ӘЛ-ФАРАБИ АТЫНДАҒЫ ҚАЗАҚ ҰЛТТЫҚ УНИВЕРСИТЕТІ

ФИЛОСОФИЯ ЖӘНЕ САЯСАТТАНУ ФАКУЛЬТЕТІ

ӘЛЕУМЕТТАНУ ЖӘНЕ ӘЛЕУМЕТТІК ЖҰМЫС КАФЕДРАСЫ

**«ӘЛЕУМЕТТАНУЛЫҚ ДЕРЕКТЕРДІ ТАЛДАУ»**

**ПӘНІ БОЙЫНША ДӘРІСТЕР**

Кредит саны - 3

Алматы, 2022

«**ӘЛЕУМЕТТАНУЛЫҚ ДЕРЕКТЕРДІ ТАЛДАУ**»

ПӘНІ БОЙЫНША ДӘРІСТЕР

**МОДУЛЬ 1. Қолданбалы әлеуметтануға кіріспе**

1. **ДӘРІС.** Арнайы бағдарламалар негізінде ақпараттарды талдау мен өңдеудің теориялық негіздері

Сәйкесінше "технология" сөзін  әдеттегі мағынасымен Қолданбалы бағдарламалауды қамтамасыз ету аясында, БҚ- мен қойылатын талапты жасауды тудыратын өндіріс процестерінің жиынтығы деп түсінеміз, сонымен қатар осы процесстердің жиынтық мағынасы. Басқаша айтқанда, Қолданбалы бағдарламалауды қамтамасыз ету біз ауқымды мағынада бағдарламалық құралдарды өңдеудің технологиясы, яғни өзіне барлық процестерді қоса , осы құралдардың туылу ойларынан бастап және сонымен қатар қажетті бағдарламалық құралдарды жасаудың  байланысы сияқты деп түсінеміз. Әдістердің және құралдардың қандай болмасын байланысы осы жиынтықтың әрбір процесін қолдануға негізделеді, мысалы, компьютер (бұл жағдайда біз бағдарламалаудың компьютерлік технологиясы жайлы сөйлесеміз).

Қолданбалы бағдарламалауды қамтамасыз ету анықтамасы әдебиетте және басқа да бірнеше айырмашылығы бар болады. Сонымен қатар әдебиеттерде бағдарламалық технология ұғымына жақын бағдарламалық инженерия ұғымы, анықталатын жүйені өңдеуге жақын, пайдалануда, ере жүруге және бағдарламалық құралдардың айналдыруынан алып тастауға қолданылады. Қолданбалы бағдарламалауды қамтамасыз ету бағдарламалық инженериядан аралық негізгі ерекшелігі дисциплинада зерттеуге арналған қарау тәсілінде және материал жүйесінің келтірулерінде болып жатыр. Қолданбалы бағдарламалауды қамтамасыз ету акцент процестердің БҚ-ң  (технологиялық процестердің) өңдеулерін  зерттеуде және олардың ретпен өтуінде- әдістер және аспаптық құралдардың БҚ-ң  өңдеулері осы процестерде (олардың қолданылуы технологиялық процестерді құрастырады) қолданылады.  Сол уақытта бағдарламалық инженерияда, ең алдымен БҚ-ң  өңдеулерінің әдістері және аспаптық құралдары көру нүктесінен айқын мақсатқа жету- әр түрлі технолоиялық процестерде (және әр түрлі Қолданбалы бағдарламалауды қамтамасыз ету); қалай бұл әдістер және құралдар технологиялық процестерді құрастырып жатыр деген болмашы сұрақ осында оқытылып жатыр.

Сонымен қатар Қолданбалы бағдарламалауды қамтамасыз ету бағдарламалау методологиясымен шатастыруға болмайды. Бірақ екі оқиғада да әдістер оқытылады, бірақ Қолданбалы бағдарламалауды қамтамасыз ету сында әдістер" үстіңгі жағынан "- технологиялық процестердің ұйымын көру нүктесінен, ал бағдарламалау методологиясында әдістер " төмен жағынан "- оларды құрудың негізін көру нүктесінен қарастырылады (жұмыста бағдарламалау методологиясы механизмдердің жиынтығы ретінде,  бағдарламалық қамтамасыз етудің өңдеуі барысында  және  бір ғана жалпы философиялық жақын келумен біріккен қолданылуына байланысты анықталады).

Нәтижесінде, сенімділік ажырамас БҚ-ң  атрибутымен пайда болады, біз Қолданбалы бағдарламалауды қамтамасыз ету сенімді БҚ-ы  өңдеудің технологиясы сияқты талқылаймыз. Бұл дегеніміз, біріншіден, БҚ ниетінің көріну кезеңінен бастап, біз БҚ өңдеулерінің барлық процесін талқылаймыз, екіншіден,  бізді тек қана бағдарламалық конструкцияларды құру сұрақтары ғана емес, сонымен қатар функцияларды сипаттау сұрақтары және қабылданушы шешімдердің көру нүктесінен олардың адамдықты қабылдаудың, және, әйтеуір, технология өнімінің арқасында біз сенімді (дұрыс емес) БҚ-ы  қабылдаймыз.

БҚ-ы  өңдеуінің процестері барлық мына әдістерді таңдауға және аспаптық құралдарға маңызды әсер етеді.

**Қолданбалы бағдарламалауды қамтамасыз ету және қоғамды ақпараттандыру.**

  Қолданбалы бағдарламалауды қамтамасыз ету әр түрлі бағдарламалаудың дамуы кезеңдерінде әр түрлі маңызға ие болды. Компьютерлердің қуаттылық өлшемінің жоғарылауы және құралдардың дамуы және бағдарламалау методологиясының өсуі және компьютерлерде тапсырмаларды шешу қиындады, соның нәтижесінде Қолданбалы бағдарламалауды қамтамасыз ету сы жоғары нәтижеге ие болды. Компьютерлер құнының арзандауы және әсіресе  компьтерлік тасымалдауда ақпараттың сақталуы барлық адамзаттың  қызмет сферасына компьютерлерді практикалық кең енгізілуіне әкелді, Қолданбалы бағдарламалауды қамтамасыз ету теріс бағытын өзгертті. Адамзат   факторы мұнда шешуші роль ойнады. Сонша емес акценттері оның тиімділігінде, жұмыс ыңғайлылығында пайдаланушыларға арналған (оның сенімділігі туралы айтпағанда) БҚ сапасының жеткілікті терең ұғымы құрылды. Компьютерлік желілерді кең қолданумен таратылған есептеулерде хабарға дистанциялық рұқсатты және адамдар арасында электрондық тәсілмен хабар алмасудың қарқынды дамуына әкелді. Компьютерлік техника бөлек мақсаттардың шешіміне арналған барлық нақты және мүмкін ақпаратты үлгілеудің   құралына әлемді айналдырып жатыр және адамдарға оларды ынталандыратын сұрақтарға жауап беруге қабілетті. Терең және толық ақпараттандырудың (компьютерлендірудің) кезеңі адам қоғамында басталады. Бұлар Қолданбалы бағдарламалауды қамтамасыз ету алдына жаңа және қиын  мәселелерді қойып жатыр.

**Бағдарламалаудың он жылдағы дамуына қысқаша сипаттама**сы.

50 жылдары компьтерлердің қуаттылығы үлкен емес болатын (бірінші ұрпақ компьютерлері), ал бағдарламалау олар үшін негізінде, машиналы кодта басқарылды. Ең алдымен ғылыми- техникалық мақсат( шот формулалармен) шешілді, ережеге сай, жеткілікті дәл тапсырманы орнатып қоятын бағдарламалау тапсырмасы болатын.  Бағдарламалаудың интуициялық технологиясы қолданылды: тапсырма бойынша бағдарламаны құрастыруға кірісті, мұнымен тапсырма бірнеше рет өзгертілетін( уақытты күшті үлкейтетін және интерциондық процессіз бағдарламаны құрастыратын), бағдарлама жұмыс істей бастаған соң ең аз құжатнама рәсімделетін.  Әйткенмен де нақ мына дәуірде Қолданбалы бағдарламалауды қамтамасыз ету арналған модульдік бағдарламалау концепциясы дүниеге келді( машиналы кодта бағдарламалау қиындықтарын жеңуге арналған).   Бағдарламалаудың жоғары дәрежелі тілдері пайда болды, солардың ішінен тек келесі он жылдықтарға қолдану үшін тек қана ФОРТРАН өтті.

60 жылдары қарқынды даму және жоғары дәрежелі(АЛГОЛ 60, ФОРТРАН, КОБОЛ және басқа ),  бағдарламалау тілдерінің кең қолдануын байқауға болады, Қолданбалы бағдарламалауды қамтамасыз ету айқын ұлғайды. Үміт мынада,  үлкен бағдарламалық өңдеулерде бұл тілдер барлық проблемаларды шешетінін ақтамады. Компьютерлердің қуаттылығының жоғарылауы нәтижесінде және бағдарламалаудың тәжірибеде қорлануы жоғары дәрежелі тілдерде қиындық тудыратын компьютерлік тапсырмаларды шешу жылдам өсті, нәтижесінде бағдарламалардың модульдік ұйымының тілдері шектелген. Сонымен тек қана  ФОРТРАН, модульдік бағдарламалаудың мүмкіншілігін қамқорлықпен сақтағандықтан, келесі он жылдықта паң өтті (барлығы оны тілдеді, бірақ оны пайдаланушылар оның қызметтерінен бас тарта алмады, себебі бағдарламалық модульдердің қорларының орасан күрделі қорланулары, жаңа бағдарламаларда табыспен қолданылуы). Сонымен қатар, түсінікті болды, маңызы тек қана қандай тілде бағдарламалап жатқанымыз емес, сонымен қатар біз қалай бағдарламалап жатқанымыз. Қолданбалы бағдарламалауды қамтамасыз ету және методологияның  үсінде парасатпен ойлану басым болды. Компьютерлердің 2-ші ұрпағының пайда болуы мультибағдарламалардың және үлкен бағдарламалық жүйелердің жасалуына әкелді. Парасатпен технологиялық бағдарламалардың қатарын  қою, соңында ұжымдық өңдеуді қолданумен мүмкін болды.

