технологиясы пайдаланса, онда сол сайтқа кіргенде браузер сұранысын ActiveX басқару элементі қондырғысына шығарады. Егер пайдаланушы сұраныспен келіссе, онда басқарушы элемент пайдаланушы компьютеріне жүктеледі.

ActiveX басқару элементтері өзіндік қосымша болмаса, олар тек ActiveX басқару элементтерін енгізуге мүмкіндік беретін және олар үшін әлдебір контейнер болып қосымшада орындалады

ActiveX басқару элементтері үшін браузер бетте тікбұрышты формадағы белгілі учаскені бөледі. Өз учаске шегінде басқару органы экранның суретін қайта жасау және пайдаланушымен әрекеттестік үшін толығымен жауап береді.

Кез келген қосымша ActiveX басқару элементтері үшін контейнер бола алмайды. Ол COM технологиясын қолдауы керек және енгізілетін компоненттермен манипуляциялық мүмкіндіктерді ұсынуы қажет. *Microsoft* фирмасының көптеген қосымшалары ActiveX басқару элементтерін қосымшаның өзінің функционалдық мүмкіндіктерін кеңейту үшін, немесе ActiveX басқару элементтерін құраушы блоктар ретінде пайдаланатын жаңа қосымшаларды тез жасақтау үшін кең пайдалануға мүмкіндік береді. Осындай қосымшаларға Microsoft Visual Basic, Microsoft Access, Microsoft Internet Explorer және кейбір басқа қосымшаларды жатқызуға болады.

ActiveX басқару элементтері көмегімен қандай да болсын фунуционалдық мүмкіндікті жүзеге асыру үшін бағдарлама жасаушы өзінің ActiveX басқару элементін дайындауы керек әлде басқа дайындаушылардың жасақтаған және таратқан көптеген ActiveX басқару элементтерінің бірімен пайдалануы керек.

ActiveX басқару элементтерін сервер жағында да, клиент жағында да пайдалануға болады. Клиент жағында Java апплеттері сияқты сол міндеттерді шешу үшін қолданылады, мысалы:

- HTML құралдарымен жасақтауға болмайтын ашпалы меню, айналдыру жолағы және басқа визуалдық басқару элементтерін жүзеге асыру;
- пайдаланушы тарапынан немесе келесі кезеңде динамикалық түрде өзгеретін диаграммаларды құрумен серверлік компоненттен деректерді қабылдау, өңдеу және шығару;
- безендіру міндеттерін шешу, мысалы, бөлінген учаскені өрнекпен қаптау, бірқалыпты түстердің ауысуы, қозғалатын мәтін немесе сурет.

Қосымша түрде олар локалдық компьютердің ресурстарына шексіз қолжетімділік ұсынады. ActiveX басқару элементтерінің компьютердің локалдық ресурстарына қолжетімділігі тек артықшылықтарға емес, кемшіліктерге де ие. Әлбетте, бұл технологияда қауіптік жасырынған. ActiveX басқарушы компоненттері – бұл Web-браузер арқылы қосылатын бағдарламалар болғандықтан, қандай болсын зиянды бағдарлама немесе вирусты «жұқтыру» қаупі әбден шынайы, өйткені зиян келтірушілер ActiveX технологиясын түрлі вирустар мен шпиондық бағдарламалар жасау және тарату үшін пайдалануы мүмкін.

HTML-бетке сыртқы объектілерді енгізіп орнату үшін енгізілетін объектінің сапалық мәнін анықтаушы <param> тегтеріне контейнер болатын <object> тегі арналған.

Компьютерде қондырылатын кез келген ActiveX басқару элементі сәйкес ақпарат енгізіліп, сол жерде сақталатын жүйелік реестрде тіркеледі. Реестрдың HKEY_ CLASSES_ROOT/CLSID/ бөлімінде компьютерде орнатылған барлық ActiveX басқару элементтерінің уникалды идентификациялық нөмірлері сақталады. Болашақта бұл ақпарат Internet Explorer браузерімен HTML-беттерін енгізілген ActiveX басқару элементтерімен қамтылған өңдеу үшін пайдаланылады.

Егер бетте пайдаланылатын ActiveX басқару элементі пайдаланушы компьютерінде орнатылмаса, онда браузер басқару элементін жүктеу және орнату процедурасын осы компьютерде бастайды.

ActiveX басқару элементтері үшін classid параметрі тапсырмасы міндетті. Оның мәні енгізіліп орнатылатын ActiveX басқару элементтерінің уникалды идентификациялық нөмірі болады. Бетті жүктеуде браузер пайдаланушы компьютерінде басқару элементі орнатылғанын тексереді, оны берілген идентификациялық нөмірі бойынша жүйелік реестрден іздеу арқылы жүзеге асырады. Реестрде жазба болмаған жағдайда браузер автоматты түрде ActiveX басқарушы элементтерін серверден жүктеу процедурасын бастайды, оның URL-адресі<object>codebase тегі параметрінде көрсетілген.

classid параметрінің мәні енгізіліп орнатылатын объектіні анықтайтын жолында ұсынылады, және тіркелген ActiveX басқару элементтері үшін келесі формада беріледі:

```
"classid:XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
```

Бірінші бөлімі classid: браузер анализаторына ActiveX басқару элементі енгізілетіні туралы хабарлайды. Екінші бөлімі сол элементтің уникалды идентификациялық нөмірін білдіреді.

codebase параметрінде пайдаланушы компьютерінде болмаған жағдайда браузердің оны жүктеуі үшін компонент адресі беріледі.

Үлгі:

```
<object id="myactx" width="50" heigth="50"
classid="CLSID:132D558D-1970-43S3-
V254-
58030FC10000"
codebase="http://mysite.ru/ActiveX/adve
rt32.cab #version=1,0,0,0">
</object>
```

Басқару элементі адресінен кейін `#version=a,b,c,d,` түрінде оның версиясы беріледі, бұл жерде a және b серверде басқару элементі нұсқасының максималды қолжетімді үлкен және қосымша сөзін сәйкес көрсетеді, ал c және d —сервердегі басқару элементі нұсқасының минималды қолжетімді үлкен және қосымша сөзін сәйкес көрсетеді. Бұл мәндер браузерлермен басқару элементін серверден жүктеу туралы шешім қабылдау үдерісінде пайдаланылады. Егер пайдаланушы компьютерінде басқару элементінің тым жаңа версиясы орнатылған болса, жүктеу үдерісі жүрмейді.

ActiveX басқару элементтерін қолдану қауіпсіздігі. Java- апплеттер және ActiveX басқару элементтері оларды қауіпсіз қолдану есебімен жасақталады. Бұл екі тип объекті үшін қауіпсіздік стратегиясы әртүрлі. Егер Java-апплеттерді дайындау құралдары оларды компьютерде орналасқан ақпаратқа жүгінуге мүлдем ерік бермесе, ал ActiveX басқару элементтерімен іс одан да күрделі.

ActiveX басқару элементтері тек Web-қосымшаларында ғана емес, сонымен қарапайым қосымшаларды дайындауда пайдаланылады. Сондықтан ActiveX басқару элементтерін дайындау ортасының ешқайсысы да бұл элемент пайдаланатын компьютерде сақталатын ақпаратқа қол жеткізуге шектеу қоя алмайды. ActiveX жүйесі компонентімен қамтылған . осх кеңейтумен файл Windows –дегі кез келген басқа орындаушы файл сияқты басқаруды тура солай иемденеді және сондай құқықтарға ие болады, мысалы, дискке бақылаусыз түрінде жазба жасау.

Осыған байланысты ActiveX басқару элементтерінің ақпаратқа қолжетімділігін шектеу бірлескен браузердің қауіпсіздік жүйесі мен пайдаланушы компьютеріне ActiveX басқару элементтерін HTML-бетті жүктеу кезінде тіркеу және орнатумен байланысты шаралармен жүзеге асады.

ActiveX басқару элементтерін жүктеу қауіпсіздігін қамтамасыз ету үшін сандық сертификаттар технологиясы пайдаланылады. Қажетті файлдарды жүктеу алдында браузер басқару элементінің сандық қолмен қамтылуын тексереді, егер қамтылса, онда диалогтық терезеде сандық қолдың ішіндегі мазмұнды бейнелейді, егер қамтылмаса – пайдаланушыны сандық қолсыз басқару элементі жүктелетіні туралы ескертеді. Пайдаланушы аталған басқару элементін жүктеу немесе жүктемеу туралы өзі шешім қабылдайды.

ActiveX басқару элементтерін дайындаушы өз элементін келесі түрде сандық қолмен қамтиды: дайындаушы сәйкес уәкілетті мекемеден сәйкесті тұпнұсқалылық сертификатын алады; сертификатты алғаннан кейін арнайы бағдарламалық қамтамасыз ету көмегімен бағдарламалық өнімді өндіруші таратуға арналған ActiveX басқару элементінің екілік кодына сандық қолды енгізеді.

Әрекеттегі сандық қол жүктелетін ActiveX басқару элементі нақты фирма немесе тұлғамен дайындалғанына кепілдік береді, және ол фирма – өндіруші қол қойған күннен кейін ешкіммен өзгертілмегенін растайды. ActiveX басқару элементтерін Web сервері үшін дайындау жағдайында оларға қол қоймауға болады. Бұл модулдер клиенттермен жүктелмейді, тек тура серверде орындалады.

ActiveX жүйесі және онда жүзеге асқан қауіпсіздік механизмі Java тілін пайдаланудан ыңғайлы. Виртуалды машинаның қорғау қабының болмауы ActiveX компоненттерінің функционалдық қабілеттерін кеңейтуге мүмкіндік береді. Оларға, Java-апплеттердің жүзеге асыруға мүмкіндігі жоқ, компьютер ресурстары үстінен тура және тиімді бақылау қолжетімді. Алайда бұл браузер қорғауындағы осалдықтармен байланысты қосымша мәселелер туындырады.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. DOM құжатының объектік моделі деп не аталады?
2. Құжаттың объектік моделі қандай артықшылықтарға қол жеткізеді?
3. HTML –дің бесінші нұсқасын жабдықтау мақсаты неде?
4. HTML5-тің алдыңғы версиялармен салыстырғандағы артықшылықтарын атап беріңдер.
5. Қандай бағдарламалар сценарий немесе скрипт деп аталады?
6. JavaScript не үшін қолданылады?
7. JavaScript сценарийлері HTML беттеріне қандай жолдармен енгізіледі?
8. JavaScript –те кластарды пайдалануда бар болатын шектеулерді көрсетіңіз.
9. AJAX технологиясы не үшін пайдаланылады?
10. Қандай сұраныс асинхронды деп аталады?
11. AJAX технологиясында XMLHttpRequest объектісі қалай пайдаланылады?
12. AJAX- әрекеттестік түрлерін атаңыз.
13. VBScript –тің негізгі кемшілігі неде?
14. Java технологиясының ерекшеліктерін атаңыз.
15. Апплет дегеніміз не, түсіндіріңіз.
16. Апплетті жүктеу қалай орындалады?
17. Клиент жағында Java-апплеттер қандай міндеттерді шешеді?
18. ActiveX қандай технологияға негізделеді?
19. ActiveX басқару элементтері нені білдіреді?

20. Қандай қосымшалар ActiveX басқару элементтері үшін контейнер бола алады?
21. ActiveX басқару элементтері клиент жағында қандай міндеттерді орындайды?
22. Компьютерде қондырылатын ActiveX басқару элементін тіркеу қалай жүреді?
23. Java-апплеттер және ActiveX басқару элементтері үшін қауіпсіздік стратегиялары ерекшеліктерін түсіндіріңіз.

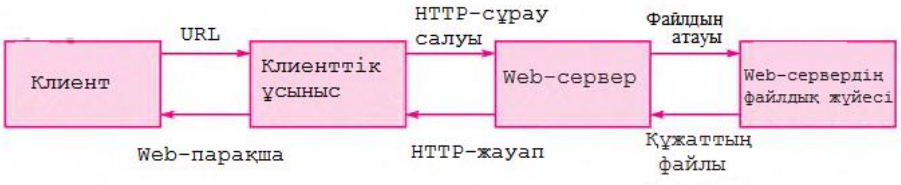
# 8 тарау

# СЕРВЕРЛІК WEB-БАҒДАРЛАМАЛАУ

## 8.1. WEB-СЕРВЕРДІҢ ЖҰМЫС ІСТЕУ МЕХАНИЗМІ

Серверлік бағдарламалау оған сұрау салуды жіберу арқылы клиентпен өзара іс-қимылды жүзеге асыратын сервердің жағында әртүрлі міндеттерді шешуді болжайды. HTTP хаттамасының шеңберінде клиенттік-серверлік өзара іс-қимылдың мүмкіндіктерін кеңейту үшін клиент жағында белгілеу тілдерімен және браузерлермен ұсынылатын стандартты мүмкіндіктерді кеңейтуді құрудан басқа, қосымшаның Web-серверінің жағында Web-сервердің өзінің мүмкіндіктерін кеңейтетін плагиндер мен сценарийлерді әзірлеуге болады. Әдетте серверлік Web-бағдарламалаудың негізгі міндеттері мыналар: клиенттік сұрау салуларды өңдеу, клиентке жауапты қалыптастыру, дерекқор серверлерімен жұмыс істеу. Ол құрылатын қосымшаға қойылатын талаптармен және тілдің мүмкіндіктерімен ғана шектелетіндіктен, бұл тізбені үздіксіз жалғастыруға болады. Серверлік Web-қосымшалар әртүрлі бағдарламалау тілдерінің көмегімен құрылады, осы тарауда олар туралы сөз қозғалады.

*Web-сервер* — кіріс HTTP-сұрау салуларды қабылдайтын, осы сұрау салуларды өңдейтін, HTTP жауабын қалыптастыратын және оны




8.1 - сурет. Браузер мен Web-сервердің өзара іс-қимылы

клиентке жіберетін бағдарлама. Web-сервердің жалпы жұмыс алгоритмі 8.1 – суреттегі схемада берілген.

Пайдаланушы HTTP хаттамасы бойынша белгілі бір ресурсқа сұрау салғаннан кейін клиент (әдетте браузер) Web-серверге HTTP-сұрау салуын қалыптастырады. Әдетте сервердің символдық атауы көрсетіледі (мысалы, http://academia-moscow.ru), бұл жағдайда браузер алдын ала құрылған атауды DNS сервистерінің көмегімен IP-адреске айналдырады. Бұдан кейін Web-серверге HTTP хаттамасы бойынша қалыптастырылған HTTP-хабарламасы жіберіледі. Осы хабарламада браузер жүктелуі тиісті ресурсты, сондай-ақ барлық қосымша ақпаратты көрсетеді. Web-сервердің міндеті — белгілі бір TCP-портты (әдетте 80-порт) тыңдау және барлық кіріс HTTP-хабарламаларын қабылдау. Егер кіріс деректер HTTP хабарламасының форматына сәйкес келмесе, ондай сұрау салу назарсыз қалады, ал клиентке қателік туралы хабарлама келеді.

Қарапайым жағдайда HTTP-сұрау салуы түскен кезде Web- сервер қатты дискіден сұратылған файлдың мазмұнын оқуы, оның мазмұнын HTTP-жауабына жүктеуі және клиентке жіберуі тиіс. Талап етілетін файл қатты дискіде табылмаған жағдайда Web-сервер 404 мәртебе беру кодын көрсетумен қателікті қалыптастырады және осы хабарламаны клиентке жібереді. Web-сервердің осындай жұмыс істеу нұсқасы әдетте тұрақты сайттар деп аталады.

*Тұрақты сайт* — бір бүтінді құрайтын, тұрақты html (htm, dhtml, xhtml) беттерінен тұратын сайт. Тұрақты сайтта (HTML түрінде белгіленген) мәтін, суреттер, мультимедиа (аудио, бейне) және HTML-тегтер орналастырылады. Бұндай жағдайда сервер тарапынан Web-сервердің өзіндік бағдарламалық кодынан басқа ешқандай бағдарламалық код іске қосылмайды.

*Динамикалық сайт* — динамикалық беттерден тұратын сайт — шаблондар, контент, жеке файлдар түріндегі скрипттер. Пайдаланушы браузерінің соңында көрсетілетін сайттың беті сервер тарапында динамикалық түрде, сұрау салу бойынша, үлгі-беттен және сақталатын мазмұннан (ақпарат, скрипттер, дерекқор және басқалары) жекеше қалыптастырылады. Әдетте, біртектес беттердің кез келген санын көрсету үшін тиісті мазмұн жүктелетін бір үлгі-бет пайдаланылады, бұл тек бір ғана үлгіні редакциялай отырып, сайттың сыртқы келбетін (оның барлық беттерін) тез түзету мүмкіндігін береді.

Өз бетіндегі мазмұнды (сондай-ақ үлгі-бетті де) сайт құралдарының көмегімен және сыртқы бағдарламалық жасақтаманы қолдану арқылы редакциялауға болады. Барлық беттерді түзету мүмкіндігі тек пайдаланушылардың белгілі бір

санатына (мысалы, әкімші немесе тіркелген пайдаланушылар) ғана беріледі. Кейбір жағдайларда белгілі бір Web-контентті түзету құқығы анонимді пайдаланушыларға беріледі, бұндай жағдай сирек кедеседі (мысалы, форумдарда — хабарламаларды қосу).

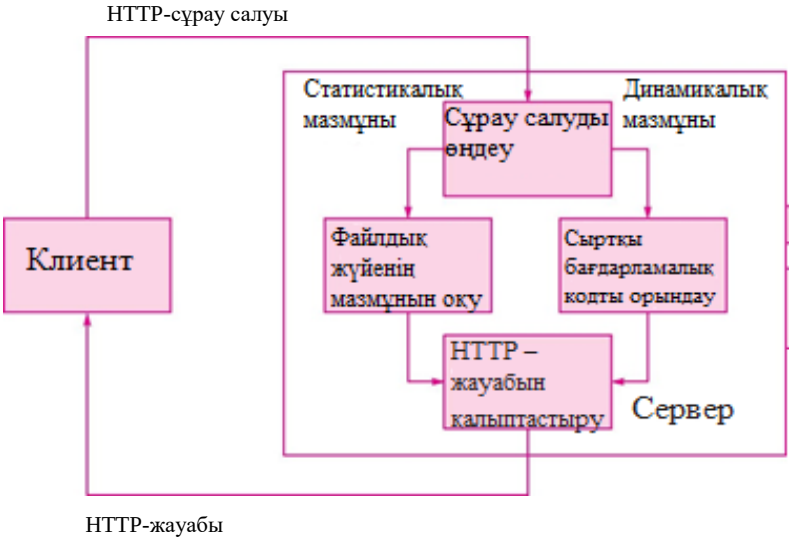
Қазіргі таңда динамикалық беттерді генерациялау үшін анағұрлым танымал технологиялар мыналар:

- PHP —Apache серверлері және GNU/ Linux басқаратын, басқа UNIX-тәріздес және өзге операциялық жүйелер үшін;
- JSP және Java Servlet — әртүрлі операциялық жүйелер басқаратын Apache, JBoss, Tomcat серверлері үшін;
- ASP. NET —IIS басқаратын Microsoft Windows серверлері үшін.

Динамикалық Web-қосымшаларда сервер файлды оның файлдық жүйесін оқығаннан кейін және оны пайдаланушының Web- браузеріне жібергенге дейін қайта құрады. Web-қосымшалардың сервері файлдарды қайта құру кезінде пайдаланатын бағдарламалық жасақтама *интерпретатор* деп аталады. Интерпретатор файлдарды оқиды және оларды файлдардың ішіндегі нұсқаулықтарға сәйкес өңдейді, одан кейін нәтижені пайдаланушының Web-браузеріне HTML-беті түрінде жібереді. Әрине, кейбір ресурстарда (CSS файлдары, суреттер және басқалары сияқты) статистикалық мазмұн сақталуы мүмкін, бірақ негізгі HTML беттері өңдеу кезінде пайда болады. Бұндай жағдайда Web-сервер HTTP сұрау салуын өңдеу кезінде мазмұнды қалыптастыруы тиісті бағдарламалық кодқа жүгінуі тиіс. Web-сервердің осы жұмыс алгоритмі есебімен 8.2 – суреттегідей схеманы көрсетуге болады.

Динамикалық сайттарды құру үшін қолданылатын бағдарламалау тілдерінің бірі - PHP тілі. HTML-да Web-беттердің кодын жазу кезінде код арнаулы операторлардың көмегімен PHP кодын орнату жеріне жүктеледі. Сондықтан, HTML әрқашан үстіне бағдарламалау элементтері өсірілетін негіз болып табылады. Кез келген HTML-беті өте оңай түрде PHP-бетіне айналады, бұл үшін файлдың атауын өзгерту ғана қажет, бұл ретте кеңейтілім .html-ден .php-ға өзгертіледі, PHP-кодын орнату алдын ала серверде өңделетінін ескере отырып, оны тек серверде ғана орындауға болады, ал одан кейін ол дайын Web-бет браузерге беріледі. Осыған орай, PHP Web- бағдарламалаудың серверлік тілі деп те аталады.

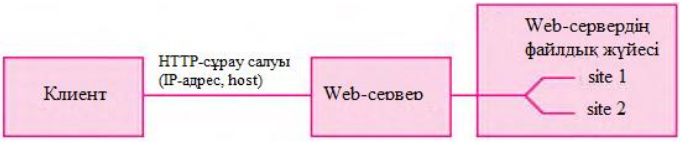
Web-серверді құру кезінде шешілетін анағұрлым маңызды міндеттердің бірі — масштабтауды, яғни қызмет көрсетілетін пайдаланушылар санын көбейту мүмкіндігін қамтамасыз ету міндеті. Web-сервер ашық ортада жұмыс істеуіне байланысты, оған қолжетімдік




8.2 - сурет. Web-сервердің жұмыс істеу алгоритмі

кез келген адамға беріледі. Осылайша, Web-серверге үлкен жүктемелер түсірілуі және потенциалды қауіп төнуі мүмкін. Web-серверге анағұрлым кең таралған шабуылдар Web-серверге көптеген сұрау салулармен жүгіну және олардың жоғары жиілігі — DDoS-шабуылдар болып табылады. Бұндай жағдайда Web-сервер тез арада барлық сұрау салуларды өңдей алмайды, бұл Web-сервердің өнімділігіне әсер етуі мүмкін. Әсіресе, осындай шабуылдар көбінесе қандай да бір сыртқы бағдарламалық код орындалатын Web-серверлерге жасалады, бұған Web-сервердің өз бағдарламалық коды ғана жатпайды. Осыған ұқсас шабуылдармен күресу үшін әдетте белгілі бір IP-адрестен түсетін барлық сұрау салулар оқшауланады. Оған қоса, қосымшаның бағдарламалық кодын оңтайландыру туралы ойлау қажет, мысалы, кэштеуді пайдалану — бұл жайдайда әрбір сұрау салуды өңдеу кезінде орталық процессорға жүктеме азырақ болады және бұл шабуыл жасаушылардың жағдайын түбегейлі қиындатуы мүмкін.

Желіде тұрақты түрде орналасқан серверде ақпаратты орналастыру үшін есептеу қуаттылығын ұсыну қызметтері *хостинг* деп аталады. Виртуалды хостинг (shared hosting) — көптеген Web-сайттар бір Web-серверде орналасатын хостингтің түрі. Бұл кішкентай жобалар үшін жарамды хостингтің тиімді түрі. Әдетте әрбір Web-сайт




8.3 - сурет. Виртуалды хостинг

Web-сервердің белгілі бір бөлімінде орналасады, бірақ олардың барлығы бір бағдарламалық жасақтаманы пайдаланады. Браузер HTTP-сұрау салуын Web-сервердің домендік атауға ұқсайтын IP-адресіне жібереді. Сондай-ақ клиент Web-сайттың (8.3 - сурет) түпнұсқалық атауы айқындалатын қосымша «host» HTTP-тақырыпатын көрсетеді. Осындай ақпараттың арқасында Web-сервер бірнеше Web- сайттарға қолжетімділікті шектей алады және бұл ретте бірдей IP-адресті пайдаланады. Бұл өте маңызды нәрсе, өйткені, басқа жағдайда әрбір домендік атау үшін жеке IP-адресті тіркеу қажеттігі туындар еді, онда IP хаттамасының адрестік кеңістігі тез бітер еді, ал Web-сайтты жаһандық Интернет желісінде орналастыру құны анағұрлым жоғары болар еді.

Виртуалды хостингте әдетте әртүрлі қызметтерді және сервистерді іске қосуға тыйым салынады, сондай-ақ орталық процессорды пайдалану деңгейіне шектеу қойылады. Кейде виртуалды хостингті серверден ресурстардың зор көлемі қажет болуына немесе осы сервердің шеңберінде қосымша қосымшалар мен қызметтерді іске қосу қажеттігі туындауына байланысты әрқашан пайдалануға болмайды. Бұндай жағдайда әдетте бөлінген – физикалық немесе виртуалды серверді жалға алады. Алайда, бұл Web- қосымшаларды Интернетте орналастырудың анағұрлым қымбат түрі.

HTTP-сұрау салуларын өңдейтін және HTTP-жауаптарын дайындайтын бағдарламалық кодты шартты түрде екі бөлікке бөлуге болады:

1) HTTP хаттамасы арқылы өзара іс-қимылды жасау бойынша қызметтік функцияларды іске асыратын бағдарламалық код (Web-сервердің өзіндік бағдарламалық коды);

2) бір Web- қосымшаның (бизнес-логика, ДҚБЖ-ға сұрау салу және басқалары) логикасын іске асыратын бағдарламалық код.

Web-қосымшаның бағдарламалық коды әдетте жеке модульдерде топталуына және тәуелсіз түрде жеткізілуіне байланысты осы екі бөліктің өзара іс-қимыл механизмі - Web-сервер және қосымша бір-бірімен өзара іс-қимыл жасайтын қағидалар жинағы талап етіледі.

## 8.2. CGI СТАНДАРТЫ

Web-сервер шешетін міндеттер шеңбері негізінде HTTP хаттамасы бойынша клиент пен сервердің өзара іс-қимылына қолдау көрсету және клиентке Web-құжаттарды жеткізу жатады. Осы өзара іс-қимыл белгілі бір қағидаларға бағынады. Осындай қағидалардың негізгі жинағы — CGI стандарты.

CGI (Common Gateway Interface — жалпы пайдалануға арналған шлюз интерфейсі) — сыртқы бағдарламаның Web-сервермен байланысы үшін пайдаланылатын интерфейс стандарты. CGI серверде бағдарламаны іске қосу тәртібін, бағдарламаға параметрлерді беру және клиентке оны орындау нәтижелерін жеткізу тәртібін айқындайды. CGI —Web-сервер мен Web-қосымшаның бір-бірімен анағұрлым ерте іс-қимыл жасау тәсілі. CGI-бағдарламаларды (шлюздерді) сервер нақты уақыт режимінде еске қосады. Сервер пайдаланушының сұрау салуларын оларды өңдейтін және пайдаланушыға жұмыс нәтижесін жіберетін CGI-бағдарламасына береді. Осылайша, кіруші әртүрлі оқиғалар мен факторлардың әсерінен өзгеруі мүмкін динамикалық ақпаратты алады.

CGI бағдарламасы әртүрлі бағдарламалау тілдерінде жазылуы мүмкін және Web-сервердің операциялық жүйесі ортасында жұмыс істейтін және дерек алмасуды жүзеге асыратын қарапайым консольдік қосымша болып табылады. CGI сервердің операциялық жадына жүктеледі және тек оған HTML құжатынан пайдаланушының нақты сұрау салуы кезінде ғана іске қосылады. Пайдаланушының сұрау салуын өңдеп біткеннен кейін CGI бағдарламасы өз жұмысын аяқтайды және сервердің жадынан шығады. CGI бағдарламасы жеке процесс ретінде жұмыс істейді. Аталған процесс шеңберінде ол біржағынан дерекқорға ақпаратты таңдау немесе жаңарту үшін сұрау сала алады. Кезекті HTTP-сұрау салуы түскен кезде Web-сервер жаңа процесті құра бастайды және оған HTTP-сұрау салуының барлық қажетті деректерін береді. Осы процесс өңделгеннен кейін, ол жұмысын тоқтатады және нәтижесін Web-серверге кері қайтарады. Web-сервер мен қосымша операциялық жүйе тұрғысынан әртүрлі процестер болып табылатындықтан, олар арасында ақпарат алмасу үшін процессаралық өзара іс-қимыл құралдары қолданылады.

Web-сервер мен қосымша арасындағы процестердің бір-бірінен оқшау орналасуы бір жағынан қарасақ CGI басымдылығы болып табылады. Web-қосымшада қателіктер жіберілген жағдайда тек қосымшаның процесі қателікпен аяқталады, ал Web-сервердегі процесс жұмыс істеуін жалғастырады. Басқа жағынан қарасақ, әр жолы жаңа процесті құру қажеттігі

процесті құруға және деректерді процестердің шегі арқылы беруге қосымша шығын ресурстарын жұмсауға әкеледі. 100 клиент бір уақытта серверге сұрау салған кезде жадыда 100 процесс құрылады, бұл жағдайда әрбір кодты орналастыру үшін бос жады қажет. Осы факт үлкен кемшілік болып табылады және Web-қосымшасының масштабталуына, яғни түсетін сұрау салулардың үлкен көлемін өңдеу мүмкіндігіне зор әсер етеді.

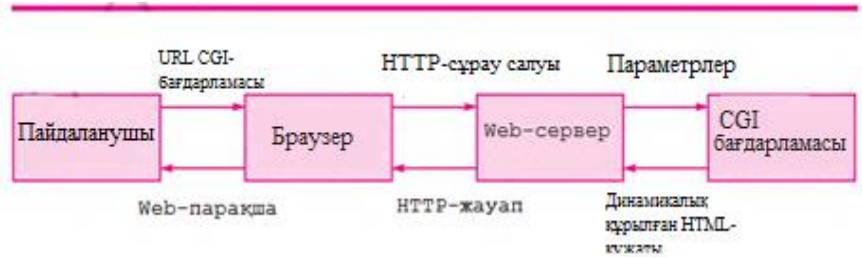
Web-сервер әр жолы клиенттен сұрау салу алғанда, сұрау салудың мазмұнына талдау жасайды және тиісті жауап қайтарады:

- егер сұрау салуда қатты дискідегі файлға қатысты нұсқаулық болса, сервер жауаптың құрамында осы файлды қайтарады;
- егер сауалда бағдарламаға және оған қажетті аргументтерге қатысты нұсқаулық болса, онда сервер бағдарламаны орындап, оның жұмыс нәтижесін клиентке қайтарады.

CGI жұмысының реттілігі мынадай (8.4 - сурет):

1. *Web-сервердің клиенттен ақпаратты алуы.* Web-серверге деректерді беру үшін нысандар пайдаланылады. Нысан HTML-құжатында <form> ... </form> тегтерінің көмегімен беріледі және браузер графикалық басқару элементтері түрінде көрсететін енгізу жолдарының жинағынан тұрады: селекторлық түймелер, опциялар, енгізу жолдары, басқару түймелері және басқалары. Пайдаланушы барлық нысанды толтырған кезде нысанның жолындағы деректер CGI бағдарламасына беріледі.

2. *Алынған ақпаратты талдау және өңдеу.* HTML-нысанынан алынған деректер CGI- бағдарламасына өңдеу үшін беріледі. Олар әрқашан CGI-бағдарламасымен өздігінен өңделмейді. Мысалы, олар дерекқорға сұрау салулардан тұруы мүмкін. Бұл жағдайда CGI-бағдарламасы алынған ақпарат негізінде қандай да бір қашықтықтан басқарылатын компьютер орындайтын ДҚБЖ ядросына сұрау салуды қалыптастырады.




8.4 - сурет. CGI көмегімен HTML динамикалық құжаттарды қалыптастыру

3. *Жаңа HTML-құжатын құру және оны браузер арқылы жіберу.* Алынған ақпарат өңделгеннен кейін CGI-бағдарламасы динамикалық HTML-құжатын құрады, қалыптастырылған құжатқа сілтемені қалыптастырады және нәтижені клиентке береді. Пайдаланушыға сауалға жауап ретінде ол үшін арнайы құрылған HTML-кодты жібереді. CGI-скрипттер түсініктеме бергіш тілде (Perl, PHP) немесе командалық жолда скриптте жазылуы мүмкін. Осылайша, Web-серверге кіру кезінде пайдаланушы енгізген деректерді қабылдайды. Алынған деректер өңделгеннен кейін пайдаланушы қорытынды бетке қайтады.

CGI-бағдарламаның өзіндік ерекшелігі бар, әдетте оның мәні мынадай бөліктерден тұратын сервердің жауабы түрінде клиентке жіберілетін HTML-құжатын құру болып табылады:

1. HTTP хаттамасы нұсқасының нөмірінде берілетін жай-күй жолы, жай-күйдің коды және жай-күйдің қысқаша сипаттамасы, мысалы:

```
HTTP/1.0 200 OK # Клиенттің сұрау салуы ойдағыдай өңделді
HTTP/1.0 404 Not Found # Аталған адрес бойынша құжат жоқ
```

2. Сервер және қайтарылатын HTML-құжат туралы ақпараттан тұратын жауаптың атаулары, мысалы:

```
Server: Apache/1.3.6 # Сервер нұсқасының атауы және нөмірі
```

3. Жауаптың мазмұны — CGI-бағдарламасын орындау нәтижесі болып табылатын HTML - құжаты.

Шлюз стандартты шығыс ағыны арқылы шығуды жүзеге асырады. Осы шығыс шлюздің көмегімен құрылған құжат немесе қажетті құжат алынатын серверге нұсқаулық болып табылады. Бұл ретте алынған деректерге талдау жасамайды және оларды өзгертпейді, ол тек жалпы ақпараттан (мысалы, ағымдағы күн және уақыт) және өзі туралы ақпараттан (мысалы, сервердің аты және нұсқасы) тұратын кейбір тақырыптармен толтыра алады.

Жауапты жол сұрау салу үшін пайдаланылатын әдіске қарай URL (GET әдісі қолданылған жағдайда) бөлігі немесе HTTP-сұрау салулардың мазмұны (POST әдісі) ретінде пайда болады. Соңғы жағдайда осы ақпарат шлюзге стандартты енгізу ағынымен жіберіледі.

Web-серверді орнату кезінде каталогтардың бірі арнайы түрде сценарийлерді сақтау үшін бөлінеді. Әдетте, осындай каталогқа

cgi-bin (немесе IIS Web-сервері үшін Scripts) атауы беріледі. Клиент cgi-bin каталогінен файлды сұраған жағдайда сервер осындай сұрау салуды сценарийді іске қосу командасы ретінде қабылдайды. Басқа каталогтардан алынған файлдар HTML-құжаттары ретінде қабылданады.

## 8.3. PERL ТІЛ

Асолютті CGI мобильді бағдарламаларын құру үшін Perl тілін пайдаланған тиімді болады. Осы тілдің интерпретаторлары барлық операциялық жүйелерде құрылған деуге болады. Осы жоғары деңгейлі тіл CGI бағдарламасын құруды жеңілдететін көптеген функциялардан тұрады. Әртүрлі платформаларға арналған Perl интерпретаторының нұсқасы (айта кететіні, тегін), әртүрлі құжаттама және бағдарламалардың мысалдары Интернетте қолжетімді.

Perl тіліндегі бағдарлама тілдің интерпретаторы әрбір іске қосу кезінде бағдарламаның бастапқы кодын орындалатын екілік кодқа қайта құрусыз орындайтын операторлардың іс-қимыл реттілігі болып табылады.

Үлгілік тілдермен салыстырғанда Perl өз объектілерінің түрлерін жариялауды талап етпейді. Бағдарламада белгіленген барлық объектілер оларға үнсіз келісу бойынша қандайда бір мағына берілгенге дейін «0» мағынасын қабылдайды. Түсіндірмелер # белгісімен белгіленеді және осы белгіден кейінгі барлық жол түсіндірме ретінде қабылданады. Кіші бағдарлама тек бағдарламаның мәтіні бойынша төмен жағынан шақырылады. Бағдарламаның денесінде логикалық тұрғыда біртұтас болып қабылданатын операторлардың іс-қимыл реттілігі блок деп аталады; әдетте, блоктар фигуралы жақшаға алынады. Әрбір команда басқалардан үтір арқылы бөлінеді. Оператор соңғы блокта жазылмаса, нүктелі үтірді қою қажеттігі жоқ.

Perl — қисынсыз мәтіндік файлдарды өңдеу, олардан қажетті ақпаратты алу және хабарламалар беру үшін бейімделген интерпретациялау тілі болып табылады. Perl қолданыста қарапайым және тиімді, өз бойында C тілінің ең үздік қасиеттерін жинаған, осыған орай, C тілін білетіндер үшін Perl тілін үйрену синтаксистің ұқсастығына қарай қиындық тудырмайды. Perl тұрақты қолданылатын сөйлемдерді пайдалану, объектілерді құру, бағдарламадағы C немесе C + +-қа Perl-дегі кодтың бөлшектерін қою мүмкіндігін береді, сондай-ақ дерекқорларға, оның ішінде Oracle-ға қолжетімділікті береді.

Perl тілінде орындау командасын іске қосу үшін оны компиляциялау қажеттігі жоқ, ол интерпретатордың басқаруымен орындалуы мүмкін. Perl бастапқы мәтіні берілген файлды орындау командасымен

іске қосу үшін бірінші жолда мыналар жазылуы тиіс:

```
#!Perl_интерпретаторына_апаратын_жол
```

Perl тіліндегі негізгі деректердің түрлері мыналар:

- скалярлар;
- скалярлық массивтер;
- ассоциативтік массивтер (хеш-кестелер);
- функциялар;
- файлдық дескрипторлар;
- константтар.

Әртүрлі ауыспалылар ауыспалының алдында тұратын таңбамен ерекшеленеді:

- `$a` — скаляр немесе көрсеткіш;
- `@b` — скалярлық массив;
- `%c` — ассоциативтік массив (хеш-кесте);
- `&d` — функция;
- `F` — енгізу-шығару дескрипторы немесе константа.

Скалярлық ауыспалылар жалғыз мағыналарды сақтау үшін пайдаланылады. Олар сандардан, жолдардан және басқа объектілерге сілтемелерден тұруы мүмкін. Скалярлық ауыспалының түрі бекітілмеген және контекстке байланысты динамикалық тұрғыда айқындалады.

Скалярлық массив скалярлардың реттелген тізімі болып табылады. Массивтің әрбір элементіне оның көмегімен оған қолжетімділік алуға болатын реттік нөмір (индекс) беріледі. Элементтерді нөмірлеу нөлден басталады. Массивтің белгілі бір элементіне қолжетімділік алу үшін массивтің белгілі бір элементі скаляр болып табылатындықтан $ белгісін қою қажет:

```
@winter = ("желтоқсан", "қаңтар", "ақпан");
print "Қыстың екінші айы", $winter[1], "\n";
```

Хеш-кесте жолды скалярмен байланыстыру мүмкіндігін беретін ассоциациялық массив болып табылады. Бұл ретте жол кілт, ал скаляр хеш-кестеде — мағына (тақ позицияларда кілттер, ал жұп позицияларда — мағыналар орналасқан) деп аталады. Ауыспалы тізім атауларының алдында % пайыз белгісін қою қажет, ал массивтің белгілі бір элементіне қолжетімділік алу үшін $ белгісін қою қажет.

Ассоциациялық массивтерді пайдалану скалярлық мағыналар массивін пайдалануды еске салады, бірақ индексация бүтін сандармен емес, негізгі сөздермен жүзеге асырылады.

Perl соңғы нұсқаларында файл дескрипторларын скалярларда сақтау мүмкіндігі пайда болды. Сондай-ақ символдар кестесі жадыда бір мағынасы бар ауыспалылардың екі атауының байланысы үшін

пайдаланылады, олар нүпнұсқалық атауға сияқты мағынаға да қолжетімділік беру және түрлендіру үшін пайдаланылуы мүмкін синонимдерді құрады. Аталған мүмкіндік Perl-ға жүктелетін модульдердің, класстардың және объектілердің негізі болып табылады.

Perl-ге жүктелетін объект белгілі бір класспен (пакетпен) байланысты қарапайым сілтеме болып табылады. Байланыс үшін `bless` функциясы пайдаланылады. Осындай байланысқан пакеттің кіші бағдарламасы әдістер болып табылады және бірінші аргумент ретінде сілтемені алады.

Perl-дің маңызды бөлігі тұрақты мағыналарды пайдалану болып табылады. Perl – дегі тұрақты мағыналар — оның ең мықты мүмкіндіктерінің бірі. Олар мәтін мен берілген үлгіні салыстыру, мәтінді үлгі бойынша массивке бөлу, мәтінді үлгі бойынша алмастыру және басқаларын жүзеге асыру мүмкіндігін береді. Осыған орай, Perl мәтіндерді өңдеу үшін өте тиімді. Тұрақты мағыналармен жұмыс жасаудың үлкен бөлігі `=~, m// и s///` лh операторларының көмегімен жүзеге асырылады.

`M//` операторы бірдей жерлерді тексеру үшін пайдаланылады. Қарапайым жағдайда `$x =~ m/abc/` мағынасының нәтижесі дұрыс болады, бұл $x жолы abc тұрақты мағынаға сәйкес келген жағдайда ғана мүмкін болады. Іздеу және алмастыру `s///` операторының көмегімен жүзеге асырылады. `$x =~ s/abc/def/` құрылымы def жолына abc тұрақты мағынасының бірінші кіруін алмастырады.

Динамикалық тұрақты мағыналар `(??{ Perl коды })` мен құрылымдардың пайда болуымен орындалатын Perl `(?{ Perl коды })` кодына тұрақты мағыналарды қосу үшін іздеу және алмастыру мүмкіндіктері шексіз деңгейге жетті деуге болады. Мысалы, қисынсыз енгізілген деңгейдегі құрылымдарды іздеу мүмкіндігі пайда болды.

Perl тұрақты мағыналарының жалпыға танымдылығы соншалық, олар PHP және JavaScript сияқты басқа тілдерге тікелей қосылды, сондай-ақ мағыналарды компиляцияланатын тілдерде пайдалану мүмкіндігін беретін, іске қосылатын кітапханалар қолданысқа енгізілген.

Perl –дағы операторлардың әртүрлі басымдылықтары бар. С тілінен алынған операторлар өздерінде С-тағы иерархияны сақтап қалды.

Осы тілдегі мықты құрылымдар көп күш жұмсамай тиімді шешімдер мен әмбебап құралдарлы құру мүмкіндігін береді. Perl бағдарламасы үшін жазылғандар жоғары төзімділікпен және пайдалануға дайындықпен ерекшеленетіндіктен, осы құралдарды бұдан әрі пайдалануға болады. Оған қоса, Perl төмен деңгейлі міндеттерге жүгіне алады.

## 8.4. PHP ТІЛІ

**PHP синтаксисінің негіздері.** PHP тілі — өзінің қарапайым және бай функционалдығына орай анағұрлым танымды сценарийлік тілдердің бірі. Оның атауы «PHP Hypertext Preprocessor» сөз тіркесінің рекурсивтік аббревиатурасы болып табылады, ол «гипермәтіннің препроцессоры» болып аударылады. Бұл динамикалық Web-беттерді құру үшін қарапайым және сол уақытта өте мықты тіл болып табылады. PHP Linux, Unix көптеген түрленімдері, MS Windows, Mac OS және басқалары сияқты көптеген операциялық жүйелер үшін қолжетімді. Сондай-ақ PHP-ға Apache, MS IIS, Netscape серверлері сияқты көптеген заманауи Web-серверлерді сүйемелдеу қосылған.

PHP тіліндегі сценарийлер берілген файлдардың . php кеңейтілімі бар және арнайы тегтердің көмегімен PHP-коды кіріктірілетін HTML-құжаттары болып табылады. PHP-сценарийдің операторлары оның құрамындағы HTML-құжат орналастырылған серверде ғана орындалады, яғни Web-бетті жүктеген пайдаланушы браузерде HTML-бетінің бағдарламалық кодын ашып, осы операторларды көрмейді. Пайдаланушы сервер тарапында қалыптастырылған қалыпты HTML-кодты ашады және бұл PHP-скрипттің жұмыс нәтижесі болып табылады.

PHP қосылатын кеңейтілімдердің ядросынан және жинағынан тұрады: дерекқорлармен, сокеттермен, динамикалық графикамен, криптографиялық кітапханалармен, PDF форматының құжаттарымен және басқалармен жұмыс істеу үшін. Оларды кейінгіде қосумен жеке меншік кеңейтілімдерді әзірлеу мүмкіндігі беріледі.

PHP интерпретаторы осы сервер үшін арнайы құрылған dll-модулі арқылы немесе CG-қосымша түрінде Web-серверге қосылады.

PHP тілінің синтаксисі зерделеу үшін айтарлықтай қарапайым және Perl, Java және C тілдерінен өз бастауын алады. Алайда, жоғарыда аталған тілдерден айырмашылығы PHP бастапқыда Web-серверде орындалатын сценарийлерді (Web-қосымшалар) жазу үшін бағдарламалау тілі ретінде әзірленген болатын.

PHP Web-беттердің HTML-кодына сценарийлерді енгізу мүмкіндігін беретіндіктен, бұл динамикалық сайттарды құру міндетін айтарлықтай жеңілдетеді. PHP-да бағдарламаның жұмыс істеуі үшін қандай да бір ауыспалыларды, пайдаланылатын модульдерді және басқаларды сипаттау қажет етілмейді. Кез келген бағдарлама PHP операторынан тікелей басталуы мүмкін:

```
<?php
echo 'Hello, world!';
?>
```

Бұнда echo енгізу операторы қолданылған, одан кейін шығарылатын мағына көрсетілген.

HTML-құжатының мәтініне PHP-сценарийін қосу үшін тегтердің бір құрылымын пайдалануға болады:

- `<?php ... ?>;`
- `<? ... ?>;`
- `<script language="php"> </script>.`

Ауыспалылардың атаулары $ символынан басталады, ауылпалының түрін жариялау қажеттігі жоқ. Ауыспалылардың атаулары тіркелімге сезімтал. Ауыспалылар қосарланған тырнақшаға алынған жолдарда өңделеді. Нұсқаулықтар нүктелі үтірмен аяқталады. PHP /* ... */ символдармен шектелген көпжолды және с // басталатын және жолдың соңына дейін жалғасатын түсіндірмелермен сүйемелденеді.

PHP ауыспалыларды жариялау, сондай-ақ ауыспалылардың өзін жариялау кезінде түрін көрсетуді талап етпейтін динамикалық үлгілеу жүзеге асырылатын бағдарламалау тілі болып табылады. Скалярлық түрлер арасында қайта құру автоматтандырылған түрде (алайда, түрлерді нақты қайта құру үшін мүмкіндіктердің бар болуына қарамастан) жүзеге асырылуы мүмкін. Деректердің скалярлық түрлеріне деректердің толық түрі, заттай түрі, логикалық түрі, жолдық түрі және арнаулы NULL түрі жатады. Скалярлық емес түрлерге «ресурс», массив және объект жатады. NULL түрі белгілі бір мағынасы жоқ ауыспалылар үшін арналған. Сыртқы ресурстарға сілтемелер resource түрінде берілген. Аталған түрдің ауыспалылары, әдетте, файлдар, динамикалық суреттер, дерекқорлардың қорытынды кестелері және басқалары сияқты сыртқы объектілерді басқару мүмкіндігін беретін дескриптор болып табылады. Массивтер сандық және жолдық кілттерді сүйемелдейді және гетерогендік болып табылады, яғни элементтердің мағыналары деректердің әртүрлі түрлеріне жатуы мүмкін. Массивтер басқа массивтерді қоса алғанда, кез келген түрлі мағыналардан тұруы мүмкін. Супержаһандық массивтер (superglobal arrays) PHP-да global негізгі сөзін қолданусыз бастапқы кодтың кез келген жерінде көрінетін алдын ала анықталған массивтер деп аталады. Массивті екі тәсілмен құруға болады:

1) array құрылымының көмегімен:

```
$array_name=array ("keyi"=>"value1",
"key2"=>"value2");
```

2) массивтің элементтеріне мағынаны тікелей беру арқылы:

```
$array name["key1"] = valuel;
```

PHP объектіге бағытталған мүмкіндіктерге қолдау көрсетеді. PHP-дағы класс class сөзінің көмегімен жарияланады. Класстың әдістері және жолдары жалпыға қолжетімді (public), қорғалған (protected) және жасырын (private) болуы мүмкін. PHP объектіге бағытталған бағдарламалаудың мынадай үш негізгі тетігіне қолдау көрсетеді: инкапсуляция, полиморфизм және мұрагерлік. Интерфейстерге қолдау көрсетіледі (implements көмегімен сәйкестікке келтіріледі). Финалдық, абстракт әдістер мен класстарды жариялауға рұқсат беріледі. Класстардың көптеген мұрагерлігі сүйемелденбейді, алайда, класс бірнеше интерфейстерді іске асыра алады. Ата – ана классының әдістерін шақыру үшін parent негізгі сөзі қолданылады. Класстың даналары new негізгі сөзінің көмегімен құрылады, объектінің жолдары мен әдістерін шақыру -> символдарын пайдалану арқылы жүзеге асырылады. Оған жататын әдіс классының мүшелеріне қолжетімділік алу үшін $this ауыспалысы қолданылады.

Тілдің толық сипаттамасын PHP бойынша анықтамалық нұсқаулықта табуға болады.

**PHP-дағы функциялар.** PHP-дағы пайдаланушының функциялары (бағдарламашы анықтайтын) мынамен ерекшеленеді:

- PHP функциялары деректердің кез келген түрін қайтарады;
- Функция берген параметрлерді өзгерту мүмкіндігін береді;
- Параметрлер үнсіз келісу бойынша сүйемелденеді, бір функцияны параметрлері ауыспалы санмен шақыруға болады. PHP – дағы белгілі бір (сипаттама) функциялардың синтаксисі мынадай:

```
function функцияның атауы (параметр[=мағына],
параметр[= мағына]...){
    функцияның денесі
}
```

Функцияны бағдарламаның кез келген жерінде жариялауға болады, бірақ міндетті түрде оны бірінші ет қолданғанға дейін жүзеге асырылуы тиіс. Функциялардың атаулары бірегей болуы және олардың арасында аралықтар болмауы тиіс. Символдардың тіркелімі осы жағдайда есепке алынбайды. Функцияның атауындағы бірінші символ $ бола алмайды. Функцияның бағдарламалық коды (денесін) фигуралы жақшаға алынады. Олар осы функцияның кодына жататын мағыналардың тобын айқындайды. Функциялардың кейбір мағыналарды қайтарғаны қажет болған жағдайда

оның денесінде оң жақта нені қайтару қажеттігі көрсетілген оператор возврата return кері қайтару операторы көзделеді. Кез келген мағына қайтарылатын өлшем бола алады: қарапайым мағына, ауыспалы атау немесе есептелінетін мағына. Return операторы функцияның кодында бірнеше рет кездесуі мүмкін. Return операторы пайдаланылмаса немесе оның оң жағында қайтарылатын өлшем көрсетілмесе, функция ешнәрсені қайтармайды. Мысалы, бұдан әрі берілген функция өзінің параметрлерін есептейді және кері қайтарады:

```
function S rec($w $h) {
$S=$w* $h;
return $S;
}
```

PHP- да көптеген кіріктірілген функциялар бар. Олардың кейбірі PHP-ды қарапайым орнату кезінде қолжетімді, басқалары PHP модулінің ерекше, алайда, күрделі емес теңшеулерін қажет етеді. Кейбір функцияларға қолжетімділік алу үшін, мысалы дерекқормен, графикалық ақпаратпен жұмыс істеу үшін арнаулы кітапханаларды іске қосу қажет.

PHP-да графикалық бейнелермен жұмыс істеу үшін функциялар кітапханасы бар. Аталған функциялардың көмегімен графикалық файлдардан мынадай әртүрлі түрлендірулерді жүзеге асыруға болады: масштабтау, бұрылыс, бір бейнені басқаның үстіне салу, мәтінді суретке салу және басқалары. Сервердің тарапынан бейнелерді құру және өңдеу өте маңызды, мысалы, Web-беттерге кіру баннерлерін және есептегіштерін құру кезінде. Графикамен жұмыс істеу үшін графикалық функциялардың кітапханасын іске қосу қажет. Әртүрлі графикалық форматтар үшін (gif, png, jpeg және басқалары) PHP-да арнаулы функциялар бар.

Қолжетімділікке қарай, ауыспалылар жаһандық және жергілікті болып бөлінеді. *Жаһандық ауыспалылар* бағдарламаның кез келген жерінен, оның ішінде функциядан да қолжетімді. *Жергілікті ауыспалылар* функцияның ішінде белгіленген және функцияның ішінен ғана қолжетімді. Үнсіз келісу бойынша функцияның ішінде қолданылатын барлық ауыспалылар жергілікті болып табылады және оларды функцияның денесінен тыс жерде өзгертуге болмайды. Тіпті оның атауы бағдарламаның басқа жерінде жарияланған жаһандық ауылпалының атауына сәйкес келсе де, функциядан тыс жерде жергілікті ауыспалыларды өзгерту жаһандық ауыспалыларға өз әсерін тигізбейді.

PHP бағдарламаларында жергілікті және жаһандық ауылпалылардан басқа *статикалық ауыспалыларды* да пайдалануға болады. Функция ішіндегі ауыспалыны статикалық деп жарияласақ,

статистикалық ауыспалы қолданылған пайдаланушы функциясының жұмыс істеу мысалы берілген:

```
<?php
function f count(){
   static $counter;
 $counter++;
 echo
 $counter;
}
 for ($i = 0;  $i++<9;) f count();
?>
```

Бағдарлама осы мысалда

123456789 жолын шығарады

Егер статикалық ауыспалы хабарламасын алып тастасақ, мынадай жол шығады:

```
111111111
```

Бұның себебі f_count () функциясын әрбір шақырған кезде функция ішіндегі ауыспалылар инициалданады, оған қоса, ауыспалыларға 0 мағынасы беріледі. Бұдан әрі осы мағына инкрементирленеді және echo операторымен енгізіледі.

PHP –дағы кейбір бағдарламалардың оларды бірнеше рет және әртүрлі жобаларда пайдалану мүмкіндігін беретін әмбебап сипатқа ие болады. Осындай жағдайларда жеке файлдарда кодтың фрагменттерін (функциялардың, класстардың және басқалардың анықтамалары) сақтауға, ал нақты сценарийге оларды кіріктірілген include (<параметр>) функциясының көмегімен қосуға болады. Параметр түрінде файлдың аты немесе мазмұнын include() функциясын шақыру орнына қою қажетті файлдың URL-мекенжайы жатады. Қосылатын файл <?php и ?> дескрипторларына қосылуы тиісті PHP-кодтан тұрады.

Алдын ала анықталған PHP ауыспалылары. PHP-да ғаламдық көру облысы бар алдын ала анықталған аттау (массив) түріндегі ауыспалылар бар. Олар PHP тіліндегі кез келген сценарийден қолжетімді.

Пайдаланушы сауалын өңдеу кезінде PHP қалыптастыратын алдын ала анықталған ауыспалылардың негізгі массивтерін атап өтейік:

- $GLOBALS — барлық ғаламдық ауыспалылардан тұрады. $GLOBALS массивіндегі ғаламдық ауыспалылардың атаулары оның го символдық индекстері (кілттері) деп аталады. Осыған орай, ғаламдық ауыспалыға қолжетімділік алу үшін, мысалы, $myvar-ға, $GLOBALS[«myvar»] мағынасын пайдалану жеткілікті болып табылады;

- $_GET — URL-адресінің бір бөлігі түрінде PHP-дағы сценарийге берілген деректерден, сондай-ақ GET әдісімен берілген HTML-нысандарының деректерінен тұрады;
- $_POST — POST әдісімен HTML-нысандарынан PHP-дағы сценарийге берілген деректерден тұрады;
- $_COOKIE — cookie тетігі арқылы ағымдағы сценарийге берілген деректерден тұрады;
- $_ENV — ауыспалы қоршаудан тұрады, осы массивтің мазмұны операциялық жүйеге байланысты;
- $_FILES — браузердің көмегімен POST әдісімен жүктелген файлдардың атауларынан тұрады;
- $_SERVER —Web-сервер орнатқан немесе ағымдағы сценарийді орындау қоршауымен тікелей байланысты ауыспалылардан тұрады. Бұл ақпарат қандай Web-сервердің пайдаланылуына байланысты;
- $_SESSION — ағымдағы сценарийде қолжетімді барлық ауыспалы сеанстардан тұрады;
- $_REQUEST — $_GET, $_POST және $_COOKIE массивтерінде орналасқан барлық ауыспалылардан тұрады.

Ғаламдық массивтер мазмұнын қарау үшін var_dump() функциясын пайдалануға болады, мысалы var_dump ($_ SERVER) .

Егер, мысалы, серверлік сценарийден HTML-нысанына POST әдісімен name = "myname" атрибутымен элементтің мазмұны берілсе, онда оны сценарийде $_POST[' myname '] ретінде алуға болады.

**Нысандарды PHP-сценарийінің көмегімен өңдеу.** Бұрын HTML-нысандарының деректерін серверге GET және POST әдістерінің көмегімен беруге болатындығы туралы айтылған болатын. GET әдісі сценарийге берілетін деректерді өңдеушінің URL-адресіне қосады және оларды тиісінше браузердің адрестік жолында көруге болады. Кейде бұл орынсыз, осыған орай, POST әдісі анағұрлым көбірек таралған. Сонымен қатар, POST әдісімен көлемі бойынша үлкен деректерді беруге болады. PHP пайдаланушыдан алынған деректерді пайдаланылатын деректерді беру әдісіне байланысты алдын ала анықталған $_GET немесе $_POST ауыспалылардың бірінде орналастырады.

