Дәріс№7.
Тақырыбы:Үрдістердің иерархиясы.Үрдістерді диспечерлеу және уақыт үйлесімдіру

Үйлесімдік түрлері Кез-келген ОЖ-нің нақты архитектуралық және функционалдық ерекшеліктері жүйелік программисттерге қатысты болуы керек және ол қарапайым қолданушыға таныс емес болуы мүмкін. Кейбір идеялар (мысалы, объектілі-бағытталған көзқарас) құрастырушыларға ғана белгілі және соңғы қолданушыға кері әсерін тигізеді. Көптеген қолданбалы орта концепциясы қолданушының көптен күткен мүмкіншілігін өзінің ОЖ программасында, басқа ОЖ және процессор үшін жазылған программада орындайды. Басқа ОЖ үшін жазылған ОЖ қосымшасы орындайтын мүмкіншілікті сипаттауы, яғни, ОЖ қасиеті үйлесімділік деп аталады.

Үйлесімділіктің бір-біріне үқсамайтын екі принципі бар, оларды шатастырмау керек: екілік деңгейдегі үйлесімділік және бастапқы мәтін деңгейіндегі үйлесімділік. Екілік түріндегі кодтар және деректері бар қосымша файлдар компьютерде орындаушы файлдар түрінде сақталады. Орындалушы программаны бір ОЖ ортасында жұмыс істейтін және оны басқа ОЖ ортасында орындағанда іске қосқан жағдайда орындалуын екілік үйлесімділік деп атаймыз.

Басқа, бөтен программаның орындалу уақытын қысқарту үшін қолданбалы орта кітапхана функциясына көңіл аударады. Осы әрекеттің тиімділігі көптеген қазіргі программаларды графикалық интерфейстің басқаруында (GUL), мысалы, Windows, Unix, сонымен бірге қосымша программа ереже бойынша аз ғана уақытын бірнеше қажетті әрекеттердің орындалуына жұмсайды. Терезелерді басқару үшін және т.б. GUL-ға байланысты әрекеттерді орындау үшін олар GUL кітапханасын шақыруға үздіксіз іске асырады. Осы ерекшелік қолданбалы ортаның уақыт шығынын толтырады. Дұрыс жоспарланған программа ортасының құрамында кітапхана, ішкі GUL кітапханасы болады, бірақ ол орнатылған ОЖ-нің «өзінің» кодында жазылған. Осылайша, басқа ОЖ-де API көмегімен программа жылдам орындалады. Бір ғана командамен орындалатын бұл жағдайды эмулирлеу процессінен ажыратуды трансляция деп атайды.

Бір ғана ОЖ-ге арналып жазылған программа басқа ОЖ үшін де орындалу керек болса API үйлесімділігімен қамтамасыздандыру жеткіліксіз. Әр түрлі ОЖ негізіне арналған концепция бір-біріне қайшы келуі мүмкін. Мысалы, бір ОЖ қамтамасында енгізу-шығару құрылғысын басқаруға рұқсат етілсе, ал басқа әрекеттерде ОЖ-нің прерогативі болып табылады. Өйткені, әрбір ОЖ-нің жеке өзінің қорын қорғау механизмі бар, яғни, олар: кез келген жағдайда қателерді өңдеу алгоритмі, процесстің ерекше құрылымы және жадыны басқару схемасы, өзінің файлға семантикалық қатынасы және қолданушының графикалық интерфейсі. Осы айырмашылықтардың бәрі ОЖ жұмыс істейтін аппаратты платформа спецификасымен, олардың жүзеге асырылу ерекшеліктерімен немесе жүйе құрастырушының ендірілуімен анықталады. Үйлесімділікті қамтамасыздандыру үшін бір ғана ОЖ үшін компьютер құрылғыларын, қорларын басқарудың бірнеше әдістерін міндетті түрде қарама-қайшылық тумайтындай үйымдастыру керек.