70 жылдары берілген базаларда және ақпараттық жүйеде кең тарату алды. 70 жылдардың ортасында болып өткен ең маңызды оқиғаның жасалуы: бір битте ақпараттың сақталу құны, дәстүрлілерге қарағанда азырақ. Қолданбалы бағдарламалауды қамтамасыз ету қарқынды дамыды: негіздеу және төмен түсетін өңдеудің кең енгізілуі және бағдарламалау құрылысы, тап осы және модульдік бағдарламалаудың абстракты үлгілерінің дамуы (сонымен қатар, егжей- тегжейін ашудың бөлінулерінде ойлардың пайда болуы және модульдердің орындаулары және модульдерді қолдану, тап осы жасырушы құрылымдарда), БҚ-ң мобильділігін және сенімділігін қамтамасыз ету проблемаларын зерттеу, БҚ-ы өңдеумен ұжым басқаруының әдістемесін жасау, аспаптық бағдарламалық құралдардың көрінуін (бағдарламалық аспаптардың) сүйеудің Қолданбалы бағдарламалауды қамтамасыз ету лары.

80 жылдары барлық адамзаттың қызмет  ортасында, сонымен бірге БҚ  пайдаланушылардың кең және әр түрлі континентінде айқын  компьютерлерді кең енгізумен сипатталады. Бұл БҚ сапасының айқын тұжырымдамасының жасалуына және пайдаланушылық интерфейстерінің қарқынды дамуына ертіп әкелді. Қолданбалы бағдарламалауды қамтамасыз ету талаптарын үйретуші бағдарламалау тілдері (мысалы, тозақ) пайда бола бастады. БҚ –ң егжей-тегжейін ашу әдістері және тілдері дамыды. БҚ-ы   өңдеудің объектісіне жақын келу позициясы алдыңғы шепке шықты. БҚ сүйемелдеудің және әр түрлі өңдеудің аспаптық орталары жасалынды. Компьютерлік желілердің тұжырымдамасы дами бастады.

90- жылдары барлық адамзат қоғамында кең қамтумен  халықаралық компьютерлік желі айрықша белгілі болды, оған арнайы компьютерлер терминал ретінде іске қосылды. Бұл компьютерлік- желі (технологиялық сияқты, дәл осылай заңгерлік және этикалық сипаты) ақпаратын реттеп отыруға мәселелер қатарын тудырды. Компьютерлік хабар және хабарлардың желімен таралуында қорғаныш мәселесі қатал тұрды. БҚ-ы өңдеудің компьютерлік технологиясы (CASE-технологиясы) және онымен бағдарламалардың егжей- тегжейін ашудың формалды әдістері қарқынды дами бастады. Толық ақпараттандырудың және қоғамды компьютерлендірудің шешуші кезеңі басталды.

 **Бағдарламалық құралдарды өңдеу ерекшелігі.**

  Бағдарламалық құралдардың өңдеуіне ерекше ерекшеліктер қатары тән.

Ең алдымен бірнеше қарама- қарсы келушілерді белгілейік: БҚ-а (тапсырмаларды орнату)  талаптардың формальды емес сипаты және ондағы қате ұғымдар, бірақ БҚ бағдарламасы- өңдеудің формализацияланған ең басты объектісі. Осы БҚ-ң  өңдеулері формализацияның айқын кезеңдерін қамтиды, ал формальды емес өтуден формальдыққа өтудің маңызы формалданбаған.

БҚ өңдеуі маңызды творчестволық сипатты алып жүреді (әрбір адымда қандай болмасын таңдау істеуді, қандай болмасын шешім қабылдауды істеу қажет болады), ал қандай болмасын жүйелі регламенттік әрекеттерді орындауға болмайды. Мына өңдеу қандай болмасын күрделі құрылғыларсыз жобалау процесіне жақынырақ , бірақ олардың ешбір бұқаралық өндірісіне емес. Мына БҚ-ы  өңдеудің шығармашылық сипаты оның ең соңынан сақталады.

Сонымен қатар өнімнің өңдеу ерекшелігін белгілеп қою керек. Ол өзімен мәтіндердің кейбір жиынтығын ұсынады (яғни статикалық объектілерді), осы процестерді іске қосатын осы текстердің мағынасы (семантика) пайдаланушылардың әрекеттерімен осы процестерді іске қосатын берілгендердің өңдеу процестерімен және пайдаланушылардың әрекеттерімен осы процестерді іске қосуды (яғни динамикалық болады) айтады. Мына ерекше қабылдаулардың, әдістердің және құралдардың қатар өңдеушісімен таңдау алдын ала анықталады.

Өңдеу өнімінің  және басқа ерекше ерекшелігі болады: БҚ өзі қолданатын (пайдаланатын) қорларға шығын шығармайды және жұмсамайды.

**Бағдарламалық құралдардың тіршілік циклы.**

 БҚ-ң тіршілік циклы деп оның барлық дәуір бойы өңдеулерін және пайдаланылуын  (қолдану), БҚ ниетінің пайда болу кезеңінен бастап және оның барлық түрлерінің қолданылуын түсінеміз.

Тіршілік циклы БҚ-ң (software process)  барлық процестерін қолданылуын және жасалуын қосады. БҚ-ң тіршілік циклы келесі  сатыларды айырып жатыр (сурет 1): БҚ-ң  өңдеуін, бағдарламалық бұйымдардың өндірісі (БӨ) және БҚ-ң пайдаланылуы.


Сурет 1. БҚ-ң тіршілік циклының сатылары және фазалары.

 БҚ-ы өңдеу сатысы оның сыртқы суреттеулерінен , БҚ-ы құрастыру кезеңінен, БҚ-ы кодтау (тар мағынада бағдарламалау) кезеңінен, БҚ-ң іскерлік сараптауларынан тұрады. Барлық мына кезеңдерге құжаттау процестері және БҚ-ы  өңдеуімен басқару (management)  бірге жүріп жатыр. Құрастыру кезеңдері және кодтау жиі немесе кейде күшті шектеледі. Бұл дегеніміз, бағдарламалық құралдардың кейбір бөліктерін кодтау құрастыру кезеңінің аяқталуына дейін басталған болуы мүмкін.

БҚ-ы сыртқы суреттеу (Requirements document)  оның сыртқы көру нүктесінен сипаты  оның салыстырмалы сапасының бақылаушы талаптарымен бекітілген болады. БҚ-ң сыртқы сипаты пайдаланушылар (тапсырыс берушілер) жағынан БҚ-ң талаптарын анықтаудан басталады.

БҚ-ы құрастыру (design) мына процестерді қамтиды: БҚ-ң  сәулетін өңдеу, БҚ бағдарламасының құрылымын өңдеу және оның бөлшегінің толық егжей- тегжейін ашу.

Кодтау (coding ) бағдарламалау тілдерінде бағдарламалардың мәтіндерін және олардың жөндеуін БҚ-ң  тестілеуімен жасау.

БҚ-ң іскерлік сараптау кезеңінде БҚ сапасының бағалауы шығарылады, табысты аяқтаудан кейін БҚ-ң өңдеуі аяқталған болып есептеледі.

Бағдарламалық бұйым (ББ)- БҚ-ң өңделуімен алынған дана немесе көшірме.

 ББ-ы даярлау- бұл генерация процесі немесе бағдарламалардың жаңғыруы (көшірме алу) және БҚ- ы бағдарламалық құжаттарының мақсатпен оларды жеткізуді пайдаланушы тағайындауымен қолдануға арналған. ББ өндірісі- бұл анықталған мезгілдерде ББ сандарын  даярау қамтамасыз етуімен жұмыстардың жиынтығы. БҚ-ң  өндіріс сатысы БҚ-ң  тіршілік циклына келеді, дәл осылай , қателерсіз және мүмкін автоматты орындалатын кертартпалыққа негізделген жұмысты ұсынады. Дәл осылай ол әр түрлі техниканың өндіріс сатысының негізінде ерекшеленеді. Әдебиетте осыған орай мына сатыны, ережеге сай,  БҚ-ң тіршілік циклына қоспайды. БҚ-ң пайдалану сатыларын сақтау , БҚ-ң енгізулер және ере жүру процестерін , сонымен қатар ББ өз тағайындауымен тасулар мен қолдануды (operation)  қамтып жатыр. Ол екі параллельді өтетін фазалардан тұрады: БҚ-ы қолдану фазасы және БҚ-ы сүйемелдеу  фазасы.

БҚ- н қолдану (operation)- бұл практикалық мақсаттардың шешіміне арналған БҚ-ы компьютерде орындалу жолымен оның бағдарламаларын қолдану.

БҚ- н сүйемелдеу (maintenance)- бұл оның пайдаланудағы сапасы туралы хабар жинау процесі, оның жөндеулер мен өзгертулерінде табылған қателерді жою, сонымен қатар оның өзгертулері туралы салынған пайдаланушылардың хабарлаулары.

**Бағдарламалық құралдар сапасының ұғымы**

 Әрбір программалық құралдар (ПҚ) белгілі функциялар атқаруы тиіс, яғни ойластырғанды орындау керек. Жақсы  ПҚ оны ұзақ период аралығында сәтті қолдануға мүмкіндік беретін қасиеттер тізбегіне ие болуы керек, яғни белгілі қасиетке ие болуы керек. ПҚ сапасы – бұл тұтынушылардың берілген қажеттіліктерін қанағаттандыру қасиетіне әсер ететін оның ерекшеліктері (черты) мен сипаттамаларының жиынтығы. Бұл дегеніміз мынаны білдіреді, әртүрлі ПҚ олардың мүмкін болған жоғары дәрежелеріндегі бірдей қасиеттер жиынтығын қамту керек. Бұған мына факт кедергі жасайды, осындай қасиеттердің бірімен ПҚ сапасының жоғарылауы – құнының өзгеруімен, өңдеу мерзімдерінің аяқталуымен және осы ПҚ сапасының оның басқа да қасиеттері бойынша төмендеуімен ғана жетуі мүмкін. ПҚ сапасы оның сәтті қолданылуына кепілдік беру дәрежесінде көрсетілген қасиеттерге ие болғанда ғана қанағаттандырылған болып табылады.