Пайдаланушы пернелер тақтасының көмегімен енгізген немесе клиенттік/серверлік сценариймен құрылған деректердің Web-серверіне жіберу үшін қолданылатын стандартты құрал контейнерлік <FORM> тегімен құрылатын HTML-нысаны болып табылады. <FORM> тегінің action атрибутында осы деректерді қабылдауы және қандай да бір тәсілмен оларды өңдеуі тиісті файлдың сценарий қоса берілген URL-адресі көрсетіледі.

<FORM ac^оп=и^_адрес> контейнерінде пайдаланушы интерфейсінің (деректерді енгізу жолдары, жолақтар, ауыстырып-қосқыштар, түймелер) әртүрлі элементтері және ең көп дегенде нысанның элементтерінде болатын деректерді серверге жіберу үшін <INPUT type=submit> тегімен құрылатын бір арнаулы түйме болуы мүмкін.

Клиенттен деректерді алып, серверлік сценарий оларды файлда немесе дерекқорда сақтай алады, қандай да бір тәсілмен өңдей алады және алынған нәтижеге байланысты клиентке кейбір хабарламаларды жібере алады, HTML-құжатын және басқаларды құра алады.

Деректерді беру әдісі (GET немесе POST) <FORM> тегінің method атрибутында көрсетіледі, мысалы:

```
<FORM action= "http://localhost/php form post.php" method=POST>.
```

<FORM> тегінің action параметрінде көрсетілген URL-адрес осы нысаннан деректерді өңдеу үшін арналған PHP-сценарийінің орналасқан жерін көрсетеді. Осы жағдайда пайдаланушының компьютерінде Интернетке шығусыз деректерді жергілікті өңдеу жүзеге асырылады. Нысан элементтерінің тегтерінде (<INPUT>, <TEXTAREA> және басқалары) name (элементтің аты) атрибутының мағынасы көрсетілген болса, онда серверге мынадай түрдегі тиісті жұптар жіберілетін болады:

```
Элементтің_аты = элементтің_мағынасы.
```

Бұл ретте элементтің_аты (name атрибутының мағынасы) қандай әдіспен деректердің берілгеніне қарай $_GET немесе $_POST массивінің символдық индексіне (кілтіне) айналады. Массивтің индексі бойынша осы элементтің мағынасын, яғни нысанның тиісті элементінің мағынасын алуға болады. Мысалы, myelement атаулы нысан элементінің мазмұнын алу үшін пайдаланылған деректерді беру әдісіне қарай $_POST['myelement'] немесе $_GET['myelement'] массивіне жүгінуге болады. Беру әдісіне қарамай нысанның деректерін алу үшін супержаһандық $_REQUEST массивін пайдалануға болады. Символдық индекс ретінде нысан элементінің атын көрсетіп, осы элементтің мағынасын алуға болады.

Клиенттің браузеріне қандай да бір нысанды ұсынатын және пайдаланушы енгізетін деректерді көрсететін файлды мысал етейік. Осылайша, клиенттен алынған деректерге қолжетімділік алу мүмкіндігі көрсетіледі. Түймеге басу (submit түрінде) арқылы пайдаланушы енгізген деректер серверге жіберіледі:

```
<FORM action= "http://localhost/myforma.php" method=POST>
```

```
<INPUT type="text" name="message" value="Hello,
world!">
<INPUT type="submit"
name="submit" value="Жіберу">
</FORM>
<?php
/* Массивтің мазмұнын шығару $ POST
*/ foreach($ POST as $key=>$value){
echo "$key=$value <br>";
}
?>
```

Нәтижесінде нысандағы элемент атауларының тізімі және олардың мағыналары көрсетіледі:

```
message = Hello, world!
submit value=жіберу
```

<FORM> тегінің method атрибутында GET мағынасын көрсетсек, деректерді алу үшін супержаһандық $_GET немесе $_REQUEST массивтерін пайдалану қажет.

Браузерге myforma. Php файлын жүктеген кезде браузердің терезесінде нысанның элементтері көрсетіледі. Осы файлдың HTML-кодын көруге болады, ал PHP-код $_POST массивінде ештеңе болмағандықтан, ешқандай нәтиже шығармайды. Енгізу жолына қандай да бір деректер енгізілгеннен кейін немесе submit тәріздес түйме басылғаннан кейін осы ақпарат myforma. Php файлының серверіне беріледі. Осылайша, клиенттің браузері myforma. Php файлын қайта сұратады. Енді әрқашан кез келген серверлік PHP-сценарийіне қолжетімді $_POST массиві бос емес. Сервер клиенттің браузеріне бос жолды нысанды, ал одан кейін echo операторы PHP-сценарийге жіберетін $_POST массивінің индекстері мен мағыналарын қайтарады.

Қарастырылған мысалдан нысан тікелей HTML кодының көмегімен құрылғанын көруге болады. Бұны echo операторын пайдалана отырып, PHP-сценарийінің көмегімен істеуге болады. Осы сценарийде echo операторының көмегімен браузердің терезесінде берілген белгілерге сәйкес нысан көрсетіледі:

```
<?php
echo '<FORM action="http://localhost/Форма.php"
method=POST>
<INPUT type="text" name="message" value="">
<INPUT type="submit"
name="submit" value="жіберу">
```

```
</FORM>';
foreach($ POST as $key=>$value){
echo "$key=$value <br>";
}
?>
```

Нысанда көптеген таңдауларды жасау (яғни бірнеше мағыналарды бір уақытта көрсету) мүмкіндігін беретін элементтер жиынтығы болуы мүмкін. Әдеттегі мысал — әрбірі <INPUT type=checkbox> тегімен берілетін жалаушалар жиынтығы, ал пайдаланушы олардың еркін комбинациясын белгілей алады. Жалаушалардың бір жиынтыққа тиесілдігі бірдей атауды—name атрибутының мағынасын қамтамасыз етеді. Бұл жағдайда PHP-сценарийде пайдаланушы бір атаулы элемент жинағында нені таңдағанын талдауды ұйымдастыру оңай болатындықтан, массив түріндегі name атрибутының мағынасын қою ыңғайлы болады.

**PHP-ның дерекқормен өзара іс-қимылы.** PHP –да Microsoft SQL Server, MySQL, Oracle, PostgreeSQL, Informix, Sybase, IBM DB2 сияқты көптеген ДҚБЖ жұмысы үшін арнаулы функциялардың жиынтықтары бар. Оған қоса, PHP барлық дерлік реляциялық ДҚБЖ сәйкес келетін ODBC (Open Database Connectivity — дерекқорға ашық қолжетімділік интерфейсі) стандартына қолдау көрсетеді. Дерекқорды басқару жүйесі PHP-ға тәуелсіз түрде орнатылады. Онымен PHP-сценарийінің көмегімен жұмыс істеу үшін ДҚБЖ деректеріне сәйкес келетін функциялар кітапханасын орнату қажет. MySQL ДҚБЖ пайдалану кеңінен таралған.

MySQL — еркін таратылатын, айтарлықтай жылдам, сенімді және пайдаланымда қарапайым дерекқорды реляциялық басқару жүйесі. MySQL онша үлкен емес жобалар үшін жарамды. MySQL-да дерекқормен жұмысты айтарлықтай жеңілдету мүмкіндігін беретін, осы ДҚБЖ жұмыс істеу үшін арналған әдеуір танымал көзбен шолу интерфейсі — PhpMyAdmin бар. Тестілік режімде дерекқормен жұмыс командалық жолға командаларды енгізу түрінде ғана көрінеді. PhpMyAdmin – да деректермен жұмыс істеудің негізгі SQL-функцияларының көбі интуициялық түсінікті интерфейстер мен іс-әрекеттерге жатады.

Сценарийдің дерекқормен өзара іс-қимылы кезінде әдетте іс-қимыл реттілігі мынадай болады:

1. Дерекқор сервері іске қосылғаннан кейін дерекқор серверіне қосылу.

2. ДҚБЖ орындауы тиісті SQL-сұрау салуын беру. SQL-сұрау салуы SQL тілінде қалыптастырылады, жол түрінде ресімделеді және

параметр түрінде арнаулы PHP-функциясымен беріледі. Қателіктер мен қандай да бір проблемалар пайда болған жағдайда ДҚБЖ SQL-сұрау салуын орындайды. Егер SQL-сұрау салуы деректерді қайтаруды болжаса, онда олар арнаулы PHP-функциясының көмегімен алынады.

3. Бұрын берілген SQL-сұрау салуына сәйкес деректерді ДҚБЖ-дан алу. Арнаулы функциялардың көмегімен деректерді алуға және бұдан әрі өңдеуге болады, мысалы, қажетті мәтінді экранға шығаруға болады.

4. Алынған деректерді өңдеу. Алынған деректерді форматтауға, талдау жасауға, қайта құруға және басқаларын жүзеге асыруға болады.

5. Қайта сұрау салу болжанбаған жағдайда дерекқорды сөндіру. Ұзақ мерзім ішінде жұмыс істеу тоқтатылған кезде дерекқорды сөндіру ұсынылады.

*Дерекқорға қосылу.* Дерекқорға қосылу мынадай деректер параметрлер ретінде берілетін PHP-дың арнаулы функцияларының көмегімен жүзеге асырылады:

- *орналасқан жері* (host) — көптеген серверлер үшін дерекқор домендік атау немесе дерекқор орналасқан компьютердің және PHP модульдің IP-адресі болып табылады, онда localhost ғана көрсетуге болады. MS SQL Server үшін оның атауын көрсетеді. Қажеттігінше, қос нүкте арқылы дерекқордың серверімен тыңдалатын портты көрсетуге болады;
- *тіркеу атауы* (username) — ДҚ-ға қолжетімділік ала алатын пайдаланушының атына сәйкес келетін символдардың жолы (оны дерекқордың әкімшісі хабарлайды);
- *пароль* (password) — тіркеу атауымен бірге дерекқорға қолжетімділікті қамтамасыз ету үшін қызмет көрсететін символдар жолы (оны дерекқордың әкімшісі хабарлайды);
- *дерекқордың атауы* — ДҚБЖ әрбіріне бірегей атау берілетін бірнеше дерекқорға қолдау көрсете алады. Нақты жұмыс істегіңіз келетін дерекқордың атауын көрсету қажет.

Дерекқор серверіне қосылғаннан кейін және қажетті дерекқорды таңдағаннан кейін онымен әртүрлі манипуляциялар жасауға болады, олар SQL тіліндегі мағыналармен беріледі.

Дерекқорға жол параметрі түріндегі SQL-сұрау салуы PHP-ның арнаулы функциясымен беріледі, ол өз кезегінде оны ДҚБЖ береді. Дерекқорды басқару жүйесі осы сұрау салуды орындайды, ал PHP-функциясы сұрау салуды орындау нәтижесі болып табылатын деректердің уақытша кестесіне көрсеткішті қайтарады.

Дерекқорға қолжетімділік дерекқорларды басқару жүйесімен қамтамасыз етіледі. Әртүрлі ДҚБЖ үшін арналған кітапханалар функциялардың әртүрлі жиынтықтарынан құралады. Дерекқорды қосымшада пайдалану алдында қандай ДҚБЖ әзірленетін Web-қосымшаны орналастыратын хостингілік компаниямен сүйемелденетінін білу қажет. Оған қандай дерекқормен жұмыс істеу функцияларын құрылатын сценарийлерде пайдалануға болатындығы байланысты болады. SQL тілінің көмегімен дерекқорды құру және жою, дерекқорда берілген кестелерді құру және жою, сондай-ақ кестелердегі деректерді қосу, редакциялау, жою және таңдау мүмкіндігін беретін дерекқорға сұрау салулар қалыптастырылады. SQL-сұрау салуларының көмегімен дерекқорды әкімшілендіруге болады: дерекқорды пайдалануға олардың құқықтарын (артықшылықатрын) белгілеп, пайдаланушыларды анықтау (аты және паролі).

PHP – да әртүрлі ДҚБЖ басқарылатын дерекқорға қосылу кезінде әртүрлі функциялар пайдаланылады:

1) MS SQL Server

```
=mssql_connect{серверінің атауы, username, pass-
word};
$db=mssql_select_db(ДҚ атауы, $connect);
```

2) MySQL:

```
$connect=mysql_connect(host,          username,
password);
$db=mysql_select_db($connect, ДҚ атауы);
```

<...>_connect() дерекқор серверімен қосылысты жүзеге асырады және қосылыс орын алған және false — орын алмаған жағдайда қосылыс сәйкестендіргішін кері қайтарады. Бұл ретте MySQL үшін осы функцияның бірінші параметрі кезінде host (домендік атау немесе компьютердің IP-адресі) көрсетіледі, ал MS SQL Server үшін — сервер тіркелген кезде орнатылатын дерекқор серверінің атауы. Функцияның көмегімен қайтарылатын қосылыс сәйкестендіргіші сценарийде, оның ішінде дерекқорды таңдау, сұрау салуларды құру кезінде пайдаланылады.

<...>_select() бірінші функциямен қайтарылған $connect сәйкестендіргіші көрсететін серверде нақты дерекқорды таңдайды. Осы функция операция ойдағыдай аяқталған және false — аяқталмаған жағдайда true қайтарады. Талап етілетін дерекқор әлі құрылмаған болса, онда оны құру қажет. Осылайша, MS SQL Server-де дерекқорды (тым болмаса бос) Enterprise Manager утилитінің немесе интерфейстің терезесіне енгізуге немесе PHP-сценарийінің көмегімен қалыптастыруға және орындауға болатын тиісті SQL мағынасының көмегімен құруға болады.

MySQL-да командалық жолда қажетті SQL-мағыналарын енгізуге немесе тиісті PHP-сценарийін жазуға, ал одан кейін оны толтыруға болады. Дерекқордан сөндіруді тиісті іске қосылу сәйкестендіргіші берілетін функциямен жүзеге асырылады. Мысалы, `mysql_close($connect)` — MySQL, `mssql_close($connect)` үшін — MS SQL Server үшін.

*Дерекқорға сұрау салуларды беру.* Дерекқор серверіне қосылу жүзеге асырылғаннан кейін және талап етілетін дерекқор таңдалғаннан кейін SQL командаларын пайдалану арқылы әртүрлі манипуляцияларды орындауға болады. SQL-сұрау салуы жол параметрі ретінде PHP-ның арнаулы функциясымен ДҚБЖ-ға жіберілгені бұрын атап өтілген болатын. Дерекқорды басқару жүйесі осы сұрау салуларды орындайды, ал PHP-функциясы көрсеткішті сұрау салуды орындау нәтижесі болып табылатын деректердің уақытша кестесіне көрсеткішті қайтарады. Әртүрлі ДҚБЖ үшін сұрау салуды беру функциялары бір-біріне ұқсас болып келеді, алайда, кішкене ерекшеліктері бар:

1) `MS SQL Server:`

```
Courier New
mssql_query(SQL сұрау салуы, қосылыс сәйкестендіргіші);
```

2) `MySQL:`

```
Mysql_query (SQL сұрау салуы, қосылыс сәйкестендіргіші);
```

Серверге қосылыс және дерекқорды таңдау қандай да бір себептерге байланысты сәтсіз өтуі мүмкін. Алайда, сценарийді орындау жалғасады, бұл іс-әрекет мағынасыз болуы және қателіктер туралы хабарламалардың көшкін тәріздес ағынды тудыруы мүмкін. Құрамына аталған хабарлама кіретін және бұдан әрі сценарийдің орындалуын тоқтататын `exit` функциясы бұның алдын алуға көмектеседі. Mysql_ `error()` пайда болған қателіктің сипаттамасын қайтарады. Әртүрлі ДҚБЖ үшін дерекқорға қосылу көптеген жағдайда ұқсас функцияларды жүзеге асырады. Алайда, түбегейлі ерекшеліктер де болуы мүмкін, мысалы, PostgreeSQL серверіне қосылу кезінде.

*Сценарийде деректерді өңдеу.* Бір немесе бірнеше кестеден қажетті деректерді таңдауды SELECT негізгі сөзінен (оператор) басталатын SQL-сұрау салуын орындайды. Деректерді таңдау критерийлері WHERE негізгі сөзінен кейін SQL-сұрау салуы түрінде қалыптастырылады. Осындай сұрау салуды орындау нәтижесі уақытша кесте, сұрау салудың талаптарын қанағаттандыратын деректер жоқ бос кесте болуы мүмкін. Сұрау салуды орындайтын PHP функциясы нәтижесінде алынған көрсеткішті кері қайтарады. Осы көрсеткіш арнаулы функциялардың көмегімен деректерге қолжетімділік алу үшін PHP-сценарийінде пайдаланылады.

Кейде уақытша кестеден алынған деректер массивке жазылады, одан кейін ол өңделеді. Егер кесте жолдарының мағыналарын өзгерту, жаңа жазбаны қосу немесе жазылған өшіру қажеттігі туындаса, онда тиісті SQL- сұрау салуымен жолды қалыптастырады, одан кейін оны сұрау салуды (мысалы, `mssql_query()` орындайтын функция параметрі ретінде береді. Осы функцияның параметрлері ретінде тиісті параметрлер көрсетілген SELECT SQL- сұрау салуын беру қажет. Барлық кестені таңдау үшін қолданылатын PHP-оператор мынадай болады:

```
$result=mysql query("SELECT * FROM staff
", $link).
```

`Mysql_query()` функциясына сұрау салу ойдағыдай жасалған кезде сұрау салу нәтижесінің сәйкестендіргішін кері қайтарады және аталған сәйкестендіргіш нәтижесін өңдеу үшін басқа функцияларға беруге болады. Осы ауыспалыда матрица түріндегі барлық кесте сақталады.

Матрицадан жеке жазбаларды шығару үшін `mysql_fetch_array()` (fetch — алып шығару) функциясы қолданылады.

Осы функцияның параметрінің ретінде `$result` атауын беру қажет. Кестеден бірінші жазбаны таңдауға арналған PHP-оператор мынадай болады:

```
$myrow=mysql fetch array($result)
```

PHP және XML өзара іс-қимылы. PHP тілін XML-құжаттарымен жұмыс істеу үшін пайдалануға болады. PHP-да бұл үшін XML-деректерін өңдеудің екі түрлі стандартын іске асыратын екі модуль бар: SAX (Simple API for XML) және DOM (Document Object Model).

SAX стандарты W3C стандарты болып табылмайды және олардан деректерді алу үшін XML-құжаттарды сипаттау әдісін сипаттайды, яғни осы XML-құжаттарын өңдеу әдісі тек XML-құжатынан алынған деректерді оқу мүмкіндігін ғана береді дегенді білдіреді. Оның көмегімен XML-құжаттарын құру және өзгерту мүмкін емес.

XML-деректерін өңдеу үшін қолданылатын басқа стандарт - DOM —*W3C стандарты* болып табылады, оның ерекшелігін консорциумның сайтында табуға болады. SAX-тан ерекшелігі, осы әдіс анағұрлым ыңғайлы түрде XML-құжатын объектілердің ағашы ретінде алып, XML-деректерімен кез келген операцияларды жүргізу мүмкіндігін береді. Осы стандартты іске асыратын модуль DOM XML деп аталады. Ол PHP модульдерінің негізгі жиынтығына кірмейді, бірақ осы модульдің .API кеңейтілімі ретінде орнатылуы мүмкін. DOM XML модулі XML-құжаттарын өңдеуге арналған мықты және пайдалануда ыңғайлы құрал болып табылады.

DOM XML модулі DomNode, DomDocument, DomElement, DomText және DomAttribute сияқты PHP-дағы бірнеше класстарды айқындайды, олардың көбі DOM стандартының ядросынан шығады. Шамамен алғанда барлық класстар үшін (оның ішінде, бұрын аталғандар үшін) DomNode классы басты болып табылады, осыған орай, оның қасиеттері және әдістері барлық қалған класстарға мұра ретінде беріледі.

Егер XML-құжатын қарасақ, онда DomDocument классына осы құжат сәйкес келеді, DomElement классына —әрбір XML-тег, DomAttribute классына — тегтердің атрибуттары, DomText классына — XML-элементтерінің мазмұны сәйкес келеді. Сол уақытта DomNode классына XML-құжатының әрбір аталған элементтері сәйкес келеді.

**PHP-дағы сеанстар.** PHP- дағы сеанстар (сессия) — сайттың басқа беттері үшін бір сайт шеңберінде әртүрлі сервер беттерінде құрылған деректердің қолжетімділігін қамтамасыз ету мүмкіндігін беретін механизм.

Сеанстар технологиясы мүмкіндіктерді жабады және Web-беттер арасында деректер алмасу бойынша cookie шектеулерін жеңеді. Сеанс қажеттігінше ашылып, жабылады. Осы оқиғалар арасындағы уақыт аралығында әртүрлі файлдарда орналасқан және осы сеансқа қосылған барлық PHP-сценарийлер осы сайттың басқа сценарийлерінде құрылған ауыспалыларға қолжетімділік беріледі. Сеанс механизмі беретін мүмкіндіктерді пайдалану үшін оны құру немесе оған қосылу қажет. Жұмыс істеп тұрған сеанс шегінде сеанс ауыспалыларын орнатуға болады. Бұл үшін супержаһандық $_SESSION массиві пайдаланылады.

Әрбір Web-қосымша әдетте сценарийден тұратын бернеше беттерден тұрады. Әрбір бетті жүктеген кезде осы сценарийлердегі ауыспалылар тазартылады және беттер арасында дерек алмасу үшін параметрлерді беру механизмін немесе дерекқорды немесе cookie-файлдарындағы ақпарат жазбасын пайдалануға болады. Алайда, анағұрлым ыңғайлы әдіс сеанстарды пайдалану болып табылады.

Сеансты ашу кезінде әрбір пайдаланушы үшін серверде осы пайдаланушыға жататын параметрлер жиынтығынан құралған белгілі бір орта қалыптасады. Мысалы, онда интерфейстің қандай да бір теңшеулері, жеке сервистердің конфигариялары сақталуы мүмкін. Тиісті деректер супержаһандық `$_SESSION` массивінде сақталады.

Әрбір сеансқа сеанс қайта ашылғанда пайдалануға балатын бірегей сәйкестендіргіш беріледі. Мысалы, осы сәйкестендіргішті cookie-файлында сақтауға болады, одан кейін

сайтқа қайта кірген кезде Сізге қайтадан интерфейс пен сервистердің жұмысын қайта жөнге салу қажеттігі болмайды. Сеансты ашу үшін параметрлері жоқ `session_start()` функциясы пайдаланылады. Бұдан кейін `$_SESSION` массиві қолжетімді болады және сіз одна сеанс аяқталғанға дейін Web-қосымшаның кез келген бөлігінен қолжетімді болатын өз деректеріңізді сақтай аласыз. Сіз сценарийдің ауыспалыларын пайдалану арқылы беттерге кіру есептегіштерін құру үшін сеанстарды пайдалана алмайсыз, олар кез келген жаңа жүгіну кезінде нөлденеді, алайда, аталған проблеманы `$_SESSION` массивінде есептегін мағынасын сақтау арқылы шешуге болады.

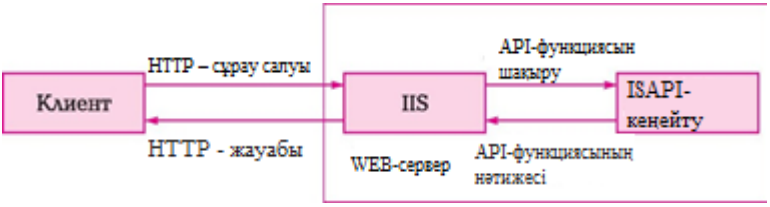
## 8.5. ISAPI ҚОСЫМШАСЫ

ISAPI — IIS –пен (Microsoft Internet Information Server) өзара іс-қимыл жасайтын және оның мүмкіндіктерін кеңейтетін, қосымшаларды жазуға арналған Web-сервер ұсынатын интерфейстер жинағы болып табылады. ISAPI қосымшасы DLL динамикалық жүктемесінің кітапханалары түрінде іске асырылады, бұл олардың өнімділігі мен масштабталуын арттыру мүмкіндігін береді.

ISAPI қосымшасы ISAPI кеңейтіліміне және ISAPI фильтрлеріне шартты түрде бөлінеді.

*ISAPI кеңейтілімі* CGI бағдарламасы сияқты функцияларды орындайды: әдетте олар клиенттік HTTP- сұрау салуларды өңдеу және HTTP-жауабын қайтару (8.5 -сурет) үшін қолданылады.

Қашықтықтан басқаратын пайдаланушы жүгінген кезде CGI кеңейтіліміне Web-серверде өзінің адрестік кеңістігінде жеке процесс ретінде орындалатын бағдарлама іске қосылады. Процесті іске қосу үшін уақыт қажет болады және серверге пайдаланушылардың көп саны жүгінген жағдайда, CGI бағдарламалары оны толықтай жүктей алады. Нәтижесінде сервердің жауап қайтару уақыты артады, бұл пайдаланушылар алдында қиындық тудырады.




8.5 - сурет. ISAPI-кеңейтілімдерінің жұмыс істеуі

CGI –мен салыстырғанда ISAPI интерфейсі шеңберінде өзара іс-қимыл жасау кезінде, кезекті сұрау салу түскен кезде Web-сервер жұмыс істейтін негізгі процес шеңберінде жаңа ағынды құруға бастамашылық етеді. Жадыда әрқашан тиісті DLL кітапхананың бір көшірмесі ғана сақталады, сондықтан бірнеше пайдаланушылардың ISAPI кеңейтіліміне бір уақытта жүгіну кезінде жүйелік ресурстар анағұрлым үнемді түрде шығындалады және оған қоса кеңейтілімнің қосымша жүктемесіне уақыт жұмсалмайды.

ISAPI қосымшаларын бағдарламалау CGI бағдарламасын құрудан басқа анағұрлым күрделі міндет болып табылады; осы қосымшалар әрқашан мультиміндеттік режимде жұмыс істейтіндіктен сыни ресурстарға қолжетімділікті синхрондауды қамтамасыз ету қажет. ISAPI бағдарламашыларға Web-серверге анағұрлым тығыз интеграцияланған Web- қосымшаларды әзірлеу мүмкіндігін береді. Операциялық жүйе тарапынан ағынды құру процесті құрумен салыстырғанда анағұрлым шығыны аз операция болып табылатындықтан, осындай қосымшалар іс жүзінде анағұрлым масштабталған болып табылады. Сонымен қатар, барлық код бір процесте жұмыс істеуіне, осы жағдайда операциялық жүйе шеңберінде бірыңғай адрестік кеңістік пайдаланылуына байланысты Web-сервер мен Web-қосымшаның өзара іс-қимылы жеңілдетіледі. ISAPI қосымшаларына CGI бағдарламасымен салыстырғанда анағұрлым жете жөнге салуға тура келеді. ISAPI қосымшасы Web-серверінің адрестік кеңістігінде жұмыс істейді, ISAPI қосымшасындағы қателік Web-серверінің жұмысын апаттық аяқтауды шақыра алады, ал CGI бағдарламаларын пайдалану кезінде еқіталай.

ISAPI-кеңейтілімі IIS-серверге жіберілетін URL-адресте нақты көрсетіледі. Мысалы:

```
http://localhost/myapi/myapi.dll
```

ISAPI-кеңейтілімін бір компонентке әртүрлі міндеттерді орындау мүмкіндігін беретін параметрлермен шақыруға болады.

IIS серверінің 4.0 – нұсқасы және одан жоғары нұсқасы жеке адрестік кеңістікте ISAPI бағдарламаларын жүктеу мүмкіндігін береді. Осындай әдіс сервердің жұмысын бәсеңдетеді, бірақ жаңа бағдарламаларды жөнге салу үшін пайдаланылуы мүмкін. Жеке адрестік кеңістікке жүктелген ISAPI бағдарламаны апаттық аяқтау Web-сервердің толық тоқтауына әкеледі.

*ISAPI фильтрлері* ISAPI кеңейтілімі сияқты DLL динамикалық жүктелімінің кітапханалары түрінде іске асырылған және HTTP хатамасының деңгейінде браузер мен Web-сервер арасында барлық деректерді (8.6 - сурет) бақылау мүмкіндігі беріледі. Фильтр пайдаланушының сұрау салуы серверге түскен және оған жауап берілгенге дейін өңдеуге тікелей қатысады.


8.6 - сурет. ISAPI-фильтрлерінің жұмыс істеуі

Ол сұрақты және жауапты түрлендіре алады, сұрау салуды өңдеу үшін жауапты адамның адресін өзгерте алады, бірақ сұрау салудың соңғы алушысы болып табылмайды.

ISAPI-фильтрлер ешқашан айқын түрде шақырылмайды — IIS сұрау салуды орындау кезінде белгілі бір оқиғаларға жауап алу үшін оларға жүгінеді. Осындай оқиғаларға мыналар жатады:

- Клиент ұсынған тақырыпты сервердің өңдеуді жалғастыруы;
- Сервердің клиентті аутентификациялау рәсімін аяқтауы;
- Сервердің логикалық URL-адресті физикалық адреспен салыстыра тексеруі;
- Деректерді клиенттен серверге жіберуді бастау кезі;
- Деректерді клиенттен жіберуді аяқтау, бірақ оларды серверде өңдеуді бастағанға дейін;
- Сервердің ақпаратты журналда тіркеуі;
- Сеанстың аяқталуы.

Осыған орай, оларды динамикалық қайта кодтау және деректерді шифрлау, пайдаланушыларды аутентификациялаудың қосымша рәсімдерін құру, сервердің ресурстарын пайдалану туралы статистикалық ақпаратты жинау және басқалары сияқты міндеттерді шешу үшін қолдануға болады.

ISAPI-кеңейтілімдері MFC (Microsoft Foundation Class Library) кітапханасының ISAPI-класстарын пайдалану арқылы жиі құрылады. Бұл ISAPI-кеңейтілімдерін әзірлеуді айтарлықтай жеңілдетеді.

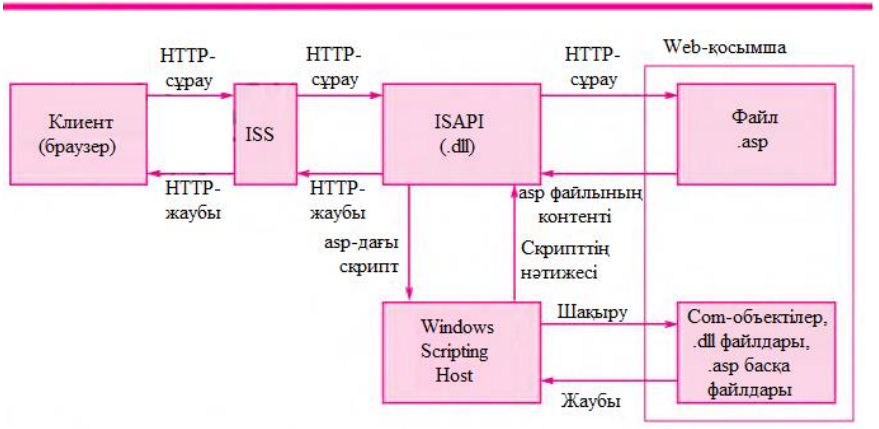
## 8.6. ASP ТЕХНОЛОГИЯСЫ

*ASP* (Active Server Pages — белсенді серверлік беттер) - динамикалық интерактивтік Web-қосымшаларды әзірлеуге және орындауға арналған серверлік орта.

ASP — Web-қосымшаларды құру үшін 1996 жылы *Microsoft* компаниясымен ұсынылған технология. Қосымшаның интерфейсін құру жағынан ASP негізгі идеясы Web- бетте

Web-сервер интерпретациялайтын код фрагментінің болуында, яғни қарапайы Web-беттерге арнайы басқару элементтері енгізіледі. Бұл қарапайым қосымшалар жобаланатын және іске асырылатын үлгіге Web-қосымшаларды жобалауды және іске асыруды жақындатқан сервер тарапынан беттерді динамикалық құру технологиясы.

ASP архитектурасы 8.7 – суретте берілген. ASP файлы . asp кеңейтілімен берілген мәтіндік файл болып табылады. Осы файлда мәтіндік деректер, HTML тілінің тегтері және и серверлік сценарийлер (VBScript, JScript — *Microsoft* компаниясының скриптілік бағдарламалау тілдері) болуы мүмкін. ASP файлын өңдеу басынан бастап аяғына дейін кезең кезеңмен жүзеге асырылады, бұл ретте онда берілген сценарийдің барлық командалары орындалады және осы бетке берілген параметрлер талданады. Сценарийлердің өңдеушісі (интерпретатор) asp. Dll динамикалық қосылатын кітапхана болып табылады және ISAPI-ның серверлік кеңейтілімі болып табылады. Сценарийдің енгізілген коды Windows Script Host (WSH) – қа беріледі. WSH — JScript және VBScript скриптілік тілдерінде, сондай-ақ басқа қосымша орнатылатын тілдерде сценарийлерді іске қосу үшін арналған Microsoft Windows компоненті. WSH оған берілген кодты орындайды және asp. dll файлына жауапты қайтарады. ASP Windows Script Components сценарийлерімен жұмыс технологиясын сүйемелдейтіндіктен, ол COM-компоненттеріне бизнес-логикаларды орындайтын барлық сценариялық процедураларды орналастыру мүмкіндігін береді. Осы компоненттер қайта пайдалану мүмкіндігін береді және Web-қосымшаларда және COM технологиясы бойынша құрылған басқа бағдарламаларда жұмыс істеуі мүмкін. Өзінің бизнес-логикасын іске асыру қажеттігі пайда болса,




8.7 - сурет. ASP-бетке сұрау салуды өңдеу

жаңа COM объектілерін құру немесе сыртқы әзірлеушілердің COM объектілерін пайдалану мүмкіндігі беріледі. ASP бетіне кіріктірілген серверлік сценарий ActiveX Data Objects (ADO) интерфейсінің әдістерін шақыру арқылы дерекқорға сұрау сала алады.

ASP-да бағдарламалаудың нақты тіліне бағдарлану жоқ, осыған орай, сценарийлердің кез келген тілімен танысу ASP-пен жұмыс істеу үшін жеткілікті болады. Бет жұмысының логикасы пайдаланушылардан жасырылған. Пайдалану Web-сервер оған беттің өзін емес, оны интерпретациялау нәтижесін жіберетіндіктен, ASP бетінің мазмұнын көре алмайды.

ASP жеткіліксіздігіне мыналарды жатқызуға болады:

- бағдарламалаудың иілмелі, мықты және масштабталатын ортаның жоқтығы;
- сценарий командаларын HTML-да орналастыру қажеттігіне байланысты контекстті алмастырудан өнімділікті азайту.

ASP технологиясы — Microsoft . NET платформасына негізделген Web-қосымшаларды құру технологиясы ASP. NET түрінде дами бастады.

## 8.7. ASP.NET

**Негізгі тұжырымдама.** ASP технологиясы өз мүмкіндіктері бойынша шектелген. Ол толық функционалды бағдарламалау тілдеріне қарағанда мүмкіндіктері скриптілік тілдерді пайдаланды. ASP коды HTML –ға арнаулы тегтер түрінде кіріктірілген болатын.

Active Server Pages — ASP . NET технолгиясының жаңа нұсқасы Microsoft .NET Framework архитектурасында негізгі болып табылады. Microsoft ASP . NET-та клиенттердің әртүрлі түрлеріне қолдау көрсетуді, басқа сөзбен айтқанда клиенттің түріне қарай әртүрлі кодтарды генерациялай алатын «қисынды» серверлік компоненттерді құруды қамтамасыз ететін тәсіл іске асырылған.

*ASP. NET* —IIS. ASP басшылығымен жұмыс істейтін Web-қосымшаларды және Web- сервистерді құруға арналған платформа. NET Web-қосымшаларды құрудың басқа технологияларынан серверлік өнімдермен жоғары интеграциялану деңгейімен ерекшеленеді. Осы платформада деректерге қолжетімділікті әзірлеу және қауіпсіздікті қамтамасыз етуге арналған Microsoft құралдары болады. Сонымен қатар, ASP. NET өте ұқсас әдістемелерді, бірдей бағдарламалау тілдерін, деректерге қол жеткізу технологияларын және басқаларды қолдана отырып, Windows-қосымшаларды және Web-қосымшаларды әзірлеу мүмкіндігін береді.

Күрделі Web-қосымшаларды құру үшін ASP. NET жетістіктерінің үлкен тізбесі бар деуге болады. Алайда, кемшілік ретінде ASP. NET-ның IIS болуын қажет ететіндіктен, тек Windows серверлерінде ғана жұмыс істейтінін атап өтуге болады. IIS қажет етпейтін, мысалы Web-сервер Apache –ты пайдаланатын және басқа операциялық жүйелердің басшылығымен жұмыс істейтін Web-қосымшаларды құру үшін басқа технологияларды қолдану қажет.

ASP. NET архитектурасындағы маңызды мәселе оның . NET Framework инфрақұрылымының бір бөлігі болып табылу фактісінде. Әзірлеушілер .NET Framework: C#, Visual Basic.NET, JScript.NET-ке және басқаларға кіретін бағдарламалау тілдерін пайдалана отырып, ASP. NET үшін кодты құра алады. Көмегімен Web-қосымшаларды әзірлеуге болатын негізгі бағдарламалау тілдері толықтай объектіге бағдарланған болып табылады, бұл модульдерді әзірлеуді, түрлендіруді, қызмет көрсетуді, жөнге салуды және қайта пайдлануды басқа технологиялармен салыстырғанда анағұрлым қарапайым етеді. Сондай-ақ HTML басқару элементтерінің кодын автоматтандырылған түрде генерациялайтын серверлік объектілерді қолдану есебінен қолмен жазылған кодтың көлемін түбегейлі қырқартады. Visual Studio зерттемесінің стандартты ортасын пайдалануға болады. Ақпарат жинауға, ойтайланыруға және басқаларға қосымша уақытты қажет етпей, бірінші рет сұрау салғанда код компиляцияланып, арнаулы кэшке орналастырылатындықтан және нәтижесінде тек орындалатындықтан, NET, яғни ASP. NET -ның сценариялық технологиялармен салыстарғанда жылдамдық басымдылығы бар.

**ASP. NET басымдықтары.** ASP. NET басымдықтары .NET платформасына жатады.

1. *Бірыңғай бағдарламалау үлгісі.* Операциялық жүйелердің бір функциясы динамикалық қосылатын кітапханалар (DLL) процедурасы, ал басқалары COM-объектілері арқылы қолжетімді болатын қолданыстағы тәсілмен салыстырғанда барлық қолданбалы сервис жалпы объектіге бағдарланған үлгі түрінде берілген.

2. *Нұсқалармен проблемалардың болмауы.* Дәстүрлі бағдарламалау кезінде Windows нұсқалардың үйлесімдігіне қатысты проблема туындайды. Ол жаңа қосымша үшін орнатылатын компоненттер бұрынғы қосымшаның компоненттерін алмастырған жағдайда пайда болады. . NET Framework архитектурасы қолданбалы компоненттерді оқшаулау мүмкіндігін береді, осыған орай, қосымша әрқашан олармен бірге әзірленген және тестіленген компоненттерді жүктейді.

3. *Әзірлеуді, ашуды және жоюды ықшамдау.* . NET Framework компоненттері (оларды жай типтер деп атайды) тізіліммен байланыспаған. NET Framework қосымшаларын орнату файлдарды қажетті каталогтерге көшірмелеумен, ал оларды жою — файлдардың жойылуымен аяқталады.

1. *Кроссплатформалану.* .NET Framework үшін код компиляцияланғанда компилятор кодты CIR (Common Intermediate Language) тілінде генерациялайды, бұған процессорлық командалардан тұратын дәстүрлі код жатпайды. CLR орындау кезінде CIL-ды процессордың командаларына трансляциялайды. Трансляция орындау кезеңінде жүзеге асырылатындықтан, нақты процессордың командалары генерацияланады. Бұл CLR және FCL нұсқалары жұмыс істейтін кез келген машинада . NET Framework үшін қосымшаны іске қосуға болады дегенді білдіреді.

2. *Бағдарламалау тілдерін интеграциялау.* COM әртүрлі тілдерге өзара іс-қимыл жасау мүмкіндігін береді. . NET Framework әртүрлі тілдерге интеграциялану, яғни бір тілге басқа тілдерде құрылған типтерді пайдалану мүмкіндігін береді. Мысалы, CLR Visual Basic-те іске асырылған класстан туындаған C++ классында құру мүмкіндігін береді. CLR-ға бағдарланған барлық тілдер пайдалануы тиісті типтердің жалпы жүйесін (Common Type System — CTS) айқындайтындықтан, CLR тілдерді интеграциялау мүмкіндігін береді. Жалпытілдік ерекшелік (Common Language Specification — CLS) тілдері басқа тілдермен интеграциялануы үшін компиляторларды әзірлеушілері ұстануы тиісті қағидаларды айқындайды.

3. *Жадыны автоматтандырылған түрде басқару (қоқысты жинау).* Жүйенің өнімділігімен байланысты ең жиі таралған проблемалардың бірі —ресурстарды босатуға немқұрайлы қарау, бұл күтпеген уақытта бағдарламаны қате орындауға әкелуі мүмкін. CLR ақпараттық жайылып кетпеуіне кепілдік бере отырып, автоматтандырылған түрде ресурстардың пайдаланылуын тексереді.

ASP.NET-те Web-қосымшаларды әзірлеу негізіне динамикалық құжаттарды құру кезінде *Microsoft* ұсынатын ұсыну кодын және іске асыру кодын бөлу үлгісі алынады. Бұл сценарийлерге арналған жеке файлда немесе арнаулы тег ішінде бағдарламалық кодты орналастыру жолымен жүзеге асырылады. Осындай тәсіл Web-дизайнерге құжатты белгілеу кодымен жұмыс жасау кезінде жинақталу мүмкіндігін береді.

Өнімділікті арттыру. Web- қосымшалардың өнімділігі — күрделі мәселе, өйткені Web-қосымшалар көптеген пайдаланушылардың сұрау салуларын бір уақытта өңдейді, бұл ретте пайдаланушылар саны белгілі болмайды. Бұл Web-қосымшаны Интернет ғаламдық торында орналастырғанда оған мыңдаған және он мыңдаған пайдаланушылар бір уақытта сұрау салу мүмкіндіні туындайтынын білдіреді. Дәл, сондықтан, Web-қосымшаларды әзірлеу кезінде қосымшалардың өнімділігі және масштабталуы мәселесіне аса назар аудару қажет.

Web-қосымша жұмыс істеген кезде өнімділікке екі құрамдас бөлік түбегейлі әсер етеді:

1) Web-сервердің бағдарламалық коды;
2) қолданбалы қосымшаның бағдарламалық коды.

ASP. NET Windows Server операциялық жүйелері құрамында жеткізілетін Internet Information Services Web-серверінің шеңберінде жұмыс істейді және қосымшалардың өнімділігін арттыру бойынша бірнеше механизмдерден тұрады.

Жеке алғанда, HTTP-сұрау салуларын тікелей өңдейтін IIS-дағы жұмыс ағындары әрбір сұрау салу түскен кезде құрылмайды, Web-сервер жұмыс істей бастаған кезде инициалданылады және күту режиміне көшеді. Осы жағдайда Web-серверге кезекті сұрау салу түскен кезде жұмыс ағыны құрылмайды және сервер жұмыс істей бастаған кезде орналастырылған Web-сервер ағындарының пулынан алып тасталады. Осындай тәсілді қолдану сұрау салуды өңдеу кезінде ағындарды құруға жұмсалатын уақыт пен ресурстарды үнемдеу және қосымшалардың жалпы өнімділігін арттуры мүмкіндігін береді.

Серверге сұрау салған кезде ағынның пулында сұрау салуларды өңдеу үшін қолжетімді ағындар болған жағдайда Web-сервер пайдаланушыға қателік туралы хабарлайды және Web- қосымшаға кейінірек сұрау салуды ұсынады. Пулда қолжетімді ағымдар саны қосымшаға түсетін жүктемеге сәйкес баптауға болады. Осы үшін IIS серверін басқару консолі пайдаланылады.

Алайда, өнімділікті арттыру бойынша Web-серверінің механизмдеріне қарамастан, Web-қосымшаның бағдарламалық коды тар жер болуы мүмкін.

Қосымшаның өнімділігін төмендететін Web-қосымшалардың мынадай проблемалары анағұрлым көп таралған:

а) сұрау салуды өңдеу процесінде дерекқорға жиі және күрделі сұрау салулармен жүгіну;

б) орталық процессорды түбегейлі жүктейтін күрделі бағдарламалық кодты (мысалы, есептеу операциялары) орындау.

Осы проблеманың күрделілігі кейде қосымшаның логикасының дерекқорға сұрау салуға және күрделі бағдарламалық кодты орындауға тура келетіндігі болып табылады, бұл ретте осы кезеңдерді орындаумауға болмайды. Дерекқорға сұрау салуларды бір жағынан ғана ойтайландыруға болады, бұл ретте олар бәрібір де қосымшаның өнімділігін қатты төмендетеді.

Осы міндетті шешу үшін сервер тарапынан деректерді кэштеуді пайдалануға болады. Осы тәсілді қолдану идеясы бетке бірінші рет сұрау салған кезде күрделі бағдарламалық кодты өңдеу және

нәтижені жадының арнаулы учаскесінде – кэште сақтау болып табылады. Одан кейінгі сұрау салуларды өңдеу кезінде сервер күрделі және ұзақ мерзімді операцияларды орындамайды, оның орнына деректерді кэштен алып, клиентке береді. Көбінесе, деректер соншалықты өте тез өзгермейді, осыған орай, пайдаланушы үшін осы процесс айқын болады, ал бұл ретте қосымшаның өнімділігі түбегейлі артады.

ASP. NET – та кэштелудің бір түрі шығыс ағынын, яғни сұрау салу өңделгеннен кейін клиентке берілетін деректерді кэштеу болып табылады. Осылайша, шығыс ағынын кэштеу клиентке берілетін барлық ақпарат (яғни, HTML) кэште сақталатын режимді таңдау мүмкіндігін береді. Осы бетке кейінгіде де сұрау салған кезде осы беттің HTML- коды кэштен алып тасталады, бұл ретте беттің коды іске қосылмайды. Кэштеуді айқындаудың осындай тәсілі іске асыру тұрғысынан қарасақ, анағұрлым қарапайым болып табылады.

Деректерді кэштеу — ASP. NET кэштеу механизмі ұсынатын балама кэштеу тәсілі. Шығыс ағынын кештеумен салыстырғанда деректерді кэштеуді пайдалану кезінде беттің бағдарламалық коды әрбір бетке сұрау салу кезінде өңделеді. Бұл ретте Web-нысанын генерациялау нәтижесі емес, пайдаланушы кодында айқындалатын деректер кэштеледі. Беттің құрылымы әрдайым өзгеріп отыратын болса, бірақ бұл ретте құрамына орындау кезінде көп ресурстар жұмсалатын бағдарламалық код кіретін болса, осындай тәсілді қолдану анағұрлым оңтайлы болып табылады.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. Web-сервер деген не?
2. Серверлік Web-қосымшаларды пайдалану қандай басымдықтар береді?
3. Web-сервер жұмысының механизмін түсіндіріңіз.
4. Статикалық сайттың динамикалық сайттан ерекшелігі неден тұрады?
5. Виртуалды хостинг дегеніміз не?
6. CGI интерфейсі арқылы клиенттік-серверлік өзара іс-қимыл қалай жүзеге асырылады?
7. CGI интерфейсінің басымдықтары мен кемшіліктерін түсіндіріңіз.
8. CGI жұмысының жүйелілігін сипаттаңыз.
9. PHP-да бағдарлама қай операторлардан басталады?
10. PHP-сценарийі HTML- құжатының мәтініне қалай қосылады?

11. PHP-да массивтер қалай жасалады?
12. PHP-дағы пайдаланушылық функцияларының қандай ерекшеліктері бар?
13. Статикалық айнымалы шаманы пайдаланудың ерекшелігі неде?
14. Пайдаланушылық сұрау салуды өңдеу кезінде PHP қалыптастыратын, алдын ала анықталған ауыспалы шамалардың негізгі массивтерін атап өтіңіз.
15. Серверге HTML-нысандарының деректерін қандай әдістермен беруге болады?
16. PHP-сценарийдің дерекқормен өзара іс-қимылы кезіндегі іс-әрекеттердің реттілігін сипаттаңыз.
17. Қандай тәсілмен және қандай функциялардың көмегімен дерекқорға қосылу жүзеге асырылады?
18. Қандай тәсілмен және қандай функциялардың көмегімен сұрау салуларды дерекқорға беру жүзеге асырылады?
19. Қалай PHP-ны XML- құжаттарымен жұмыс істеу үшін пайдалануға болады?
20. PHP-дағы сеанс/сессия дегеніміз не?
21. Не үшін супер жаһанды $_session массиві пайдаланылады?
22. ISAPI дегеніміз не?
23. ISAPI қосымшаларын бағдарламалау CGI бағдарламаларын құруға қарағанда, неге күрделілеу міндет болып табылады?
24. ISAPI фильтрлері, сондай-ақ ISAPI кеңейтілімі немен ерекшеленеді?
25. Сервер ISAPI-фильтрлерін қалай шақырады?
26. ASP технологиясының мәні неде?
27. ASP қандай бағдарламалау тіліне негізделеді?
28. ASP кемшіліктеріне нені жатқызуға болады?
29. ASP.NET-тің басқа Web- қосымшалардың технологияларын құрудан ерекшеленуі?
30. ASP.NET –тің басымдықтарын атаңыз.
31. Web-қосымшалар жұмыс істеген кезде өнімділікке қандай құрауыштар әсер етеді?

# «1С:КӘСІПОРЫН» ПЛАТФОРМАСЫНДАҒЫ ҚОЛДАНБАЛЫ БАҒДАРЛАМАЛАУ

# III

## БӨЛІМ

# 9 тарау

# «1С» НЕГІЗГІ ҰҒЫМДАРЫ МЕН МЕХАНИЗМДЕРІ

## 9.1. «1С»ЖҮЙЕСІНІҢ ТҰЖЫРЫМДАМАСЫ

«1С:Кәсіпорын» жүйесі (қысқаша - «1С») кәсіпорындар мен ұйымдардың әртүрлі қызметін автоматтандыруға арналған әмбебап жүйе болып табылады. Бастапқыда «1С» жүйесі бухгалтерлік есептерді және басқару есебін (тауарлық және материалдық қалдықтарды есептеу, контрагенттермен өзара есеп айырысу, жалақыны есептеу, негізгі құралдардың және басқаларды есептеу) автоматтандыруға арналған, бірақ қазіргі таңда осы өнім бухгалтерлік міндеттерден алшақ облыстарда да қолданылады.

«1С» жүйесінің негізгі ерекшелігі - жиынтығы пайдалануға дайын өнім болып табылатын платформаға және қолданбалы шешімге (конфигурацияға) бөлу. «1С:Кәсіпорын» қолданбалы шешімдерінің конфигурациялары ашық болып табылады, сондықтан әзірлеуші өзінің күшімен кез келген қолданбалы шешімді «өзіне сәйкес» түрлендіре және жөнге сала алады.

*«1С» платформасы* — конфигурацияны басқару үшін аспаптық және тіл құралдарының жинағы. Қолданбалы шешімді құру, түрлендіру және оның жұмыс істеуі платформаның технологиялары мен механизмдерін қолданусыз мүмкін емес. Қолданбалы шешімдерді түрлендіру үшін қандай да бір жеке бағдарламалық өнімдерді пайдалану қажеттігі жоқ — барлық әзірлеу құралдары технологиялық платформаның құрамына кіреді.

*«1С» конфигурациясы (қолданбалы шешім)* — белгілі бір қолданбалы міндеттерді шешуге арналған ақпаратты өңдеу объектілерінің, қасиеттерінің, әдістерінің және алгоритмдерінің нақты жинағы.

«1С» типтік конфигурациялары — «1С» фирмасы әзірлеген, жалпы тұтынушылардың пайдалануы үшін жарамды және кәсіпорындардың белгілі бір түріне және міндеттер классына бағдарланған әмбебап конфигурациялар. Типтік конфигурацияларды пайдаланушы кейбір талаптарды сақтаған кезде өзгерте, толықтыра алады. Анағұрлым таралған типтік конфигурациялардың мысалы ретінде мыналарды атауға болады:

- «1С:Бухгалтерия 8»;
- «1&ERP Кәсіпорынды басқару 2.0»;
- «1С:Жалақы және персоналды басқару 8»;
- «1С:Сауданы басқару 8» және басқалары.

Сондай-ақ «нөлден» құруға болады.

«1С:Кәсіпорын» жүйесі мыналарды қамтамасыз етуі тиіс:

- қолданбалы шешімдердің тапсырыс берушінің талаптарына бейімделуі;
- қолданбалы шешімді құруға қатыспаған әзірлеушінің дайын қолданбалық шешімді өзгерту мүмкіндігі (өте көп жағдайда жаңа қолданбалы шешімдер құрылмайды, тек қолданыстағылар пысықталады);
- компьютерлік технологиялар мен платформаларды тиімді пайдалану, бұл ретте әзірлеушіден арнаулы терең білім қажет етілмейді;
- зерттемені стандарттау.

«1С:Кәспорын» жүйесінде екі жұмыс режимі бар:

1) «1С:Кәсіпорын» (орындау ортасы);
2) конфигуратор (әзірлеу ортасы).

«1С:Режимі» жүйені пайдаланушылардың жұмысы — деректерді енгізу, ақпаратты өңдеу, есептерді қалыптастыру үшін арналған.

Конфигуратор режимі әзірлеу үшін қолданылады және қолданыстағыларды редакциялау және жаңа конфигурацияларды құру үшін арналған аспаптар мен механизмдер болып табылады. Әзірлеу кезеңінде өңделетін ақпарат құрылымын қалыптастыру, шығыс деректерді енгізуге арналған нысандарды құру, әртүрлі деректер тізімін қарау жүзеге асырылады; енгізілген және қорытынды ақпаратты сақтау ұйымдастырылады; әртүрлі пайдаланушылар тобы үшін командалық интерфейс қалыптастырылады.

«1С:Кәсіпорын» жүйесінің мүмкіндіктері зор, алайда, оның архитектурасы және платформа механизмдері мен технологияларын нақты іске асыру ең алдымен бизнес-қосымшаларды құру бойынша мамандандырылған міндеттерді шешу қажеттігіне байланысты болады.

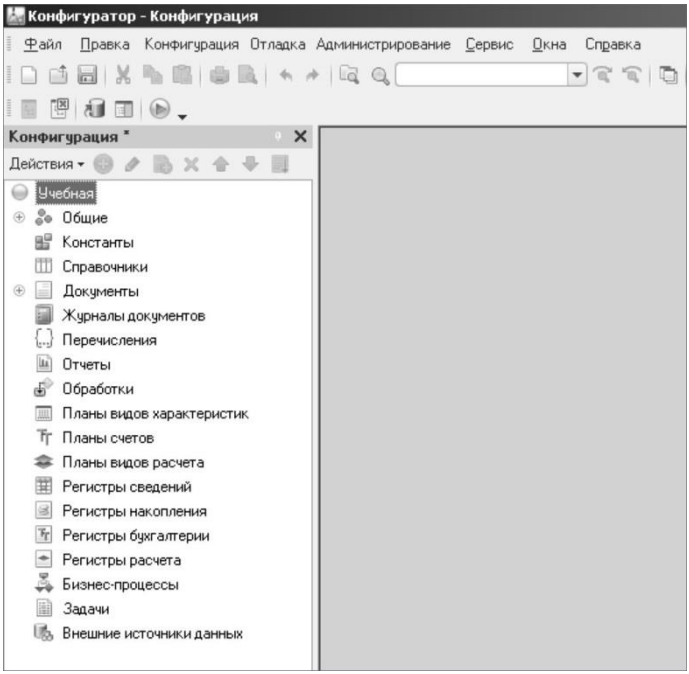
## 9.2. КОНФИГУРАЦИЯЛАУ ОБЪЕКТІЛЕРІ

Қолданбалы міндетті әзірлеу кезінде «1С:Кәсіпорын» әзірлеушісі дерекқорға тікелей сұрау салмайды, ол төмендеңгейлі технологиялардан оқшауланған болады. Әзірлеуші «1С:Кәсіпорын» платформасымен тікелей жұмыс істейді, атап айтқанда:

- Конфигураторда деректердің құрылымын сипаттайды;
- Кіріктірілген тіл объектілерінің көмегімен деректерді басқарады;
- Сұрау салу тілін пайдалана отырып, деректерге сұрау салуларды құрады.

Бұл ретте абстрактілік технология — метадеректер технологиясы қолданылады. *Метадеректер* барлық қолданбалы шешімдерді толықтай сипаттайтын объектілердің иерархиялық құрылымы болып табылады. Осы құрылым «конфигурациялау объектілерінің ағашы» деп аталады (9.1 - сурет).

*«1С:Кәсіпорын» жүйесіндегі конфигурациялау объектісі* дегеніміз сипаттамалары және міндеттері бірдей ұғымдар тобының формалды сипаттамасы (қолдану облысы, пайдаланушының жүйемен өзара іс-қимыл жасау құралдары)




9.1 - сурет. Конфигурациялау объектілерінің ағашы

болып табылады. Конфигурациялау объектісі конфигурациялау объектісінің көмегімен «1С:Кәсіпорын» жүйесінде іске асырылған қолдану облысының нақты ұғымының компьютерлік аналогы болып табылады деуге болады. Мысалы, конфигурациялау объектісі - «1С:Кәсіпорын» жүйесіндегі анықтамалық деректердің бірдей элементтер тізімін – карточкалардың нормативтік анықтамалықтарды, тізімдерді және басқаларды енгізу үшін арналған. Осы типтік конфигурациялау объектісін пайдалану кәсіпорын қызметін автоматтандыру үшін қажетті кез келген анықтамалықтарды жүргізуді ұйымдастыру мүмкіндігін береді. Әдетте, Анықтамалық түріндегі конфигурациялау объектілері кәсіпорында іс жүзінде жұмыс істейтін анықтамалықтар түрлерін, мысалы, қызметкерлер анықтамалығының немесе номенклатураның компьютерлік аналогы болып табылады, алайда, нақты физикалық аналогы жоқ тізімдерді қолдану үшін пайдалануға болады. Әдетте, Құжат түріндегі объектілер қарапайым қағаз түріндегі құжаттардың электрондық аналогтары болып табылады.

