ӘОЖ 378:865.336.3

Б.К.Шаяхметова, К.С.Шаукенова, Г.Ш.Искакова, Н.Т.Орумбаева

*Е.А.Бөкетов атындағы Қарағанды мемлекеттік университеті*
*(E-mail: OrumbayevaN@mail.ru)*

# Күрделі жүйелер үшін бағдарламалық нәтижелерді құру

Күрделі жүйелерге бағдарламалық нәтижелерді құру үшін, оны, біріншіден, ұсақтап бөлу (декомпозиция процесі) қажет, ал кейін есептеп, міндетті қою керек. Мысалы, табиғатта және техникада көпшілік күрделі жүйелер ішкі сатылы құрылымдардан тұрады. Мақалада бағдарламалық жасақтаманы құруда қолданылатын әдістемелер қарастырылған, блоктық-сатылы амалдар, есептеу міндеттердің негізі тәсілдері тұжырымдалған, күрделі бағдарламаның жиындық декомпозициялық үрдісі зерттелген.

*Кілт сөздер:* иерархия, блок, декомпозиция, жобалау, бағдарламалық жиынтық, құрылым, үрдіс (процесс), нысан (объект), қарапайым, күрделі.

Бұл жұмыста сапалы бағдарламалық қамтуды құруда қолданылатын әдістер келтірілген. [1] қарастырылған бағдарламалаудың технологиясын оқытудың кейбір сұрақтары зерттелген, ойлардың жалғасы болып табылады.

Күрделі жүйе үшін бағдарламалық нәтижелерді құру үшін, алдымен, оны неғұрлым ұсақ бөліктерге (декомпозиция үдерісі) бөліп, содан кейін есептеп, міндетін қою керек.

Блоктық-иерархиялық амалдар қарастырып, есептер қойылуының негізгі мәселелерін тұжырымдаймыз.

Күрделі жүйелердің көпшілігі табиғатта және техникада ішкі сатылы құрылымдардан тұратыны белгілі. Бұл күрделі жүйелердің элементтерінің байланысы, әдетте, түріне де, күшіне де қарағанда әр түрлі болатынына байланысты. Осы байланыс жүйелерді өзара байланысты ішкі жүйелердің қандай да бір жиынтығы ретінде қарастыруға мүмкіндік береді. Осындай ішкі жүйелердің элементтерінің ішкі байланысы ішкі жүйелердің өзара байланысынан әлдірек болады. Мысал ретінде процессордан, сыртқы құрылғылардан, сақтау жүйесінен тұратын компьютерді немесе Күнді және Күнді айналып тұратын планеталарды қамтитын Күн системасын келтіруге болады.

Ішкі жүйелерге ажырату, байланыстың әр түрлілігі сияқты, әрбір ішкі жүйелерді ішкі жүйелерге ең төменгі «қарапайым» деңгейге дейін бөлуге мүмкіндік береді. Бөлшектерін (компоненттерін) қарапайым деп санауға болатын деңгейді анықтау — зерттеушінің еркінде. Қарапайым деңгейде жүйе әр түрлі топталған және ұйымдастырылған аздаған ішкі жүйелерден құралады. Осындай түрдегі иерархия «бүтін-бөлік» деген атқа ие болады.

Жүйенің жұмысы оның жеке бөліктерінің жұмысынан күрделі болады, күрделілігі жүйенің неғұрлым әлді ішкі байланыстарының ерекшеліктеріне, негізінен оның бөліктерінің қатынасының шарттарына байланысты болады.

Біздің жағдайымызда иерархияға қатысты келесі орын алады: табиғатта тағы да бір иерархия — «қарапайым-күрделі» иерархия орын алады, немесе эволюция үдерісінде жүйе дамуындағы иерархия (күрделенген). Бұл иерархияда кез келген бейнеленуші жүйе неғұрлым қарапайым жүйенің дамуының нәтижесі болып табылады. Иерархияның осы берілген түрі нысанды-бағытталған бағдарламаның зерттеу механизмімен таратылады.