Қолданушылар үшін ПҚ сапасының қанағаттандырылуын ұйымдастыратын ПҚ қасиеттерінің жиынтығы осы ПҚ шарттары мен эксплуатация сипатынана тәуелді, яғни осы ПҚ сапасы қарастырылуы керек позициядан тәуелді. Сондықтан да ПҚ сапасын сипаттау кезінде, ең алдымен, талап етілген  ПҚ қасиеттерінің іріктеу критерийлері тіркелуі керек. Қазіргі уақытта ПҚ сапасының критерийлері ретінде мыналарды қарастыру қолданған:

* функционалдылық;
* сенімділік (надежность);
* қолдану жеңілдігі;
* эффективтілік;
* жанамалық (сопровождаемость);
* мобильділік.

**Функционалдылық** – тұтынушының берілген немесе ойластырылған қажеттіліктерін қанағаттандыратын функциялар жинағын орындау қабілеті. Көрсетілген функциялар жинағы ПҚ-ның ішкі сипаттамасында анықталады.

Сенімділік бірінші лекцияда анық талқыланған.

**Қолдану жеңілдігі**– бұл бастапқы берілгендердің, ПҚ-ын қолдану мен алынған нәтижерді бағалау дайындығы бойынша тұтынушының ынтасын (усилие) азайтуға, сонымен бірге тұтынышының анықталған немесе ойластырылған оң эмоцияларын тудыруға мүмкіндік беретін ПҚ сипаттамалары.

**Эффективтілік** – бұл ПҚ-ның сипаттамалары берілген шарттар, яғни қолданылатын ресурстардың көлемі бойынша ұсынатын қызметтердің деңгейінйң қатынасы.

**Жанамалық** – бұл тұтынушылардың өзгеріп отыратын қажеттіліктеріне сәйкес ондағы қателерін жоюға арналған және оның модификациясы үшін өгерістер енгізу бойынша күш-жігерін азайтуға мүмкіндік беретін ПҚ сипаттамасы.

**Мобильділік** – бұл ПҚ-ның бір ортадан басқасына, яғни бір ЭЕМ-нен басқасына өту қасиеті.

Функционалдылық пен сенімділік ПҚ сапасының міндет критерийлері б.т., мұнда сенімділікті қамтамасыз ету ПҚ-ны өңдеудің барлық этаптары мен үрдістері бойынша қызыл жолмен өтеді. Қалған критерийлер тұтынушының ПҚ-ға қойылған талаптарына сәйкес қажеттіліктеріне тәуелділікте қолданыады. Олардың қамтамасыз етілуі курстың сәйкес бөлімдерінде қарастырылатын болады.

**Бағдарламалық құрылғыларды жасау сенімділігін қамтамасыз ету**

 ПҚ-ны жасаудың барлық техникалық үрдістеріне спецификациялық бояу(покраска) қойылатын ПҚ өңдеудің негізі мотиві б.т. ПҚ сенімділігін қамтамасыз ететін жалпы принциптерін қарастырайық. Техникада сенімділікті қамтамасыз етудің төрт жолы бар:

* қателерді ескерту;
* қателерді табу;
* қателерді түзелеу;
* қателерге тұрақтылықты қамтамасыз ету.

Қателерді ескерту жолының мақсаты – дайын өнімдерде қатеге жол бермеу. Біздің жағдайда – ПҚ-да.

ПҚ-ын жасау кезіндегі қателер табиғатын жүргізілген қарастыру келесі сұрақтар бойынша назар аударуды қажет етеді:

* қиындықпен күресте;
* аудару дәлдігінің қамтамасыз етілуіне;
* тұтынушы мен дайындаушының арасындағы кедергіні жоюда;
* қабылданатын шешімдердің реттелуін қамтамасыз етуді.

Бұл әдіс ПҚ жасау үрдістердің ұйымымен байланысты, яғни программалау технологиясымен. Дегенмен, біз жоғарыда айтып өткендей, ПҚ-да қателердің жоқтығына кепіл беру мүмкін емес, бірақ бұл әдітің шеңберінде ПҚ сенімділігінің ыңғайлы деңгейіне қол жеткізуге болады.

Қалған үш әдістер технологияның өз өнімінің ұйымымен байланысты, біздің жағдайда программамен. Олар программалардағы қателердің мүмкіндігін ескеріп отырады. Программада қателерді өздігінен табу – программаның   программаның оның орындалу үрдісінде қабылдамауларды табу құрылғыларын қамтитынын білдіреді.

Программаларда қателерді өздігінен түзету орындалу үрдісінде қабылдамауды табуды ғана емес, бұл қабылдамау салдарын түзетуді білдіреді, сондықтан да программада сәйкес құрылғылар болу керек.  Программаның қателерге тұрақтылығын қамтамасыз ету мынаны білдіреді, яғни программада программалардың қабылдамау әсерінің облысын, не оның жағымсыз салдарын азайтуға, ол кейде қабылдамаудың (отказ) апаттық салдарын қабылдамауды локалдауға мүмкіндік беретін құрылғылар болады. Дегенмен, бұл әдіс-жолдар сирек қолданылады (мүмкін, қатысты түрде қателерге тұрақтылықтың қамтамасыз етілуі жиірек қолданылады). Бұл біріншіден, мынамен байланысты, техникада осы әдістер шеңберінде қолданылатын қарапайым тәсілдер программалауда қолданылмайды, мысалы жеке блоктардың және қондырғылардың қайталануы (бір не сол программаның екі көшірмесінің орындалуы әрдайым бірдей әсердің – дұрыс не дұрыс емес орындалуына әкеп соғады). Ал екіншіден, программаға қосымша құрылғылардың қосылуы оның күрделіленуіне әкеледі, бұл қандай да бір өлшемде қателерді ескерту әдісіне бөгет алады.

**Құрылымдық программалау**

  Модульді программалау кезінде мынаны ескерген жөн, бағдароама тек компьютерге ғана емес, адамға да түсінікті болуы керек: модульді тексеретін тұлғалар да, модульдің жөнге салуына (отладкасына) арналған тестті дайындаушы тест жасаушылар да, модульдің талап етілген өзгерістерін жүзеге асыратын – ПҚ жүргізушілер де модуль жұмысының логикасын көп рет жіктеуі керек. Программалаудың заманауи тілдерінде осы логиканы күштірек шатастыруы, сол арқылы оны адам үшін модульді қиынырақ түсіндірілетін етеді, осыған сәйкес оны сенімсіз немесе қиынырақ жанамалы (сопровождаемый) ететін жеткілікті программалар құралдары бар. Сондықтан да сәйкес тілдік құралдардың таңдауы үшін шараларды қолдану міндетті және программалаудың белгілі тәртібімен жүру керек. Бұған Дейкстра назар аударып, программаны басқарушы құрылымдардың бірнеше типінен тұратын композиция түрінде тұрғызуды ұсынды. Олар программа-ның жұмыс логикасын ұғынуды артырады. Тек осындай құрылымдарды қолданатын программалауды құрылымдық деп аталады.



Сурет 2. Құрылымдық программалаудың негізгі басқарушы құрылымдары

Құрылымдық программалаудың негізгі құрылымдары мыналар болып табылады: алғасу, тармақтану және қайталану (сурет1). Бұл құрылымдардың компоненттері S,S1,S2 жалпыланған операторлары (өңдеу түйіндері) және Р шарты ретінде программалаудың қолданылатын тілінің қарапайым операторы (тиесілеу, кіріс, шығыс, процедураға жүгіну операторлары) басқарушы құрылымының композициясы болып табылатын бағдарлама үзіндісі.

Осы құрылымның әрбіреуі басқару бойынша тек бір кіріс пен бір шығысқа ие екені анық. Сол арқылы жалпыланған оператор да тек бір кіріс пен бір шығысты иеленеді.

Бұл құрылымдардың енді математикалық объектілер (құрылымдық прораммалаудың сәттілік себебін түсіндіреді) болып табылатыны өте маңызды. Әрбір құрылымдалынбаған программа үшін функционалды эквивалентті құрылымдық пограммаға құруға болатыны дәлелденген. Құры-лымдалған программа үшін программаларда кейбір қателерді анықтауға мүмкіндік беретін кейбір қасиеттерді математикалық дәлелдеуге болады. Бұл сұраққа жеке лекция арналатын болады.

Құрылымдық программалауды кейде «GO TO-сыз программалау» деп те атайды. Бірақ мұнда мәселе GO TO операторында ғана емес, оның ретсіз қолданылуында. Құрылымдық програмалауды программалаудың кейбір тілдерінде (мысалы ФОРТРАН) өте жиі іске асыру кезінде. Өту операторы (GO TO) құрылымдық программалаудың негізгі артықшылықтарын төмендетпей құрылымдық конструкциялардың жүзеге асырылуы (реализация) үшін қолданылады. Программаны «құрылымдық емес» өту операторы шатастырады, әсіресе жоғарыда орындалатын өту операторының модулінің текстінде орналасқан операторға өту болып табылады. Дегенмен де, кейбір қарапайым жағдайларда өту операторынан құтылу әрекеті өте үлкен құрылымдалған программаларға әкелуі мүмкін, бұл олардың нақтылығын жоғарылатады және модуль текстінде қосымша қателердің пайда болу қиындығын қамтиды. Сондықтан мүмкін болатын орындарда өту операторын қолданудан құтылуды ұсынуға болады, бірақ программаның нақтылық тәсілімен емес.