Конфигурациялау объектісі нақты мағынаны емес, тек оны түрін ғана сипаттайтынын есте сақтау қажет. Мысалы, Қызметкерлер анықтамалығы нақты адам сипаттамасынан емес, деректемелер тізбесінен (қызметкер туралы сипаттамалар түрлерінің жинағы— ТАӘ, туған күні, жынысы және басқалары), олардың мағыналарын енгізу нысанынан, тізімдерді қарау нысанынан және ақпаратты басып шығару үшін арналған макеттерден тұрады. Өзге сөзбен айтқанда, бұл бағдарламаның көмегімен конфигурацияда қолдану саласының барлық біртектес объектілері есепке алынатын сипаттау схемасы (мысалы, Қызметкерлер анықтамалығы үшін бір сипаттамасы Петров, Иванов үшін де, кез келген басқа қызметкер үшін де пайдаланылады) құрылады.

Метадеректер технологиясы қолданбалы шешімнің көзбен шолу құрылымдауын пайдаланады. Оның негізгі құрылымы конфигурациялау объектілерінің құрылымымен сипатталады. Көзбен шолу құралдарының көмегімен қажетті типтердің, құрылымдық деректердің, құқықтар жинағының сипаттамасын, объектілер арасындағы байланыстарды, мінез-құлық ерекшеліктері жөніндегі ақпаратты, көзбен шолу көрінісін және басқаларды алатын жаңа объекті қосылады. Әзірлеуші бағдарламаны орындаудың әртүрлі уақыттарында қандай да бір конфигурациялау объектілерін өткізудің арнайы алгоритмдерін сипаттау үшін кіріктірілген тілді және сұрау салу тілдерін пайдаланады.

Барлық конфигурациялау объектілерінде дерлік кіріктірілген тілде алгоритмдер сипатталатын модульдерден тұрады. Осы модульдер қолданбалы шешім жұмысының алдын ала белгіленген жерлеріне – нақты оқиғаларға нақты орындау ортасымен шақырылатын болады.

«1С:Кәсіпорын» жүйесінің маңызды ерекшелігі әзірлеуші қолданбалы шешімнің құрылымын сипаттау үшін қисынсыз емес, нақты белгіленген конфигурациялау объектілерін пайдалану болып табылады. «1С:Кәсіпорын» платформасында конфигурациялау объектілерінің шектелген түпнұсқалар жинағы (анықтамалық, құжат, жинақтау тізілімі, бизнес-процес және басқалары) берілген. Осындай әрбір түпнұсқа конфигурациялау объектісінің белгілі бір базалық іске асырылымынан тұрады. Әзірлеуші конфигурациялау объектілерінің ағашына жаңа конфигурациялау объектісін қосады, осы объект түпнұсқаның базалық іске асырылымын мұра етеді. Мысалы, жаңа анықтамалықты құру кезінде оған «Код» және «Атауы» деректемелері автоматтандырылған түрде қосылады, анықтамалықтың тізімдері мен элементтерін редакциялау үшін экрандық нысандар генерацияланады.

Бұрын аталғандай, конфигурациялардың нақты құрылымы қолдану саласының үлгісі болып табылады. Конфигурацияны құру конфигуратордың көмегімен орындалады. Құрылған конфигурация қажетті есептік міндеттерді орындау үшін жарамды бағдарламалық ортаны іске асыру «1С:Кәсіпорын» жүйесімен пайдаланылады.

Әрбір конфигурациялау әдісінің бірегей қасиеттер (атауы, синонимі, тақырып деректемелері, өткізу мүмкіндігі және басқалары) жинағы болады. Бұл жинақ жүйенің деңгейінде сипатталған және міндеттер конфигурациясын баптау процесінде өзгертуге жатпайды. Конфигурациялау объектісінің қасиеттер жинағы негізінде «1С:Кәсіпорын» жүйесіндегі мақсатымен айқындалады.

Кез келген конфигурациялау объектісінің негізгі қасиеті атау – конфигурациялау объектісінің қысқаша атауы болып табылады. Жаңа конфигурациялау объектісін құру кезінде оған автоматтандырылған түрде шартты атау беріледі. Конфигурациялау объектісіне тән қасиеттер жинағының ішінен алынған кейбір қасиеттер редакциялау үшін қолжетімді және оларды қалай балғанда да жүйені конфигурациялау процесінде өзгертуге болады. Өзгерістер сипаты және олардың шектері жүйенің деңгейінде белгіленеді. Жүйені конфигурациялауды жүзеге асыратын маман конфигурациялау объектісінің қасиетін мақсатқа сай өзгертумен жүйенің жұмысы кезінде объектінің қажетті іс-әрекетіне қол жеткізе алады. Алайда, осындай өзгерістердің объектінің болмысына қатысы жоқ және осы түрге жататын объектілерге тән емес іс-қимылға қол жеткізу мүмкіндігін бермейді.

Конфигурациялау объектісінің түріне қарай әртүрлі бағыныстағы объектілер тобы болуы мүмкін, мысалы, деректемелер, өлшемдер, нысандар, кестелік бөліктер. Бағыныстағы объектілердің құрамы объектінің түріне байланысты болады.

*Деректемелер* — осы объектінің шегінде ғана қолжетімді объекті туралы қосымша ақпарат.

*Кестелік бөліктер* — кесте түрінде берілген объекті туралы қосымша ақпарат жинағы.

*Кестелік бөліктер деректемелері* — объектінің кестелік бөлігі шегінде ғана қолжетімді объектінің кестелік бөлігінің құрамы.

Нысандар конфигурациялау объектісінде сақталатын ақпаратты енгізу, қарау және редакциялау үшін пайдаланылады және нысанның модулінен — «1С:Кәсіпорын» жүйесінің кіріктірілген тіліне бағдарламадан тұрады. Көзбен шолу көрінісін көру мүмкіндігі конфигурациялау объектісіне пайдаланушымен интерактивтік өзара іс-қимылды ұйымдастыру мүмкіндігін береді.

## 9.3. РӨЛДЕР ЖӘНЕ ҚОСЫМША ЖҮЙЕЛЕР

«1С:Кәсіпорын» жүйесіндегі *рөлдер* пайдаланушының жүйеде өңделетін ақпаратпен жұмыс істеуге құзыретін айқындайды. Пайдаланушыға берілетін құзыреттер жиынтығы әдетте оның міндеттері шеңберімен айқындалады. Пайдаланушыға рөлдерді тағайындау операциясы екі негізгі міндетті шешеді:

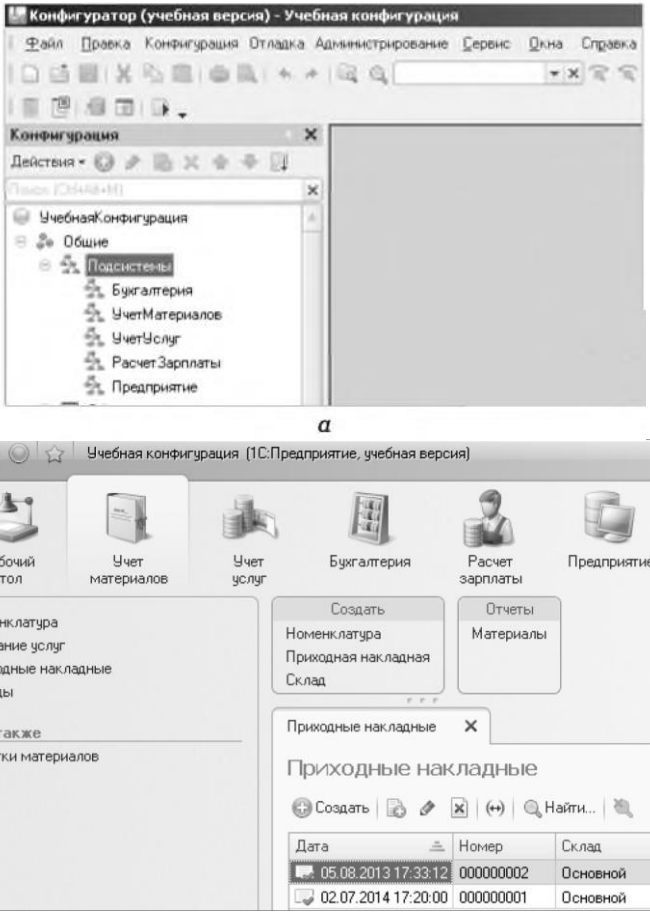
1) кез келген есеп жүйесінде сөзсіз орын алатын құпия ақпаратты пайдаланушылар шеңбері шектеледі;

2) қандай да бір операцияларды (бірінші кезекте, деректерді жою және түзету операциялары) орындауға тыйым салу белгілі бір деңгейде ақпарат жоғалту қаупінің алдын алу мүмкіндігін береді.

Конфигурациялаудың барлық қалған бөліктері бір –бірімен тығыз байланысқан және келісілген түрде өзгерістер енгізуді талап етеді.

«1С:Кәсіпорын» платформасы қолданбалы шешімдерге логикалық тұрғыда құрылатын қолданбалы шешімдерге бөлінетін жеке функционалдық бөліктерді – қосымша жүйелерді ерекшелеу мүмкіндігін береді. Қосымша жүйелердің көмегімен пайдаланушыға ол жұмыс істейтін объектілер ғана (яғни, қолданбалы шешімнің функционалдығы) көрінетін ыңғайлы және функционалды интерфейсті ұсынуға болады.

Қосымша жүйелердің құрылымы иерархиялық болуы мүмкін, яғни бір қосымша жүйе бірнеше басқа қосымша жүйелерден құралуы мүмкін. Қосымша жүйелер құрылымы барлық қолданбалы шешімнің функционалдығын айқындайды және қосымшаның ғаламдық командалық интерфейсін құру негізі бола алады. Жүйелердің құрылымы (9.2 - сурет, *а)* қосымша интерфейсіндег бөлімдерге (9.2 - сурет, *б)* сәйкес келетіндіктен, іс жүзінде кез келген қолданбалы шешімді құру




9.2 - сурет. Конфигурациялаудың қосымша жүйелерінің құрылымы (а) және қолданбалы шешімінің бөлімдері (б)

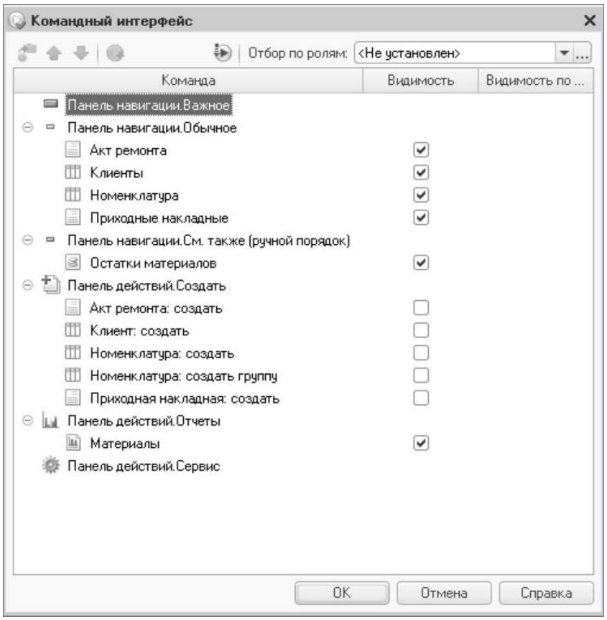
қосымша жүйелер құрамын құрудан басталады. Бұл ретте қосымша жүйелердің құрамын жете пысықтау және нәтижесінде құрылатын объектілерді қосымша жүйелерге байлау қажет. Осы қолданбалы шешімді сипаттайтын конфигурациялаудың әрбір объектісі бір немесе бернеші қосымша жүйелерге жатқызыла алады.

Пайдаланушыға қолданбалы шешімнің құрылымы (қосымша жүйелер иерархиясы) көрсетіледі және қолданбалы объектілердің функционалдығына

(анықтамалықтар тізімдерін, құжаттарын шақыру, есептерді, пысықтауларды ашу және басқалары) қолжетімділікті беру үшін стандартты командалар ұсынылады. Командалық интерфейс пайдаланушыны конфигурациялау функционалдығы бойынша бағыттаудың негізгі құралы болып табылады. Конфигурацияны әзірлеуші тиісті қосымша жүйелерге қолданбалы объектілерден тұрады.

Командалық интерфейсті қалыптастыру пайдаланушылардың құқықтары, рөлдер бойынша командаларды көзбен көру мүмкіндігі, қосымшаның функционалдық опциялары және пайдаланушының баптаулары есебімен платформаның көмегімен жүзеге асырылады. Мысалы, пайдаланушыға қандай да бір анықтамалықты қарауға тыйым салынған жағдайда осы анықтамалықтың тізім нысанын ашу командасы командалық интерфейстен автоматтандырылған түрде алып тасталады. Сондай-ақ нысандар автоматтандырылған түрде нысандарды көрсету кезінде құқықтардың бар-жоғын есепке алады.

Әзірлеуші, әрине, жүйе ұсынатын командалық интерфейстің құрылымын (командалар тәртібін, көру мүмкіндігін өзгерту) редакциялай алады. Бұл үшін нақты қосымша жүйе және барлық қосымша жүйелер үшін шақырылатын командалық интерфейстің редакторы қолданылады (9.3 - сурет).



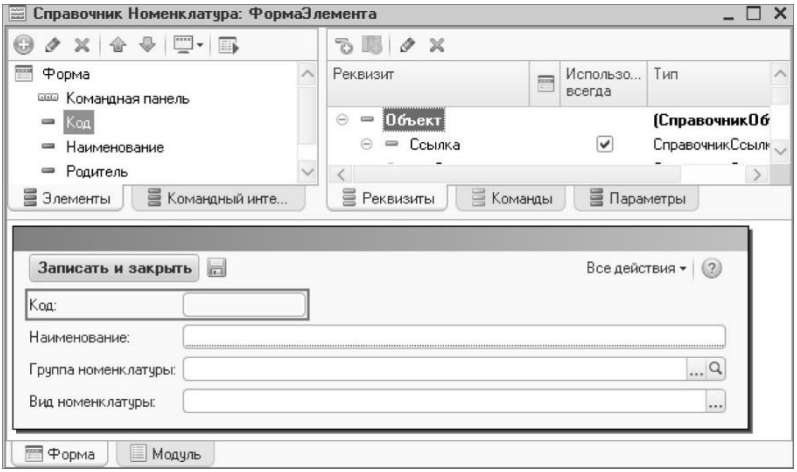

9.3 - сурет. Командалық интерфейсті баптау

## 9.4 НЫСАНДАРЫ

«1С:Кәсіпорын» нысаны дерекқордағы ақпаратты көрсету және редакциялау үшін арналған. Нысандар нақты конфигурациялау объектілеріне жатуы немесе олардан бөлек орналасуы және жалпы алғанда барлық қолданбалы шешімді пайдалана алады. Мысалы, *Номенклатура* анықтамалығының белгілі бір мақсатта – анықтамалық элементін редакциялау, тізімді көрсету және басқалары үшін пайдаланылатын бірнеше нысандардан тұруы мүмкін. Осыған қоса, конфигурациялаудың нақты объектілеріне жатпайтын жалпы нысандар болуы мүмкін.

«1С:Кәсіпорын» жүйесіндегі көптеген конфигурациялау объектілерінде көзбен шолу нысаны болуы мүмкін. Конфигураторда нысандарды әзірлеу үшін өзара байланыс арқылы нысанның барлық компоненттерін редакциялау мүмкіндігін беретін нысандар редакторы қолданылады. Жалпы жағдайда конфигурациялау объектісі ретіндегі нысан мынадай бөліктерден тұрады (9.4 - бет):

- ақпаратты енгізу және редакциялау үшін пайдаланылатын экрандық диалог;
- нысанның модулі — «1С:Кәсіпорын» жүйесінің кіріктірілген тіліндегі бағдарлама әдетте кіріс бақылауын жүргізу, есептерді жасау және басқалары үшін диалогқа енгізілетін ақпаратты;




9.4 - сурет. Конфигуратор режиміндегі анықтама элементінің нысаны

- деректемелер тізімі;
- нысанда пайдаланылатын командалар өңдеуді жүзеге асырады.

Нысанның осы кез келген құрама бөліктері болмауы, яғни онда ақпарат болмауы мүмкін.

Нысанның көмегімен қолданбалы объектінің пайдаланушымен өзара интерактивті іс-қимылын іске асыруға болады.

Әрбір конфигурациялау объектісі кейбір стандартты іс-қимылдарды орындау үшін пайдаланылуы мүмкін. Мысалы, кез келген анықтамалық үшін оның элементтерінің тізімін көрсету, анықтамалықтың жеке элементтерін көрсету, анықтамалықтың тобын көрсету, анықтамалықтан элементтерді және элементтер тобын көөрсету қажеттігі туындауы мүмкін. Кез келген құжат үшін осындай іс-қимылдар тізімі анағұрлым азырақ болады: құжаттар тізімін қарау, құжаттар тізімінен таңдау және жеке құжатты қарау. Осындай стандартты іс-қимылдың қолданбалы шешім объектілерінің деректерімен орындалуын қамтамасыз ету үшін олардың әрбіріне арналған тиісті іс-қимылдарды орындау кезінде пайдаланылатын негізгі нысандар жинағы болады. Осы объектіге бағынысты нысандардың кез келгені негізгі болып тағайындалуы мүмкін. Мысалы, анықтамалықта мынадай негізгі нысандар болуы мүмкін: элемент нысаны, топтың нысаны, тізімнің нысаны, таңдау нысаны және топты таңдау нысаны. Ал құжатта негізгі нысандар құрамы басқаша болады: құжаттың нысаны, тізімнің нысаны, таңдау нысаны. Осылайша, пайдаланушыға анықтамалықтың тізімін немесе құжаттар тізімін қарау қажет болуы мүмкін, жүйе осы объектілер үшін тізім нысаны ретіндегі тиісті нысандарды ашады.

Нысанды деректермен байланыстыру үшін нысанның көмегімен көрсетілетін деректер тізбесі көрсетілетін *нысандардың деректермелері* пайдаланылады.

*Нысанның элементтері* енгізу жолдарынан, жалаушалардан, ажыратып-қосқыштар, түймелер және басқалары болуы мүмкін. Оған қоса, құрамына басқа элементтер кіретін топ болуы мүмкін. Элементтер құрылымы нысан көрінісіне сәйкес сипатталады.

Нысанның барлық функционалдығы деректемелер және командалар түрінде сипатталады. *Деректемелер* — нысан жұмыс істейтін деректер, ал *командалар* — орындалатын іс-қимылдар. Осылайша, әзірлеуші нысан редакторындағы нысанға қажетті деректемелер мен командаларды, олардың элементтерін көрсететін нысандарды құруы, қажеттігі туса, элементтерді бір топқа топтастыруы тиіс. Барлық нысандардың қандай деректерді көрсетуіне қарамаста, әрекеттері бірдей. Алайда, нысан деректемелерінің бірі ол үшін негізгі болып тағайындалуы мүмкін (ол Жуан қаріппен ерекшеленеді),

және бұл жағдайда нысанның қалыпты әрекеті және қасиеттері қандай түр нысанның негізгі деректемелеріне ие болуына қарай толықтырылатын болады (9.4 – суретті қараңыз).

Осы логикалық сипаттаманың негізінде жүйе автоматтандырылған түрде пайдаланушыға көрсету үшін нысанның сыртқы бейнесін қалыптастырады. Бұл ретте нысанның элементтерін пайдаланушы үшін мүмкіндігінше ыңғайлы орналастыру үшін жүйемен көрсетілетін деректердің әртүрлі қасиеттері (мысалы, түрі) есепке алынады.

Әзірлеуші нысандарда нысанның командаларын ғана емес, барлық конфигурацияның командалық интерфейсінде пайдаланылатын ғаламдық командаларды да пайдалана алады. Оған қоса, ағымдағы нысанның нақты деректерін есепке алумен басқа нысандарды ашатын параметрлік командаларды құру мүмкіндігі іске асырылған. Мысалы, бұл қазір шығыс жөнелтпе құжаты нысанында таңдалған қоймадағы қалдықтар бойынша есепті шақыру болуы мүмкін.

## 9.5. МОДУЛЬДЕР

Қолданбалы шешімнің модульдері кіріктірілген тілде бағдарламаның мәтінін орналастыруға арналған. Бұл модульдер конфигурацияның әртүрлі жерлерінде орналасады және олардың әртүрлі мағыналары бар.

*Модульдер* — конфигурациялау құрылымының тапсырылған нүктелерінде орналасқан және «1С» жүйесі жұмысының алдын ала белгілі кезеңдерін орындау үшін шақырылатын «1С» жүйесінің кіріктірілген тіліндегі бағдарламалар.

Жүйені конфигурациялауды жүзеге асыратын әзірлеуші қолданыстағы конфигураторда көзбен шолу заттары жеткіліксіз конфигурация объектілерінің өзара іс-қимыл алгоритмдерін жүзеге асыру үшін модульдерді пайдалана алады.

Конфигурацияда модульдердің бірнеше түрі болады. Бұл басқарылатын қосымша модулі, қалыпты қосымша модулі, сыртқы қосылу модулі, сеанс модулі, жалпы модульдер, нысандар модульдері және конфигурация объектілерінің (константа мағыналарының менеджерлері, анықтамалықтар, құжаттар, сипаттама түрлерінің жоспарлары, шоттар жоспарлары, есеп айырысу түрлерінің жоспарлары, алмасу жоспарлары, бизнес-процестер, міндеттер, есептер, еңбекпен өтеу) модульдері, конфигурация объектілері (анықтамалықтар, құжаттар, сипаттама түрлерінің жоспарлары, шоттар жоспарлары, есеп айырысу түрлерінің жоспарлары, алмасу жоспарлары, бизнес-процестер, міндеттер, есептер, еңбекпен өтеу, мәліметтер тіркелімдері, жинақ тіркелімдері, бухгалтерия тіркелімдері,

есеп айырысу, аудару, журналдар, құжаттар, баптау қоймаларының тіркелімдері) менеджерлерінің модульдері, жазбалар жиынтығының (мәліметтер тіркелімдері, қор тіркелімдері, бухгалтерия тіркелімдері, есеп айырысу тіркелімдері) модульдері, командалар модульдері.

Модульдердің көпшілігі конфигурацияның белгілі объектілеріне (мысалы, нысандарға) немесе қолданбалы шешімге (мысалы, басқарылатын қосымшаның модулі) бекітілген. Бұндай модульдер қолданбалы шешімнің белгілі бір жұмыс кезеңінде шақырылады (мысалы, басқарылатын қосымша режимінде «1С:Кәсіпорын» жүйесін іске қосқанда жіберіліп басқарылатын қосымша модулі немесе анықтамалық (элемент) объектісін құру кезінде жіберілетін анықтамалық объектісінің модулі). Осылайша, егер модульде біршама бағдарламалық код болса, онда ол код орындалып, қолданбалы шешімнің жұмысы жалғасатын болады. Бұндай модульдерде қолданбалы шешімнің әртүрлі объектілері үшін белгіленген оқиғаларды өңдеу прцедуралары орналасуы мүмкін. Бұл процедуралар тиісті оқиғалар болғанда орындалатын болады.

Қолданбалы шешімдер жұмысының процесінде туындаған модульдермен бірге қолданбалы шешім жұмысының процесінде өздігінен шақырылмайтын жалпы модульдер бар. Олар қолданбалы шешімнің басқа модульдерінен шақыралытан фукнциялар мен процедуралардың мәтіндерін орналастыру үшін қызмет етеді. Осылайша, жалпы модульдердегі код конфигурацияның басқа модулінен немесе командалық интерфейстен анық өтініш орындалғанда жүзеге асырылатын болады.

Әрбір модуль конфигурацияның қалған бөлігімен байланысты және бұл байланыс *модульді орындау мәнмәтіні* деп аталады. Мәнмәтін модуль коды орындалатын бағдарламалық ортаны айқындайды, - модульге қолжетімді объектілердің, айнымалы, процедуралар мен функциялар жиынтығы.

«1С:Кәсіпорын» бар мәнмәтіндердің мынадай түрлерін бөліп шығаруға болады:

1. Жаһандық мәнмәтін барлық бағдарламалық модульдер үшін қолжетімді. Ол жаһандық мәнмәтіннің ерекшелік мағыналарынан және әдістерінен (мысалы, `ЖұмысМерзімі` ерекшелігі), сондай-ақ жүйелік аударымдардан және мағыналардың жүйелік жинақтарынан пайда болады (мысалы, `ДиалогтыҚайтаруКоды` және `Таңбалар`).
2. Қосымша модулінің мәнмәтінінде (немесе сыртқы біріктіру модулінде) жалпы модульдердің экспортталатын процедуралары және функциялары қолжетімді.
3. Жалпы модульдің мәнмәтінінде экспортталатын процедуралар мен басқа жалпы модульдердің функциялары қолжетімді. Бұл мәнмәтінде

қосымша модулінің экспортталатын ауыспалы, процедурасы мен функциялары қолжетімсіз.

4. Қолданбалы объекті модулінің мәнмәтінінде объектінің оқиғаларына қолжетімдік бар. Мысалы, Шығыс Жөнелпе құжатының модулінде құжаттың деректемелері және оның кестелік бөлшектері қолжетімді, құжат әдістерін шақыруға және оқиғаларды өңдеуге болады.

5. Нысан модулінің мәнмәтінінде нысанның деректемелері, сондай-ақ оның қасиеті, әдістері мен оқиғалары қолжетімді болады. Егер нысанда негізгі деректеме белгіленсе, нысан модулінде негізгі деректеме ретінде пайдаланылатын, сондай-ақ осы қолбанбалы объектінің экспортталатын өзгермелі, процедуралар мен функциялар қолданбалы объектінің ерешелігі мен әдістері қолжетімді болады.

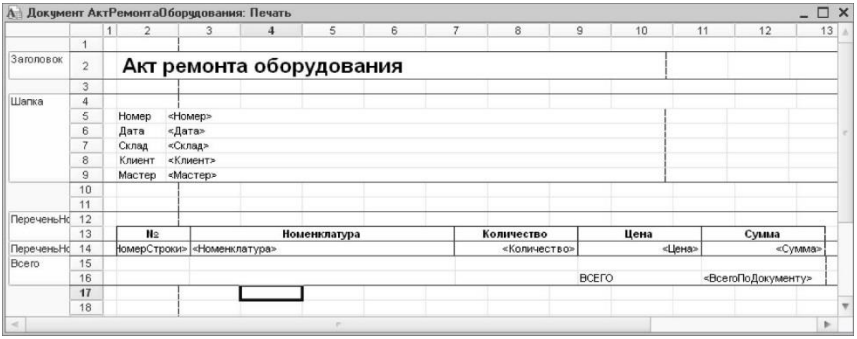
Егер нысанда негізгі деректеме белгіленсе, нысан модулінің мәнмәтінде негізгі деректемемен байланысты қосымша ерекшеліктері мен әдістер болады. Мысалы, Номенклатура анықтамалығы элементі нысанының модулінде `АнықтамалықОбъект. Номенклатура` объектісінің ерекшеліктері мен әдістері қолжетімді.

## 9.6. МАКЕТТЕР

«1С:Кәсіпорын» бағдарламаларының жүйесіндегі *макет* деп баспа нысандарын қалыптастыруға арналған конфигурация объектісі аталады.

Баспа нысандарының жалпы макеттері Макеттер тармағындағы конфигурацияның Жалпы ағаштарында орналасады, конфигурация объектілерінің баспа нысандары (анықтамалықтар, құжаттар, құжаттар журналдары, шоттар жоспарлары, сипаттама түрлерінің жоспарлары, есеп айырысу түрлерінің жоспарлары, тіркелімдер, есептер мен өңдеулер және басқа объектілер) Макеттер тәуелді объектілерінде, сондай-ақ сыртқы файлдарда (бұл жағдайда Макет кестелік құжатының қасиеті орнатылуы тиіс) орналасады. Макеттердің мынадай түрлері болуы мүмкін:

- *кестелік құжат* — макеттерді құрудың және пайдаланудың стандартты технологияларын пайдалануды көздейді. Макетті дайындау кестелік редактор арқылы жүзеге асырылады. Макеттің мысалы 9.5-суретте келтірілген;
- *екілік деректер* — екілік деректер пайдаланылады;
- *ActiveDocument* — OLE Active document технологиясын пайдалануды көздейді;
- *HTML-құжаты* — HTML-құжатының редакторын пайдалануды көздейді;




9.5-сурет. Құжаттың макеті

■ *мәтіндік құжат* — макет ретінде мәтіндік құжатты пайдалануды көздейді. Мәтіндік макетті дайындау мәтіндік макеттердің редакторы арқылы жүзеге асырылады;

■ *географиялық схема* — макет ретінде географиялық схемалардың редакторында дайындалған географиялық схеманы пайдалануды көздейді;

■ *графикалық схема* — редакторда дайындалған графикалық схеманы пайдалануды көздейді;

■ *деректерді жиынтықтау схемасы* — конструкторда дайындалған деректерді жиынтықтау схемасын пайдалануды көздейді;

■ *деректерді жиынтықтауды ресімдеу макеті* — деректерді жиынтықтау жүйесін ресімдеу макетін пайдалануды көздейді. Жиі жағдайда «Кестелік құжат» макетінің түрі қолданылады.

Кестелік құжаттардың редакторын шағын құжаттарды құру үшін, сондай-ақ көлемді тізімдеме, журналдар мен күрделі есептер үшін пайдалануға болады. Кестелік құжаттардың редакторы пайдаланушыларға ресімдеу құралдарының кең жинағын ұсынады. Ақпаратты графикалық түрде (диаграмма) шығару мүмкіндігі бар. Кестелік құжаттар редакторының басты ерекшеліктерінің бірі «1С» кіріктірілген тілі арқылы есептерді қалыптастыруға бағдарлау болып табылады.

## 9.7. КІРІКТІРІЛГЕН ТІЛДІҢ СИПАТТАМАСЫ

Кіріктірілген тіл «1С:Кәсіпорын» технологиялық платформаның маңызды бөлігі болып табылады, өйткені әзірлеушіге қолданбалы шешімнің жұмыс істеуінің жеке алгоритмдерін сипаттауға мүмкіндік

береді. «1С» кіріктірілген тілі кәсіби бағдарламашылармен ғана қолдану мүмкіндігін есепке алумен арнайы әзірленген заттық-бағдарланған бағдарламалау тілі болып табылады. Тілдің барлық операторларында орыс, сондай-ақ ағылшын тілдерде жазу мүмкіндігі бар, оларды бір уақытта бір негізгі мәтінде қолдануға болады.

Тілде өзінің біршама оңайлығына қарамастан, бірнеше объектілік-бағдарланған мүмкіндіктері бар, мысалы, деректердің мамандандырылған түрлерінің қасиеттері мен әдістеріне (құжаттарға, анықтамалықтарға және басқаларға) қолжетімділік қағидалары басқа объектілік-бағдарланған тілдерде пайдаланылатын объектілердің қаситеттеріне және әдістеріне ұқсас. Алайда деректердің мамандандырылған түрлері тілдің өз құралдарымен белгілене алмайды, олар көзбен шолу режимінде беріледі.

«1С» кіріктірілген тіл бағдарламалаудың Pascal, JavaScript, Basic секілді басқа тілдерімен анағұрлым ұқсастығы бар, бұл оны жаңадан бастап жүрген әзірлеушілерге оқуды жеңілдетеді. Бірақ ол аталған тілдердің қандай да бір аналогы болып табылмайды.

Бұдан әрі кіріктірілген тілдің аса маңызды ерекшеліктері келтірілген:

- алдын ала компиляциялау — модульді орындау алдында кіріктірілген тілдегі мәтін ішкі кодқа айналады;
- жадыда компиляцияланған модульдерді кэштеу;
- жұмсақ типтеу — жұмыс процесінде өзгере алатын және құрамында бар айнымалы түрі мағына түрімен анықталады. Айнымалының күңгірт анықтамасы оны иелену операторының сол бөлігінде алғашқы рет ескертілгенде болып табылады. Сондай-ақ тиісті оператордың көмегімен айнымалыларды жариялауға болады. Массивтерді, құрылымдарды, сәйкестіктер мен басқа әмбебап мағыналардың топтамаларын қолдануға болады;
- конфигурация объектілерінің бағдарламалық сипаттамасының болмауы — әзірлеуші платформаға кіріктірілген объектілерді немесе қолданбалы шешімді көзбен шолу үшін құрылымдау нәтижесінде жүйемен құрылған объектілерді пайдалана алады.

Қолданбалы шешімдерді «1С:Кәсіпорын» толық кодтауға қажеттілік жоқ. Әзірлеуші қолданбалы шешімнің бас бөлігін көзбен шолу үшін құрылымдау арқылы құрады – конфигурацияның жаңа объектілерін, олардың құрылым тапсырмасын, ұсыну нысанын, өзара байланысын және басқаларды құру. Кіріктірілген тіл қолбанбалы шешім үлгіліктен өзгеше объектілерінің тәртібін анықтау және деректерді өңдеудің жеке алгоритмдерін құру үшін пайдаланылады. Осы себеп бойынша кіріктірілген тілде мәтіні бар модульдер жүйемен нақты, қолданбалы шешімнің жұмыс процесінде

туындауы мүмкін алдын ала белгілі жағдайларда пайдаланылады. Бұндай жағдайларды *оқиғалар* деп атайды. Оқиғалар қолбанбалы шешім объектілерінің жұмыс істеуіне немесе қолбанбалы шешіммен байланысты болуы мүмкін. Мысалы, Анықтамалық объектісінің жұмыс істеуімен бірнеше оқиғалар байланысты, оның арасында ЖазбаАлдында оқиғасы бар. Бұл оқиға анықтамалық элементінің деректері дерекқорларға жазылуы керек алдында тікелей пайда болады. Әзірлеуші кіріктірілген тілді пайдалана отырып, алгоритмді сипаттай алады, мысалы пайдаланушымен енгізілген деректердің дұрыс болуын тексереді. Осы алгоритмді тиісті модульде орналастыра отырып, әзірлеуші пайдаланушы анықтамалық элементінің жазбасын әрбір орындаған сайын жүйе әзірлеушімен құрылған алгоритмді орындайды және пайдаланушы анықтамалықтың міндетті деректемелерін толтыруды ұмытып кетпегенін тексеруді қамтамасыз етеді. Кіріктірілген тіл бизнеслогиканы бағдарламалау тілі болып табылатындығын атап өтуге болады, ал кіріктірілген тілде модульдерді пайдалану оқиғалық-тәуелді болып табылады, яғни модульдерді орындау қолданбалы шешімнің жұмыс істеу процесінде белгілі оқиғалар болғанда болады.

Кейбір конфигурация объектілерінің негізгі қасиеттері Деректер түрі болып табылады. Бұл қасиет конфигурация объектісінің қандай ақпаратты қамтуы мүмкін екенін анықтайды. Конфигурация объектісінің деректер түрі конфигурацияны баптау процесінде объект қасиеттерін құру немесе редакциялау кезінде тағайындалады.

Объектіде, «1С:Кәсіпорын» жүйесіндегі ақпарат түрі көрсетілуі мүмкін конфигурация объектілері *конфигурацияның типтелген объектілері* деп аталады.

Анықтамалық, Құжат, Өңдеу секілді конфигурация объектілері типтелген объектілер болып табылмайды, өйткені онда «кешенді» ақпарат бар және ол өз кезегінде конфигурацияның типтелген объектілерін қамтиды.

Конфигурация объектісі қабылдай алатын деректердің түрін екі топқа бөлуге болады. Бірінші топты деректердің қарапайым түрлері құрайды: Саны, Жолы, Күні және Булево. Тиісінше, конфигурация объектісінде сақталатын ақпарат сан, таңбалардың ерікті жолы, күні немесе логикалық шама болуы мүмкін. Бұдан басқа «1С:Кәсіпорын» жүйесі конфигурациясының кейбір объектілері деректер түрлерін құруы мүмкін. Мысалы, константта ҚұжатСілтеме деректерінің түрі тағайындалуы мүмкін. Бұл жағдайда константа мағынасы

«1С:Кәсіпорын» жүйесіндегі қолданыстағы құжаттардың біріне сілтемені ұсынатын болады.

«1С:Кәсіпорын» жүйесінде конфигурация мағыналарының түрлерін қалыптастыра алатын конфигурация объектілері конфигурацияның *үлгі құратын объектілері* деп аталады. «1С:Кәсіпорын» жүйесіндегі осындай объектілер анықтамалықтар, құжаттар, жоспарлар шоттары, есептеу, аударым түрлерінің жоспарлары және басқалар.

## 9.8. БАҒДАРЛАМАЛЫҚ МОДУЛЬДІҢ ҚҰРЫЛЫМЫ

Бағдарламалық модульдің құрылымын мынадай бөлімдерге бөлуге болады: айнымалыларды анықтау бөлімі; процедуралар мен функциялар бөлімі; негізгі бағдарлама бөлімі.

Нақты бағдарламалық модульде бөлімнің кез келгені болмауы мүмкін.

*Айнымалы шаманы анықтау бөлімі* модуль мәтінінің басынан бастап Процедура бірінші операторына немесе Функция операторына немесе кез келген орындаушы операторға дейін орналастырылады. Бұл бөлімде тек айнымалыларды хабарлау операторлары болуы мүмкін.

*Процедуралар мен функциялар бөлімі* Процедура бірінші операторынан немесе Функция операторынан бастап процедураларды немесе функцияларды денеден тыс сипаттау кез келген орындаушы операторға дейін орналастырылады.

*Негізгі бағдарлама бөлімі* процедуралардан немесе функциялардан тыс бірінші орындалатын операторлан бастап модульдің соңына дейін орналастырылады. Бұл бөлімде тек орындалатын операторлар ғана болуы мүмкін. Негізгі бағдарлама бөлімі модульдің инициализациясы кезінде орындалады. Әдетте негізгі бағдарлама бөлімінде айнымалыларды инициализациялау операторларын модульдің кез келген процедураларынан немесе функцияларынан бірінші шақыруға дейін өткізуге қажетті қандай да бір нақты мағыналармен орналастыру мәні бар.

Бағдарламалық модульдің бастапқы мәтіні операторлардан және түсіндірмелерден тұруы мүмкін. Бағдарламалаудың жақсы бағыты болып бастапқы мәтінде алгоритмді сипаттаумен егжей-тегжейлі түсіндірмелердің болуы саналады. Бағдарламалық модульдің мәтінінде түсінік «//» қос белгімен басталып, жолдың шетімен аяқталады.

Операторларда иелену операторын (A = B;) және кіріктірілген тілдің синтаксистік құрылымын (мысалы, Үшін, Әзірше, Егер) қоспағанда процедураға стандартты қарау түрі бар. Операторлардың арасын «нүктелі үтір» белгісімен міндетті

түрде бөлу қажет. Жолдың соңы оператор соңының белгісі болып табылмайды.

Нақты бағдарламалық модульді орналастыру орны жұмыс істеудің ерекше алгоритмдерін сипаттауды талап ететін конфигурацияның сол нүктелерінде конфигуратормен ұсынылады. Бұл алгоритмдерді алдын ала көзделген жағдайларда (мысалы, диалогтық терезеде түймені басқан кезде) жүйемен шақырылған процедуралар немесе функциялар түрінде ресімдеген жөн.

Процедура негізгі сөзі бастапқы мәтіннің секциясын бастайды, оның орындалуы параметрлер тізімімен (егер параметрлер берілмесе, онда дөңгелек жақшадалар міндетті) тек ПроцедураАтауын() ғана көрсетіп бағдарламалық модульдің кез келген нүктесінен бастамашылық жасауға болады. Егер қосымша модулінде немесе жалпы бағдарламалық модульде процедураны сипаттау кезінде негізгі сөз Экспорт пайдаланылса, онда бұл конфигурацияның барлық басқа бағдарламалық модульдерінен аталған процедура қолжетімді екенін көрсетеді. Процедураның бағдарламалық секциясының соңы ПроцедураныңСоңы операторы бойынша айқындалады.

Жергілікті айнымалыларды хабарландыру бөлімінде процедурада жарияланған айнымалылар осы процедураның жергілікті айнымалысы болып табылады, сондықтан олар тек осы процедурада қолжетімді (өзге процедураларды, функцияларды немесе әдістерді шақыру кезіндегі параметрлер ретінде оларды беру жағдайын қоспағанда).

Процедура, ПроцедураСоңы негізгі сөздері операторлар емес, олар операторлық жақшалар болып табылады, сондықтан олар нүктелі үтірмен аяқталмауы тиіс (бұл модульді орындауда қатерлерге әкелуі мүмкін).

Синтаксис:

```
Процедура <ПроцедураныңАты>([[Мағына] <Парам 1>
[=<Деф- Мағына>], ... , [Мағына] <Парам N>
[=<ДефМағына>]])[Экспорт]
// Жергілікті айнымалы шамаларды жариялау;
// Операторлар;
...
[Қайтару;]
// Операторлар;
...
ПроцедураныңСоңы
```

Функция негізгі сөзі функцияның бастапқы мәтінінің секциясын бастайды, оны бағдарламалық модульдің кез келген нүктесінен бастамашылық етіп орындауға болады, тек ФункцияАтын параметрлер

тізімімен (параметрлер берілмеген жағдайдың өзінде дөңгелек жақшалар міндетті) көрсету қажет. Егер қосымша модулінде немесе жалпы бағдарламалық модульде функцияны сипаттауда `Экспорт` негізгі сөзі пайдаланылса, бұл аталған функция конфигурацияның басқа бағдарламалық модульдерінің ішінен қолжетімді болып табылатынын білдіреді.

Функцияны орындау Қайтару операторымен аяқталады. Функциялар процедуралардан `ҚайтарылатынМағына` қайтарумен ғана ерекшеленеді. Функцияның бағдарламалық секциясының соңы `ФункцияныңСоңы` операторы бойынша айқындалады.

Бағдарламалық модульдің мәтініндегі кез келген функцияны шақыруды процедураны шақыру ретінде жазуға болады, яғни тілде функциядан қайтарылған мағынаны қабылдамауға болады. Егер функцияда Қайтару негізгі сөзі көрсетілмесе немесе оның құрамындағы модульдің жолы орындалмаса, функция Белгісіз түрінің мағынасын қайтарады.

Жергілікті айнымалылар жарияланған бөлімдегі функция ішінде жарияланған айнымалылар осы функцияның жергілікті айнымалылары болып табылады, сондықтан олар тек осы функцияда қолжетімді (өзге процедураларды, функцияларды немесе әдістерді шақыру кезінде оларды параметрлер ретінде беру жағдайын қоспағанда). Функция, ФункцияныңСоңы негізгі сөздері операторлар емес, операторлық жақшалар болып табылады, сондықтан олар нүктелі үтірмен аяқталмауы тиіс (бұл модульді орындауда қатерлерге әкелуі мүмкін).

Синтаксис:

```
Функция <ФункцияАты>([[Мағына] <Парам 1>[=<ДефМағына>], ... ,[Мағына] <Парам N>[=<ДефМағына>]])[Экспорт]
// Жергілікті айнымалы шамаларды жариялау;
// Операторлар;

Қайтару <Қайтарылатын мағына>;
// Операторлар;
```

## БАҚЫЛАУ СҰРАҚТАРЫ

1. «1С» платформасы нені білдіреді?
2. «1С» конфигурациясы дегеніміз не?
3. «1С:Кәсіпорын» жүйесінің қандай жұмыс режимдері бар?
4. «1С:Кәсіпорын» жүйесінде конфигурация объектісі нені білдіреді?

5. Метадеректер технологиясының мәні неде?
6. Конфигурацияның әр объектісі қандай бірегей қасиеттер жинағына ие болады?
7. Конфигурация объектілерінің кестелік бөліктері не үшін пайдаланылатынын түсіндіріңіз.
8. Командалық интерфейс нені білдіреді?
9. «1С» жүйесінде рөл деп нені айтамыз?
10. Пайдаланушыға рөльдерді тағайындау операциясы қандай негізгі міндеттерді шешетінін көрсетіңіз.
11. «1С» жүйесіндегі қосымша жүйелер деп нені айтады?
12. Неге кез келген қолданбалы шешімді әзірлеу қосымша жүйелердің құрамын жобалаудан басталады?
13. Конфигурация объектісі ретінде тұрған нысанның бөлшектерін атаңыз.
14. Пайдаланушыға көрсету үшін нысанның сыртқы түрі қалай қалыптасады?
15. Модульге анықтама беріңіз.
16. Модульдің жергілікті және жаһандық мәнмәтіні нені білдіреді?
17. Макет дегеніміз не?
18. «1С» макеттердің түрін атаңыз.
19. «1С» кіріктірілген тілдің аса маңызды ерекшеліктері қандай?
20. Бағдарламалық модуль қандай бөлімдерден тұрады?
21. Қандай объектілерді типтелген деп атайды?
22. Қандай объектілерді типті құрайтын деп атайды?
23. Қосымшаның негізгі терезесінде қандай ақпарат көрсетіледі?
24. Қосымшаның қосалқы терезесінде қандай ақпарат көрсетілетінін атаңыз.
25. Бөлімдер панелінде қандай ақпарат көрсетіледі?
26. Навигация панелі не үшін пайдаланылады?
27. Іс-әрекет панелі не үшін пайдаланылады?
28. Қосымшаның Жұмыс үстелі не үшін пайдаланылады?
29. Қосымшаның командалық интерфейсі нені білдіреді?

# 10 тарау

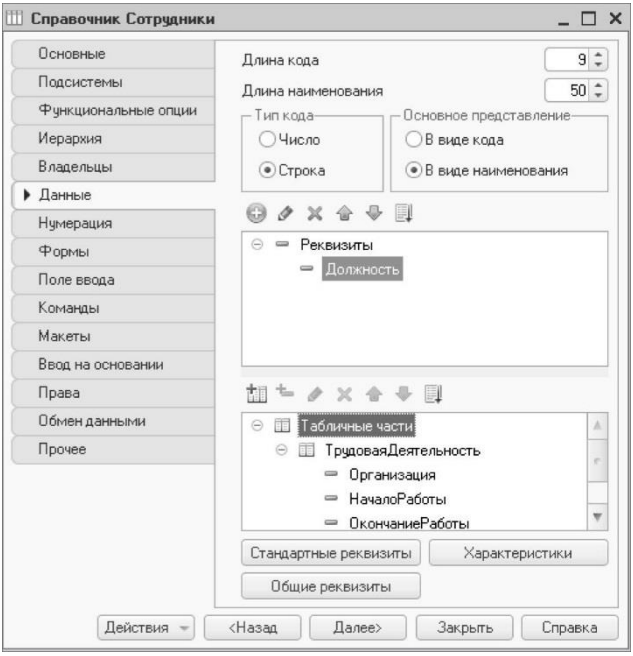
# «1С» ПЛАТФОРМАСЫНЫҢ ҚОЛДАНБАЛЫ МЕХАНИЗМДЕРІ

## 10.1. ШАРТТЫ-ТҰРАҚТЫ АҚПАРАТТЫ САҚТАУ

«1С» конфигурация объектілері технологиялық платформа деңгейде сүйемелденетін пәндік-бағдарланған нысандарды білдіреді. Әзірлеушінің міндеті осы объектілерден қолданбалы шешімге қажетті құрылымды құру және осы объектілердің жұмыс істеуін және өзара іс-қимыл алгоритмдерін әзірлеу болып табылады. Әзірлеуші қолдану саласын және пайдаланушылардың талаптарын талдайды, конфигурация объектілерін құрады, олардың арасында байланыс орнатады, экрандық нысандарды және есеп макеттерін визуалды құрастырады, конфигурацияның белгілі бір нүктелерінде бағдарламалық модульдер жазады. Нәтижесінде қолданбалы шешім пайда болады. Қолданбалы шешімнің (конфигурацияның) құрылымы конфигурация объектілерінің құрамымен және олардың арасындағы өзара байланыспен айқындалады.

**Анықтамалықтар.** Анықтамалық түріндегі объектілер жүйедегі кейбір көптеген мағыналармен тұрақты және шартты тұрақты ақпаратпен жұмыс істеу үшін пайдаланылады. Әдетте, анықтамалықтар ретінде материалдар, тауарлар, ұйымдар, валюта, қызметкерлер және басқалар тізімі болып табылады. Анықтамалықтарды сүйемелдеу механизмі түрлі анықтамалықтарды жобалауға және сүйемелдеуге мүмкіндік береді.

Анықтамалық элементтерден тұрады. Мысалы, студенттер анықтамалығы үшін элемент болып студент, ал материалдар анықтамалығы үшін – материал және т.б. Пайдаланушының қосымшамен жұмыс істеу процесінде элеметтерді өздігінен қосуға, жоюға немесе редакциялауға мүмкіндігі бар. Әрбір элементте жиі жағдайда қосымша ақпарат болады, ол сол элементті толық сипаттайды. Сипаттау үшін Анықтамалық конфигурациясы объектісінің деректемелері пайдаланылады (10.1-сурет). Мысалы, тауар үшін ол сатып алу және сатылым бағасы,




10.1-сурет. Конфигуратордың режимінде анықтамалықты редакциялау терезесі

өнім беруші болуы мүмкін, қызметкер үшін – лауазымы, кәсібі, тұрғылықты жердің мекенжайы және басқалар болуы мүмкін. Деректемелер сондай-ақ конфигурация объектілері болып табылады. Конфигурацияның осы объектілері Анықтамалық объектісімен логикалық байланысты болғандықтан, олар осы объектіге тәуелді болады. Анықтамалық конфигурациясының әрбір объектісінде үнсіз келісім бойынша стандартты деректемелер жинағы болады (10.2-сурет). Осы


10.2-сурет. *Анықтамалықтың* стандартты деректемелері

деректемелердің қолжетімділігі анықтамалықтың сипатына байланысты болады. Мысалы, егер анықтамалық иерархиялық болса, онда `Ата-ана` сипаты қолжетімді болады.

Анықтамалықтарда кестелік бөліктерді сақтау мүмкіндігі бар, мысалы, бұйымдардың жинақтау құрамының, қызметкердің отбасы құрамының, клиент телефонының сипаттамасы. Бұл ақпарат барлық құрылым бойынша бірдей, бірақ санымен ерекшеленеді (әртүрлі бұйымдардың және басқалардың құрамдас сандары әртүрлі). Бұл жағдайда, дерекқорда анықтамалықтың нақты элементіне бағынысты кестелік бөліктерін сақтау үшін қосымша кестелер құрылады.

Әрбір анықтамалық үшін нысандардың: элемент, топ, тізім, таңдау, топты таңдау бірнеше түрі берілуі мүмкін. Нысандардың әрбір түрі бойынша нысандардың ықтимал саны құрылуы мүмкін.

Бағыныңқы анықтамалықтарды құруға болады, онда әр элемент анықтамалықтың белгілі бір элементіне-иесіне «тиесілі» болады.

**Константтар.** «1С:Кәсіпорын» жүйесіндегі Константа конфигурациясының объектісі уақыт өте өзгермейтін немесе өте сирек өзгеретін ақпаратты сақтауға арналған. Бұл ретте, константтың алдыңғы мағыналары маңызды емес. Егер қандай да бір деректердің мағыналары уақыт өте келе өзгеруі мүмкін немесе уақытты есепке ала отырып, мағыналар қажет болуы болжамы болса, онда мұндай деректерге константаны емес өлшемдері жоқ мәліметтер тіркелімін пайдалану қажет.

Константада редакцияланатын қасиеттер жинағы бар, олардың ішінде ең маңыздысы константа атауы, синоним, деректер түрі болып табылады. Константаларда кәсіпорынның атауы, оның ЖСН, директордың және бас бухгалтердің тегі және басқа толық ақпарат сақталуы мүмкін. Жүйеде константалардың шексіз саны сипатталуы мүмкін.

**Аударымдар.** Аударымдар «1С:Кәсіпорын» жүйесінде конфигурацияның жұмыс процесінде өзгермейтін мағыналардың тұрақты жиынтығын сипаттау үшін пайдаланылады. Анықтамалыққа қарағанда аудару мағыналары конфигурациялау кезеңінде тапсырылады және орындау кезеңінде өзгере алмайды. Аударымдардың әдеттегі мысалдары төлем түрлері («қолма-қол», «аударма», «айырбас»), жынысы («ер», «әйел») және басқалар болып табылады. Пайдаланушы аударудың жаңа мағынасын немесе қолданыстағыны өшіре алмайды, ол тек қолданыстағы тізімнен таңдай алады. Анықтамалықтар үшін аударымдарға қарағанда нақты мағыналар пайдаланушымен әдетте бағдарламамен жұмыс істеу уақытында енгізіледі.

**Сипаттама түрлерінің жоспарлары.** Сипаттама түрлерінің жоспары арқылы конфигурацияны әзірлеу кезіне белгісіз объектілердің қасиеттерін сақтауды ұйымдастыруға болады. Бұл пайдаланушы өздігінен

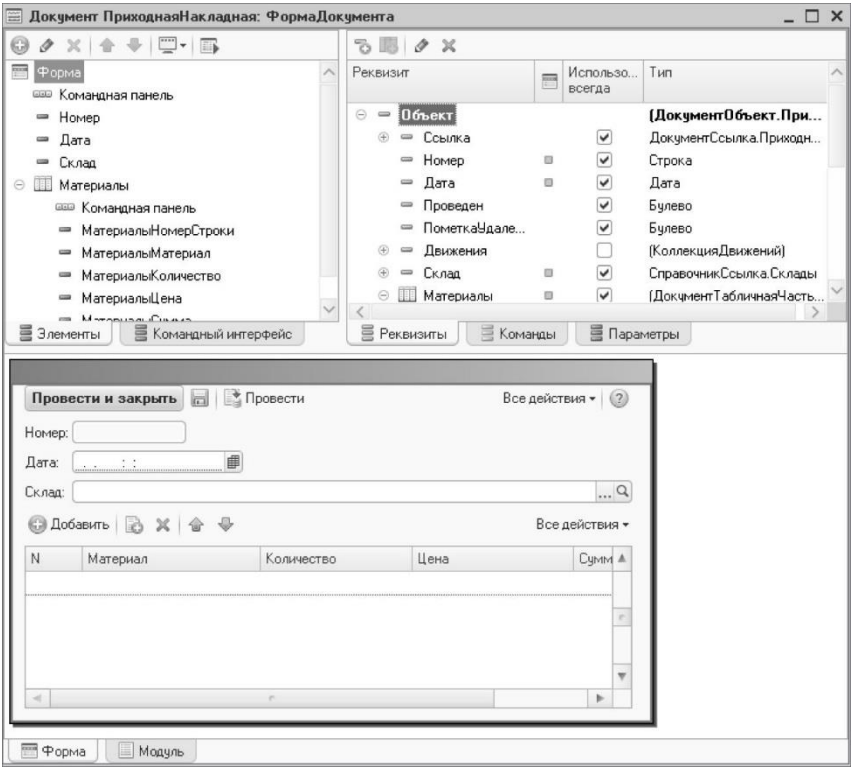
жаңа қасиеттерді енгізе алатынын білдіреді, мысалы, түс, көлем, аумақты көлемдер, қуаттылық. Тауарлардың әрбір тобы үшін қасиеттердің өз жинағы болуы мүмкін: компьютерлер үшін – жедел жады көлемі, қатты диск көлемі; киім үшін – өлшемі, бойы, түсі және т.б. Содан кейін, осы сипаттамалардың негізінде есептер құруға, сату көлемін талдауға, шешімдер қабылдау үшін құнды ақпаратты алуға болады.

**Шоттар жоспарлары.** Шоттар жоспарлары бухгалтерлік немесе басқарушылық есепке алу үшін шоттар тізімін қамтиды. Барлық жазбалар сәл төменде сипатталған бухгалтерия тіркелімдерінде сақталады. Бір жазбаны басқа жазбалардан бөлу үшін белгілі бір шоттар жоспарында негізделген бухгалтерияның жаңа тіркелімі құрылады.

**Есептеу түрінің жоспарлары.** Есептеу түрлерінің жоспары бір-біріне бір нәрсесі ортақ: есептеудің бірдей негізгі түрлері, қайта есептеудің бірдей қағидалары, мерзім бойынша жалпы ығыстыру қағидалары бар есептеу түрлерін қамтиды. Мысалы, «Негізгі есептеу», «Салықтар» жоспарлары. Жоспарлардың негізінде тікелей есеп айырысу жазбалары бар есептеу тіркелімдері құрылады. Есептеу түрлері деректер объектісі ретінде есептеу түрлерінің жоспарларында сақталады, яғни олар орындау режімінде енгізілуі мүмкін. Конфигурация құрылатын есептеудің алдын ала белгіленген түрлерін енгізуге, онда конфигурация құрылатын болады және жетекші және негізгі есептеу түрлерін ығыстыру қағидалары үшін оларды жасауға болады.