Блоктық-сатылы (блоктық-иерархиялық) амалдар анықтамасына көшейік. Табиғаттық және техникалық жүйелердің мәндік дәрежеде бейнесі және бағдарламалық жүйелер, әдетте, жоғарыда айтылған қасиеттерге ие болғандықтан, иерархиялық болып табылады. Иерархиялық жүйелердің осы қасиеттерге сүйеніп, блоктық-сатылы (блоктық-иерархиялық) амал құрылады. Біз блоктық-сатылы амалдарды зерттеумен немесе құрумен айналысамыз. Бұл амалда алдымен осындай нысанның бөліктерін (блоктар, модульдер) құрып, содан кейін олардан нысанды жинау керек [2].

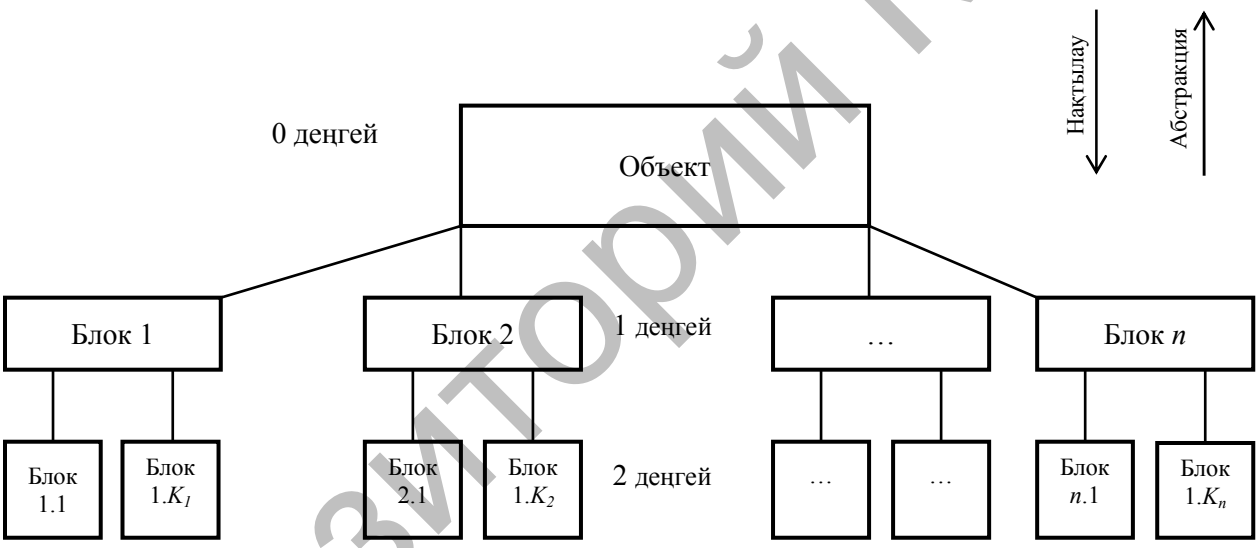
Сонымен бірге біз келесі анықтама беруіміз қажет: күрделі нысанды салыстырмалы түрде тәуелсіз бөліктерге бөліктеу үдерісі декомпозиция деп аталады. Декомпозиция кезінде әр түрлі бөліктер арасында байланыс, бөліктер ішіндегі элементтер байланысына қарағанда, әлсіз болу керектігі ескеріледі. Сонымен бірге алынған бөліктерден дайындалатын нысанды жинау үшін декомпозициялық процесс барысында барлық өзара байланыс түрлерін анықтау қажет.

Өте күрделі нысандарды құру барысында декомпозициялық процесс бірнеше рет орындалатынын ескеру керек. Әрбір блок салыстырмалы түрде дайындауға жеңіл блок алғанға дейін бөліктерге бөлшектенеді. Дайындаудың бұл әдісі қадамдап декомпозициялау деп аталады. Декомпозициялық процесс кезінде жалпы түрде дайындауға болатын ұқсас блоктарды ажыратып алу керек.

Олай болса, жоғарыда айтылғандай, бөліктерге бөлу кодтардың қайталану дәрежесін жоғарылатады, сонымен жұмыстың бағасын төмендетеді.

