AZƏRBAYCAN RESPUBLİKASI TƏHSİL NAZİRLİYİ

AZƏRBAYCAN DÖVLƏT İQTİSAD UNİVERSİTETİ

MAGİSTRATURA MƏRKƏZİ

Əlyazması hüququnda

Babaşova Xoşqədəm Rail qızı

"TƏTBIQI PROQRAM PAKETLƏRI GENERASIYA EDƏN SISTEMLƏRIN YARADILMASI TEXNOLOGIYASININ BƏZI PROBLEMLƏRI" *mövzusunda*

MAGİSTR DİSSERTASİYASI

İstiqamətin şifri və adı:	**060509 "Kompüter Elmləri"**
İxtisasın şifri və adı:	**"İqtisadi informasiya sistemləri"**
Elmi rəhbər	**dos. Musayev M.N.**
Magistr proqramının rəhbəri	**dos. Bayramov H.M.**
Kafedra müdiri	**dos. Bayramov H. M.**

BAKI – 2018

MÜNDƏRİCAT

GİRİŞ

Hesab-məntiq problem-yönümlü sistemlərin əsas inkişaf istiqamətlərindən biri də tətbiqi proqramlar paketləri hesab olunur. Çox mürəkkəb daxili quruluşa malik olmaları hesabına proqramlar paketi özünəməxsus xüsusiyyətlərə malikdir. Birinci növbədə proqramlar paketi hər hansı bir bilik sahəsindən məsələləri həmin sahələrin anlayişları ilə şərh etməyə (bütün problem-oriyentasiyalı sistemlərdə olduğu kimi) imkan verir.

Yaranma tarixlərindən və tətbiq olunmasından xeyli vaxt kecməsinə baxmayaraq onların yaranması metodları və strukturları artıq bir neçə inkişaf mərhələsi kecmişdir. Proqramlar paketi sadə altproqramlar toplusundan mürəkkəb proqram kompleksinə çevrilmişdir. Bu kompleksin həll olunacaq məsələləri həll etmək üçün öz xüsusi giris dili vardir.

İlk əvvəl yaradılan Elektron Hesablama Maşınlarda tez-tez istifadə olunan altproqramlar toplanmağa başlamışdı. Belə altproqramlara misal olaraq elementar funksiyalaları hesablayan, ədədlərin çevrilməsini həyata kecirən və başqa alt proqramları göstərmək olar. Bu proqramlar bəzi standart razılaşmalar əsasında yaradıldıqlarına gorə onlara standart altproqramlar deyilirdi. Bu proqramlar standart altproqramlar kitabxanalarında saxlanılırdı. Sonralar müxtəlif tətbiq sahələrindən olan məsələlərin həlli proqramları da proqramlar kitabxanalarında saxlanmağa başlandı və onlar tezliklə tətbiqi proqramlar paketi adlandırıldı. Xarici firmalar istehsal etdikləri EHM-lərlə birlikdə müxtəlif tətbiqi proqramlar paketləri də satmağa başladılar. Lakin sonralar təcrübə göstərdi ki, bu paketlər həyatda o qədər də əhəmiyyətli deyil. Keçmiş ittifaqda statistika məsələlərini, xətti və matrislər cəbri məsələlərini həll edən tətbiqi proqramlar paketləri geniş yayılmışdı. Bu paketlər sadə quruluşlu proqramlar kitabxanası şəklində idi.

Hesab-məntiq problem-yönümlü sistemlərin bu tipi, yəni tətbiqi proqramlar paketləri praktikada ən çox istifadə olunan proqram kompleksləridir. Tətbiqi proqramlar paketlərinin yaradılması sahəsində bu gün də gərgin iş gedir. Əməliyyat sistemlərinin imkanlarını genişləndirən tətbiqi proqramlar paketləri (məsələn,

Microsoft Office paketi) buna əyani sübutdur. İndi elə bir təşkilat, firma, müəssisə tapılmaz ki, Microsoft Office paketinin tərkibinə daxil olan Word, Excel, Power Point Access, Outlook, Publisher və s. tətbiqi sistemlərdən istifadə etməsin. Tətbiqi proqramlar paketlərinin yaradılması sahəsində bu gün də gərgin iş gedir. Bu isə onu göstərir ki, hesab-məntiq problem-yönümlü sistemlərin, o cümlədən intellektual tətbiqi proqramlar paketlərinin yaradılması problemi hal-hazırda da akdtual olaraq qalır.

Mövzunun aktuallığı. Hesab-məntiq problem-yönümlü sistemlərin, o cümlədən də tətbiqi proqramlar paketlərinin yaradılmasında məqsəd tətbiqi problemlərin həlli texnologiyasının məlum mərhələlərini avtomatlaşdırmaqla kompüterləri müxtəlif tətbiq sahələrində çalışan istifadəçilərə yaxınlaşdırmaqdır.

Təbiidir ki, bu tətbiqi proqramlar paketləri elə tərtib olunurdu ki, bir məqsədlə istifadə olunan verilənlər eyni təsvir formasına malikidilər. Bununla da bir paketin proqramlarının birgə işləyə bilməsi təmin olunurdu. Belə paketlərdən istifadə etmək üçün kataloqda lazım olan modulların adları tapılırdı və onları bir yerə toplayan (çağıran) baş proqram əllə yazılırdı.

Tətbiqi proqramlar paketə daxil edilən alt proqramlar digər adla, modullar adı ilə adlandırılırlar , belə ki, bu alt proqramlar baxılan məsələnin həlli proqramının müəyyən hissələridirlər və bu modullar tətbiqi proqramlar paketinin daxilində "bir yerdə işləyə bilmək" tələbini ödəməklə tərtib olunular.

Belə xüsusu tipə malik tətbiqi proqramlar paketlərdən hal-hazıra qədər də müəyyən tətbiq sahələrdə istifadə olunurlar. Belə sahələrə misal olaraq hesablama xarakterli məsələrin həll olunduğu nəzəri və praktiki fizikanın müxtəlif bölmələrini göstərmək olar. Bu halda mürəkkəb struktura malik tətbiqi proqramlar paketlərinin yaradılması əlverişli deyildir. Çünki əlavə modulların yaradılması müəyyən qədər xərc tələb edir və sonradan bu xərc ödənilmir.

Tətbiqi proqramlar paketlərinin inkişafında növbəti mərhələ paketə yeganə təşkiledici proqram daxil etmeklə proqram modullarının cağırılması və birləşdirilməsinin avtomatlaşdırılması hesab olunur. Proqramçı tərəfindən tərtib olunan məsələnin həlli proqramında yalnız paketin təşkiledici proqramına müraciət

nəzərdə tutulurdu. Bu zaman hansı modulların lazım olduğu haqqında zəruri olan informasiyadan minimum istifadə olunurdu. Belə tətbiqi proqramlar paketlərində modullardan rekursiv şəkildə istifadə etməyə imkan yaranmışdır. Bele sistemə BESM -2 Elektron Hesablama Maşını üçün yaradılan IS-2 proqramlaşdırma sistemini göstərmək olar.

Paketə təşkiledici proqram (monitor) daxil edildikdən sonra çox sadə şəkildə verilənlərin idarə olunmasının bəzi funksiyalarının onun üzərinə qoymaq mümkün oldu. Proqramlar paketində yeni komponent, daha doğrusu tətbiqi proqramlar paketinin proqramlarının istifadə etdiyi verilənlər toplusunu saxlayan komponent əmələ gəldi. Tətbiqi proqramlar paketə belə komponentin daxil edilməsi təşkiledici proqramda, yəni monitorda çox da boyük olmayan dəyişiklik edilməklə əldə olunmuşdur. Lakin bu tətbiqi proqramlar paketinin istifadəsini, hər bir məsələnin həlli üçün daxil edilən verilənlər toplusunun tutduğu yaddaşın həcmini kiçiltməklə, müəyyən qədər sadələşdirmişdir.

Tətbiqi proqramlar paketlərinin inkişafında növbəti uğur özü proqram yarada bilən, yəni proqram generasiya edən tətbiqi proqramlar paketlərinin meydana gəlməsi hesab olunur. Bu tətbiqi proqramlar paketləri modullar arasında əlaqə yaratmaqdan başqa, qoyulmus məsələnin həlli üçün hansı proqram modullarının istifadə üçün lazım olduğunu da müəyyən edirlər. Belə təşkiledici proqramların meydana gəlməsinin əsas səbəbi qoyulmuş məsələni tətbiq sahəsində istifadə olunan anlayişların kpməyi ilə problem – yönümlü dildə şərh etmək zərurəti idi. Belə olan halda artiq qoyulmuş məsələnin həlli proqramı tətbiqi proqramlar paketində olan modullardan əllə tərtib olunmur. Bu proqram məsələnin şərhinə əsasən avtomatik generasiya olunur. Məsələnin şərhi paketin istifadəçisi tərəfindən hazırlanır və idarəedici proqram vasitəsilə paketin giriş dilindən müəyyən proqramlaşdırma dilinə cevrilir və məsələnin həlli proqramı hesab olunur. Paketlərin inkişafının bu mərhələsində onların giriş dili anlayişı meydana çıxır. Paketin giriş dilində qoyulan məsələlər şərh olunurlar. İdarəedici proqram belə tətbiqi proqramldar paketlərində həm də giriş dilinin translyatoru funksiyasını yerinə yetirir. Birinci bu tipdən olan

sistemlərə misal olaraq proqramla idarə olunan dəzgahlar üçün yaradılmış proqramlaşdırmanın avtomatlaşdırılması sistemlərini göstərmək olar.

Qeyd edək ki, özü proqram yarada bilən, yəni generasiya edən tətbiqi proqramlar paketlərinin təşkiledici proqramları müxtəlif struktura və iş prinsipinə malikdirlər. Ən əsas məsələ ondan ibarətdir ki, bu tətbiqi proqramlar paketlərinin təşkiledici proqramı qoyulmuş məsələnin həlli üçün hansı modulların tətbiq oluna bilməsi informasiyasından istifadəedə bilməlidir. Bu informasiyalar tətbiqi proqramlar paketini yeni elementində, məsələnin həll olunduğu bilik sahəsinin "semantik model" adlanan elementində saxlanılır.

Qeyd etmək lazımdır ki, bir sıra tətbiqi proqramlar paketlərində təşkiledici proqram «qoyulmuş məsələ həlledicisi» rolunu oynayır. Bu tətbiqi proqramlar paketlərində təşkilediciedici proqram həll olunan məsələ haqqında yeni bilikləri "semantik model" və "verilənlər" şəkllərində yaddaşda saxlamaqla öyrənə bilmək imkanına malikdir. Aydındır ki, belə tətbiqi proqramlar paketləri yaradılarkən "süni intellekt" sahəsindən alınan nəticələrdən istifadə olunurdu. Tətbiqi proqramlar paketlərinin inkişafının bu dönəmində onların verilənlər bazaları ilə çox yüksək dərəcədə yaxınlaşması hiss olunur. Son zamanlar yaradılan, yəni generasiya olunan tətbiqi proqramlar paketlərinin giriş dilinin təbii dilə daha da yaxınlaşdırılmasına səylər göstərilir və tətbiqi proqramlar paketləri ilə işləmək üçün interaktiv rejimdə işləməyə daha çox üstünlük verilir.

Beləliklə, tətbiqi proqramlar paketi dedikdə, bir-birilə birlikdə işləmək xüsusiyyəti olan, təşkiledici proqramın idarəsi altında fəaliyyət göstərən və müəyyən tətbiq oblastından olan məsələlərin həlli üçün təyin olunmuş proqramlar kompleksi başa düşülür.

Tətbiqi proqramlar paketi müəyyən tətbiq oblastından olan məsələlərin həlli üçün yaradılan proqramlar kompleksinin kompüterin xarici yaddaşında saxlanılmasının təşkili forması hesab olunur.

Nəzərə almaq zərurudir ki, tətbiqi proqramlar paketlərindən istifadə edərkən mütləq «məsələnin sərhi» verilməlidir. Məsələnin şərhi ya hər hansi bir proqram formasında, yaxud məsələnin həlli üçün zəruri hesab edilən «məsələnin şərtləri

siyahisi» formasında ola bilər. Bir birindən xeyli fərqlənən struktura malik tətbiqi proqramlar paketlərə, həm də məsələnin şərhinin müxtəlif təsvir formalarına rast gəlindiyindən, tətbiqi proqramlar paketlərinin giriş dilləri bir-birlərindən sintaksis və semantikalarına və mürəkkəb quruluşlarına görə kəskin sürətdə ferqlənə bilərlər.

Hal-hazırda istifadə olunan tətbiqi proqramlar paketlərinin demək olar ki, əksəriyyəti interaktiv rejimində işləyən intellektual proqram kompleksləridir. Bu komplekslərin təşkiledici proqramı aşağıdakı funksiyaları yerinə yetirir: istifadə olunan əməliyyat sistemi ilə əlaqə yaradaraq tətbiqi proqramlar paketinin bütün işini idarə edir; tətbiqi proqramlar paketinin giriş dilində tərtib olunmuş həll olunacaq məsələni qəbul edərək onu kompleksin daxili dilinə tərcümə edir; qoyulmuş məsələni həll etmək üçün zəruri olan modullar ardicilliğını müəyyən edir; həll üçün zəruri olan modulları birləşdirərək məsələnin həlli proqramını formalaşdırır.

Məlumdur ki, bütün tətbiqi proqramlar paketləri arxitekturu iki komponentdən ibarətdir: tətbiq oblastından və həlli tələb olunan məsələlərdən asılı olan funksional komponent; tətbiqi proqramlar paketinin butün işini idarə edən sistem komponenti. Sistem komponenti həm də idarəedici proqram, təşkiledici proqram, monitor, supervizor, dispetçer adlanır. Tətbiqi proqramlar paketinin sistem komponentinə paketin giriş dilinin translyatoru, planlasdırıcısi, kompilyatoru, işçi proqramı, paketin giriş dilindəki sərhi, məsələnin paketin daxili dilindəki sərhi, məsələnin həlli alqoritmi, başlamğıc verilənlər, paketin işinin son nəticəsi daxil edilir. Həlli tələb olunacaq məsələlərdən asılı olan funksional komponent isə tətbiq sahəsinin modeli və funksional modullardan təşkil olunmuşdur.

Tətbiqi proqramlar paketin giriş dilində «tərtib olunmuş» məsələ translyator tərəfindən tətbiqi proqramlar paketinin daxili dilinə çevrilir. Aydın məsələdir ki, bu mərhələdə paketin tətbiq sahəsinin semantik modelindəki anlayışların semantikası nəzərə alınır. Həll olunan məsələnin paketin daxili dilindəki sərhindən istifadə etməklə planlaşdırıcı modulların icrası ardıcıllığını müəyyən edir və daxili dildə məsələinin həlli alqoritmini qurur. Alınan alqoritmə əsaslanaraq paketin

kompilyatoru məsələnin həlli proqramini tərtib edir. Yaranan bu proqram artıq yerinə yetirilməyə hazır olan işçi proqramdır.

Tədqiqatın məqsədi. Hesab-məntiq problem-yönümlü sistemlərin, o cümlədən intellektual tətbiqi proqramlar paketlərinin layihələndirilməsi metodikasını öyrənmək, tətbiqi proqramlar paketlərininyaradılması mərhələlərini və bu sistemlərdə tətbiq sahəsinin şərhi üçün istifadə olunan müxtəlif modelləri araşdırmaq, təsnifatlaşdırmaq, tətbiqi proqramlar paketi generasiya edən instrumental proqramlaşdırma sistemlərinin, o cümlədən "VİLNÜS" hesab-məntiq problem-yönümlü instrumental sistemin arxitekturunu və fəaliyyət prinsipini öyrənmək tədqiqatın məqsədinə daxildir.

Tədqiqatın obyekti. Hesab-məntiq problem-yönümlü sistemlər, o cümlədən intellektual tətbiqi proqramlar paketləri, onların tətbiq sahəsinin modelləri, tətbiqi proqramlar paketlərinin yaradılması mərhələləri, bu sistemlər layihələndirilərkən çəkilən xərclərin qiymətləndirilməsi metodları, "VİLNÜS" instrumental proqramlaşdırma kompleksinin strukturu tədqiqat obyekti kimi araşdırılmışdır.

Tədqiqatın nəzəri və praktiki əhəmiyyəti. Hesab-məntiq problem-yönümlü sistemlərin yaradılması səbəbləri araşdırılmış, bu sistemlər vasitəsi ilə generasiya olunan tətbiqi proqramlar paketlərinin lahiyələndirilməsi mərhələləri, sistemə qoyulan tələblərin analizi, spesifikasiyaların çüəyyən olunması, layihələndirmə, kodlaşdırma, testləşdirmə, istismar və müşayət etmə mərhələləri, tətbiq sahəsinin funksional, informasiya, alqoritmik, dil, proqram modelləri öyrənilmiş, hesab-məntiq problem-yönümlü sistemlər vasitəsi ilə generasiya olunan tətbiqi proqramlar paketlərinin tətbiq sahəsi modellərinin təsnifatı aparılmış və bu sistemlərin tətbiq sahəsi modelinin layihələndirilməsi üçün modul prinsipi təklif olunmuşdur. Hesablama şəbəkəsi vasitəsi ilə təsvir olunan tətbiq sahəsi modelinin adekvatlığının yoxlanması üçün mövcud olan «cüt-cüt yoxlama» metodu təkmilləşdirilmişdir.

Dissertasiya işinin birinci fəsli – «Tətbiqi proqram təminatının yaradılması mərhələləri»- kompüterin proqram təminatının tərkib hissəsi olan tətbiqi proqram təminatının yaradılması mərhələlərinə həsr olunmuşdur. Bu fəsildə hesab-məntiq

problem-yönümlü sistemlərin yaradılması səbəbləri araşdırılmış, onların inkişaf istiqamətləri öyrənilmişdir. Konkret olaraq hesab-məntiq problem-yönümlü sistemlər yaradılan zaman yaradılacaq sistemin spesifikasiyalarının müəyyən olunması, yaradılan sistemə verilən tələblərin analizi, layihələndirmə, kodlaşdırma, avtonom və kompleks testləşdirmə, istismar və müşayət etmə mərhələləri araşdırılmış, yaradılma mərhələləri üzrə xərclərin bölüşdürülməsi metodları probleminə baxılmışdır.

İkinci fəsil - « Tətbiqi proqramlar paketlərinin layihələndirilməsi vasitələri » - tətbiqi proqramlar paketinin fəaliyyətini təmin edən vasitələrin öyrənilməsinə həsr olunmuşdur.

Tətbiqi proqramlar paketlərinin layihələndirilməsi prosesi araşdırılmış, tətbiq sahəsi modellərinin, o cümlədən funksional modelin, dil modelinin, alqoritmik modelin, informasiya modelinin və proqram modelinin qurulması üsulları tədqiq olunmuşdur. Tətbiq sahəsinin informasiya modelinin həmin sahənin əsas obyektlərini və bu obyektlər arasındakı münasibətləri göstərdiyi aşkar olunmuşdur. Tətbiq sahəsinin funksional modelinin obyektlər üzərində görüləcək işlərin və bu işlərin yerinə yetmə şərtlərini müəyyənləşdirdiyi qeyd olunmuş, funksional modeli şərh etmək üçün çox müxtəlif formalizmlərin tətbiq oluna biləcəyi göstərilmişdir.