Үйлесімділікті жүзеге асыру әдістері Қазіргі кезде қосымша программалық қамтама кейбір ОЖ қолданушыларына «бөтен» программаларды іске қосуына мүмкіндік береді (мысалы, MaCOS және UNIX, DOS және Windows үшін программаларды орындауға мүмкіндік береді). Бірақ қазіргі ОЖ-де «бөтен» программаларды орындауға арналған құралдар жүйенің стандартты бөлігіне айналып келеді. ОЖ-ні таңдау қолданбалы программаны таңдауды көп шектемейді. Бірақ, MaCOS, Windows және UNIX үшін қолданушы интерфейс программасымен жұмыс істегенде экранда біраз қиыншылықтармен күресуге тура келеді, бірақ ОЖ әр түрлі қолданбалы ортасы тышқан немесе мәзір (меню) сияқты стандартты түрге айналады. Әр түрлі қолданбалы ортаны жүзеге асыру кезінде құрастырушылар қарама қайшылық талаптарымен қақтығысады. Бір жағынан әрбір қолданбалы ортаның міндетті программаны мүмкіншілігіне қарай орындауы, яғни, ол «өзінің» туған ОЖ-де орындалса. Бірақ бұл программалардың талап ету дәрежесінің үлкендігінен берілген ОЖ құрылысында қайшылық пайда болады. Арнайы құрылғы драйверлері қауіпсіздендіру талабына әруақытта жауап бере бермейді, жадыны басқару схемасы мен терезелі жүйе арасында қайшылық пайда болуы мүмкін. Бірақ, ең үлкен деңгейдегі проблема–өнімділік, яғни, қолданбалы орта программаны болғанша жылдамдықпен орындау керек. Бұл талапты көп қоданылып жүрген ерте эмулирленген жүйе қанағаттандыра алмайды. «Бөтен» программаларды орынауда

қолданбалы орта жылдамдықты қысқарту үшін кітапхана деңгейіндегі программаға үқсас қолданылады.

Үрдістерді басқаратын уақиғалық тітіктерҮрдістер әрекеттестігі.

Келесі мысалды қарастырайық. Егер квадрат түбірді алу программасын енгізу керек болса, онда әр түрлі бастапқы (берілгендердің) деректердің түбірінің мәнін ескере отырып компьютерлік жүйе есептеудің әр түрлі екі процессімен айналысады да, әр түрлі бастапқы деректер әр түрлі есептеу жиынын алуға әкеліп тірейді. Қолданушының көзімен қарасақ бірдей программа жүктелген, бірақ екі әртүрлі тапсырма ұйымдастырылған. Енді басқа мысал. Екі қолданушы да 1 ден (бір санынан) квадрат түбір алсын, онда олар ұқсас тапсырманы ұйымдастыру, бірақ оны есептеу жүйесінде уақыт бойынша жылжыту арқылы жүктеді. Сол кезде орындап жатқан тапсырмалардың біреуі нәтижесінде алынған мәнді баспаға шығаруға дайындалса және енгізу-шығару операциясының аяқталуын күтсе, ал екіншісі жаңадан тапсырманы орындай бастайды. Дәл осы жағдайды есептеу жүйесінің ішінде тапсырмалар ұқсастығы туралы айтуға бола ма? Жоқ, өйткені оларды орындайтын процесстің күйі әр түрлі. «Программа» және «тапсырма» қолданушы ұғымында есептеу жүйесінде болатын жағдайларды түсіндіру үшін орындалмайды. Мұндай жағдайда «программа» және «тапсырма» термині статикалық активті емес объектілерді сипаттауға арналған. Оның жұмыс істеу барысында компьютер әр түрлі командаларды өңдейді және айнымалының мәнін өзгертеді. Программа орындалуы үшін ОЖ оперативті жадының белгілі бір бөлігін (санын) бөліп беру керек, одан кейін арнайы енгізу-шығару құрылғысын немесе файлдарды (кірістегі деректер қайдан келеді және алынған нәтижесін қайда жеткізу керек) бекітіп беру керек, яғни толықырақ айтсақ есептеу жүйесінің жалпы ресурсының ішінен керек ресурстарды таңдап, белгілеп қою керек. Олардың саны және конфигурациясы белгілі бір уақытта өзгереді. Мұндай активті объектілерді сипаттау үшін компьютер жүйесінің ішінде «программа» және «тапсырма» терминімен бірге жаңа термин – «процесс» терминін қолданатын боламыз. ОЖ-нің басқаруымен табылатын «процесс» ұғымы орындалатын командалар жиынымен, ресурстар қауымдастығымен ерекшеленеді (жады немесе адрестік кеңістіктің орындалуы үшін, сетка, қолданылған файлдар және енгізу-шығару құрылғысы үшін бөлінген) және ағымдағы моментте оның орындалымен (регистр мәні, есептеуіш, сетка күйі және айнымалы мәні) ерекшеленеді. Есептеу жүйесін өңдеуде процесс пен программа арасында өзара-біркелкі сәйкестік болмайды. Арнайы программа жұмысы үшін кейбір ОЖ-де бір ғана прцесс немесе бірдей процесс бірнеше әртүрлі программаларды орындай алуды ұйымдастыруы мүмкін. Сонымен бірге, бір ғана процессте бір ғана программаны өңдеу керек жағдайы туса «процессті» орындаушы файлдың кодын, деректерді және оның ресурстарын жай динамикалық сипаттау түрінде қрастыруғаболмайды. Процесс ОЖ-нің басқаруымен орындалады, сондықтан онда оның ядросының бір бөлігі ғана орындалады (орындаушы файлда болмайтын), арнайы жоспаланған программа авторларымен (мысалы, жүйелік шақыруды қолданыда), сонымен бірге қарастырылмаған, жоспаланбаған жағдайда орындалады (мысалы, ішкі үзулерді өңдегенде).