## 10.2. ҚҰЖАТТАР

Автоматтандырылатын қолдану саласына қатысы бар кәсіпорынның немесе ұйымның оқиғаларын көрсетуге арналған *құжаттар*. Құжаттарға енгізілетін деректер (құжаттың деректемелері және кестелік бөліктері) әдетте өткен оқиға туралы ақпаратты қамтиды. Мұндай оқиғалар материалдардың түсуі, банк арқылы ақша аудару, қызметкерді жұмысқа қабылдау және т.б. болуы мүмкін. Конфигурациялау процесінде құжаттар түрінің ерікті саны құрылуы мүмкін. Құжаттың әрбір түрі өз оқиғалар түрін көрсету үшін арналған. Бұл конфигурацияда сипатталатын оның құрылымын және қасиетін айқындайды. Құжаттың әрбір түрінде деректемелер мен кестелік бөліктердің шексіз саны болуы мүмкін. Бірнеше кестелік бөліктер мәні әртүрлі, бірақ оқиғалары байланысты бір құжатпен тіркеу қажет болғанда талап етіледі. Мысалы,




10.3-сурет. Конфигуратор режиміндегі құжаттардың нысаны

«Жабдықтарды жөндеу актісі» құжаты үшін кестенің бір бөлігінде атқарылған жұмыстардың түрі, ал кестенің басқа бөлігінде – жұмсалған материалдардың тізбесі тіркелуі мүмкін.

Құжат үшін енгізу нысаны құрылады — нақты құжаттардың экрандық аналогтары (10.3-сур.). Құжаттар тізімін қарау үшін тізімдер нысандары құрылады.

Барлық құжаттар нөмірмен, күні және уақытымен сипатталады. Құжат есептің жай-күйіне өзгерістер енгізітіндіктен, ол уақыттың нақты мерзіміне үнемі «байлаулы» болады. Бұл дерекқорда оқиғалардың нақты дәйектілігін көрсетуге мүмкіндік береді.

Құжат конфигурациясының әрбір объектісінде үнсіз келісім бойынша стандартты деректемелердің жинағы бар (10.4-сур.).

*Құжаттардың журналы* әртүрлі құжат түрлерін көруге арналған. Құжаттың әрбір түрі бірнеше журналда көрсетілуі мүмкін. Құжаттар журналы жүйеге жаңа деректерді қоспайды,


10.4-сурет. *Құжаттың* стандартты деректемелері

ол бірнеше түрлі құжаттардың бірыңғай тізімінде көрсетуге арналған құрал болып табылады.

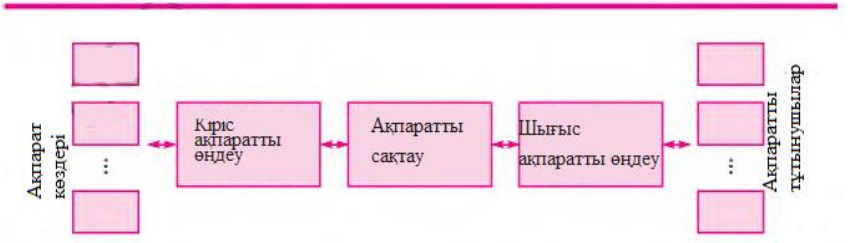
Нөмірлеуіш қызметтік объектілердің көмегімен әртүрлі құжаттарға «толассыз» нөмірлер ұйымдастыруға болады. Реттілік қызметтік объектісі құжаттарды өткізудің қатаң тәртібі арқылы тіркелімдер бойынша қозғалыстардың дұрыстығын сүйемелдеуге арналған.

## 10.3. АҚПАРАТТЫ ӨҢДЕУ ЖӘНЕ ШЫҒАРУ

Қолданбалы шешім бірнеше функционалдық блоктар (қосымша жүйелер) жиынтығы ретінде жалпы түрінде ұсынылуы мүмкін – кіріс деректерімен жұмыс істеу блогы, деректерді ДҚБЖ сақтау және кіріс ақпаратты өңдеу және есептерді қалыптастыру блогы (10.5-сур.).

Өңдеу және есептерді енгізілген дерекқорларда пайдаланушыға өңдеу мен ұсыну үшін пайдаланады (мөрлер). *Өңдеулер* ақпарат базасында іс-әрекеттер мен есептеулерді орындау үшін арналған, ал *есептер* мысалы, әртүрлі баспа нысандарын қалыптастырады.

Есептің кез келген автоматтандыру жүйесінде жиналған ақпаратты өңдеу және қарау мен талдау түрінде ыңғайлы жиынтық деректерді алу үшін құралы болуы тиіс.

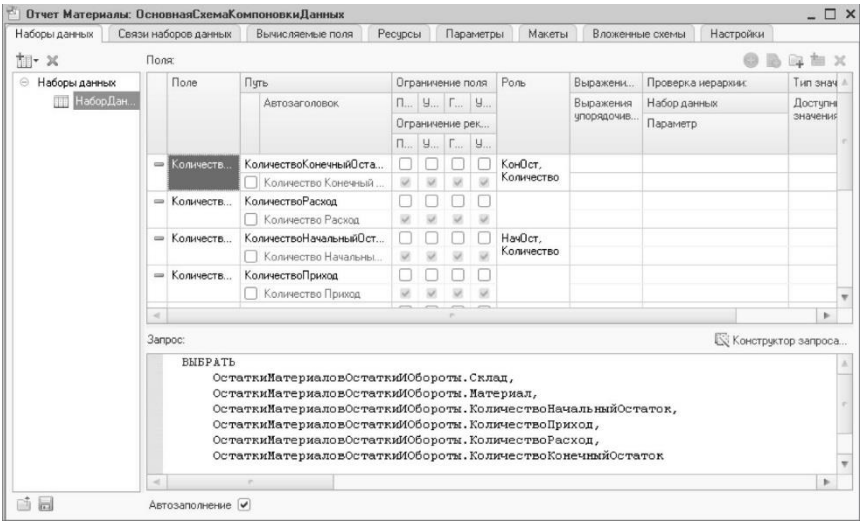



10.5-сурет. Қолданбалы шешімнің функционалдық блоктары

Есеп конфигурация объектісі пайдаланушы өзіне қажетті шығыс мәліметтерді ала алатын алгоритмдерді сипаттауға арналған. Шығыс деректерін қалыптастыру алгоритмі көзбен шолу құралдарымен немесе кіріктірілген тілді пайдалана отырып, сипатталады. Нақты өмірде Есеп конфигурациясының объектілеріне шығыс деректерінің түрлі кестелері, жиынтық деректер, диаграммалар және т.б. сәйкес келеді.

Есеп бір немесе бірнеше нысандарды қамтуы мүмкін, олардың көмегімен, қажет болғанда алгоритм барысына әсер ететін қандай да бір параметрлерді енгізуді ұйымдастыруға болады. Алгоритмді орындау қорытындыларын экранға және принтерге шығару үшін конструктордың көмегімен құрылған баспа нысандарын (макеттерді) сипаттау макеттері болуы мүмкін. Объектілер қасиеттерін редакциялау және бағынышты объектілерді құру редакциялау терезесінде орындалады.

Күрделі экономикалық және талдамалық есептерді қалыптастыру үшін «1С:Кәсіпорын» платформасында күшті және икемді механизм – *деректерді біріктіру жүйесі* – ресми түрдегі сипаттамасы бар есептің арнайы макеті (10.6-сур.) бар. Деректерді біріктіру схемасы әзірлеушімен құрылады және деректер жинағының сипаттамасын (есептер үшін ақпарат көздері) және олардың арасындағы байланысты, деректерді алу параметрлерін, есеп жолдарын қамтиды, сондай-ақ



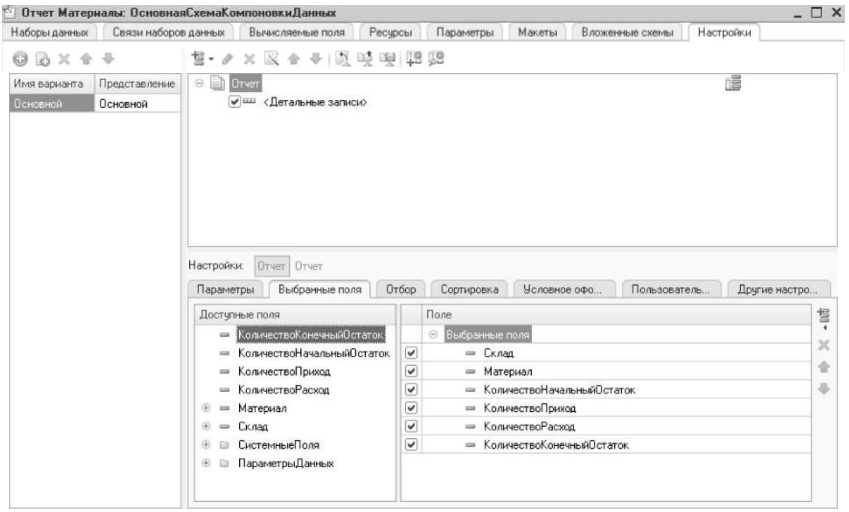

10.6-сурет. Деректерді біріктіру схемасы. Деректердің жиыны

онда әзірлеуші деректерді біріктірудің стандартты баптауларын береді – есептің құрылымы, тәртібі, іріктелуі және басқалар. Әрбір есепте әзірлеуші диаграммалар, кестелер немесе жолдардың, қорытындылардың және т.б. басқа құрамы бар топтар түрінде деректерді ұсынатын баптаулардың бірнеше нұсқаларын көздеуі мүмкін. Әзірлеуші пайдаланушыға қандай есептің баптаулары қолжетімді болатынын (пайдаланушы баптаулары) және осы баптаулардың қайсысы есеп нысанында (жедел пайдаланушы баптаулары) тікелей қатысатынын сипаттай алады. Олардың көмегімен пайдаланушы есептің нұсқасын өзіне ыңғайлы етіп баптай және одан одан әрі пайдалану үшін сақтай алады.

Деректерді біріктіру схемасы базаны білдіреді, соның негізінде түрлі есептер қалыптасуы мүмкін және олар мыналарды қамтуы мүмкін:

- деректерді біріктіру жүйесінің нұсқаулықтары бар сұрау салынған мәтін;
- бірнеше деректер жиынтығының сипаттамасы;
- қолжетімді жолдардың толық сипаттамасы;
- деректерді бірнеше теру арасындағы байланыстар сипаттамасы;
- деректерді алу параметрлерін сипаттау;
- жолдар мен топтардың макеттерін және т.б. сипаттау.

Деректерді біріктіруді баптау деректерді біріктірудің кейбір белгіленген схемасында әзщірлеуші немесе пайдаланушы баптай алатын барлық әрекетті сипаттайды (10.7-сур.). Деректерді біріктіру макеті




10.7-сурет. Деректерді біріктіру схемасы. Баптаулар

біріктіру схемасына нақты баптауларды қолдану нәтижесін білдіреді және нақты баптауларды есепке алумен, есептің қажетті құрылымын қалыптастыруға біріктіру процесіне дайын тапсырма болып табылады. Деректерді біріктіру жүйесі есепті біріктіреді және оны пайдаланушыға кестелік құжат ретінде шығарады (деректерді бағдарламалық түрде мағыналар кестесіне, мағыналар ағашына және т.б. шығаруға болады).

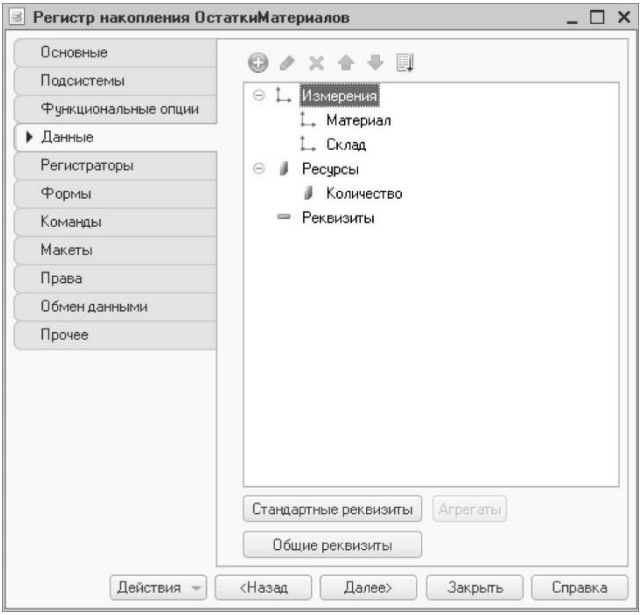
Пайдаланушы өзінің көзқарасы бойынша анағұрлым ақпаратты болып табылатын есептің нұсқасын таңдай алады, ал қалаған жағдайда тиісті біліктілігі бар, өз баптауларын тапсыруға және деректерді біріктірудің сол схемасында негізделген басқа есеп ала алады. Көптеген есептерді әзірлеуші деректерді өңдеу біріктіру схемасында сипатталып, бапталуы тиіс.

«1С:Кәсіпорын» жүйесінде ақпаратқа әртүрлі іс-әрекеттерді орындау үшін Өңдеу конфигурация объектісі пайдаланылады. Мысалы, Өңдеу көмегімен жүйеден ескірген деректерді жоюға, басқа жүйелерден ақпаратты импорттауды және тағы басқаларды орындауға болады. Бұл жағдайда орындалатын іс-әрекеттердің сипаты Өңдеу конфигурация объектісінің атауын көрсетеді, өйткені нәтижесінде, жүйеде сақталатын ақпарат қандай да бір өзгерістерге ұшырайды. Өңдеу бір немесе бірнеше нысандарды қамтуы мүмкін, олардың көмегімен, қажет болған жағдайда, алгоритм барысына әсер ететін қандай да бір параметрлерді енгізуді ұйымдастыруға болады. Алгоритмді орындау нәтижелерін экранға және принтерге шығару баспа нысандарын (макеттер) сипаттау макеттері конструкторының көмегімен жүзеге асырылады. Есептің өңдеуден негізгі айырмашылығы деректерді біріктіру схемасын пайдалану мүмкіндігінде. Қалғанында есептің өңдеулен айырмашылығы жоқ.

Жүйе конфигурацияның өзінде емес, жеке файлдарда сақталатын сыртқы өңдеуді әзірлеу мүмкіндігін сүйемелдейді.

## 10.4. ЖИНАҚТАУ ТІРКЕЛІМДЕРІ. ҚҰЖАТТАРДЫ ЖҮРГІЗУ

Тіркелімдерде, әдетте, объектілердің жай-күйінің өзгеруі туралы ақпарат немесе қолдану сала объектілерін тікелей көрсетпейтін басқа ақпарат сақталады. Мысалы, тіркелімдерде тауарлардың кірісі мен шығысы туралы, валюта бағамдарының өзгерістер туралы ақпарат сақталуы мүмкін. Жинақтау тіркелімдері дерекқорларда

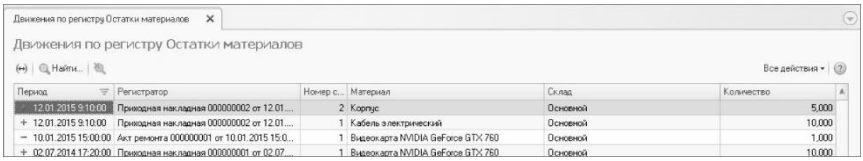



10.8-сурет. Жинақтау тіркелімінің құрылымы

әрі қарай ыңғайлы талдау үшін ақпаратты жинақтау үшін құрылымдарды құруға мүмкіндік береді. Бұл құралдардың қозғалысын (қаржы, тауарларды, материалдарды және т.б.) есепке алу механизмнің негізін құрайтын конфигурацияның қолданбалы объектілері, ол қоймалық есеп, есеп айырысу, жоспарлау секілді бағыттарды автоматтандыруға мүмкіндік береді.

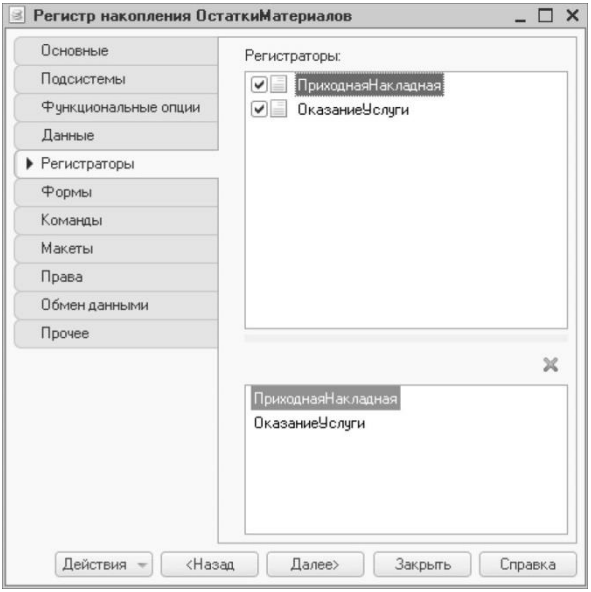
Жинақтау тіркелімі өлшеудің көп камералы өзгерістерін болдырады және бірнеше өлшем бөлінісінде сандық деректерді жинақтауға мүмкіндік береді. Мысалы, мұндай тіркелімде номенклатура және қойма бөлінісінде материалдардың қалдықтары туралы ақпаратты жинақтауға болады (10.8-сур.). Жинақтау тіркеліміндегі ақпарат жазбалар түрінде сақталады, олардың әрқайсысында өлшем мағыналары және оларға сәйкес келетін ресурстар мағыналары бар. Жинақтау тіркелімімен жинақталатын сандық ақпараттың түрлері ресурстар деп аталады, олар бағынысты объектілер болып табылады және конфигураторда сипатталады.

Жинақтау тіркелімінің жай-күйін өзгерту, әдетте, құжатты жүргізу кезінде жүзеге асырылады және тіркелімге біраз жазбалар саны қосылады. Әрбір жазбада өлшем мағыналары, ресурстардың үстелу мағыналары, осы өзгерістерді тудырған құжатқа сілтеме бар




10.9-сурет. Жинақтау тіркеліміндегі материалдардың қозғалысы туралы ақпарат

(тіркелім), үстелу «жолдамасы» (кіріс немесе шығыс) және күні (10.9-сур.). Бұндай жазбалар жиынтығы қозғалыстар деп аталады. Жинақтау тіркелімінде жазбалар құра алатын құжаттардың құрамы әзірлеушімен қолданбалы шешім құру процесінде қойылады (10.10-сур.). Тіркелімде қалыптастырылатын алгоритмдер қолданыстағы құжаттардың процедураларында кіріктірілген тілдің құралдарымен сипатталады. Жүйе құрамында қозғалыстар конструкторы бар, ол әзірлеушіге құжаттарды жүргізу алгоритмдерін құруға мүмкіндік береді. Конструктор жазба енгізілетін тіркелімдерді таңдауға және тіркелім жолдарына жазылатын түрін автоматтандырылған түрде немесе қолмен жазбаларды енгізуге мүмкіндік береді. Конструктор жұмысының нәтижесі




10.10-сурет. Жинақтау тіркелімінің тіркеуіштері

Өткізуді өңдеу атымен кіріктірілген тілдегі дайын процедура болып табылады. Бұл процедура құжаттың модулінде орналасады және құжатты жүргізу сәтінде жүйемен шақырылатын болады.

Жүйе жинақтау тіркелімінде сақталатын жазбалардың бірегейлігін бақылауды қамтамасыз етеді. Осының арқасында жинақтау тіркелімінде бір құжатттың бір жолына жататын екі жазба бола алмайды.

Жинақтау тіркелімінің екі түрі бар: қалдықтары жинақтау тіркелімдері және айналымдарды жинақтау тіркелімдері.

*Қалдықтарды жинақтау тіркелімі:* ресурстардың қорытынды мағыналары – қалдықтары, сондай-ақ осы ресурстардың өзгерістері – айналымдарды сақтауға мүмкіндік береді. Айналымдарды жинақтау тіркелімі ресурстар өзгерістері – айналымдарды сақтауға ғана мүмкіндік береді. Экономикалық қызметті автоматтандыру кезінде тек айналымдарды жинақтауға талап етілетін, ал қалдықтардың мағыналарың маңызды емес жағдайлардың көп саны болады. Айналымдарды жинақтау тіркелімін пайдаланудың типтік мысалы сату көлемі туралы ақпаратты сақтайтын тіркелім болып табылады. Жинақтаудың айналым тіркелімдері үшін платформа көптеген жазбарды қамтитын тіркелімдерден деректер алуды айтарлықтай жылдамдатуға мүмкіндік беретін агрегаттардың арнайы механизмін сүйемелдейді. Бұл есептерді қалыптастыру уақытын айтарлықтай қысқартуға мүмкіндік береді, ол әсіресе үлкен ақпараттық базалар үшін маңызды болып табылады. Әрбір агрегат – бұл осы ақпараттық базада есептерді қалыптастыру үшін ыңғайлы, әртүрлі бөліктерде тіркелімнің біріктірілген деректерін қамтитын мамандандырылған қойма. Агрегаттарды пайдалану сарапшылар мен менеджерлерге жүйенің аз уақытымын жауап беруді көрудің әртүрлі бөлшектерінің арасында қайта қосылумен қолда бар ақпаратты талдауға мүмкіндік береді. Бұнымен қоса жүйе бір жақтан жинақталған біріктірілген деректерді пайдаланады, ал екінші жақтан – алынатын есептердің өзектілігін қамтамасыз етеді.

Әзірлеушіге ұсыналатын жинақтау тіркелімінің негізгі функционалдық мүмкіндіктері мыналар болып табылады:

- берілген критерийлер бойынша берілген арақашықтықтағы жазбаларды таңдау;
- жазбаларды тіркелім бойынша таңдау;
- өлшемдердің берілген мағыналары бойынша көрсетілген уақытта қалдықтарды және айналымдарды алу;
- тіркелімге жазбаның анағұрлым параллельдігін қамтамасыз ететін қорытындыларды бөлумен жұмыс режімі;
- ағымдағы қорытындыларды пайдалануды сөндіру;
- көрсетілген мерзімге қорытындыны есептеу;

- тіркелімге жазбаларды оқу, өзгерту және жазуды теру;
- қорытындыларды қайта есептеусіз тіркелімге жазу мүмкіндігі;
- көрсетілген мерзім үшін қорытындыларды толық қайта есептеу және қорытындыларды есептеу.

## 10.5. МӘЛІМЕТТЕР ТІРКЕЛІМДЕРІ

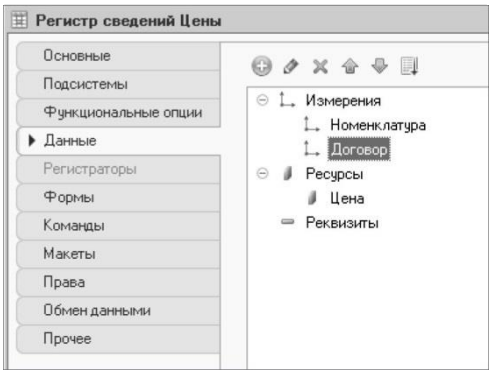
«1С:Кәсіпорын» платформасында іске асырылған мәліметтерді сақтау тетігі қолданбалы шешімде бірнеше өлшемдер бөлінісінде еркін деректерді сақтауға мүмкіндік береді. Мысалы, валюталар бөлінісінде валюта бағамын немесе номенклатура бөлінісінде кәсіпорындар бағаларын сақтауға болады. Бұдан басқа, мәліметтерді сақтаудың әр түрлі кезеңділігі берілуі мүмкін, бұл мағыналарды ғана емес, сондай-ақ олардың өзгеру тарихын сақтауға мүмкіндік береді.

*Мәліметтер тіркелімі* – бірнеше өлшемдер бөлінісінде деректерді сақтау құрылымын сипаттау үшін арналған конфигурациясының қолданбалы объектісі (10.11-сур.).

Мәліметтер тіркелімі конфигурациясы объектісінің негізінде платформа дерекқорда ақпараттық құрылымды жасайды, онда өлшемдер жинағына «тіркелген» еркін ақпарат сақталуы мүмкін.

Мәліметтер тіркелімінің жинақтау тіркелімінен маңызды ерекшелігі мәліметтер тіркелімі сандық ғана емес, сонымен қатар кез келген деректер сақтай алатындығы болып табылады.

Уақытқа байланысты пайдаланатын мәліметтер тіркелімін әдетте мәліметтердің *мерзімді* тіркелімі деп атайды. Мәліметтер тіркелімінің кезеңділігін мынадай мағыналардың бірінен анықтауға болады:




10.11-сурет. Қос өлшемі бар мәліметтер тіркелімінің құрылымы

секунд шегінде; күн шегінде; ай шегінде; тоқсан шегінде; жыл шегінде; тіркеуші шегінде.

Мәліметтердің мерзімдік тіркелімі жүйемен автоматтандырылған түрде қосылатын Мерзім қызметтік жолын үнемі қамтиды. Онда Күні түрі бар және қандай да бір мерзімге қатыстығын көрсету үшін қызмет етеді. Тіркелімге деректерді жазу кезінде платформа үнемі өзі түсетін мерзімнің басына сол жолдың мағынасын келтіреді.

Басқа тіркелімдерге секілді жүйе мәліметтер тіркелімі үшін жазбалардың бірегейлігін бақылайды. Алайда, егер басқа тіркелімдер үшін жазбаның бірегей сәйкестендіргіші тіркеуіш және жол нөмірі болса, онда мәліметтер тіркелімі үшін бастапқы мағынаны қалыптастырудың басқа қағидаты қолданылады. Жазбаны бірмәнді сәйкестендіретін жазба кілті болып бұл жағдайда тіркелім мен кезеңнің өзгерістер мағыналарының жинағы болып табылады (егер мәліметтер тіркелімі мерзімдік болса).

Мәліметтер тіркеліміне өзгерістер енгізу қолмен, сондай-ақ құжаттардың көмегімен орындалуы мүмкін. Мәліметтер тіркеліміне өзгерістер құжаттардың көмегімен енгізілген жағдайда, онда тіркеуіш – осы жазба байланысты құжат туралы ақпарат сақталған тіркелімнің әрбір жазбасына арнайы жол қосылады. Қолданбалы шешімді құру процесінде әзірлеуші осы мәліметтер тіркелімімен жазбаның қандай режімі пайдаланылатынын көрсетеді.

Мәліметтер тіркелімі әзірлеушіге ұсынатын негізгі функционалдық мүмкіндіктер мыналар болып табылады:

- жазбаларды құру, өзгерту және жою;
- берілген критерийлер бойынша берілген аралықты жазбаларлы таңдау;
- тіркеуіш бойынша жазбаларды таңдау;
- көрсетілген мерзімге және өлшем мағыналарына жазба ресурстарының мағыналарын алу;
- көрсетілген мерзімге және өлшем мағыналарына сәйкес келетін ең ерте және ең кеш ресурстардың мағыналарын алу.

## 10.6. «1С» БАСҚА ДА ҚОЛДАНБАЛЫ МЕХАНИЗМДЕРІ

**Бухгалтерлік есеп механизмдері.** Бұл механизмдер бухгалтерлік есептің қосарлы жазбасының жүйесін іске асыруға мүмкіндік береді. Олар Ресейде, сонымен қатар басқа елдерде қолданылатын есеп үлгілерін

құруға мүмкіндік береді. Бухгалтерлік есеп механизмдерімен мынадай негізгі мүмкіндіктер жүзеге асырылады:

- шоттар кодының тиянақталған немесе айнымалы дәрежелігі қолдау табатын ерікті иерархиясы бар шоттардың көп деңгейлі шоттарын жүргізу;
- бірнеше бөліктерде және деңгейлерде талдау есебін жүргізу;
- бір мезгілде бірнеше шот жоспарларының есебін жүргізу;
- бірнеше заңды тұлғалар бойынша шоғырландырылған есеп жүргізу;
- сандық, сомалық, валюталық және т.б. сияқты есепке алу түрлерінің еркін санының талдамасының жекелеген бөлімдері үшін нұсқаулар мүмкіндігі.

**Күрделі мерзімді есеп айырылысу механизмдері.** Бұл механизм еңбекақыны есептеудің әртүрлі үлгілерін іске асыруға мүмкіндік береді. Механизмнің жұмысы екі құрамдасқа негізделген.

1. Күрделі мерзімді есеп айырысу механизмдері қолданбалы шешімде пайдаланылатын есеп айырысудың әртүрлі түрлерін сипаттауға арналған құралдарды қамтиды. Мысалы, ол айлық, алименттер, айыппұл және т.б. есеп айырысу түрлері болуы мүмкін. Осы есеп айырысу түрлерін сипаттағаннан басқа, есеп айырысудың бір түрі есеп айырысудың басқа түрлеріне әсер ететін қағидаларды беру мүмкіндігі бар.
2. Күрделі мерзімді есеп айырысу механизмдері есеп айырысуды және есеп айырысудың соңғы нәтижелерін орындау үшін пайдаланылатын аралық деректерді сақтау мүмкіндігін ұсынады.

Бұл механизм көбінесе еңбекақыны есептеу үшін қолданылады, бірақ күрделі өзара байланыстары бар мерзімдік есеп айырысулардың сипаттамасын қажет ететін басқа міндеттерді шешу үшін қолданылуы мүмкін, мысалы коммуналдық қызметтердің құнын есептеу.

**Бизнес-процестер механизмі.** Бұндай механизм қолданбалы шешімдерде бизнес-процестерді сипаттауға, құруға және орындауды басқаруға мүмкіндік береді. Бұл механизмнің мақсаты жалпы мақсатқа қолжеткізуге бағытталған операциялармен байланысты тізбектерді автоматтандыру болып табылады, әдетте функционалдық рөлдер мен байланыстарды айқындайтын ұйымдық құрылымның мәнмәтінінде.

Бұл механизм қолданбалы жүйелерде бизнес-процесті басқару және оның қолданбалы шешімнің басқа функцияларымен байланысын ұйымдастыру үшін бағыттың әрбір нүктесінде орындалатын тапсырмаларды қалыптастыру үшін бизнес-процестерді және олардың рөлдік бағдарлауышын сипаттауға арналған құралдардын қамтиды.

Аталған механизм кәсіпорын қызметкерлерінің бірлескен қызметін автоматтандырудың дайын стратегиясын ұсынады. Қарапайым

бизнес-процестерді сипаттау үшін бағыт схемасын визуалды беруге және олардың тораптық нүктелерінде тармақталу шарттарын көрсету жеткілікті. Қалған барлық іс-әрекеттер жүйемен автоматты түрде орындалады. Күрделі бизнес-процестерді іске асыру кезінде әзірлеушінің күш-жігері қолданбалы шешімнің функцияларымен тығыс байланыстыру үшін талап етіледі.

**Деректерді талдау және болжау механизмі.** Бұндай механизм қолданбалы шешімдерде әдетте үлкен көлемдегі ақпараттарда жасырылған заңдылықтарды анықтау үшін құралдарды іске асыруға мүмкіндік береді.

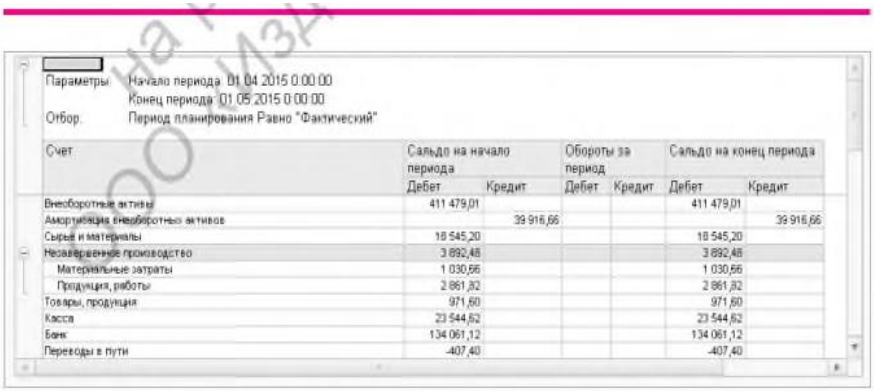
Мысалы, тауарларды сату туралы деректерді талдап, әдетте бірге сатып алынатын тауарлардың топтарын анықтауға болады және кезекті сатып алу кезінде клиентке табылған заңдылықтардан және клиент таңдаған тауарлардан қосымша тауарлар ұсыну.

Бұл механизм деректерді талдаудың бірнеше түрін орындауды сүйемелдейді, олар мыналар жалпы статистика, қауымдастықты іздеу, шешімдер ағашы, біріздікті іздеу, кластерлік талдау.

**Мәтіндік және талдау деректерінің тұсаукесер механизмдері.** «1С:Кәсіпорын» технологиялық платформасы интерактивті өзара іс-қимылды үлкен немесе кіші дәрежеде қамтамасыз ете отырып, пайдаланушыға ыңғайлы түрде қорытынды ақпаратты ұсынуға мүмкіндік беретін бірқатар объектілерді қамтиды.

Кез келген ақпараттың тұсаукесерінің немесе оны басып шығарудың бірден бір қуатты құралы кестелік құжат болып табылады (10.12-сур.). Ол баспа құжаттарының тиімді дайындығын ғана қамтамасыз етпейді, сондай-ақ оларды пайдаланушыға ыңғайлы түрде экраннан көрсетеді. Кестелік құжаттың негізгі мүмкіндіктерін атап шығуға болады:

- шрифт түрі және көлемін, мәтіннің және фонның түсін, жақтаудың, суреттің түрі мен түсін және т.б. қоса алғанда есепті ресімдеу;



Параметры: Начало периода: 01.04.2015 0:00:00
Конец периода: 01.05.2015 0:00:00
Отбор: Период планирования Равно "Фактический"

| Счет | Сальдо на начало периода | | Обороты за период | | Сальдо на конец периода | |
|---|---|---|---|---|---|---|
| | Дебет | Кредит | Дебет | Кредит | Дебет | Кредит |
| Внеоборотные активы | 411 479,01 | | | | 411 479,01 | |
| Амортизация внеоборотных активов | | 39 916,66 | | | | 39 916,66 |
| Сырье и материалы | 18 545,20 | | | | 18 545,20 | |
| Незавершенное производство | 3 892,48 | | | | 3 892,48 | |
| Материальные затраты | 1 030,66 | | | | 1 030,66 | |
| Продукция, работы | 2 861,82 | | | | 2 861,82 | |
| Товары, продукция | 971,60 | | | | 971,60 | |
| Касса | 23 544,62 | | | | 23 544,62 | |
| Банк | 134 061,12 | | | | 134 061,12 | |
| Переводы в пути | -407,40 | | | | -407,40 | |

10.12-сурет. Кестелік құжат

- топтастыруды пайдалану (тік және көлденең), олардың көмегімен аралық қорытындыларды көрсетуге болады;
- есептің жолында немесе ұяшығында шерту кезінде егжей-тегжей есеп қалыптасқан немесе дерекқорлардың объектісі ашылғандан жағдайдағы мағынаны ашу механизмін сүйемелдеу;
- ұяшықта немесе құжат саласында орналасқан деректер туралы қосымша ақпаратты қамтитын ескертпелерді пайдалану;
- қоса енгізілген тақырыптары бар жазық кесте түрінде көпөлшемді деректерді көрсетуге мүмкіндік беретін жиынтық кестелерді сүйемелдеу;
- экономикалық ақпаратты графикалық түрде көрнекі етіп ұсыну үшін әртүрлі диаграммаларды сүйемелдеу;
- кестелік құжатты әртүрлі форматтарда және т.б. сақтау мүмкіндігі.

Форматталған құжатты пайдалана отырып, суреттер мен гиперсілтемелері бар шрифті, түсі әртүрлі ресімделген мәтіннің үзділерін құруға болады. Негізінен ол мәтін әртүрлі шрифпен, түспен ресімделуі тиіс гиперсілтемелер мен суреттерді қамтыған жағдайдағы нысандарда редакциялау үшін пайдаланылады.

Қорытынды деректерді «1С:Кәсіпорын» олардың географиялық жағдайы бөлінісінде ұсыну үшін арнайы объект – *географиялық схема* пайдаланылады. Ол мысалы, елдің әртүрлі аймақтарында сол немесе өзге де тауарлардың сату көлемін көрсететін есептерді құруға мүмкіндік береді. Сондай-ақ географиялық схема сол немесе басқа географиялық деректерді көрсету үшін пайдаланылуы мүмкін, мысалы, офиске өту немесе көлік құралының қозғалыс бағытының схемалары.

Графикалық схема қолданбалы шешімді графикалық ресімдеу үшін әртүрлі ұйымдастырушылық, құрылымдық және өзге де схемаларды құруға мүмкіндік береді. Ол сол немесе өзге де ұйымдастырушылық процестерді, блок-схемаларды және басқаларды ұсынуды қажет ететін нысандар мен есептерді ресімдеу үшін арналған. Бұдан басқа, графикалық схема қолданбалы шешімді ресімдеудің бөлігі болып табылатын жеке құжат ретінде пайдалануы мүмкін. Графикалық схемалар арқылы сол немесе өзге де алгоритмдердің құрамын, сол немесе өзге де процестердің құрылымын, ұйымдастырушылық схемаларды және басқаларды түсіндіретін суреттемені құруға ыңғайлы.

Деректерді графикалық ұсыну үшін пайдаланылатын тағы бір нысан диаграмма болып табылады.

Платформамен сүйемелденетін диаграммалардың арнайы түрлерінің бірі Ганта диаграммасы болып табылады. Оның құрамында уақыт белдігінде орналасқан аралықтардың жинағы бар және ресурстар (сериялар) объектілерінің (нүктелерінің) пайдалануын көрсетеді. Диаграмманың бұл түрі нақты сандық мағыналарымен емес, уақытша аралықтардың жинағымен ұсынылатын міндеттердің орындалу барысын көру, ресурстарды, жұмыс уақыты графигін және басқа деректерді жоспарлау үшін кең пайдаланылады.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. Анықтамалық түріндегі объектілер не үшін пайдаланылады?
2. Әрбір Анықтамалық үшін қандай нысандардың түрлері берілуі мүмкін?
3. Анықтамалықтың стандартты деректемелерін атаңыз.
4. Құжат түріндегі объектілер не үшін пайдаланылады?
5. Әрбір Құжат үшін нысандардың қандай түрлері берілуі мүмкін?
6. Құжаттың стандартты деректемелерін атаңыз.
7. Тұрақты шама және Аударымдар не үшін пайдаланылады?
8. Құжаттар журналы не үшін арналған?
9. Есептер не үшін арналған?
10. Деректерді біріктіру жүйесі не үшін арналған?
11. Деректерді біріктіру жүйесі қалай бапталады?
12. Есептің Өңдеуден негізгі айырмашылығы неде?
13. «1С» тіркелімдер дегеніміз не? Олар не үшін арналған?
14. Тіркелімдерде ақпаратты сақтау бірлігі не болып табылады?
15. Құжатты жүргізу нені білдіреді?
16. Уақытты жедел белгілеу дегеніміз нені білдіреді?
17. Жинақтау тіркелімі конфигурация объектісі не үшін арналған?
18. Жинақтау тіркелімінің айрықша ерекшелігі не болып табылады?
19. Тіркеуіш дегеніміз не?
20. Қалдық тіркелімдерінің және айналым тіркелімдерінің арасындағы негізгі айырмашылықтар қандай?
21. Мәліметтер тіркелімі нені білдіреді?
22. Мәліметтер тіркелімінің жинақтау тіркелімінен түбегейлі айырмашылығы қандай?
23. Мәліметтердің мерзімдік тіркелімінде қандай қызметтік жол үнемі болады?
24. Бухгалтерлік есеп механизмдерімен іске асырылатын негізгі мүмкіндіктерді атап шығыңыз.

# 11 тарау

# «1С:КӘСІПОРЫН» ЖҮЙЕСІНІҢ АРХИТЕКТУРАСЫ

## 11.1. КЛИЕНТТІК-СЕРВЕРЛІК АРХИТЕКТУРА ЖӘНЕ КЛИЕНТТІК ҚОСЫМШАЛАР

*Клиенттік қосымша* — бұл пайдаланушының компьютерінде жұмыс істейтін және «1С:Кәсіпорын» жүйесінің пайдаланушымен интерактивтік өзара іс-қимылын қамтамасыз ететін бағдарлама. «1С:Кәсіпорын» жүйесінде клиенттік қосымшалардың үш түрі бар: «Жуан» клиент, «Жіңішке» клиент және Web-клиент. Қосымшаларды әзірлеуге және ақпараттық базаларды әкімшілдендіруге мүмкіндік беретін клиенттік қосымшалардың бірі конфигуратор болып табылады.

*«Жуан» клиент* қосымшаны әзірлеу, әкімшілдендіру, орындау жоспарында «1С:Кәсіпорын» толық мүмкіндіктерін жүзеге асыруға мүмкіндік береді. «Жуан» клиент кіріктірілген тілмен ұсынылатын барлық функционалдықты орындай алатындықтан, ол пайдаланушының компьютеріндегі аппараттық ресурстардың айтарлықтай санын қажет етеді және файлдік қолжетімділік немесе жергілікті желі арқылы дерекқормен немесе «1С:Кәсіпорын» серверлерінің кластерімен өзара іс-қимыл жасай алады. «Жуан» клиент Интернет арқылы ақпараттық базалармен жұмысты қолдамайды, ол пайдаланушының компьютеріне алдын ала орнатуды қажет етеді және дистрибутивтің әсерлі көлемін қамтиды.

«Жіңішке» клиент қолданбалы шешімдерді әзірлеуге және әкімшілдендіруге мүмкіндік бермейді, бірақ ол Интернет арқылы ақпараттық базалармен жұмыс істей алады.

Ол сондай-ақ пайдаланушының компьютеріне алдын ала орнатуды қажет етеді, бірақ «Жуан» клиентке қарағанда ондағы дистрибутивтің көлемі айтарлықтай біршама аз. «Жіңішке» клиент деп кіріктірілген тілдің шектеулі жинағын орындағандықтан аталады.

Мәселен, «Жіңішке» клиентте деректердің барлық қолданбалы түрлері қол жетімді емес. Оның орнына «Жіңішке» клиент жадыдағы деректерді бейнелеу және өзгерту үшін арналған кіріктірілген тіл түрлерінің шектеулі жинағын пайдаланады.

Дерекқорлармен, объектілік деректермен, сұрау салуларды орындаумен байланысты барлық жұмыстар сервер жағында жүзеге асырылады. «Жіңішке» клиент көрсету үшін дайындалған дайын деректерді алады.

«Жіңішке» клиент «1С:Кәсіпорын» интерфейсімен Интернет арқылы жұмыс істеуге мүмкіндік береді. Ол үшін «1С:Кәсіпорын 8» жұмыс істеу үшін бапталған Web-сервер қолданылады. «Жіңішке» клиент Web-сервермен HTTP немесе HTTPS хаттамасы бойынша өзара әрекеттеседі. Web-сервер өз кезегінде «1С:Кәсіпорын 8» файлдік немесе жұмыстың клиент-серверлік нұсқасында өзара әрекеттеседі.

*Web-клиент* «Жуан» клиентке және «Жіңішке» клиентке қарағанда компьютерге қандай да бір алдын ала орнатуды қажет етпейді. Ол компьютердің операциялық жүйесінің ортасында емес, Web-браузердің ортасында орындалады. Web-клиентте қолданбалы шешімдерді әзірлеу мүмкін емес. Web-клиентте орындалатын файл жоқ. Кез келген пайдаланушыға өзінің браузерін іске қосып, ақпараттық база жарияланған Web-сервердің адресін енгізу жеткілікті. Web-клиент DHTML және HTTPRequest технологияларын қолданады. Web-клиент жұмыс істеу кезінде конфигурацияда әзірленген клиенттік модульдер «1С» кіріктірілген тілден автоматтандырылған түрде құрастырылады және тікелей Web-клиенттің жағында орындалады.

Клиенттік қосымшаға («Жуан», «Жіңішке», Web-клиент) тәуелсіз қолданбалы шешімнің барлық зерттемесі толығымен «1С:Кәсіпорын» конфигураторында жүргізіледі, серверлік және клиенттік кодтар «1С:Кәсіпорын» кіріктірілген тілінде жазылады.

Осы клиенттік қосымшалардың жиынтық түріндегі мүмкіндіктерін 11.1-кестесінде көрсетілгендей ұсынуға болады.

11.1-кесте. Клиенттік қосымшалардың мүмкіндіктері

| Іс-қимылы | «Жуан» клиент | «Жіңішке» клиент | Web-клиент | Конфигуратор |
|---|---|---|---|---|
| Жергілікті желідегі жұмыс | Иә | Иә | Иә | Иә |
| Интернет арқылы жұмыс | Жоқ | Иә | Иә | Жоқ |
| Алдын ала орнату | Иә | Иә | Жоқ | Иә |

## 11.2. ЖҮЙЕНІҢ ЖҰМЫС ІСТЕУ НҰСҚАЛАРЫ. ИНТЕРНЕТ АРҚЫЛЫ ҚОСУ

«1С:Кәсіпорын» екі нұсқада жұмыс істей алады: файлдық және клиент-серверлік.

Екі нұсқада да барлық қолданбалы шешімдер толығымен бірдей жұмыс істейді, бұл қолданыстағы қолданбалы шешімді өзгертпестен жұмыстың бір нұсқасын таңдауға мүмкіндік береді.

*Жұмыстың файлдық нұсқасы* бір пайдаланушының дербес жұмысына немесе жергілікті желіде пайдаланушылардың аз санының жұмысына арналған. Бұл нұсқада ақпараттық базаның барлық деректері (конфигурация, деректерқор, әкімшілік ақпарат) бір файлда – деректердің файлдық базасында орналасады (11.1-сурет). Осы дерекқормен жұмысты файлдық ДҚБЖ іске асырады. Деректерді сақтаудың осы форматы «1С:Кәсіпорын» қолданбалы шешімдері үшін арнайы әзірленген.

«Жіңішке» клиент іске қосылған компьютермен жұмыс істеу кезінде жұмыстың файлдық нұсқасында «Жіңішке» клиент іске қосылған мамандандырылған орта ұйымдастырылады. Осы мамандандырылған орта шеңберінде мыналар орындалады:

- жүйенің жұмысы үшін қажетті серверлік компоненттерді жүктеу;
- қолданбалы конфигурацияны жүктеу;
- ақпараттық базасы бар жүйенің қалыпты жұмысын ұйымдастыруға қажетті басқа да әрекеттер.

«Жіңішке» клиент тарапынан осы орта сервер рөлін атқарады. Операциялық жүйе тарапынан осы мамандандырылған орта жеке процеске бөлінбеген және Жіңішке клиент процесі шеңберінде орындалады.

Жұмыстың мұндай нұсқасы орнатуды жеңілдетеді және қолданбалы шешімді пайдалануды қамтамасыз етеді. Бұл ретте, ақпараттық базамен жұмыс істеу үшін қосымша бағдарламалық құралдар қажет емес, операциялық жүйенің және «1С:Кәсіпорын» болуы жеткілікті.

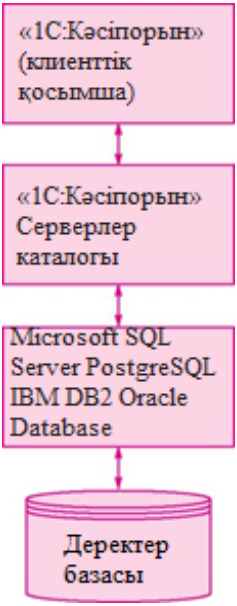

11.1-сурет Жұмыстың файлдық нұсқасы

Жұмыстың клиент-серверлік нұсқасы жұмыс топтарында немесе кәсіпорын ауқымында пайдалануға арналған. Ол «клиент-сервер» үш деңгейлі архитектура негізінде жүзеге асырылған (11 2-сурет).

Пайдаланушыда жұмыс істейтін клиенттік қосымша («Жуан» клиент, «Жіңішке» клиент немесе Web-клиент) «1С:Кәсіпорын 8» серверлерінің кластерімен өзара іс-қимылды жүзеге асырады, ал кластер қажетінше дерекқор серверіне(Microsoft SQL Server, PostgreSQL, IBM DB2, Oracle Database) сұрау сала алады.

1С:Кәсіпорын» үш деңгейлі архитектура келесі тәсілмен әзірленген: клиент үшін «1С:Кәсіпорынмен» байланысты бірде-бір файлдық ресурстарға қолжетімділік қажет етілмейді.

«1С:Кәсіпорын» клиенттік қосымшасын пайдаланушыға дерекқор серверінің дерекқорына қолжетімділік талап етілмейді. Ақпараттық базаға қолжетімділік үшін клиенттік қосымша «1С:Кәсіпорын» серверлерінің кластерімен өзара іс-қимылды жүзеге асырады.




11.2-сурет. Жұмыстың клиент-серверлік нұсқасы

*Кластер* — ортақ қосымшаларды орындау үшін бірлесіп жұмыс істейтін және пайдаланушы үшін бірыңғай жүйе болып табылатын бірнеше есептеу жүйелерінің жиынтығы.

Кластер клиенттік қосымша мен дерекқор серверінің арасында аралық бағдарламалық қабатты түзеді. Бұл ретте, «1С:Кәсіпорын» серверінің кластері және дерекқор сервері бір компьютерде, сондай-ақ әртүрлі компьютерлерде орналасуы мүмкін. Бұл әкімшіге қажет кезде әртүрлі компьютерлер арасындағы жүктемені бөлуге мүмкіндік береді. Барлық қосымшаларды әртүрлі компьютерлерде орналастырған жөн.

«1С:Кәсіпорын» серверлерінің кластерін пайдалану онда деректерді өңдеу бойынша айтарлықтай көлемді операцияларды орындауды шоғырландыруға мүмкіндік береді. Мысалы, аса күрделі сұрау салуларды орындаудың өзінде пайдаланушы жұмыс істеп тұрған бағдарлама өзіне қажетті ғана іріктемені алатын болады, ал барлық аралық өңдеу серверде орындалады. Әдетте, клиенттік машиналар паркін жаңартуға қарағанда, серверлер кластерінің қуатын арттырған әлдеқайда оңай.

Үш деңгейлі архитектураны пайдаланудың басқа маңызды аспектісі әкімшілендіру және ақпараттық базаға пайдаланушылардың қолжетімділігін реттеу болып табылады. Бұл нұсқада пайдаланушы конфигурацияның немесе дерекқордың табиғи орналасуын білмеуі тиіс. Барлық қолжетімділік «1С:Кәсіпорын» серверлерінің кластері арқылы жүзеге асырылады.

Пайдаланушы белгілі бір ақпараттық базаға жүгінген кезде тек қана кластердің атауын және ақпараттық базаның атауын көрсетуге міндетті, ал жүйе, тиісінше пайдаланушы аты мен құпия сөзін сұратады.

«1С:Кәсіпорын» ақпаратты тиімді іріктеу үшін дерекқорды басқару жүйесінің мүмкіндіктерін пайдаланады:

а) сұрау салу механизмі есептеулерді орындау және есептерді қалыптастыру үшін ДҚБЖ-ны барынша пайдалануға бағытталған;

б) үлкен динамикалық тізімдерді қарау дерекқорға жасалатын көптеген жүгінулерді орындамастан қамтамасыз етіледі; бұл ретте, пайдаланушыға тиімді іздеу, сондай-ақ іріктеуді және сұрыптауды баптау мүмкіндіктері ұсынылады.

Клиент-серверлік нұсқаны өрістету және оны әкімшілендіру айтарлықтай оңай түрде орындалады. Мысалы, дерекқорды құру конфигураторды іске қосу процесі кезінде тікелей жүзеге асырылады (файлдық нұсқа үшін де).

«1С:Кәсіпорын» серверлері кластерінің жұмыс процестері бір компьютерде (қарапайым түрде), сонымен қатар Windows және Linux сияқты түрлі операциялық жүйелердің басқаруымен жұмыс істейтін бірнеше компьютерлерде жұмыс істей алады. Жүйе архитектурасы барлық функционалдылықты орындауды серверлер кластеріне барынша ауыстыруға және клиентті барынша «жеңілдетуге» бағытталған. Серверлер кластерінде қолданбалы объектілердің барлық жұмысы орындалады, нысандарды (ақпараттық базадан объектілерді оқу және нысан деректерін толтыру, элементтерді орналастыру, өзгерткеннен кейін нысан деректерін жазу) және командалық интерфейсті бейнелеуге дайындық жүзеге асырылады, есептер қалыптастырылады. Клиентте серверлердің кластерінде әзірленген ақпаратты бейнелеу ғана орындалады, пайдаланушымен өзара іс-қимылы және қажетті әрекеттерді орындау үшін серверлік әдістерді шақыру жүзеге асырылады.

Кластердің болуы көптеген пайдаланушылардың ірі ақпараттық базалармен толассыз, бұзылуға төзімді, бәсекелестікке жарамды жұмысын қамтамасыз ету мүмкіндігін береді.

Интернет арқылы қосылу пайдаланушылардың ақпараттық базалармен қашықтықтан online-жұмысын қамтамасыз ету мүмкіндігін береді. Бұл «Жіңішке» клиентті және Web-клиентті пайдалану арқылы мүмкін болады. Олар арнайы бапталған Web-серверге қосылады, ол өз кезегінде олардың «1С:Кәсіпорын» серверлерінің кластерімен немесе файлдық ақпараттық базамен өзара іс-қимылды жүзеге асырады.

Қолданбалы шешімдер олармен Интернет арқылы жұмыс істеу үшін ешбір пысықтауды қажет етпейді.

«Жіңішке» және Web-клиент те пайдаланушының компьютерінде «1С:Кәсіпорын» интерфейсінің жұмыс істеуін өздігінен қамтамасыз етеді. Осы клиенттік қосымшаларды пайдаланудағы айырмашылық келесіде:

а) «Жіңішке» клиент пайдаланушының компьютерінде алдын ала орнатуды қажет етеді, Web-клиент — қажет етпейді;

б) «Жіңішке» клиент толық функционалдылықты қамтамасыз етеді, Web- клиент кейбір маңызды платформаларды қолдамайды;

в) «Жіңішке» клиент Windows немесе Linux басқаруында жұмыс істейді, ал Web-клиент интернет-браузердің басқаруында жұмыс істейді; сондықтан Web-клиентті пайдаланушы интернет-браузердің басқаруымен жұмыс істейді; сондықтан пайдаланушы мыналарда қосымша жұмыс істей алады:

- Mac OS X операциялық жүйесі бар (Safari браузерінде) компьютерде;
- Apple iOS операциялық жүйесі бар (Safari мобильдік браузерде) iPad интернет-планшетінде;
- «Жіңішке» және Web-клиент жұмысының эргономикасында кейбір ерекшеліктер бар.

Файлдық және клиент-серверлік нұсқада тікелей кластер, сондай-ақ Web-сервер арқылы да жұмыс істеуге болады. Web-сервер арқылы қосылған кезде «Жіңішке» клиент пен Web-клиент HTTP немесе HTTPS хаттамасын қолданады. Кластерге тікелей қосылған жағдайда «Жуан» клиент және «Жіңішке» клиент TCP/IP хаттамасын сүйемелдейді (11.3-сурет).

Web-сервер арқылы файлдық дерекқормен жұмыс істеу «Жіңішке» клиент немесе Web-клиенттің көмегімен жүзеге асырылады (11.4-сурет). Бұл жағдайда Web-серверді кеңейті модулі әрбір ақпараттық база үшін Web-серверде ұқсас серверлік ортаны құрады.


11.3-сурет. Интернет арқылы қосылу: жұмыстың клиент-серверлік нұсқасы


11.4-сурет. Интернет арқылы қосылу: жұмыстың файлдық нұсқасы

Web-клиент жағдайында файлдық дерекқорға қосылу Web-сервер арқылы жүзеге асырылады және деректермен тікелей жұмысты клиенттік қосымша емес, файлдық ДҚБЖ қамтитын Web-серверді кеңейту модулі орындайды. Жүйе элементтерінің файлдық дерекқорымен өзара іс-қимылы «1С» фирмасы әзірлеген деректермен алмасудың меншікті хаттамасы бойынша жүзеге асырылады. Web-сервердің адрестік кеңістігіне файлдық дерекқормен жұмыс жасау үшін компонент және ақпараттық базаның деректері де тікелей жүктеледі. Бұл ретте, Web-серверге жүктеме біршама артады, ал бір ақпараттық базаны пайдаланушылардың қатарлас жұмыс істеуі мүмкін болмайды. Осы себеп бойынша жұмыстың мұндай нұсқасы тестілік болып табылады, мысалы, ақпараттық база Web-сервер арқылы Web-клиентпен қалай жұмыс істейтінін көру үшін пайдаланылады. Жұмыс нұсқасы ретінде мұндай нұсқаны аса үлкен емес жұмыс топтары үшін айрықша жағдайларда пайдалануға болады.

Клиенттер Интернетке шығудың әртүрлі тәсілдерін пайдалана алады. Бұл бөлінген желілер немесе жергілікті желі бойынша жоғары жылдамдықта қосылу болуы мүмкін. Сондай-ақ, төменгі жылдамдықтағы қосылыстар болуы мүмкін, мысалы, мобильді GPRS-қосылу. Ақпараттық базамен жұмыс жасау байланыстың төмен жылдамдықты арналары арқылы іске асырылатын осындай жағдайлар үшін «Жіңішке» клиент пен Web-клиентте арнайы іске қосу режимі – қосылыстың төменгі жылдамдықтығы режимі болады. Нәтижесінде, платформа байланыстың төменгі жылдамдықтағы арналарында да пайдаланушылар жұмысына қолайлы жылдамдықты қамтамасыз етеді. Web-сервер ретінде Apache немесе IIS пайдаланылады.

Apache — еркін Web-сервер. Apache – GNU/Linux, BSD, Mac OS, Microsoft Windows, Novell NetWare, BeOS операциялық жүйелерді қолдайтын, кросс-платформалық бағдарламалық жасақтама болып табылады. Apache жүйеде Web-серверлердің бірі ретінде пайдаланылады, олардың көмегімен пайдаланушылар қашықтықтан Интернет арқылы ақпараттық базамен жұмыс істей алады.