Üçüncü fəsil -« “VILNÜS” instrumental proqramlaşdirma sisteminin tərkibi və əsas funksiyalari »- hesab-məntiq problem-yönümlü instrumental proqramlaşdirma sistemi tətbiq sahəsi modelinə əsasən tətbiqi proqramlar paketləri generasiya edən instrumental proqramlaşdırma sistemlərinin bir sinfinə həsr olunmuşdur. Bu sinif hesab-məntiq problem-yönümlü instrumental proqramlaşdirma sistemləri vasitəsilə tətbiq sahəsini seçməklə, geniş funksiyalara malik tətbiqi proqramlar paketləri generasiya olunurlar. Bu fəsildə “VILNÜS” instrumental proqramlaşdirma sisteminin tərkibinə daxil olan əsas komponentlər, o cümlədən generasiya vasitələri, konstruktorlar, idarəedici proqram və onun əsas elementləri, məsələnin həlli planının interpretatoru və nonitorun iş prinsipi araşdırılmış, monitorun problem

modullarla qarşılıqlı fəaliyyətini atəmin etmək, həm də monitorun reallaşdıracağı sorğuları formalaşdırmaq üçün sorğu dili təklif olunmuşdur

Dissertasiya işi girişdən, üç fəsildən, nəticə və təkliflərdən, istifadə edilmiş ədəbiyyat siyahısından ibarətdir. Dissertasiya işində 17 şəkildən istifadə olunmuş, işin ümumi həcmi 84 səhifədə verilmişdir.

FƏSİL I. TƏTBİQİ PROQRAM TƏMİNATININ YARADILMASI MƏRHƏLƏLƏRİ.

1.1. Hesab-məntiq problem-yönümlü sistemlərin yaradilmasi səbəbləri.

Məlumdur ki, tətbiqi proqramlar paketləri generasiya edən instrumental proqramlaşdırma sistemləri kompüterin tətbiqi proqram təminatının əsas komponentləri hesab olunurlar. Qeyd edək ki, proqramlaşdırmanın ilk mərhələsi birinci növbədə proqramların yazılmasının texniki (standart) vərdişlərinin qazanılması sahəsindəki təcrübələrin toplanması ilə əlaqədar idi. Lakin hesablama texnikası vasitələrinin inkişafı və bu təcrübələrin toplanması ilə əlaqədar olaraq vəziyyət kökülü surətdə dəyişdi. Təbii olaraq belə suallar ortaya çıxırdı – biz səhvlər etməkdə hələ də davam edirikmi ?; proqramların yazılması prosesinin özü düzgündürmü; proqram təminatının hazırlanmasının yeni metodlarının yaradılması zərurəti meydana gəlməmişdirmi ?

Bu suallara cavab verməklə məşğul olan yeni elmi istiqamət – **proqram təminatının yaradılması metodları (texnologiyaları)** adlanır. Bu sahənin məşğul olduğu əsas problemlər klassik proqramlaşdırmanın tərkib hissələrinin öyrənilməsidir:

- Spesifikasiyaların yazılması;
- Lahiyələndirmə;
- Testləşdirmə;
- Proqramların fəaliyyəti.

Bu elmi sahənin böyük praktiki əhəmiyyət kəsb etdiyini texniki sistemlərlə böyük proqram sistemlərinin yaradılması sahəsindəki işlərin vəziyyətini müqayisə etməklə nümayiş etdirmək olar.

1958-ci ildə ABŞ-da Verra-tsano-Harrous körpüsünün tikintisinə başlanılır. Mühəndislərin hesablamalarına görə tikintiyə 325 mln. dollar vəsait xərclənəcəyi və tikintinin 1965-ci ildə başa çatacağı planlaşdırılmışdı. Bu dünyada ən böyük

asma körpü idi. Körpünün tikintisi 1964-cü ilin noyabr ayında başa çatdırıldı və smetada nəzərdə tutulan miqdarda vəsait xərcləndi.

Təxminən eyni vaxtda İBM firması 08 əməliyyat sistemini yaratmağa başladı. Yaradıcıların planına görə əməliyyat sistemi 5000 insan-ilə hazır olmalı idi. Bütün mümkün tədbirlərin görülməsinə baxmayaraq əməliyyat sistemi nəzərdə tutulduğundan 4 il sonra hazır oldu.

Ortaya belə bir sual çıxır. Nə üçün texniki sistemlərdə dəqiq proqnoz vermək mümkün olduğu halda, proqram sistemləri yaradılarkən belə düzgün proqnoz vermək mümkün deyil ?

Bu suala qismən də olsa belə cavab vermək olar ki, mühəndis körpünün ölçülərinin böyüməsi ilə əlaqədar ortaya çıxan çətinlikləri nəzərə ala bildiyi halda, proqram təminatı yaradılarkən sistem proqramlarının həcminin böyüməsi ilə əlaqədar çətinlikləri müəyyən etmək mümkün deyildir.

Ümumi halda **proqram təminatının yaradılması metodları**, proqramlaşdırma onun əsas tərkib hissəsini təşkil etməsinə baxmayaraq, proqramlaşdırma deyildir. Bu sahə həmçinin kompüterlərin və əməliyyat sistenlərinin öyrənilməsini nəzərdə tutmur, baxmayaraq ki, həm kompüterlərin, həm də əməliyyat sistemlərinin baxılan texnologiyada mühüm rolu vardır. Eləcə də elektron texnikanın problemləri, kompüterlərin strukturu burada tədqiqat obyekti deyildir, lakin bu sahədə olan biliklər də mühüm əhəmiyyət kəsb edir.

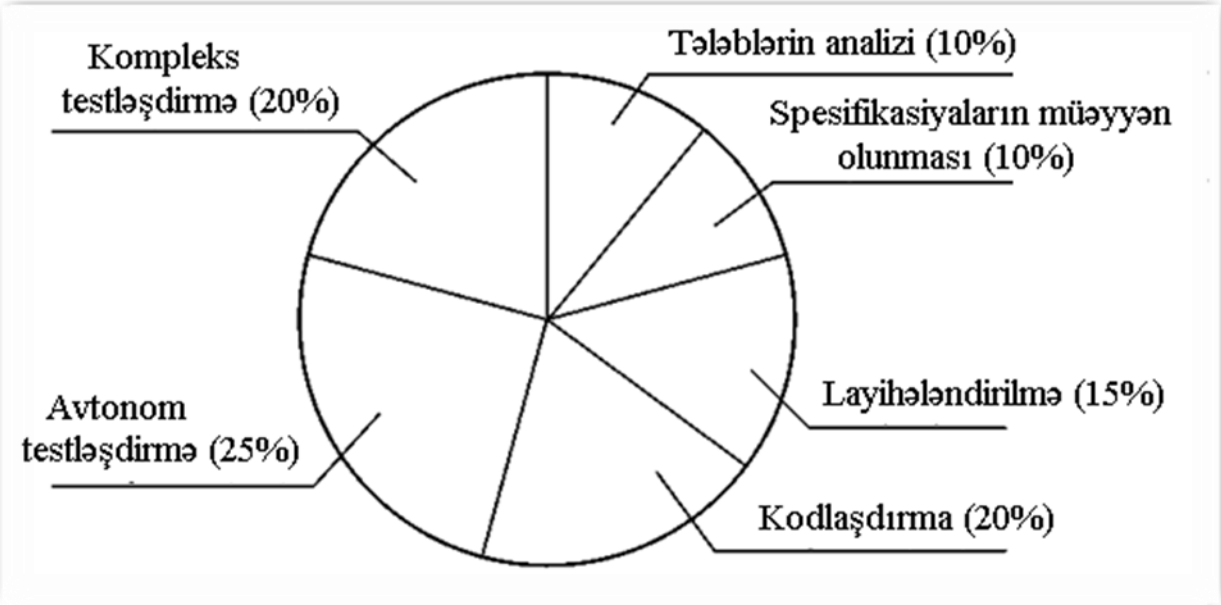
Proqtam təminatının yaradılması metodları – sintez edilmiş (ümumiləşdirilmiş) bir elm sahəsidir. Burada alqoritmlərin qurulması üçün riyazi metodlar, çəkilən xərclərin və kompromiss həllərin srçilməsi üçün mühəndis hesablamaları metodları, sistemə qoyulan tələblərin, itkilərlə əlaqədar vəziyyətlərin uşotunun aparılması, proqramçıların işinin təşkili və proqnozlaşdırma üçün idarəetmə metodları istifadə olunur.

Təbiidir ki, təkcə bir adam çox böyük bir sistemin proqram təminatını bütünlükdə dərk edərək yaratmaq iqtidarında deyildir. Böyük proqram sistemlərinin yaradılması gedişini idarə etmək üçün altı mərhələ ayrılır (nəzərdə tutulur) ki, bu mərhələlər proqram təminatının yaradılması dövrünü təşkil edirlər:

1) Sistemə qoyulan tələblərin analizi;
2) Spesifikasiyaların müəyyən olunması;
3) Lahiyələndirmə;
4) Kodlaşdırma;
5) Testləşdirmə;
   a) avtonom;
   b) kompleks;
6) istismar və müşayiət etmə.

Yaradılma mərhələləri üzrə xərclərin bölüşdürülməsi aşağıdakı diaqramda göstərilmişdir (Şəkil 1.1).

Diaqramda proqram təminatının yaradılmasının ayrı ayrı mərhələlərinin reallaşdırılmasına sərf olunacaq xərclər təxmini olaraq göstərilmişdir.

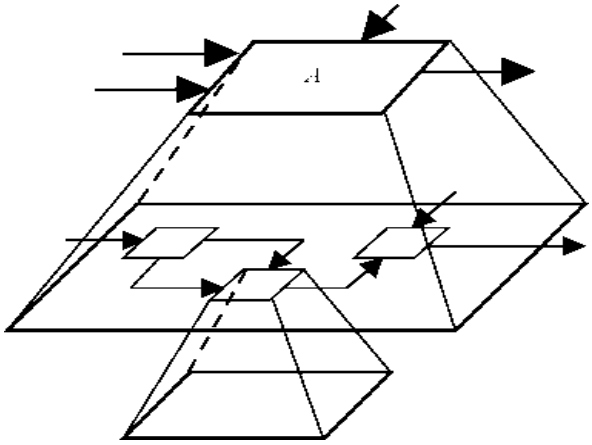


Şəkil 1.1 Yaradılma mərhələləri üzrə xərclərin bölüşdürülməsi

Birinci mərhələdə, bu mərhələ əsaslandırılmadan buraxılır, problemin mümkün həllinin alınmasına imkan verən tələblər müəyyən olunur. Bu mərhələdə yaradılan proqramm sisteminin təyinatının məqsədi və əsas xüsusiyyətləri müəyyən olunur.

İşlərin yerinə yetirilməsi prosesi və nəticələrin formalaşdırılması bu mərhələdə o biri mərhələlərə nisbətən çox kiçik həcmdə tədqiq olunur, və ümumi halda professional proqramçının fəaliyyət obyektləri deyildir.

Əgər yaradılmanın predmeti (obyekti) proqram sistemi deyilsə, və daha mürəkkəb obyektdirsə (məsələn, texnologi prosesin idarə olunmasısistemi), və proqramlar sadəcə olaraq onun tərkib hissələrindən biridirsə, onda tələblər bütün yaradılan obyekt üçün formalaşdırılır. Əsas məqsəd proqram təminatının yaradılması olduqda adətən ilkin şərhlərin tərtib olunması metodları istifadə olunur. İlkin şərhlərin ən effektiv metodlarından biri **struktur analizi metodu** hesab olunur. Bu metodun əsas məğzi (məzmunu) ilkin obyektin, yəni baxılan obyektin dekompozisiyası, yəni tərkib hissələrinə bölünməsidir.



Şəkil 1.2 Sistemin dekompozisiyası sxemi

Beləliklə bir biri ilə əlaqədə olan (mütləq kəsişməyən) alt sistemlərin iyerarxiyası yaradılır.Ümumi halda, sistemə qoyulan tələblərin analizi insanla (istifadəçi ilə) alət (kompüter) arasında interfeysə yönəldilmədidir.Bu zaman proqram sistamləri üçün yalnız baza tələbləri ayıraraq göstərmək olar:

- proqramın işləmə müddəti;
- işlənmənin qiyməti;
- səhvlərin baş vermə ehtimalı;

- operatorun nəzərdə tutulmayan (ağılagəlməz) hərəkətlərinə reaksiya (axmaqdan müdafiə və s.).

Tələblərin dekompozisiyası zamanı ciddi tələblərlə çox da ciddi olmayan tələblər arasındakı fərqə fikir vermək lazımdır. Xüsusilə fəza-zaman məhdudiyyətlərinə və gələcəkdə dəyişikliyə uğraya biləcək sistem ehtiyatlarına fikir ayırmaq lazımdır.

Ehtiyyat (resurs) tələbləri və sistemin reallaşdırılması xərcləri ən əsas tələblərə aid edilir.

Faktiki olaraq, tələblərin analizi, sistemin texniki tapşırığının aşkar şəkildə tərtib olunması ilə nəticələnir ki, bu da klassik SAPR terminologiyası ilə desək avan-proyekt adlanır.

1.2. Spesifikasiyaların müəyyən olunması və layihələndirmə mərhələləri

Spesifikasiyaların müəyyən olunması mərhələsində kompüterlə realizə olunacaq funksiyaların şərhi yerinə yetirilir, həmçinin ilkin (giriş) və son nəticə (çıxış) informasiyalarının strukturları və onların yerləşdirilməsi metodları və vasitələri verilir.

Spesifikasiyaların müəyyən olunmasının ən əsas məsələsi (mərkəzi yəni başlıca sualı) verilənlər bazasının təşkili problemidir. Bu zaman kompleks məsələlər (suallar) həllini tapmalıdır, o cümlədən faylların strukturu, onlara müraciətin təşkili, modifikasiyası və silinməsi (yox edilməsi) sualları həllini tapmalıdır.

Yeni sistem artıq mövcud olan sistemin bazasında yaradılarkən spesifikasiyaların əsas hissəsini mövcud olan verilənlər bazasının yeni formata gətirilməsi sxemi (alqoritmi) təşkil edir. Belə çevrilmənin həyata keçirilməsi üçün yeni bir proqramın yazılması tələb oluna bilər, belə ki, bu proqram bir dəfə yeganə istifadəsindən sonra lazımsız bir proqrama çevrilər.

Bütün bu sualar, sistemin yaradılması prosesində əsas olan sənəd kimi təqdim olunan funksional spesifikasiyalarda əks olunmalıdır, belə ki, sistemin konkret

şərhi bu spesifikasiyalarda saxlanılır. Spesifikasiyalar nə qədər ətraflı tərtib olunarlarsa, səhvlərin baş verməsi ehtimalı bir o qədər az olar.

Spesifikasiyalarda sistemin elementlərinin və bütövlükdə sistemin özünün testləşdirilməsi üçün **verilənlər** təqdim olunmalıdır. Bu tələb obyektiv və mütləqdir, belə ki, bu mərhələdə sistemin konkret realizasiyası testləşdirmənin parametrlərinə təsir göstərməyəcəkdir.

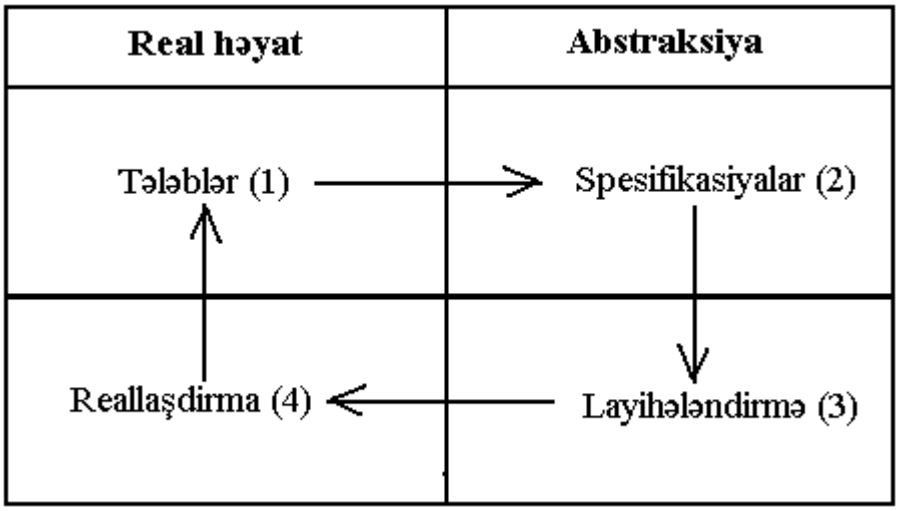
Funksional spesifikasiyalar qərarın bütövlükəd qəbul olunmasını şərh etdiyindən, baxılan (yəni qəbul olunan) sənəd vaxt itkisinin, mütəxəsislərin sayının və işlərin yerinə yeturilməsi üçün lazəm olan digər ehtiyatların ilkin qiymətləndirilməsi üçün istifadə oluna bilər.

Ümumi halda, spesifikasiyalar, sitemin yerinə yetirəcəyi bütün funksiyaları, onların necə həll olunacağından asılı olmayaraq, müəyyən edir. Bu mərhələdə uyğun alqoritmlərin tərtib olunması vaxtından əvvəl olduğundan məsləhət deyildir və arzuolunmaz çətinliklər əmələ gətirə bilər.

Lahiyələndirmə mərhələsində spesifikasiyalarla verilən alqoritmlər qurulur və hesablama sisteminin ümumi strukturu formalaşdırılır. Bu zaman sistem (lazım gəldikdə) tərkib hissələrinə elə bölünür ki, hər bir hissənin reallaşdırılması məsuliyyətini sistemi yaradanlardan birinin (yaxud icraçılar qrupunun) üzərinə qoymaq mümkün olsun. Bu zaman sistmin bu şəkildə müəyyən olunan hər bir modulu üçün ona qoyulan tələblər formalaşdırılmalıdır (müəyyən olunmalıdır):

- reallaşdırılacaq funksiyalar;
- ölçülər (modulun ölçüləri);
- yerinə yetmə müddəti və s.

Bütövlükdə “tələblər-spesifikasiyalar-lahiyələndirmə” münasibətini aşağıdakı sxem vasitəsi ilə nümayiş etdirmək olar (şəkil 1.3).