Өту операторын қолданудың жағымды жағдайына берілген цикл немесе берілген процедураның жұмысын «уақытынан бұрын» (досрочно) тоқтататын, яғни кейбір құрылымдық бірліктердің (жалпыланған операторлардың) жұмысын аяқтайтын және сол арқылы программаның құрылымдылығын тек локалды түрде бұзатын ерекше шарт бойынша циклдан немесе процедурадан шығуды жатқызуға болады.

Үлкен қиындықтарды (және құрылымдардың күрделіленуін) реакцияның пайда болатын ерекше (жиі қателік) жағдайларға құрылымдық реализациясы туындатады. Бұл жағдайда құрылымдық бірліктен уақытынана бұрын шығуды тек орындап қана қоймай, және бұл жағдайдың (мысалы, сәйкес келетін диагностиканың ақпараттық берілуі) қажеттіөңделуін жүзеге асыруды да талап етеді. Ерекше жағдайлардың өңдеушісі программа құрылымдарының кез-келген деңгейінде орналаса алады, ал оған жүгіну (обращение) түрлі төменгі деңгейлерден жүзеге асады. Техникалық көзқарас жағынан ерекше жағдайларға реакцияның келесі «құрылымдық емес» реализациясы анағұрлым тиімдірек блоып табылады. Ерекше жағдайлардың өңдеушілері сол не басқа құрылымдық бірліктердің соңында орналасады және әрбір осындай өңдеуші өзінің жұмысын аяқтаған соң өзі орналасқан  құрылымдық бірліктен шығуды жүзеге асыратын жолмен программаланады. Мұндай өңдеушіге жүгіну берілген құрылымдық бірліктен(кез-келген құрылымдық бірлікті өзіне енгізе отырып) орындалады.

**Программалық құралдарды тестілеу және ретке келтіру** **(отладка)**

 ПҚ ретке келтірілуі – бұл оның программаларын орындау үрдістерін қолданумен ПҚ-дағы қателерді табу мен түзетуге бағытталған іс-әрекет.

ПҚ тестілеу – бұл оның программаларын берілгендердің кейбір жинақтарында орындау үрдісі. Оған қолданылу нәтижелері алдын-ала белгілі немесе осы программаның тәртіп ережелері белгілі. Берілгендердің көрсетілген жинағы тесттік не жай ғана тест деп аталады. Осылайша, ретке келтіруді үш үрдістің көп ретті қайталануы түрінде көрсетуге болады: ПҚ-да қателер айырмашылығы тұрақтанатын тестілеу, ПҚ программасында және документациясында қате орнын табу және программалар мен документация-ларды анықталған қателерді жою мақсатымен редактрлеу. Басқаша сөздермен:

Ретке келтіру(отладка) = тестілеу+қателерді іздеу+редактрлеу

Шетел әдебиеттерінде ретке келтіруді жиі түрде айырмашылық факті тестілеу кезінде қойылатын тек қателерді іздеу мен түзету үрдісі түрінде түсінеді. Кейде тестілеу мен ретке келтіруді синонимдер деп қарастырады.біздің елімізде ретке келтіру ұғымы ретінде тестілеуді де қоса қарастырады, сондықтан орын алған дәстүр бойынша жүретін боламыз. Негізінен, бұл үрдістердегі бірігіп қарастыру көрсетілген түрлі оқуды аса маңызды етпейді. Бірақ айта кеткен жөн, тестілеу ПҚ аттестация үрдісінің бөлігі ретінде де қолданылады.

**Программалық құралдарды ретке келтірудің (отладка) принциптері мен түрлері**

  Айтарлықтай дәрежедегі ретке келтіру жетістігін тестілеуді рационал ұйымдастыру анықтайды. Ретке келтіру кезінде негізінен болу-болмауы ПҚ-ды тестілеу кезінде орнатылатын қателер ізделінеді және жойылады. Тестілеу ПҚ-ның дұрыстығын дәлелдей алмайды, ол ондағы қателердің болу-болмауын тек қана көрсете алады. Басқаша айтқанда, ПҚ-ны орындалатын тесттер жиынымен тестілеу арқылы ондағы бар әрбір қатенің болуын орнату мүмкін екендігіне кепіл беруге болмайды. Сондықтан да екі тапсырма туындайды. Біріншісі: осындай тесттер жиынын дайындау және ондағы мүмкін болған қателердің үлкен санын анықтау үшін оларға ПҚ қолдану. Алайда тестілеу процесі ( сондай-ақ ретке келтіру де) неғұрлым ұзақ жалғасқан сайын, ПҚ бағасы да өсе береді. Осыдан екінші тапсырма шығады: ПҚ-ны (немесе оның жеке компоненттерін) ретке келтіруді аяқтау сәтін анықтау.  Ретке келтіруді аяқтау мүмкіндігінің белгісі ПҚ тесттері арқылы ПҚ программасын орындау кезінде туындайтын  өткізіліп алынған түрлі себептердің қоршап алу толықтығы(полнота) және тестілеу процесінің соңғы кесіндісіндегі ПҚ қателерін қатысты сирек анықтауы болып табылады. Соңғысы ПҚ-ның оның сапасының спецификациясында көрсетілген талап етілетін сенімділік дәрежесімен сәйкестікте анықталады.

Тест жиынын оптимизациялау үшін, яғни берілген санда (немесе берілген уақыт интервалында) қателердің үлкен санын анықтауға мүмкіндік беретін осындай тесттер жиынын дайындау үшін қажет, біріншіден, осы жиынды алдын-ала жоспарлау, және екіншіден, тестті жоспарлаудың рационал стратегиясын қолдану. Тестті жобалауды ПҚ-ның ішкі сипаттау этапын аяқтаған соң дереу бастауға болады. Тесттері жобалау стратегиясын өңдеудің екі келесі шектік жолдары арасында шартты түрде графикалық орналастыруға болатын (1 сурет) түрлі жолдар болуы мүмкін. Сол жақ шеткі жол тесттердің ПҚ спецификациясын (ішкі сипаттамасы, архитектура сипаттамасы және модульдер спецификациясы) меңгеру негізінде ғана жобаланатындығына негізделеді. Мұнда модульдерді тұрғызу еш ескерілмейді, яғни олар қара жәшіктер сияқты қарастырылады. Шындығында мұндай жол кіріс берілгендерінің барлық жиындарын толық іріктеп алуды талап етеді, өйткені тест ретінде осындай жиындардың бөлшектерін қолдану кезінде ПҚ-ның  кейбір программалар участоктері ешқандай тесттерде жұмыс жасай алмайды, демек ондағы қателер білінбейді. Дегенмен ПҚ-ны кіріс берілгендерінің толық жиынымен тестілеу тәжірибе жүзінде іске аспайды. Оң жақ шеткі жол мынаған негізделеді, тесттер БҚ-дың әрбір бағдарламаларын орындаудың барлық жолдарын тестілеу мақсатымен бағдарламалардың текстерін сәулелендіру негізінде жобаланады. Егер программаларда қайталанудың айнымалы сандары бар циклдардың бар болуын есепке алатын болсақ, онда ПҚ бағдарламасын орындаудың түрлі жолдары өте көп болып кетуі мүмкін  сондықтан оларды тестілеу тәжірибе жүзінде орындалмайды.



                        Сурет 3. Тесттерді жобалау жолдарының спектрі

Тесттерді жобалаудың оптимал стратегиясы екі шектік жолдар арасындағы, бірақ сол жақ шетке жақынырақ интервалдар ішінде орналасқан. Ол принциптерден шыға отырып, тесттердің айтарлықтай бөліктерін жобалауды қамтиды: әрбір қолданылатын функцияға немесе мүмкіндікке – ең болмағанда бір тест, әрбір облыс пен қандай да бір кіріс шамасының өзгеруінің әрбір шекарасына – ең болмағанда бір тест, әрбір ерекше жағдайға немесе спецификацияларда көрсетілген әрбір ерекше ахуалға(ситуация)- ең болмағанда бір тест. Бірақ ол сонымен қатар кейбір тесттердің және принциптерден шыға отырып минимум сияқты) бағдарлама мәтіндері бойынша жобалауды талап етеді: ПҚ-дың әрбір бағдарламасының әрбір командасы ең болмағанда бір тестте жұмыс жасау керек.

Тесттерді жобалаудың оптималды стратегиясын келесі принциптің негізінде нақтылауға болады: ПҚ құрамына енетін әрбір программалық документ үшін (бағдарламалардың мәтіндерін қоса) ондағы қателерді анықтау мақсатымен өзінің тесттері жобалануы керек. Қандай  жағдайда да, бұл принципті ПҚ-ды анықтаумен және программалау технологиясы ұғымын сенімді ПҚ-ды өңдеу технологиясы сияқты  қамтумен сәйкестікте сақтау қажет. Осымен байланысты Майерс тіпті, тестілеудің түрлі түрлерін тест тұрғызылатын программалық документ түрінен тәуелділікте анықтады. Біздің елімізде ретке келтірудің(отладка) екі негізгі түрі(тестілеуді қоса отырып) ажыратылады: автономдық және комплекстік ретке келтіру. Автономды ретке келтіру ПҚ-ға енетін бағдрламаның тек қандай да бір бөліктерін қателерді тестілеу кезінде анықталатын іздеумен және түзетумен тестілеуді білдіреді. Ол шындығында, әрбір модульдің ретке келтіруі мен модульдердің түйіндесуінің(сопряжение) ретке келтіруін қамтиды. Комплекстік ретке келтіру ПҚ-ға жататын барлық документтердегі (ПҚ бағдарламалар текстерін қоса алғанда) қателерді тестілеу кезінде анықталатын іздеумен және түзетумен ПҚ-ды тестілеуді білдіреді. Мұндай документтерге ПҚ-ға талаптарды анықтау, ПҚ сапасының спецификациясы, ПҚ-дың функционалды спецификациясы, ПҚ архитектурасын сипаттау мен ПҚ бағдарламаларының текстері кіреді.