Internet Information Services (IIS) —Интернеттің бірнеше қызметтері үшін *Microsoft* компаниясы серверлерінің жиынтығы. IIS – Windows NT жататын операциялық жүйелермен таралады. IIS жүйеде Web-серверлерінің бірі ретінде пайдаланылады, оның көмегімен пайдаланушылар қашықтықтан Интернет арқылы ақпараттық базамен жұмыс істей алады.

## 11.3. СЕРВЕРЛЕРДІҢ КЛАСТЕРІ

«1С:Кәсіпорын» серверлерінің кластері логикалық түсінік болып табылады және бір немесе бірнеше жұмыс процесінің, бір немесе бірнеше компьютерде жұмыс істейтін сервер агенттерінің және кластер менеджерлерінің және осы процестер жұмыс істейтін ақпараттық базалар тізімінің жиынтығын білдіреді.

Қарапайым түрде «1С:Кәсіпорын» серверлерінің кластері бір комьютерде жұмыс істей алады.

Серверлер кластерінің жұмысында келесі элементтер пайдаланылады:

а) ragent, rmngr, rphost серверлері кластерінің процестері;

б) деректер қоймасы: кластерлер тізімі, кластер тізілімі. *Сервердің агенті* деп аталатын ragent процесі кластер құрамында компьютердің жұмыс істеуін қамтамасыз етеді. Тиісінше, сервер агенті іске қосылатын компьютер *жұмыс сервері* деп аталады. Сервер агенті функцияларының бірі осы жұмыс серверінде орналасқан кластерлердің тізімін жүргізу болып табылады.

Сервер агенті және кластерлердің тізімі серверлер кластерінің құрамына кірмейді, онда орналасқан серверлер мен кластерлердің жұмысын ғана қамтамасыз етеді.

Серверлер кластерінің құрамына мынадай элементтер кіреді:

- rmngr бір немесе бірнеше процесі;
- кластер тізілімі;
- rphost бір немесе бірнеше процесі.

Rmngr процесі *кластер менеджері* деп аталады. Бұл процесс барлық кластердің жұмыс істеуін басқарады. Кластер менеджері жұмыс істейтін және кластер тізілімі орналасқан жұмыс сервері

*кластердің орталық сервері* деп аталады. Үнсіз келісу бойынша серверлер кластеріне *кластердің бас менеджері* деп аталатын кластердің бір менеджері кіреді.

Масштабталуды арттыру үшін кластер әкімшісінде кластердің құрамында бір немесе бірнеше кластердің қосымша менеджерлерін анықтау мүмкіндігі болады. Кластерлердің қосымша менеджерлері жұмыс серверлерінде іске қосылуы мүмкін. Бұл ретте, кластердің функционалдылығы кластердің бас және қосымша менеджерлері арасында бөлінуі мүмкін.

Кластердің бас менеджерінің функцияларының бірі – төмендегі ақпаратты қамтитын кластердің тізілімін жүргізу болып табылады:

- осы кластерде тіркелген ақпараттық базалардың тізімі;
- кластерге енетін жұмыс серверлерінің тізімі;
- кластерге енетін жұмыс процестерінің тізімі;
- кластердің қосымша менеджерлерінің тізімі;
- кластер сервисінің және кластердің қосымша менеджерінің сәйкес келуі;
- әкімшілерінің тізімі.

Rphost процессі *жұмыс процесі* деп аталады. Жұмыс процесі тікелей клиенттік қосымшаларға қызмет көрсетеді, дерекқор серверімен өзара іс-қимылды жүзеге асырады және онда, айтап айтқанда, конфигурацияның серверлік модулінің процедуралары орындалуы мүмкін.

## 11.4. СЕРВЕР МЕН КЛИЕНТТЕ ФУНКЦИОНАЛДЫЛЫҒЫН ОРЫНДАУ

Қолданбалы объектілермен барлық жұмыс, дерекқорды оқу және жазу тек қана серверде орындалады. Нысандар мен командалық интерфейстің функционалдылығы да серверде іске асырылған. Бұл қолданбалы шешімнің өнімділігін және сенімділігін арттыруға мүмкіндік береді. Серверде аталған нысандарды дайындау, элементтерді орналастыру, өзгерткеннен кейін нысандардың деректерін толтыру жүзеге асырылады. Командалық интерфейс пен есептер де серверде толықтай қалыптастырылады және клиентте көрінеді. Бұл ретте, платформаның механизмдері клиенттің компьютеріне берілетін деректер көлемін азайтуға бағытталған.

Серверде: дерекқорға сұрау салуларды жолдау, деректерді жазу, құжаттарды өткізу, әртүрлі есептеулер, өңдеулерді орындау, есептерді қалыптастыру, нысандарды көрсетуге әзірлеу іске асырылады.

Клиентте: нысандарды алу және ашу, нысандарды көрсету, пайдаланушымен интерактивті өзара іс-қимыл жасау (ескерту, сұрақтар және басқалары.), тез жауап беруді талап ететін нысандардағы шағын есептеулер (мысалы, жалпы соманы алу үшін бағаны көлемге көбейту), жергілікті файлдармен жұмыс, сауда жабдықтарымен жұмыс және тағы басқалары орындалады. Клиентте деректерді енгізу және енгізілген деректер үшін серверді шақыру және басқа да қажетті әрекеттер орындалады.

Клиентте дерекқормен тікелей жұмыс істеуге, сондай-ақ тікелей қолданбалы объектілермен жұмыс істеуге рұқсат берілмейді, мысалы, кіріктірілген тілдің АнықтамалықОбъект.<аты> сияқты түрлері қолжетімсіз. Клиентте сұрақтарды пайдалану мүмкін емес. Клиенттік кодта деректермен іс-әрекетті шақыру қажеттігі туындаған кезде серверлік процедураларды шақыру қажет, олар өз кезегінде деректерге сұрау салады. Модульдердегі клиенттік процедуралар серверлерден анық бөлінеді және оларда кіріктірілген тілдің объективті үлгісінің шектеулі құрамы пайдаланылады.

## 11.5. ДЕРЕКТЕРМЕН ЖҰМЫС ІСТЕУ

«1С:Кәсіпорын» дерекқоры үлгісінің әзірлеушілер әмбебап жүйелерде кездестіретін дерекқорды басқару жүйелерінің классикалық үлгілерінен ажыратын бірқатар ерекшеліктері бар (мысалы, реляциялық кестелерге негізделген). Негізгі ерекшелік – «1С:Кәсіпорын» әзірлеушісі дерекқорға тікелей сұрау салмайды. Ол «1С:Кәсіпорын» платформасымен тікелей жұмыс істейді. Бұл ретте, ол:

- конфигураторда деректер құрылымын сиапттай алады;
- кіріктірілген тіл объектілерінің көмегімен деректерді басқара алады;
- сұрау салу тілін пайдалана отырып, деректерге сұрау салуларды әзірлей алады.

«1С:Кәсіпорын» платформасы сұрақтарды орындау операцияларын, деректер құрылымын сипаттауды және деректерді тиісті командаларға трансляциялап, оларды басқаруды қамтамасыз етеді. Жұмыстың клиент-серверлік нұсқасы жағдайында бұл деректерді басқару жүйесінің командалары немесе файлдық нұсқа үшін дербес ДҚБЖ командалары болуы мүмкін.

«1С:Кәсіпорын» деректердің екі санатына қолдау көрсетеді: объектілік және объектілік емес. Осы деректермен жұмыс істеу тәсілдері ерекшеленеді. Бұдан әрі бұл санаттар анағұрлым толық қарастырылады.

*Объектілік деректерге* анықтамалардың, құжаттардың, мінездеме түрлері жоспарларының, есептер жоспарларының, есептесу түрлері жоспарларының, бизнес-процестердің, міндеттердің, алмасу жоспарларының деректері жатады. Объективті емес деректерге мәліметтер тізілімінің, жинақтау тізілімінің, есептеулер, қайта есептеулер тізілімдерінің, бухгалтерия тізілімдерінің деректері және жүйелік деректер, константалар жатады.

Жүйе тұрғысынан объективті деректер жеке объектілерден тұрады. Объект өріс мәндерінің жиынтығы болып табылады. Объектінің әрқайсысы ішкі бірегей сәйкестендіргішке ие, осыған орай, дерекқорда сақталатын кейбір мәндер жиынтығына бірыңғай тұтастық ретінде (объектіге) жүгіне алады. Мысалы, Қызметкерлер анықтамалығының элементі – әлдебір жеке тұлға, онда оны сипаттайтын мәндер жиынтығы бар: аты, тегі, әкесінің аты, лауазымы, төлқұжат деректері және т.б. Оның, мысалы, тегі немесе төлқұжат деректері өзгеруі мүмкін, бірақ, бұл ретте, жүйе тұрғысынан ол сол қызметкер (жеке тұлға) – объекті болып қала береді. Жүйеден қандай да объектіні жою қолданбалы шешім тұрғысынан дерекқор жай-күйінің өзгеруіне әкеледі. Қызмет анықтамалығы деректемелерінің сол мәндерінен жаңа элементін құрсақ та, басқа бірегей сәйкестендіргіші бар басқа объектіні аламыз.

«1С:Кәсіпорындағы» объектілік деректерге мынадау конфигурациялау объектілерінің мәндері жатады:

- анықтамалық;
- құжат;
- сипаттама түрлерінің жоспары;
- есептеулер жоспары;
- есептесу түрлерінің жоспары;
- алмасу жоспары;
- бизнес-процесс;
- міндет.

Конфигурацияның әрбір объектісі үшін жүйе өзара байланысқан кестелер жиынтығын құрады, мұнда осы объектінің деректері сақталады. Кестелердің саны мен құрамы метадеректердің әрбір объектісі үшін әртүрлі.

Объектілік деректерді сақтау құрылымы негізгі кестені және онымен байланысты басқа кестелерді (объектінің әрбір кестелік бөлігіне бір кестеден) құру дегенді білдіреді. Мысалы, деректемелердің кейбір жиынтығы және «Материалдар» кестелік бөлікті қамтитын «Кіріс жүккұжаты» құжаты үшін дерекқорда екі кесте құрылады. Құжат кестесі құжаттың әрбір деректемесі үшін өрістерді қамтиды, ал

| Сілтеме | Күні | Нөмірі | Қойма |
|---|---|---|---|
| … | … | … | … |
| ljt546dt]'tdyj546456684 | 05.06.2014 12:00:00.000 | A00002563 | Негізгі |
| … | … | … | … |

| Сілтеме | Жолдың нөмірі | Материалы | Саны | Бағасы | Сомасы |
|---|---|---|---|---|---|
| … | … | … | … | … | … |
| Ijt546dtjtdyj54645668 | 1 | Цемент | 100 | 300 | 30000 |
| ljt546dtjtdyj54645668 | 2 | Шиыршық тас | 35 | 20 | 700 |
| … | … | … | … | … | … |

11.5-сурет. *Кіріс жөнелтпе* құжатының кестелері

Кестелік бөлік үшін құжаттың кестелік бөлігінің барлық деректемелері үшін өрістерді қамтитын жеке кесте құрылады (11.5-сурет).

Осы кестелердің айрықша ерекшелігі – олардың әрқайсысы құжаттардың әрбіріне сәйкес келетін ішкі сәйкестендіргіш сақталатын Сілтеме өрісін қамтиды. Осылайша, құжат объектісі негізгі кесте жазбаларының жиынтығын және осы құжатқа жататын кесте бөліктерінің өрістерін білдіреді.

Кіріктірілген тілдегі объектілік деректермен жұмыс жасау үшін екі негізгі түр бар: сілтеме және объекті.

*Сілтемелік түрдің* мәні (АнықтамалықСілтеме. .<аты>, ҚұжатСілтеме, <аты> және т.б.) дерекқор объектілерін сәйкестендіретіні анық. Мұндай мағына дерекқордың кестелер Сілтемесі өрісінде сақталатын ішкі сәйкестендіргішті білдіреді. Сілтеме өрісі – қызметтік өрістердің бірі.

Сілтеменің мағынасы дерекқордың басқа кестелерінің өрісінде сақталуы мүмкін. Мысалы, Қойма өрісі Қоймалар анықтамалығы элементтерінің сілтемесін сақтайды.

Сілтемелік түрдің мағынасын өзара салыстыруға болады. Кіріктірме тілдегі конфигурацияның әрбір объектісі үшін сілтеменің өз түрі құрылады. Осылайша, мысалы, Қоймалар анықтамалығына сілтеме ешқашан Өнім берушілер анықтамалығының сілтемесіне тең болмайды, себебі бұл мағыналардың түрлері әртүрлі. Бұл ретте, Қоймалар анықтамалығына екі сілтемеге өзара тең болуы мүмкін және егер де дерекқордың белгілі бір объектісінің сілтемелері болған жағдайда ғана орындалатын болады.

Дерекқордың белгілі бір объектісіне көрсететін сілтемелер алынған жолдарына қарамастан, өзара тең болады.

Мысалы, Номенклатура анықтамалығының менеджері арқылы алынған «000005» кодындағы Номенклатураға сілтеме Номенклатура анықтамалығының іріктемесінен алынған осы номенклатураға тең болады:

```
   Ауыст.Сілтеме = Анықтамалықтар.Номенклатура. Код
бойынша іздеу ("000005");
   НоменклатураІріктемелері = Анықтамалықтар.
Номенклатура. Таңдау();
   ӘзіршеНоменклатураІріктемесі.Келесі() Цикл
   Ауыст. Сілтеме бар ма = НоменклатураІріктемесі.
Сілтеме Сонда
   ...
```

Сілтемелер түрінде қалыпты жағдайдағы мағынасы – бос сілтеме бар. Бос сілтеме – дерекқордағы бірде бір объекті сәйкес келмейтін сілтемелердің мағынасы.

Сілтеме түрі конфигурацияның әрбір объектісі үшін құрылатындықтан, онда әртүрлі анықтамалықтардың бос сілтемелері ешуақытта өзара тең болмайды.

*Сілтеме* дерекқор объектілерінің қасиеттеріне жүгінуге мүмкіндік береді. Бұл ретте, дерекқордан ақпаратты оқу орындалады, себебі сілтеменің өзі бұл деректерді қамтымайды.

*Объекті* негізгі кестенің және осы объектіге жататын кестелік бөліктер өрісі жазбаларының жиынтығын білдіреді. Объекті түрі әрекеттегі объектілерді редакторлар және жою үшін жаңа объектілерді құру кезінде пайдаланылады. Бұдан басқа, объектінің түрі объекті нысанындағы объектінің барлық деректерін көрсету және редакторлар үшін пайдаланылады. Бұл түрдің мәндерін сілтемелердің мәндеріндей өзара салыстыруға болады. Алайда, сілтемелерге қарағанда бұл түрдің мағыналары екеуі де бағдарламалық объектінің бір данасы болып табылған кезде ғана олар өзара тең болады.

Мысалы, әртүрлі тұрақсыз шамаларда Номенклатура анықтамалығының белгілі бір элементіне сәйкес келетін объектінің даналарын алсақ, онда оларда дерекқордың бір объектісі есептеліп және осы объектінің барлық деректері сәйкес келгеніне қарамастан, осы тұрақсыз шамалардың мағыналары тең болмайды.

Бұдан әрі келтірілген мысалда 1Объект және 2Объекті тұрақсыз шамаларында келтірілген мағыналар тең болмайды:

```
   1Объект      =      Анықтамалықтар.Номенклатура.Код
бойынша іздеу ("001").ОбъектініАлу();
   2Объект      =      Анықтамалықтар.Номенклатура.Код
бойынша іздеу- ("001").ОбъектініАлу();
   // 1Объект<> 2Объект
   ...
```

Объекті түрінің мағынасы келесі тәсілдермен алынуы мүмкін:

- тиісті әдісті қолданумен объектінің менеджері арқылы (мысалы, анықтамалықтар үшін бұл әдіс ЭлементҚұру (), құжаттар үшін – `ҚұжатҚұру ()` және т.б.). Бұл ретте, бағдарламалық объектінің жаңа данасы құрылады;
- `ОбъектініАлу ()` әдісін орындау арқылы сілтемеден. Бұл жағдайда бағдарламалық объектінің данасы құрылады және дерекқордан (немесе кэштен) деректерді оқу орындалады. Бұл ретте, объектінің барлық деректемелерінің мағыналары және оның кестелік бөліктерінің барлық мағыналарының деректемелері есептеледі;
- ОбъектініАлу () ұқсас атаудағы әдісті орындау арқылы іріктемеден. Бұл жағдайда да бағдарламалық жасақтаманың нұсқасы жасалады, алайда дерекқордан оқу орындалмайды, себебі іріктеме дерекқор объектінің барлық деректерін есептейді және объектінің құрылған данасының деректері іріктеменің бағдарламалық жасақтамасынан тікелей толтырылады.

Бұл түр қолдайтын негізгі оқиғалардың арасынан келесілерді атап өтуге болады.

`Жазбадан Бұрын` — бұл оқиға объектіні дерекқорға жазбастан бұрын туындайды, жазба транзакциясы басталғаннан кейін, бірақ деректерді тікелей жазудан бұрын. Осы оқиғаны өңдеуіште деректерді жазуды орындау қажеттігін (немесе мүмкіндігін) талдауға және қайсыбір шарттар орындалмаса, одан бас тартуға болады.

Жазба Кезінде — бұл оқиға дерекқорға жазылғаннан кейін, бірақ жазба транзакциясы аяқталғанға дейін туындайды. Осы оқиғаның өңдеуішінде объект жазылған жағдайда міндетті түрде орындалуы тиіс іс-қимылды орындауға болады. Сондай-ақ, мұнда әзірлеуші, егер, мысалы, осы деректерді базаға жазу нәтижесінде қайсыбір шарттар бұзылса, деректерді жазудан бас тарта алады.

Көшіру Кезінде — жаңа объекті әрекеттегі деректер объектісін көшіру (интерактивті немесе бағдарламалық) арқылы құрылған болса, онда бұл оқиға деректердің жаңа объектісінде туындайды.

`Толтыруды Өңдеу` — бұл оқиға жаңа объектіні интерактивті құру, негіз бойынша объектіні енгізу кезінде (интерактивті немесе бағдарламалық), сондай-ақ объектінің Толтыру () әдісін орындаған кезде туындайды. Осы оқиғаның өңдеушісінде негіз-объектінің түріне байланысты жаңа объектінің деректемелерін бастапқы толтыруды орындауға мүмкіндік беретін біршама алгоритм қарастырылады.

`Толтыруды Тексеруді Өңдеу` — бұл оқиға объектінің деректерін жазбастан бұрын, жазбаның транзакциясы басталғанға дейін шақырылады. Осы оқиғаның өңдеушісінде әзірлеуші объектінің деректемелірн толтыруды тексерудің меншікті алгоритмдерін жүзеге асыра алады, тексерілетін деректемелердің ауқымына платформа оларды тексеруі үшін қосымша деректемелерді қоса алады немесе стандартты тексеруден бас тартқан тексерілетін деректемелердің ауқымын тазалай алады. Сондай-ақ, әзірлеуші, мысалы тексерудің қайсыбір шарттары орындалмаған жағдайда объектіні жазудан бас тарта алады.

`Жоймастан Бұрын` — бұл оқиға дерекқордан объектіні тікелей жоймастан бұрын жою транзакциясында туындайды. Осы оқиғаның өңдеушісінде объектіні жоймастан бұрын қайдай да бір іс-қимылды орындауды қарастыруға болады, сондай-ақ қажет кезде қандай да бір шарттар орындалмаған жағдайда объектінің жойылуынан бас тартуға болады.

*Объектілік емес деректер* объектілік деректерге қарағанда өз өрістерінің мағыналарымен толық сипатталады және оларда ішкі бірегей сәйкестендіргіштер болмайды. Кейбір жазбаны жойып және тура осындай өрістердің мағыналары бар жаңа жазбаны құра отырып, біз сол жазбаны аламыз, яғни жазбаны жойғанға дейінгі дерекқордың сол қалпына қол жеткіземіз. Бұл объектілік емес деректерді объектілік деректерден түбегейлі ажыратады: объектіні екі рет құруға болмайды, ол әрекет ету факторымен құнды. Бұдан басқа, жазба өрістерінің мағыналарын өзгертіп, біз басқа жазбаны аламыз, бұл ретте, объекті өрістерінің мағыналарын өзгерту жаңа объектіні тудыруды туындатпайды.

«1С:Кәсіпорындағы» объектілік емес деректерге конфигурацияның келесі объектілерінің деректері жатады: деректер тізілімі, жинақтау тізілімі, бухгалтерия тізілімі, есептесу тізілімі және т.б.

«1С:Кәсіпорын» тұрғысынан объектілік емес деректер кестелік деректердің біршама жазбалар жиынтығын білдіреді. Осы жазбалардың әрқайсысы өз өрістерінің мағыналарымен сипатталады.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. «1С:Кәсіпорын» клиенттік қосымшасы нені білдіреді?
2. «Жуан» клиенттің, «Жіңішке» клиент пен Web-клиенттің жұмысындағы мүмкіндіктер мен шектеулерді атап өтіңіз.
3. «1С» жүйе жұмысының файлдық және клиент-серверлік нұсқалары неге арналған?
4. «1С» жүйесінде файлдық жұмыс нұсқасының мүмкіндіктері мен шектеулерін атап өтіңіз.
5. «1С» жүйесінің жұмысындағы клиет-серверлік нұсқасының мүмкіндіктері мен шектеулерін атап өтіңіз.
6. Кластер дегеніміз не?
7. Кластерді пайдалану қандай мүмкіндіктерді береді?
8. Интернет арқылы «1С» жүйесіндегі жұмыс қандай режимдерде мүмкін?
9. Файлдық нұсқада тікелей кластермен жұмыс істеу мүмкін бе?
10. Web-сервер арқылы кластермен файлдық нұсқада жұмыс істеуге бола ма?
11. Серверде қандай функциялар орындалады?
12. Клиентте қандай функциялар орындалады?
13. Объектілік деректерге қандай деректер жатады?
14. Объектілік деректерді сақтау құрылымы нені білдіреді?
15. Сілтеме түрінің мағынасы не үшін қызмет етеді?
16. Объектілік түрдің мағыналары не үшін қажет?
17. Объект түрінің мағынасын қандай тәсілдермен алуға болады?
18. Объект түрінің көмегімен қандай негізгі оқиғалар сүйемелденеді?
19. Қандай деректер объектілік емес деректерге жатады?
20. «1С:Кәсіпорын» тұрғысынан объектілік емес деректер нені білдіреді?

# 12 тарау

# «1С» «КІРІКТІРІЛГЕН ТІЛІН ПАЙДАЛАНУ

## 12.1. ЖҰМЫСТЫҢ НЕГІЗГІ ТӘСІЛДЕРІ

Кіріктірілген тіл «1С:Кәсіпорын 8» технологиялық платформасының бөлігі болып табылады және әзірлеушіге қолданбалы шешім функционалдауының меншікті алгоритмдерін сипаттауға мүмкіндік береді.

Бағдарламалаудың басқа да объектілік-бағдарланған тілдеріндегідей «1С:Кәсіпорын» жүйесінің тілінде объектінің қасиеттеріне сұрау салу нүкте арқылы іске асырылады, мысалы:

```
Объект.ШартБойыншаТөлемСомасы = ШартДеректерінАлу
(Объект.Шарт,"ШартБойыншаТөлемСомасы")
```

Нүкте арқылы жасалатын негізгі сұрау салудың басқа [] (төртбұрышты жақша) операторының көмегі арқылы қасиеттің атауымен өріс бойынша объектінің қасиеттеріне сұару салу механизмі қарастырылған. Мұндай құрылым қасиеттің атауын көрсетумен нүкте арқылы жасалатындай, туралы солай объектінің қасиеттеріне жүгінуге мүмкіндік береді. Мысалы:

```
Анық = Анықтамалықтар.Номенклатура.
КодБойыншаІздеу (ІзделетінКод);
// Қасиеттің атауымен өріс бойынша
анықтамалықтың атауына жүгіну
А Анық.["Атауы"];
// Қасиеттің атауы бойынша анықтамалықтың
атауына жүгіну
А = Анық. Атауы;
// Қасиетке жасалатын осы екі жүгіну де мүлдем
әртүрлі мәнде.
```

Тілде объектілерге және басқа бағдарламалық модульдердің сырттағы нысандарына жүгіну кезінде объектілер мен нысандардың өздерінің қасиеттері мен әдістеріне жүгінгендей, осы модульдердің тұрақсыз шамаларға, процедуралар мен функцияларға жүгіну мүмкіндігі бар.

Экспорт кілт сөзімен жарияланған тұрақсыз шамаларға, процедуралар мен функцияларға жүгінуге болады. Нысандар үшін нысандар деректемелеріне қосымша жүгіну мүмкін.

Үлгі:

```
// Журналдан құжаттарды басып шығару процедурасын пайдалану үлгісі
Процедура Басып шығару (Элемент)
// Курсор орнатылған ағымдағы құжатты аламыз.
МәтіндікҚұжат = НысанЭлементтері.ТізімЖурналы.АғымдағыЖол;
// Ағымдағы құжаттың негізгі нысанын аламыз.
МәтіндікҚұжатНысаны = МәтіндікҚұжат.НысандАлу();
// Құжат нысанының үлгісінде
// орналасқан басып шығару процедурасын шақырамыз.
МәтіндікҚұжатНысаны. Басып шығару();
ПроцедураныңАяқталуы
```

«1С:Кәсіпорын» жүйесінің тілдегі бірқатар объектілер *мәндер топтамасын* білдіреді (ауқымдар, қолданбалы объектілердің кестелік бөліктері және т.с.с.). Топтамалардың басым бөлігінде `Сан ()`, `Индекс()`, `Қосу()`, `Жою()` және т.б. сияқты ұқсас әдістер мен қасиеттердің жиынтығы бар. Әдетте, топтама қасиеттері ретінде оның элементтері жүреді. Топтамалар үшін Әрқайсысы үшін – -дан – Цикл құрылымы арқылы топтама элементтерін айналып өту қолжетімді. Топтамалардың басым бөлігі үшін оператордың көмегімен `[<Аргумент>]` (төртбұрышты жақшалар) топтама элементтеріне жүгіну қолжетімді. Әдетте, аргумент ретінде топтама элементінің индексі беріледі. Топтама элементтерін интекстеу 0-ден басталады. Бұл соңғы элементтің индексі минус 1 топтамасындағы элементтердің санына тең екендігін білдіреді.

«1С:Кәсіпорын» жүйесінің тілінде белгілері (нөмірлері) бар бірқатар объектілер, жеке бөліктер бар. Мұндай объектілерге, мысалы, жол жатады, оның символдарында жолда нөмірі бар немесе кестелік құжат, оның жолдары мен бағандарында нөмір бар және т.с.с. нөмірлер 1-ден басталады. Топтама элементтеріне жүгінген кезде Индекс түсінігі қолданылады. Топтама элементтерін индекстеу 0-ден басталады.

«1С:Кәсіпорын» жүйесіндегі кіріктірілетін тілде жүйелік тізбелердің түсінігі бар. Олар алдын ала анықталған мағыналардың біршама шектелген жиынтығын анықтауға арналған.

Жүйелік тізбелерге қолжетімділік оның атауының жаһандық мәнмәтінінің қасиеттері ретінде жүзеге асырылады. Нақты мағыналар жүйелік тізбелердің атынан нүкте арқылы көрсетіледі. Әдетте, жүйелік тізбелер жүйелік әдістердің немесе объекті қасиеттерінің өлшемдерінің мағыналарын беру үшін, сондай-ақ әдістердің қайтарылатын мағыналары ретінде қолданылады.

## 12.2. ДЕРЕКТЕРДІҢ УАҚЫТША ЖИЫНДАРЫН САҚТАУҒА АРНАЛҒАН ОБЪЕКТІЛЕР

Конфигурация объектілерінің негізгі тобын бизнес-логиканың (анықтамалықтар, құжаттар және т.б.) функционалдау алгоритмдерін сипаттауға мүмкіндік беретін қолданбалы объектілер құрайды. Бірақ пайдаланушы жұмысының сеансы ішінде деректердің уақытша жиынтығын сақтауға арналған объектілердің маңыздылығы кем емес. Әдетте, олар ақпаратты қосымша жинау, топтау, талдау және өңдеу қызметін атқарады.

Ауқым еркін түрдегі мағыналардың нөмірленген топтамасын білдіреді. Ауқым элементіне оның индексі бойынша жүгінуге болады. Ауқым элементтері ретінде басқа ауқымдар жүруі мүмкін. Бұл көп шамалы ауқымдарды құруға мүмкіндік береді.

Құрылым «кілт – мән» жұбынан құралған аты аталған топтаманы білдіреді. Кілт жолды ғана, мән – еркін түрде бола алады. Құрылым элементіне оның кілтінің мәні, яғни аты бойынша жүгінуге болады. Әдетте, әрқайсысында бірегей аты бар мағыналардың шағын көлемін сақтау үшін пайдаланылады.

Құрылым ретінде сәйкес келу де «кілт – мағына» жұбының топтамасын білдіреді. Алайда, құрылымға қарағанда, кілт кез келген түрде болуы мүмкін.

Әдетте, мағыналар тізімі интерфейстік міндеттерді шешу үшін қолданылады. Мағыналардың динамикалық жиынтықтарын құруға және олармен әрекет етуге (элементтерді қосу, редакторлау, жою, сұрыптау) мүмкіндік береді. Ол кез келген түрдегі мағыналарды қамти алады, бұдан басқа бір тізімде сақталатын мағыналардың түрлері әртүрлі бола алады. Мысалы, мағыналар тізімі күрделі алгоритм бойынша қалыптасқан болжамды құжаттар тізімінен нақты құжатты таңдау үшін қолданылуы мүмкін.

Мағыналар кестесі мағыналардың динамикалық жиынтықтарын құруға және олармен әрекет етуге мүмкіндік береді.

Ол кез келген түрдегі мағыналармен толтырылуы мүмкін және бір кестеде сақталатын мағыналар түрлері әртүрлі болуы мүмкін. Мағыналар кестесін пайдаланудың бір үлгісі ретінде күрделі алгоритм бойынша іріктелген анықтама элементтерінің тізімі нысанында ұсынылған ұйым бола алады.

`Мағыналар ағашы` мағыналар кестесіне ұқсайтын динамикалық қалыптасатын кез келген түрдегі мағыналар жиынтығын білдіреді. Мағыналар кестесіне қарағанда, мағыналар ағашының жолдарында бағынышты жолдар жиынтығы болуы мүмкін, бағынышты әрбір жолда өз кезегінде бағынышты жолдардың жиынтығы және басқалары болуы мүмкін. Бұл ретте, мағыналарды іздеу, сұрыптау, нәтижені алу иерархияның ағымдағы деңгейі бойынша немесе барлық бағыныштыларды қоса іске асырылуы мүмкін.

`COMSafeArray` – COM-нан `SAFEARRAY` көп шамалы массивтің объектілік қабығын білдіреді. COM-объектілер арасында деректермен алмасу үшін `SAFEARRAY` құруға және пайдалануға мүмкіндік береді.

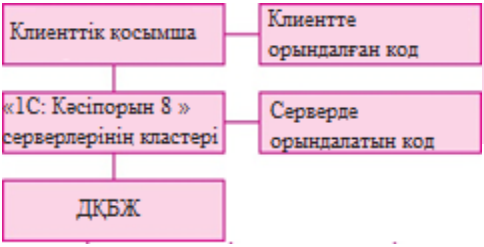
БекітілгенМассив — ТиянақталғанМассив –осы түрдегі объектілердің инициализациясы кезінде жүйе немесе конструктордың көмегімен әзірлеуші толтыратын өзгертілмейтін массив.

Кіріктірілетін тілде мәтіндерді құру және өзгерту үшін әзірлеуші модульдерді құру, редакторлау және синтаксикалық тексерудің ыңғайлы құралдарына ие мәтін мен модульдің редакторын пайдалана алады.

## 12.3. МОДУЛЬДЕРДІ ОРЫНДАУДЫҢ КЛИЕНТ-СЕРВЕРЛІК МӘНМӘТІНІ

Модульдерді орындау мәнмәтіні модуль орындалатын бағдарламалық ортаны анықтайды. «1С:Кәсіпорын» жұмысының клиент-серверлік нұсқасында клиент мәнмәтіні және сервер мәнмәтіні болады (12.1-сурет). Бұл ретте, «1С:Кәсіпорын» жұмысының клиент-серверлік нұсқасындағы барлық бағдарламалық компоненттер «1С:Кәсіпорын» жұмысының клиент-серверлік нұсқасындағы барлық бағдарламалық компоненттер бар компьютерде, сондай-ақ әртүрлі компьютерлерде орналаса алады.

Клиенттің мәнмәтінінде (клиентте) және сервер мәнмәтінінде (серверде) кіріктірілетін тілдің әртүрлі қасиеттері, әдістері және объектілері қолжетімді. Деректерге қолжетімділікпен байланысты (олардың оқылуы және жазбасы) барлық әрекеттерді серверде ғана жүзеге асыруға болады, ал пайдаланушыға осы деректерді көрсету және басқа да интерактивтік әрекеттер




12.1-сурет. Жұмыстың клиент-серверлік нұсқасында «1С» бағдарламалық құрауыштар

клиентте ғана мүмкін болады. Сондықтан да модульдердегі клиенттік процедуралар серверлерден анық түрде ажыратылады. Серверде пайдаланылатын кіріктірілетін тілдің объектілік үлгілерінің барлығы клиентте қолжетімді бола алмайды.

Әдетте, «1С» сервері және ДҚЖБ сервері бір жергілікті желінің шегінде, ал клиенттік қосымшасы бар клиенттік компьютер осы желіде, сондай-ақ ақпаратты берудің әртүрлі жылдамдығы бар байланыстың түрлі арналарын пайдаланумен Интернет арқылы серверге қосылуы мүмкін. Жұмыс процессі барысында клиенттік қосымшаның «1С:Кәсіпорны» белгілі бір әрекеттерді орындау үшін үнемі серверді шақыртады. Сервер осы әрекеттерді орындайды және басқару мен нәтижені клиентке кері қайтарады.

Платформа механизмдері байланыстың төменгі жылдамдықтағы арналарында да жүйе жұмысының қолайлы жылдамдығын қамтамасыз ету үшін клиент-серверлік өзара іс-қимылды оңтайландырады. Сондықтан да, клиентте орындалатын процедуралардан серверлік процедураларды шақыру арқылы клиент-серверлік өзара іс-қимылды ұйымдастыратын әзірлеуші осындай оңтайландыруға да ұмтылуы тиіс.

## 12.4. ЖАЛПЫ МОДУЛЬ

Жалпы модульдер конфигурацияның басқа кез келген модулінен шақырылуы мүмкін олардың ішіндегі мәтіндер функциялары мен процедураларын орналастыруға арналған. Конфигурацияда жалпы модульдердің ерікті саны анықталуы мүмкін немесе бірде-біреуі анықталмайды. Жалпы модульдің мәнмәтіні ең жалпы болып табылатын модульдің жаһандық мәнмәтінімен және жергілікті мәнмәтінмен, яғни жалпы модульдің ішінде анықталған процедуралар мен функциялармен қалыптасады.

Жалпы модуль тікелей жүйемен орындалмайтындықтан,

онда ауыспалыларды сипаттау тарауы және негізгі бағдарлама тарауы жоқ. Жалпы модуль процедуралар мен функциялардың анықтамаларын ғана қамти алады. Егер жалпы модульдің процедуралары немесе функциялары экспортталушы ретінде анықталса, онда олар жаһандық мәнмәтіннің бөлігі болады және қолданбалы шешімнің басқа модульдеріне қолжетімді болады.

Егер жалпы модульде Жаһандық қасиеті орнатылған болса, онда жалпы модульдің экспортталатын әдістері жаһандық мәнмәтіннің бөлігі болып табылады және оларға жаһандық мәнмәтіннің басқа функциялары, әдістері мен қасиеттеріне жүгінгендей, кіріктірілген тілден атаулары бойынша жүгінуге болады. Егер де Жаһандық қасиеті жалғанға тең болса (жаһандық емес модуль), онда оның экспортталатын процедураларына немесе функцияларына жүгінген кезде, оның атын модульдің атынан нүкте арқылы көрсету қажет (мысалы, Алмастыру.БағаныАлу()). Соңғы тәсілді қолданған жөн, себебі жаһандық емес жалпы модульдер жаһандық сияқты жүйені іске қосу кезінде емес, оларға жүгіну бойынша құрастырылады.

Жалпы модуль конфигурацияның қандай болса да объектісіне таңылмай, барлық қолданбалы шешімге жататындықтан, экспортталатын процедуралар мен функциялардың аттары әртүрлі жалпы модульдерде әртүрлі болуы тиіс. Кері жағдайда синтактсистік қате беріледі, себебі жаһандық мәнмәтін қайталанатын аттарды қамтитын болады.

Клиент және Сервер қасиеттерінің көмегімен басқарылатын қосымшада, сондай-ақ көрсетілген препроцессорге қосымшаның серверінде немесе клиентте жалпы модульдердің әртүрлі процедураларын және функцияларын орындауды ұйымдастыруға болады.

Жалпы модульдердің қасиеттерінде қалып бойынша Сервер жалаушасы орнатылады. Бұл жалпы модульдің барлық процедуралары мен функциялары серверде ғана қолжетімді болатынын білдіреді. Осындай жағдайда серверді Шақыру қасиетін орнатса, онда жалпы модульдің процедуралары мен функциялары клиентте қолжетімді болады.

Тиісінше, егер жалпы модульде Клиент (басқарылатын қосымша) қасиеті және (немесе) Сыртқы қосылыстар орнатылса, онда осы жалпы модульдің экспортталатын процедуралары мен функциялары клиенте (басқарылатын қосымша режимінде «Жіңішке» клиентте, Web-клиентте, «Жуан» клиентте) және (немесе) сыртқы қосылыста қолжетімді болады.

Егер жаһандық емес жалпы модульде Клиент (басқарылатын қосымша) қасиеті орнатылса, онда осы модульдің мәнмәтінінде басқарылатын қосымша модулінің экспортталатын ауыспалылары, процедуралары мен функциялары қолжетімді болады. Егер жаһандық емес жалпы модульде

сыртқы Қосылыс қасиеті орнатылған болса, онда осы модульдің мәнмәтінінде сыртқы қосылыс модулінің экспортталатын ауыспалылары, процедуралары мен функциялары қолжетімді болады.

Жаһандық емес жалпы модульдерде қайтарылатын мағыналардың Қайта пайдалану қасиеті де қолжетімді. Бұл қасиетті орнату жаһандық емес жалпы модульдердің экспортталатын функцияларын (процедураларды емес, тек қана функцияларды) орындауды жылдамдатады. Бұл кештеу механизмін пайдалану есебінен жүзеге асады, себебі функцияға берілген өлшемдердің мағыналары және бұл ретте, оған қайтарылған нәтиже жадыда сақталады және бұдан әрі пайдалану үшін сақталады.

## 12.5. ОБЪЕКТІНІҢ МОДУЛІ

Конфигурацияның әрбір қолданбалы объектісінің өз модулі болады. Бұл модуль конфигурация объектісінің деректерін түрлендіруге мүмкіндік беретін кіріктірілетін тілдің объектісін құру кезінде орындалады. Кіріктірілетін тілдің тиісті объектісі, мысалы, жаңа объектіні енгізу, көшіру, қолданыстағы объектінің деректерін алу кезінде және т.б. құрылады.

Объект модулінің мәнмәтіні:

- жаһандық мәнмәтін, соның ішінде серверде компиляцияланған жалпы модульдердің экспортталатын функцияларымен және процедураларымен (бұл модульдер үшін Сервер қасиеті орнатылған);
- мәнмәтін модульмен кеңейтілетін кіріктірілітен тіл объектісінің қасиеттерімен және әдістерімен;
- модуль тиесілі конфигурация объектісінің деректемелерімен;
- объекті модулінің жергілікті мәнмәтінімен қалыптасады.

Объекті модулінің ауыспалылары, процедуралары немесе функциялары экспортталушы ретінде анықталса, онда олар кіріктірілген тілдің тиісті объектісінің қасиеттері мен әдістері ретінде қолжетімді.

Мысалы, Номенклатура анықтамалығының модулінде экспортталатын процедура анықталды делік:

```
ТауардыТексеру Процедурасы() Экспорт
//...
// ...
ПроцедураныңСоңы
```

Онда осы процедураның келесі шақырылымы мүмкін, мысалы, басқа модульдерден (мысалы, сыртқы өңдеу модулінен):

```
Анықтамалық = Анықтамалықтар.Тауарлар.
КодБойыншаІздеу("01"). ОбъектініАлу();
Анықтамалық.ТауардыТексеру();
```

Ауыспалылардың және негізгі бағдарламаның сипаттамасынан басқа объектінің модулі конфигурацияның осы объектісімен байланысты оқиғалардың өңдеушісі – процедуралардың сипаттамасын қамти алады. Осы оқиғалардың құрамы түрлі объектілер үшін әртүрлі, бірақ барлық объектілер үшін шақырылатын үш оқиға бар:

ТолтырудыТексерудіӨңдеу;
ЖазбаданБұрын,
ЖазбаКезінде.

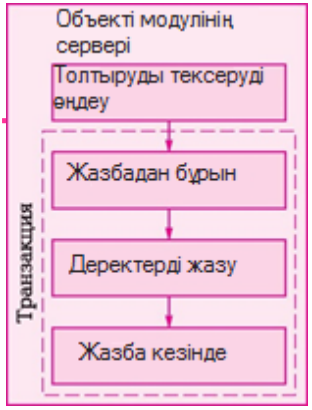
Оларды шақыру реттілігі 12.2-суретте келтірілген.

ТолтырудыТексерудіӨңдеу оқиғасы объектінің деректерін жазудан бұрын, жазбаның транзакциясына дейін шақырылады. Әзірлеуші осы оқиғаның өңдеушісінде объекті деректемелерін толтыруды тексерудің меншікті алгоритмдерін жүзеге асыра алады. Ақиқат мәніндегі Бас тарту өлшемін орнатып, әзірлеуші егер, мысалы, тексерудің қайсыбір шарттары орындалмаса объектінің жазбасынан бас тарта алады.

ЖазбаданБұрын оқиғасы деректерді жазбастан бұрын, жазба транзакциясы басталғаннан кейін, бірақ деректерді дерекқорға жазу тікелей басталмастан бұрын шақырылады. Осы оқиғаның өңдеушісінде әзірлеуші егер, мысалы, талап етілетін шарттар орындалмаса объектінің жазбасынан бас тарта алады.

ЖазбаКезіндегі оқиға дерекқорға деректерді жазу орындалғаннан кейін, бірақ жазба транзакциясы аяқталғанға дейін шақырылады. Осы оқиғаның өңдеушісінде негізгі деректерден бөлек өзгертілмейтін объектінің деректерімен байланысты деректердің әрекеттері орындалады. Сондай-ақ, мұнда, егер, мысалы, осы деректерді базаға жазу нәтижесінде қайсыбір шарттар бұзылса, әзірлеуші деректерді жазудан бас тарта алады.

ӨткізудіӨңдеу оқиғасы құжатпен байланысты маңызды оқиғалардың бірі болып табылады. Бұл оқиға




12.2-сурет.Объекті модулінің оқиғаларын шақырудың реттілігі

құжатты өткізу кезінде туындайды. Осы оқиғаның өңдеушісінің негізгі мәні – құжат бойынша қозғалыстарды тудыру. Өңдеуші процедурасында деректермен әртүрлі операцияларды орындауды есептеу қалпына әсер етеді. Әзірлеуші дәл осы процедураға құжатты өткізу кезінде орындалатын деректерді түрлендірудің меншікті алгоритмдерін орналастыруға міндетті.

## 12.6. ОБЪЕКТ МЕНЕДЖЕРІНІҢ МОДУЛІ

Әрбір қолданбалы объекті үшін конфигурация объектісі ретінде осы объектіні басқаруға арналған менеджер болады. Менеджердің көмегімен объектілерді құруға, нысандармен және макеттермен жұмыс істеуге болады. Менеджер модулі кіріктірілген тілде процедуралар мен функцияларды жазу есебінен жүйе ұсынатын менеджерлердің функционалдығын кеңейтуге мүмкіндік береді. Іс жүзінде бұл дерекқор объектісінің нақты данасына емес, конфигурация объектісінің өзіне тән конфигурация объектісіне (мысалы, анықтамалыққа) арналған әдістерді сипаттауға мүмкіндік береді.

Менеджер модулінің мәнмітіні:

- жаһандық мәнмәтінмен, соның ішінде серверде компиляцияланған жалпы модульдердің экспортталатын функциялары мен процедураларымен (бұл модульдер үшін Сервер қасиеті орнатылған);
- менеджер модулінің жергілікті мәнмәтінімен қалыптасады.

Менеджер модулі тек қана серверде орындалады және онда модульдің ауыспалылары және денелері бола алмайды. Егер менеджер модулінің функциялары немесе процедуралары экспортталушы деп жарияланса, онда оларға қолжетімділікті объектінің менеджері арқылы алуға болады.

Мысалы, Номенклатура анықтамалығы менеджерінің модулінде экспортталатын функция анықталған делік:

```
ҚызметтердіАлу Функциясы()
Экспорт
 // ...
Функцияның Аяқталуы
```

Онда осы функцияның келесі шақырылым мүмкін, мысалы, сыртқы өңдеу модулінен:

```
Дебиторлар = Анықтамалықтар.Контрагенттер.ҚызметтердіАлу( ) ;
```

Объекті менеджерлерінің модульдерінде ТаңдауДеректерінАлудыӨңдеу оқиғасының өңдеушісі орналасады. Бұл оқиға жол бойынша енгізген кезде, мәтінді автоіріктеу және тез теру кезінде тізімді стандартты қалыптастырудан бұрын серверде туындайды.

## 12.7. ПРЕПРОЦЕССОРДЫҢ НҰСҚАУЛЫҚТАРЫ ЖӘНЕ КОМПИЛЯЦИЯЛАУ ДИРЕКТИВАЛАРЫ

Әртүрлі модульдердің процедураларын және функцияларын қолданудың рұқсатын көрсету үшін препроцессор нұсқаулықтары және компиляция директивтері қолданылады. Препроцессор нұсқаулықтарының компиляция директивтерінен негізгі айырмашылығы препроцессор құрылымы кодтың қайда копиляцияланатынын сипаттаса, компляция директивтері кодты қайда компиляциялау қажет екенін көрсетеді. Бұдан басқа, препроцессор нұсқаулықтары процедуралар мен функциялардың ішінде, ал компиляция директивтері процедураны, функцияны немесе ауыспалыны (нысан модулі үшін) сипаттаудан бұрын орналаса алады.

Басқаша айтқанда, келесі түрдегі препроцессор нұсқаулықтарының кодында пайдалану:

```
#Егер Сервер Сонда
...
#СоңыЕгер
```

шарт ішінде орналасқан код сервер тарапында ғана компиляциялануы (және орындалуы) мүмкін екенін көрсетеді.

Алайда, компиляция директивінің кодында

```
&Клиентте
Процедура МенiңПроцедурам()
...
ПроцедураныңАяқталуы
```

түрін қолдану, процедура компиляция директивасынан кейін клиент тарапындағы серверде ғана компиляциялануы (және орындалуы) тиіс екенн көрсетеді.

Бұдан ары директив копиляцияның тізімі және олардың қысқаша сипаттамасы келтірілген. Жалпы, нысанның үлгісінде компиляцияның әртүрлі төрт директивасы пайдаланылуы мүмкін:

```
&Клиентте,
&Серверде,
& СервердеМәнмәтінсіз,
&КлиенттеСервердеМәнмәтінсіз.
```

`&Колиенттегі` директива процедура немесе функция клиенттік қосымшаның мәнмәтінінде орындалатынын көрсетеді. Бұл процедурада нысанның барлық мәнмәтіні қолжетімді болады: нысанның деректемелері, элементтері және өлшемдері. Бұл директива нысанның клиенттік оқиғаларының барлық өңдеушілері, сондай-ақ нысанның жергілікті

командаларын сипаттайтын процедуралар үшін қолданылады. Бұл процедураларды *нысанның клиенттік процедуралары* деп жиі атайды.

*&Сервердегі* директива алдыңғы директиваға ұқсас, бірақ кодты орындау серверлік мәнмәтінінде орындалуымен ерекшеленеді. Бұл директива нысанның серверлік оқиғаларының барлық өңдеушілері үшін пайдаланылады. Сондай-ақ, әзірлеушілер бұл директиваны кодтың орындалуын серверге беру мақсатында меншікті процедуралары үшін жиі қолданады. Мұндай процедураларды *нысанның серверлік процедуралары* деп жиі атайды. Ал клиенттік процедурадан мұндай процедураны шақыру *мәнмәтіндің серверлік шақырылым* деп аталады.

*&СервердеМәнмәтінсіз* директивасы да кодтың серверлік мәнмәтінде орындалатынын анықтайды, бұл ретте нысан мәнмәтіні (деректемелері, элементтері, өлшемдері) қолжетімсіз болады. Әзірлеушілер бұл директиваны кодтың орындалуын серверге беру мақсатында меншікті процедуралары үшін қолданады. Мұндай процедураларды нысанның серверлік процедуралары деп атайды. Мұндай процедураны клиенттік процедурадан шақыруды *мәнмәтінсіз серверлік шақыру* деп атайды.

*&КлиенттеСервердеМәнмәтінсіз* директивасы процедура немесе функция клиенттің мәнмәтінінде де, сервердің мәнмәтінінде де орындала алатынын анықтайды. Бұл директива сирек қолданылады. Әдетте, ол нысанды құру кезінде, сондай-ақ клиентте оның өміршеңдігі процессінде бірдей әрекеттерді орындау қажет болған жағдайларда талап етіледі. Ондай жағдайда біреуі серверде, ал екіншісі клиентте орындалатын екі бірдей процедураның орнына *&Клиентте – СервердеМәнмәтінсіз* директивасымен бір процедура құрылады.

Егер компиляцияның бірде-бір директивасы көрсетілмесе, онда қалып бойынша *&Серверде* директивасы қолданылады.

Нысан үлгісінде компиляция директивалары процедуралар мен функциялардың ғана емес, сондай-ақ ауыспалылардың анықтамаларынан бұрын көрсетілуі тиіс.

Егер ауыспалылықты сипаттаудан бұрын компиляцияның *&Клиентте* директивасы қолданылса, онда осындай ауыспалылық нысанның клиенттік бөлігінде ғана нысанды құру кезінен оның жабылуына дейін өмір сүреді. Нысанның серверлік процедураларынан ол қолжетімсіз болады.

Егер ауыспалылықтың сипаттамасынан бұрын компиляцияның *&Серверде* директивасы қолданылса, онда осындай ауыспалылық қандай да бір серверлік процедураны немесе функцияны шақыру немесе орындау мерзімінде ғана өмір сүреді. Кодты орындау клиентке қайтарылғаннан кейін серверде орындалған модуль осы ауыспалылықпен бірге жойылады.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. «1С» тілінде объектілерге сұрау салу қалай жүзеге асырылады?
2. Мағыналар кестесі және мағыналар тізімінің айрмашылығы неде, түсіндіріңіз.
3. Жалпы модульдің *Жаһандық* қасиеті нені білдіреді?
4. «1С»-те мағыналар топтамасы деп нені атайды?
5. «1С»-те қандай объектілер пайдаланушы жұмысы сеансының ішінде деректердің уақытша жинақтарын сақтауға арналған, атаңыз.
6. Клиент-серверлік өзара іс-қимылдың қандай ерекшеліктері бар?
7. Жалпы модульдер не үшін арналған?
8. Жалпы модульде қандай бөлімдері жоқ, атаңыз және неге екенін түсіндіріңіз.
9. Объекті модулінің мәнмәтіні қалай жасалады?
10. Барлық объектілер үшін қандай оқиғалар шақырылады?
11. *ТолтырудыТексерудіӨңдеу, ЖазбаданБұрын, ЖазбаКезінде* оқиғалары қалай шақырылады?
12. Объекті менеджерінің модулі неге арналған?
13. Объекті менеджері модулінің мәнмәтіні қалай жасалады?
14. Объекті менеджерінің модулі серверде немесе клиентте орындалады ма?
15. Препроцессор және компиляциялау директивасының нұсқаулықтары не үшін пайдаланылады?
16. `&Клиентте`, `&Серверде`, `&СервердеМәнмәтінсіз`, `&КлиенттеСервердеМәнмәтінсіз` директивалары нені көрсетеді?
17. Объекті модулінің ауыспалы шамалары, процедуралары немесе функциялары не үшін экспорттаушы ретінде анықталуы мүмкін?

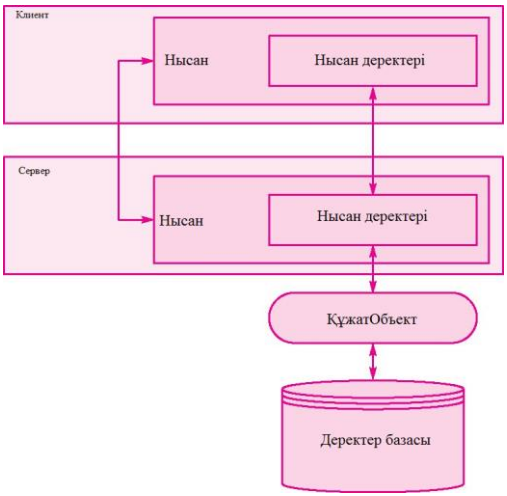
# 13 тарау

# НЫСАНДАРДЫ БАҒДАРЛАМАЛАУ

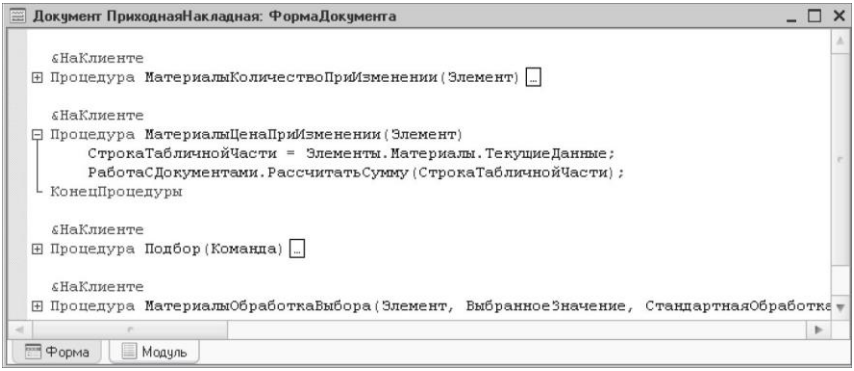
## 13.1. НЫСАННЫҢ БАҒДАРЛАМАЛЫҚ ОБЪЕКТІСІ

Нысан «1С: Кәсіпорын» режимінде қолданбалы шешімнің жұмысы кезінде құрылатын бағдарламалық объекті болып табылады. Нысанді әзірлеушінің қатысуынсыз платформаның көмегімен құруға болады, немесе ол әзірлеушімен орнатылған тілде суреттелген алгоритмді орындау нәтижесінде құрылады.

Нысанның бағдарламалық объекті ретіндегі негізгі ерекшелігі оның бір уақытта клиентте де, серверде де болуымен айқындалады (сур. 13.1). Сәйкесінше әр процедура үшін




13.1-сурет. Нысандардың клиент-серверлік өзара әрекеттестігі




13.2-сурет. Нысан модулі

нысан модулінде әзірлеуші оның орындалу мәнмәтінін көрсетуі қажет: серверде немесе клиентте (бұл үшін компиляция нұсқаулары қолданылады). Нысанның бағдарламалық объектісі платформамен Нысан конфигурация объектісінің негізінде құрылады. Конфигурация объектілерінің дарағында суреттелген әр Нысан конфигурация объектісінің модулі бар. Бұл модульде әзірлеушімен орнатылған тілде жасалған процедуралар орналасады (сур. 13.2). Бұл процедуралар уақыттың нақты, алдын ала белгілі мезеттерінде шығарылып, Нысан бағдарламалық объектісінің іс-әрекетін және оның мүмкіндіктерін белгілейді.

«1С:Кәсіпорын» режимінде қандай да бір әрекеттің орындалуы клиенттік қосымшада басталады, содан соң басқару серверге беріледі, ол бірнеше әрекетті орындап (серверлердің кластерімен байланыс орнату, ақпарат базасында қосылатын пайдаланушыны сәйкестендіру, жүйенің негізгі терезесін, жұмыс үстелін құру жүзеге асырылады), басқаруды клиентке қайтарады. Басқаша айтқанда, клиенттік қосымша серверді қандай да бір әрекеттер орындау үшін шақырады, сервер оларды орындап, басқаруды және бұл әрекеттердің нәтижесін клиенттік қосымшаға қайтарады.

Бұл өзара әрекеттестікке жауап беретін платформаның механизмдері тіпті жылдамдығы төмен байланыс арналарында жүйе жұмысының қолайлы жылдамдығын қамтамасыз ету үшін оңтайландырылған. Нысанді ашатын кезде:

- объект дерекқордан оқылады;
- объект нысан деректеріне конверсияланады;
- объект жадтан жойылады;
- нысан деректері клиентке жеткізіледі.

Деректерді нысаннен жазып алатын кезде:

- нысан деректері клиенттен алынады;
- нысан деректері объектіге конверсияланады;
- объект дерекқорға жазылады;
- объект жадтан жойылады.