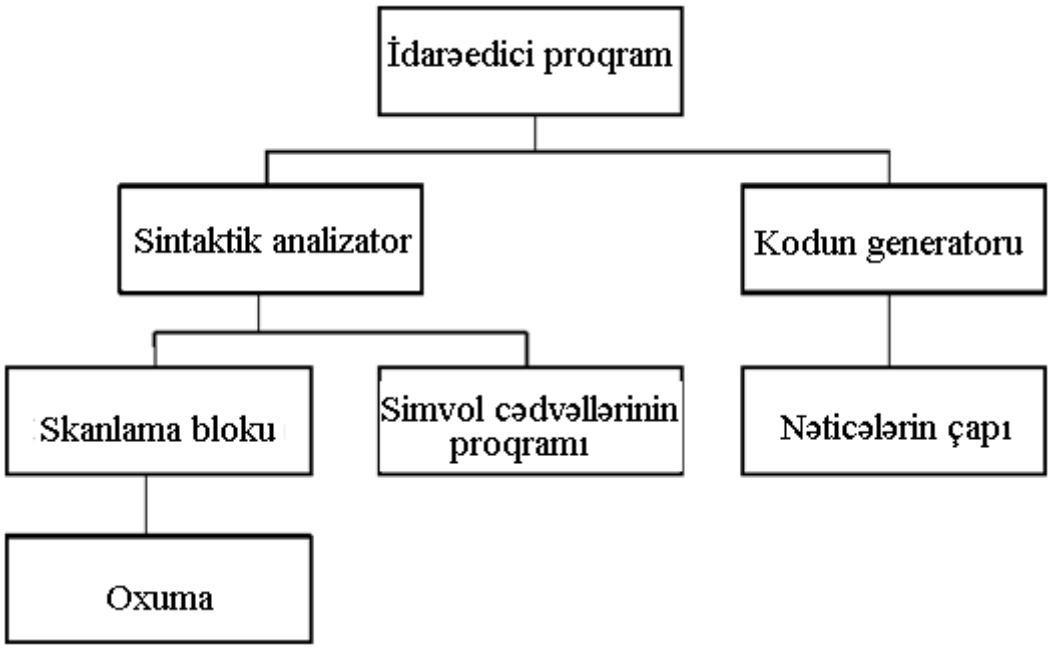
Şəkil 1.3 Proqram sistemlərinin lahiyələndirilməsi sxemi.

Hər şeydən əvvəl Sifarişçi proqram sisteminə verilən (qoyulan) tələblərdə öz əksini tapan real tələbləri (obyektiv faktları) analiz edir və müəyyənləşdirir (qısa və dürüst ifadə edir). Lakin kompüter məsələni bilavasitə həll etmək iqtidarında deyildir, odur ki, real verilənləri hansı şəkildəsə kodlaşdırmaq və kompüterə daxil etmək lazımdır. Həll olunacaq məsələnin belə (bu tərzdəki) modeli real həqiqətin (dünyanın) abstrakt ifadəsisir və spesifikasiyalarda öz əksini tapır.

Əgər proqramın spesifikasiyaları müəyyən olunubsa, onda informasiyaların işlənməsi prosesinin şərh olunması zərurəti ortaya çıxır, bu isə artıq lahiyələndirmə mərhələsinə aiddir. Proqram real məsələ üçün istifadə olunduğundan, lahiyənin reallaşdırılması mərhələsində (kodlaşdırma, testləşdirmə) formal lahiyənin yerinə yetiriləcək proqrama çevrilməsi həyata keçirilir.

Nəhayət, istifadəçi proqramın planlaşdırılan funksiyalarını reallaşdırılan funksiyalarla müqayisə edə bilər. Bu funksiyalar nadir hallarda təmamilə üst üstə düşürlər. Beləliklə, **müşayət** lahiyələndirmə dövrünü sona yetirir və sistem tələblərini, spesifikasiyaları, proqramların lahiyələrini dəyişməyə imkan verir.

Sistemin lahiyələndirilməsi prosesində müəyyən alt modullar üzərində spesifikasiyaların yerinə yetirməsi ağacabənzər çəkildə aşağıdakı kimi göstərilə bilər (şək. 1.4).



Şəkil 1.4 Kompilyatorun struktur sxemi.

Belə sxem bazis sxem adlanır və o spesifikasiyalara adekvat deyildir. Lahiyələndirmə mərhələsinin əvvəlində bəzən funksional məsələlərin həlli müəyyən olunmadığından , alt məsələlərə bölünmə prosesi kifayət qədər mürəkkəb ola bilər. Proqram sistemləri lahiyələndirilərkən Sifarişçinin dəqiq nə istədiyini bilmədiyi adi haldır. Bu xüsusilə pis yaxud da zəif formallaşdırılan proseslərə aiddir (tibb, hərbi təyinatlı sistemlər və s.). Lahiyə yaradıldıqca Sifarişçi spesifikasiyaları dəyişir. Əgər bu dəyişmələr çox tez-tez baş verirsə, onda sistemin yaradılması kifayət qədər mürəkkəbləşir.

1.3. Kodlaşdırma və testləşdirmə mərhələləri.

Bu mərhələ kifayət qədər sadə hesab olunur, onun reallaşdırılması yüksək səviyyəli alqoritmik dillərin istifadəsi zamanı xeyli yüngülləşir. Kodlaşdırma – proqram təminatının yaradılması mərhələsində sistemin yaradıcılarına ən az narahatşılıq gətirən mərhələdir. Əldə olunan statistik göztəricilərə əsasən demək olar ki, səhvlərin 64% -i lahiyələndirmə mərhələsinə, yalnız 36% -i kodlaşdırma mərhələsində baş verir. Lakin bu səsvlər çox baha başa gələ bilər (DO 4 I=1,5 и

DO 4 I=1.5). Ümumi halda, demək olar ki, proqramların yaradılması zamanı ən yaxşı öyrənilən və dəqiq formallaşdırılan kodlaşdırma mərhələsidir.

Testləşdirmə mərhələsi adətən maliyyə xərcləri cəhəttən sistemə sərf olunan xərclərin yarısını təşkil edir. Pis planlaşdırımış testləşdirmə sistemin yaradılması müddətini əhəmiyyətli dərəcədə artırır və sistemin yaradılması qrafikinin pozulmasına səbəb olur.

Testləşdirmə prosesində işçi vəziyyətdə olan sistem üçün xarakterik olan verilənlər istifadə olunur, başqa sözlə desək, testləşdirmə üçün verilənlər təsadüfi olaraq seçilir. Sınağın aparılması planı əvvəlcədən tərtib olunmalıdır, bu adətən lahiyələndirmə mərhələsində yerinə yetirilir.

Testləşdirmə üç mərhələni (pillə) nəzərdə tutur:

- avtonom;
- kompleks;
- sistem.

Avtonom testləşdirmə zamanı modul proqramçı tərəfindən hazırlanan verilənlərin köməyi ilə yoxlanılır. Bu zaman modulun proqram mühiti testləşdirmənin idarə olunması proqramının köməyi ilə imitasiya olunur. Testləşdirmənin idarə olunması proqramı real alt proqramlar əvəzinə baxılan moduldan müraciət olunan saxta (yalançı) proqramlar saxlayır (zaqluşka-qapaq). Uyğun prosedur proqram testləşdirməsi, testləşdirmə proqramı isə - UUT (testləşdirən proqram) adlanır. Avtonom testləşdirmədən keçən modul, kompleks testləşdirmədən keçirilir.

Kompleks testləşdirmə prosesində proqram komponentləri qrupunun birlikdə yoxlanılması keçirilir. Nəticədə tam şəkildə yoxlanılmış sistemə malik oluruq. Testləşdirmənin bu mərhələsində avtonom testləşdirmə mərhələsində buraxılan səhvlər aşkar olunur.Bu səhvlərin aradan qaldırılması ümumi xərcin ¼ - ni təşkil edir.

Sistem (yaxud qiymətləndirici) testləşdirmə - sistemin yoxlanılmasının son mərhələsi hesab olunur, başqa sözlə, sistemin bütövlükdə yoslanılması müstəqil testlərin köməyi ilə aparılır. Testlərin müstəqilliyi əsas tələb hesab olunur. Adətən

Sifarişçı işin qəbulu pilləsində (mərhələsində) xüsusi sistem testləşdirilməsinin aparılmasını təkid edir. Bir neçə sistemin xarakteristikaları müqayisə olunduğu hal üçün belə prosedur **müqayisəli testləşdirmə** adı ilə məlumdur.

Testləşdirmə prosesində, proqramların yerinə yetməsinin düzgünlüyünü müəyyən etmək üçün bir sıra kriterilər (meyarlar) daxil edilir:

1) hər bir operator verilmiş testlər toplusu üçün ən azı bir dəfə yerinə yetməlidir, və proqram düz nəticə verməlidir;
2) proqramın hər bir budağı yoxlanılmalıdır, və proqram bu zaman düz nəticə verməlidir;
3) proqramda hər bir yol heç olmasa bir dəfə test verilənləri toplusundan istifadə etməklə sınaqdan keçirilməlidir, və proqram bu zaman düz nəticə verməlidir;
4) proqramın hər bir spesifikasiyası üçün, bu spesifikasiyasının düzgün realizə olunduğunu müəyyən edən test verilənləri toplusunun olmasl zəruridir.

1 və 2 kriteriyaları bir birlərinə oxşasalar da, əslində onlar bir birlərindən kəskin fərqlənirlər. Məsələn Fortran dilində **İF** operatoru:

İF (ifadə) N1, N2, N3.

1 kriteriyası nəzərdə tutur ki, **İF** operatoru yerinə yetməlidir, halbuki 2 kriteriyası **N1, N2, N3** şərtlərinin yerinə yetirilməsi üçün müxtəlif verilənlər toplularının olmasını nəzərdə tutur (başqa sözlə, idarənin bu nişanlara verilməsi).

Ümumi halda "yaxşı yoxlanılmış" proqramı müəyyən edən vahid proqram kriterisi yoxdur.

Testləşdirmə ilə "verifikasiya" (yoxlama deməkdir) və "sınaq" anlayışları sıx əlaqədardırlar.

Sistemin sınağı testləşdirmə vasitəsi ilə həyata keçirilir. Belə yoxlamada məqsəd – sistemin onun üçün yaradılmış (müıyyən olunmuş) spesifikasiyalara uyğun olaraq fəaliyyətini göstərməkdir.

Verifikasiya proqramın öz spesifikasiyalarını qane etdiyinin isbatını yerinə yetirməkdir.

Müasir proqramların yaradılması prosesi göstərilən hər iki konsepsiyanın reallaşdırılmasına imkan vermir. Nəzərə alınmayan test verilənləri halında sınaqdan keçən sistem düzgün olmayan nəticələr verə bilər. Verifikasiya aparıldıqdan sonra sistem yalnız başlanğıc spesifikasiyalara nəzərən düzgün işləyir. Proqramların düzgünlüyününformal isbatı çox mürəkkəbdir və az (zəif) öyrənilmişdir.

Sınaq prosedurlarının və verifikasiyanın köməyi ilə düzgün proqramların yaradılmasının ümumi prosesi ***attestasiya*** adlanır.

Sistemin normal işindən sapmaların üç növünü bir birindən fərqləndirirlər.

Sistemin zədələnməsi (сбой) - bu hal qəbul olunmuş spesifikasiyaların sistem tərəfindən pozulmasıdır.

Verilənlər, sistemin düzgün alqoritmləri ilə emal olunan zaman zədələnmə baş verir, bu ***atma*** (***tullanış***) adlanır. Tullanışın aradan qaldırılması (düzəldilməsi) proqramda nəzərə alına bilər, belə ki, hər bir tullanış zədələnməyə səbəb olmur.

Səhv – alqoritmik nöqsan hesab olunur, bu nöqsanı atma (tullanış) yaradır (proqram səhvi).

"düzgün" və "etibarlı" proqram anlayışlarını bir birindən fərqləndirirlər. **Düzgün** proqram – öz spesifikasiyalarını qane edən proqramdır. **Etibarlı** proqrama gəldikdə isə o düzgün proqram olmaya da bilər, lakin bu proqram hətta ilkin verilənlər yaxud da onların istifadə olunması şərti qəbul olunmuş fərziyyələri qane etmədikdə belə qəbul oluna biləcək nəticələr verir. Təbiidir ki, Sifarişçi "canlı" sistemə malik olmaq istəyir., başqa sözlə desək, elə sistemə malik olmaq istəyir ki, hətta əlverişli olmayan şəraitdə belə geniş spektrli verilənləri qəbul etmək iqtidarında olsun.

Əgər sistemdə səhvlər yoxdursa və onun daxili verilənləri tullanış saxlamırlarsa, onda belə sistem ***düzgün sistem*** adlanır.

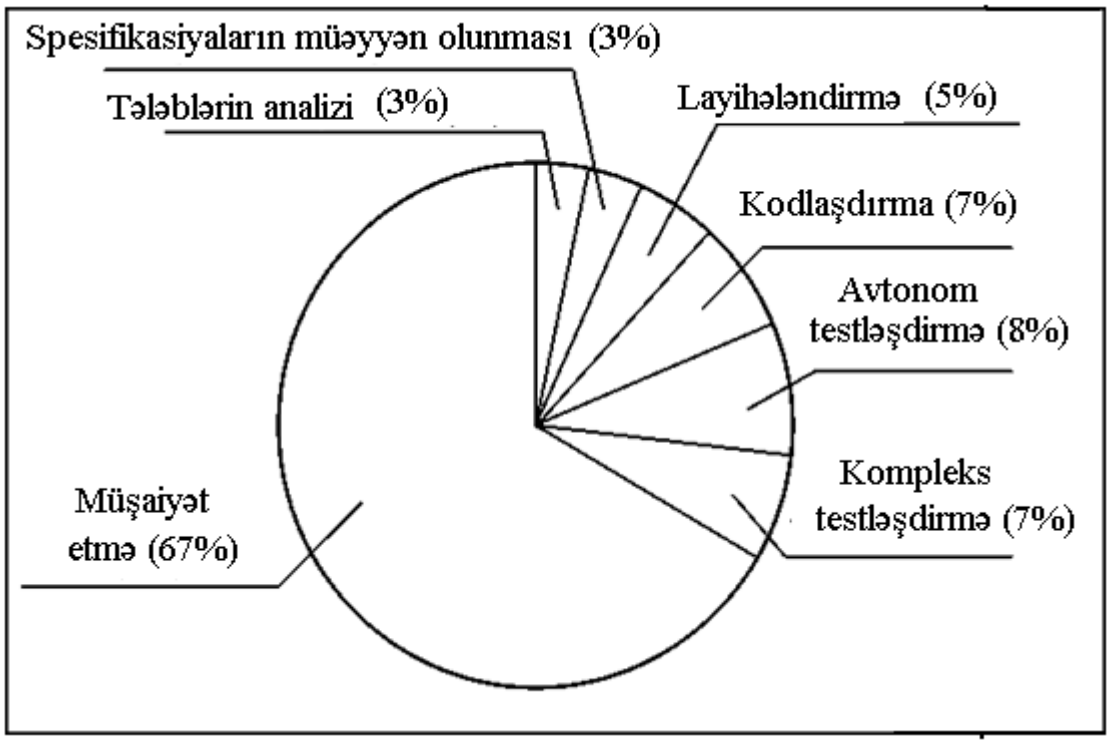
Əgər sistem zədələnmələrə baxmayaraq qaneedici dərəcədə fəaliyyətini davam etdirirsə, onda belə sistem ***etibarlı sistem*** adlanır. Bu xüsusilə zədələnmələrin işlənməsi sisteminə malik əməliyyat sistemləri misalında daha aydın görünür. Tullanmalara rast gəlindikdə belə sistem cari informasiyaları

saxlamaqla işini dayandırır və tullanmalar aradan qaldırıldıqdan sonra işini davam etdirir.

1.4. İstismar və müşayət olunma mərhələləri.

İstismara və müşayətə sərf olunan xərcləri nəzərə almaqla, proqram sisteminin yaradılmasının zaman itkisini aşağıdakı kimi təsvir etmək olar (şəkil 1.5):

1) tələblərin analizi;
2) spesifikasiyaların müəyyən olunması;
3) lahiyələndirmə;
4) kodlaşdırma;
5) avtonom testləşdirmə;
6) kompleks testləşdirmə;
7) müşatət.



Şəkil 1.5 Proqram təminatının həyat dövrünün reallaşdırılmasına sərf olunan zaman xərcləri.

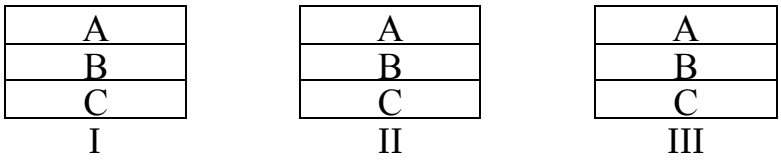
Heç bir hesablama sistemi istismar olunarkən dəyişilməmiş qalmır, yəni mütləq sistemdə dəyişiklik baş verir. Bu bir sıra səbəblərlə izah olunur ki, onların içərisində aşağıdakıları nəzərdən keçirmək olar:

1. Sifarişçi adətən öz tələblərini heç vaxt dəqiq formalaşdıra bilmir, çox nadir hallarda yaradılan sistem onu qane edir və bu səbəbdən hazır sistemdə dəyişiklik etməkdə israrlı olur.
2. Testləşdirmə zamanı buraxılan səhvlər aşkar oluna bilər.
3. Müxtəlif tətbiqlərlə əlaqədar olaraq xüsusi şərtlərdə fəaliyyət üçün sistemin modifiksiyası tələb oluna bilər.
4. Sistemin çoxsaylı komponentlərinin müşayət olunması.

Sistemin yaradılması prosesinə ən çox təsir göstərən xərcləri nəzərdən keçirək. Birinci növbədə onu qeyd etmək lazımdır ki, lahiyənin tez başa çatmasını stimullaşdıran yaradılma metodları sistemin müşayət olunmasına çəkilən xərcləri əhəmiyyətli dərəcədə artıra bilər. Ona görə də kodlaşdərma mərhələsinə vaxtındn əvvəl keçidə çalışmaq lazım deyildir. Baxmayaraq ki, kodların yzılması lahiyənin

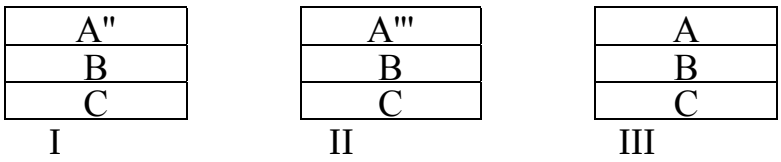
rəhbərində müsbət illuziya yarada bilər, lakin bu təkrarlanan testləşdirmələrə və sonrakı müşayət mərhələsində bir sıra problemlərin yaranmasına səbəb ola bilər.

Tutaq ki, hər hansı bir sistem **A, B, C** komponentlərinə malikdir və bu sistem **I, II, III** istehlakçılarında istifadəyə verilmişdir (şək. 1.6).



Şəkil 1.6 İstehlakçıların ilkin sistemləri

Sistemin istifadəai zamanı **I** –ci istifadəçi səhv tapır və bu barədə sistemin yaradıcısına məlumat verir. Sistemin yaradıcısı bu səhvi düzəldir və yeni $\mathbf{A^1}$ modulunu bütün istifadəçilərə göndərir. Etibarlı proqram təminatının tətbiqi təcrübəsi göstərir ki, əksər istifadəçilər sistemdə düzəldilən səhvlərə çox ehtiyatla yanaşırlar. Ona görə də **II** – ci və **III** – cü istifadəçilər **I** – ci istifadəçinin həll etdiyi məsələlərlə rastlaşmayana qədər, " əgər sistem işləyirsə, onda qarışma" prinsipini rəhbər tutaraq sistemin ilkin variantından istifadə etməkdə davam edirlər. Bir müddətdən sonra **I** – ci və **II** – cü istifadəçilər **A** modulunda başqa bir səhv aşkar edirlər. Sistemin yaradıcısı aşkar olunan səhvlərin eyni olduğunu müəyyən etməlidir, çünki bu istifadəçilər **A** modulunun müxtəlif versiyalarından istifadə etmişlər.Aşkar olunan səhvlərin **A** və $\mathbf{A^1}$ modullarında aradan qaldırılması nəticəsində istismara $\mathbf{A^{11}}$ və $\mathbf{A^{111}}$ modulları daxil edilir. Göründüyü kimi artıq sistemin üç versiyası fəaliyyət göstərir (şək. 1.7).