**Программалық құралдарды өңдеу аспаптары**

 Программалық құралдарды сол немесе басқа шамада өңдеу процессінде ПҚ-ды өңдеу процестерін компьютерлік қолдау қолданылады. Бұл ПҚ-дың ең болмағанда бірнеше программалық документтерін берілгендердің компьютерлік тасымалдағыштарында (мысалы, дискеттерде) көрсету және арнайы ПҚ-дың ПҚ өңдеушінің билігін немесе қандай да бір осындай өңдеулер үшін құрылған арнайы құрылғылар компьютерінің құрамына қосылған көрсетулер жолымен жүзеге жетеді. Мұндай арнайы ПҚ ретінде қандай да бір программалау тілінің компиляторын көрсетуге болады. Компилятор ПҚ өңдеушісін ПҚ өңдеушісіне аса ыңғайсыз болатын компьютер тілінде программаларды жазу қажеттілігінен құтқарады, -оның орнына ол компилятор автоматты түрде компьютер тіліне аударатын өзіне ыңғайлы программалау тілінде программалар құрады. ПҚ өңдеу процесін қолдайтын арнайы құрылғы ретінде қандай да бір тілдің эмуляторы қызмет ете алады. Эмулятор ПҚ өңдеуін қолдайтын компьтер тілінен тамаша тілдерде программалар орындауға (интерпретаторлауға) мүмкіндік береді.

Басқа ПҚ-ды өңдеуді қамтамасыз етуге арналған ПҚ-ды ПҚ өңдеудің программалық аспабы деп атаймыз, ал ПҚ өңдеуін қолдауға арнайы арналған компьютер құрылғысын ПҚ өңдеудің аппараттық аспабы деп атаймыз.

ПҚ өңдеу аспаптары түрлі программалық документтермен жұмыс істеуге арналған ПҚ-дың барлық өмірлік цикл аралығында қолданыла алады. Осылай мәтіндік редактор түрлі программалық документтерді өңдеу үшін қолданыла алады. Функция жағынан алғанда, ПҚ-ды өңдеу кезінде орындайтын аспаптарды келесі төрт топтарға ажыратуға болады:

Редакторлар;

Анализаторлар;

Түрленгіштер;

Бағдарламаларды орындау үрдісін қолдайтын аспаптар

Редакторлар өмірлік циклдің түрлі этаптарындағы сол немесе басқа программалық документтердің құрылымын(жүйеленуін) қолдайды. Алдын айтылғандай, бұл үшін бір әмбебап(универсальный) мәтіндік редакторды қолдануға болады. Алайда, өте күштірек қолдауды арнайыландырылған редакторлар қамтамасыз ете алады: документтің әр түрі үшін – өзінің редакторы. Өңдеудің алдыңғы этаптарында документтерде сипаттаудың графикалық құралдары (диаграммалар, схемалар және т.б.) кеңінен қолданыла алады. Мұндай жағдайларда графикалық редакторлар аса тиімді болуы мүмкін. Программалау этапында (кодтау) мәтіндік редактор орнына қолданылатын программалау тіліне бағытталған синтаксистік басқарылатын редактор анағұрлым ыңғайлы болуы мүмкін.

Анализаторлар оларды басқарудың түрлі түрлерін, олардың анықталған қасиеттерін не анықтау және статикалық берілгендердің(мысалы документтердің көрсетілген стандарттарға сәйкестігін анықтау) жиналуын  жүзеге асыра отырып документтердің статикалық өңделуін, не бағдарламалардың динамикалық анализін(мысалы, бағдарлдаманың жұмыс уақытын программалық модульдер бойынша орналастыру мақсатымен) жүзеге асырады. Түрленгіштер документтерді көрсетудің басқа формасына  (мысалы, форматерлер) автоматты түрде келтіруге және бір түр документтің документтің басқа түріне (мысалы, конверторлар немесе компиляторлар)аударуға, жеке бөліктерден қандай да бірдокументтерін синтездеуге мүмкіндік береді.

Бағдарламалардың орындалу процесін қолдану аспаптары  компьютерде процестердің сипаттамасын немесе түрі машиналық кодтан тамаша көрсетілген олардың жеке бөліктерін, немесе оынң интерпретациясының қосымша мүмкіндіктерімен машиналық кодты орындауға мүмкіндік береді. Мұндай  аспаптың мысалы басқа компьютер кодының эмуляторы  болып табылады. Аспаптардың мұндай тобына түрлі ретке келтірулерді жатқызуға болады. Негізінде, программалаудың әрбір жүйесі программалау тіліне арналған едәуір типтік программалық фрагменттерді орындайтын және программаларды орындау кезінде туындайтын ерекше ахуалдарға(мұндай жүйешелерді орындаушы қолдау деп атайтын боламыз) стандартты реакцияларды қамтамасыз ететін  орындау периодының программалық жүйешесін құрайды, - сонымен қатар берілген топтың аспабы ретінде қарастыруға болады.

 **Программалық құралдарды өңдеу мен шығарып салудың (сопровождение) аспаптық орталары**

Қазіргі уақытта программалаудың әрбір жүйелерімен жеке элементтер емес (мысалы, компилятор), ал берілген программалау тілінде ПҚ- ды өңдеу мен шығарып салуды қолдайтын программалық және аппараттық аспаптардың немесе қандай да бір нақты заттық облысқа бағытталған аспаптардың кейбір логикалық жиынтығы байланысқан. Мұндай жиынтықты ПҚ- ды өңдеу мен шығарып салудың аспаптық ортасы деп атаймыз. Мұндай аспаптық орталары былайша сипатталады, біріншіден программалық сияқты аппараттық аспаптарды қолдану, екіншіден анықталған ориентация не нақты программалау тіліне, не нақты заттық облысқа ориентация.

Аспаптық ортаны өзінде қолданылатын сол компьютерде ғана функционалдау міндетті емес. Мұндай бірігу жиі- жиі ыңғайлы болады (егер тек қолданылатын компьютер қуаты осыған, мүмкіндік берсе): түрлі тип компьютерлерімен жұмыс істеудің керегі жоқ, ал өңделетін ПҚ-ға ең аспаптық ортаның компоненттерін қосуға болады. Дегенмен, ПҚ қолданылуы керек компьютер осы ПҚ өндірушілері үшін (мысалы, ол үнемі үзуге болмайтын басқа жұмыспен босамаса, немесе ол әліде өңдеу кезеңінде болса), не ПҚ  өңдеу үшін ыңғайсыз болса, не бұл компьютердің қуаты аспаптық ортамен талап етілетін функционалды қамтамасыз ету үшін жеткіліксіз болса, онда аспаптық объектілік деп аталатын, жол қолданылады. Оның мәні мынада, ПҚ аспаптық деп аталатын бір компьютерде жасалады,  және мақсаттық (немесе объектілік) деп аталатын басқа компьютерде қолданылатын болады.

ПҚ- ды өңдеу мен шығарып салу аспаптық ортасының үш негізгі түрін ажыратады (1 сурет): программалау орталары, компьютерлік технологияның жұмыс орындары, программалау технологиясының аспаптық жүйелері.

Программалау ортасы негізінен ПҚ-ды   программалау (кодтау), тестілеу және ретке келтіру процестерін қолдау үшін арналған. Компьютерлік технологияның жұмыс орны ПҚ-ды жасаудың бұрынғы этаптарын қолдауға және программаларды  спецификациялар бойынша автоматты генерациялауға бағытталған. Программалау технологиясының аспаптық жүйесі өңдеудің барлық процестерін қолдау үшін және ПҚ- дың бүкіл өмірлік циклінің аралығында шығарып салу үшін арналған және ұзақ өмірлік циклі бар үлкен программалық жүйелерді ұжымдық өңдеуге бағытталған. Мұндай жүйелер үшін шығарып салу құны әдетте өңдеу құнын арттырады.



Сурет 4. ПҚ-ды өңдеумен шығарып салудың аспаптық орталарының негізгі класстары

 **Программалаудың аспаптық орталары**

Программалаудың аспаптық орталары ең алдымен, программасы берілген программалау тілінде құрылымдауға мүмкіндік беретін мәтіндік редакторды, программаны осы тілде компилярлауға және интерпретаторлауға, сонымен қатар алынған программаларды тестілеу мен ретке келтіруге мүмкіндік беретін аспаптарды қамтиды. Сонымен қатар басқа да аспаптардың болуы мүмкін, мысалы программалардың статикалық және динамикалық анализіне арналған аспаптар. Бұл аспаптар бір- бірімен өзара файлдық жүйенің стандартты мүмкіндіктерінің көмегімен қарапайым файлдар арқылы әсерлеседі.

Программалаудың аспаптық орталарының келесі кластарына ажыратады (Сурет 1): 1) жалпы тағайындау (назначение)орталары; 2) тілдік бағытталған орталар.

Жалпы тағайындаудың программалау аспаптық орталары түрлі программалау тілдерінде (мысалы, мәтіндік редактор, байланыс редакторы немесе мақсаттық компьютер тілінің интерпретаторы) программалардың өңделуін қолдайтын программалық аспаптардың жиынын қамтиды және әдетте, қолданылатын операциялық жүйенің кейбір мүмкіндіктерінің кеңеюін көрсетеді. Мұндай ортада программалау үшін программалаудың қандай да бір тілінде осы тілге бейімделген (мысалы, компилятор) қосымша аспаптар қажет болады.



Сурет 5. Программалаудың аспаптық орталарының классификациясы

Программалаудың тілдік- бағытталған аспаптық ортасы қандай да бір программалау тілінде ПҚ өңдеуін қолдау үшін арналған және осы тіл туралы осы орталарда  құрылысы қолданылды. Сондықтан бұл осы орталарда осы тілдің спецификациясын үйретуші қуатты мүмкіндіктер болуы мүмкін. Осы орталар екі классқа бөлінеді.

Тілдік- білгібағдарламалаудың аспаптық ортасы ПҚ өңдеуін сүйеу үшін қандай болмасын бір тілде бағдарламалауда және осы тіл туралы білім осы орталарда маңызды қолданылуға арналған. Сол себепті мына осы орталарда тап осы тілдің ерекшелігін үйретуші жеткілікті қуатты мүмкіншіліктер болуы мүмкін. Тап осы орталар екі класқа бөлінеді: 1) түсіндіруші орта; 2) синтаксистік- басқарылатын орта.