Клиентте орындалатын процедурадан бағдарлама негізінде серверде орындалуы тиіс процедураны шақырып алуға болады. Бұл жағдайда да сервер шақыртылады, басқару оған беріледі, қажет әрекет орындалып, басқару клиентке, шақырту жүзеге асырылған процедураға қайтарылады. Клиенттің серверге қалай байланысатыны алдын ала белгілі болмайтындықтан, мұндай серверлік шақырту жүйе үшін «мүлдем байқалмайтын» немесе, керісінше, өте «шығынды» болуы мүмкін. Мысалы, мобильді байланыс арналары қолданылған кезде жүйе серверге өтініш жасайды, оны байланыс арнасы арқылы жіберіп, содан соң оны серверде орындап, жауапты байланыс арнасы арқылы қайтарады. Егер «сезімтал» клиент жұмыс жасаса, мысалы, GPRS арқылы, онда әрбір шақырту пайдаланушымен сезілетін болады. Сол себепті де сервердің бағдарламалық кодтағы шақыртулары әрдайым мақсатқа сәйкестік тұрғысынан мұқият ойластырылуы қажет, әзірлеуші сервердің шақырылу жиілігін және жіберілетін ақпарат көлемін басқарып отыруы тиіс. Код құрылымы шешілетін тапсырманың қолданбалы логикасымен емес, клиент-серверлік өзара әрекеттестіктің логикасымен айқындалуы тиіс, және клиенттік код орындалуы тиіс әрекеттердің жүйелілігі ретінде белгіленбейді, ол клиенттен серверге және кері бағытта басқаруды тапсыру сценарийі негізінде ойластырылады. Олай болмаған жағдайда әзірленетін жүйенің тезәрекеттігі мен өнімділігі өте төмен болуы мүмкін.

## 13.2. НЫСАН ПАРАМЕТРЛЕРІ МЕН РЕКВИЗИТТЕРІ

*Нысан реквизиттері* нысан жұмыс барысында қолданатын деректерді сақтау үшін арналған. Реквизиттерде нысан элементтерінде көрсетілетін және редакцияланатын деректер сақталады. Сондай-ақ реквизиттерде көрсетілмейтін, бірақ нысанмен жұмыс барысында қолданылатын деректер сақталады. Нысан реквизиттерінде сақталатын деректердің барлығы *нысан деректері* деп аталады.

Нысанның деректерді өзгертуге мүмкіндік беретін әрбір элементі нысанның кейбір реквизиттерімен байланысты. Пайдаланушы деректерді нысан элементінде өзгертетін кезде, ұқсас өзгерістер онымен байланысты реквизитте де орын алады. Кері бағытта да солай. Нысан реквизиті

бағдарламалық негізде өзгертілетін кезде, онымен байланысты нысан элементінде көрсетілетін мән де өзгереді.

Нысанның барлық реквизиттері міндетті түрде нысанде көрсетілуі тиіс емес. Реквизиттердің бөлігі онымен байланысты болмауы мүмкін, ал мұндай реквизиттерде сақталатын деректер көрсету немесе интерактивті редакциялау үшін арналмаған, нысан жұмысының ішкі алгоритмдерін қамтамасыз ету немесе орнатылған тілден бұл нысанмен бағдарлама негізінде өзара әрекеттесу үшін арналған.

Нысан реквизиттері нысан бағдарламалық интерфейсінің бөлігін құрайды, оның көмегімен бұл нысанмен «сырттан» өзара әрекеттестік құрылады — басқа нысандерден немесе бағдарламалық кодтың үзінділерінен.

Нысан реквизиттері нысан жұмыс жасау уақыты бойына сақталатыны маңызды жайт болып табылады.

*Нысан параметрлері,* реквизиттермен салыстырғанда, нысанді ашар кезде оның функционалдылығын басқаруға арналған, басқаша айтқанда, оны ашатын және жабатын мезетте ғана болады. Нысанді құрғаннан немесе ашқаннан кейін негізгі параметрлерден басқа барлық параметрлер жойылады, өйткені нысанның кейінгі жұмысы үшін олардың қажеті жоқ.

*Негізгі параметрлер* — бұл параметрлердің ерекше түрі, олардың мәндері нысан құрылғаннан кейін де қажет болуы мүмкін, мысалы, басқа ашық НЫСАНДАРДЫҢ арасынан нысанның дәл осы түрін табу үшін.

Нысанді шақырту кезінде әзірлеушімен құрылған параметрлердің мәндерін нысанның жүйелік параметрлерімен (ондайлар болған жағдайда) бірге параметрлердің құрылымында көрсетуге болады.

Нысанның параметрлерін нысанге оны ашатын мезетте жіберуге болады. Жіберілген параметрлерге талдауды ПриСозданииНаСервере()/СервердіҚұрғанКезде() оқиғасында жүзеге асыруға болады (Параметрлер топтамасы УправляемаяФорма/БасқарылатынНысан объектісінің қасиеті болып келеді):

```
// Шақырту орнында.
// Нысан параметрін құрамыз.
Параметрлер = Жаңа Құрылым ();
Параметрлер.Қосу ("Маңыздылық",
Алдын ала анықталған мән("Аудару.Маңыздылық.
Маңызды"));
// Параметрлер көрсетілген нысанді ашамыз.
Нысанді ашу("ЖалпыНысан.ҚарауНысані",Параметрлер)
...
```

```
// Нысан модулінде.
&Серверде
НысандеҚұрғанКезде процедурасы(Бас тарту,
Стандартты өңдеу)
Егер Параметрлер.Маңыздылық = Аудару.Маңыздылық.
Маңызды, онда
...
СоңыЕгер;
ПроцедураСоңы
```

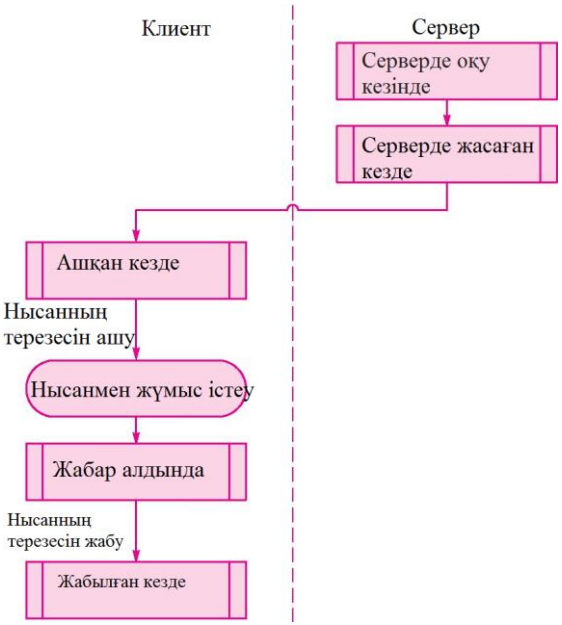
Нысандер арасындағы автоматты өзара әрекеттестікті сақтау үшін жүйе НЫСАНДАРДЫ ашатын кезде оларды басқару үшін қолданылатын стандартты параметрлердің қатарын ұсынады. Бұл параметрлердің көмегімен жүйе нысан жолдарында таңдалатын нысандерден таңдауды, объект нысандерін ашуды, стандартты тапсырмалардың жұмысын жүзеге асырады, яғни, олар жүйеге кіріктірілген түрлі интерфейс жұмысының сценарийлерін қамтамасыз етеді. Алайда әзірлеуші де бұл параметрлерді орнатылған тілде қолдана алады, және НысандіАшу() тәсілін шақыртқанда оларды тапсырады.

## 13.3. НЫСАН ОҚИҒАЛАРЫНЫҢ КЛИЕНТТІК ЖӘНЕ СЕРВЕРЛІК ӨҢДЕУІШТЕРІ

Айнымалы шамаларды және негізгі бағдарлааманы сипаттаудан бөлек, нысан модулі нысанмен байланысты оқиғаларды өңдейтін процедуралардың сипаттамасын қамти алады. Ең алдымен бұл – серверде нысанді құрған кезде пайда болатын оқиға. Әлдеқашан жұмыс жасап келе жатқан ақпараттық база объектісінің нысанін құратын кезде пайда болатын оқиғалардың жүйелілігі, сондай-ақ нысан терезесінің ашылу мен жабылу жүйелілігі 13.3 суретте көрсетілген.

Нысанді серверде ашуға дайындайды, содан соң ол клиентте ашылады. Нысанді сервер жағында ашатын кезде екі оқиға шақыртылады: СервердеОқуКезінде және СервердеҚұруКезінде.

Нысанның бірінші оқиғасы СервердеОқуКезінде тек ақпараттық базада бар объектілер үшін шақыртылады. Бұған қоса, бұл оқиға нысанның өзімен емес, нысанның негізгі реквизит түрі белгілейтін кеңейтілген нұсқасымен жеткізіледі. Бұл оқиғаның өңдеуішінде, АғымдағыОбъект параметрінде әзірлеушіге нысан құрамындағы қолданбалы объект және оның бар функционалдығы қолжетімді, яғни, осында аталған объектінің экспортталатын әдістерін шақыртуға немесе




13.3-сурет. Нысан модулінің оқиғаларын шақыртудың жүйелілігі

нысанның деректерін объектіге конверсияламай-ақ оның реквизиттерінің мәндерін алуға болады. Осылайша бұл оқиғаның өңдеушісінде әзірлеуші объектінің деректеріне тәуелді нысанның деректерін ашу үшін оларды дайындай алады.

Нысанның екінші оқиғасы Серверде құру кезінде әрдайым жаңа және бұрыннан бар объекттердің нысандерін ашатын кезде шақырылады. Бұл жерде нысанде көрсетілетін қолданбалы объект қолжетімсіз. Бұл оқиғаның өңдеушісінде әзірлеуші ашу үшін нысанді, оның сыртқы көрінісін толығымен дайындап, түрлі жағдайларға байланысты оның интерфейсті қасиеттерін ретке келтіре алады.

Бұл оқиғаның өңдеушісінде әзірлеушінің нысанді ашудан да (`Бас тарту = Ақиқат`), нысанді ашатын кезде стандартты әрекеттерді орындаудан да (`стандартты өңдеу = өтірік`) бас тартуға мүмкіндігі бар, егер, мысалы, талап етілген жағдайлар орындалмаса. Стандартты әрекеттердің жиынтығы түрлі нысандер үшін әртүрлі, және нысанның негізгі реквизитіне сәйкес келетін нысанның кеңейтілген нұсқасымен айқындалады. Мысалы, тізімнің нысані үшін стандартты өңдеу нысанді ашу кезінде көрсетілетін параметрлерді динамикалық тізімге тапсырудан тұрады, ал есептің нысанін ашатын кезде стандартты өңдеуді жүзеге асыру барысында

нысанге есептің нұсқасы мен пайдаланушының теңшелімдері жүктеледі.

Бұдан кейін көрсетілуге толығымен дайын нысан клиентке жіберіледі, және нысанның клиенттік оқиғасы `АшатынКезде` шақырылады. Бұл оқиға нысанді ашатын кезде пайдаланушыға оның терезесін көрсетер алдында пайда болады. Егер де нысанның қандай да бір себептермен жабық тұруы қажет болса, сондай-ақ бұл жерде нысанді ашудан бас тартуға болады. Егер нысан нақты ашық болса, онда әзірлеуші бұл оқиғаның өңдеушісінде серверде орындауға келмейтін кейбір интерактивті әрекеттерді орындай алады: пайдаланушыға ескерту көрсету немесе деректері негізгі ашылатын нысанге тәуелді байланыстырылған нысанді ашу.

Нысанді жабатын кезде клиенттің жағында екі оқиға орын алады: `ЖабарАлдында` және `ЖабарКезде` (бұл жерде объектіні нысаннен жазып алатын кезде (нысанді жабар уақытта) туындайтын оқиғалар қарастырылмайды).

Жабар Алдында оқиғасы нысанді жабар кезде нысан терезесінің жабылуына дейін туындайды. Бұл оқиғаның өңдеушісінде әзірлеуші нысанді жабудан бас тарту `(Бас тарту = = Ақиқат)`, сондай-ақ нысанді жабу кезінде стандартты әрекеттерді орындаудан `(СтандарттыӨңдеу = Өтірік)` бас тарту мүмкіндігіне ие, мысалы, талап етілген жағдайлар орындалмағанда. ЖабарАлдында оқиғасынан кейін орындалатын стандартты әрекеттердің жиынтығы түрлі нысандер үшін әртүрлі болып келеді, және нысанның негізгі реквизитіне сәйкес келетін нысанның кеңейтілген нұсқасымен айқындалады. Мысалы, анықтамалықтың элементі өзгертілсе, онда жабар алдында өзгерістерді сақтау өтініші стандартты әрекеттердің бірі болады.

Жабар кезде оқиғасы нысанді жабатын кезде нысан терезесін жауып болғаннан кейін туындайды. Бұл оқиғаның өңдеушісінде нысан жабық болған кезде ғана орындалуы тиіс алгоритмдерді сипаттауға болады, мысалы, негізгі нысан ашық болған жағдайда ғана ашылатын қосымша нысанді жабу қажет болғанда.

## 13.4. НЫСАН ПӘРМЕНДЕРІ. ПӘРМЕН МОДУЛІ

Метадеректердің белгілі бір объектісімен байланысты операцияларды орындау үшін бұл объектінің пәрмендері қолданылады. Бұған қоса, объектінің параметрленбеген пәрмендері құрамында метадеректердің объектісі бар қосалқы жүйелердің пәрмендер интерфейсінде қолжетімді болады. Егер пәрмен параметрленген болса,

құрамында пәрмен параметрінің түріне сай келетін түрдегі нысан реквизиттері бар нысандерде (нысан негізгі реквизитінің бірінші деңгейлі бағыныңқы реквизиттерін қоса) қолжетімді болады. Пәрмендер үшін пәрменді орындаудың тәртібін жазып беру қажет. Бұл үшін пәрмен модулі қызмет етеді, ол модульде ОбработкаКоманды()/ПәрмендіӨңдеу() өңдеушісін жүзеге асыру қажет. Бұл процедура &НаКлиенте/&Клиентте директивасымен жүзеге асырылуы тиіс, себебі пәрмен клиенттік қосымшада орындалады. Дегенмен, егер бұл пәрменде орындау қажет болса, пәрмен модулінде орналасқан басқа процедуралар мен атқарымдар өзге директивалармен жүзеге асырылуы мүмкін (&НаСервере/&Серверде, &НаКлиенте/&Клиентте, &НаСервереНа-Клиенте/&СервердеКлиентте). Пәрмен модулі келесіні қамти алады, мысалы, белгілі бңр бухгалтерлік шоттың карточкасын басып шығару үшін алдын ала қойылған параметрмен есеп нысанін ашу немесе тауардың түрі бойынша іріктеп алу негізінде тауарлар тізімінің нысанін ашу.

Пәрмен модулі басқарылатын нысанның модулі секілді серверде де, клиентте де болады. Пәрменнің модулінде компиляцияның келесі директивалары қолданылуы мүмкін:

- `&НаКлиенте/&Клиентте` — процедура/атқарым басқарылатын клиентте орындалады;
- `&НаСервере/&Серверде` — процедура/атқарым серверде орындалады;
- `&НаКлиентеНаСервере/&КлиенттеСерверде` — процедураны/атқарымды басқарылатын клиентте де, серверде де орындауға болады.

`ОбработкаКоманды( )/ПәрмендіӨңдеу( )` процедурасы міндетті түрде &На- Клиенте/&Клиентте компиляция директивасымен жүзеге асырылуы тиіс екендігін ұмытпаған жөн, себебі пәрмен клиенттік қосымшада орындалады.

Пәрмен модулінің клиенттік процедураларынан серверлік процедураларды шақыртуға болады, оларды жүзеге асырып болған соң, кодтың орындалуы клиентке қайтарылады. Бірақ бұл жағдайда серверлің процедуралардан/атқарымдардан клиенттік процедураларды шақыртуға болмайды.

Пәрмен модулі клиенттік процедураларының мәнмәтіні келесінің көмегімен құрылады:

- басқарылатын клиенттерде (басқарылатын қосымша режиміндегі «жіңішке» клиент, Web-клиент, «жуан» клиент) қолжетімді жаһандық мәнмәтіннің қасиеттерімен және әдістерімен;
- Клиент қасиеті қосылған (басқарылатын қосымша) клиенттік жалпы модульдердің және Серверді шақырту қасиеті қосылған жаһандық емес серверлік жалпы модульдердің экспортталатын атқарымдарымен және процедураларымен;
- басқарылатын қосымша модулінің экспортталатын айнымалы

мәндерімен, процедураларымен және атқарымдарымен;

- пәрмен модулінің жергілікті мәнмәтінімен.

Пәрмен модулі серверлік процедураларының мәнмәтіні келесінің көмегімен құрылады:

- жаһандық мәнмәтіннің қасиеттерімен және әдістерімен;
- серверде құрылған жалпы модульдердің экспортталатын атқарымдарымен және процедураларымен (бұл модульдер үшін Сервер қасиеті орнатылған);
- пәрмен модулінің серверлің әдістерімен.

Пәрмен модулі тек процедуралар мен атқарымдардың сипаттамасын қамтуы қажет, және онда модульдің айнымалы мәндері мен денелері болмайды.

## 13.5. КЛИЕНТ-СЕВЕРДІҢ ӨЗАРА ӘРЕКЕТІН ТЕСТІЛЕУ ЖӘНЕ ОҢТАЙЛАНДЫРУ

«1С:Кәсіпорын» интерфейсі тек қарапайым жергілікті желіде ғана емес, сонымен қатар интернет арқылы, оның ішінде, жылдамдығы төмен байланыс каналдары, мәселен, GPRS арқылы жұмыс істеу мүмкіндігіне бейімделген (бағытталған). Платформада жүйенің осындай жағдайларда істейтін жұмысын автоматты түрде оңтайландыратын механизмдерінің көп көлемі қамтылған.

Сонымен қатар, «қосылу жылдамдығының төмендігі» деген арнайы іске қосу параметрі бар, онда қосымша оңтайландыру құралдарының жұмысы қамтылған. Алайда, аталған платформада оңтайландыру жұмыстары конфигурацияны жасаушылардың қатысуынсыз толық орындалуы мүмкін емес.

Қолданбалы шешімдерді әзірлеу кезінде жылдамдығы төмен байланыс каналдарын қолдану барысында конфигурацияның тиімді жұмыс істеуін қамтамасыз ету үшін белгілі бір әдістмені ұстану қажет.

Оңтайландырудың негізгі екі мәні бар, олар:

- қоңыраулар саны;
- берілетін деректер (трафик) көлемі.

Ескеретін жайт, қосылыстардың кейбір түрлерінде қоңыраулар саны жүйенің өнімділігіне өте қатты әсер етеді.

Платформада жүйе жұмысын оңтайландыруға және бағалауға арналған мынадай құралдар бар:

*1. Өнімділік көрсеткіштері.* Механизм «1С:Кәсіпорын» режимінде қоңыраулар саны және пайдаланушының интерактивті әр іс-әрекеті бойынша берілетін мәліметтер көлемі туралы жалпы ақпарат береді. Ол ақпаратты жұмыс барысында беретіндігімен және оған арнайы әркет жасау талап етілмейтіндігімен қолайлы, оның көмегімен тек арнайы бағытталған талдау жұмыстарын ғана жасап қоймай, талдау жасау барысында көрсеткіштерін қадағалауға да болады.

*2. Конфигуратордағы өнімділік деңгейін өлшеу.* Сервердегі қоңыраулар санын орындалатын модульдер кодының жекелеген жолдары арқылы бағалауға мүмкіндік береді.

*3. Серверден қоңырау түскенде кідірісті имитациялау.* Жасау кезінде жүйе жылдамдығы төмен байланыс каналдарында жұмыс істейтін жағдайда қалай әрекет ететінін көзбен бағалауға мүмкіндік береді.

Конфигурацияны жасау кезінде интерфейстің нақты жұмысына платформа мен конфигурацияның бірлескен жұмысы жататындықтан, оңтайландыру жұмыстарындағы негізгі қиындық осында. Платформа жұмысын оңтайландырудың негізгі стратегиясы түскен қоңыраулар саны мен мәліметтер көлемі жөнінде клиентке берілетін ақпаратты азайтуға бағытталған. Сол үшін платформа клиентке бірден бар ақпаратты емес, тек керегін ғана береді, яғни мәліметтерді көп көлемде бермеу үшін ақпарат беру қағидасын, мүмкіндігінше, «талап етуіне қарай» іске асырады. Серверге көп жүк (салмақ) салмау үшін ақпарат белгілі бір порциямен беріліп отырады. Клиентке бұған дейін берілген ақпаратты қайтадан бермеу үшін платформада көп деңгейлі көшіріп сақтау әдісі қолданылады. Осылайша, жүйе жұмысын ондағы жүктеменің көлемі мен санына байланысты оңтайландыру қажет. Соған сәйкес платформа жұмысына қатысты компромисс шешімдер қабылданады.

Дайын шешімді тексеру үшін дұрыс сценарий таңдау қажет. Қолданбалы шешімдерді оңтайландыру кезінде жұмыстың типтік сценарийлерін тексеру қажеттігін ескеру қажет, сол себепті тексеріс кезінде мәліметтердің барынша шынайы көлемін және пайдаланушы іс-әрекетінің жүйелілігін ескеру қажет. Мәселен, кестесі бар бөлік жұмысын кестесі бар сол бөлікте болжанып отырған жолдар саны болған кезде ғана тексеру керек. Бұл ретте, өнімділіктің төмендеп кетпеуін білу үшін типтік (орташа) жолдар саны сияқты күтілетін сирек жағдайлардағы жұмыстарды тексерген дұрыс. Негізгі принцип — сценарийдегі тест аралығын шынайы өмірдегіге барынша жақын келтіру (жақындату).

Сонымен қатар, қосылу жылдамдығы төмен режимге қосылған тесттерді жеке өткізуге болады, оны сервер жүктемесін имитациялау режимімен шатастыруға болмайды. Қосылу жылдамдығы төмен режим платформаның жұмысын жылдамдығы төмен каналдарға сай етіп оңтайландыру үшін оның қалпын өзгертеді. Имитация режимі жүйенің жылдамдығы төмен каналда жұмыс істеген кезде (сырттай) қалай әрекет ететінін көруге мүмкіндік береді. Имитация режимі кезінде пайдаланушының жұмыс істеу қолайлығын оперцияны орындау уақытын өлшеу арқылы бағалау да маңызды болып табылады.

Барлық аталған өлшемдерді, жалпы алғанда, әзірлемелерді жасау кезінде, сондай-ақ ақпарат базасының файлдық нұсқасын қолданып жүргізуге болады.

Ең күрделісі тест нәтижелерін түсіндіру. Жүйе іске қосылған кезде платформа мен конфигурация өзара тығыз жұмыс жасайды, сол себепті қашанда, нақты жағдайға конфигурация оң немесе теріс әсер етуі мүмкіндігін түсіну оңай емес.

Әрине, таладау жүргізуге жетерлік жай жағдайлар да бар, мысалы, үлгі модулінен серверге түсетін айқын қоңыраулар (жүктемелер). Алайда, детальдары нақты механизмдер жұмысына жататын, бұдан күрделі әсерлер бар. Мысалы, үлгі элементтерінің кейбір сипаттарындағы (қасиеттеріндегі) өзгерістер жүйе жұмысын бәсеңдетуі мүмкін, өйткені бұл кезде платформа серверден жаңа ақпарат алуға мәжбүр болады.

Модуль кодтарындағы сервер қоңырауларын (жүктемелерін) талдау жұмыстарын конфигуратор өнімділігін өлшейтін режим көмегімен орындауға болады. Алайда, бұл режим тек модульдерді орындау барысында ғана тікелей орындалатын қоңырауларды (жүктемелерді) көрсететінін және ол модуль орындалмайтын, яғни одан тыс платформа арқылы орындалатын қоңырауларды (жүктемелерді) көрсетпейтінін ескеру қажет.

Конфигурация жұмысының тиімділігіне моуль кодтары ғана емес, сонымен қатар үлгі сипатынан алынатын әр түрлі күйге келтірушілер де әсер ктуі мүмкін. Сол себепті, талдаудың қосымша әдістемесі ретінде нақты көрсеткіштерді ешқандай спецификалық күйге келтірушілер енгізілмейтін, бірақ платформаның сондай мүмкіндіктері жасырын күйге келтірушілерімен қолданылатын кейбір тест конфигурациясындағы осыған ұқсас көрсеткіштермен салыстыру жұмыстарын пайдалануға болады.

Оңтайландыру кезінде жұмысты қоңыраулар (жүктемелер) санын берілетін деректер көлемін азайтуға бағыттау қажет. Ол үшін әр түрлі әдістемелер қолданылады: бірнеше қоңырауларды (жүктемелерді) бір қоңырауға (жүктемеге) біріктіру және т.б. Сол себепті, конфигурация кодының қолданбалы ұғымына себерші емес, керісінше клиент-сервердің өзара жұмыс жасау ұғымына себепші болуы тиіс.

Мысалы, қызметтерді жеке бөліктерге бөлу көзқарасы бойынша конфигурацияны іске қосу кезінде клиентке әр шағын жүйеден жеке-жеке қажет кейбір бастапқы деректерді алу үшін серверге қоңырау жіберу қажет. Алайда, клиент-сервер әзірлемесінің ерекшеліктеріне байланысты қажет ақпаратты бір қоңырау арқылы алуға болатынын жүзеге асыру қажет.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. Сізге үлгінің бағдарламалық объект ретінде қандай негізгі ерекшеліктері белгілі?
2. Үлгіні ашқан кезде қандай жағдайлар орын алады?
3. Мәліметтерді үлгіден жазып алған кезде қандай жағдайлар орын алады?
4. Неге әзірлеушіге сервердің әр жүктемесінің маңыздылығын ойластырып отыруы қажет?
5. Үлгі мәліметтері болып не аталады?
6. Үлгі деректемелері үлгі параметрлерінен несімен ерекшеленеді?
7. Үлгінің қандай параметрлері оның негізгі параметрлері деп аталады?
8. Серверде () жасаған жағдайда қандай әрекеттер жасауға болады?
9. Бұрыннан бар объект үлгісін жасаған кезде болатын оқиғалар ретін сипаттаңыз.
10. Серверде немесе клиенте қандай үлгіні ашуға дайындық жүруде?
11. Серверден оқыған кезде, оқиғаны сипаттаңыз. «Ағымдағы объект» деген параметрдің мақсаты (атқаратын қызметі) қандай?
12. Серверде жасалған кезде, оқиғаны сипаттаңыз.
13. Үлгі жабылған кезде клиент жақта қандай жағдай болады?
14. Команда модулінде компиляцияның қандай директивасы қолданылады?
15. Команданы өңдеу () процедурасы компиляцияның қандай директивасы арқылы алдын алуы тиіс?
16. Команда модулінің клиент процедурасының контексі қалай жасалады?
17. Команда модулінің сервер процедурасының контексі қалай жасалады?

# 14 тарау

# САУАЛДАРМЕН ЖҰМЫС ЖАСАУ

## 14.1. САУАЛДАРДАҒЫ МӘЛІМЕТ КӨЗДЕРІ

Әзірлеуші өз алгоритмдерін жүзеге асыратын сауалдарды пайдалану мүмкіндігіне ие болу үшін платформадан сауал тілін таңдау мүмкіндігі берілген. Бұл тіл SQL негізінде құрылған, сондай-ақ ауқымы жағынан қаржы-экономикалық міндеттер спецификасын көрсетуге және қолданбалы шешімдерді әзірлеуге жұмсалатын күшті барынша қысқартуға бағытталып, айтарлықтай кеңейтілген.

Деректер базасындағы таблицаларға жіберілетін сауалдарды қалыптастыру және орындау үшін жүйеде соған арналған «Сауал» деген арнайы объект бар. Тиісінше топталған және іріктелген деректерді, сондай-ақ ақпаратты әр түрлі уақытқа алу қажет болған жағдайда, оларды іріктеу күрделілігіне қарай, осы сауал қолданылады.

Сауалдар тілінің деректер көзіне деректер базасындағы таблицалар жатады. Таблицалар негізгі екі класқа бөлінеді: шынайы және виртуалды.

*Шынайы таблицалар* деректер базасында сақталады, яғни шын мәнінде бар деректер базасындағы таблица дегенді білдіреді. Шынайы таблицаны қолданған кезде мәні бірнеше шынайы жолдар функциясы болып есептелетін жолдар болуы тиіс.

*Виртуалды таблицалар* деректер базасында сақталмайды. Сауалды орындау үшін виртуалды таблицадағы ақпаратқа жүгінген кезде жүйе автоматты түрде деректер базасындағы шынайы таблицалардан ақпарат жинайды. Виртуалды таблицаның шынайы толтырылғандығы сауал мәтініне қойылатын параметрлердің мәні арқылы анықталуы мүмкін. Әр виртуалды таблица үшін

Әр виртуалды таблицаға сауалдарда таблицаны сәйкестендіру (идентификациялау) үшін қолданылатын атау беріледі. Таблица атауы ағылшын және орыс тілдерінде берілуі мүмкін. Мысалы: Анықтамалық. Номенклатура. Таблица мен жолдардың атауы сауал тіліндегі негізгі сөздермен сәйкес келмеуі мүмкін.

Таблицалардың жекелеген шағын кластарынан объектілі таблицалар жасалады. Бұл таблицалар «1С:Кәсіпорын» жүйесі объектілерінің анықтамалық, құжаттар және т.б. жай-күйін жазып сақтауға арналған. Әр объектілі таблицаға «1С:Кәсіпорын» жүйесі объектілерінің типі сәйкес келеді. Мысалы, Анықтамалық сияқты объект. Материалдары да, бір таблица Анықтамалық сияқты объектіге сәйкес келеді. Контрагенті — басқа. Объектілі таблицадағы жеке әр жазбада тиісті типтегі жеке объектінің жай-күйі жазылып сақталады. Осыған сәйкес әр объектілі таблицада «Ағымдағы жазбаның сілтемесі» сияқты жол бөлінген. Сондай-ақ объектілі таблицада тиісті жолдардағы жазбадан объектінің пайдаланушылық көрінісін алу әдісі көзделген.

Сондай-ақ объектілі таблицалар иерархиялық болуы мүмкін. Иерархиялық таблицалар үшін жазбаға сілтеме жасалған «Родитель» деген арнайы бөлінген жол қарастырылған, сілтемеге иерархияға сәйкес ағымдағы жазба бағынады. Таблицаның жолдары ретінде виртуалды немесе шынайы таблицаның жолдары, салынған таблица болуы мүмкін.

Салынған таблицаның кәдімгі жолдардан негізгі айырмашылығы мынада: бір жазбада кәдімгі жолға жалғыз бір ғана мән сәйкес келеді, ал салынған таблицада «Сауал нәтижесі» типті мән алдын ала жасалған бағандар жинағымен сәйкес келеді. Құжаттың немесе анықтамалықтың таблицалар бөлігі салынған таблицаға мысал бола алады.

Таблица жолдарындағы мәндер NULL типті болуы мүмкін. Мұндай мәндер таблицадағы жазба жолдарында болуы мүмкін, оларға мұндай жолдар көзделмеген немесе олардың болуы маңызды да емес.

## 14.2. САУАЛ ТІЛІ

Сауалды орындау үшін сауал мәтінін құрастыру қажет.

*Сауал мәтіні* — ол нұсқаулық, соған сәйкес сауал орындалуы тиіс.

Сауал мәтінінде ақпарат базасының қандай таблицалары сауалда пысықталуын талап ететін сауалдағы деректер көзі, таблица жолдары, нәтижелерді топтау, сұрыптау ережесі және т.б. ретінде қолданылатыны сипатталады.

Нұсқаулық арнайы тілде (сауал тілінде) жасалады және мынадай жеке бөліктерден — секциялардан, сөйлемдерден, сөздерден, функциялардан және комментариялардан тұрады.

«1С: Кәсіпорын» жүйесіндегі сауалдар тілінің бірден-бір басты ерекшеліктерінің бірі енгізілген тілдегі сияқты барлық негізгі сөздер екі тілде жазылады, яғни екі тілде жазылу нұсқасы бар: орыс және ағылшын тілдерінде.

Сауал мәтінін мына ережемен сипаттауға болады:

```
<Сауал мәтіні>
<Сауал сипаттамасы>
[<Сауалдарды біріктіру>]
[Нәтижелерді реттеу>]
[АВТОРЕТТЕУ]
[<Қорытындыларды сипаттау>]
```

Көріп отырғандай, сауал мәтіні бірнеше бөліктерден немесе секциялардан тұрады.

**<Сауал сипаттамасы>** — бұл сауал мәтініндегі жалғыз міндетті секция және көп жағдайларда соны ғана көрсетсе болғаны. Секцияда сауалдағы деректер көздері, іріктеу, топтау жолдары және т.б. анықталады. Бұл секция, өз кезегінде, жалпы ережелер жинағы ретінде сипатталады және одан әрі толық қарастырылады.

**<Сауалдарды біріктіру>** — сауалдар тілі орындалған бірнеше сауалдың нәтижелерін біріктіруге мүмкіндік береді.

**<Нәтижелерді реттеу>** деген секцияда сауал нәтижесінде жолдардың реттелу деңгейін анықтауға болады.

**<Автореттеу>** деген секция сауал нәтижесінде жолдарды автоматты түрде реттеу режимін іске қосуға мүмкіндік береді.

<Қорытындыларды сипаттау> деген секцияда сауалда қандай қорытындылар есебін шығару қажеттігін көрсетуге болады.

Бұған дейін айтылғандай, сауал мәтінінде міндетті түрде сауалды сипаттайтын секция болуы тиіс және онда мыналар айқындалады:

- сауал нәтижесінен болатын жолдар;
- сауалдағы деректер көзі — бастапқы таблицалар;
- сауалдағы деректердің іріктелуіне ықпал ететін шарттар;
- сауал нәтижелерін топтау тәртібі.

Сауалды сипаттайтын секция бір-бірімен өзара тығыз байланысты бірнеше сөйлемдерден тұрады:

```
<Сауал сипаттамасы>
ТАҢДАУ [РҰҚСАТ ЕТІЛГЕНДЕР] [ӘР ТҮРЛІ] [БІРІНШ <Саны>]
<Іріктелетін жолдар тізімі>
[ИЗ <Көздер тізімі>]
```

[ГДЕ <Іріктеу шарттары>]
[<Топтау жолдары> БОЙЫНША ТОПТАУ]
[<Іріктеу шарттары> БАРЛАРЫ]
[ӨЗГЕРТУ ҮШІН [<Жоғары деңгейдегі таблицалар тізімі>]]

Сауал сипаттамасы «ТАҢДАУ» деген негізгі міндетті сөзден басталады.

«РҰҚСАТ ЕТІЛГЕН» деген негізгі сөз – сауал тек пайдаланушының құқығы бар жазбаны ғана таңдайды дегенді білдіреді. Егер осы сөзді көрсетпесе, онда сауал пайдаланушының құқығы жоқ жазбаны таңдау кезінде қате орындалады. Бұл негізгі сөз жоғарғы жақта тұрған сөйлемде «ТАҢДАУ» деген сөзбен болуы тиіс және енгізілген сауалдармен қатар барлық сауалдарға қатысты болады. «ӘР ТҮРЛІ» деген негізгі сөздің көмегімен нәтижеде қайталанатын жолдар болмауы тиіс дегенді көрсетуге болады. БІРІНШ <Саны> деген конструкция сауал нәтижесі арқылы шекті жолдар санын сұрауға мүмкіндік береді. Ең бірінші тұрған жолдар іріктеп алынатын болады (сауал нәтижелерін реттеу ережесіне сәйкес). Ең бірінші тұрған жолдар іріктеп алынатын болады. Саны тұтас санмен жазылады. Сауал тіліне, егер енгізілген сауалда БІРІНШ деген конструкция болса, онда реттеу жұмыстарын енгізілген сауалда орындауға мүмкіндік берілді.

<Іріктелген жолдар тізімі> деген секцияда сауал нәтижесінен болатын жолдар сипатталады.

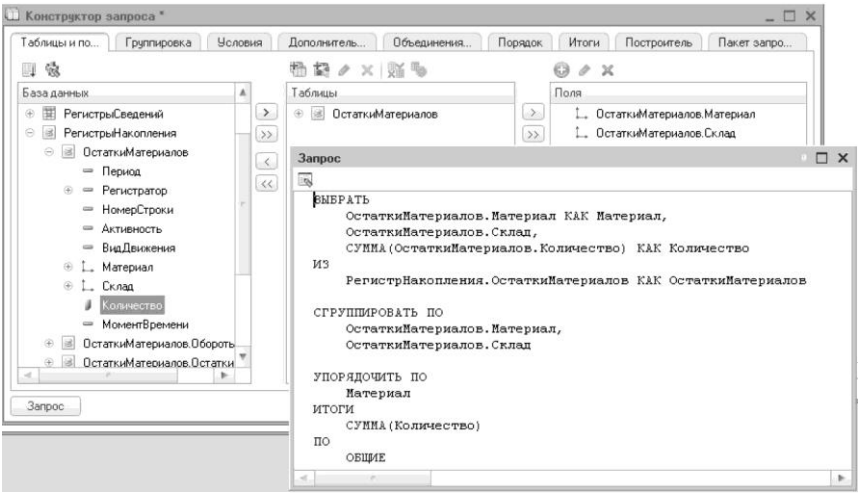
ИЗ <Көздер тізімі> деген сөйлемде ішіндегі мәліметтер сауалда өңделетін ақпарат базасындағы деректер көзі – таблицалар көрсетіледі. Егер дереккөздер сипаттамасы іріктелген жолдар тізімінде көзделген болса ғана олар төмен түсірілуі мүмкін. <Іріктеу шарттары> қайда деген сөйлем сауал нәтижесін сүзгіден өткізуге мүмкіндік береді. Нәтижеге көрсетілген шарт ақиқат болып танылатын жазбалар кіреді. ӨЗГЕРТУ ҮШІН деген сөйлем транзакциядан жазып алынатын Жазып алынған деректер басқа сессияларда оқуға қолжетімсіз болады. Файлдық нұсқа үшін көрсетілген таблицалар, ал клиент-сервер нұсқасы үшін тек іріктеп алынған жазбалар ғана бұғатталады. Олар транзакция аяқталған соң бұғаттан алынады. ТОПТАУ деген сөйлем сауал нәтижелерін топтау тәртібін сипаттауға мүмкіндік береді. БАРЛАРЫ деген сөйлемдер топтау нәтижелеріне шарт қоюға мүмкіндік береді.

Келесі мысалда, сауалдың орындалу нәтижесінде деректер материалдардың атауы бойынша реттелген «Материалдар қалдығы» деген регистр ішінен іріктеп алынатын болады:

```
   ТАҢДАУ
      Материалдар қалдығы. Материал Материал
ретінде,
      Материалдар қалдығы. Қойма Қойма ретінде,
СОМАСЫ(Материалдар қалдығы. Саны) ИЗ Саны ретінде
      Жианақтау регистрі. Материалдар қалдығы
Материалдар қалдығы ретінде
      МЫНАЛАР БОЙЫНША ТОПТАУ
      Материалдар қалдығы. Материал,
      Материалдар қалдығы. Қойманы Қорытындылар
материалы БОЙЫНША РЕТТЕУ
      СОМАСЫ (Саны)
   МЫНАУ БОЙЫНША
      Материал,
      Қойма
```

Сауалы топтау М а т е р и а л және Қ о й м а деген жолдар бойынша жүргізіледі, қоймадағы әр материалдың қорытынды санының есебі шығарылады.

Сауал мәтінін жасау үшін сауал конструкторын – визуалды мүмкіндік беретін құралды пайдалануға болады.




14.1. сурет. Сауал конструкторының әйнегі

(14.1-сурет.). Тіпті, сауал тілімен таныс емес пайдаланушының өзі конструктор көмегімен синтаксис жағынан дұрыс сауал жасау алады.

Сауал тілімен іске асырылатын мынадай маңызды мүмкіндіктерге бөлінеді:

- *бағынышты жолдарға нүкте арқылы жұмыс істеу.* Егер қандай да бір таблицадағы жолдарда сілтмелер болса (сілтемелер басқа таблица объектілерінде сақталады), онда әзірлеуші сауал мәтінінде оларға нүкте арқылы сілтеме жасай алады. Бұл ретте, осындай сілтемелерге енгізілетін деректер көлемі шектелмеген;
- *сызыған (салынған) таблицалармен жұмыс істеу* (мысалы, құжаттар мен анықтамалық элементтердің таблица бөлігінде). Салынған таблица бөліктерімен жеке таблицалармен және бір таблицадағы тұтас бір жолмен жұмыс істегендей жұмыс істеуге болады;
- *автоматты түрде реттеу.* Автоматты түрде реттеу режимі ақпаратты белгілі бір тәртіпте шығаруға мүмкіндік береді;
- *Қорытындыларды көп өлшемде және көп деңгейде қалыптастыру.* Жалпы қорытынды мен аралық қорытынды топтау және иерархия есебімен қалыптастырылады, деңгейлерін қарап шығу аралық қорытындысы шығарылып, еркін түрде, жүзеге асырылуы мүмкін. Уақыт өлшемдері бойынша қорытындысын шығару бағандары дұрыс құру қамтамасыз етіледі;
- *виртуалды таблицаларды қолдау.* Жүйе ұсынатын виртуалды таблицалар қолданбалы шешімдердің басым көпшілігіне күрделі сауалдар құрастыруға деген қажеттіліксіз-ақ дайын деректер алуға мүмкіндік береді. Мәселен, сондай виртуалды таблица тауар қалдықтары туралы деректерді қандай да бір уақыт аралығындағы кезеңдерге бөліп бере алады. Бұл ретте, виртуалды таблица сақталатын ақпаратты (бұрын есептелген қорытындыларды және т.б.) барынша пайдаланады;
- *стандарт SQL-операциясы.* Сауал тілінде SQLоперациясы үшін стандарт болып табылатын біріктіру (Union), қосу (Join) және т.б. сияқтыларға қолдау көрсетіледі;
- *уақытша таблицаларды пайдалану.* Уақытша таблицалар сауалдардың өнімділігін арттыруға, ал кейбір жағдайларда бұғат санын төмендетіп, сауал тілін түсінуге жеңілірек етуге мүмкіндік береді;
- *пакет сауалдары.* Пакет сауалдары уақытша таблицаны құру және оны пайдалану жұмыстары бір сауалда қамтылатындықтан, уақытша таблицамен жұмыс жасау тәсілін ыңғайлырақ етуге мүмкіндік береді. Пакет сауалында мынадай «;» таңбалармен бөлінген сауалдар бір жүйеде болады. Сауалдардың бірінен соң бірі орындалады.

Соңғы пакет сауалымен қайтарылған нәтиже немесе пакеттегі сауалдар жүйесімен іске асырылатын барлық пакет сауалдарының нәтижелер көлемі қолданылатын әдіске байланысты орындалған пакет сауалының нәтижесі болып табылады.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. Не сауал тілінің дереккөзі ретінде қолданылады?
2. Шынайы таблицалардың виртуалды таблицалардан айырмашылығы неде?
3. Объектілі таблицалардың неге арналғанын түсіндіріңіз.
4. Сауалда сызылған (салынған) таблицалар таблицаның жолдары бола ала ма?
5. Сауал жасалатын секцияны атаңыз.
6. Сауал мәтініндегі қандай секция міндетті болып табылады?
7. Сауал сипаттамасы деген секция нені анықтайды?
8. Сауал нәтижесін қалай топтауға болатынын түсіндіріңіз.
9. Сауалдағы қорытындыларды қалай санауға болады?
10. `РҰҚСАТ ЕТІЛГЕНДЕР` деген негізгі сөз сауалда не үшін қолданылуы мүмкін?
11. Сауал конструкторы қандай мүмкіндіктер береді?
12. Сауал тілімен іске асырылатын негізгі маңызды мүмкіндіктерді атаңыз.

# ЖҮЙЕЛІК БАҒДАРЛАМАЛАУ

# IV

## БӨЛІМ

# 15-тарау

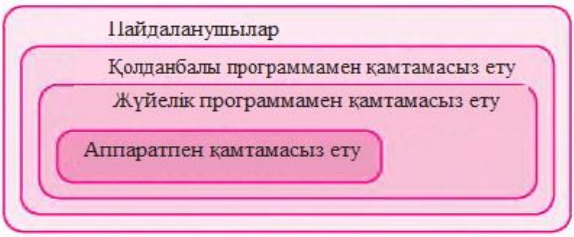
# РЕСУРСТАРДЫ БАСҚАРАТЫН ШАҒЫН ЖҮЙЕЛЕР

## 15.1. РЕСУРСТАРДЫ БАСҚАРУ ТУРАЛЫ ЖАЛПЫ МӘЛІМЕТТЕР

Компьютердің физикалық немесе аппараттық ресурстарына орталық процессор, жедел жады, сыртқы жады, деректерді беру шинасы және ақпаратты енгізіп, шығаратын әр түрлі құрылғылар жатады. Компьютердің жадында сақталатын мәліметтер мен ақпараттар компьютердің логика немесе ақпарат ресурстары деп аталады. Физикалық және логикалық ресурстарымен қоса компьютердің барлық ресурстары туралы айтқанда, әдетте, компьютер ресурстары немесе жүйе ресурстары деген термин қолданылады.

Компьютер арқылы қандай да бір бағдарламаны орындау үшін комьютер ресурстары қолжетімді болуы керек, ал оны жедел жүйе қамтамасыз етеді (15.1 рис.).

Жүйелік бағдарламалау әдісі бағдарламалар жасауға арналған жедел жүйелермен және оның интерфейстерімен тығыс байланысты. Жедел жүйелер барлық бағдарламалардың жұмысын қамтамасыз етіп, есептеу жүйелерін аппаратпен іске асыру тетіктерін жасыратын сервис функциясын ұсынады. Операциялық жүйе – бұл компьютер ресурстарының комплексі және оны ол басқарады, яғни операциялық жүйе




15.1. сурет. Бағдарламамен және аппаратпен қамтамасыз етудің өзара жұмыс жасауы

- бұл компьютер ресурстарының менеджері деп те айтуға болады.Осыдан компьютер ресурстарын басқару және осы ресурстарды диспетчерлеу немесе жоспарлау қызметі операциялық жүйенің негізгі функциялары болып табылатынын көруге болады.

Процестерді, жады, файлдарды және сыртқы құрылғыларды басқаратын шағын жүйелер ресурстарды басқаратын негізгі маңызды шағын жүйелер болып табылады, ал пайдаланушының интерфейсі, деректерді қорғайтын және әкімшілік ететін шағын жүйелер барлық ресурстарға ортақ шағын жүйелер болып табылады.

**Процестерді басқару.** Процестерді басқаратын шағын жүйелер есептеу жүйесінің жұмыс істеуіне тікелей әсер етеді. Операциялық жүйе әр орындалатын бағдарламаға бір немесе одан да көп процестерді ұйымдастырады. Операциялық жүйедегі әр сондай процестің ондағы ресурстардың процеске деген мұқтаждығы, сондай-ақ оған нақты бөлінген ресурстар (жедел жады саласы, процессордегі уақыт көлемі, файлдар, енгізу-шығару құрылғылары және т.б.) туралы мәліметтер қамтылған ақпарат құрылымы (таблица, дескриптор, процессор контексі), болады. Сонымен қатар, осы ақпарат құрылымында процестің жүйеде болу тарихын: ағымдағы жай-күйін (іске қосылғандығын немесе бұғатталғандығын), басымдығын, регистрлер мен бағдарлама есептегіштің жай-күйін және т.б. сипаттайтын мәліметтер сақталады.

Қазіргі заманауи мультибағдарламалық операциялық жүйелерде пайдаланушылардың бастама етуімен туындаған бірнеше процестер мен олардың қосымшалары, сондай-ақ бастама етілген, өз функциясын орындайтын операциялық жүйе де (жүйелік процестер) болуы мүмкін, өйткені процестер, бір уақытта, сол бір ресурстан үмітті болуы мүмкін, ал процестерді басқаратын шағын жүйе процестердің орындалу кезегін жоспарлайды, оларды тиісті ресурстармен қамтамасыз етеді, процестердің өзара жұмысына және үйлестірілуіне кепілдік береді.

**Жадыны басқару.** Жадыны басқаратын шағын жүйе физикалық жадыны жүйедегі барлық процестер арасында өзара бөледі, бағдарлама кодтарын және процесс деректерін жазады және оларды өшіріп, жады шеңберінен бөлінген жерге көшіреді, процесс кодының адреске тәуелді бөліктерінің күйін арнайы бөлінген жердегі адреске келтіреді. Жадыны басқару стратегиясы іріктеу, бағдарламаны немесе негізгі жадыдағы деректер блогін орналастыру және алмастыру стратегиясынан тұрады. Соған сәйкес кезекті блогты (сауал бойынша немесе алдын алу мақсатында) жадыға қашан жазу керектігін, оны жадыдағы қай орынға және қандай бағдарлама немесе деректер блогына көшіру керектігін, сондай-ақ жаңа блоктарды орналастыруға орын босату үшін негізгі жадыдан өшіру қажеттігін анықтайтын әр түрлі алгоритмдер қолданылады.

Заманауи операциялық жүйелердегі жадыны басқарудың ең танымал әдістерінің бірі – виртуалды жады. Виртуалды жады механизімін іске асырған кезде ол программистке өзінің иелігінде біртекті жедел жады бар деп санауына мүмкіндік береді, осы біртекті жедел жадының көлемі бағдарламалау жүйесі беретін адресация мүмкіндіктерімен ғана шектеледі.

Жадыны басқаратын маңызды функция — ол жадыны қорғау. Жадыны қорғау кезіндегі бұзушылықтар қолданбалы бағдарламалардың немесе операциялық жүйелердің өзінің бағдарламалардың басқа процестерімен бөлінген жады бөлігіндегі процестермен жұмыс істеумен тығыз байланысты. Жадыны қорғау құралы бұзақының бағдарламасын авариялық жолмен аяқтау арқылы осындай бұзушылықтардың жолын кесуі тиіс.

**Файлдарды басқару.** Файлдарды басқару функциясы операциялық жүйенің файлдар жүйесіне шоғырланған. Операциялық жүйе сыртқы жинақтағышта файл – таңбалы атауы бар байттардың қарапайым құрылымсыз жүйесі түрінде сақталатын деректердің жеке топтамасын виртуалдайды. Деректермен жұмыс істеу ыңғайлы болу үшін файлдар каталогтарға топталады, ал ол, өз кезегінде, деңгейі жоғары топтар – каталогтар құрады. Файлдық жүйе пайдаланушы немесе программист жұмыс істейтін файлдардың таңбалы атауларын өзгертіп, дискідегі деректердің физикалық адрестеріндегі файлдарды ортақ пайдалану жұмыстарын ұйымдастырып, оларды рұқсатсыз пайдаланудан қорғайды.

**Сыртқы құрылғыларды басқару.** Сыртқы құрылғыларды басқару функциясы сыртқы құрылғыларды басқаратын шағын жүйелерге, сондай-ақ енгізіп-шығаратын шағын жүйелер деп де аталатын шағын жүйелерге жүктеледі. Осы құрылғылардың аясы өте кең (олар: принтер, сканер, монитор, модем, манипулятор, жүйелік адаптер, осыған ұқсас әр текті цифрлық жаңарту құрылғылары және т.б.), осындай құрылғылардың жүздеген үлгілері (модельдері) процессормен және басқа да детальдармен ақпарат алмасу үшін қолданылатын командалар жинағымен және олардың жүйелілігімен ерекшеленеді.

Сыртқы құрылғының нақты бір моделін басқаратын және оның барлық ерекшеліктері қамтылатын бағдарлама *драйвер* деп аталады. Драйверді құру жұмыстарымен операциялық жүйелерді жасаушылармен қатар сыртқы құрылғыларды жасайтын компаниялар да айналысады. Операциялық жүйе драйверлер мен операциялық жүйелердің басқа да қалған бөліктері арасындағы нақты белгілі бір интерфейске қолдау көрсетуі тиіс.

Сол кезде жасаушы компаниялар – енгізіп-шығаратын құрылғыларды өндірушілер өз құрылғыларымен бірге нақты операциялық жүйеге арналған драйверлерді жеткізе алады.

**Деректерді қорғау және әкімшілік ету.** Есептеу жүйелеріндегі деректердің қауіпсіздігі аппаратуралардың істен шығуы мен жұмыс істемей қалуынан, бағдарламамен қамтамасыз ету жүйелерін қателесуден қорғауға бағытталған операциялық жүйенің істен шығып, жұмыс істемей қалуға төтеп бере алатын құралдармен, сондай-ақ оларды рұқсатсыз пайдаланудан қорғайтын құралдармен қамтамасыз етіледі. Жүйенің әр пайдаланушысына жұмыс процесінде операциялық жүйе сол жүйеге әкімшілік қызмет тарапынан рұқсат берілген пайдаланушының кіретініне көз жеткізетіндей, логикалық тұрғыдан кіру процедурасын жасауға міндетті болып табылады.

Есептеу жүйесінің администраторы пайдаланушының сол немесе басқа да іс-әрекетті жүзеге асыру мүмкіндігін анықтайды немесе шектейді, яғни олардың жүйенің ресурстарымен жұмыс істеу және оны пайдалану құқығын бекітеді.

Операциялық жүйенің жүйелер қауіпсіздігімен байланысты барлық оқиғаларды есепке алып отыратын аудит функциясы негізгі қорғау құралы болып табылады. Есептеу жүйесін істен шығаратын себептерге төтеп бере алатын қасиетін қолдау жұмыстары (дискілік RAID-массивін, резервтегі принтерлерді және тағы да басқа құрылғыларды, кей кездері орталық процестерді және т.б. резервтеу) резервтеу негізінде іске асырылады. Жүйенің жұмысқа төтеп беру қасиетін қамтамасыз ету мәселесі – жүйе администраторының маңызды міндеттерінің бірі, бұл үшін ол бірқатар арнайы құралдар мен инструменттерді пайдаланады.

**Қолданбалы бағдарламалау интерфейсі.** Қолданбалы программистер өз қосымшаларында, оларға сол немесе басқа да іс-әрекетті жүзеге асыру үшін тек операциялық жүйе ғана ие болатын ерекеше деңгей (мәртебе) талап етілетін жағдайда, операциялық жүйеге жүгінеді (операциялық жүйемен жұмыс істеу тәсілін) қолданады. Программиске операциялық жүйенің мүмкіндіктері қолданбалы бағдарламалау (ApplicationProgrammingInterface— API) интерфейсі деп аталатын функциялар топтамасы (жинағы) ретінде қолжетімді.

Қосымшалар жүйедегі қоңыраулардың (жүктеме) көмегі арқылы API функциясына жүгінеді. Қосымша операциялық жүйенің қызметін алатын әдіс шағын бағдарламалар жүктемесіне (қоңыраулар) өте ұқсайды. Жүйе жүктемесін (қоңырауларын) іске асыру әдісі операциялық жүйе құрылымының ұйымдастырылуына, бағдарламалау тілінің аппарат платформасының ерекшеліктеріне байланысты.

Бұдан әрі қолданбалы бағдарламалау интерфейсі толығырақ қарастыралытн болады.

## 15.2. WIN API ҚОЛДАНБАЛЫ БАҒДАРЛАМАЛАУ ИНТЕРФЕЙСІ

Кез келген бағдарлама, ол аппаратура (процессор, дисковод, принтером) немесе операциялық жүйе арқылы орындалатын командалар жиынтығы (топтамасы). Соңғы жағдайда ақыр соңында жабдық өзі орындайтын кейбір функциялар (шағын бағдарламалар) операциялық жүйе командасы болып табылады. Қолданбалы бағдарламалау интерфейсі, ол қолданбалы бағдарламалауда қолданылатын, Windows үшін жазылған стандарт блоктар. Сондай-ақ API бұл функцияның қалай іске асырылатынына байланысты оның дерексізденуіне (абстракциялануына) мүмкіндік береді. Бағдарлама компоненттері API арқылы бір-бірімен өзара жұмыс жасайды (15.2. сурет). Бұл ретте, әдетте, компоненттер иерархия құрады, яғни деңгейі жоғары компоненттер деңгейі төмен компоненттердің API-ін қолданады, ал олар болса, өз кезегінде, деңгейі бұдан да төмен компоненттердің API-ін қолданады. WindowsAPI арқылы жұмыс істеу – бұл қолданбалы бағдарламалардың ішінен онымен өзара жұмыс істеәтін операциялық жүйеге жақындау әдіс. Құрылғы драйверлері үшін ғана пайдалану деңгейінің төмен болғаны қажет.

*API қолданбалы бағдарламалау интерфейсі*— бұл қолданбалы бағдарламаларға берілетін көптеген жүйе функциялары мен жүйе қоңыраулары-жүктемелері (сервистері).

Қауіпсіздік үшін қосымша тікелей жүйелі қоңыраулар жасай алмайды, керісінше тоқтату механизмі пайдалануы тиіс, бұл кезде бағдарламадағы кідірісті процессорға генерациялайды, бұл өз кезегінде, ядро режимінде тоқтау жұмыстарының іске қосылуына алып келеді. Заманауи операциялық жүйелер тоқтау (Interrupt), алып тастау (Exception, Trap) әрекетін өңдейді және жүйедегі қоңыраулар (жүктемелерге) (Systemcall) қолдау көрсетеді.Қосымшаларға Windows арқылы мынадай түрлі API қызметтер қолжетімді:

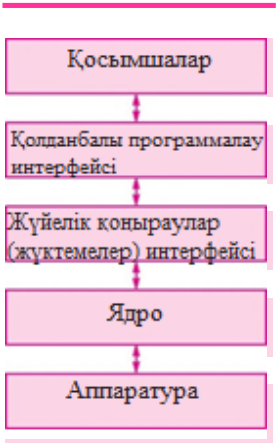
- базалық (BaseServices), оған процесстер мен жадыны басқару, енгізіп-шығару функциялары мен қауіпсіздік функциялары жатады;
- компоненттер (ComponentServices) — қосымшалардың өзара жұмыс жасауына арналған;
- пайдаланушы интерфейсі (UserInterfaceServices) — әр түрлі менюлермен және парақтармен (әйнектермен) жұмыс істеуге арналған;
- графиктер мен мультимедиа(Multimedia and Graphics Services);
- өзара хабарламалармен алмасу және бірігіп жұмыс істеу (MessagingandCollaboration);
- желі қызметтері (Networking);
- Web-қызметтер (Web Services).