Şək. 1.7 Sistem iki səhv aradan qaldırıldıqdan sonra

Bir çox hallarda sistemin yaradıcısı sistemin əvvəlki versiyalarında aşkar olunan səhvlərin aradan qaldırılmasına böyük əmək sərf edir. Sistemin belə selşəkilli böyüməsinin qarşısını almaqdan ötrü, sistemdə müəyyən zaman müddətində düzəlişlər aparılır ki, bu da ***yenilənmə dövrü*** adlanır.

Sistemin müşayət mərhələsində ortaya çıxan çoxsaylı problemlər "sistemin verilənlər bazası" konsepsiyasını cəlb etməklə (nəzərə almaqla) həll olunmalıdır. Belə konsepsiyanın formalaşdırılması spesifikasiyaların müəyyən olunması mərhələsində başlayır. Bu verilənlər bazası müxtəlif sifarişçilərin tələblərininəzərə almalı və sistemdə dəyişiklik aparmaq üçün istifadə olunan testkəşdirmə, səhvlərin aradan qaldırılması, əlamət vasitələrini daxil etməlidir.

1.5. Yaradılmanın idarə olunması və layihənin yerinə yetirilməsi.

Proqram təminatının yaradılması metodlarını öyrənərkən məlum olur ki, sistemin yaradılmasının hər bir mərhələsi özündən nisbətən sonrakı mərhələyə təsir göstərir ("sintez-analiz" texnologiyası). Məsələn, spesifikasiyaların yazılması prosesi ilkin tələblərin analizi mərhələsinə təsir göstərir. Lahiyələndirmə mərhələsində spesifikasiyaların yazılması prosesində buraxılan səhvlər aşkarlanır. Kodlaşdırma, testləşdirmə və istismar mərhələlərində yalnız lahiyələndirmə mərhələsində həll oluna biləcək problemlər aşkara çıxır. Bütün yuxarıda deyilənləri nəzərə alaraq "proqram təminatının yaradılması metodları" elmi fənninin əsas məqsədlərini aşağıdakı kimi ifadə etmək olar:

1) Mürəkkəb sistemlərin idarə olunması metodlarının yaradılması;
2) Proqram təminatının etibarlılığının və düzgünlüyünün yüksəldilməsi;
3) Proqram təminatının yaradılmasına çəkilən xərclərin daha dəqiq proqnozlaşdırılması metodlarının inkişafı.

Bütün bu problemlər iki hissəyə, yaradılmanın idarə olunması metodları və yaradılmanın aparılması metodları hissələrinə bölünür.

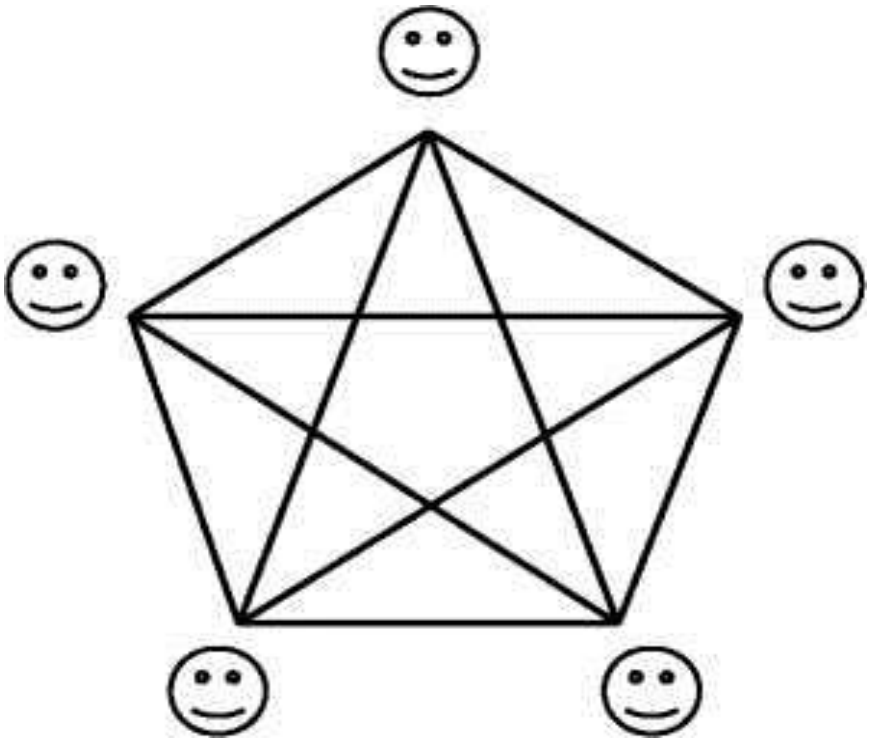
Yaradılmanın idarə olunması metodları icraçıların işinin təşkilinin effektiv olmasını nəzərdə tutur.

Yaradılmanın aparılması metodları proqramçıların əmək məhsuldarlığını yüksəltməklə onların işinin texniki cəhətdən əhatə olunmasını nəzərdə tutur.

Layihənin rəhbəri iki əsas ehtiyata (resursa) nəzarət edir – icraçılar və hesablama vasitələri. Beləlikdə, onun əsas məqsədi bu ehtiyatları optimallaşdırmaqdır. Yaradılmanın uğuru çox yüksək dərəcədə rəhbərin yaradılmanın gedişinə necə nəzarət etməsindən asılıdır. Layihənin bir il yubanması birgünlük yubanmaların toplanması nəticəsində əmələ gəlir.

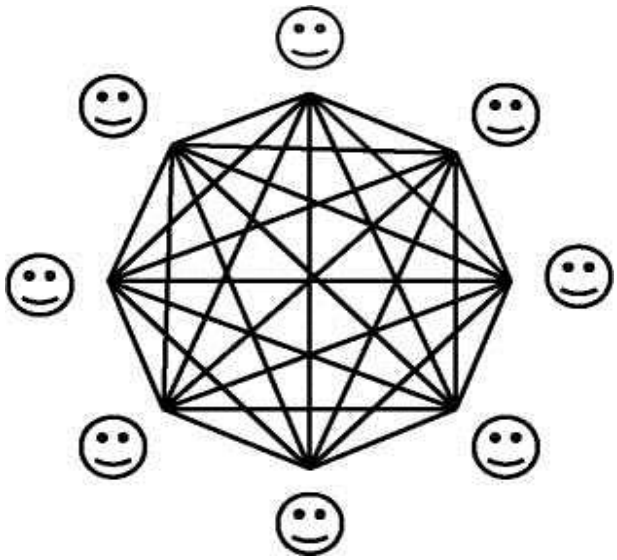
Adətən ən böyük çətinlik müxtəlif proqramçılar tərəfindən yazılan proqram modulları arasında interfeysin yaradılması zamanı əmələ gəlir. N sayda icraçı arasındakı interfeyslərin sayı N(N-1)/2 olduğundan və icraçıların sayının kvadratına uyğun artdığından, problem, layihənin yaradılması dörd və yaxud daha çox adamdan ibarət qrup tərəfindən yerinə yetirilən zaman, kifayət qədər mürəkkəb olur.

Misal. Proqramçı bir il ərzində 5000 sətirdən ibarət proqram yaza bilər, layihələndirilən sistem 50 000 sətirdən ibarət olmalıdır, və onun yaradılması iki il ərzində başa çatdırılmalıdır. Aydındır ki, belə sistemi yaratmaq üçün beş proqramçı kifayətdir. Lakin bu beş proqramçı bir birləri ilə qarşılıqlı əlaqədə olmalıdırlar, bu isə vaxt tələb edir və müəyyən dərəcədə əmək nəhsuldarlığını aşağı salır, çünki, proqramçıların bir birlərini bəzən başa düşməməsi əlavə testləşdirmə xərclərinə gətirib çıxarır.



Şəkil 1.8 Beş nəfərdən ibarət qrup arasındakı qarşılıqlı əlaqələrin sayı ona bərabərdir.

Tutaq ki, qarşılıqlı əlaqə yollarından hər biri proqramçıya 250 sətir/ildə başa gəlir. Belə olan halda hər bir proqramçı yalnız 4000 sətir/il yaza bilər, iki il ərzində isə yalnız 40 000 sətir yaza bilər. Bu isə o deməkdir ki, 50 000 sətirdən ibarət proqram yazmaq üçün hər biri 3250 sətir/il yazan 8 proqramçı lazımdır. Onların işini idarə etmək üçün rəhbər lazımdır.



Şəkil 1.9 Səkkiz icraçı və 28 əlaqə.

Lakin il ərzində yazılan sətirlərin sayı proqramçının əmək effektivliyinin dəqiq qiyməti deyildir. Proqramçıların məhsuldarlığına təsir göstərən bütün faktorları nəzərə almaq çox çətindir. Beləliklə, Proqramçılar arasındakı əlaqələri yaxşılaşdırmaq üçün müxtəlif üsul və metodlardan istifadə etmək və onların əmək məhsuldarlığını artırmaq lazımdır.

Proqram təminatını adətən üç kateqoriyaya bölürlər:

- Idarəedici proqramlar (məsələn, əməliyyat sistemləri – ƏS);
- Sistem proqramları (məsələn, kompilyatorlar);
- Tətbiqi proqramlar (məsələn, verilənlərin işlənməsi, hesablayıcı proqram).

Statistika föstərir ki, proqramçı bir il ərzində 600 sətirdən ibarət idarəedici proqram, 2000 sətirdən ibarət sistem proqramı və 6000 sətirdən ibarət tətbiqi proqram yazıb sazlamaq (отладка) iqtidarındadır.

Təbiidir ki, bu ədədlərə çox ehtiyatla yanaşmaq lazımdır, belə ki, proqram sətri (sətrin kodu) dedikdə nə başa düşülür ? Buraya verilənlərin elan olunması və şərhlər də daxildir ya yox ? Yaxud bu translyatorun generasiya etdiyi maşın əmrləridir ? Kitabxanalarda saxlanılan proqramlar necə nəzərə alınırlar? Bu sualların cavabındaan asılı olaraq proqram sətirlərinin sayı 2-4 dəfə dəyişə bilər.

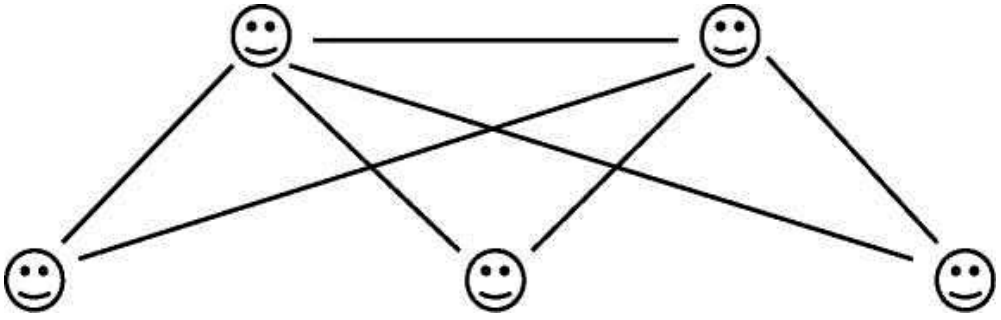
Ayrıca proqramçının iş effektivliyi yüksək dərəcədə məsələnin tipindən asılıdır. Proqramçının işinin təşkili də həmçinin işin nəticəsinə təsir göstərir. Misal üçün, vaxtən az olmasə üzündən sənədləşməyə çox az vaxt ayrılır. Lakin ümumi

xərcin 70% -i müşaiyətə sərf olunduğundan sənədlərin yaradılması vaxtına qənaət zərərli (çox pis) ola bilər.

Qoyulmuş problemin həlli metodlarından biri, EHM-lə proqramçı arasında interfeys rolunu oynayacaq ***kitabxanaçı*** vəzifəsinin təsis olunmasıdır. Proqramlar kodlaşdırılır və onların ***operativ sistem*** kitabxanasına daxil edilməsi üçün kitabxanaçıya ötürülür (verilir). Modulların sazlanması proqramçı tərəfindən həyata keçirilir, lakin proqramların kitabxanada rəsmi surətlərinin dəyişdirilməsi yalnız kitabxanaçı tərəfindən yerinə yetirilir. Kitabxanaların istifadə olunması verilənlərin idarə olunmasının operatin sistemi olduğu halda daha da genişlənir. Kitanxanaçı vəzifəsinin daxil edilməsinin daha bir müsbət cəhəti vardır: sistem kitabxanasındakı proqram modullarında bütün dəyişikliklər yalnız bir adam tərəfindən aparıla bilər, ona görə də bu prosesi idarə etmək çox asandır. Bu cür yanaşma düzgün olmayan dəyişiklikləri aradan qaldırmağa imkan verir və proqramçını hər bir dəyişiklik haqqında düşünməyə imkan verir. Bu zaman rəhbər layihələndirmənin gedişinə sistematik olaraq nəzarət edə bilir və işlərin vəziyyətini yoxlamaq asanlaşır.

Böyük miqyaslı layihələrdə sənədlərin yazılması üçün ***texniki icraçı*** cəlb edilir ki, bu da proqramçıları daha vacib ixtisas işlər üçün azad etməyə imkan verir.

Çox böyük təşkilatlarda (İBM firması) baş proqramçının briqadası yaradılır. Briqada yaradılarkən çalışırlar ki, proqramçılar müxtəlif ixtisaslaşma səviyyəsinə malik olsunlar. Baş proqramçının briqadasının təşkili kommunikasiyaların (əlaqələrin) sayının azaldılması yollarından biri hesab olunur (şəkil 1.10).



Şəkil 1.10 Baş proqrmçının briqadasının istifadə olunması zamanı qarşılıqlı əlaqələrin sayının azaldılması

1.6. Xərclərin qiymətləndirilməsi metodikası.

1.6.1 *Xərclərin mühəndis-texniki qiymətləndirilməsi metodikası.*

Proqram təminatının yaradılması prosesinin əsas aspektlərindən biri də zəruri ehtiyatların və tələb olunan xərclərin qiymətləndirilməsidir.

Xərclərin qiymətləndirilməsinin baza metodikası aşağıdakı addımlardan ibarətdir:

Addım 1. Hazırlanmış texniki tapşırığa əsaslanaraq sistemə qoyulan (verilən) ümumi tələblər formalaşdırılır. Sistemin potensial yaradıcılarına sistemdən nə gözlənildiyi haqqında məlumat verilir. Bu sənəd "tələblərın siyahısı" adlandırılır. Tələb olunan ehtiyatların daha dəqiq müəyyən olunması üçün tələblərin analizi aparılır.

Addım 2. Analoji informasiyalar toplanılır, məsələn, oxşar sistemlər haqqında verilənlər.

Addım 3. Əsas relevant verilənlər seçilir.

Addım 4. İlkin qiymətləndirmə aparılır.

Addım 5. Sistemin yekun qiymətləndirilməsi aparılır.

Bu işin əsas mərhələsi *addım 4* hesab olunur. Bu mərhələ yerinə yetirilərkən aşağıdakı əməllər icra olunur.

Addım 4a. Layıhələndirilən sistem artıq mövcud olan uyğun sistemlərlə müqayisə olunur.

Addım 4b. Sistemin hissələrə bölünməsi və bu hissələrin artıq mövcud olan digər sistemlərin uyğun hissələri ilə müqayısə olunması.

Addım 4c. Hər bir ay üçün işlərin planlaşdırılması və xərclərin qiymətləndirilməsi.

Addım 4q. İşin gedişi zamanı istifadə oluna biləcək razılaşmaların yaradılması.

Qeyd edək ki, kifayət qədər təcrübənin olmadığı zaman *addım 4a*-nın reallaşdırılması çox uzun müddətli vaxt hesabına başa gələ bilər. *Addım 4b* "sistemin hissəsi" anlayışının dəqiq müəyyən olunmasını tələb edir. *Addım 4q* də

çox da ciddi reqlamentlə fərqlənmir, belə ki, adekvat standart razılaşmalar yoxdur. Ona görə də real praktikada təklif olunan metodun müxtəlif modifikasiyaları istifadə olunur. Bu metodlar ya daha dəqiq formallaşdırılmış anlayışlara, yaxud da subyektiv qiymətləndirməyə əsaslanırlar.

1.6.2 *Ekspert qiymətləndirmələri metodu.*

Qiymətləndirmə yüksək ixtisaslı layihələndiricinin (ekspertin) şəxsi təcrübəsi əsasında aparılır. Bir çox əlavələr (proqramlar) üçün layihələndirici sistemin mürəkkəblik proqnozunu və ehtiyat xərclərinin qiymətini, hətta hələ alqoritmlər aşkar şəkildə müəyyən olunmadıqda belə verə bilər. Belə uyğun qiymətləndirmə oxşar sistemlər üzərində layihələndiricinin şəxsi təcrübəsi əsaslanır. Lakin bu zaman subyektiv faktorların təsiri həddindən artıq böyük olur, metod o qədər də kefiyyətli hesab olunmur. Buna baxmayaraq, ciddi nəzəriyyə və empirik biliklər kifayət qədər olmadıqda, bu metod xərclərin qiymətləndirilməsi strategiyasının əsası kimi qəbul olunur.

Alqoritmik analiz metodu. Xərclərin qiymətləndirilməsinin bu metodunda bəzi alqoritmdən istifadə olunur. Qiymətləndirmə obyektiv və təkrarlanan olur. Hal hazırda belə alqoritmlər nəzəriyyəsi çox sürətlə inkişaf edir, lakin praktikada hələ ki, belə alqoritmlərin az bir qismi istifadə olunur. Bundan başqa, alqoritm yalnız o zaman düzgün nəticəyə gətirib çıxarır ki, hesablama üçün verilənlər (spesifikasiyalar) kifayət qədər dəqiq olsun, bu isə praktikda nadir halarda olur.

Addımlı analiz. Spesifikasiyalar enmə (azalan) yaxud artma (yüksələn) prinsipinə görə addımlı analizə əsasən verilir. Belə yarımalqoritmik proses ekspert qiymətləndirilməsi metodu ilə kombinasiya edilmiş (birləşdirilmiş) ola bilər.

Parkinson qanunu. Bir çox hallarda hər hansı bir işi (tapşırığı) yerinə yetirərkən, bu işin yerinə yetirilməsinin zəruru olub olmamasından asılı olmayaraq, onun üçün ayrılan vaxt sərf olunur. Hər bir icraçı sistem üzərində iş zamanı müəyyən tövhə (xeyir) verir və müəyyən vaxt sərf edir. Belə yanaşma digər metodların istifadə olunmasına əsaslanır, başqa sözlə desək, əgər artıq

qiymətləndirilmə aparılmışdırsa (məsələn, ekspertlərin köməyi ilə), işə cəlb olunan bütün icraçılar öz işlərini onun vacib və zəruri olub olmadığını nəzərə almadan yerinə yetirəcəklər. Belə yanaşma zamanı sistemin strukturu çox vaxt yaradıcı kollektivin tərkibi ilə müəyyən olunur (əgər sistemin üzərində 3 adam işləyirsə, çox güman ki, bu sistem üç seqmentdən ibarət olacqdır).

