АВТОМАТИЗИРОВАННАЯ ИНФОРМАЦИОННАЯ СИСТЕМА ВУЗА И ОСНОВНЫЕ ТРЕБОВАНИЯ ПРЕДЪЯВЛЯЕМЫЕ К НЕЙ

Г. А. МЕЛИКЯН

Работа посвящена описанию общих принципов разработки автоматизированной информационной системы ВУЗа. Описаны базовые структурные подсистемы и приведены основные требования, которым должна удовлетворять разрабатываемая система.

HEI'S AUTOMATED INFORMATION SYSTEM AND ITS BASIC REQUIREMENTS

H. A. MELIKYAN

This paper describes the general principles of developing an automated information system of the university. It also describes the basic structural subsystems and provides the basic requirements that must be satisfied by the developed system.

ԿԱՌԱՎԱՐՄԱՆ ԱՎՏՈՄԱՏԱՑՎԱԾ ՀԱՄԱԿԱՐԳԵՐԻ ԾՐԱԳՐԱՅԻՆ ԱՊԱՀՈՎՄԱՆ ՃԱՐՏԱՐԱՊԵՏՈՒԹՅԱՆ ՆԱԽԱԳԾՄԱՆ ՀԱՐՑԵՐԸ

Հ. Ա. ՄԵԼԻՔՅԱՆ
Տեխն. գիտ. թեկն., դոցենտ, ԳՊՀ պրոֆեսոր

Կառավարման ավտոմատացված համակարգի (ԿԱՀ) ծրագրային ապահովումը (ԾԱ) կամ ինչպես երբեմն անվանում են՝ ծրագրային միջոցը (ԾՄ) ծրագրերից, ընթացակարգերից, կանոններից, ինչպեց նաև, եթե նախատեսված է, դրանց ուղեկցող փաստաթղթերից և տվյալներից կազմված ամբողջություն է, որով իրականացվում են ինֆորմացիայի մշակման համակարգի հաշվողական գործընթացները: ԿԱՀ-ի ծրագրային ապահովումը բաժանվում է երկու հիմնական մասի.

- ընդհանուր կամ ներքին ԾԱ,
- հատուկ կամ կիրառական ԾԱ:

Ընդհանուր ծրագրային ապահովումը նախատեսված է հաշվողական գործընթացների կազմակերպման, ինչպես նաև ծրագրային ապահովման ստեղծման և ստուգման տեխնոլոգիական գործընթացների ավտոմատացման համար:

Հաշվողական գործընթացների կազմակերպման ծրագրային ապահովումը (օպերացիոն համակարգը) իրականացնում է ինֆորմացիայի մուտք - ելքի կազմակերպման, ընդհատումներին արձագանքման, հաշվողական պրոցեսի և տեխնիկական միջոցների ֆունկցիոնալ վերահսկման, խնդիրների և հանձնարարությունների գերակայության սկզբունքով կատարման, ինֆորմացիայի վերա-

կանգնման և ավտոմատացման միջոցների փոխդասավորության փոփոխության և այլ ֆունկցիաներ:

Ծրագրային ապահովման ստեղծման և ստուգման տեխնոլոգիական գործընթացների ավտոմատացման համալիրը բաղկացած է ծրագրավորման ապահովման, ծրագրերի կարգաբերման, փաստաթղթերի թողարկման և շտկման, ծրագրային ապահովման մշակման ընթացքի ստուգման և այլ միջոցներից:

Հատուկ կամ կիրառական ծրագրային ապահովումը (ԿԾԱ) ծառայում է ԿԱՀ-ի ինֆորմացիայի հավաքման, մշակման և արտապատկերման կոնկրետ ֆունկցիոնալ խնդիրների իրականացման համար:

ԿԱՀ-ի ընդհանուր ԾԱ-ն հիմնականում հավաստագրված ծրագրային պատրաստի փաթեթներ են, որոնք ձեռք են բերվում և տեղադրվում ԿԱՀ-ի օբյեկտներում՝ տեխնիկական միջոցների հետ միասին: Հետևաբար ԾԱ-ի ստեղծման աշխատանքների հիմնական մասը կիրառական ծրագրային ապահովման մշակումն է, որը, բավականաչափ մեծ համակարգերի համար, երկարատև և աշխատատար գործընթաց է: Ուստի կարելի է ասել, որ ԿԱՀ-ի գործառության արդյունավետությունը մեծապես կախված է այն բանից, թե ինչպիսին է ԿԾԱ-ի ճարտարապետությունը, խնդիրների և համակարգային ռեսուրսների ինչպիսի բազմություններ են ընտրված, ինչպես են կազմակերպված հաշվողական գործընթացները:

ԿԾԱ-ի ճարտարապետությունը նրա ներկայացումն է փոխգործող ենթահամակարգերի ինչ-որ ամբողջության տեսքով: Որպես այդպիսի ենթահամակարգեր հաճախ հանդես են գալիս առանձին ծրագրերը (ծրագրային բաղադրիչները):

ԿԾԱ-ի ճարտարապետության մշակման հիմնական խնդիրներն են.

- ծրագրային ենթահամակարգերի առանձնացում և դրանց վրա տեխնիկական առաջադրանքով տրված ֆունկցիաների արտապատկերում և ենթահամակարգերի ֆունկցիոնալ մասնագրերի մշակում,
- առանձնացված ծրագրային ենթահամակարգերի միջև փոխազդեցության եղանակների որոշում:

Այս փուլում ընդունված որոշումների հիման վրա կարող է կատարվել, անհրաժեշտության դեպքում, ֆունկցիոնալ մասնագրերի հետագա կոնկրետացում:

Տարբերվում են ԿԾԱ-ի ճարտարապետության հետևյալ հիմնական տիպերը.

- ամբողջական ծրագիր,
- ինքնավար կատարվող ծրագրերի համալիր,
- շերտավոր ճարտարապետություն,
- զուգահեռ կատարվող ծրագրերի համալիր:

Ամբողջական ծրագիրը ԿԾԱ-ի ճարտարապետության այն տիպն է, երբ այն պարունակում է միայն մեկ ծրագիր: Այդպիսի ճարտարապետություն ընտրում են այն դեպքերում, երբ ԿԾԱ-ն պետք է կատարի վառ արտահայտված որևէ ֆունկցիա, որի իրականացումը իրենից դժվարություն չի ներկայացնում: Բնական է, որ այդպիսի ճարտարապետությունը չի պահանջում որևէ նկարագրություն, քանի որ դա միայնակ ծրագիր է, որը կապված չէ արտաքին ֆունկցիաների հետ և, ուրեմն, չի պահանջում փոխազդեցության եղանակի որոշում: Միայն ապահովում է փոխազդեցությունը օգտագործողի հետ և այդ փոխազդեցությունը պետք է նկարագրված լինի ԾԱ-ի կիրառման փաստաթղթերում:

Ինքնավար կատարվող ծրագրերի համալիրը կազմված է այնպիսի ծրագրերի հավաքածուից, որ՝

- այդ ծրագրերից յուրաքանչյուրը կարող է օգտագործողի կողմից ակտիվացվել (գործարկվել) մյուս ծրագրերից անկախ,
- ակտիվացված ծրագրի կատարման ընթացքում այդ հավաքածուի այլ ծրագրեր չեն կարող ակտիվացվել, մինչև չավարտվի ակտիվացված ծրագիրը (բազմախնդիր համակարգերում այս պահանջը կարող է չկատարվել),
- այդ հավաքածուի բոլոր ծրագրերը կիրառվում են միևնույն ինֆորմացիոն միջավայրում:

Այսպիսով, այդ հավաքածուի ծրագրերը, որպես կանոն, ըստ կառավարման չեն փոխազդում, նրանց միջև փոխազդեցությունը իրականացվում է միայն ընդհանուր ինֆորմացիոն միջավայրի միջոցով:

Ծրագրային շերտավոր ճարտարապետությունը կազմված է ծրագրային ենթահամակարգերի ինչ-որ կարգավորված ամբողջությունից, որոնք կոչվում են շերտեր (նկար 1), այնպիսիք, որ՝