Енді «иерархия» түсінігінің анықтамасын берейік. Блоктық-сатылы тәсілдемеде блоктарды сипаттауда абстрактылы және нақтылы арақатынас сызбасын құра білу қажет. Ол жайында төменде сөз болады. Декомпозиция нәтижесі, әдетте, иерархия сызбасы түрінде болады. Төменгі деңгейде салыстырмалы түрде — қарапайым блоктарды, ал жоғарғы деңгейде дайындалушы нысан орналастырылады. Әрбір иерархиялық деңгейде блоктарды сипаттау қандай да бір бөліктеу дәрежесіне байланысты кішкене бөлшектерден абстракты түрде орындалады. Осыдан әрбір деңгейге әрбір блокта орындалатын үдеріс маңыздылығын бейнелейтін құжаттың өзіне арналған түрін және өз моделін қолданады.

Бұл жерде алдағы пайымдауға қажетті қандай да бір ереже тұжырымдау керек. Тек қана нысан үшін жалпы талаптар ғана, толығымен тұжырымдау мүмкін болады. Ал төменгі деңгейдегі блоктардың өзгешеліктерін арнайы анықтап, олардан, шынымен, жұмыс істейтін нысан жинауға болатындай тұжырымдау қажет. Басқаша айтқанда, блок саны көп болса, оның сипаттау да неғұрлым абстрактылы болу тиіс (сур. қара).




Сурет. Блоктық-сатылы тәсілдемеде блоктарды сипаттауда абстрактылы және нақтылы арақатынас

Олай болса, блоктық-иерархиялық тәсілдеме негізінде декомпозиция және иерархиялық реттеу жатады. Сонымен бірге келесі қағидалар маңызды роль атқаратынын ескеру қажет:

– қайшылықсыздық — элементтердің өзара келісуін бақылау;
– толықтық — артық элементтердің қатысуын бақылау;
– нысандандыру — әдістемелік тәсілдеме қатаңдығы;
– қайталану — дайындауды арзандату және тездету үшін бірдей блоктарды ажырату қажеттігі;
– жергілікті оңтайландыру — иерархия дейгейінің шектеу оптимизациясы.

Жоғарыда айтылғандар негізінде екі түсінік енгізуге болады. Біріншісі — модуль тілдерінің, есепті қоюдың, қандай да бір иерархиялық деңгейдің сипаттау әдісінің жиынтығын жобалау деңгейі деп атайды.

Екіншісі — жобалау үрдісінде әрбір нысанды ережеге сәйкес жан-жақты қарастыруға тура келеді. Нысанға әр түрлі көзқарасты жобалау аспектісі деп атайды.

Олай болса, жоғарыда айтылғандарды ескеріп, жобалау парадигмасын тұжырымдайық: бағдарламалық жүйелерге блоктық-иерархиялық тәсілдемені қолдану тәсілдеменің жалпы ережелерін нақтылағаннан кейін және жобалау үрдісіне қандай да бір өзгерістер енгізуден кейін ғана мүмкін

болды. Сонымен бірге құрылымдық тәсілдеме иерархияның «бүтін-бөлік» қасиетін ескереді, ал нысандық сонымен бірге иерархияның «қарапайым-күрделі» қасиетін қолданады.

Енді, блоктық-иерархиялық тәсілдеме енгізгеннен кейін жобалау тәсілдемесі түсінігін қарастырамыз. Сонымен, кез келген күрделі бағдарламалық жиынтық жобалау тәсілдемесінің негізіне декомпозиция әдісі жатады (оның неғұрлым қарапайым бөліктерге — компоненттерге, модульдерге бөліктеуі).

Мұнда жобалау үрдісінің негізгі кезеңдерін атап өту қажет;

1. Бағдарламалық жиынтық архитектурасын жобалау.

2. Бірінші кезеңде ажыратылған компоненттердің сыртқы спецификасын (айрықшалығын) дайындау.

3 Компоненттердің құрылымын жобалау.

4. Жобалаудың үшінші кезеңінде ажыратылған әрбір компоненттің құрылымдық бірлігін (ішкі бағдарлама, кластар, модульдер) спецификациясын (айрықшалығын) дайындау.