1.6.3 R*eley paylanmasına əsaslanan qiymətləndirmə.*

Hal hazırda EHM-lərin aparat vasitələrinin etibarlılığı nəzəriyyəsində alınan nəticələr proqram təminatı yaradılarkən çəkilən xərclərin və yaradılmaya sərf olunan vaxtın qiymətləndirilməsi üçün istifadə olunur. Müəyyən olunmuşdur ki, böyük sistemlər yaradılarkən məkilən ümumi xərclərin zamandan asılılığı (50 insan-ildən yuxarı) aşağıdakı tənlik vasitəsi ilə çox yaxşı əks olunur:

$$E(t) = K\left(1 - e^{-at^2}\right), \qquad (1)$$

Burada, E(t) – t zamanı anına çəkilən ümumi xərclər,

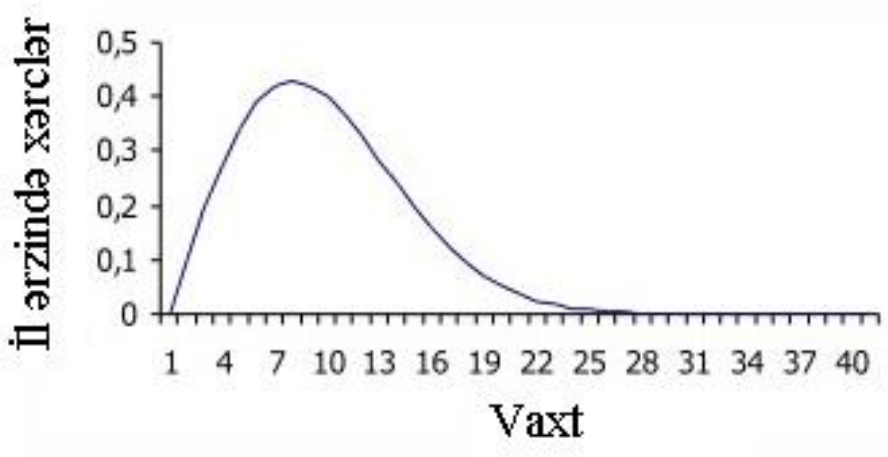
K – sistemin ümumi qiyməti,

a – vahid zaman kəsiyində maksimum xərcin xarakteristikasıdır.

Differensial formada ifadə olunan belə asılılıq Reyel əyrisi vasitəsi ilə əks olunur.

$$E\left(t\right) = 2\ K\ a\ t\ e^{-at^2}, \qquad (2)$$

burada E(t) – çəkilən xərcin sıxlığı yaxud illik xərclərdir (şək. 5.4).



Şəkil 1.11

Sistem yaradılarkən çəkilən xərclərin 60% müşaiyət olunma mərhələsinə sərf olunduğundan, heç də təəccüblü deyildir ki, əyrinin maksimumu sistemin yaradılması anına yaxındər, yəni ənənəvi olaraq sistem üzərindəki işin bitdiyi ana uyğundur.

Nəzarət anları işlərin yekunlaşdığı anları göstərir; onlar sistemin yaradılması vəziyyəti haqqında fikir yürütməyə imkan verirlər. Nəzarət anları layihənin rəhbəri tərəfindən sistemin yaradılmasına nəzarəti həyata keçirmək üçün planlaşdırılır. "proqram 90% yerinə yetmişdir" tipli vəziyyətlər nəzarət anlarına aid edilə bilməz, çünki, layıhənin rəhbəri proqram bitməyənə qədər, 90% in proqramın hansı həcmini əks etdirdiyini bilmək iqtidarında deyildir.

Çoxlu sayda "standart" hesab olunan nəzarət anları mövcuddur:

- funksional spesifikasiyaların buraxılması;
- müstəqil modulların layihələndirilməsinin başa çatması;
- səhvsiz modulların kompilyasiyası;
- modulların testləşdirilməsinin uğurla keçirilməsi və s.

Texniki tapşırıqda, proqramların yazılması müddətindən başqa, mütləq nəzarət anları da göstərilməlidir ki, yubadılan yarımçıq işlər vaxtında müəyyən olunsun.

Hər bir nəzarət anı üçün sistemin *dəyər*, *bitmə müddəti*, *çətinlik* kimi ümumi xarakteristikaları nəzərə alınmalıdır.

Kitabxanaçı ilə işləyərkən nəzarət anları üzrə uyğun formada hesabat sənədləri hazırlanmalıdır.

Kompilyatorlar və müxtəlif sazlanma vasitələri artıq çoxdan məlumdur. Hal hazırda sistemlərin yaradılmasına çox yaxşı kömək edən bir sıra yeni proqram vasitələri yaradılmışdır (və yaradılmaqdadır).

Belə vasitələr içərisində verilənlər bazalarının idarə olunması sistemlərini (**VBİOS**) xüsusilə qeyd etmək lazımdır. Bu sistemlər proqram təminatının yaradılmasının təşkilinin idarə olunmasına xeyli kömək edirlər. Qarşılıqlı istinadlar cədvəllərinə, təyini (atributiv) listinqlərinə, yaddaşın bölüşdürülməsi cədvəllərinə nəzarət üçün çox əlverişlidir.

İlkin kodda olan kitabxana modullarının aparılması imkanı olan ilk verilənlər bazasının idarə olunması sistemlərindən biri **İSDOS** Miçiqan Universitetində yaradılmışdır. Bu sistemə tapşırıqlarım müəyyən olunması dili və tapşırıqlarım müəyyən olunması analizatoru daxildirlər (**PSL/PSA**). Bu sistemə, proqramlar layihələndirilərkən, proqramlar arasındakı qarşılıqlı əlaqəni avtomatik yoxlamağa imkan verən interfeysin şərhi üçün dil daxildir. **RLS** (tələblərin müəyyən olunması dili) sistemi də yuxarıda göstərilən sistemə oxşardır və verilənlərin idarə olunması sistemi vasitəsi ilə tələblərin və interfeyslərin müəyyən olunması üçün təyin olunmuşdur.

Yaradılan proqram sisteminin etibarlılığının əsas parametrlərindən biri konseptual tamlıqdır, yəni struktur sadəliyi və uslub (tərz) oxşarlığı. Adətən bu, yəni konseptual tamlıq, sistemi yaradanların sayının minimuma çatdırılması hesabına əldə olunur. Baş proqramçının briqadasının təşkili də həmçinin sistemin konseptual tamlığına şərait yaradır, çünki layihələndirmə üzrə əsas işlər bu halda baş proqramçı tərəfindən yerinə yetirilir.

Layihələndiricilər arasında bu metod *intellektual proqramlaşdırma* adlanır. Bu yanaşmanı əks etdirən texniki sənəd proqramın məntiqi sturukturu hesab olunur (struktur sxemi).

Sistemin attestasiyası yaradılmanın bütün mərhələlərində həyata keçirilməlidir. Layihələndirmənin hər bir səviyyəsi, kodlaşdırma yaxud

testləşdirmə üçün göstərmək zəruridir ki, sistemə ixtiyari yeni hissə əlavə olunan zaman sistemin düzgünlüyü saxlanılır. Bu struktur sxemində öz əksini tapmalıdır. Düzgünlüyə nəzarət üçün *nəzarət analizi* adlanan analiz istifadə olunur. Nəzarət analizinin keçirilməsi periodik olaraq bütün icraçılar üçün planlaşdırılır. Nəzərdən keçirməl üçün sistemin hər hansı bir hissəsi seçilir. Analizi aparan hər bir iştirakçıya zəruri informasiyalar verilir (məsələn, texniki tapşırıq). Lazım gəldikdə sistemin yaradıcısı izahlar verir. Ekspertlər yaradılma mərhələsinin düzgünlüyü haqqında rəy verməlidirlər. Nəzarət analizinin məqsədi – səhvlərin aşkar olunmasıdır, onların aradan qaldırılması yox. İcraçı layihəni digərlərinə izah edərkən dəqiq müəyyən olmayan spesifikasiyaları yaxud verilməmiş (unudulmuş) şərtləri aşkar edə bilər.

FƏSİL II. Tətbiqi proqramlar paketinin layihələndirilməsi vasitələri

2.1 Tətbiqi proqramlar paketlərinin yaradılması mərhələləri

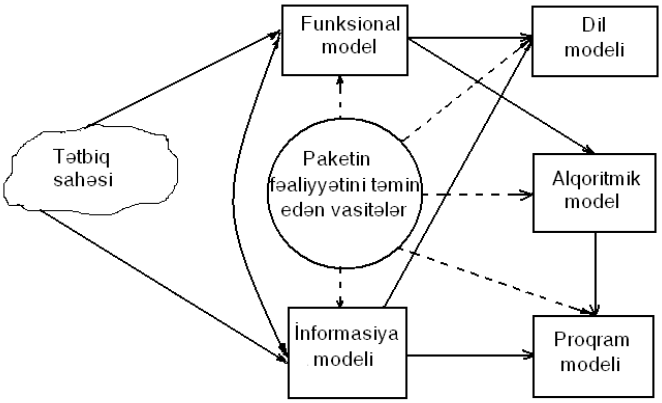
Tətbiqi proqramlar paketinin yaradılması adətən aşağıdakı sxem üzrə yerinə yetirilir (şəkil 2.1).

1. Tətbiq sahəsi müəyyən edilir və tədqiqat aparılır. Tətbiq sahəsinin informasiya və funksional modelləri yaradılır.

2. Tətbiq sahəsinin dil modeli yaradılır.

3. Məsələnin həlli metodları yaradılır və tətbiq sahəsinin alqoritmik modeli qurulur.

4. Alqoritmik və informasiya modellərinə əsasən tətbiq sahəsinin proqram modeli yaradılır.



Şəkil 2.1 Tətbiqi proqramlar paketinin layihələndirilməsi sxemi.

Tətbiq sahəsinin informasiya modeli bu sahənin əsas obyektlərini və bu obyektlər arasındakı münasibətləri müəyyən edir. O müəyyən verilənlər bazası şəklində realizə olunur. İnformasiya modelinin şərhi vasitəsi kimi verilənlərin şərhi dilindən istifadə olunur.

Tətbiq sahəsinin funksional modeli obyektlər üzərində görüləcək işlərin və bu işlərin yerinə yetmə şərtlərini müəyyən edir. Funksional modeli şərh etmək üçün çox müxtəlif formalizmlər tətbiq oluna bilər.

Tətbiq sahəsinin modeli, tətbiq sahəsində baş verən prosesləri modelləşdirmək üçün istifadə olunur. Lakin, funksional modelləşdirmədən fərqli

olaraq, bu məqsədlə digər vasitələrdən istifadə olunur. Tətbiq sahəsinin dil modeli müəyyən bir giriş dili şəklində realizə olunur. Dil modellərinin şərh etmək üçün adətən formal qrammatikalar tətbiq olunur.

Tətbiq sahəsinin alqoritmik modeli, funksional modeldə nəzərdə tutulmuş işlərin həyata keçirilməsinin, yəni yerinə yetirilməsinin potensial mümkün sxemlərini müəyyən edir. Bunun üçün paketin tətbiq sahəsində elementar hesab olunan problem-orientasiyalı əməllər çoxluğunu müəyyən etmək və bu əməllərin kompozisiyası (tərtibi) qaydalarını vermək zəruridir. Tətbiq sahəsinin alqoritmik modeli predikatlar cəbri, semantik şəbəkələr və s. şəklində verilə bilər.

Tətbiq sahəsinin proqram modeli, informasiya modeli vasitəsi ilə müəyyən olunmuş, alqoritmik modeldə nəzərdə tutulmuş obyektlər üzərində elementar əməlləri yerinə yetirir, yəni reallaşdırır. Tətbiq sahəsinin proqramm modeli funksional modullar kitabxanası və bu kitabxanadan istifadə qaydalarından ibarətdir.

Paketin tətbiq sahəsi bu sahənin proqramm modeli vasitəsi ilə tam şəkildə modelləşdirilir. Yerdə qalan modellərin heç biri mütləq deyildir və yalnız layişə sənədləri qismində istifadə oluna bilərlər. Lakin bir sıra faktları nəzərə almaqla, paketdə modellərin aşkar şəkildə istifadə olunması məqsədə uyğundur. Belə olan halda tətbiq sahəsinin hər hansı bir modelinin başqa bir modelə inikas olunmasının proqramm vasitələrinin, başqa sözlə desək paketin fəaliyyətini təmin edən vasitələrin yaradılması zərurəti meydana çıxır.

Tətbiqi proqramlar paketi yaradan istənilən instrumental sistem tətbiq sahəsi modelinin şərhi vasitələrinə və bu şərhlərin paketin fəaliyyətini təmin edən proqram vasitələrinə çevrilməsi (əks olunması) vasitələrinə malik olmalıdır.

Proqramm orientasiyalı proqramlaşdırma sistemi dedikdə aşağıdakı onluq başa düşülür.

$$\Pi = \langle L_1(G_1), F(G_2), A(R, N_1), P(B, N_2), J(L_2), T_1, T_2, T_3, T_4, M \rangle \qquad (1)$$

Burada,

$L_1(G_1)$- hər hansı bir G_1 formalizmi vasitəsi ilə verilən tətbiq sahəsinin dil modeli;

$F(G_2)$- hər hansı G_2formalizminin vasitəsi ilə verilən tətbiq sahəsinin funksional modeli;

$A(R, N_1)$- R, problem- orentasiayalı əməllər çoxluğu və N_1, bu əməllərin kompozisiyası qaydaları ilə verilən tətbiq sahəsinin alqoritmik modeli;

$P(B, N_2)$- B,funksional modullar kitabxanası və N_2,bu kitabxanalardan istifadə qaydaları ilə verilən tətbiq sahəsinin proqramm modeli;

$J(L_2)$ - L_2, verilənlərin şərhi dili vasitəsi ilə verilən tətbiq sahəsinin informasiya modeli;

T_1 - dil modelinin funksional və informasiya modellərinə əks etdirən dil prosessoru (məsələlər translyatoru);

T_2 - funksional modeli alqoritmik modelə çevirən planlaşdırıcı;

T_3 - alqoritmik modeli proqramm modelinə əks etdirən alqoritmin translyatoru;

T_4 - verilənlərin translyasiyası vasitələri;

M - sistemin işini idarə edən monitor.

Tətbiq sahəsinin informasiya modelini adətən sistemin verilənlər bazası, dil modelini sistemin giriş dili, proqramm modelini funksional hissə, $\langle T_1, T_2, T_3, T_4, M \rangle$ beşliyində isə sistem hissə adlandırırlar.

Tətbiq sahəsi müəyyən bir tətbiqi fəaliyyətə adekvat olan problem-orientasiyalı proqramlaşdırma sistemini tətbiqi proqramlar paketi adlandıracağıq. Bu tərif sistem hissəsinin bu və ya digər komponentləri olmadıqda alınan bir çox xüsusi hallarını əhatə edir. Bu halların hər birində sistem tətbiqi proqramlar paketi

olaraq qala bilər və yaxud qala bilməz. Bu sistem hissənin strukturu və onun tərkibi ilə deyil, tətbiq sahəsinin funksional modeli ilə müəyyən olunur.

Problem-orientasiyalı sistemlərin bəzi xüsusi hallarını nəzərdən keçirək. Ən mühüm və Ən geniş yayılmış halların üzərində dayanaq.

1. Sistemin giriş dilinə tətbiq sahəsinin alqoritmik modelinin şərhi vasitəsi kimi baxılır. Hər bir konkret halda istifadəçi öz məsələsini həll etmək üçün özü elementar əməllərin müəyyən kompozisiyasını qurur. Belə olan halda sistem problem-orientasiyalı proqramlaşdırma sisteminə prosedur giriş dili ilə çevrilir:

$$\Pi = \langle L_1(G_1), P(B, N_2), J(L_2), T_3, T_4, M \rangle \qquad (2)$$

(2)-də verilənlər dazası və verilənlərin translyasiyası vasitələri olmadıqda, alırıq:

$$\Pi = \langle L_1(G_1), P(B, N_2), J(L_2), T_3, M \rangle \qquad (3)$$

(3)- də monitor olmadıqda, sistemin ən sadə halı alınır, bu isə imkan verir ki, istifadəçi öz məsələsini həll edərkən tətbiq sahəsinin leksikası ilə əməliyyat aparsın:

$$\Pi = \langle L_1(G_1), P(B, N_2), T_3 \rangle \qquad (4)$$

Bu zaman istifadəçi sistemin giriş dilində həm tətbiq sahəsi obyektlərinin informasiya modellərini, həm də bu obyektlər üzərində yerinə yetiriləcək əməliyyatların reallaşdırılması alqoritmlərini müəyyən etməlidir.

2. Sistemin giriş dilinə tətbiq sahəsinin proqramm modelinin şərhi vasitəsi kimi baxılır. İstifadəçiyə müəyyən proqramm modulları kitabxanası və yerinə yetiriləcək modullar ardıcıllığını şərh etmək üçün giriş dili verilir. Kitabxana müəyyən bir monitorun idarəsi altında işləyir və müəyyən bir Verilənlər Bazasının İdarə olunması Sistemi ilə qarşılıqlı surətdə fəaliyyət göstərər. Ola bilsin ki, kitabxananın tərkibinə verilənlərin daxil edilməsi generatoru, verilənlərin çapı generatoru və yaxud verilənlərin translyasiyasının digər vasitələri daxil edilsin.

Bu halda Π aşağıdakı şəkildə olur:

$$\Pi = \langle P(B, N_2), J(L_2), T_4, M \rangle \qquad (5)$$

(5) halı (2) halından giriş dilinin leksikası və onun səviyyəsi ilə fərqlənir. Əgər (5) –də Verilənlər Bazasının İdarə olunması Sistemi ilə qarşılıqlı əlaqə nəzərə alınmamışdırsa və verilənlər bazasının idarə olunması, həm də verilənlərin translyasiyasının xüsusi vasitələri yoxdursa, onda sistem aşağıdakı vəziyyətə düşür:

$$\Pi = \langle P(B, N_2), M \rangle \qquad (6)$$

Bu hər hansı monitorun idarəsi altında işləyən və giriş-çıxış verilənlərinin ciddi formatına malik, alt proqramlar kitabxanasıdır. Monitor olmadıqda sistem modullar kitabxanasına çevrilir:

$$\Pi = \langle P(B, N_2) \rangle \qquad (7)$$

3. Verilənlər bazasının olmadığı hal. İstifadə olunan verilənlər toplusunun strukturu, ilkin verilənlərin təqdimat maketi və alınan nəticələrin çap forması proqramm modullarının məntiqi ilə qeyd olunur. Bu halda sistem aşağıdakı şəkildə olur:

$$\Pi = \langle L_1(G_1), F(G_2), A(R, N_1), P(B, N_2), T_1, T_2, T_3, M \rangle \qquad (8)$$

«Vilnüs» sistemi vasitəsi ilə yaradılan tətbiqi proqramlar paketləri (1) şəklində olurlar. Aydındır ki, «Vilnüs» sistemi vasitəsi ilə təkcə tətbiqi proqramlar paketləri deyil, digər prblem-orientasiyalı sistemlər də yaradıla bilər. Tətbiqi proqramlar paketlərinə mümkün giriş strukturlarını tələb olunan çıxış strukturlarına çevirən «qara qutu» kimi də baxmaq olar. Əks olunma, yəni çevrilmə qanunları müəyyən Q funksional münasibəti ilə verilə bilər:

$$Q \subset X \times Y \qquad (9)$$

burada

X- paketin mümkün giriş strukturları çoxluğu,

Y– paketin mümkün çıxış strukturları çoxluğudur.