Бағдарламалауды түсіндіруші аспаптық орта бағдарламалаудың тап осы тілінде бағдарламаларды қамтамасыз етеді, яғни ең алдымен бағдарламалау тілінің интерпретаторы осы ортаны қамтиды. Осындай орта үлгіні (осынайлардың бірі, Лисп сияқты) түсіндіруші бағдарламалау тілдері үшін қажет, бірақ және басқа тілдер (мысалы, аспаптық компьютерлерде) үшін қолданылуы мүмкін.  Бағдарламалаудың синтаксистік- басқарылатынаспаптық ортасы бағдарланатын бағдарламалау тілінің синтаксистік біліміне негізделеді. Осындай ортада мәтіндіктің орнына пайдаланушыға мүмкіндік беретін синтаксистік конструкциялардың (нәтижесінде осы өңделетін бағдарлама әрқашанда синтаксистік дұрыс болады) әр түрлі үлгілерінде синтаксистік-  басқарылатын редактор қолданылады. Бір уақытта осындай редактор бағдарламамен яғни басқа аспаптармен қолданылуы мүмкін синтаксистік ағашын қалыптастырады.

####

# ****Бағдарламалық  құралдарды  өңдеудің  компьютерлік технологиясының  ұғымы  және  оның  жұмыс  орны****

 CASE технологиясын (ПҚ өңдеуінің компьютерлік технологиясы) қатал анықтамаларын өндіруде бірсыпыра қиындықтар болады. CASE- бұл ағылшыннан абревиатурасы Computer-Aided Software Engineering (Компьтерлік - Көмектесудің Бағдарламалық Инженериясы). Бірақ компьтердің көмегінсіз (сүйеуінсіз) әлдеқашаннан бері өңделмейді (бірақ құрастырушы қолданылады).  Ақиқатта, бұл ұғым көбірек тар (арнайы) мағынаны береді, бірте- бірте шайылып жатыр (мынаның  әрқашанда, қандай болмасын ұғымының қатал анықтамасы болмайды). Алғашында CASE бағдарламалық сүйеуді (бағдарламалық аспаптардың) қолданумен, ПҚ (ПҚ сәулетінің және сыртқы суреттеуді өңдеу, талаптарынң анықтамасы) өңдеулерінің ерте кезеңдеріндегі инженериясы ретінде түсінілді. Қазір  CASE –пен  тек қана осы оқиғада ПҚ-ң барлық инженерияның құжаттарымен бағдарламалар жарым- жарты немесе толық генерирленгенде, ертелерде көрсетілген өңдеу кезеңдерімен алынған тіршілік циклы түсінілуі мүмкін. Осы оқиғада CASE-технологиясы қолдық ( дәстүрлі) ПҚ өңдеуінен принципте айырмашылығы болды: тек қана технологиялық процестердің мазмұны емес, сонымен қатар олардың жиынтығы өзгерді. Осы шақта ПҚ өңдеудің *компьютерлік технологиясы* талаптардың өңдеуіне арналған бағдарламалық сүйеулер және ПҚ графикалық спецификациясын, бағдарламалардың автоматты генерациялары қандай болмасын бағдарламалау тілінде немесе машиналы кодта (жарым- жарты немесе толық) бағдарламалық сүйеулердің прототипімен сипаттамасы қолданылады.

Сонымен қатар, ПҚ өңдеуінің компьютерлік технологиясы, яғни бағдарламалық құжаттардың ұсынуын компьютерде есептегенде "қағазсыз" болып табылады. Бірақ, ПҚ өңдеулерінің қолдық технологиясын компьютерліктен мына белгілермен ажырату қиын. Яғни, ең маңыздысы компьютерлік технологияда белгіленбеген.

ПҚ өңдеуінің қолдық технологиясы компьютерліктен негізгі айырмашылығы келесідей болады. Қолдық технология ПҚ-ң әр түрлі өңдеуімен хабарланған,  бірдей түсінетін түрлі өңдеушілермен,  сол уақытта компьютерлік технология сияқты семантикалық түсінуді қамтамасыз етуге хабарланған (түсіндірудің)  құжаттардың компьютерлік технологиясының бағдарламалық  сүйеуі.

Сол уақытта семантикалық түсіну сияқты бағдарламалардың автоматты генерация мүмкіншілігі құжаттарды бағдарламалық сүйеуге береді, ал сондай түсінуді қамтамасыз етудің қажеттілігін кіріс құжаттардың қажет және әр түрлі графикалық түрлері істейді. Нақ осы ПҚ ере жүрулерін және дәл технологиялық процестерді өңдеуді өзгертуді орынды рұқсат етеді.

Өкізілген талқылаулардан компьютерлік технологияны ПҚ тіршілік циклымен байланысты өзгеруді түсінуге болады. Егер өңдеуде қолдық технологияның негізгі күшейтуін  қолдансақ ПҚ өзіндік бағдарламалаудың (кодтаудың) кезеңдерінде істелді, яғни ПҚ ( функционалды спецификацияны және талаптарды анықтау) өңдеуімен ертедегі кезеңдерде компьютерлік технологияны қолданумен. Осымен құжаттаманың сипаттамасы өзгерді: формальды емес құжаттардың бүтін бауының орнына, хабарланған тапсырыс берушінің ( пайдаланушының) хабар тапсыруында әр түрлі категорияларды өңдеуші ПҚ прототипі формаланады, ол таңдалынған пайдаланушының интерфейсін және формальды функционалдың спецификациясын қолдаушы, ПҚ (немесе бірақ олардың бөліміне маңызды) бағдарламаларының автоматты синтезіне (генерациясына)өзгерді.

**Программалық  құралдардың  өңдеуделуі компьютер технологиясы және оның жұмыс орынының ұғымы.**

 CASE-тың  анықталғанда  өндiрудегi кейбiр қиындықтар пайда болады - (ПТ өңдеудегi компьютер технологиясы) .CASE - бұл (Компьютер - программалаудағы көмектесетін инженерия) ағылшын Computer-Aided Software Engineeringлерiн абревиатура. ПТ-да  бiрақ компьютердi (қолдану ) ендi әлдеқашан кем дегенде (компилятор ең болмаса қолданылады) бірақ  өндемейдi. Ұғым шындықта, бұл бiртiндеп қалай бұл әрдайым ұғым қандай болмасын қатал анықтауы болмайтында болатынын бұзылып қалатын мағына (арнайы ) тарлау салады. CASE-ға бастапқы (программалық аспаптар) программалық қолдауды қолданумен (талаптардың анықтауы, ПТ-ның  сыртқы сипаттамасы және архитектурасының өңдеуi) ПТ-ның өңдеудiң ерте кезеңдерiнiң CASE-қа  ендi ақпаратты алады және (қоса және оның бақылап отыруы) ПТ-ның  жинағы тiршiлiк циклдасының инженериясы, бiрақ тек қана сол жартылай немесе толық бағдарлама өңдеудiң көрcетiлген ерте кезең алған құжаттар бойынша шығаратында жағдайда. CASE осы жағдайда - технология ПТ-ты өңдеудi (дәстүрлi ) қол технологиясынан айырмашылығы бола бастады: технологиялық үдерiстердi мазмұнғана емес, олардың жиынтығы да өзгердi. ПТ өңдеулер компъютер технологиясын дәл қазiр қолданумен мiнездеуге болады ПТ-ның график түрiнде талаптар және график түрiнде спецификацияларының өңдеулерi үшiн программалық қолдау, программалау тiлi немесе (жартылай немесе толық ) машина кодында, түпнұсқасын анықтауды программалық қолдау бағдарламалардың автоматты генерациясы қандай болмасын ол өзгермейді.Сонымен бiрге ПТ-ның  өңдеуiн компьютер технологиясы «Қағассыз» болып табылғанын айтады, яғни компьютер ұсыныс өлшеулi программалық құжаттар. Алайда, қиын бұл белгiлер ПТ-ның  өңдеуiн қол технологиясын компьютерден болған деген  сенiмдi айыру. Демек компьютер өзi маңызды  технологияларында ерекшеленбеген. ПТ-ның өңдеудi қол технологиясының бас айырмашылығы компьютерден төмендегiдей болады. Қол технологиясы компьютер технологиясы әртүрлi өңдеушi ПТ бiрдей түсiнiлетiн құжаттардың өңдеуiне сонда бағытталған компьютер технологиясының программалық қолдауын құжаттардың (интерпретация ) семантикалық түсiнуiн қамтамасыз етуге бағытталған. Құжаттардың компьютер ұсынысы мұндай олардың түсiнуiн әлi бiлдiруге жеткiлiксiз болады.Құжаттардың сонда семантикалық түсiнуi бағдарламалардың автоматты генерациясының мүмкiндiгiн программалық қолдауға бередi, мұндай түсiнудi қамтамасыз етудi қажеттiлiк қажет және кiретiн құжаттардың әр түрлi график түрiнде формалары iстейдi. Тап солай ПТ–ның өңдеу және бақылап отырудың технологиялық  үдерiстерiн   жиынтығы да өзгертуге ұтымды мүмкiндiк бередi.