Қақпан өңдеушісі өзінің шақыру кезінде машинаның қалып-күйін сақтау үшін уақытша үзуге тиым салады, өйткені егер жаңа үзу немесе нәтиже пайда болса, онда ақпараттар жоғалып кетер еді. Сол үшін ол қақпан кадырын құрады және оған үзілген ағынның орындалатын күйі жайында ақпаратты орналастырады. Үзу немесе нәтиже өңдеуді аяқталған соң бұл ақпарат ядроға ағынның әрі қарай орындалуына мүмкіндік береді.

Кейбір нәтиже өңдеуіш өзі шешеді, бірақ көбінесе ол пайда болған күйдің типін анықтап оны ядроның басқа модуліне немесе жүйелік атқарушыға басқаруды береді.

Мысалы, егер үзу құрылғыдан болса, онда басқару үзуді өңдейтін процедураларына жібереді. (interrupt service routine ISR). Егер үзу жүйелік қызмет көрсетуден пайда болса, онда өңдеуіш басқаруды NT жүйесінің жүйелік қызмет көрсету кодына жібереді. Бөлек нәтиже ядроның нәтижелер диспетчері арқылы өңделеді.

Үзуге үзу диспетчері жауап қайтарады. Ол үзудің көзін анықтап басқаруды сыртқы қызмет көрсететін процедураларға немесе ядроның ішкі процедураларына береді.

Үрдістің біресепті және көпесепті орындалуы.

Процесс бір күйден екіншісіне өз еркімен ауыса алмайды. Процесс күйін өзгерту қызметін ОЖ атқарады. Мұндай операцияларүрлері жоғарыдағы біздің модельде

бағыттаушы (стрелка) санымен сәйкес келеді. Оларды үш топқа жүптастырып қарастырған өте қолайлы:

-процесстің құрылуы – процесстің аяқталуы;

-процессті тоқтату («орындалу» күйінен «дайындық» күйіне ауысуы) – процессті жүктеу («дайындық» күйінен «орындау» күйіне ауысуы);

-процессті матау (блокировка) («орындау» күйінен «күту» күйіне ауысуы) – процессті матаудан құтқару (разблокировка) («күту» күйінен «дайындық» күйіне ауысу).