Paketin funksiyalarının hər tərəfli şərhi tətbiq sahəsinin funksional modeli vasitəsi ilə verilir. Lakin, paketə qara qutu kimi baxılırsa, (9) münasibətindən istifadə etmək daha əlverişlidir. (9) münasibətinin daxil edilməsi problem-

orientasi-yalı proqramlaşdırma sisteminin korrektliyi üçün zəruri olan bəzi şərtləri qısaca və dürüst ifadə etməyə imkan verir.

Dil modelinin korrektliyi şərti : Z(9) funksional münasibəti ilə şərh olunan paketin giriş dili, aşağıdakı şərt ödəndikdə korrekt hesab olunur.

$$\exists F\left(\forall x\left(\forall y\left(\left(x \in X\ @\ y \in Y\ @\left(x, y\right) \in Q\ \right) \Rightarrow \exists l\left(l \in L @ F:\left(x, y\right) \rightarrow l\ \right)\right)\right)@$$

$$@\ \forall l\left(l \in L \Rightarrow \exists x\left(\exists y\left(\left(x \in X\ @\ y \in Y\ @\left(x, y\right) \in Q @ F:\left(x, y\right) \rightarrow l\ \right)@\right.\right.\right.$$

$$@\ \forall Z_1\left(\forall Z_2\left(F:\left(Z_{1,} Z_2\right) \rightarrow l \Rightarrow \left(Z_1 = x\right)@\left(Z_2 = y\right)\ \right)\right)\left.\left.\left.\left.\left.\right)\right)\right)\right) \quad (10)$$

Başqa sözlə (9) funksional münasibəti ilə mümkün olan ixtiyari (x,y) inikası, paketin L giriş dilində l düzgün mətni ilə şərh oluna bilər və giriş dilində ixtiyari düzgün l mətni (9) münasibətinin imkan verdiyi hər hansı (x,y) inikasını birqiymətli şərh edir.

Alqoritmik modelin korrektliyi şərti:

(9) funksional münasibəti ilə şərh olunan paketin tətbiq sahəsinin alqoritmik modeli, aşağıdakı şərt ödənildikdə korrekt hesab olunur:

$$\forall x\left(\forall y\left(x \in X\ @\ y \in Y\ @\left(x, y\right) \in Q \Rightarrow \exists a\left(a \in A @ a\left(x\right) = y\ \right)\right)\right)@$$

$$@\ \forall a\left(a \in A \Rightarrow \exists x\left(\exists y\left(x \in X\ @\ y \in Y\ @\left(x, y\right) \in Q @ a\left(x\right) = y\right)\right.\right. \quad (11)$$

başqa sözlə desək, (9) funksional münasibətinin imkan verdiyi ixtiyari (x,y) inikası üçün, tətbiq sahəsinin elementar əməlləri vasitəsi ilə əks olunan, heç olmasa bir Q alqoritmi vardır.

Proqramm modelinin korrektliyi şərti:

(9) funksional münasibəti ilə şərh olunan paketin tətbiq sahəsinin proqramm modeli aşağıdakı şərt ödənildikdə korrekt hesab olunur:

$$\forall a\left(a \in A \Rightarrow \exists p\left(\left(p \in P @ \forall x\left(\forall y\left(\left(\left(x, y\right) \in Q @ x \in X\ @\ y \in Y\ @\right.\right.\right.\right.\right.$$

$$@\left(a\left(x\right) = y @ q\left(x\right)\right) \Rightarrow \left(\left(p\left(x\right) = y\right)@ \wp\left(y\right)\right) V\left(\left(\left(x, y\right) \in QV\right.\right.$$

$$\left.\left.\left.\left.\left.V\left(\left(x, y\right) \in Q @ q\left(x\right)\right)\right) \Rightarrow\ p\left(x\right) = STOP\right)\right)\right)\right) \quad (12)$$

burada

q- ilkin verilənlər üçün məqbul hesab edilən predikatdır;

$\wp$ - çıxış verilənləri üçün məqbul hesab edilən predikatdır;

başqa sözlə, alqoritmik modelin imkan verdiyi ixtiyari alqoritm, problem modulların müəyyən kompozisiyası vasitəsi ilə realizə oluna bilər.

2.2 Tətbiq sahəsinin funksional modeli

Tətbiq sahəsinin funksional modeli, tətbiqi proqramlar paketi ilə reallaşdırılan, yəni yerinə yetirilən funksiyaları şərh edir. Bu məqsədlə müxtəlif formalizmlərdən istifadə oluna bilər. «Vilnüs» sistemində tətbiq sahəsinin funksional modelini şərh etmək üçün məqsədlər ağacından istifadə olunur. Belə olan halda tətbiqi proqramlar paketinə, paketin istismarı prosesində dəyişməz qalan, müəyyən T əsas (baş) məqsədin reallaşdırıl-ması vasitəsi kimi baxılır. Şəraitdən asılı olaraq baş məqsəd dəqiqləşdirilir. Dəqiqləşdirmə hər hansı H məhdudiyyətlər toplusunun verilməsi ilə yerinə yetirilir. Baş T məqsədinə çatmaq üçün $\{t_1, t_2, ..., t_n\}$ aralıq məqsədlərinin reallaşdırılması zəruri hesab edilir, belə ki, bu aralıq məqsədlərin özləri də öz növbəsində aşağı səviyyə alt məqsədlərə malik ola bilərlər.bütün səviyyələrin məqsədləri iyerarxik struktur əmələ gətirir və o məqsədlər ağacı adlanır. VƏ/VƏ YA qrafı şəklində təsvir olunan məqsədlər ağacı tətbiq sahəsinin funksional modelini müəyyən edir. VƏ/VƏ YA qrafının üç tip təpə nöqtələri vardır:

- VƏ tipli təpə nöqtələri; bu təpə nöqtələrindən məqsədlərin daha aşağı səviyyə altməqsədlərə bölünməsini göstərmək üçün istifadə olunur.

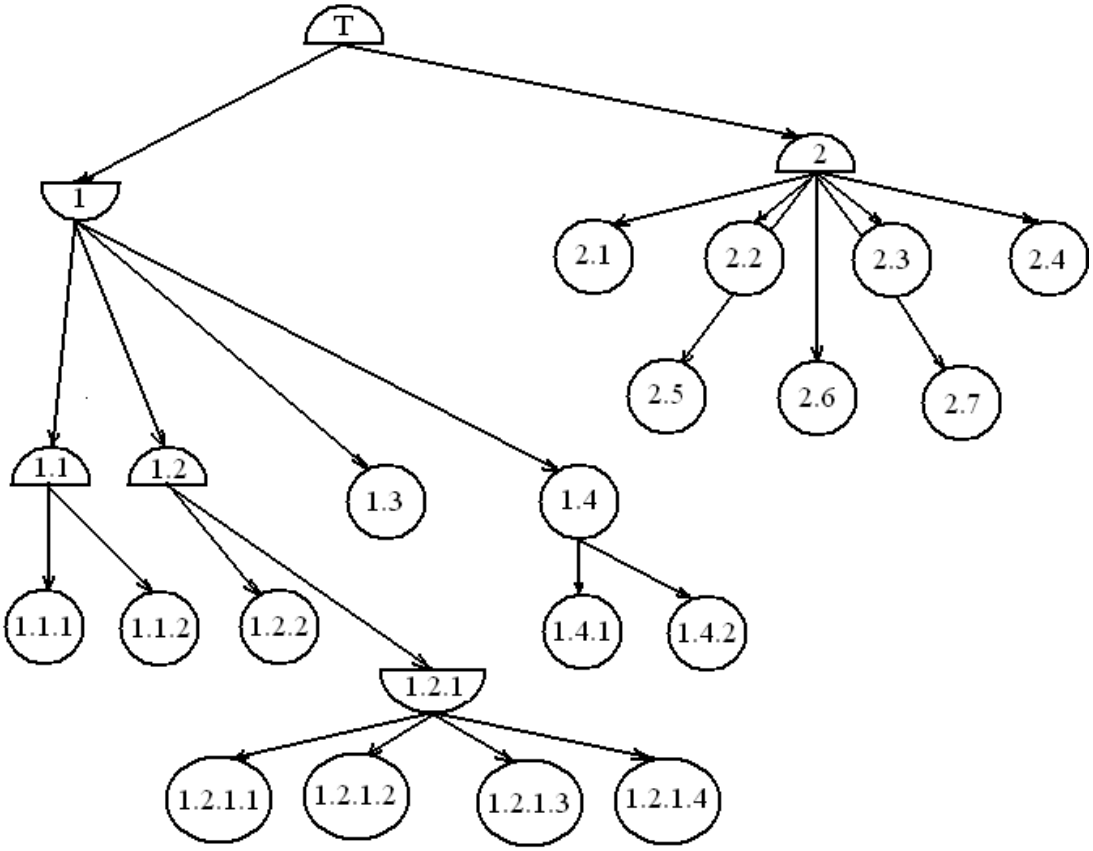
- VƏ YA tipli təpə nöqtələri; öz təpə nöqtələrindən məqsədlərin alternativ alt məqsədlərə kompozisiyasını göstərmək üçün istifadə olunur;

- İstisna edici və ya; bu təpə nöqtələrindən məqsədlərin rəqabət edici altməqsədlərə parçalanmasını göstərmək üçün istifadə olunur.

Bu və ya digər təpə nöqtəsi üçün tipin seçilməsi əksər hallarda sərbəstdir və istifadəçinin paketin tətbiq sahəsinə münasibəti ilə müəyyən oluna bilər.

Misal 1. Şəbəkə planlaşdırma və idarəetmə tətbiqi proqramlar paketi üçün T baş məqsədi aşağıdakı kimi ifadə oluna bilər: «mürəkkəb qarşılıqlı əlaqələrə malik böyük həcmlm proseslərin ilkin planlaşdırılmasını həyata keçirmək və planın yerinə yetirilməsi zamanı meydana çıxan şərtləri nəzərə almaqla müddəti təzələməli».

Şəkil 2.2-də uyğun məqsədlər ağacını əks etdirən VƏ VƏ YA qrafı təsvir olunmuşdur.



Şəkil 2. 2

Şəbəkə planlaşdırma və idarəetmə tətbiqi proqramlar paketinin funksional modeli.

Qrafda VƏ tipli təpə nöqtələri dairələrlə, VƏ YA tipli təpə nöqtələri yuxarı yarımdairələrlə, İSTİSNA EDİCİ VƏ YA tipli təpə nöqtələri isə aşağı yarımdairələrlə göstərilmişdir. Məqsədlər aşağıdakı kimi müəyyən olunmuşdur:

1. Verilənlər bazasının təşkili.

1.1 Verilənlər bazasının yüklənməsi.

1.1.1. Şəbəkə haqqında informasiyanın daxil edilməsi.

1.1.2. Normativ-sorğu informasiyaların daxil edilməsi.

1.2. Verilənlər bazasının təzələnməsi .

1.2.1. Şəbəkə haqqında informasiyaların təzələnməsi.

1.2.1.1. Bilavasitə təzələnmə.

1.2.1.2. İlkin düzəlişlər.

1.2.1.3. Verilənlər bazasında ilkin düzəlişlərin aparıl- ması.

1.2.1.4. İlkin düzəlişlərin yox edilməsi.

1.3. Verilənlər bazasının məzmununun çapı.

1.4. Arxivin təşkili.

1.4.1. Arxivdən görülən işlərin çıxarılması.

1.4.2. Çıxarılmış işlərin çapı.

2. Planlaşdırılan prosesin müvəqqəti parametrləri haqqında informasiyaların alınması.

2.1. İşlərin müvəqqəti xarakteristikaları haqqında verilənlərin toplanması.

2.2. Hadisələrin müvəqqəti xarakteristikaları haqqında verilənlərin toplanması.

2.3. Yerinə yetirilmiş işlərin siyahısının alınması.

2.4. Hadisələrin əmələgəlməsi proqnozunun alınması.

2.5. Yoxlama hadisələrinin verilmiş siyahısının əmələ gəlməsini təmin edən işlərin müvəqqəti xarakteristikalarının alınması.

2.6. Şəbəkə üzrə işlərin nomenklatur siyahının alınması.

2.7. Prosesin bütünlükdə vəziyyəti haqqında ümumi göstəricilərin alınması.

Misal qismində göstərilən məqsəd ağacı həm funksiyaların tərkibinə görə, həm də onların xırdalanması dərəcəsinə görə çox sadələşdirilmişdir. Bu misal üçün H məhdudiyyətlər toplusu aşağıdakı kimi müəyyən oluna bilər:

- nəzərdə tutulan şəbəkələrin siyahısı;

- ilkin düzəlişlərin siyahısı;

- təsdiqlənən (silinən) korreksiyaların siyahısı;
- verilmiş hesablamalarda nəzərdə tutulan korreksiyaların siyahısı;
- yoxlama hadisələrin siyahısı;
- böhranlılıq əmsalı;
- çıxış sənədlərinin çapı zamanı işlərin seçilməsi üsulu;
- çap zamanı işlərin çeşidlənməsi üsulu;
- çıxış sənədlərin hazırlanması üsulu.

İşlərin seçilməsi üsulu icraçının şifrinin göstərilməsi ilə verilir (yalnız verilmiş icraçının işləri çap olunur). Çeşidləmə üsulu rekvizitlərin adlarının sadalanması ilə verilir. Sənədin hazırlanması üsulu dedikdə, onun çap rejimlərini nəzərdə tutur.

2.3 Tətbiq sahəsinin dil modeli.

Tətbiqi proqramlar paketi üçün giriş dilini problem-orientasiyalı prosedur dil, əmr dili, təbii dilə yaxın olan dil və s. kimi yaratmaq olar. Prosedur dilin istifadə olunması ….. tələb edir. Bu tip dilin istifadəsi zamanı istifadəçi nəinki öz məsələsini dürüst ifadə etməli, həm də dilin imkan verdiyi elementar əməllər anlayışları vasitəsi ilə məsələnin həlli alqoritmini şərh etməlidir. Həll olunacaq məsələlər sinfi həddindən artıq geniş olduqda alqoritmik modelin qurulması çox çətinləşir və yaxud ümumiyyətlə mümkün olmur. Belə məsələləri həll edənlər isə elmi işçilər, mühəndis-texniki personal və s. olurlar. Planlaşdırma, uçot, yaxud digər iqtisadi fəaliyyətlə bağlı olan tətbiq sahələri üçün, bu yol, bir qayda olaraq qəbul olunmazdır.

Təbii dilə yaxın olan dilin reallaşdırılması böyük mexaniki çətinliklərlə əlaqədardır. Bundan başqa, öz tətbiq sahəsinin yaxşı bilən istifadəçi üçün sadə əməliyyatları təbii dildə uzun mətn şəklində yazmaq maraqlı deyildir. Ona daha qısa və aydın vasitələr lazımdır. Təbii dilə yaxın olan dillər, təsadüfi, yaxud bir istifadəçi üçün nəzərdə tutulan, informasiya-axtarış və sual-cavab sistemlərdə özünü daha yaxşı doğruldur.

Misal üçün belə dil aerokassalarda istifadə olunan sorğu sisteminin giriş dili kimi çox əlverişli ola bilər.

İqtisadi məsələlərin həlli üçün nəzərdə tutulan tətbiqi proqramlar paketlərinin giriş dili kimi əmr dilinin istifadə olunması daha əlverişlidir. İstifadəçi problem-orientasiyalı yaxud professional-orientasiyalı əmrlərin köməyi ilə öz məsələsini çox əlverişli şəkildə şərh edir. Odur ki, bu tip istifadəçilər üçün əmr dilləri sinfinin istifadəsi daha təbiidir. Bu sinfə direktiv dillər, cədvəl dilləri «menyu» tipli dillər və s. aid edilə bilər. Praktikada giriş dili prosedur və əmr dillərinin birləşməsindən ibarət olan tətbiqi proqramlar paketləri daha geniş yayılmışlar və istifadə olunurlar. Belə tətbiqi proqramlar paketləri qurulmuş (standart) tətbiqi proqramlar paketləri adlanırlar. Bu tip giriş dilləri problem-orientasiyalı prosedur dillərin malik olduğu bütün üstünlüklərə və çatışmamazlıqlara malik olurlar.

«Vilnüs» sisteminin köməyi ilə yaradılan tətbiqi proqramlar paketlərinin giriş dili və cədvəl tipli əmr dilidir. Cədvəl «Hesablama tapşırığı blankı» (HTB) adlanır. O bir neçə vərəqə bölünür. Belə formaya malik giriş dili bir sıra üstünlüklərə malik olur:

- cədvəlin əyaniliyi hesabına cədvəl tipli dilin öyrənilməsi digər tip dillərə nisbətən daha sadədir;

- istifadəçi giriş dilində proqramm tərtib edərkən daha az səhvlərə yol verir, çünki dilin sintaksisi cədvəl vasitəsi ilə qeyd olunur;

- cədvəl tipli dilin reallaşdırılması direktiv yaxud «menyü» tipli dillərin reallaşdırılmasınanisbətən daha sadədir;

- cədvəl dillərinin istifadəsi zamanı paketin giriş dili yeni vərəq əlavə etməklə, asanlıqla dəyişdirilə bilir;

- cədvəlin ayrı-ayrı vərəqləri müəyyən alt sinif istifadəçilər üçün istiqamətləndirilmiş imkanları bir yerə toplanmaqla tərtib oluna bilərlər,başqa sözlə desək paketin problem - orientasiyalı giriş dilindən professional-orientasiyalı alt dil ayırmaq mümkündür;

- bir tətbiqi proqramlar paketi üçün bir neçə müstəqil cədvəllər verilə bilər, başqa sözlə, bir paket bir-birlərindən istifadə olunan terminlərinə görə,məqsədlərin təfsilat dərəcəsinə görə və s. görə bir-birlərindən fərqlənən bir neçə giriş dilinə malik ola bilər;

- alınan proqramların effektivliyi istifadəçinin peşəkarlığı ilə deyil, paketin sistem imkanları ilə müəyyən olunur;

İstifadəçinin həll etdiyi məsələnin şərhi ilə cədvəl həmin məsələnin təbii sənədləridir.

Cədvəl tipli dillərin müəyyən çatışmayan cəhətləri də vardır:

- belə dillər (və ixtiyari digər əmr dilləri) vasitəsi ilə şərh olunan məsələlər sinfi, bir qayda olaraq, prosedur dillərlə şərh olunan məsələlər sinfinə nisbətən dardır;

- mürəkkəb məsələlər üçün cədvəlin vərəqlərinin sayı kifayət qədər böyük ola bilər, bu isə istifadəçi üçün yeni problemlər yaradır;

- cədvəllər tipoqrafik üsulla yaradıldığından, onların modifikasiyası əlverişli deyildir;

- cədvəl forması dəyişən uzunluqlu siyahıların verilməsi üçün əlverişli deyildir.