Компьютер технологиясының мәндерi өткiзiлген талқылаудан  циклдегi оған қатысты өзгерiстерi де түсiнуге болады. Егер ПТ   өңдеу бойынша негiзгi күштердi  қол технологиясының қолдануы (кодтау ) программалау және (тестеу) талқылау шындығында кезеңде iстелiнсе, онда компъютер технологиясының қолдануында - (талаптардың анықтауы және функционалдық спецификация) ПТ-ның өңдеудiң ерте кезеңдерiне жатады. Сонымен бiрге құжаттаманың сипаты айтарлықтай өзгердi: өңдеушi әр түрлi дәрежелерге (қолданушы ) тапсырма берушiдегi мәлiметтерi берiлу бағдарланған табандатқан үйреншiктi емес құжаттардың тiркесi орынына таңдаулы қолданбалы интерфейс қолдайтын ПТ және үстiрт функционалдық спецификациялар түп тұлғалар қалыптасады, (немесе түбегейлi олардың бөлiгi  болмаса)ПТ  бағдарламалардың (генерация ) автоматты синтезi үшiн жеткiлiктi.Сонымен бiрге өңдеушiлерге және қолданушыларға қажеттi құжаттаманың бiр бөлiгiнiң автоматты генерациясының мүмкiндiгi қажеттi пайда болды. (кодтаулар ) қол программалауы орынына - керек емес автономды талқылаумен iстейтiн және бағдарламалардың тестеу бағдарламалардың автоматты генерациясы: ол орынына қабылдайды құжаттаманың терең автоматты семантикалық бақылауы жеткiлiктi. Айтарлықтай ПТ-ның бақылап отыруын сипаттайық: ол  өзгередi өңдеушi- жүргiзушiнiң барлық өзгерiстерi итке тек қана (қоса және түп тұлға) спецификацияда, өңге өзгерiстердi енгiзiледi автоматты iске асады. Компъютер технологиясының қолдануы бар ПТ-ның айтылған тiршiлiк циклiнiң есепке алуымен (1-шi сурет) келесi схемамен көрсетуге болады.



6-шi сурет. Программалық құралдың тiршiлiк циклi компьютер технологиясында қолданылуы.

Түпнұсқасын анықтау қолданушының аралығында өзара әрекеттесудi жанама сипаттаманы алмастыруға мүмкiндiк бередi және барлық қажеттi бөлшектердi бекiтумен әдiс және бұл өзара әрекеттесудi стильнiң қолданушысын төте сайлауды (ПТ-ға талаптарының анықтауы және сыртқы сипаттамада) автоматты түрде орнатуға болады.

Қолданбалы интерфейстiң дәл сипаттамасы, компъютер технологиясының түсiнiктi программалық қолдауына кәзiргi жағдайда өндiрiп алады, және де қолданушының жауапты қатысуымен: өңдеушi әр түрлi мүмкiндiктер және қолданушы мониторда қолданушыға көрсетедi қолайлы оны варианттар үшiн таңдайды, қолданушысы олардың үстiнде ол жұмыстанылатын ақпараттық объекттер және операциялардың белгiсi, оны өңдеушi арқылы ендiредi рұқсаттың әдiсiн таңдайды олардың әр түрлi терезелермен, мәзiр, виртуалды клавиатуралармен ол ұластырады әр түрлi бөлiктер және перденiң бөлшектерiнiң даярлауларын көлемдi кiтапханасы бар күйiне келтiрiлетiн қабықтың компьютер технологиясының программалық қолдауындағы бар болуға бұның барлық негiзделу тағы сол сияқтылар.Нәтижеде көрcетiлген әсерлер ПТ  қабықпен анықталады – ПТ  жоғарғы бағдарлаушы деңгей. Сiрә, мұндай түпнұсқасын анықтау қолданушымен және өңдеушiнiң аралығында тосқауылдың жеңуiн жақсы әдiс болып табылады. Спецификациялардың өңдеуi бiрнеше әртүрлi процесстерге ыдырайды. Егер (талаптардың анықтауы) спецификациялардың өңдеуiн бастапқы кезеңдi шығарса, онда процесстер бұл  формализацияланған құжаттардың жасауына, өйткенi жетектеп жүнетiн әдiстердi қолданылады. формализацияланған спецификациялар тiлдерiн қолданылады.Сонымен бiрге спецификациялардың әр түрлi схемалар және жеткiлiктi ақпараттық ортаның құрылымы және ПТ-ның басқаруын құрылымдарды анықтауға формалданатын диаграммалардың жасау жетектеп жүнетiн графикалық тәсiлдерi кең қолданылады. Мысалы, мұндай құрылымдарға мәлiмет сипаттамасының спецификацияларының алгебралық тiлiнелерi операциялық немесе аксиоматикалық семантиканы ) қолданушы көрсетiлген бөлiктерi және бағдарламалар немесе (негiзделетiн бағдарламалардың спецификациясына логикалық жолдарында) логикалық спецификациялар тiлдерi жалғанады. Мұндай спецификациялар бағдарламаларды автоматты шығаруға едәуiр дәрежеде немесе толық мүмкiндiк бередi.Программалауды технологияның аспапты жүйелерi. Өңдеулер және ұзақ тiршiлiк циклi бар үлкен Пт-ды бақылап отыруын компьютер қолдауы үшiн программалауды технологияның аспапты жүйелерiн қолданылады. Программалауды технологияның аспапты жүйесi - бұл программалық және аппаратты аспаптардың интеграцияланған жиынтығы, өңдеу және шеңберiнде нақтылы технологияның ағымында жинағы оның тiршiлiк циклдасының үлкен ПТ-ның бақылап отыруының қолдайтын барлық процесстерi. Бұл анықтаудан компьютер қолдауын бұл сыныптың келесi негiзгi сызықтары шығады:  жинақтылық, ұжымдық өңдеуге хабарланғандық, технологиялық айқындық,  үйлесiмдiлiк.

Ұжымдық өңдеуге хабарланғандық жүйе ұжымның жұмысымен (management ) басқаруды қолдайтынын және бұл ұжымның әртүрлi мүшелерi үшiн технологиялық үдерiстердi өнiмнiң әр түрлi бөлiктерiне әртүрлi қол жеткiзу құқықтары қамтамасыз етуi керек болатынын бiлдiредi. Компьютер қолдауын технологиялық айқындық оның жинақтылығы программалауды қандай болмасын нақты технологияның шеңберлерiмен шектелгенiн бiлдiредi. Программалауды технологияның аспапты жүйелерi болады бiрде аяқтаған олардың аспапты қол тимеушiлiгi болу үшiн үлкен және ПТ  жеткiлiктi.Қосылатын аспаптардың оларында жиыны сондықтан тiлдер және программалауды таңдаулы технология қолданылатын пәндiк облыстардың қажеттiктерiнiң есепке алуымен мұқият iрiктеледi. Компьютер қолдауын үйлесiмдiлiктi бiлдiредi  мәлiметтiң үйлесiмдiлiк,  қолданбалы интерфейс бойынша үйлесiмдiлiк,  (функцияларға) әсерлер бойынша үйлесiмдiлiк, Мәлiметтiң үйлесiмдiлiк аспаптар жүйенiң (ақпараттық объекттер) мәлiметтердiң бөлiктерiнiң әр түрлi қолданылатын жүйелерiнiң тәуелдiлiгi бiр-бiрiмен анықтайтын бекiтiлген (үлгiмен ) ақпараттық схемасымен сәйкес жұмыс iстегенiн бiлдiредi. Қолданбалы интерфейс бойынша үйлесiмдiлiк барлық аспаптар бiртұтас қолданбалы интерфейспен топтастырғанын бiлдiредi. Бiрiншiден, әсерлер бойынша үйлесiмдiлiк жүйеде барлық аспаптар жалпы бөлiмдерде болатынын бiлдiредi, екiншiден орындаудың жанында өз функцияларының бiр аспаптары басқа аспаптарға айнала алады.

* Әрбір  қабат барлық қабаттардың ішкі  құрылуын   білмей-ақ, алдын-ала анықталған интерфейс арқылы міндетті түрде қатысатын (анағұрлым төменірек) қабатпен басқару бойынша (компоненттерге айналу) өзара әсерлесе алады;
* Әрбір қабат не басқа қабаттардан  жасыратын, не келесі қабаттарға (көрсетілген, интерфейс арқылы)  олардың кейбір абстракцияларын қамтамасыз ететін нақты ресурстарға ие.

      Осылайша, қабатталған бағдарламалық жүйеде әрбір қабат бірілгендердің кейбір абстракциясын  ұйымдастыра алады. Қабаттар арасындағы байланыстар әрбір қабаттың  төменгі алмасуына айналу параметрлерінің  мәндерін таратумен және төменнен жоғарғы қабатқа айналу нәтижелерін жіберумен  шектелген. Бірнеше  қабаттары бар глобальды берілгендерді қолдану рұқсат етілмеген.

      Мысал ретінде операциялық жүйені тұрғызуға арналған мынадай архитектураны қолдануды  қарастырайық. Мұндай архитектураны ТНЕ операциялық жүйесін құру кезінде дейсктра қолданған. Бұл операцииялық жүйе қабаттан тұрады (Сурет 1). Нөлінші қабатта барлық үзлістердің өңделуі және орталық процессордың пакеттік режимде бағдарламаларға (үрдістерге) бөлінуі жүзеге асырылады. Тек қана осы қабат жүйенің мультбағдарламалық аспектілері жөнінде хабардар етілген. Бірінші қабатта жүйенің беттік ұйымдасуын басқару орындалады. Барлық жоғары тұрған қабаттарға виртуалды үздіксіз (беттік емес) жад тиесілі.Екінші қабатта оператор консолімен (басқару тетігі) байланыс жүзеге асады. Тек осы қабат қанаконсольдің техникалық  сипаттамаларын біледі. Үшінші қабатта берілгендердің кіріс және шығыс ағындарының буферизациясы жүзеге асады және кіріс пен шығыс абстрактілі  каналдары жүзеге асады, сондықтан қосымша бағдарламалар кіріс пен шығыс құрылғыларының  техникалық сипаттамаларын білмейді.

|  |
| --- |
|                                Қосымша бағдарламалар |
| 3: Берілгендердің кіріс және шығыс байланысын басқару; |
| 2:Оператордың консолімен байланысты қамтамасыз ету; |
| 1: Жадпен басқару; |
| 0: Үрдістердің диспетчерленуі және синхронизациясы. |

                      Сурет 7.ТНЕ операциялық жүйесінің архитектурасы.