- յուրաքանչյուր շերտում ոչինչ հայտնի չէ հաջորդ (ավելի բարձր) շերտերի հատկությունների (նույնիսկ գոյության) մասին,
- յուրաքանչյուր շերտ կարող է ըստ կառավարման փոխգործել (դիմել բաղադրիչներին) անմիջապես նախորդող (ավելի ցածր) շերտերի հետ նախօրոք որոշված ինտերֆեյսի միջոցով, ընդ որում՝ ոչինչ չիմանալով նախորդ բոլոր շերտերի ներքին կառուցվածքների մասին,
- յուրաքանչյուր շերտ տիրապետում է որոշակի ռեսուրսների, որոնք նա կամ թաքցնում է ուրիշ շերտերից, կամ տրամադրում է անմիջական հաջորդ շերտին (նշված ինտերֆեյսով) այդ ռեսուրսների որոշ ընդհանրացումներ:

Նկ.1

Այսպիսով, շերտավոր ծրագրային ճարտարաետության մեջ յուրաքանչյուր շերտ կարող է իրականացնել տվյալների մի որոշ աբստրակցիա: Շերտերի միջև կապերը սահմանափակված են պարամետրերի արժեքների և արդյունքների փոխանցումներով, որոնք իրականացնում է տվյալ շերտը՝ իր ներքևի և վերևի կից շերտերին:

Անթույլատրելի է գլոբալ տվյալների օգտագործումը մի քանի շերտերի կողմից: Շերտավոր հիերարխիկ մեթոդի բավականին բնութագրական օրինակ է հանդիսանում ժամանակակից օպերացիոն համակարգերի ճարտարապետությունը: Դրա մեջ ստորին շերտերը իրենցից ներկայացնում են դրայվերները (ապարատուրայի հետ փոխգործող ծրագրային բաղադրիչները), այնուհետև գալիս են պրոցեսների սինքրոնացման և գործակարգավորման, հիշողության կառավարման, օպերատորի հետ կապի, տվյալների հոսքերի կառավարման և այլ ֆունկցիաների համար պատասխանատու շերտերը: Իսկ ամենավերին շերտում գտնվում են օպերացիոն համակարգի և կիրառական ծրագրերի միջև կապ ապահովող բաղադրիչները:

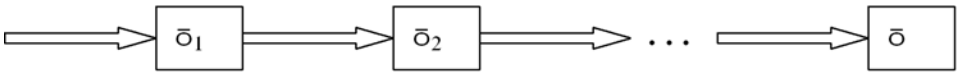
Զուգահեռ կատարվող ծրագրերի համալիրը իրենից ներկայացնում է ծրագրերի հավաքածու, որոնք ընդունակ են փոխազդել իրար հետ, միաժամանակ գտնվելով կատարման փուլում: Դա նշանակում է, որ այդպիսի ծրագրերը՝

• կանչվել և գտնվում են օպերատիվ հիշողությունում, ակտիվացված են և կարող են հերթով ըստ ժամանակի բաշխել մեկ կամ մի քանի կենտրոնական պրոցեսորներ,

• իրականացնել իրար միջև դինամիկ փոխազդեցություններ (կատարման ընթացքում), որոնց հիման վրա կատարվում է նրանց սինքրոնացումը:

Սովորաբար այդպիսի գործընթացների միջև փոխազդեցությունը կատարվում է իրար ինչ-որ հաղորդագրություններ փոխանցելու ճանապարհով:

Այդպիսի ճարտարապետության պարզագույն տարատեսակ է *հոսքագիծը,* որն իրենից ներկայացնում է ծրագրերի հաջորդականություն, որի մեջ յուրաքանչյուր ծրագրի (բացի վերջինից) ստանդարտ ելքը կապված է հաջորդ ծրագրի ստանդարտ մուտքի հետ (նկար 2):



Նկ. 2

Հոսքագիծը մշակում է հաղորդագրությունների ինչ-որ հոսք: Այդ հոսքի յուրաքանչյուր հաղորդագրություն տրվում է առաջին ծրագրի մուտքին, որը վերամշակված հաղորդագրությունը փոխանցում է հաջորդ ծրագրին, իսկ ինքը սկսում է հոսքի հերթական հաղորդագրության մշակումը: Այս ձևով է գործում հոսքագծի յուրաքանչյուր ծրագիր: Հոսքագծի վերջին ծրագիրը արտածում է ամբողջ աշխատանքի արդյունքները:

Այսպիսով, n ծրագրերից կազմված հոսքագծի մեջ կարող են մշակման գործընթացում գտնվել միաժամնակ n հաղորդագրություններ:

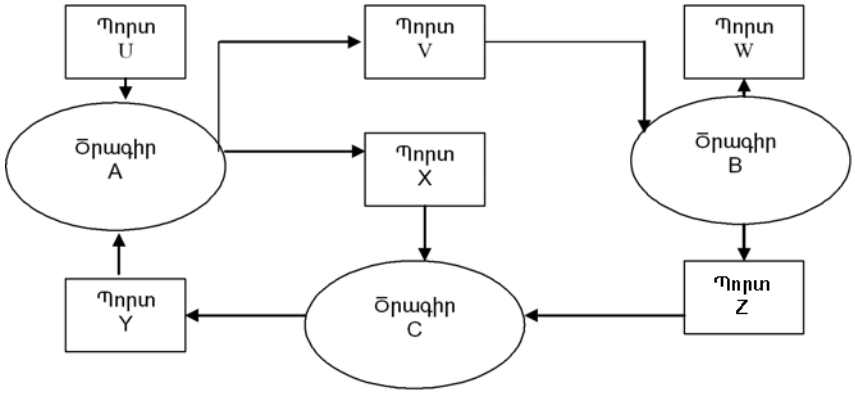
Քանի որ հոսքագծի տարբեր ծրագրեր հերթական հաղորդագրությունների մշակման վրա կարող են ծախսել ժամանակի տարբեր հատվածներ, անհրաժեշտ է ինչ-որ եղանակով ապահովել այդ գործընթացների սինքրոնացումը (որոշ գործընթացներ կարող են գտնվել սպասման վիճակում կամ հաղորդագրություն տալու, կամ ստանալու պատճառով):

Զուգահեռ կատարվող ծրագրերի համալիրն ընդհանուր դեպքում կարող է կազմակերպվել որպես հաղորդագրությունների «փոխանցման խողովակներով» (պորտերով) համակարգ (pipe-and–filter network): Հաղորդագրությունների պորտը (pipe) իրենից ներկայացնում է ծրագրային ենթահամակարգ, որը սպասարկում է հաղորդագրությունների ինչ-որ հերթ: Այն կարող է պահելու համար ծրագրից (filter) ստանալ ինչ-որ հաղորդագրություն և դնել հերթի, ինչպես նաև տալ հերթական հաղորդագրությունը ուրիշ ծրագրի՝ նրա պահանջով: Ինչ-որ ծրագրի կողմից որևէ պորտին տրված հաղորդագրությունը արդեն չի պատկանելու այդ ծրագրին (և չի օգտագործելու նրա ռեսուրսները, բայց չի պատկանելու նաև որևէ այլ ծրագրի): Այն կտրվի որևէ ծրագրի միայն հարցման միջոցով: Այսպիսով՝ ծրագիրը մեկ այլ ծրագրի փոխանցող հաղորդագրությունը տալիս է պորտին և չի սպասում, թե երբ ստացող ծրագիրը պատրաստ կլինի այն մշակելու:

Նկար 3-ում բերված է հաղորդագրությունների պորտերով ծրագրային համա-

կարգի օրինակ, որտեղ U պորտը դիտարկվում է որպես մուտքային հաղորդագրությունների, իսկ W պորտը՝ ելքային հաղորդագրությունների պորտ:

Հաղորդագրությունների պորտերով ծրագրային համակարգը կարող է լինել *կոշտ* կամ *ճկուն* փոխդասավորվածությամբ: Կոշտ դասավորվածության պորտերով համակարգերում յուրաքանչյուր ծրագիր խիստ կապված է մեկ կամ մի քանի մուտքային պորտերի հետ: Հաղորդագրությունը փոխանցելու համար այդպիսի ծրագիրը պետք է բացահայտ նշի փոխանցման հասցեն՝ ծրագրի անունը և նրա մուտքային պորտի անունը: Այս դեպքում պորտերի փոխդասավորվածության փոփոխումը բերում է օգտագործվող ծրագրերի փոփոխման:



Նկ. 3

Ճկուն փոխդասավորվածության պորտերով համակարգերում յուրաքանչյուր ծրագրի հետ կապված են ինչպես մուտքային, այնպես էլ ելքային վիրտուալ պորտերը: Այսպիսի համակարգերի գործարկումից առաջ պետք է կատարվի նրա նախնական կարգաբերումը հատուկ ծրագրային բաղադրիչի օգնությամբ, որն իրականացնում է մի ծրագրի յուրաքանչյուր ելքային վիրտուալ պորտի համընկեցումը մյուս ծրագրի ինչ-որ մուտքային վիրտուալ պորտի հետ: Հետևաբար, այս դեպքում պորտերի փոխդասավորվածության փոփոխությունները կարող են բերել միայն համակարգային կարգավորման աղյուսակների փոփոխության:

ԿԾԱ-ի ենթահամակարգերի միջև փոխազդեցության ապահովումը հիմնականում իրականացվում է ներքին ԾԱ-ի ստանդարտ հնարավորություններով, սակայն որոշ դեպքերում (ինչպես, օրինակ, ինքնավար կատարվող ծրագրերի կամ շերտավոր ծրագրային համակարգերի) կարող է պահանջվել լրացուցիչ ծրագրային բաղադրիչների ստեղծում:

Այսպիսով, ԿԱՀ-ի ԿԾԱ-ի ճարտարապետության վերը նշված որևէ տարբերակով ընտրության ժամանակ անհրաժեշտ է հաշվի առնել ԿԱՀ-ի նպատակային խնդիրների բարդությունը, տեխնիկական միջոցների և ներքին ԾԱ-ի հնարավորությունները, ինչպես նաև ԿԾԱ-ի մշակումը կառավարող մասնագետների փորձառությունը:

ԳՐԱԿԱՆՈՒԹՅՈՒՆ

1. **Лисков Б., Гатэг Д.** Использование абстракций и спецификаций при разработке программ. М.: Мир, 1989.
2. **Марка Д. А., Мак Гоуэн К.** Методология структурного анализа и проектирования. М., МетаТехнология, 1993.
3. **Орлов С. А.** Технологии разработки ПО. СПб., Питер, 2004.

НЕКОТОРЫЕ ВОПРОСЫ ПРОЕКТИРОВАНИЯ АРХИТЕКТУРЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ АВТОМАТИЗИРОВАННЫХ СИСТЕМ УПРАВЛЕНИЯ

Г. А. МЕЛИКЯН

В работе рассмотрены четыре основных типа архитектуры прикладного программного обеспечения автоматизированных систем управления: цельная программа, комплекс автономно выполняемых программ, слоистая архитектура и коллектив параллельно выполняемых программ. Определены основные задачи, особенности и области применения каждого типа архитектуры.

SOME QUESTIONS OF ARCHITECTURE DESIGN FOR SOFTWARE AUTOMATED CONTROL SYSTEMS

H. A. MELIKYAN

The paper discusses four main types of architecture of software application for automated control systems: solid programme, a set of independently running programmes, layered architecture and the team concurrently executing programmes. It also defines the main problems, characteristics and scope of each type of architecture.

ОБ ОДНОМ СПОСОБЕ РАЗРАБОТКИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ОБУЧЕНИЯ

Г. А. МЕЛИКЯН
Кандидат техн. наук, доцент, профессор ГГУ, зав. кафедрой ИВТ
С. Х. НАХАТАКЯН
Кандидат техн. наук, доцент, преподаватель кафедры системного программирования РАУ

На рынке программного продукта за последнее десятилетие появилось достаточно большое количество обучающих систем, в том числе и автоматизированных систем обучения (АСО), которые охватывают различные предметные области, и призваны решать задачи обучения на различных этапах жизни человека - от начальных классов средней школы до обучения в высших учебных заведениях. Вместе с тем, большая часть программ