Misal 2. Birinci misalda «Şəbəkə planlaşdırma və idarəetmə» tətbiqi proqramlar paketinin tətbiq sahəsinin funksional modelinə baxdıq. İndii isə elə bu paket üçün giriş dilinin mümükün variantlarından birinə baxaq. Paketin giriş dili üç vərəqdən ibarət cədvəl tipli dildir. Birinci vərəqdə tətbiqi proqramlar paketini (problemi) idintifikasiya edən başlıq,istifadəçinin adı və hesablaşma seansı göstərilir. Vərəqin qalan hissəsi məsələnin həlli zamanı istifadə olunan ilkin verilənlərin şərhi üçün nəzərdə tutulmuşdur. Burada şəbəkələr («Tema» qrafası), alt şəbəkələr («Fraqment» qrafası) və məsələ həlli zamanı nəzərdə tutulan ilkin düzəlişlər («Data kalendarnoy privyazki» qrafası) sadalanırlar. «№ n/n» qrafası istifadəçi məsələlərini, yəni ayrı-ayrı məsələləri adlandırmaq üçün istifadə olunur. İxtiyari məsələnin həlli zamanı birinci vərəqənin doldurulması mütləqdir.

İkinci vərəq verilənlər bazasının aparılması xidməti üçünnəzərdə tutulmuşdur və verilənlər bazasının təşkili zamanı istifadə olunur. Bu vərəqdə aşağıdakılar göstərilə bilər:

1. Şəbəkə verilənlər bazasına yüklənir («Vvod» qrafası);

2. İlkin düzəlişlər daxil edilir («Korrektirovka» qrafası);

3. Bəzi ilkin düzəlişlərtəsdiq olunur, yaxud silinir;

4. Normativ-sorğu informasiyaları daxil edilir yaxud təzə-lənir (« Vvvod şifr» və «Korrek. şifr» qrafaları);

5. İşlərin arxivə göndərilməsi tələb olunur («Data udaleniya» qrafası);

6. İşlər haqqında ilkin informasiya çap olunur («PİR» qrafası).

İkinci vərəq birinci vərəqlə birlikdə istifadə olunur. Bu zaman üçüncü vərəqdən də istifadə oluna bilər, lakin bu mütləq deyildir.

Üçüncü vərəq şəbəkə planlaşdırma və idarəetmə xidməti işçiləri üçün nəzərdə tutulmuşdur. O şəbəkələrin hesablanması zamanı birinci vərəqlə birlikdə istifadə olunur. Bu zaman ikinci vərəqi istifadəsi mütləq deyildir. Üçüncü vərəqin köməyi ilə istifadəçi aşağıdakıları tələb edə bilər:

- işlərin müvəqqəti xarakteristikaları haqqında verilənlər (RP1.1qrafası);

- hadisələrin müvəqqəti xarakteristikaları haqqında verilənlər (RP1.2 qrafası);

- hadisələrin əmələ gəlməsi proqnozu (RP1.4 qrafası);

- yerinə yetirilmiş işlərinsiyahısı (RP1.3 qrafası);

- işlərin nomenklatur siyahısı (RP1.5 qrafası);

- prosesin bütünlükdə vəziyyəti haqqında ümumi göstərici-lər («Analiz» qrafası);

- alt şəbəkənin işləri yaxud, hadisələrin müvəqqəti xarakteristikaları haqqında verilənlər.

Göstərici çıxış sənədində informasiyaların qaydaya salınması üsulunu müəyyən edə bilər, həm də çapın formatını idarə edə bilər, kritik zonanı ayıra bilər.

İxtiyari məqsəd ağacı üçün «Vilnüs» sistemi müxtəlif giriş dillərinin qurulmasına imkan verir. Bu dillər bir0birindən qrafaların yerləşməsinə, onların

ayrı-ayrı vərəqlərdə paylanmasına, həm də qrafaların paylanmasına görə fərqlənə bilər. Yeganə tələb (10) şərtinin ödənməsidir.bundan başqa hər bir vərəqdə tapşırığın, yəni məsələnin nömrəsi («№ n/n» qrafa-sı) mütləq göstərilməlidir. Dildə vəziyyətləri şərh etmək üçün «adi hal» ifadəsi ilə xarakterizə olunan susmaya gbrə aparatından geniş istifadə etmək olar. Əgər müəyyən sinif istifadəçilər üçün standart olmayan əməllər ümumiyyətlə lazım deyildirsə, onda uyğun qrafalar cədvəldən çıxarıla bilərlər. Paketin susmaya görə qəbul etdiyi qiymətlər, paketin giriş dilinin şərhi zamanı, «Vilnüs» sisteminin giriş dilində verilir.

İlkin verilənlər paketinin giriş dilində şərh olunmuşlar, yalnız məsələnin həllində iştirak edən uyğun obyektlərin (məs. şəbəkələr) adları göstərilir. Verilənlərin məntiqi strukturu paketin informasiya modeli vasitəsi ilə qeydə alınır. Nəzərdə tutulur ki, bu verilənlər artıq verilənlər bazasında saxlanılırlar. Əks halda, verilənlər «Vilnüs» sisteminin nüvəsinə daxil olan daxiletmə generatoru vasitəsi ilə verilənlər bazasına yüklənə bilər. İstifadəçi verilənlər bazasının yüklənməsi zamanı verilənlər bazasında bu və ya digər strukturun hansı ilkin verilənlərdən təşkil olunduğunu şərh etməməlidir. Bu proqramm yolu vasitəsi ilə müəyyən olunur. İlkin sənədlərin məntiqi strukturu tətbiq sahəsinin informasiya modeli vasitəsi ilə qeydə alınır.

2.4 Tətbiq sahəsinin alqoritmik modeli.

Tətbiq sahəsinin alqoritmik modeli istifadəçi məsələlərinin həlli zamanı istifadə olunan alqoritmlər toplusunu müəyyən edir. Bu alqoritmlər tətbiq sahəsinin elementar əməlləri şərh edən anlayışlar vasitəsi ilə müəyyən olunurlar. Belə müəyyən etmə üsulu paketi yeni məsələlərin həlli alqoritmləri ilə genişləndirməyə, paketdə olan alqoritmlərdə dəyişikliklər etməyə və s. imkan verir. Alqoritmik model müxtəlif üsullarla şərh oluna bilər. Onların ən sadəsi hər bir alqoritmin ayrılıqda yazılmasıdır. Bu üsul həm də ən əlverişli üsuldur, belə ki, hesablama prosesinin planlaşdırılması bu halda uyğun alqoritmin seçilməsinə gətirir. Lakin, alqoritmlərin sayı çox olduqda belə üsul səmərəsizdir və praktikada az hallarda

tətbiq olunur. Adətən, alqoritmləri aşkar şəkildə vermək mümkün deyildir, odur ki, alqoritmin qurulması qaydalarını tətbiq sahəsinin elementar əməlləri çoxluğunda vermək lazım gəlir. Belə qaydalar kompozisiya (birləşmə, toplanma) qaydaları adlanırlar və onları müxtəlif formalizmlərin köməyi ilə vermək mümkündür. Bu məqsədlə, məsələn, əmələ gətirən qrammatikalar, semantik şəbəkələr və s. istifadə oluna bilər. Qaydaları həmçinin hər hansı cəbr şəklində aksimlar kimi də vemək olar. Bu və ya digər formalizmin seçilməsi həm əmələ gələn alqoritmlərin sayından, həm də onların strukturundan asılıdır.

«Vilnüs» sistemində alqoritmik modeli şərh etmək üçün semantik şəbəkələr aparatı seçilmişdir. Hal -hazırda bu aparat daha mükəmməl araşdırılmışdır. Ümumi halda semantik şəbəkə aşağıdakı kimi müəyyən olunur:

$$C = \langle x_1, x_2, x_3, ..., x_n, R_1, R_2, ..., R_m \rangle ,$$

burada

$x_1, x_2, x_3, ..., x_n$ - hər hansı qeyd olunmuş çoxluqlardır;

$R_1, R_2, ..., R_m$ - bu çoxluqların elementləri üzərində müəyyən olunmuş elementlər sistemidir.tətbiq sahəsinin alqoritmik modelinin şərhi üçün istifadə olunan semantik şəbəkələr, adətən qraf modeli adlandırılır. Plpnlaşdırmanın səmərəliliyi baxımdan semantik şəbəkələr alqoritmlərin bir başa sadalanmasından çox az geri qalırlar. Bu aparatın tətbiqi şəbəkədə təpə nöqtələrinin sayı həddindən çox olduqda əlverişli deyildir, çünki belə olan halda şəbəkə bütünlüklə operativ yaddaşa yerləşdirilə bilmir. «Vilnüs» sistemində aşağıdakı növ semantik şəbəkələrdən istifadə olunur:

$$C=< V,R_1,R_2> \qquad (13)$$

burada

V- paketin tətbiq sahəsinin elementar əməllələrinə uyğun təpə nöqtələri çoxluğudur;

R_1- «ardınca getməli» münasibətlərini şərh etməli üçün istifadə olunan bütöv tillər çoxluğu;

R_2- «alternativ olmaqla ardınca getməli» münasibətlərini şərh etmək üçün istifadə olunan qırıq-qırıq xətlərlə göstərilən tillər çoxluğu.

Belə şəbəkələrin göməyi ilə paketin tətbiq sahəsinin ele-mentar əməllərinin bütün mümkün ardıcıllıqları şərh olunur, yəni bütün mümkün alqoritmlər çoxluğu şərh olunur. Belə tip seman-tik şəbəkələrə «iş-təpə nöqtəsi» tipli şəbəkə qrafikləri kimi, planlaşdırma prosesinə isə işlərin yerinə yetirilməsini uyğun planının tərtib olunması prosesi kimi baxıla bilər. Bu planı məsələnin həlli planı adlandıracağıq.

Verilmiş məqsədlər ağacı üçün semantik şəbəkə aşağıdakı kimi qurulur: məqsədlər ağacı ilə müəyyən olunan istifadəçinin məqsədləri, paketə nəzərən xarici hesab olunurlar. Hər bir belə məqsədi reallaşdırmaq üçün bir ya və bir necə verilənlər toplusu təşkil etmək zəruridir. Bu verilənlər toplularına paketin daxili məqsədləri kimi baxıla bilər. Hər bir verilənlər toplusu paketin tətbiq sahəsinin elementar əməllərini hər hansı ardıcıllığı, başqa sözlə hər hansı alqoritm ilə yaradılır. Bu və ya digər aralıq nəticələrdən asılı olaraq alqoritmin reallaşdırılmasının alternativ üsulları meydana gələ bilər. Əməllərə uyğun təpə nöqtələrini (bu əməllər yerinə yetən zaman məqsəd verilənlər topluları yaranır) məqsəd təpə nöqtələri adlandıraq. Bütün məqsəd təpə nöqtələri çoxluğu

$$W = \{W_1, W_2, ..., W_m\} \quad , \quad m \geq n \qquad (14)$$

$$W_1, W_2, ..., W_m \qquad (15)$$

altçoxluqlarına bölünür. Bir alt çoxluğunun məqsəd təpə nöqtə-ləri birinci səviyyənin alt məqsədlərindən birinin reallaşmasını təmin etməlidir. Uyğun məqsəd verilənlər toplu-ların yaradılması alqoritmləri hər hansı bir şəbəkə əmələ gətirir. Bu şəbəkə ye-ganə başlanğıc təpə nöqtəsinə və bir son təpə nöqtəsinə malik-dir, özü də bunlar yalancı təpə nöqtələridirlər, başqa sözlə de-sək, bu təpə nöqtələrinə paketin tətbiq sahəsinə heç bir əməl uyğun deyildir. Başlanğıc təpə nöqtəsinin START adlandırıb S ilə son təpə nöqtəsini isə FİNİSH adlandıraraq F ilə işarə edək.

Bir semantik şəbəkə ilə şərh olunan alqoritmlər toplusu hər hansı bir proqramlar toplusu ilə reallaşdırılır. Proqramlar toplusunu prosessor, uyğun

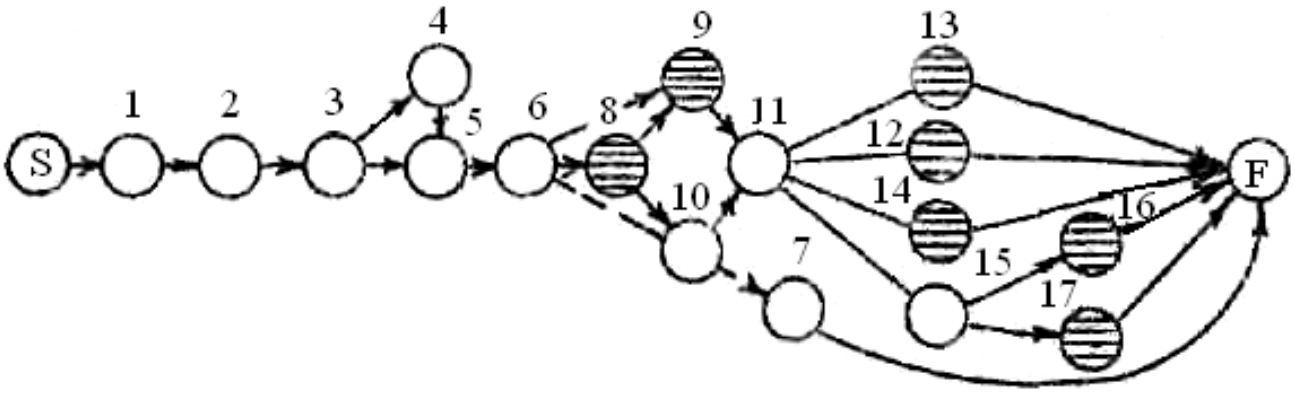
şəbəkəni isə prosessorun alqoritmik modeli adlandıraq. Beləliklə «Vilnyus» sistemi vasitəsilə yaradı-lan tətbiqi proqramlar paketləri üçün tətbiq sahəsinin alqoritmik modeli semantik şəbəkələrin nizamlanmış çoxluğu ilə verilir.

$$C = \langle C_1, C_2, ..., C_n \rangle \quad , \quad C_i = \langle V_i, R_{1i}, R_{2i} \rangle \quad , \qquad (16)$$

burada, i=1,2,…,n; n- paketin məqsəd ağacında birinci səviyyə alt məqsədlərin sayıdır. Hər bir semantik şəbəkə hər hansı bir prosessorun alqoritmik modelidir. Prosessor, məlum olduğu kimi istifadəçinin uyğun məqsədlərini proqramm reallaş-dırılmasını təmin edir. Hər bir şəbəkə müəyyən bir müəyyən bir sayda məqsəd təpə nöqtələri saxlayır. Konkret məsələnin həlli zamanı məqsəd ağacında paketin giriş dilində şərhinə əsasən uyğun terminal təpə nöqtələri bu təpə nöqtələrinə isə proses-sordan hər birinin semantik şəbəkədə məqsəd təpə nöqtələri müəyyən olunur. Sonra isə bu şəbəkələrdən, tələb olunan məq-səd verilənlərinin strukturunun reallaşdırılması təmin edən, alt şəbəkələr ayrılır. Bu alt şəbəkələr məsələnin həlli planının əsasıdır.

Misal 3. Misal 1- dəki məqsədlər ağacı iki birinci səviyyə altməqsədə malikdir. Uyğun olaraq tətbiq sahəsinin alqoritmik modeli bu altməqsədlərin reallaşdırılması alqoritmini şərh edən iki semantik şəbəkədən ibarətdir. Bu modelə əsaslanaraq yara-dılan şəbəkə planlaşdırma və idarəetmə tətbiqi proqramlar pake-tinin iki prosessoru vardır. Bu verilənlər bazasını yaradan Ввод prosessoru, digəri isə planlaşdırılan prosesin müvəqqəti parametirlərini hesablayan VREMYA prosessorudur. Şəkil 2.3 - də VREMYA prosessorunun semantik şəbəkəsi göstərilmişdir.

Şəkildə məqsəd təpə nöqtələri ştrixlənmişdir. İstifadə olunan elementar əməllər çoxluğu cədvəl 1-də verilmişdir. Bu əməllərin şəbəkənin təpə nöqtələrinə uyğunluğu şəkil 5-də verilmişdir.



Şəkil 2.3 VREMYA prosessorunun semantik şəbəkəsi.

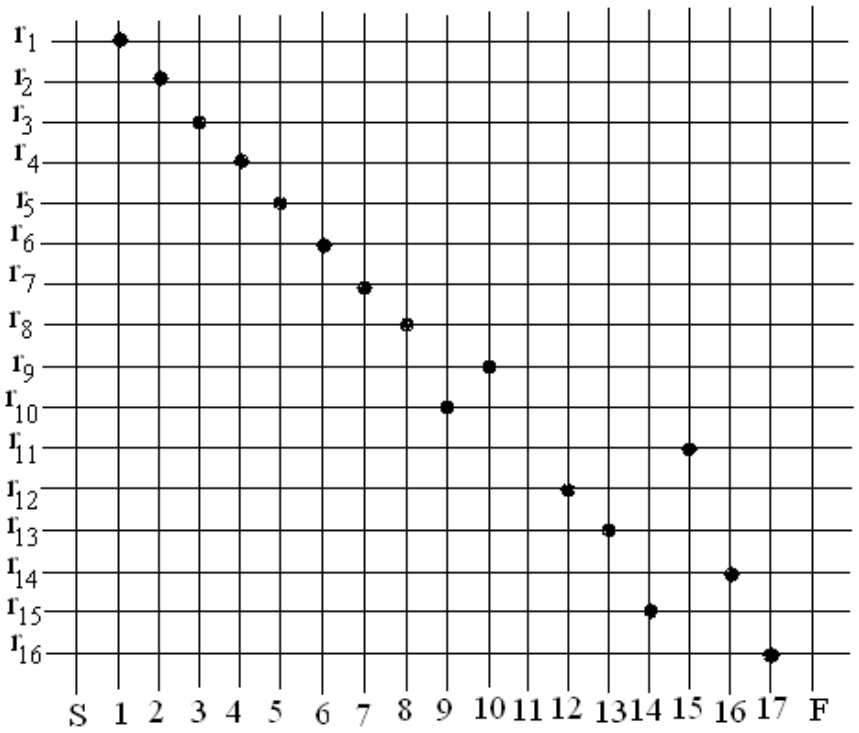
Planlaşdırılan prosesin müvəqqəti parametrlərinin hesablanmasında istifadə olunan elementar problem - orientasiyalı əməllər.

Cədvəl 1.

№	Sıra nömrəsi	Əməlin şərhi
1.	$r_1(G,S)$	Verilmiş kriteriyaya əsasən şəbəkənin nizamlanması.
2.	$r_2(G)$	Hadisələrin nömrələrinin sıxlaşdırılması, başqa sözlə, hadisələri yenidən elə nömrələməli ki, hadisələrin Ən böyük nömrəsi şəbəkədəki hadisələrin sayına bərabər olsun.
3.	$r_3(G)$	Şəbəkənin korrektliyinin yoxlanılması (yalan ilkin hadisələr, tupiklər və s.). Səhv işlərin siyahısının tərtib edilməsi və çapı.
4.	$r_4(G,F_1)$	Hadisələrin nömrələnməsinin yenidən kodlaşdırılması, başqa sözlə,
5.	$r_5(G)$	hadisələr yenidən elə nömrələnməli ki, (sıxılmış nömrələnmə saxlanılmaqla) ixtiyari işin başlanğıc hadisənin kodu onun son hadisəsinin kodundan kiçik olsun. Şəbəkənin kənarlarının olmamasına görə yoxlanılması. Kontura düşən işlərin siyahısının tərtib edilməsi və çapı.
6.	$r_6(G)$	Verilmiş hadisələr siyahısının yerinə yetməsini təmin edən alt
7.	$r_7(G,F_2)$	şəbəkənin ayrılması. İşlərin zaman xarakteristikalarının hesab-lanması.