  Параллель жұмыс жасайтын бағдарламалардың ұжымы бір мезгілде орындалу кезінде қатыса отырып, өзара әсерлесуге қабілетті бағдарламалрды  қамтиды. Бұл дегеніміз, мұндай бағдарламалар біріншіден оперативті жадқа шақырылған, белсендірілген және бір немесе бірнеше орталық процессорларды уақыт бойынша өзгермелі түрде бөле алады, екіншіден, базасында олардың синхронизациясы жүзеге асатын динамикалық әсерлерді (орындалу үрдісінде) өзара  жүзеге асыра алады.

Әдетте мұндай үрдістер арасындағы өзара әсерлесу бір-біріне кейбір мәліметтерді жіберу жолымен жүзеге асады.

      Мұндай архитектураның қарапайым түрі UNIX операциялық жүйесінде ұйымдасу құрылғылары бар болатын конвейерлер болып табылады. Конвейер әрбір бағдарламаның ең соңғысынан басқа стандарт шығысы осы тізбектің келесі бағдарламасының стандарт кірісімен байланысқан бағдарламалар тізбегін көрсетеді.(Сурет 2). Конвейер мәліметтердің кейбір ағындарын өңдейді.Осы ағынның әрбір мәліметі бірінші бағдарламаның кірісіне келіп түседі, ал оны өңдеп өңделген мәліметті келесі бағдарламаға жібереді, ал өзі ағынның келесі реттегі мәліметтерін өңдеуге кіріседі. Осылайша конвейердің әрбір бағдарламасы жұмыс істейді: орындалған бағдарламадан мәліметті алып жән өңдей отырып, өңделген мәліметті келесі бағдарламаға жібереді, ал конвейердің  соңғы бағдарламасы бүкіл конвейердің жұмыс нәтижесін шығарады.( нәтижелі мәлімет) .Осылайша,  n бағдарламалардан тұратын конвейерде бір мезгілдегі өңделу де  n мәліметтерге дейінгі мәліметтер орналаса алады. Әрине конвейердің түрлі бағдарламалары келесі мәләметтерді өңдеуге уақыттың түрлі үзіктерін жұмсай алатынын ескере отырып, осы үрдістердің (кейбір үрдістер күту кезеңінде, не өңделген мәліметті жіберу мүмкіндігінде бола алады) қандай да бір синхрнизациялау тәсілін қамтамасыз ету керек.



 Сурет 8. Параллель жұмыс жасайтын бағдарламалар конвейері

Анағұрлым жалпы жағдайда параллель әсерлесетін бағдарламалардың ұжымы мәліметтер порты бар жүйеге ұйымдаса алады.

     Мәліметтер порты өзімен бірге мәліметтердің кейбір ретіне қызмет  ететін бағдарламалық жүйелерді қамтиды: ол  бағдарламалардан қайсыбір мәліметті сақтауға қабылдай алады, және келесі мәліметтерді оның талабы бойынша басқа бағдарламаға бере алады. Қайсыбір бағдарламамен кейбір порттарға берілген мәлімет енді, бұл бағдарламаға (және оның ресурсын қолдануға) тиесілі болмайды, оның сұранысы бойынша қайсыбір бағдарламаға реті бойынша жіберілмейінше, ол басқа ешқандай бағдарламаға тиісті болмайды. Осылайша, мәліметті жіберетін бағдарлама осы мәліметті қабылдайтын бағдарлама оны өңдеуге дайын болмаса (егер тек қабылдайтын порт толып қалмайтын болса) күту кезеңінде орын алмайды.

      Мәліметтер порттары бар бағдарламалық жүйе мысалы 3- суретте келтірілген  И порты осы суретте келтірілген паралель әсерлесетін бағдарламалар ұжымына арналған шығыс мәліметтер порты ретінде қарастырылады.



        Сурет 9.  Мәліметтер  порттары бар бағдарламалық жүйе мысалы.

Бағдарламалық жүйелі мәліметтер порты қатқыл конфигурациялы және иілгіш конфигурациялы болуы мүмкін.

     Қатқыл  конфигурациялы портты жүйелерде әрбір бағдарлама бір немесе бірнеше  кіріс  порттармен  тығыз байланыста  болады.

Мұндай  бағдарлама  мәліметтерді  беру үші беру адресін  нұсқау қажет: бағдарама  аты және оның кіріс портының аты. Бұл жағдайда, жүйенің конфигурациясын жөндеу барысында қолданбалы  бағдарламалар  өзгереді: таратылатын мәлімет адресін өзгертеміз. Иілгіш конфигурациялы портты жүйелерде әрбір бағдарлама кіріс және шығыс виртуалды портымен байланысты. Мұндай жүйені қосу барысында пайдаланушы көрсеткен әрбір шығыс виртуалды  портының  кез-келген кіріс виртуалды портының мәліметтер негізінде алдын-ала  арнайы бағдарламалық компонент көмегімен баптау жүргізу қажет.Осылайша, бұл жағдайда жүйенің конфигурациясын өзгертуде қолданбалы бағдарламаны қандай да бір түзетудің қажеті жоқ. Керек түзетулер мәліметтерді баптауда көрсетіеді.Алайда бұл жағдайда жүйені баптайтын арнайы бағдарламалық компоненттің болуы шарт.

**Архитектуралық функциясы**

      Жүйе ішілік (подсистемами) өзара әсерлесу қатарында қандай да бір қосымша бағдарламалық компоненттің қажеті жоқ. Ол үшін алдын-ала келісімді реттеу және базалық бағдарламаның стандартты мүмкіндіктерінің болуы жеткілікті. Осылайша автономды орындалатын бағдарламалар комлексінде өзара әсерлесуді қамтамасыз ету үшін ішкі ақпараттық орта мен операциялық жүйенң мүмкіндіктері  жеткілікті қабатталған бағдарламалық жүйеде ерекшеленген бағдарламалық қабаттар мен қарапайым аппараттық үрдіске қаратылуы жеткілікті спецификация бола алады.Бағдарламалық конвейерде бағдарламалардың өзара әрекеттесуін операциялық жүйе (UNIX операциялық жүйесіндегі орны сияқты) қамтамасыз ете алады.

      Алайда бағдарламалық жүйе ішілікте (подсистемами) өзара, әрекеттесуді қамтамасыз ету үшін  қосымша бағдарламалық компоненті құру  қажеттігі туындайды.Автономды орындалатын бағдарламалардың жұмыс істеу комплексін меңгеру үшін жиі арнайы командалалық интерпретатор жасайды.Ол заттық облыста ішкі ақпараттық орта үшін және керекті бағдарламаны жіберу үшін операциялық қолданған базалық командалық интерпретаторға қарағанда өте қолайлы.Қабатталған бағдарламалық жүйеде процедуралық қабатпен арнайы қаратпа (обращения) аппарат жасалынуы мүмкін (мысалы, осы процедураларды параллельді орындауды қамтамасыз ететін.Параллельді орындайтын бағдарламалар тобын мәліметтер портымен басқару үшін жүйеішілік арнайы бағдарлама қажет. Мұндай бағдарламалық компоненттер ешқандай ішкі функцияларды орындамайды.Олар бағдарламалық ортаны жасалу барысында туындаған архитектураны жүзеге асырады.Осыған байланысты  осындай функцияларды архитектуралық деп аталады.

**Қолданылған әдебиеттер**

   1. И.Г.Гоулд, Дж.С.Тутилл. Терминологическая работа IFIP (Международная федерация по обработке информации) и ICC (Международный вычислительный центр)// Журн. вычисл. матем. и матем. физ., 1965, #2. - С. 377-386.

 2. Г.Майерс. Надежность программного обеспечения. - М.: Мир, 1980.

 3. Ian Sommerville. Software Engineering. - Addison-Wesley Publishing Company, 1992. - P.

 4. Э. Дейкстра. Заметки по структурному программированию// У. Дал, Э. Дейкстра, К. Хоор. Структурное программирование. - М.: Мир, 1975. - С. 7-97.

 5. Criteria for Evalution of Software. - ISO TC97/SC7 #367 (Supersedes Document #327).

 6. С.И. Ожегов. Словарь русского языка. - М.: Советская энциклопедия, 1975.

 7. Ф.Я. Дзержинский, И.М. Калиниченко. Дисциплина программирования Д: концепция и опыт реализации методических средств программной инженерии. - М.: ЦНИИ информации и технико-экономических исследований по атомной науке и технике, 1988. - С. 9-16.

 8. В. Турский. Методология программирования. - М.: Мир, 198

 9. Г. Буч. Объектно-ориентированное проектирование с примерами применения: пер. с англ. - М.: Конкорд, 1992.

 10. Е.А. Жоголев. Система программирования с использованием библиотеки подпрограмм// Система автоматизация программирования. - М.: Физматгиз, 196  С. 15-52.

 11.  Ф.П. Брукс, мл. Как проектируются и создаются программные комплексы/ Пер. с англ. А.П. Ершова. - М.: Наука, 1979.

 12. R.C. Holt. Structure of Computer Programs: A Survey// Proceedings of the IEEE, 1975, 63(6). - P. 879-893.

 13. Дж. Хьюз, Дж. Мичтом. Структурный подход к программированию. М.: Мир, 1980.

 14. Е.А. Жоголев. Технологические основы модульного программирования//Программирование, 1980, #2. - С. 44-49.

 15. Б. Боэм, Дж. Браун, Х. Каспар и др. Характеристики качества программного обеспечения. - М.: Мир, 198

 16. В.В. Липаев. Качество программного обеспечения. - М.: Финансы и статистика, 1983.

 17. Б. Шнейдерман. Психология программирования. - М.: Радио и связь, 1984.

 18. Revised version of DP9126 - Criteria of the Evaluation of Software Quality Characteristics. ISO TC97/SC7 #610. - Part 6.

 19. В.Ш. Кауфман. Языки программирования. Концепции и принципы. М.: Радио и связь, 1993.

 20. Требования и спецификации в разработке программ: пер. с англ. - М.: Мир, 1984.

 21. В.Н.Агафонов. Спецификация программ: понятийные средства и их организация. - Новосибирск: Наука (Сибирское отделение), 1987.