5. Әрбір компонентте ажыратылған құрылымдық бірліктің алгоритмдерінің және берілгендерінің құрылымын жобалау.

Одан кейін жоғарыда келтірілген кезеңдерді таладау керек. Алғашқы екі кезең міндетті болып табылмайды және бірінен тәуелсіз компоненттерге (мысалы, бағдарлама жүйесі текстік редакторға, файлдармен басқару, компилятор, анықтама жүйесі және т.с.с) қатысты бөлшектенетін үлкен бағдарламалық жүйелер жобалау кезінде ғана орындалады.

Келесі, үшінші, кезеңді жеке қарастыру керек. Себебі ол жобалау үрдісі үшін негізгі болып табылады. Олай болса, бағдарламалық жиынтық архитектурасын жобалау кезеңінде келесі анықталады:

– әрбір компонентпен орындалатын функциялар;
– компоненттер арасындағы дәл және бірмәнді түйіндестер (интерфейстер);
– компоненттер арасындағы басқару құрылымы;
– мәліметтер ағымының құрылымы;
– асинхронды (параллель) орындалатын үрдістер иерархиялық құрылымы (егер ондай қарастырылса);
– компоненттер арасындағы оперативтік жадын үлестіру құрылымы;
– бөлінетін құрылғылар компоненттерін қолдану құрылымы (мысалы, ішкі коммуникациялар).

Бағдарламалық жиынтық архитектурасын жобалау нәтижесі оның компоненттерінің сыртқы айрықшалығына әсер етеді. Бір кезеңнен келесі кезеңге ауысуын бақылап отыру керек. Сонымен бірге бағдарламалық жүйенің компоненттерінің құрылымын жобалауға көңіл бөлу қажет. Мақсат — компоненттердің барлық құрылымдық бөліктерін (оларды құрылымдық бірліктер деп атаймыз), олардың иерархиясын және олардың арасындағы интерфейстерді анықтау. Орындау нәтижесі осының негізінде құрылым және жұмыс алгоритмі жобалауы орындалатын құрылымдық бірліктер қасиетінің айрықшылығы түрінде көрінуі тиіс.

Әдебиеттер тізімі

1 *Шаяхметова Б.К.* Пути повышения качества создания программного обеспечения для информационных специальностей университета // Классический университет в парадигме современных знаний. — Караганда: Изд-во КарГУ, 2012. — С. 376–378.

2 *Хорев П.Б.* Объектно-ориентированное программирование: Учеб. пособие для студ. учрежд. высш. проф. образования.— 3-е изд., испр. — М.: Академия, 2011. — 448 с.

Б.К.Шаяхметова, К.С.Шаукенова, Г.Ш.Искакова, Н.Т.Орумбаева

## Проектирование программного продукта для сложных систем

Для создания программного продукта сложной системы необходимо сначала разбить ее на более мелкие части (процесс декомпозиции), а затем сделать постановку задачи. Подавляющее большинство сложных систем как в природе, так и в технике имеет иерархическую внутреннюю структуру. В статье освещены приемы, применяемые при создании программного обеспечения, рассмотрен блочно-иерархический подход, сформулированы основные положения постановки задачи, показан процесс декомпозиции сложного программного комплекса.

B.K.Shayakhmetova, K.S.Shaukenova, G.Sh.Iskakova, N.T.Orumbayeva

## Design software for complex systems

To create a complex software system, in the first place is to divide it into smaller parts (decomposition process), and then make a formulation of the problem. The vast majority of complex systems, both in nature and in technology have a hierarchical internal structure. The article highlights the techniques that involved in creating software, block-hierarchical approach are considered, the basic situation of formulation of the problem are given, the process of decomposition of complex software systems are mentioned.

### References

1 Shayakhmetova B.K. *The ways to improve the quality of software development for information university's specialties / Classical university in the paradigm of modern knowledge,* Karaganda: Publ. KSU, 2012, p.376–378.

2 Horev P.B. *Object-oriented programming*, The manual for students of higher education institutions, 3rd pub., rev, Moscow: Akademia, 2011, 448 p.