Процессті құру және аяқтау операциясы бір тер қана болады (одноразовый), яғни процессе бірден көп қолданбайды (кейбір жүйелік процесстер есептеу жүйесі жұмысы кезінде ешуақытта аяқталмайды, аяқсыз қалады). Ал қалған операциялардың бәрі процесс күйінің өзгеруімен байланысты, яғни олар көп (многоразовый) түрлі. Процессте ОЖ операцияны қалай орындайтынына толығымен тоқталайық.

*Process Control Block және процесс контексті.*ОЖ процессте операцияны орындау үшін әрбір процесстің деректерінің құрылымы көрсетілуі керек. Бұл құрылым берілген процесс үшін спесификалық ақпаратқа толы болады. Олар:

-процесстің сол кездегі күйі;

-процесстің программалық канауышы немесе басқа сөзбен айтқанда келесі орындалатын команда адресі;

-процессор регисторының мазмұны;

-берілген деректер, яғни ол процессті және жадыны басқаруды орындауды жоспарлау үшін қажет (процесс артықшылығы, кеңістіктегі адрестің көлемі және қалай түрғаны);

-есепке алатын деректер (процесстің нөмері, оның жұмысын қолданушы қалай өзгерткені, сол процессте процессордың пайдаланылған жалпы уақыты және т.с.с.).

-процесске байланысты енгізу-шығару құрылғысы туралы мәлімет (мысалы, процесске қандай құрылғы бекітілген, ашық файлдар кестесі).

Оның құрамы және құрылуы нақты бір ОЖ-ге байланысты. Көптеген ОЖ-лерде процессті сипаттайтын ақпарат бір ғана емес, бірнеше деректердің байланысқан құрылғысында сақталады. Бұл құрылыстар әр түрлі атауға ие болады және қосымша ақпаратқа бай болады немесе керісінше сипатталған ақпараттың тек бір бөлігі ғана болады. Бізге оның әсері тимейді. Бізге керек ең маңыздысы – есептеу жүйесіндегі кез-келген процесс үшін барлық ақпарат операцияны әрі қарай дамыту үшін өте қажеттігі. Қарапайым тілде ақпарат бір деректер құрылымында сақталады деп есептейміз. Оны PSB (Process Control Block) немесе процессті басқаратын блок деп атаймыз. Процессті басқаратын блок ОЖ үшін процесстің моделі болып табылады. ОЖ процессінде жүргізілетін кез-келген операция РСВ-да белгілі бір өзгерістің болуын тудырады. РСВ құрамындағы қабылданған модель процесс күйі мен операция арасында түрақты болып қалады.

NT орындалу жүйесі – ығыстыратын көпесептілік жүйесі және Windows негізгі ортасы мен Win32 бағыныңқы жүйесі болып табылады. Ығыстырмайтын Windows версиясында MS-DOS-қа көпесептілікке жету үшін, ағын өз бетінше санашықтың басқаруын беру қажет. Нашар программалар басқа қолданбаларға немесе жүйеге кедергі жасап санашықты жаулап алуы мүмкін еді.

Кейбір кезде екі ағынға жалпы бір мақсатқа жету үшін бір бірімен әрекеттесуі қажет. Мысалы, С компиляторында бір ағын С тілінде программаның өндеуін орындаса, ал екінші ағын – бірінші ағынның жұмысының нәтижесін қабылдап және оны объективті кодқа компилляциялауы мүмкін. Бұл екі ағынға бір бірінің деректерінің алмасу тәсіліне қажет.

Санашықтың өндеуі және программаның компилляциясы екі үрдіспен (әр қайыссының бір ағыны бар), бір үрдістің орындалуына қарағанда тез болуы керек, өйткені көпесептілік операциялық жүйеде санашықтың ағынын немесе компиллятордың ағынын кезек кезек орындауы мүмкін. Санашық бірігіп қолданатын буферге бір нәрсе орналастырса, компилятор өзінің жұмысын бастауы мүмкін. Осыған ұқсас екі немесе бірнеше орында бір уақытта орындалатын қолданбаларды, *паралельді қолданбалар дейміз.* Бір санашықты компьютерде паралельді қолданбалар пайдалы, ал көпсанашықтыларда ол өте пайдалы.