Хабарлама Windows шеңберіндегі басты ұғым болып табылады.

Жүйе қосымшаға хабарлама жібереді, ал ол өз кезегінде, дұрыс әрекет етуі тиіс.

Жүйе қосымшадағы API қосымшаларын жасау кезінде бағдарламалауға уақыттың басым бөлігі кететін әйнек функциясы хабарлама алушылар болып табылады.

Стандартты API жалпы API қолданылатын барлық бағдарламалар бірдей жақсы немесе, тіпті болмағанда, әдеттегі типтік үлгіде жұмыс істейтініне кепілдік береді. API графикалық интерфейстер дегеніміз бағдарламаларында оған ұқсас пайдаланушы интерфейсі болады дегенді білдіреді, бұл жаңа бағдарлама өнімдерін меңгеру процестерін жеңілдетеді.




15.2-сурет. Қолданбалы бағдарламалау интерфейсі

WinAPI басынан бастап C (немесе C++) тілінде жазылған бағдарламаларда қолдану үшін жобаланған болатын және оның көптеген функциялар, Си тіліндегі деректер құрылымы мен сандық константтары бар. Барлық бағдарламалау тілдері Windows шеңберінде орындалатын бағдарламалардағы деректер типі сияқты функцияларды туындатуға қабілетті және осы API қолдана алады. Негізінен, операциялық жүйеге арналған бағдарламалау кез келген бағдарламалау тілінде (Delphi, C + + , Assembler және т.б.) жүзеге асырылады. Алайда, Си тілі көпшілікпен танылғандай, жүйені бағдарламалау тілі болып саналады және деңгейі жоғары тілдер ішінен аппаратураға жақындау тіл болып табылады. Жүйе бағдарламасындағы құжаттардың басым бөлігі (барлық құжатты — MSDN қосқанда) C + + тілі қолданылып жазылған.

Windows API – Windows базасындағы барлық қолданбалы бағдарламаларда қолданылады. Сол бір функция, әдетте, Windows Win64 — 64-разрядты Win32 нұсқалы қосымша функциядан тұратын 32 және 64 разрядты Windows-та болады.

Win API интерфейсін нақты бағдарламалау тілінен барынша тәуелсіз ету үшін немесе, мүмкін, компьютердің аппаратпен қамтамасыз етілуіне сай келтіру үшін осы интерфейсті жасаушылар жаңа қарапайым деректер типін айқындады. Бұл деректер типі WinAPI интерфейсі функцияларының прототиптерінде қолданылады.

## 15.3. ЕНГІЗУ-ШЫҒАРУДЫҢ ҚОСЫМША ЖҮЙЕСІ

*Енгізу-шығару іші жүйесі*— пайдаланушылар, қосымшалар және компьютердің шеткері құрылғысының арасында деректер алмасуды орындайтын операциялық жүйенің арнайы қосымша жүйесі. Осы міндеттерді орындау үшін қазіргі операциялық жүйелердің прототипінің қызметін атқарған алғашқы жүйелік бағдарламалар әзірленді. Сыртқы құрылғыларды басқаратын драйверлер және файлды жүйе енгізу-шығару қосымша жүйесінің компоненттері болып табылады.

Енгізу-шығару қосымша жүйесінің жұмысында үзілімдер диспетчері белсенді қызмет атқарады. Одан бөлек, үзілімдер диспетчерінің негізгі жүктемесі дәл осы енгізу-шығару қосымша жүйесіне негізделген, сондықтан үзілімдер диспетчерін кейде енгізу-шығару қосымша жүйесінің бөлігі деп есептейді.

Енгізу-шығару қосымша жүйесі келесі негізгі функцияларды орындайды:

- енгізу-шығару және процессор құрылғыларының параллельді жұмыс істеуін ұйымдастыру;
- алмасу жылдамдықтарын келістіру және деректерді кэштеу;
- құрылғылар мен деректерді процестердің (орындалатын бағдарламалардың) арасында бөлу;
- құрылғылар мен жүйенің қалған бөлімінің арасында ыңғайлы логикалық интерфейсті қамтамасыз ету;
- жүйеге жаңа драйверді қарапайым қосу мүмкіндігімен драйверлерді қолдау;
- операциялық жүйемен қосымша әрекеттерсіз драйверлерді серпінді жүктеу және түсіру (выгрузка?);
- бірнеше әртүрлі файлды жүйелерді қолдау;
- енгізу-шығарудың синхронды және асинхронды операцияларын қолдау.

Енгізу-шығару құрылғылары процестерге монополиялы және бөлінген режимдерде ұсынылуы мүмкін. Бұл ретте операциялық жүйе процестердің есептеу жүйесінің басқа ресурстарына қолжетімділігінде қолданылатын амалдармен (пайдаланушының немесе солардың атынан процесс әрекет ететін пайдаланушылар тобының құрылғыға белгілі бір операцияны орындау құқығын тексеру ) қолжеткізуді бақылауды қамтамасыз етуі тиіс. Операциялық жүйе жалпы құрылғыға ғана емес, осы құрылғы сақтайтын деректердің жеке үлесіне қол жеткізуді қадағалайды. Диск осындай құрылғының типтік үлгісі бола алады, мұнда файлдар және каталогтарға қол жеткізуді қадағалаудың маңызы зор. Соңғы жағдайда құрылғыны жалпы бірлесіп пайдалану режимін белгілеу міндетті болып табылады. Бірлесіп пайдалану жағдайында жалпы өнімділікті арттыру мақсатында операциялық жүйе әртүрлі процестер үшін енгізу-шығару операциялардың ретін оңтайландыруы тиіс.

Құрылғыларды процестердің арасында бөлу барысында аталған процестерді бір-бірінен ажырату қажеттілігі пайда болуы мүмкін. Әдетте, мұндай қажеттілік реттілік құрылғыларын бірлесіп пайдалануда пайда болады, реттілік құрылғылар тура қол жеткізу құрылғыларымен салыстырғанда адрестелмейді. Осындай құрылғының типтік өкілі— принтер. Осындай құрылғылар үшін шығаруға тапсырмалар кезегі ұйымдастырылады, бұл ретте әр тапсырма деректердің мөлшері болып табылады, оны бөлуге болмайды, мысалы, басып шығаруға арналған құжат.

Енгізу-шығару құрылғыларының әралуандығы шеткері құрылғылар мен қосымшалардың арасында логикалық интерфейсті құру бойынша операциялық жүйенің фунциясын өзекті етеді. Барлық дерлік операциялық жүйелер осындай интерфейс ретінде шеткері құрылғылардың файлды үлгісін қолдайды, яғни кез келген құрылғы қолданбалы бағдарламашы үшін байттардың ретті жиынтығы болып көрінеді, файл-құрылғының атын және байттардың басталу ретінен жылжуды белгілей отырып, бұл жиынтықпен сәйкестендірілген жүйелі шақырулармен (мысалы, read, write) жұмыс істеуге болады.

Файл-құрылғының қарапайымдылығы және кез келген типті құрылғылар үшін сәйкестендірілуі файл-құрылғы моделінің артықшылығы болып табылады. Осы модель кей құрылғының енгізу-шығару операцияларын бағдарламалауда базис ретінде қолданылады, онда енгізу-шығару ішкі жүйесі нақты типті құрылғының анағұрлым мазмұнды моделін құрады.

Енгізу-шығару операциясы операцияны сұраған бағдарламалық модульге қатысты синхрондық немесе асинхрондық режимдерінде орындала алады. *Синхрондық режим* бағдарламалық модуль өз жұмысын енгізу-шығару операциясы аяқталғанша уақытша тоқтатады. *Асинхрондық режимде* бағдарламалық модуль мультибағдарламалық режимде енгізу-шығару операциясымен қатар орындала алады.

## 15.4. ФАЙЛДАРДЫ БАСҚАРУ

Әртүрлі типтердің реттелген жиыны *логикалық жазба (құрылым)* деп аталады. Осы деректердің орналасу реті *жазбаның құрылымы* деп аталады. Қолданбалы бағдарлама деңгейінде файл көптеген логикалық жазбалар На уровне прикладной программы

жазбаларды ұсынады. Физикалық деңгейде файл көптеген секторлар немесе дискте сақталатын кластерлердің аттары аталатын көптік болып табылады. Әдетте логикалық жазбаның ұзындығы кластердің ұзындығымен сәйкес келмейтіндіктен, кластер бірнеше логикалық жазбалардан тұруы мүмкін, немесе керісінше, логикалық жазба бірнеше кластерлерден тұруы мүмкін. Файлдарға қол жеткізуді қамтамасыз ететін және файлдың логикалық жазбаларын және олардың физикалық түсінігін байланыстыруды орындайтын операциялық жүйенің бөлігін *файлдарды немесе файлды жүйені басқару жүйесі* деп аталады.

Файлдың логикалық жазбаларына қол жеткізу операцияларын орындау үшін әрбір файлды файлдың ағымдағы логикалық жазбасына сілтейтін файлмен байланыстырады.

Логикалық жазбаны жазу немесе оқудың әрбір операциясынан кейін файлды жүйе файлды сілтеушінің келесі логикалық жазбаға ығыстырады.

Файл жазбаларын уақытша сақтауға арналған жедел жад саласы *енгізу-шығару буфері* деп аталады. Әдетте буфердің ұзындығы кластердің еселі ұзындығына сәйес сақталады. Енгізу буферлері екі міндетті шешуге арналған:

1) қосымшада анықталатын файлдың логикалық жазбасы мөлшерінің арасындағы және дискке жазылатын кластердің мөлшерінің сәйкессіздігін жою;

2) сыртқы құрылғылардың процессор жұмысының жылдамдылығына ықпалын азайту, ол сыртқы құрылғылар жұмысының жылдамдығына едәуір жоғары.

Деректерді шығаруда осы міндеттерді шешу үшін файлды жүйе буферді толығымен логикалық жазбалармен толтыра бастйды, сосын сыртық құрылғыға деректерді дискке жазу пәрменін береді. Деректерді енгізгенде файлдарды басқару жүйесі бастапқыда буферді кластерлермен толтыра бастайды, сосын буферден пайдаланушының бағдарламасына жіберілген логикалық жазбаларды оқуды басқарады.

Деректерді енгізу-шығаруды жылдамдату үшін сақиналық кезекпен ұйымдастырылған бірнеше енгізу-шығару буферлері қолданылады. Бір буфермен пайдалану процесінің жұмысы барысында файлды жүйе параллельді түрде деректерді басқа буферлерге енгізу немесе шығаруды жүзеге асырады.

Деректерді енгізуді кэштеу төмендегіні көздейді: жүйе магниттік дисктің деректерін алдын алып оқуды орындайды, қосымшадан деректерді оқу үшін келесі пәрменді күтудің қажеті жоқ. Жазбалар қосымшамен рет-ретімен оқылатын жағдайда, бұл файл жазбаларын оқу уақытын қысқартады.

## 15.5. ОПЕРАЦИЯЛЫҚ ЖҮЙЕДЕГІ ОБЪЕКТІЛЕР

Операциялық жүйеде жүйелі ресурс болып табылатын деректер құрылымы *операциялық жүйедегі объект* деп аталады, мысалы, файл, канал, графикалық сурет.

Windows операциялық жүйесі қосымшаға объектілердің келесі санаттарын ұсынады:

- User — қосымшаның пайдаланушымен интерфейс үшін қолданылатын объектілер ;
- Graphics Device Interface — ақпаратты графикалық құрылғыларға шығару үшін қолданылатын объектілер;
- Kernel —Windows операциялық жүйесінің объектілері (мысалы, файлдер және каналдар).

*Ядро (Kernel)* — қосымшаларға компьютердің ресурстарына үйлестірілген қол жеткізуді қамтамасыз ететін операциялық жүйенің орталық бөлігі. Сонымен қатар, кәдімгі ядро файлды жүйе мен желілік хаттамалар сервистерін ұсынады. Жүйелі бағдарламалауды зерттеуде Kernel санатының объектілері ғана жете қарастырылады. Қалған екі санаттың объектілері графикалық интерфейстерді бағдарламалауды үйренгенде талданады.

*Объектілерге қол жетізу деп* қосымшаның объектіге кейбір функцияларды орындау мүмкіндігін айтамыз. Қосымшаның объектілерге тура қол жетізуі жоқ, ол оған жанама түрде қарайы. Ол үшін Windows операциялық жүйелерінде әрбір объектіге тиісті дескриптор (handle) қойылады.

*Объект дескрипторы* жазба болып табылады, оған жүйе қолдау жасайды және объектінің типін сәйкестендіру үшін объектінің адресін және объектінің түрін сәйкестендіруге арналған құралдарды қамтиды. Объектілердің дескрипторларын операциялық жүйе құрады және объектілерді құратын Win API функцияларымен қайтарылады. Сирек кездесетін ерекше жағдайларды ескермегенде, CreateObject түрінде бұл функциялар бар, мұнда Object сөзі нақты объектінің атымен ауыстырылады. Мысалы, CreateProcess функциясын шақыру арқылы құрылады. Әдетте, осындай функциялар құрылған объектінің дескрипторын қайтарады. Егер бұл мән NULL (немесе теріс мән) болса, онда объект табысты құрылғаны.

Объектімен жұмысты аяқтаған соң оның дескрипторын жабу керек, ол үшін келесі прототипі бар CloseHandle қолданылады

```
BOOL CloseHandle (
  HANDLE hObject      // объект дескрипторы
) ;
```

Табысты аяқтаған жағдайда CloseHandle нөлді емес мәнді қайтарады, кері жағдайда — FALSE. CloseHandle функциясы объектінің дескрипторын жояды, бірақ объектінің өзі үнемі жойылмайды. Себебі Windows-та бұрын құрылған объектіге қол жеткізу үшін басқа фунциялармен құрылған бірнеше дескрипторлар сілтеме жасайды. CloseHandle функциясы қай объектіге бір де бір дескриптор сілтеме жасама, оны жояды.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. Жүйелі бағдарламалауға қандай ерекшеліктер тән?
2. Компьютердің аппараттық ресурстарына нелер жатқызылады?
3. Пайдаланушының компьютер ресурстарымен өзара әрекеттеуі қалай өтеді?
4. Ресурстарды басқарудың ең маңызды ішкі жүйелерін атаңыздар?
5. Процестерді басқару жүйелерінің мақсаттары қандай
6. Файлдарды басқару ішкі жүйесінің мақсаты қандай?
7. Жадыны басқару ішкі жүйесінің мақсаты қандай?
8. Қолданбалы бағдарламалаушыларға қосымшаларда операциялық жүйеге жүгінулер қашан талап етіледі?
9. Қолданбалы бағдарламалау дегеніміз не?
10. Неге қосымшалар тікелей жүйелік шақыруларды жасай алмайды?
11. Қосымшаларға Windows API арқылы қолжетімді қызметтерді атаңдар.
12. Операциялық жүйемен өзара әрекеттесу үшін бағдарламалаудың қай тілдері жарамды?
13. Windows операциялық жүйесінің қосымшаға ұсынатын объектілердің санаттарын тізіп шығыңдар.
14. Операциялық жүйе ядросының функциялары қандай?
15. Операциялық жүйеде объект деп нені айтады?
16. Объектілерге қол жеткізу дегеніміз не?
17. Объект дескрипторы дегеніміз не?
18. Операциялық жүйеде енгізу-шығару ішкі жүйесі не үшін қолданылады?
19. Енгізу-шығару ішкі жүйесінің негізгі функцияларын атаңыздар.
20. Файл-құрылғы модельдері қалай қолданылады?
21. Логикалық жазба және жазба құрылымы дегеніміз не?
22. Енгізу-шығару буферлері неге арналған?
23. Деректерді енгізуді кэштеу дегеніміз не?

# 16 Тарау

# ПРОЦЕСТЕР МЕН АҒЫНДАРДЫ БАСҚАРУ

## 16.1. ПРОЦЕСТЕР

Процесс операциялық жүйелермен байланысты негізгі ұғымдардың бірі болып табылады. Компьютерде жұмыс істеп тұрған барлық бағдарламалық қамтамасыз етуді, оның ішінде, операциялық жүйені процестер жинағы ретінде ұсынуға болады.

*Процесс* деп компьютерде орындалатын қосымшаны және оны орындауға қажетті барлық ресурстарды айтамыз. Процесті орындауға қажетті барлық ресурстар *процестің мәнмәтіні* деп аталады.

Операциялық жүйенің міндеті – компьютердің процестері мен ресурстарын басқару, басқаша айтқанда, процестерді анағұрлым тиімді орындау мүддесінде ресурстарды тиімді қолдануды ұйымдастыру. Осы міндетті орындау үшін операциялық жүйе әрбір процесс пен ресурстың ағымдағы жай-күйі туралы ақпаратқа ие болуы тиіс. Осындай ақпаратты ұсынудың әмбебап тәсілі әрбір басқару объектісі бойынша ақпараты бар кестелерді құру және қолдау дегенді білдіреді. Компьютердің ресурстарын басқаруда ең көп қиындықтар мультибағдарламалық операциялық жүйелерде пайда болады.

*Мультибағдарламалау (көпміндеттілігі)* — бұл бір процессорда кезекпен бірнеше бағдарлама орындалатын есептеу процесін ұйымдастыру әдісі. Қысқаша айтсақ, мультибағдарламалау— бір процессорда бірнеше міндеттерді орындау.

Мультибағдарламалауды сақтау үшін операциялық жүйе өзі үшін жұмыстың ішкі бірліктерін анықтап алуы тиіс, бұл бірліктердің арасында процессор және компьютердің ресурстары бөлінеді. Процесс бір немесе бірнеше ағынның формасында орындалады.

Кейбір қазіргі операциялық жүйелерде тапсырма (Job) сияқты жұмыс бірлігі бар. Мысалы, Windows–та бірыңғай тұтастық ретінде басқарылатын бір немесе бірнеше процестердің жиынтығы *тапсырма* болып саналады.

Операциялық жүйе процестерді процесорлық уақыттан басқа ресурстардың барлық түрлері үшін өтінімдер немесе контейнерлер ретінде қарастырады. Операциялық жүйе бұл аса маңызды ресурсты басқа жұмыс бірліктерінің — ағындардың арасында бөледі, ағындар деп атаудың себебі – олар пәрмендерді орындаудың реттілігі (орындау ағындары) болып табылады.

Операциялық жүйеде әрбір процесс келесі ресурстарға ие:

- виртуалды адрестік кеңістік;
- шынайы жадыда беттердің жұмыс көптігі;
- қауіпсіздік жүйесі үшін ақпаратты қамтитын қол жеткізу маркері;
- ядро объектілері дескрипторларын сақтауға арналған кесте.

*Адрестік кеңістік* — бағдарламаны қосу үшін процеске берілген виртуалды жады. Әртүрлі процестердің адрестік кеңістіктері өзара қиылыспайды. Процестің басқа процестің адрестік кеңістігіне қол жетімділігі жоқ. Бұл процестердің біреуінде орын алған қателердің басқа процестердің орындалуына ықпалынан арылуға мүмкінді береді. Процесс контексінде орындалатын ағындар осы процеске тиесілі бір адрестік кеңістікте орындалады. Негізінде, процестердің адрестік кеңістіктерінің бөлінуі жүйелі бағдарламалауға ағын ұғымын енгізуге негізгі себебі болды. Бұл жағдайда параллельді процестердің арасындағы өзара әрекеттесу деректерді жіберуге үлкен шығындарды талап етеді, ол қосымшалардың жұмысын едәуір баяулатады. Ал ағындар бір процестің адрестік кеңістігінде орындалады және жадының ортақ адрестеріне жүгіне алады, бұл жайт олардың өзара ірекеттесуін жеңілдетеді.

*Мультипроцессорлық өңдеу (мультипроцестеу)* — бір есептеу жүйесіне кіретін бірнеше процессорда бірнеше міндетті бір уақытта орындау. Қазіргі таңда компьютердің сәулетінде бірнеше процессордың болуы үйреншікті құбылысқа айналды, ал серверлер ретінде пайдаланылатын компьютерлер үшін — бұл қажеттілік. Мультибағдарламалық өңдеуде процессорда әрбір уақытта бір ғана бағдарлама орындалады, ал бір мезетте бірнеше бағдарламаны өңдеу әртүрлі құрылғылардың параллельді жұмысының есебінен жүзеге асырылады. Міндеттерді процессорда параллельді өңдеу —көзге көріну ғана.

Мультипроцестеу жағдайында жүйеде іс жүзінде бірнеше міндеттер әртүрлі процессорларда орындалады. Бұл ретте әрбір процессор өзінің міндеттер жынтығын мультибағдарламалау режимінде орындай алады. Бұл жағдайда барлық ресурстарды басқару алгоритмдері күрделене түсетіні анық.

## 16.2. АҒЫНДАР

«Ағын» ұғымына процессордың бір пәрменнен екінші пәрменге реттілікпен ауысуы сәйкес келеді. Операциялық жүйенің процесіне адрестік кеңістік және ресурстар жиынтығын тағайындайды, бұл жиынтықты оның барлық ағындары бірлесіп қолданады. Бұл олардың бірдей жаһандық айнымалы шамаларды бөлетінін білдіреді. Әр ағын кез келген виртуалды адреске қолжетімді болғандықтан, бір ағын екінші ағынның стегін қолдана алады. Бір процестің ағындарының арасында толық қорғаныс жоқ. Өзара әрекеттесуді және деректермен алмасуды ұйымдастыру үшін ағындарға операциялық жүйеге жүгінудің қажеті жоқ, оларға ортақ жадыны қолдану жеткілікті: бір ағын деректерді жазып отырса, екіншісі оларды оқып отырады. Екінші жағынан, әртүрлі процестердің ағындары бір-бірінен жақсы қорғалған. Сонымен, мультибағдарламалау процестер деңгейінде емес, ағындар деңгейінде анағұрлым тиімді. Көпағынды өңдеудің одан да жоғары әсеріне мультипроцессорлық жүйелерде қол жеткізіледі, мұнда ағындар әртүрлі процессорларда іс жүзінде параллельді орындалады.

Әр ағынға келесі ресурстар тиесілі:

- орындалатын функцияның коды;
- процессор регистрларының жиынтығы;
- қосымшаның жұмысы үшін стек;
- операциялық жүйенің жұмысы үшін стек;
- қауіпсіздік жүйесі үшін ақпаратты қамтитын қал жету маркері.

Барлық осы ресурстар Windows-та ағын мәнмәтінін құрады. Windows операциялық жүйелерінде екі типті ағындарды ажыратады:

1) *жүйелік* — операциялық жүйенің әртүрлі сервистерін орындайды және операциялық жүйенің ядросымен іске қосылады;

2) *пайдаланушылық* — пайдаланушының міндеттерін шешуге арналған және қосымша арқылы іске қосылады.

Жұмыс істеп тұрған қосымшада жұмыс төбелері (working threads) және пайдаланушының интерфейсі ағындары (user interface threads) ажыратылады. Жұмыс ағындары қосымшада әртүрлі фондық міндеттерді орындайды.

Пайдаланушының интерфейс ағындары терезелермен байланысты және осы терезелерге түсетін хабарларды өңдейді.

Әр қосымшаның кем дегенде, бір ағыны бар, ол *бастапқы* (primary), немесе *басты* (main) ағын.

Дұрыс жүзеге асырылған жағдайда ағындардың процестерден бірқатар артықшылықтары бар:

- оларға жаңа ағын құру үшін азырақ уақыт керек, себебі құрылатын ағын ағымдағы процестің адрестік кеңістігін қолданады;
- ағынды аяқтау үшін азырақ уақыт жұмсалады;
- оларға процесс шегінде екі ағынды ажыратып-қосу үшін азырақ уақыт жұмсалады;
- оларға азырақ коммуникациялық шығын қажет, себебі ағындар барлық ресурстарды, оның ішінде адрестік кеңістікті бөледі.

Ағындардың бірімен өңделетін деректер тез арада барлық басқа ағындарға қолжетімді болады.

## 16.3. АҒЫНДАРДЫ ПАРАЛЛЕЛЬДІ ӨҢДЕУ МЕН ОРЫНДАУДЫ ЖОСПАРЛАУ

Бір немесе бірнеше процессорлары бар компьютерлер үшін қосымша әзірлеуде қосымша пайдаланушымен өзара әрекеттесуді, тіпті бұл қосымша қазіргі сәтте басқа амалдармен айналысып жатса да, қамтамасыз етуді талап етуі мүмкін. Қосымша басқа оқиғаларды өңдеу барысында қосымша мен пайдаланушының өзара әрекеттесуін қамтамасыз ететін әдістерінің бірі – орындаудың бірнеше ағынын пайдалану болып табылады.

Бірнеше ағынды параллель өңдеу бағдарламаның өнімділігін едәуір арттыра алады, бірақ бұл ретте бірнеше ағынды бақылау қажеттілігі пайда болады, ол қосымшаларды ретке келтіруді қиындатады. Бірнеше процессордың болуы компьютерге бір уақытта бірнеше міндеттерді орындауға мүмкіндік береді. Бірнеше ағынның болуы процеске жұмысты параллельді орындауға бөлуге мүмкіндік береді. Жұмыстың көпағынды режимі әзірлеушілер үшін жаңа мүмкіндіктерді ұсынады. Бірақ енді қосымша және ретке келтіруді жобалау процесі күрделене түседі. Негізгі қиындық — бір уақытта жұмыс істеп тұрған ағындарды синхрондау .

Синхрондау келесі жағдайларда қолданылады:

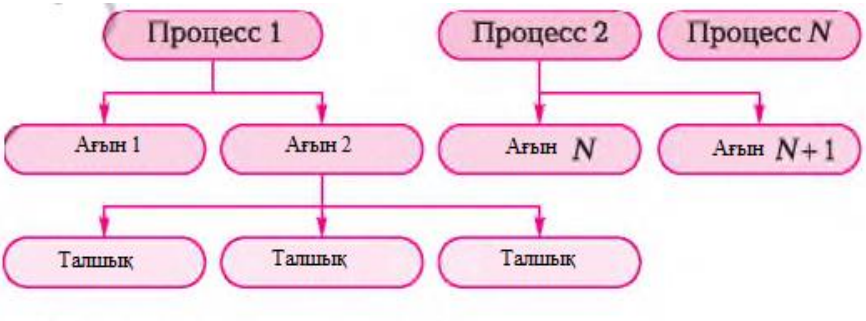
а) егер міндеттер белгілі бір ретпен орындалуы тиіс болса, кодты орындау тәртібін анық басқару үшін;

б) екі ағын бір уақытта бір ресурсты қолдануда пайда болатын мәселелердің алдын алу үшін.

Мысалы, басқа ағында жұмыс істеп жатқан деректер алу процедурасы аяқталмай жатып, ақпаратты шығару процедурасын уақытша тоқтату үшін синхрондауды пайдалануға болады. Бір ағын басқа ағынның нәтижелерін күтіп тұруы мүмкін, немесе бір ағынға екінші ағын пайдаланып жатқан ресурсқа монополиялы қол жеткізу керек болып қалуы ықтимал. Кейде ағын ешқашан қолжетімді болмайтын ресурсты күтіп, аяқталуы мүмкін. Синхрондау мәселелері көпағынды қосымшаларда кең тараған қателердің себебі болып табылады. Ол *өзара бұғаттаулық* деп аталатын жай-күймен аяқталады.

Ағындарды операциялық жүйеде ажыратып-қосу айтарлықтай көп уақыт алады, себебі ол үшін ядро режиміне ажыратып-қосу керек, содан кейін пайдалану режиміне оралу керек. Ағындарды жоспарлауға және диспетчерлеуге процессорлық уақыттың үлкен шығындары кетеді. Windows 2000 және кейінгі нұсқаларында өте қатты жеңілдетілген псевдопараллелизм ұсынылғанда ағындарға ұқсас, бірақ оларды бағдарламамен құрған пайдаланушының кеңістігінде жоспарланатын *талшықтар* (Fiber) қолданылады. Әр ағында бірнеше талшығы болуы мүмкін, мұндағы айырмашылық – талшық логикалық бұғатталғанда, ол бұғатталған талшықтар кезегіне қойылады, одан кейін жұмыс үшін тура сол ағын контексінде басқа талшық таңдалады. Бұл ретте тура сол ағын жұмысын жалғастырып тұрады. Операциялық жүйенің Windows –қа қатысты жұмыс бірліктерінің иерархиясы.16.1-сур. көрсетілген.

Жалпы жағдайда, ағынның орындалуы барысында қолжетімділігі бар жадының мазмұны *ағын мәнмәтіні* деп аталады.




16.1-сурет. ОЖ жұмыс бірліктерінің иерархиясы

Процессор жұмысының уақыты кванттарға (интервалдар) бөлінеді, олар ағындарға жұмыс істеу үшін беріледі. Уақыттың кванты аяқталған соң ағын үзіледі және процессор басқа ағынға тағайындалады.

Ағындардың арасында уақыт кванттарын бөліп берумен арнайы бағдарлама айналысады, ол *ағындар менеджері* деп аталады. Ағындар менеджері процессорды басқа ағынды орындауға қойғанда, ол келесі әрекеттерді орындауы тиіс:

- үзілмелі ағынның мәнмәтінін сақтау;
- іске қосылатын ағынды оны үзу сәтінде мәнмәтінін қалпына келтіру;
- басқаруды іске қосылатын ағынға тапсыру.

Ағын жұмысының әрбір сәтінде оның мәнмәтіні толығымен микропроцессор регистрлерінің құрамымен анықталады. Бұдан шығатын қорытынды, ағын мәнмәтінін сақтау үшін ағынның үзілу сәтінде микропроцессор регистрлерінің құрамын сақтау керек, ал ағын мәнмәтінін қалпына келтіруде осы регистрлердің құрамын қалпына келтіру керек. Ағындарға қызмет көрсету ережелері бәсекелес ағындарға компьютер ресурстарын бөлу тәртібін анықтайды. Синхрондауға қатысты екі тәсіл бар: сұрақ және синхрондау объектілерін пайдалану.

Егер барлық ағындардың басымдықтары бірдей болса, олар процессорге қызмет көрсетілуге кезекке тұрады. Процессор ағындарға «бірінші келді –бірінші қызмет көрсетілді» тәртібімен қызмет көрсетеді, үзілген ағындар кезектің соңына тұрады. Қызмет көрсетудің осы түрі *циклдік қызмет көрсету* деп аталады.

Егер ағындардың басымдықтары әртүрлі болса, онда оларды басқару үшін бірнеше кезегі бар қызмет көрсету пәндері қолданылады. Бірнеше кезекке қызмет көрсетудің ең қарапайым алгоритмі – ең жоғары басымдықтары бар ағындарға бірінші қызмет көрсетіледі.

Үзілген ағындарды кезекке қою үшін қолданылатын алгоритм, *жоспарлау* деп аталады. Ағындар менеджері (диспетчер) үзілген ағынның басымдылығын өзгерте алады, ол осы ағын жеткізілетін кезекті өзгертеді. Жоспарлау алгоритмдері математикалық пәнмен меңгертіледі, ол кестелер теориясы деп аталады.

Жоспарлаудың әртүрлі алгоритмдері бар:

- *ығыстырушы /ығыстырмайтын алгоритмдер*. Ығыстырушы алгоритм жағдайында операциялық жүйе уақыттың кез елген сәтінде ағып бара жатқан ағынды үзіп жіберіп, процессорды басқа ағынға қоса алады.

ығыстырмайтын алгоритмдерде процессор ұсынылған ағын басқаруды операциялық жүйеге қашан тапсыруды өзі ғана шешеді;

- *кванттау бар алгоритмдер.* Әрбір ағынға уақыт кванты беріледі, осы уақыт барысында ағын процессорда орындала алады. Квант біткеннен кейін операциялық жүйе процессорды кезектегі басқа ағынға қосады. Квант әдетте жүйелі таймер интервалдарының бүтін санына тең;
- *басымдықтары бар алгоритмдер.* Әрбір ағынға басымдылық (priority) тағайындалады — ағынның басымдылық дәрежесін білдіретін бүтін сан. Операциялық жүйе ағындарды дайындауға бірнеше дайын ағын болған жағдайда олардың ішінен басымдылығы ең жоғары ағынды дайындайды.

Windows-та жоспарлаудың аралас алгоритмі жүзеге асырылған — ығыстырушы, кванттау және басымдықтар негізінде.

Процессордың кезектерге қызмет көрсетудің тәртібін орнататын алгоритм *диспетчерлеу* деп аталады. Диспетчерлеу алгоритмдеріне математикалық пән қызмет көрсетеді, ол жаппай қызмет көрсету теориясы деп аталады.

Ағынның бірнеше жай-күйі бар:

- *дайындылық (ready)* — ағын орындауға дайын және процессорға қз кезегін күтіп тұр;
- *орындау(running)* — ағын процессорда орындалады;
- *күту (waiting)* —ағын орындала алмайды, себебі белгілі бір оқиғаның басталуын (мысалы, енгізу-шығару операциясының аяқталуын немесе басқа ағыннан хабарламаны) күтіп тұр.

Негізгілерден басқа тағы бірнеше жай-күйлер бар: инициализация (бастапқы жүктеме)
(init), аяқтау (terminate), тоқтап тұру (standby), ауыспалы жай-күй (transition), шегерілген дайындық жай-күйі (deferred ready).

Ағынның негізгі жай-күйлері, жай-күйлердің арасындағы ықтимал ауысулар және ауысу шарттары 16.2-сур. көрсетілген.

NET Framework бағдарламалық платформада қосымша домені деп аталатын механизм жүзеге асырылған, ол бір процесте қосымшалар тобын іске қосуға мүмкіндік беріп, олардың бір-бірінен салыстырмалы оқшауланын қамтамасыз етіп, жеке процесс жағдайынан гөрі бір-бірімен тезірек өзара әрекеттесуге мүмкіндік береді. Win API-да әрбір қосымша бір немесе бірнеше процестен тұра алады, олардың әрқайсысы, өз кезегінде, бір немесе бірнеше ағын тудыра алады, сөйтіп, басқарылмайтын код үшін оқшауланған ортаның рөлі айрықша процесс атқарады, ал NET-та оқшауланған ортаның рөлі қосымша домені атқарады.


16.2.-сурет. Ағынның жай-күйі

Домен оның шеңберінде қолданылатын ресурстарды тура сол процестің басқа домендерінен толығымен оқшаулайды.

Қосымшаның бір доменінде орындалатын қосымша басқа қосымша доменінің деректерін (атап айтқанда, жаһандық айнымалы шамалардың немесе статикалық өрістердің мәндері) ала алмайды, ол тек қана .NET қашықтан өзара әрекеттесу хаттамасының көмегімен ғана мүмкін болады.

. NET платформасы үшін қосымша домендері мен ағындардың арасында тура біржақты сәйкестілік көзделмеген. Керісінше, қосымша домені кез келген уақытта осы доменнің шеңберінде орындалатын көптеген ағындарға ие болуы мүмкін. Одан бөлек, нақты ағын ағынның бар болған барлық уақытында бір қосымша доменіне тіркелмеген. Ағындар Win API ағындары ережелеріне және мақсаттылығына CLR (Common Language Runtime — жалпы тілдік орындалу ортасы) бағына отырып, қосымша доменінің шекараларын кесіп өтуі мүмкін. Бірақ белсенді ағындар қосымша домендерінің шекаралары арқылы жылжыуы мүмкін, уақыттың кез келген нқты уақытында бір нақты ағын тек бір қосымша доменінің шеңберінде орындала алады (басқа сөздермен айтқанда, бір де бір ағын бірнеше қосымша домендерінде бір уақытта жұмыс істей алмайды).

NET Framework платформасы қосымша пайдаланушының әрекеттеріне жауап бере алатындай, ал пайдаланушы компьютерінің өнімділігі максималды болып қала беретіндей етіп, бірнеше орындау ағындарын қолданудың әртүрлі әдістерін ұсынады.

## 16.4. WINDOWS-та ПРОЦЕСТЕР МЕН АҒЫНДАРДЫ ҚҰРУ

Құру үрдісі өзіне аталған орындалатын үрдістің кодтары мен деректерді дисктен жедел жадыға жүктеуден тұрады. Ол үшін осы бағдарламаны диске салу керек, жедел жадыны қайта бөліп, жаңа үрдістің орындалатын бағдарламасына жады үлестіру керек. Одан бөлек, жұмыстың бағдарламасы барысында әдетте стек қолданылады, оның көмегімен процедураларды шақыру және параметрлерді жіберу жүзеге асырылады. Процестің бағдарламасы, деректері, стектары және атрибуттары кіретін көптік *процесс образы* деп аталады.

Windows-та жаңа процесс Create- Process функциясын шақырумен құрылады, бұл функцияның прототипі төмендегідей:

```
    BOOL CreateProcess(
      LPCTSTR lpApplicationName, // орындалатын
модульдің атауы LPTSTR lpCommandLine, // пәрмендік
жол
      LPSECURITY_ATTRIBUTES lpProcessAttributes, //
процесті қорғау
      LPSECURITY_ATTRIBUTES lpThreadAttributes, //
ағынды қорғау
      BOOL bInheritHandles, // дескрипторды мұра ету
белгісі
      DWORD dwCreationFlags, // процес құрудың
жалаулары LPVOID lpEnvironment, // жаңа қоршаған
орта блогы
      LPCTSTR lpCurrentDirectory, // ағымдағы
каталог
      LPSTARTUPINFO lpStartUpInfo, // бас терезенің
түрі

       LPPROCESS_INFORMATION lpProcessInformation //
процесс туралы ақпарат

        ) ;
```

CreateProcess функциясы процесс табысты құрылса, нөлдік емес мәнді қайтарады. Кері жағдайда бұл функция false мәнін қайтарады. Жаңа процесті құратын процесс құрылатын процеске қатысты *ата-ана процесі* (parent process) деп аталады. Басқа процеспен құрылатын жаңа процесс ата-ана-процесіне қатысты *туынды процессом* (child process) деп аталады

.

Бірінші параметр lpApplicationName орындалатын файлдың атымен жолды анықтайды, ол exe типті және жаңа процесті құрғанда іске қосылады. Бұл жол нөлмен аяқталуы және орындалатын файлға толық жолды қамтуы тиіс.

Процесс өзінің жұмысын ExitProcess функциясын шақырумен аяқтала алады. ExitProcess функциясын шақыруда процестің барлық кодтары қайталу кодымен аяқталады, ол осы функцияның параметрі болып табылады. Осы функцияны орындауда жүйе процесті жүктеген динамикалық кітапханаларға DLL_PROCESS_DETACH хабарламасын жібереді, ол динамикалық кітапхананы процестен ажырату керектігін білдіреді.

Жаңа ағын Windows-та CreateTh- read функциясын шақырумен құрылады, бұл функцияның прототипі төмендегідей:

```
HANDLE CreateThread(
  LPSECURITY_ATTRIBUTES lpThreadAttributes, //
қорғаныс атрибуттары
  DWORD dwStackSize,       //    ағын    стегінің
мөлшері, байт
  LPTHREAD_START_ROUTINE lpStartAddress, //
функция адресі
  LPVOID lpParameter,        // параметр адресі
  DWORD dwCreationFlags,   //     ағын     құрудың
жалаулары
  LPDWORD lpThreadId       ағын  сәйкестендіргіші)
  ;
```

Табысты аяқталуда CreateThread функциясы құрылған ағынның дескрипторын және оның бүкіл жүйе үшіе бірегей болып табылатын сәйкестендіргішін қайтарады. Кері жағдайда бұл функция NULL мәнін қайтарады.

lpThreadAttributes параметрі құрылатын ағынның қорғаныс атрибуттарын орнатады.

dwStackSize параметрі іске қосуда ағынға бөлінетін стек мөлшерін анықтайды. Егер бұл параметр нөлге тең болса, онда ағынға бөліне алатын стектің ең аз мөлшері бөлінеді.

lpStartAddress параметрі ағынның атқаратын функциясын көрсетеді.

dwCreationFlags параметрі ағын қандай жай-күйде құрылатынын анықтайды.

lpThreadId параметрі шығу параметрі болып табылады, яғни оның мәнін Windows орнатады. Бұл параметр Windows ағын сәйкестендіргішін орнататын айнымалы шаманы көрсетуі тиіс.

Бұл сәйкестендіргіш бүкіл жүйе үшін бірегей және әрі қарай ағынға сілтемелер жасау үшін қолданыла алады. Ағын сәйкестендіргіші көбінесе жүйелі функциялармен және сирек қосымша фукнцияларымен қолданылады. Ағын сәйкестендіргіші ағын бар кезде ғана жарамды. Ағын аяқталғаннан кейін тура сол сәйкестендіргіш басқа ағынға берілуі мүмкін.

Ағын Поток ExitThread функциясын шақырумен аяқталады. Бұл функция мән ағын функциясынан қайтқанда анық және анық емес шақырыла алады. Осы функцияны орындауда жүйе процеске жүктелген динамикалық кітапханаларға DLL_THREAD_DETACH хабарламасын жібереді, ол ағын өз жұмысын аяқтап жатыр дегенді білдіреді.

## 16.5. ҮРДІСТЕР АРАСЫНДА ДЕРЕКТЕРМЕН АЛМАСУ

Параллель үрдістер арасында деректермен алмасу деректерді бір ағыннан екіншісіне жіберуді білдіреді, бұл ағындар әртүрлі процестер контексінде орындалады деп көзделеді. Деректерді басқа ағынға жіберген ағын *жіберуші деп аталады.* Деректерді басқа ағыннан алатын ағын *адресат немесе алушы* деп аталады.

Егер ағындар бір үрдісте орындалса, олардың арасында деректермен алмасу үшін жаһандық айналмалы шамаларды және ағындарды синхрондау құралдарын қолдануға болады. Егер ағындар әртүрлі процестерде орындалса, онда ағындар ортақ айнымалы шамаларды қолдана алмайды, олардың арасында деректер алмасу үшін операциялық жүйенің арнайы құралдары бар. Аталған үрдістердің арасында деректермен алмасу үшін деректерді жіберу каналы құрылады.

Үрдістердің арасында деректерді жіберу каналын бағдарламалық ұйымдастыру үшін операциялық жүйе ядросының ағындарын және деректермен алмасу үшін қолданылатын ортақ жадыны файлмен ауыстыру керек. Үрдістердің арасында ортақ файл арқылы деректер алмасу процестердің арасында қарапайым деректер жіберу каналын ұйымдастыру болып табылады. Кейде операциялық жүйе деректер алмасуды жеделдету үшін бөлінетін файлға қолжетімділікті жеңілдете алады. Мысалы, Windows операциялық жүйелері осы мақсаттарда файлды процестің адрестік кеңістігіне проекциялайды.

Параллель процестерінің арасында деректермен алмасуда деректер жіберудің екі әдісін ажыратады:

1) *ағынмен* —байттардың үздіксіз реттілігімен;
2) *хабарламамен* —байттар тобымен.

Процестердің арасында деректерді жібермей тұрып, осы процестердің арасындағы байланысты анықтау керек. Процестердің арасындағы байланыс физикалық (немесе аппараттық) деңгейде де, логикалық (немесе бағдарламалық) деңгейде де орнатылады. Деректерді жіберу бағытының тұрғысынан байланыстың келесі түрлері ажыратылады: *жартыдуплекстік,* яғни бұл байланыс бойынша деректер бір бағытта ғана жіберіледі; *дуплекстік,* яғни т. е. бұл байланыс бойынша деректер екі бағытта да жіберіледі.

## 16.6. ХАБАРЛАМАЛАРДЫ ЖІБЕРУ

Процестердің арасында хабарламалармен алмасу екі функцияның көмегімен орындалады: send — хабарлама жіберу; receive — хабарлама алу.

Хабарлама екі бөлімнен тұрады: хабарламаның негізінен және атауынан тұрады. *Хабарлама негізі* хабарламаның өзінен тұрады. *Хабарламаның атауында* қызметтік ақпарат қамтылады:

- хабарламаның түрі;
- хабарлама адресатының аты;
- хабарламаны жіберушінің аты;
- хабарламаның ұзындығы (бақылау ақпараты).

Хабарламаларды жіберуде процестердің тура немесе жанама адрестеу қолданыла алады. Процестердің *тура адрестеуінде* send және receive функцияларында процестер, жіберуші және адресат анық көрсетіледі. Бұл жағдайда деректермен алмасу функциялары төмендегідей болады:

```
  send(Process P, хабарлама); Процеске P хабарлама
жіберу
   receive(Process Q, хабарлама); //
Q процестен хабарлама алу
```

*Жанама адрестеуде* send және receive функцияларында адрестер емес, хабарлама жіберілетін адрестер емес, байланыс түрі көрсетіледі. Бұл жағдайда деректермен алмасу функциялары төмендегідей болады:

```
    send(Connection S, хабарлама); // S байланысы
бойынша хабарлама жіберу
    receive(Connection R, хабарлама); //
R байланысы бойынша хабарлама алу
```

Үрдістерді адрестеу симметриялы және асимметриялы болуы мүмкін. Егер үрдістер арасында хабарламамен алмасуда

Тек қана тура немеес жанама адрестеу қолданылса, процестердің осындай адрестеуі *симметриялы деп аталады.* Егер процестер арасында хабарлама алмасуда тура адрестеу де, жанама адрестеу де қолданылатын болса, процестердің осындай адрестеуі *асимметриялы* деп аталады. Процестердің асимметриялы адрестеуі «клиент-сервер» жүйелерінде қолданылады. Бұл жағдайда клиенттер сервердің адресін біледі және келесі функцияны қолдана отырып, хабарлама жібереді:

```
Send (Process Сервер, хабарлама);
```

Сервер байланыс каналын «тыңдайды» және функцияны қолдана отырып:

```
receive(Connection S, хабарлама);
```

барлық клиенттерді қабылдайды

Байланыстар орнату және үрдістердің арасында деректер жіберу бойынша ережелердің жиынтығы *хаттама* деп аталады. Деректер жіберуді бағдарламалау тұрғысынан хаттама ережелері процедуралар немесе деректерді жіберу үшін қолданылатын функциялардың жиынтығынан тұрады, және осы функцияларды қолдану тәртібі, яғни функцияларды қолданғанда олард шақырудың белгілі бір ретін ескеру керек. Мысалы, жіберуші мен адрестаттың арасында байланыс орнатпай, деректерді жіберуді бастауға болмайды.

*Буфером* деп процестердің арасындағы байланыстың сыйымдылығын айтады, яғни осы байланыс бойынша бір уақытта жіберіле алатын хабарламалардың саны.

## 16.7. АНОНИМДІ ЖӘНЕ АТАУЛЫ КАНАЛДАР

*Анонимді канал* деп бір компьютерде орындалатын процестердің арасында деректер жіберуді қамтамасыз ететін операциялық жүйе ядросының объектісі аталады. Анонимді канал құратын процесс *анонимді каналдың сервері* деп аталады. Анонимді каналмен байланысатын процестер *анонимді каналдың клиенттері* деп аталады. Басқа сөзбен айтқанда, анонимді канал — процестердің арасында деректер жіберудің аты жоқ каналы. Демек, осындай каналға ата-аналар процесс-сервері және осы каналдың туынды процестері-клиенттерінің ғана қолжетімділігі бар. Параллель процестердің арасында деректер алмасу үшін қолданғанда анонимді каналдардың келесі сипаттамаларын ажыратады:

- аты жоқ;

- жартыдуплекстік;
- деректерді ағынмен жіберу;
- синхрондық деректермен алмасу;
- кбайланыстың кез келген топологиясын модельдеу мүмкіндігі.

Анонимді канал бойынша деректерді тек қана бір бағытта жіберуге болады. Бірақ анонимді каналдың әрбір ұшының өзінің дескрипторы бар, оны кез келген туынды процеске жібере алады. Сондықтан әрбір процесс деректерді анонимді каналға жаза алады және деректерді сол жақтан оқи алады. Анонимді каналдармен жұмыс істеу тәртібі төмендегідей:

1) сервермен анонимді канал құру;
2) клиенттерді каналмен байланыстыру;
3) канал бойынша деректермен алмасу;
4) каналды жабу.

Анонимді каналдардың аты жоқ болғандықтан, процесс-клиентті осындай каналмен байланыстыру үшін оған анонимді каналдың дескрипторларының бірін жіберу керек. Бұл ретте жіберілетін дескриптор мұралы болуы тиіс, ал процесс-клиент анонимді каналдың процесс-серверінің туынды процесі болуы және процесс-сервердің мұралы дескрипторларын мұралыққа иеленуі тиіс.

Процесс-клиент анонимді каналының дескрипторларын мұралыққа иеленуі үшін ол CreateProcess функциясымен анонимді каналдың процесс-серверінде құрылуы тиіс және осы фунцияның bInheritHandles параметрі TRUE-да орнатылуы шарт.

Анонимді каналдар стандартты енгізу-шығаруды қайта бағыттау үшін жиі қолданылады.

*Атаулы канал* деп бір жергілікті тораптағы компьютерлерде орындалатын үрдістердің арасында деректерді жіберуді қамтамасыз ететін операциялық жүйе ядросының объекті аталады. Атаулы канал құрушыны - *атаулы каналдың сервері* деп аталады. Атаулы каналмен байланысатын процестер *атаулы каналдың клиенттері* деп аталады. Атаулы каналдардың сипаттамалары төмендегідей:

- олар жартыдуплексті де, дуплексті де бола алады;
- деректерді жіберу ағынмен де, хабарламалармен де жүзеге асырыла алады;
- деректермен алмасу синхронды да, асинхронды да бола алады;
- байланыстардың кез келген топологиясымен модельдеу мүмкіндігі бар. Енді әрі қарай қолданылатын атаулы каналдармен жұмыс істеу тәртібін келтіреміз:

1) атаулы каналды сервермен құру:

2) серверді атаулы каналдың данасымен байланыстыру;
3) клиентті атаулы каналдың данасымен байланыстыру;
4) атаулы канал бойынша деректермен алмасу;
5) серверді атаулы каналдың данасынан ажырату;
6) атаулы каналды клиентпен және сервермен жабу. Атаулы каналдар процесс-сервермен CreateNamedPipe функциясының көмегімен құрылады:

```
HANDLE CreateNamedPipe (:
    LPCTSTR lpName,              // каналдың атауы
    DWORD dwOpenMode,            // каналдың
атрибуттары
    DWORD dwPipeMode,      // деректерді жіберу
режимі
    DWORD nMaxInstances,  // канал данасының
максималды саны
DWORD  nOutBufferSize // буфердің шығу мөлшері
    DWORD nInBufferSize //буфердің кіру  мөлшері
    DWORD nDefaultTimeOut, // клиентпен
```

Табысты аяқталған жағдайда CreateNamedPipe функциясы атаулы каналдың дескрипторын қайтарады.

Сервер атаулы каналды құрғаннан кейін, ол клиенттің осы каналмен байланыстырылғанын күтуі тиіс. Ол үшін сервер ConnectNamedPipe функциясын шақырады, оның прототипі төмендегідей:

```
    BOOL ConnectNamedPipe (
      HANDLE hNamedPipe,          //       каналдың
дескрипторы
      LPOVERLAPPED   lpOverlapped   //   асинхронды
байланыс) ;
```

Табысты аяқталу жағдайында бұл функция нөлдік емес мәнді, ал сәтсіздік жағдайында —FALSE мәнін қайтарады. Сервер бұл функцияны клиентпен байланыс үшінатаулы каналдың әрбір бос данасы бойынша қолданады. Клиентпен деректер алмасып біткеннен кейін сервер DisconnectNamedPipe функциясын шақыра алады, оның келесі прототипі бар:

```
BOOL DisconnectNamedPipe (
  HANDLE hNamedPipe // каналдың дескрипторы
  ) ;
```

Бұл функция сервердің клиентпен байланысын үзеді. Содан кейін клиент осы атаулы канал бойынша сервермен деректер алмаса алмайды, сондықтан осы атаулы каналға қол жеткізу бойынша клиенттің тарапынан кез келген операциясы қате тудырады. Бір клиентпен байланысты үзгеннен кейін сервер дәл осы атаулы канал бойынша басқа клиентпен байланыс орнату үшін қайтадан ConnectNamedPipe функциясын шақыра алады,.

Анонимді каналмен сияқты, атаулы канал бойынша деректер алмасу үшін ReadFile и WriteFile функциялары қолданылады.

## 16.8. СОКЕТТЕРДІ ЖЕЛІЛІК БАҒДАРЛАМАЛАУ

Атаулы каналдар бір жүйеде орындалатын процестер жағдайында да, бір-бірімен жергілікті немесе жаһандық желімен байланыстырылған компьютерлерде орындалатын процестер жағдайында да процесаралық өзара әрекеттесуді ұйымдастыру үшін жарамды. Бірақ атаулы каналдардың кемшілігі – олар кәсіпорындық стандарт болып табылмайды. Басқа жүйелермен өзара әрекеттесу мүмкіндігі Windows-та сокеттердің (sockets) қолдауымен қамтамасыз етіледі.

*Сокеттер* — бұл үрдістердің арасында деректер алмасуды қамтамасыз етуге арналған бағдарламалық интерфейс. Үрдістер мұндай алмасуда бір ЕЭМ және өзара желімен байланысқан әртүрлі ЕЭМ-мен де орындала алады. Сокет — желілік коммуникациялардың ақырғы нүктесі болып табылатын абстрактілі объект. Әрбір қолданыстағы сокеттің типі және сонымен ассоцияланған процесі бар.

Желідегі TCP/IP компьютерлердің әрқайсысының өзінің бірегей IP-адресі бар, ол басқа компьютерлермен деректер алмасу үшін қолданылады. Бір компьютерден екіншісіне жіберілетін әрбір пакетте жіберушінің және алушының адресі бар, ол оны біржақты сәйкестендіруге мүмкіндік береді. Алайда, компьютерде бір уақытта желіні қолданатын көптеген қосымшалар жұмыс істеп тұрғанда, мұндай атрибуттар жинағының жеткіліксіз екендігі анық. Бірмәнді еместікті шеш үшін әрбір қосылыстың әр ұшында адрестен басқа бүтін сан ретінде «порт» деп аталатын сәйкестендіргіш бар. Сонымен, адрес және порт жұбы сокет-канал болып табылады, ол бойынша екі компьютер бір-бірімен деректер алмасады. Бір компьютерде бір уақытта бір ғана қосымша нақты портті қолдана алады, бірақ серверлік бөлімдер үшін бірнеше клиентпен жұмыс істеу үшін бір портта бірнеше сокеттер құру мүмкіндігі бар.

Порттың мәні сервер мен клиенттікімен міндетті түрде сәйкес келуі шарт емес — клиенте байланысқа түсу үшін сервердің портын білу ғана маңызды, клиенттің портын клиент өз бетімен таңдай алады және серверге клиент қосылуға сұрау салғанда белгілі болады. Қосылыс орнатылғаннан кейін, операциялық жүйе серверлік қосымша үшін тиісті соетті құрады, қосымша сонымен жұмыс істей береді, демек, клиенттің порты серверге мүлдем маңызды емес.

Сокеттер жұмысының механизмі келесідей: серверлік жақта серверлік сокет іске қосылады, ол іске қосыла салысымен тыңдау режиміне (клменттердің қосылуын күту) ауысады. Тыңдау процесі әдетте күту циклінде болады, яғни жаңа қосылыс пайда болғанда оянады. Бұл ретте қосылыстардың қазіргі сәтте бар болуын тексеру, операция үшін тайм аут орнату мүмкіндігі және т.б. сақталады. Әдетте клиент тыңдаушыға анық қосылады, содан кейін кез келген оқу немесе жазба оның файлдық дескрипторы арқылы оның және сервердің арасындағы деректерді жібереді.

Клиенттің жағында сокет құрылады, ол үшін IP-адрес және сервер порт көрсетіліп, қосуға пәрмен беріледі. Сервер қосылуға сұрауды алғаннан кейін операциялық жүйе сокеттің жаңа данасын құрады, оның көмегімен сервер клиентппен дерек алмаса алады. Бұл ретте тыңдау үшін құрылған сокет қосылыстарды қабылдау режимінде қала береді, сөйтіп, бағдарламашы клиенттерден бірнеше қосылыстарымен жұмыс істейтін сервер құра алады.

Сокеттермен жұмыс дегеніміз ол синхронды және асинхронды болатын енгізу-шығару операциялары. Сокеттер терминологиясында асинхронды режимдегі жұмыс *бұғаттайтын сокеттер,* ал синхронды режимдегі жұмыс — *бұғаттамайтын сокеттер* деп аталады. Бұғаттау режимінде қосылу немесе деректер қабылдау талпынысы (жіберу әрқашан синхронды, себебі іс жүзінде кезекке қою болып табылады) бағдарлама қосылып, деректерді қабылдамаса, келесі операторға басқаруды тапсыру мүмкін емес дегенді білдіреді.

TCP/IP хаттамасының негізінде желіде қосымшаларды жүзеге асыру үшін құрылған бағдарламалау интерфейсі *Winsock API* деп аталады. Winsock API Win32/64-тің бөлігі болып табылмайды. Одан бөлек, Winsock стандарттарға бағынбайтын қосымша функцияларды ұсынады, бұл функциялар аса қажеттілік жағдайында ғана пайдаланылады.

## 16.9. DLL ДИНАМИКАЛЫ ҚОСЫЛАТЫН КІТАПХАНАЛАР

*Динамикалы қосылатын кітапханалар* (Dynamic Link Library — DLL) — код, деректер немесе ресурстары бар модульдер, олар бірлесіп Windows-тың бірнеше қосымшаларымен қолданылуы мүмкін. Windows қолданбалы бағдарламалық интерфейс DLL динамикалық қосылатын кітапханалар жиынтығы ретінде жүзеге асырылады. DLL-ға сонымен қатар, ActiveX басқару элементтері және драйверлер жатқызылады. UNIX жүйелерінде ұқсас функцияларды жалпы объектілер (shared objects) орындайды. DLL түрінде ресімделген бағдарламалық модуль дискте файл түрінде сақталады, кеңеюі dll, функцияларды да, деректерді де қамти алады.

DLL үрдістің виртуалды жадысына процестің орындалып жатқан модулін құру кезінде де (статикалық), операциялық жүйе процесті орындап жатқанда да (динамикалық) жүктеле алады. DLL файлдарының форматы келісімдерді орындалатын файлдардың форматы сияқты бірдей ұстанады. DLL-ді жадыға жүктеу үшін файлдарды жадыға кейіптеу в механизмі қолданылады. Файлды жадыға кейіптеу — бұл кейбір операциялық жүйелерде файлдармен жұмыс істеу әдісі, тұтас файл немесе осы файлдың үзілмейтін бөлігінде жадының белгілі бір учаскесі қойылады (жедел жады адрестерінің диапазоны). Бұл ретте осы адрестерден деректерді оқу кейіптелетін файлдан деректерді оқуға алып келеді, ал осы адрестер бойынша деректерді жазу осы деректерді файлға жазуға алып келеді. Жадыға кәдімгі файлдарды ғана емес, құрылғылардың файлдарын да кейіптеуге болады. Динамикалық қосылатын кітапханалар бір кітапхананы пайдаланып жатқан бірнеше қосымшаның жұмысы барысында қолданылатын физикалық жадының көлемін азайтуға мүмкіндік береді. Осыған DLL-ді үрдістертің виртуалды жадысына проекциялау механизмінің нәтижесінде қол жеткізіледі, себебі бұл жағдайда барлық қосымшалар физикалық жадыға жүктелген орындалатын DLL кодының бір данасына ортақтасады. Бірнеше қосымша бір уақытта бір функционалды мүмкіндіктерді қолданып жатқанда DLL жадының өнімсіз шығынын азайтуға көмектеседі, себебі әр қосымша өзінің деректер көшірмесін алса да, олар бәрібір бірлесіп кодты қолдана алады.

DLL кітапханалары қолданылмай Windows қосымшасын құру мүмкін емес дерлік.

DLL — бұл кітапханаларға жиналған функциялардың жиынтықтары. Бірақ өздерінің статикалық туыстарымен (файлдар . lib) салыстырғанда, DLL кітапханалары тікелей орындалатын файлдарға байланыстар редакторының көмегімен қосылмаған. Орындалатын файлға олардың орналасуы туралы ақпарат қана енгізілген. Бағдарламаны орындау кезінде кітапхана тұтас жүктеледі. Соның арқасында әртүрлі процестер жадыда орналасқан тура сол кітапханаларды пайдалана алады. Бұл тәсілкөптеген ортақ кітапханаларды қолданатын бірнеше қосымшаға қажет жадының көлемін қысқартуға, сонымен қатар, EXE-файлдардың өлшемдерін қадағалауға мүмкіндік береді. Бірақ егер кітапхана бір ғана қосымшаны қолданса, оны кәдімгі, статикалық істеген дұрыс. Әрине, оның құрамына кіретін функциялар бір ғана бағдарламада қолданылса, оған бастапқы мәтінімен тиісті файлды енгізу жеткілікті.

## 16.10. СЕРВИСТЕР

*Сервис* — бұл қызметтік функцияларды атқаратын процесс. Сервис – ол операциялық жүйе жүктелгенде немесе оның жұмысы барысында арнайы пәрменмен іске қосылатын және өзінің жұмысын операциялық жүйенің жұмысы тоқтатылғанда немесе арнайы пәрменмен жұмысын аяқтайтын бағдарлама деп айтуға болады. Әдетте сервистер қосымшалардың немесе нақты бір қосымшаның жұмысына қажетті қызметтік функцияларды орындайды. Мәліметтер базасына қолжетімділікті қамтамасыз ететін сервер-процесс сервистің мысалы бола алады. Сервистің басқа түріне драйверлер — сыртқы құрылғыларға қолжетімділікті қамтамасыз ететін бағдарламалар жатқызылады. Монитор деп аталатын қосымшаның жұмысын бақылайтын процесс те сервис ретінде жүзеге асырылады.

Сервистердің жұмысын сервистердің менеджері(Service Control Manager — SCM) деп аталатын операциялық жүйенің арнайы бағдарламасы басқарады. Бұл бағдарлама келесі функцияларды атқарады:

- орнатылған сервистердің мәліметтер базасын қолдау;
- операциялық жүйелерді жүктеуде сервистердің іске қосылуы;
- жұмыс істеп тұрған сервистердің жай-күйі туралы ақпаратты қолдау;
- жұмыс істеп тұрған сервистерге басқару сауал жіберу;
- сервистердің мәліметтер базасын бұғаттау және бұғаттан шығару.

Сервистер менеджері белгілі бір сервистің бар екендігін білу үшін, оны орнату керек.

Барлық орнатылған сервистер туралы ақпарат Windows операциялық жүйесінің реестрінде сақталған. Сервис операциялық жүйемен жүктеуде де, қосымшадан бағдарламалық негізде де іске қосылады. Егер сервис енді керек болмаса, оны операциялық жүйенің мәліметтер базасынан алып тастау керек. Windows операциялық жүйелерінде сервистермен жұмыс істеу үшін Win API арнайы функциялар бар.

Сервистер сервис менеджерлерінің қол астында жұмыс істейді, сондықтан олар сервистің интерфейсін анықтайтын белгілі бір келісімдерге сәйкес келулері тиіс. Сервистің өзі консольді қосымша да, графикалық интерфейсі бар қосымша да бола алады. Бұның маңызы жоқ, себебі сервистер пайдаланушымен тек жұмыс үстелі немесе хабарламалар терезесі арқылы ғана өзара әрекеттесе алады, сондықтан әдетте сервистер консольді қосымшалар ретінде ресімделеді. Сондай-ақ, әр сервистің операциялық жүйемен шақырылатын екі кері шақыру функциясы болуы тиіс.

## 16.11. ВИРТУАЛДЫ ЖАДЫ. ҮРДІС ТЕРГЕ ЖАДЫНЫҢ БӨЛІНУІ

Windows операциялық жүйесінде барлық процестерге аса маңызды ресурс — виртуалды жады ұсынылады. Дәл виртуалды жадыда процестер тікелей жұмыс істейтін барлық деректер сақталады.

Процеске жадының нақты (физикалық) орналасуы белгісіз — бұл жады жедел-сақтау құрылғысында да, дискте де орналасуы мүмкін, сондықтан атауы да — «виртуалды жады». Операциялық жүйе процеске белгілі бір көлемдегі виртуалды адрестік кеңістікті ұсынады, және процесс ұяшықтармен осы кеңістіктің кез келген виртуалды адресінде жұмыс істей алады, деректер нақты қай жерде сақталды екен деп мазаланбайды. Виртуалды жадының көлемі теориялық тұрғыдан алғанда, операциялық жүйенің разрядтылығымен шектеледі. Практикада операциялық жүйені нақты жүзеге асыруда шектеулер теориялық шектеуден төмен қойылады.

Виртуалды жадыны енгізудің келесі артықшылықтары бар:

а) қолданбалы бағдарламашыларға деректерді жадыда нақты орналасуы сияқты күрделі мәселелермен айналыспауға мүмкіндік береді;

б) операциялық жүйеге бірнеше процесті бір уақыта іске қосуға мүмкіндік береді, себебі қымбат, шектеулі ресурс — жедел жадының орнына арзан және сыйымдылығы үлкен сыртқы жады қолданылады.

Виртуалды жады бірдей өлшемді блоктарға— виртуалды парақтарға бөлінеді. В Windows-та үлкен парақтар (x86 — 4 Мбайт, x64 — 2 Мбайт) және қосымша (4 Кбайт) парақтар болады. Физикалық жады (жедел сақтау құрылғысы запоминающее устройство) виртуалды жады сияқты өлшемді парақтарға бөлінеді.

Әдетте процесстер виртуалды жадының барлық көлемін емес, шағын ғана бөлігін қолданады. Тиісінше барлық процестердің әрбір виртуалды парағы үшін физикалық жадыда парақты белгілеудің еш маңызы жоқ. Оның орнына жедел сақтау құрылғысында процеске тікелей қажет парақтардың шектеулі саны бар. Физикалық жадыда орналасқан виртуалды парақтардың ішкі жиынтығы *процестің жұмыс жиынтығы* деп аталады. Процеске әзірше керек емес виртуалды парақтарды операциялық жүйе дискке, *басқылау файлы* деп аталатын арнайы файлға орната алады. Процес дәл қазір қажет парақтың қайда екендігін анықтау үшін, деректердің арнайы құрылымдары— парақтардың кестелері (парақтарға көрсеткішітер массивтері) қолданылады.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. Үрдіс дегеніміз не?
2. Мультибағдарламалау дегеніміз не?
3. Әр үрдістің операциялық жүйедегі ресурстарын тізіп шығыңыз.
4. Адрестік кеңістік деген не?
5. Процестердің өзара әрекеттесуіне қарағанда, неге ағындардың өзара әрекеттесуі қарапайым?
6. Мультипроцессорлық өңдеу дегеніміз не?
7. Ағын дегеніміз не?
8. Неге мультибағдарламалау ағындар деңгейінде тиімдірек?
9. Ағынға тиесілі ресурстарды атаңыз.
10. Жүйелі және пайдаланушылық ағындардың бір-бірінен қандай айырмашылығы бар?
11. Ағындардың процестердің алдында қандай артықшылықтары бар?
12. Ағындарды синхрондау қандай жағдайларда кездеседі?
13. Талшықтар дегеніміз не?
14. Ағын мәнмәтіні дегеніміз не?
15. Ағындар менеджері қандай әрекеттерді орындайды?

16. Бірнеше ағындарды параллельді өңдеудің қандай артықшылықтары бар?
17. Ағындарды жоспарлау дегеніміз не?
18. Жоспарлаудің қандай алгоритмдері бар?
19. Ағын қандай жай-күйлерде болуы мүмкін?
20. Жаңа процесс қандай функциямен құрылады? Оның параметрлерінің тағайындалуын сипаттаңыз.
21. Параллель процестерінің арасындағы деректер алмасу нені көздейді?
22. Үрдістер арасындағы деректер ортақ файл арқылы қалай алмасады?
23. Жартыдуплексті және дуплексті байланыстың қандай айырмашылықтары бар ?
24. Процестер арасында хабарламалармен алмасу қандай функциялардың көмегімен орындалады?
25. Анонимді канал дегеніміз не?
26. Анонимді каналдармен жұмыс жасау ретін сипаттаңыз
27. Атаулы канал дегеніміз не?
28. Атаулы канал қандай функциямен құрылады?
29. Сокет дегеніміз не?
30. Сокеттер жұмысының механизмін сипаттаңыз.
31. Қандай модульдер динамикалық қосылған кітапханалар деп аталады?
32. Сервис дегеніміз не?
33. «Виртуалды жадыға» анықтама беріңіз. Виртуалды жады физикалық жадыдан несімен ерекшеленеді?

# 17 тарау

# КОНСОЛДЫ ҚОСЫМШАЛАРДЫ БАҒДАРЛАМАЛАУ

## 17.1. КОНСОЛДЫ ҚОСЫМШАНЫҢ ҚҰРЫЛЫМЫ

*Консоль* деп мәтіндік ақпаратты енгізу-шығару үшін қосымшамен пайдаланылатын интерфейс аталады. Бұл жағдайда, консоль пайдаланушылармен ақпарат алмасу үшін пайдаланатын қосымша консолды қосымша деп аталады.

Windows-тың консолды қосымшалары сыртқы көрінісі бойынша Windows-та іске қосылған DOS қосымшаларына ұқсайды. Бұған қарамастан бұл қосымшалар DOS-пен жұмыс істемейді. Ұқсас болып көрінгенімен айырмашылығы айтарлықтай. MS DOS — бұл принциптік бір міндетті жүйе, онда іске қосылған бағдарлама барлық ресурстарды басқарады. Мұндай бағдарламаның өзегі жиі repeat... until сияқты шексіз цикл болып табылады: бағдарлама қандай да бір жалаудың болуын тексереді (пернеге басылуын, портқа деректердің келіп түсуін және т.б.) және егер болса, онда белгіленген әрекеттерді орындайды. Windows қосымшалары өзгеше құрылу керек: мұнда әрекет оқиғамен басталады. Кез-келген енгізуді операциялық жүйеден алады, ол енгізуді қосымшаларға жібереді.

Консолды қосымшалар графикалық қосымшаның жұмысында туындайтын, жойылмайтын қателіктерді өңдеу және әртүрлі сервисті әзірлеуүшін жүйелік бағдарламалауда қолданылады. Консоль бір кіру буферден және бір немесе бірнеше экран буферінен тұрады. Кіру буфер енгізу оқиғалары туралы ақпаратты құрайды. Әрбір енгізу оқиғасы жазумен сипатталады. Барлық жазбалар енгізу буферінде сақталатын кезекке тұрғызылған. Экран буфері қосымшаның терезесіне шығарылатын ақпаратты құрайды және таңбалар мен түсі туралы деректерді құрайтын екі өлшемді массив болып табылады.

Бағдарламалық деңгейде ақпаратты енгізу және шығару үшін консолды бағдарламалар стандартты енгізу-шығару құрылғыларын пайдаланады, алайда өзге файлдарды да, желілік қосылыстарды да аша алады және оларды орындайтын ортада қолжетімді өзге әрекеттерді де жасай алады. Консольды бағдарламалар аса озық операциялық жүйелер үшін, әсіресе UNIX үшін, әдетте, интерфейсті пайдаланушымен іске асырудың барынша кең класында жұмыс істеуге қабілетті. Консоль мәтіндік ақпаратты енгізудің-шығарудың екі деңгейін қамтамасыз етеді: жоғары және төмен. *Жоғарғы деңгей функциялары* қалған оқиғаларды ескермей, таңбалардың консольге енгізілуін-шығарылуын қамтамасыз етеді. *Төменгі деңгей функциялары* консолды қосымшаға байланысты барлық оқианың өңделуін қамтамасыз етеді.

Консолды бағдарлама стандартты құрылғыларға енгізу-шығарумен, кітапханалар немесе өзге бағдарламалық интерфейстердің пайдаланылуымен шектеле отырып, пайдаланушымен өзара әрекеттесуді іске асырудың қамын ойламауы мүмкін. Әдетте пайдаланушымен өзара әрекеттесуді операциялық жүйе немесе өзге бағдарламалық қамтамасыз ету жүзеге асырады.

Консольдің кіру буфері енгізу оқиғаларын сипаттайтын, жазбалар кезегін құрайды. Енгізу оқиғалары мына санаттарға бөлінеді: пернетақтадан енгізу, тінтуірмен енгізу, терезе көлемін өзгерту.

## 17.2. КОНСОЛЬМЕН ЖҰМЫС ІСТЕУ

Процесс тек бір консольмен байланысты болуы мүмкін. Жаңа консоль келесі тәсілдермен құрылады:

1. CreateProcess командасымен консольді процесті құру кезінде CREATE_NEW_CONSOLE жалауын орнату керек. Бұл жағдайда, егер консольді процесс консолды қосымшадан құрылса, ал аталған жалау орнатылмаса, онда жаңа процесс ата-аналық процестің консольіне қосылады.
2. AllocConsole функциясы пайдаланылады, оның мынадай прототипі бар:

```
BOOL AllocConsole(VOID);
```

Егер консоль сәтті құрылса, бұл функция нөлдік емес мәнді қайтарады, олай болмаған жағдайда — `false`.

Қосымша FreeConsole функциясын шақыру арқылы консольді босатады, оның мынадай прототипі бар:

```
BOOL FreeConsole(VOID);
```

Сәтті аяқталған жағдайда бұл функция нөлдік емес мәнді қайтарады, ал олай болмаған жағдайда – `false`.

Жүйе жаңа консольді құру кезінд үш дескрипторды құрады, олар STDIN, STDOUT, STDERR болып белгіленеді және енгізу, шығару және қателіктің стандартты дескрипторлары деп аталады. STDIN дескрипторы енгізу буферімен байланысады, ал STDOUT және STDERR дескрипторлары экран буферімен байланысады. Бұл дескрипторлар консольмен жұмыс істеуге арналған функцияларда пайдаланылады.

Стандартты енгізу-шығару дескрипторларының мәндерін GetStdHandle функциясын пайдалана отырып, алуға болады.

Консоль терезесінің дескрипторын `GetConsoleWindow` функциясын шақыру арқылы алуға болады, оның мынадай прототипі бар:

```
HWND GetConsoleWindow(VOID);
```

Сәтті өткен жағдайда, бұл функция консоль терезесінің дескрипторын қайтарады, ал сәтсіздік жағдайында — NULL. Сәтсіздік қосымшада консольдің жоқтығын білдіреді.

## 17.3. ЭКРАН БУФЕРІМЕН ЖҰМЫС ІСТЕУ

Экран буфері CreateConsoleScreenBuffer функциясын шақыру арқылы құрылуы мүмкін. Сәтті аяқталған жағдайда бұл функция экранның жаңа буферінің дескрипторын қайтарады. Құрылғаннан кейін экран буферінде бос орындар болады, ал меңзер (0, 0) координаттары бар позицияға орнатылған. Экран буферін активті ету үшін, яғни экранға шығару үшін SetConsoleActiveScreenBuffer функциясын шақыру керек. Сәтті аяқталған жағдайда бұл функция нөлдік емес мәнді қайтарады, ал сәтсіздік жағдайында — false мәні. Осы функцияның жалғыз параметрі экран буферінің дескрипторы болып табылады.

Меңзердің қалыбы мен көрінерлігі туралы ақпаратты GetConsoleCursorInfo функциясын пайдалана отырып алуға болады.

Консоль атрибуттарын орнату үшін FillConsoleOutputAttribute функциясы пайдаланылады:

```
BOOL FillConsoleOutputAttribute (

    HANDLE hConsoleOutput, //экран буферінің
дескрипторы
```

```
        WORD    wAttributes, //фонның түсі және
        мәтіннің түсі
        DWORD nLength, //толтырылатын
торлардың саны
        COORD   dwWriteCoord,   //бірінші   тордың
        координаттары
        LPDWORD lpNumberOfAttrsWritten
//толтырылған торлардың саны
     ) ;
```

Сәтті аяқталған жағдайда нөлдік емес мән қайтарылады, ал сәтсіздік жағдайында — FALSE мәні қайтарылады. Саны nLength параметрімен белгіленген, Fill- ConsoleOutputAttribute функциясы және wAttribute параметрімен белгіленген атрибуттар экран торларын толтырады. dwWriteCoord параметрі бірінші толтырылатын тордың координаталарын белгілейді, ал lpNumberOfAttrsWritten параметрі DWORD типті ауыспалыны көрсету керек, оған FillConsoleOutputAttribute функциясы толтырылған торлар санын орналастырады. Іс жүзінде FillConsoleOutputAttribute функциясы консольдің тік бұрышты аяларын жаңа атрибуттармен толтыру үшін пайдаланылады.

Экран консолынің кіру буферінен таңбалар жолын оқу үшін ReadConsole функциясы пайдаланылады:

```
     BOOL ReadConsole(
       HANDLE hConsoleInput, // экран буферінің
дескрипторы
       LPVOID   lpBuffer,//таңбалар енгізуге арналған
       массив
       DWORD    nNumberOfCharsToRead, //оқылатын
       таңбалардың саны
       LPDWORD lpNumberOfCharsRead,// оқылған
таңбалардың саны
       LPVOID lpReserved //резервіленген)
```

ReadConsole функциясы таңбаларды бірінен кейін бірі ретімен енгізеді, бұл ретте меңзер келесі бос позицияға жылжиды. Егер өздігінен қосылып тұрған, енгізілген таңбаларды көрсету режимі қосылған болса, онда енгізу кезінде таңбалар экранда көрсетіледі.

Экран буферіне таңбалар жолын жазу үшін WriteConsole функциясы пайдаланылады:

```
BOOL WriteConsole (
  HANDLE hConsoleOutput, //экран
буферінің дескрипторы
  CONST VOID *lpBuffer, //шығаруға
арналған таңбалармен массив
  DWORD nNumberOfCharsToWrite,//жазылатын
таңбалардың саны
  LPDWORD lpNumberOfCharsWritten,//жазылған
таңбалардың саны
  LPVOID lpReserved
//резервіленген
) ;
```

Осы функцияның параметрлерінің тағайындалуы айқын.

## 17.4. КОНСОЛЬГЕ ЕНГІЗУ-ШЫҒАРУ

Жоғарғы деңгейдің енгізу-шығару функцияларына WriteConsole, ReadConsole, WriteFile және ReadFile функциялары жатады.

Экран консольінің кіру буферінен таңбалар жолын оқу үшін ReadConsole функциясы пайдаланылады, оның мынадай прототипі бар:

```
BOOL ReadConsole (
  HANDLE hConsoleInput,// экран
буферінің дескрипторы
  LPVOID  lpBuffer,//таңбаларды      енгізуге
  арналған массив
           DWORD
          nNumberOfCharsToRead,//оқылатын
 таңбалардың саны
  LPDWORD lpNumberOfCharsRead, //оқылған
таңбалардың саны
  LPVOID lpReserved //резервіленген) ;
```

Сәтті аяқталған жағдайда бұл функция нөлдік емес мәнді, ал сәтсіздік жағдайында — FALSE мәнін қайтарады.

Таңбалар жолын экран буферіне жазу үшін WriteConsole функциясы пайдаланылады, оның мынадай прототипі бар:

```
BOOL WriteConsole (
  HANDLE hConsoleOutput,        //      экран
  буферінің дескрипторы
```

```
      CONST VOID *lpBuffer,//шығаруға арналған
таңбалармен массив
        DWORD    nNumberOfCharsToWrite,//жазылатын
        таңбалардың саны
        LPDWORD lpNumberOfCharsWritten, //жазылған
таңбалардың саны
        LPVOID   lpReserved//резервіленген
      ) ;
```

Сәтті аяқталған жағдайда бұл функция нөлдік емес мәнді, ал сәтсіздік жағдайында — FALSE мәнін қайтарады.

Төменгі деңгейдің енгізу функциясы тікелей консольдің кіру буферінің жазбаларымен жұмыс істейді. Кіру буферден жазбаларды оқу үшін ReadConsoleInput функциясы пайдаланылады, оның мынадай прототипі бар:

```
      BOOL ReadConsoleInput (
        HANDLE   hConsoleInput,//      консольдің      кіру
буферінің дескрипторы
        PINPUT RECORD lpBuffer, // деректер буфері
        DWORD      nLength, оқылатын жазбалардың саны
        LPDWORD  lpNumberOfEventsRead  оқылған
жазбалардың саны
      ) ;
```

Сәтті аяқталған жағдайда бұл функция нөлдік емес мәнді, ал сәтсіздік жағдайында — `FALSE` мәнін қайтарады. Консольдің кіру буферінен жазбаларды оқығаннан кейін ReadConsoleInput функциясы оны одан жоятынын атап өткен жөн.

`hConsoleInput` параметрі консольдің кіру буферінің дескриторын құрауы керек.

`lpBuffer` параметрі консольдің кіру буферінен жазбалар оқылатын, жад аясына көрсету керек.

`nLength` параметрі пайдаланушы консольдің кіру буферінен оқығысы келетін, жазбалар санын құрауы керек. Ал `lpNumberOfEventsRead` параметрімен белгіленген мекен-жай бойынша функция консоль буферінен оқылған жазбалар санын жазады.

Консольдің кіру буферінен жазбаларды жоймайқ оқу үшін `PeekConsoleInput` функциясы пайдаланылады. Осы функцияның параметрлері ReadConsoleInput функциясының параметрлерімен толық сәйкес келеді.

Төменгі деңгейдің шығару функциясы тікелей экран буферінің элементтерімен жұмыс істейді.

`WriteConsoleInput` функциясы консольдің кіру буферіне енгізу оқиғаларын жазуға арналған. Бұл функцияның мынадай прототипі бар:

```
BOOL WriteConsoleInput (
    HANDLE  hConsoleInput,//консольдің кіру буферінің дескрипторы
    CONST INPUT_RECORD *lpBuffer, // жазбалары бар буферінің көрсеткіші
    DWORD      nLength, // жазылатын жазбалардың саны
    LPDWORD lpNumberOfEventsWritten // жазылған жазбалардың саны) ;
```

Сәтті аяқталған жағдайда бұл функция нөлдік емес мәнді, ал сәтсіздік жағдайында — `FALSE` мәнін қайтарады.

## БАҚЫЛАУ СҰРАҚТАРЫ

1. Консоль дегеніміз не?
2. Жаңа консоль қандай тәсілдермен құрылады?
3. Қай функцияны шақыру арқылы экран буфері құрылады?
4. Меңзердің қалыбы мен көрінуі туралы ақпаратты қалай алуға болатынын түсіндіріңіз.
5. Консоль атрибуттарын орнату үшін қандай функция пайдаланылады?
6. Экран консолінің кіру буферінен таңбалардың жолдарын оқу үшін пайдаланылатын функцияны атаңыз.
7. Экран буферіне таңбалар жолын жазу үшін қандай функция пайдаланылады?
8. Жоғарғы деңгейдің енгізу-шығару функцияларына жататын функцияларды атаңыз.
9. Қандай функция экран консольінің кіру буферінен таңбалар жолын оқуға арналған?
10. Экран буферіне таңбалардың жолдарын жазу үшін пайдаланылатын функцияны атаңыз.
11. Кіру буферден жазбаларды оқу үшін қандай функция пайдаланылады?
12. Қандай функция консольдің кіру буферіне енгізу оқиғаларын жазуға арналған?

# Терминдер сөздігі

**Процестің атаулы кеңістігі** — бұл бағдарламаларды іске қосу үшін процеске берілген виртуалды жады.

**Альфа-тестілеу** - бұл әлеуетті пайдаланушылардың немесе тапсырыс берушілердің бағдарламалық қамтамасыз етумен нақты жұмысы не әзірлеушілердің нақты жұмысты еліктеуі.

**Бета-тестілеу** - бұл жоспарланған функциялардың толық жинағы бар бағдарламалық қамтамасыз етудің шамамен дайын толық нұсқасын қарқынды пайдалану.

**Браузер** - бұл Web-беттің әртүрлі құрауыштарын өңдеуге және шығаруға, Web-сайт пен оның келушілерінің арасындағы интерфейсті ұсынуға арналған кешенді қосымша.

**Веб-сервер (Web-сервер)** — бұл клиенттерден, әдетте Web-браузерлерден HTTP-сұрау салуды қабылдайтын және оларға HTML-бетпен, суретпен, файлмен, медиа-ағынмен немесе өзге деректермен бірге HTTP-жауаптарды беретін сервер.

**Веб-сайт (Web-сайт)** — бұл жалпы мекен-жайы бар (доменді аты немесе IP-мекен-жайымен) компьютерлік желідегі жеке тұлға немесе ұйымның электронды құжаттар (деректер мен код файлдары) жүйесі.

**Верификациялау** — бұл берілген функцияларды орындаудың дұрыстығын және бағдарламалық қамтамасыз етудің тапсырыс берушінің талаптарына, сондай-ақ берілген ерекшеліктерге сәйкес келуін тексеру.

**Визуалды бағдарламалау** - бұл мәтінді жазудың орнына графикалық объекттермен айла-амал жасау жолымен бағдарламаны құру тәсілі.

**Шығудағы тестілеу** - бұл тапсырыс берушіге/пайдаланушыларға жеткізу үшін бағдарламалық қамтамасыз етудің дайындығын тексеру мақсатында тестілеу.

**Гипермәтін** — бұл ол кезінде жекелеген ақпараттық элементтер қажетті ақпаратты тез іздеуді және/немесе өзара байланысты деректерді қарауды қамтамасыз ететін қауымдастықты қатынастармен байланысты ақпараттық массивтерді ұйымдастыру қағидаты.

**Динамикалық қосылатын кітапханалар (Dynamic Link Library, DLL)** — бұл Windows-тың бірнеше қосымшасымен бірге пайдаланылуы мүмкін, кодты, деректерді немесе ресурстарды құрайтын модульдер.

**Диспетчеризациялау** — бұл процессор кезектерге қызмет көрсететін тәртіпті белгілейтін алгоритм.

**Домен (domain)** — бұл қандай да бір елге, ұйымға және т.б. берілген, Интернеттің домен аттарының (DNS) жүйесіндегі белгілі аймақ.

**Драйвер** — бұл сыртқы құрылғының нақты моделін басқаратын және оның барлық ерекшеліктерін есепке алатын бағдарлама.

**Бағдарламалық қамтамасыз етудің тіршілік циклі** - бұл бағдарламалық қамтамасыз етуді құру туралы шешім қабылдау сәтінен басталатын және оны пайдаланудан толық алу сәтінде аяқталатын үздіксіз процесс.

**Қапшықтану** — бұл осы объектілердің деректерін және сыртқы ортадан объектіні максималды деңгейде оқшаулауға мүмкіндік беретін, бұл деректерді өңдеу алгоритмдерін біртұтас біріктіру.

**БҚ инкрементті әзірлеу** — бұл уақыт кестесімен жүретін, кезеңді стратегия, онда жүйенің әр бөлігі әртүрлі уақытта және түрлі қарқынмен әзірленеді, егер бір бөлігі дайын болса, онда оны жүйеге кіріктіріледі.

**Аспаптық бағдарламалық қамтамасыз ету** - бұл бағдарламаларды жобалау, әзірлеу және сүйемелдеу барысында пайдалануға арналған бағдарламалық қамтамасыз ету.

**Біріктірілген тестілеу** — бұл бүкіл жүйенің біртұтас ретінде тестілеуінің алдындағы жеке модульдердің бірлескен жұмысын тексеруге арналған бағдарламаны тестілеу.

**Интернет (Internet, «желілер желісі»)** — бұл бірыңғай басқару орталығы жоқ, бірақ бірыңғай ереже бойынша жұмыс істейтін және өз пайдаланушыларына бірыңғай қызметтер жинағын ұсынатын, жаһандық ақпараттық желі.

**Интернет-мекенжайы (IP-мекенжайы, Internet Protocol Address)** — бұл IP хаттамасы бойынша құрылған, компьютерлік желідегі тораптың бірегей желілік мекенжайы.

**Объекттің интерфейсі** — бұл ол қоршаған әлеммен қалай әрекет ете алуының сипатын білдіретін, объектті-бағдарланған бағдарламалау объектінің сипаттамасы.

**API қолданбалы бағдарламалаудың интерфейсі** - бұл қолданбалы бағдарламаларға ұсынылатын, көптеген жүйелік функциялар мен жүйелік шақырулар (сервистер).

**БҚ сынау**- бұл шынайы ортада бағдарламаны орындау кезінде қателерді табу әрекеті.

**Бағдарламалық қамтамасыз етудің сапасы** - бұл бағдарламалық қамтамасыз етудің пайдаланушының белгіленген қажеттіліктерін қанағаттандыру қабілетін айқындайтын, оның сипаттамаларының жиынтығы.

**БҚ әзірлеудің каскадты стратегиясы** - бұл әзірлеу кезеңдерінің бір жолғы өтуі, стратегия әзірлеу процесінің басында әзірленетін бағдарламалық құралға қойылатын барлық талаптарды толық анықтауға негізделген. Әзірлеудің әрбір кезеңі алдыңғы кезең аяқталғаннан кейін басталады. Әлдеқашан орындалған кезеңдерге қайту көзделмеген. Әзірлеудің аралық өнімдері бағдарламалық өнімнің нұсқасы ретінде таралмайды.

**БҚ коммуникативтілігі** - бұл бағдарламалық қамтамасыз етудің басқа бағдарламалармен максималды мүмкін біігуі, дерекқорының экспорты/импорты, өңдеу объектілерін енгізу немесе байланыстыру және т.б.

**Процестің мәнмәтіні** - бұл процесті орындау үшін қажет барлық ресурстар.

**Конфигурациялық тестілеу** - бұл өзгерістер жүйесінің конфигурациядағы өнімділігіне әсерін тестілеу.

**1 с конфигурациясы (қолданбалы шешім)** - бұл белгілі бір қолданбалы міндеттерді шешуге арналған ақпаратты өңдеу объектілерінің, қасиеттерінің, әдістерінің сондай-ақ алгоритмдерінің нақты жиыны.

**Жергіліктендіру** - бұл оны орындау есептеу процесінің бұзылуын тудырған бағдарламаның операторын/операторларын анықтау.

**БҚ мобильдігі** - бұл бағдарламалық өнімнің деректерді өңдеу жүйесінің техникалық кешенінен, опеарциялық ортадан, деректерді өңдеудің желілік технологиясынан, пәндік саланың ерекшелігінен және т.б. тәуелсіздігі.

**БҚ өзгеруі** - бұл бағдарламалық қамтамасыз етудің өзгерістер енгізу, мысалы, өңдеу функцияларын кеңейту, өзге техникалық өңдеу базасына өту және т.б. қабілеті

**Модульді бағдарламалау** - бұл бүкіл бағдарлама модульдер деп аталатын компоненттер тобына бөлінетін, сол уақытта әрбіреуінің бақыланатын өз көлемі, нақты мақсаты және сыртқы ортасы бар толық өңделген интерфейсі бар, бағдарламалау тәсілі.

**Модульді тестілеу** - бұл жекелеп алынған модульдер, функциялар немесе кластар деңгейінде бағдарламаны тестілеу.

**Мультибағдарламалау (көп міндеттілік)** - бұл бір процессорда кезекпен бірнеше бағдарлама орындалатын, есептеу процесін ұйымдастыру тәсілі.

**Мультипроцессорлық өңдеу (мультипроцессорлеу)** - бұл бір есептеу жүйесіне кіретін, бірнеше процессорда бірнеше міндетті бір уақытта орындау.

**БҚ сенімділігі** — бұл бағдарламалық қамтамасыз етудің жоғары ықтималдық дәрежесімен белгіленген уақыт кезеңі ішінде белгіленген жағдайда белгіленген функцияларды бас тартпай орындау қабілеті.

**Мұралану** - бұл әлдеқайда бар (ата-аналық) клас негізінде жаңа класты сипаттау мүмкіндігін беретін механизм, бұл ретте ата-аналық кластың қасиеттері мен функционалдығы жаңа класпен пайдаланылады.

**Жүктеп тестілеу** - бұл өнімділігін анықтау және осы жүйеге қойылатын талаптарға сәйкес келуін анықтау мақсатымен, бағдарламалық қамтамасыз етудің сыртқы сұрау салуға жауап беру уақытының көрсеткіштерін жинау.

**Процестің түрі** - бұл онда процестің бағдарламасы, деректері, стектері мен атрибуттары кіретін жиын.

**1С жүйесіндегі конфигурациялау жүйесі –** бұл ұқсас сипаттамалары бар және бірдей бағытталған ұғымдар тобының (пәндік саланың, пайдаланушының жүйемен өзара әрекеттесу құралдарының) формалды сипаты.

**Операциялық жүйедегі объект** — бұл жүйелі ресурсты, мысалы, файлды, арнаны, графикалық суретті білдіретін, деректер құрылымы.

**Ретке келтіру** - бұл бағдарламалық қамтамасыз етуді тестілеу кезінде анықталған, жергіліктендіру және қателіктерді түзету процесі.

**Ашық жүйе** - бұл жалпыға қолжетімді және жалпыға бірдей қабылданған халықаралық стандарттарға сәйкес әзірленген, аппараттық және бағдарламалық өнімдер мен технологиялардан тұратын қандай да бір есептеуіш орта.

**Ағындарды жоспарлау** - бұл кезектегі үзілген ағындарды қою үшін пайдаланылатын алгоритм.

**1с платформасы** - конфигурацияны басқаруға арналған аспаптар мен тілдік құралдар жинағы. Қолданбалы шешімді құру, өзгерту және оның шын мәнінде қызмет етуі платформаның технологиялары мен механизмдерін пайдаланусыз мүмкін емес.

**Енгізу-шығарудың қосымша жүйесі** - бұл компьютердің пайдаланушыларының, қосымшалары мен перифериялық құрылғыларының арасында деректерді алмастыруды орындайтын операциялық жүйенің арнайы қосымша жүйесі.

**Полиморфизм** - бұл әртүрлі класқа кіретін әдістер үшін бірдей аттарды пайдалану мүмкіндігі (яғни бірдей ерекшелігі бар объектілердің әртүрлі іске асыру мүмкіндігі).

**Ағын** - бұл процессордың бір командадан басқасына жүйелі өтуі. Операциялық жүйенің процесіне атаулы кеңістік және ол оның барлық ағындарымен пайдаланылатын ресурстар жинағы тағайындалады.

**Қабылдау тестілеу** - бұл инсталляция, бағдарламалық жүйені сүйемелдеу және соңғы пайдаланушыны оқыту үшін жауап беретін ұйыммен жүргізілетін тестілеу.

**Қолданбалы бағдарламалық қамтамасыз ету** - бұл белгілі пайдаланушылық міндеттерді орындауға арналған бағдарламалар кешені.

**Қолданбалы бағдарламалау** - бұл қолданбалы бағдарламалық қамтамасыз етуді әзірлеу процесі.

**Бағдарламалық қамтамасыз ету (БҚ)** - бұл компьютерде кез-келген қызмет саласындағы әртүрлі мақсатты міндеттерді шешу мүмкіндігін беретін, сонымен қатар ЭЕМ аппараттық құралдарының жұмысын қамтамасыз ететін бағдарламалық құралдардың және оларды сүйемелдейтін құжаттамалардың жиынтығы.

**Бағдарламалық модуль** – бұл белгілі бір мақсаты бар және белгіленген өңдеу функцияларын басқа бағдарламалық модульдерден дербес қамтамасыз ететін, бағдарламаның жеке бөлігі.

**Хаттамалар** - бұл бір деңгейді білдіретін, бірақ әртүрлі желі торабында орналасқан, желілік компоненттермен алмасатын, хабарламалардың тәртібі мен форматын белгілейтін, нысандандырылған ережелер.

**ПҚ прототипі** - бұл жеңіл өзгеретін және кеңейтілетін және толық іске

**Процесс** — бұл оны орындау үшін қажет барлық ресурстармен бірге компьютерде орындалатын қосымша.

**Кемімелік тестілеу** — бастапқы кодтың тестіленген учаскелерінде қателерді табуға бағытталған бағдарламалық қамтамасыз етуді тестілеу.

**Объектінің қасиеті** — бұл деректердің және оларды оқу мен жазу әдістерінің жиынтығы.

**World Wide Web қызметі (WWW, «ғаламтор»)** — бұл желіге қосылған кез-келген серверде ақпаратқа қолжетімділікті алуға мүмкіндік беретін, Интернет желісінің негізгі қызметі.

**Желілік интерфейс** — бұл бір торапта көршілес деңгейлерде жатқан желілік компонеттер алмасатын хабарламалардың бірізділігі және форматы.

**Жүйелі бағдарламалық қамтамасыз ету** — бұл олар компьютерлік жүйенің компонеттерін (процессорды, жедел жадын, енгізу-шығару құрылғыларын, желілік жабдықтарды және т.б.) басқаруды қамтамасыз ететін бағдарламалар кешені.

**Жүйелік бағдарламалау** — жүйелі бағдарламалық қамтамасыз етудің әзірлемесі. .

**Жүйелік тестілеу** — бүкіл бағдарламалық жүйені біртұтас ретінде тестілеу.

**Сокеттер** — процестердің арасында деректермен алмасуды қамтамасыз етуге арналған бағдарламалық интерфейс.

**Сүйемелдеу** — бұл туындаған проблемалармен немесе оның негізгі функцияларын өзгеріссіз сақтаған кезде өзгерту қажеттіліктерімен пайда болған, жеткізілетін бағдарламалық қамтамасыз етудің жаңа жағдайларға бейімделу, бағдарламалық қамтамасыз етуге өзгерістер және тиісті құжаттаманы енгізу процесі.

**БҚ сүйемелдеу** — оған өзгерістер енгізу бойынша күшті минималдандыруға мүмкіндік беретін, бағдарламалық қамтамасыз етудің сипаттамасы: пайдаланушылардың өзгеретін қажеттіліктеріне сәйкес қателерді жою үшін нем есе өзгерту үшін.

**Ерекшелік** — бұл бағдарламалық және аппараттық компоненттердің, олардың қызмет ету, басқа компоненттермен өзара әрекет ету тәсілдерінің, пайдалану талаптарының, айрықша сипаттамаларының нысандандырылған сипаты.

**Стилі** — бұл HTML құжатының сыртқы түрін белгілейтін параметрлердің жиыны, оны браузердің терезесінде көрсеткен кезде: әртүрлі деңгейдегі тақырыптардың қаріптері мен түсін, азат жолдың тегінде берілетін негізгі мәтіннің қарпін және т.б..

**Стресс-тестілеу** — бағдарламалық қамтамасыз етуді тестілеудің бір түрі, ол жүйенің қалыпты қызмет ету шегін арту жағдайларындағы сенімділік пен тұрақтылықты бағалайды.

**CSS стильдерінің кестесі** — бұл Web-құжатта HTML тегтерін форматтауды басқаратын үлгі.

**БҚ тестілеу** — бұл бағдарламалық қамтамасыз етудің оның жұмысын арнаулы, жасанды құралған ахуалдарда бақылау көмегімен

**БҚ тұрақтылығын тестілеу** — бұл бағдарламалық қамтамасыз етудің күтілетін жүктеме деңгейімен ұзақ уақыт тестілеу кезінде жұмыс қабілеттілігін тексеру.

**Техникалық тапсырма** — бұл бағдарламалық қамтамасыз етуге қойылатын талаптардың жиынтығы бар бағдарламалық құжат, ол тексеру және әзірленген бағдарламаны қабылдау критерийі ретінде пайдаланылуы мүмкін.

**БҚ қателерге беріктігі** — бұл кіріс деректерінің кез келген жинағының жағдаларында бағдарламалық өнім жұмысының болжамдығы немесе жабдықтармен өзара әрекет еткен кезде қателердің болуы.

**БҚ функционалдылығы** — бұл бағдарламалық қамтамасыз етудің оның сыртқы ерекшеліктерімен белгіленген функциялардың жиынын орындау қабеліттілігі.

**БҚ әзірлеудің эволюциялық стратегиясы** — әзірлеу кезеңдерін көп мәрте өту, әзірлеу процесінің басында әзірленетін бағдарламалық құралға немесе жүйеге қойылатын талаптарды ішінара белгілеуге негізделген. Талаптар әзірлеудің жүйелі циклдерінде ұдайы нақтыланады. Әр әзірлеу циклінің нәтижесі әдетте бағдарламалық өнімнің кезекті жеткізілетін нұсқасын білдіреді.

**БҚ тиімділігі** — бұл бағдарламалық қамтамасыз етудің пайдаланатын есептеу ресурстарының көлеміне ұсынылатын қызметтер деңгейінің қатынасы.

# Әдебиеттер тізімі

## I бөлім


*Бахтизин В. В.* Бағдарламалық қамтамасыз етуді әзірлеу технологиясы: оқу құралы/ В.В. Бахтизин, Л. А.Глухова. — Мн. : БГУИР, 2010.

*Благодатских В.А.* Бағдарламалық құралдарды әзірлеуді стандарттау: оқу құралы / В. А. Благодатских, В.А. Волнин, К. Ф. Поскакалов ; ред. О. С. Разумов. — М. : Қаржы және статистика, 2003.

*Гагарина Л. Г.* Бағдарламалық өнімдерді әзірлеу технологиясының негіздері: оқу құралы / Л. Г. Гагарина, Б. Д. Виснадул, А. В. Игошин. — М. : ФОРУМ : ИНФРА-М, 2006.

*Иванова Г. С.* Бағдарламалау технологиясы: оқулық / Г. С. Иванова. — М. : Н. Э. Бауман ат. ММТУ баспасы, 2002.

*Леонников А. В.* UML үйрететін кітап / А.В. Леонников. — 2-ші басыл. — СПб. : БХВ-Петербург, 2004.

*Пономарев В.А.* Delphi-дегі COM және ActiveX / В.А.Пономарев. — СПб. : БХВ-Петербург, 2001.

*Рамбо Дж.* UML 2.0. нысанға бағытталған модельдеу және әзірлеу / Дж. Рамбо, М. Блаха. — 2- ші басыл. — СПб. : Питер, 2007.

*Рудаков А. В.* Бағдарламалық өнімдерді әзірлеу технологиясы: оқу құралы / А. В. Рудаков. — М. : «Академия» баспа орталығы, 2010.

*Шураков В.В.* Бағдарламалық қамитамасыз етудің сенімділігі / В.В.Шураков. — М.: Қаржы және статистика, 1987.

*Якобсон А.* Бағдарламалық қамтамасыз ету әзірлеудің жүйеленген процесі / А. Якобсон, Г. Буч, Дж. Рамбо. — СПб. : Питер, 2002.

2002 жылғы 27 желтоқсандағы «Техникалық реттеу туралы» № 184-ФЗ федералдық заңы.

IEEE 830 —1998. Бағдарламалық қамтамасыз етуге қойылатын талаптардың ерекшеліктерін жасаудың ұсынылатын әдістемесі.

МЕМСТ 19.001 —77. Бағдарламалық құжаттаманың бірыңғай жүйесі. Жалпы ережелер.

МЕМСТ 19.502 — 78. Бағдарламалық құжаттаманың бірыңғай жүйесі. Жалпы сипаты. Мазмұны мен рәсімдеуге қойылатын талаптар.

МЕМСТ 19.504 — 79. Бағдарламалық құжаттаманың бірыңғай жүйесі. Бағдарламашының басшылығы. Мазмұны мен рәсімдеуге қойылатын талаптар.

МЕМСТ 27.002 — 89. Техникадағы сенімділік. Негізгі ұғымдар. Терминдер мен анықтамалар.

МЕМСТ 34.602 — 89. Ақпараттық технология. Автоматтандырылған жүйелерге стандарттар кешені. Автоматтандырылған жүйе жасауға техникалық тапсырма.

Р ИСО/ХЭК 12207 МЕМСТ. Ақпараттық технология. Бағдарламалық құралдардың тіршілік циклінің процестері.

Р ИСО/ХЭК 15910 МЕМСТ. Ақпараттық технология. Бағдарламалық құралдың пайдаланушысының құжаттамаларын жасау процесі.

Р ИСО/ХЭК ТО 9294 МЕМСТ. Ақпараттық технология. Бағдарламалық қамтамасыз етудің құжаттандыруды басқару басшылығы.

Р ИСО/ХЭК ТО 15271 МЕМСТ. Ақпараттық технология. Р ИСО/ХЭК 12207 МЕМСТ қолдану басшылығы (Бағдарламалық құралдардың тіршілік циклінің процестері).

Р ИСО/ХЭК ТО 16326 МЕМСТ. Бағдарламалық инженерия. Жобаны басқарған кезде Р ИСО/ХЭК 12207 МЕМСТ қолдану басшылығы.

Р ИСО/ХЭК 12119 МЕМСТ. Ақпараттық технология. Бағдарламалар пакеттері. Сапаға қойылатын талаптар және тестілеу.

Р ИСО/ХЭК 9126 МЕМСТ. Ақпараттық технология. Бағдарламалық өнімді бағалау. Сапа сипаттамалары және оларды қолдану жөніндегі басшылықтар.

Р ИСО/ХЭК 8631 МЕМСТ. Ақпараттық технология. Бағдарламалық конструктивтер және оларды көрсетуге арналған шартты белгілер.

## II - бөлім


AJAX және PHP. Динамикалық веб-қосымшалар әзірлеу / К. Дари, Б. Бринзаре, Ф. Черчез-Тоза, М. Бусика. — СПб. : Таңба-Плюс, 2009.

*Дунаев В. В.* Барлығына арналған Web-бағдарламалау / В. В. Дунаев. — СПб. : БХВ-Петербург, 2008.

*Гудман* Д. JavaScript. Пайдаланушының інжіл кітабы / Д. Гудман, М. Моррисон. — СПб. : Вильямц 2006.

*Колисниченко Д. Н.* PHP 5/6 және MySQL 6. Web-қосымшаларды әзірлеу / Д. Н. Колисниченко. — СПб. : Вильямц 2009.

*Олифер В.* Компьютерлік желілер. Қағидаттар, технологиялар, хаттамалар/ В. Олифер, Н. Олифер. — СПб. : Питер, 2006.

*Олифер В.* Компьютерлік желілердің негіздері. / В.Олифер, Н. Олифер. — СПб. : Питер, 2009.

*Пауэлл Т.* Ajax. Бағдарламашының күнделікті қажет кітабы / Т. Пауэлл. — М. : Эксмо, 2009.

*Прохоренок Н.А.* HTML, JavaScript, PHP және MySQL. Web-шебердің джентльмендік жинағы / Н.А.Прохоренок. — 3-ші басыл. қайта өңд.және толықт. — СПб. : БХВ-Петербург, 2010.

*Томсон Л.* PHP және MySQL-да Web-қосымшалар әзірлеу/ Л. Томсон, Л.Веллинг. — К. : ДиаСофт, 2001.

*Храмцов П.Б.* Internet желісін және сервистерін басқару/ П. Б. Храмцов. — М. : Ақпараттық технологиялар орталығы, 1997.


## III - бөлім


*Бояркин В. Э.* «1С:Кәсіпорын 8». Деректерді айырбастау: қолданбалы шешімдердің арасында дерек алмасу (CD-ROM-дағы қосымшаларымен) / В.Э. Бояркин, А. И. Филатов. — М. : 1С-Паблишинг, 2015.

*Гончаров Д. И.* «1С:Кәсіпорын 8.2» арнайы қолданбалы міндеттерді шешу / Д.И.Гончаров, Е. Ю.Хрусталева. — М. : 1С-Паблишинг, 2015.

*Гончаров Д.И.* «1С:Кәсіпорын» бірігу технологиялары ( + CD) / Д.И.Гончаров, Е.Ю.Хрусталева. — М. : 1С-Паблишинг, 2015.

«1С: Кәсіпорын 8» жүйесіндегі кәсіби әзірлемелер (+DVD- ROM) / В.А. Ажеронок, А. П. Габец, Д. И. Гончаров, Д. В. Козырев және т.б.; М.Г. Радченконың редакциясымен — М. : 1С-Паблишинг, 2015.

*Радченко М. Г.* Сәулет және «1С:Кәсіпорын 8.2» деректерімен жұмыс істеу / М.Г. Радченко, Е.Ю. Хрусталева. — М. : 1С-Паблишинг, 2015.

*Радченко М. Г.* «1С:Кәсіпорын 8.2» таралымға шығатын қосымшаларды жасауға арналған құралдар/ М. Г. Радченко, Е.Ю.Хрусталева. — М. : 1С-Паблишинг, 2015.

*Радченко М.Г.* «1С:Кәсіпорын 8.2». Басты туралы қысқаша. 8.2 нұсқасының жаңа мүмкіндіктері / М.Г. Радченко. — М. : 1С-Паблишинг, 2015.

*Радченко М.Г.* «1С:Кәсіпорын 8.2». Әзірлеушінің тәжірибелік құралы. Мысалдар және типтік мысалдар / М. Г. Радченко, Е. Ю. Хрусталева. — М. : 1С-Паблишинг, 2013.

*Радченко М.Г.* «1С:Кәсіпорын 8.3». Әзірлеушінің тәжірибелік құралы. Мысалдар және типтік мысалдар / М. Г. Радченко, Е. Ю. Хрусталева. — М. : 1С-Паблишинг, 2013.

*Радченко М.Г.* «1С:Кәсіпорын 8.3». Әзірлеушінің тәжірибелік құралы. Мысалдар және типтік мысалдар / М. Г. Радченко, Е. Ю. Хрусталева. — М. : 1С-Паблишинг, 2015.

Басқарылмалы интерфейстерді әзірлеу (+CD) / В. А.Ажеронок, А. В. Островерх, М.Г. Радченко, Е.Ю.Хрусталева. — М. : 1С-Паблишинг, 2015.

«1С:Кәсіпорын 8.2» жүйесінде қолданбалы міндеттерді іске асыру (+CD) /

А. П.Габец, Д. В. Козырев, Д. С. Кухлевский, Е.Ю.Хрусталева. — М. : 1С-Паблишинг, 2015.

*Рыбалка В. В.* Hello, 1С. «1С:Кәсіпорын 8.2» платформасында қосымшаларды тез әзірлеу мысалы. Шеберлік кластары ( + CD). 2 нұсқа/

В. В. Рыбалка. — М. : 1С-Паблишинг, 2015.

*Рыбалка В. В.* Hello, 1С. «1С:Кәсіпорын 8.3» платформасында қосымшаларды тез әзірлеу мысалы. Шеберлік кластары ( + CD). 3 нұсқа / В.В. Рыбалка. — М. : 1С-Паблишинг, 2015.

*Рыбалка В. В.* Mobile 1С. «1С:Кәсіпорын 8.3» платформасында мобильді қосымшаларды тез әзірлеу мысалы. Шеберлік кластары. 1 нұсқа / В.В. Рыбалка. — М. : 1С-Паблишинг, 2015.

*Филиппов Е. В.* Технологиялық сұрақтар бойынша 1С:сарапшының күнделікті қажет кітабы / Е. В. Филиппов. — М. : 1С-Паблишинг, 2015.

«1С» фирмасы: «1С:Кәсіпорын 8.3. Бағдарламалауды оқытуға арналған нұсқа». — М. : 1С-Паблишинг, 2015.

*Хрусталева Е. Ю.* «1С:Кәсіпорын» платформасындағы мобильді қосымшалардың әзірленімдерімен танысу ( + CD) / Е.Ю.Хрусталева. — М. : 1С-Паблишинг, 2015.

*Хрусталева Е.Ю.* «1С:Кәсіпорын 8» жүйесінің сұрау салу тілі / Е.Ю. Хрусталева. — М. : 1С-Паблишинг, 2015.

## IV бөлім

*Бурк Р.* Жүйе әкімгерлеріне арналған UNIX. Пайдаланушы энциклопедиясы/ Р. Бурк, Д.Хорват және т.б. — К. : ДиаСофт, 1998.

*Воеводин В. В.* Параллельді есептеулер / В. В. Воеводин. — СПб. : БХВ-Петербург, 2002.

*Гудыно Л. П.* Операциялық жүйелер. Практикум / Л. П.Гудыно, А. А. Кириченко, С.В.Назаров. — М. : КУДИЦ-ПРЕСС, 2008.

*Карпов В.Е.* Операциялық жүйелердің негіздері / В.Е.Карпов, К.А.Коньков. — М. : Ақпараттық технологиялардың интернет-университеті, 2004.

*Королев Л. Н.* Электронды есептеу машиналары процессорларының құрылымы / Л. Н. Королев. — М. : М. В.Ломоносов ат. ММУ ЕМК факультетінің баспа бөлімі, 2003.

*Костромин В. А.* Пайдаланушыға арналған Linux үйретететін кітап / В. А. Костромин. — СПб. : БХВ-Петербург, 2002.

*Ляхов* Д. А. Linux жаңа бастағандар үшін / Д.А.Ляхов. — М. : Бестселлер, 2003.

*Назаров С. В.* Операциялық орталар, жүйелер және сырты. Құрылымдық және функционалдық ұйымдастыру негіздері / С.В.Назаров. — М. : КУДИЦ- БАСПА, 2007.

*Назаров С. В.* Заманауи операциялық жүйелері [Электрондық ресурс] / С.В.Назаров, А. И. Широков., http://www.untuit.ru/studies/ courses/631/487/info

*Олифер В. Г.* Желілік операциялық жүйелер / В.Г. Олифер, Н.А.Оли- фер. — СПб. : Питер, 2001.

*Пильщиков В. Н.* IBM PC Ассемблер тілінде бағдарламалау /Н. Пильщиков. — М. : Диалог-МИФИ, 2005.

*Побегайло А.П.* Windows-та жүйелі бағдарламалау / А. П.Побегайло. — СПб. : БХВ-Петербург, 2006.

*Полубояров В.В.* Интернет-тораптарды жасау технологиясына кіріспе [Электрондық ресурс] / В. В.Полубояров., http://www.untuit.ru/studies/ courses/1036/239/info

*Таненбаум Э. С.* Заманауи операциялық жүйелер / Э. С. Таненбаум. — 4-ші басыл. — СПб. : Питер, 2006.

# Мазмұны

## IV БӨЛІМ

## ЖҮЙЕЛІК БАҒДАРЛАМАЛАУ

*Оқу басылымы*

**Федорова Галина Николаевна**

**Компьютерлік жүйелер үшін бағдарламалық қамтамасыз етудің бағдарламалық модульдерін әзірлеу**

**Оқулық**

Редакторы Л. *В. Толочкова*
Компьютермен беттеу: *Р. Ю. Волкова*
Корректор Л. В. *Гаврилина, Е. О. Беркутова*
Басылым № 101116887. 20.04.2016 ж. баспаға қол қойылды. Форматы 60 х 90/16.
«Балтика» гарнитурасы. Офсеттік қағаз. Офсеттік баспа. Баспа б. шарты 21,0.
Таралым 1000 дана. Тапсырыс №

«Академия» баспа орталығы ЖШҚ. www.academia-moscow.ru 129085, Мәскеу,
бейбітшілік даңғ. 101В, 1-құрыл.
Тел./факсі: (495) 648-0507, 616-00-29.
25.05.2015 ж. № РОСС RU. АЕ51. Н 16679 Санитарлық-эпидемиологиялық
қорытынды.

Идел-Баспада басып шығарылды.