8.	$r_8(G,G_1)$	Verilmiş direktiv məhdudiyyətlər daxilində işlərin zaman xarakteristikalarının hesab-lanması.
9.	$r_9(G,M_1)$	İşlərin verilmiş zaman xarakteristikalarına görə hadisələrin zaman
10.	$r_{10}(G,D,M_1)$	xarakteristikala-rının hesablanması.
11.	$r_{11}(M_1,M_2)$	Bütünlükdə prosesin vəziyyəti haqqında ümumi göstəricilərin tərtib edilməsi və çapı.
12.	$r_{12}(G)$	İşlərin zaman xarakteristikalarını saxla-yan RP1.1 formasının tərtibi və çapı.
13.	$r_{13}(M_1)$	Hadisələrin zaman xarakteristikalarını saxlayan RP.2 formasının tərtibi və çapı.
14.	$r_{14}(M_2)$	İşlərin nomenklatur siyahısını saxlayan RP1.5 formasının tərtibi və çapı.
15.	$r_{15}(M_1)$	Hadisələrin əmələ gəlməsi (baş verməsi) proqnozunu saxlayan RP1.4 formasının tərtibi və çapı.
16.	$r_{16}(M_2)$	

Semantik şəbəkənin təpə nöqtələrinin elementar əməlləri uyğunluğu aşağıdakı şəkildə verilmişdir.



Şəkil 2.4

2.5. Tətbiq sahəsinin informasiya modeli.

Paketin tətbiq sahəsi istifadəçi üçün qarşılıqlı fəaliyyətdə olan proseslərin və obyektlərin toplusudur. Bu proseslər və obyektlər hər hansı bir formal modellər vasitəsilə modelləşdirilirlər. Məsələ həlli zamanı hər hansı bir modeldən istifadə etmək üçün onu hər hansı bir kod vasitəsilə işarə etmək və paketin verilənlər bazasında qeyd etmək lazımdır. Bu kod müəyyən qaydalar əsasında tərtib olunmalı və müəyyən struktura malik olmalıdır. Beləliklə, ona hər hansı bir formal dildə mətn kimi də baxmaq olar. Bu tip yazıları gələcəkdə verilənlərin dili adlandıracağıq.

Verilənlər bazası istifadəçi üçün modellər toplusu kimi təqdim olunur. Modelləşdirilən obyetin hər bir nüsxəsi ayrıca mətn şəklində hər hansı bir Q verilənlər dilində şərh olunur, mətnlərin bu dildəki toplusu isə bir tipli obyektləri yaxud proses-sorları şərh edir.

Q dilində mətni obyektin informasiya modeli, Q dilində verilmiş mətnlər toplusunu obyektlər sinifinin informasiya modelini, müxtəlif siniflərin informasiya modellərinin verilmiş toplusunu isə tətbiq sahəsinin informasiya modeli adlandıracağıq.

İstifadəçi üçün tətbiq sahəsinin informasiya modelinin tər-tibi və saxlama vasitəsi giriş sənədləridir. Sinfin informasiya modelinə, bu sinfin ayrı-ayrı elementlərinin şərhi üçün istifadə olunan hər hənsı bir giriş sənədləri toplusu qarşı qoyulur. Mo-delin giriş sənədlərinə dekompozisiyası (parçalanması, ayrılma-sı), ümumiyyətlə desək, ixtiyaridir, bir qiymətli deyildir və praq-matik təsəvvürlərlə şərtləndirilir. Obyektin xassələrinin bəziləri müəyyən sənədlərlə, digər xassələri isə digər sənədlərlə şərh oluna bilər. Bəzi hallarda təmamilə eyni xassələrin müxtəlif giriş sənədlərinin köməyi ilə şərh olunması daha məqsədə uyğundur. Hər bir D giriş sənədi hər hansı bir K verilənlər dilində mətndir və bu mətnin yazıldığı K dili Q dilindən fərqlidir.

K_1, K_2 ,..., K_n (burada n-giriş sənədlərinin sayıdır) dillərinimüəyyən etmək üçün «Vilnüs» sisteminin giriş dilinin alt çoxluğu olan verilənlərin şərh dilindən

istifadə olunur. Verilənlərin şərh dilinin köməyi ilə hər bir K_i (i=1,2,...,n) dili üçün onun qramatikası müəyyən olunur və sonra isə operator semantikası müəyyənləşdirilir. bütün K_i (i=1,2,...,n) dilləri ümumi əlifba üzərində müəyyən olunmuşdur. Əlifba isə «Rekvizitlərin şərhi blankı»nın köməyi ilə müəyyən olunur.

«Vilnüs» sisteminin verilənlərinin şərhi dilinin köməyi ilə müəyyən olunan dillərin terminal simvollarına xas olan ümumi xüsusiyyətləri nəzərdən keçirək. Bu simvollar göstəricidirlər. Onlar çoxlu ada və müəyyən təsvir formalarına malikdirlər. Hər bir təsvir forması ASCİİ kodlar cədvəlinin simvollarından ibarət sətirdir. Təqdimatlar yalnız dillərin semantikası səviyyəsində bir-birindən fərqlənirlər. Sintaksis səviyyəsində isə onlar eynidirlər. Göstəricilərin mümkün qiymətlər çoxluğuna aid olması hər hansı bir pedikatın köməyi ilə müəyyən olunur. Lakin bu müəyyən etmə bir qiymətli deyildir. Ona görə də terminal simvolları onların təqdimatına görə müəyyən etmək olmaz.

«Vilnüs» sistemində verilənlərin şərhi dili hər bir terminal simvol üçün üç ad nəzərdə tutulur. Birinci sistem adıdır və qrammatikanın, həm də dilin semantikasının şərhi zamanı istifadə olunur. İkinci ad istifadəçi adı adlanır və layihələndirilən tətbiqi proqramlar paketinin giriş dilinin vasitələri ilə hər hansı bir göstəriciyə istinad etdikdə istifadəçi tərəfindən istifadə olunur. Üçüncü ad çıxış sənədlərində qrafaların adlarının tərkibi zamanı çıxış generatoru tərəfindən istifadə olunur.

İxtiyari K_i (i=1,2,...,n) dilinin qramatikası şablonların köiəyi ilə müəyyən olunur. Qrammatika aşağıdakı kimi müəyyən olunur:

$$\boldsymbol{G = <H,T,P,Ş>} \qquad (17)$$

burada

H- mürəkkəb rekvizitlər adlanan, sonlu qeyri terminal simvollar çoxluğu;

T- terminal simvollar çoxluğu; onlar elementar rekvizitlər (göstəricilər) adlanırlar;

P- elementar rekvizitlərdən mürəkkəb rekvizitlərinin qurulması qaydalarını verən sonlu çoxluq;

Ş- giriş sənədində şablonun adı ilə adlandırılan aksiom.

T çoxluğu «rekvizitlərin şərhi blankı» vasitəsi ilə müəyyən olunan əlifbanın hər hansı bir alt çoxluğudur. Bundan başqa

$$H \cap T = \varnothing \tag{18}$$

və

$$\psi \notin (H \cup T) \tag{19}$$

Şablonun adı Ş hərfi ilə başlamalı və uzunluğu 8 simvoldan çox olmamalıdır. P-dən olan hər hansı bir element (a, b) nizamlanmış cütüdür,

$$\begin{aligned} a &= \varphi A \psi \\ b &= \varphi w \psi \end{aligned} \tag{20}$$

burada φ, ψ, w $(HUT)^*$ -dan olan, hər hansı bir sətirlərdirlər.

A- aksiom yaxud qeyri terminal simvoldur.

$(HUT)^*$ -(HUT) əlifbası üzərində qurulmuş sətrlər çoxluğudur.

(a, b) hasilatın aşağıdakı kimi işarə edəcəyik.

$$a \longrightarrow b$$

«Vilnüs» sisteminin verilənlərin şərhi dili yalnız iki tip hasilatdan istifadə etməyə imkan verir.

$$a \to b, \quad b \in ((HUT) - a)^*, \quad a \in (HUH) \tag{22}$$

və

$$a \to ab, \quad a \in H, \quad b \in (HUT) \tag{23}$$

birinci tip mürəkkəb rekvizitlərin şərhi üçün istifadə olunur, ikinci tip isə təkrarlanan rekvizitlər, yaxud təkrarlanan rekvizitlər qrupunun şərhi üçün istifadə edilir. Təkrarlanan qrupların bir-birinin tərkibinə daxil olması dərəcəsi üçə bərabərdir.

«Vilnüs» sisteminin verilənlərin şərhi dilinin vasitələri ilə giriş sənədlərini şərh edərkən, bu sənədin strukturunu müəyyən edən qrammatika ağac şəklində təsvir olunur. Qramatikanın belə təsviri sənədin formasının qrafik təsvirinə adekvatdır və aydındır ki, bu paketin layihələndirilməsi zamanı daha əlverişlidir. Ağacın hər bir qeyri terminal təpə nöqtəsi üçün verilənlər dilinin sintaktik

strukturuna uyğun olan əməliyyat semantikası müəyyən oluna bilər. O, bu strukturun semantik korrektliyini yoxlayan predikatı reallaşdıran, prosedurun adını göstərməklə verilir. Belə prosedurları giriş sənədlərinin mənaca analizinin xüsusi şərtləri adlandırırlar. Bu xüsusi şərtlər «Vilnüs» sistemində nəzərdə tutulmuş qaydalara uyğun olaraq reallaşdırılmalıdırlar və layihələndirilən paketin funksional hissəsinə daxil edilirlər. Onlar ilkin verilənlərin ziddiyyətdə olmadıqlarını yoxlamaq üçün istifadə olunurlar.

Sistem proqramçı nöqteyi nəzərindən verilənlər bazası məntiqi verilənlər toplularının çoxluğu kimi təqdim olunur. Sinfin hər bir modelinə hər hansı bir verilənlər toplusu qarşı qoyulur, obyektin hər hansı bir konkret modelinə bu toplulların hər birində hər hansı bir hissə qarşı qoyulur və bu hissə verilənlər seqmenti adlanır. Modelin bir neçə topluya parçalanması giriş sənədlərinin sayı və strukturu ilə heç də əlaqədar deyildir.

Daimi saxlanılan verilənlər toplularının bir neçə tipini bir-birindən fərqləndirirlə: əsas; paralel; cədvəl; bufer; arxiv. Əsas verilənlər toplusu obyektin uyğun modelinin əsas xassələrini saxlamaq üçün təyin olunmuşlar. Əsas toplu modellər sinfi haqqında məlumatları saxlayan kataloqdan və bu sinfin konkret nüsxələrini şərh edən ayrı-ayrı verilənlər seqmentindən ibarətdir.

Paralel verilənlər toplusu, çıxış sənədlərinin tərtibində istifadə olunan az aktiv informasiyaları saxlamaq üçün istifadə olunur. Əsasən bu müxtəlif mətnlərdir. Paralel topluların yazıları hər hansı bir əsas verilənlər toplusunun yazılarına paraleldir. Əsas toplunun yazılarında təkrarlanan qruplar olduqda, belə qruplar paralel toplunun yazılarında da müəyyən oluna bilər. Əsas və paralel toplular arasındakı uyğunluq xüsusi ünvanlar cədvəli vasitəsi ilə qeydə alınır.

Cədvəl verilənlər topluları modelləşdirilən obyektlər və onlar arasındakı əlaqələr haqqında əlavə informasiyalar verən normati-sorğu verilənlərini saxlamaq üçün təyin olunmuşlar. Əsas toplunun yazılarında cədvəl toplularına istinad identifikatorlar vasitəsi ilə edilir. Cədvəl topluları ünvanlar cədvəli ilə təyin olunurlar. Ünvanlar cədvəli toplunun elementlərini identifikatorlarını və bu elementlərə ünvan istinadlarını saxlayır. Ünvanlar cədvəlində lazım olan

identifikatorun axtarışı metodu paketin verilənlər bazası layihələndirilərkən müəyyən olunur.

Bufer topluları tətbiq sahəsinin müvəqqəti (cari) informasiya modelinin qurulması üçün təyin olunmuşdur. Cari modelin qurulması zərurəti çox variantlı hesablamaların aparıldığı zaman meydana çıxır. Belə hesablamalar aparmaq üçün «Vilnüs» sistemi tətbiq sahəsinin cari informasiya modelinin qurulması vasitələri təklif edilir. Cari model yalnız cari hesablamalar aparılarkən saxlanılır və əsas informasiya modelinə heç bir təsir göstərmir.

Arxiv verilənlər topluları modelləşdirilən obyektlərin və tətbiq sahəsinin proseslərinin tarixini saxlamaq üçün təyin olunmuşlar. «Vilnüs» sistemində proqramm modulları arxiv toplularına bir başa müraciət edə bilmirlər. Bu toplularda saxlanılan verilənlər, hesablamalara qatıla bilmirlər və yalnız paketin tətbiq sahəsinin inkişaf tarixi haqqında arayış vermək üçün istifadə olunurlar.

Yuxarıda baxılan verilənlər topluları məntiqi obyektlərdirlər. Onlar üzərində bütün əməllər «Vilnüs» sisteminin verilənlərin manipulyasiyası dilinin köməyi ilə yerinə yetirilir. Məntiqi toplular fiziki toplulara bu dilin reallaşdırılan proqramlar vasitəsi ilə inikas olunur. Daimi saxlanılan ixtiyari verilənlər toplusuna, bufer topluları istisna olmaqla, «Vilnüs» sistemində hər hansı bir T verilənlər dilində mətnlər toplusu kimi baxılır. T dili həm K_1, K_2,…,K_n, həm də Q dilindən fərqlənir. T_1, T_2,…,T_m (m-verilənlər toplularının sayıdır) dillərinin qramatikası «Faylların şərhi blankının» köməyi ilə müəyyən olunur.

2.6. Tətbiq sahəsinin proqram modeli.

Tətbiq sahəsinin proqram modeli paketin fuksional hissəsi ilə verilir. Bu hissə paketin tətbiq sahəsinin elementar problem – oriyentasiyalı əməl-lərini reallaşdırır. Proqram modelinin qurulması qaydaları kitabxananın strukturu ilə qeydə alınır.

«Vilnüs» sistemində paketin funksional hissəsi üç səviyyəli modul struktura malikdir. Modulların ən aşağı səviyyəsini alt proqramlar və makrotəyinlər təşkil

edir. Onlar makrodil əmələ gətirirlər. Makrodil isə növbəti səviyyə modulları proqramlaşdırmaq üçün istifadə olunur. Bu modullar makrobloklar adlanırlar. Alt proqramlar və makrotəyinlər tətbiq sahəsinin anlayışları ilə müəyyən oluna bilməzlər. Onlar verilənlərin emalı alqoritmlərinin anlayışları ilə müəyyən olunurlar.

Makrobloklar – paketin tətbiq sahəsində elementar hesab olunan problem-oriyentasiyalı əməllərin anlayışları vasitəsilə müəyyən olunan modulardırlar. Onlar paketin tətbiq sahəsinin alqoritmik modelinin reallaşdırması vasitələri hesab olunurlar.

Prossesorlar – bilavasitə tətbiq sahəsinin anlayışları ilə müəyyən olunmuş makroblokların müəyyən qayda ilə təşkil olunmuş toplusudur. Prosessorlar tətbiq sahəsinin funksional modelini reallaşdıran vasitələrdir-lər. İxtiyari prosessorun strukturu alqoritmik modeli şərh edən semantik şəbəkə vasitəsilə müəyyən olunur.

«Vilnüs» sistemində ayrı-ayrı tip funksional modulların strukturunu nizama salan bir sıra standartlar qəbul olunmuşdur. Bu standartlar sistemin konseptual tamlığını, translyasiyanın eyni formalı olmasını ayrı-ayrı modul-ların redaktə və istifadə olunmasını təmin edir. Onlardan bəzilərinə baxaq. İxtiyari alt proqrammı çağırmaq üçün hər hansı bir makroəmr tətbiq olunur. Aşağı səviyyə moduldan istifadə edən problem proqramçısı onun necə reallaşdırıldığını, yəni alt proqram və yaxu makrotəyin şəklində olduğunu bilmir. Bu səviyyənin bütün modulları üçün interfeys standartlaşdırılmışdır.

Funksional modulların əsas tipii makrobloklar hesab olunur. Onlar səkkizə qədər müstəqil yerinə yetmə rejiminə malik ola bilirlər. Onlar həm də elementar problem-orientasiyalı əməllərin yaxud belə əməllərin müxtəlif şəkillərinireallaşdırırlar. Bir makroblokla reallaşdırılan bütün əməllər eyni bir verilənlər strukturu üzərində yerinə yetirilir. Makroblok hər hansı bir prosessorun tərkibində yerinə yetirilir və bilavasitə «Vilnüs» sisteminin monitorunun idarəsi altında işləyir. Eyni bir makroblok prosessorun bir neçə yerində və yaxud müxtəlif prosessorlarda istifadə oluna bilər. Makroblokun yerinə yetirmə rejimləri və çağırılma momenti məsələnin pəlli planı ilə müəyyən olunur. Yerinə yetirmə rejimi

makrobloka parametr qismində ötürülür. Makroblokun verilənlər bazasına müraciət verilənlərin manipulyasiyası dili vasitəsi ilə təyin olunur. Bu dilin vasitələri ilə makroblok verilmiş struktura malik işçi verilənlər toplusunun yaradılması və yaxud, verilənlər bazasında hər hansı bir hesablamanın nəticələrinin saxlamasını tələb edə bilər. Lakin, bu cür fəaliyyətə makrobloka istina hal kimi icazə verilir. Bir qayda olaraq, onlar prosessor səviyyəsində yerinə yetirilir.

«Vilnüs» sistemi ilə qarşılıqlı əlaqə üçün makroblok problem modulların sorğu dilindən istifadə edir. Bu dildən aşağı səviyyənin modulları da istifadə edə bilərlər. Sorğu dilinin köməyi ilə makroblok paketinin qlobal parametrlərinə müraciət imkanı əldəedə bilər. Ayrı-ayrı makrobloklar bir-biri ilə monitor və xarici yaddaşda yerləşdirilən prosessorun işçi topluları vasitəsi ilə əlaqə saxlayırlar. Verilənlərin dəyişdirilməsi əsas yaddaş yaxud reqimtrlər vasitəsi ilə aparıla bilməz. İdarənin bilavasitə bir makroblokdan digərinə ötürülməsinə də icazə verilir. İş başa çatdıqdan sonra, makroblok idarəni sistemin monitoruna qaytarmalıdır. «Vilnüs» sistemində makroblokun strukturuna görə aşağıdakı razılaşmalar qəbul olunmuşdur:

1. Makroblok yalnız bir giriş və bir çıxış nöqtəsinə malikdir.

2. makroblok dinamik strukturlu moduldur. Makrobloku təşkil edən ayrı-ayrı yükləmə modulları makroblokun səhifələri adlanır. Səhifələr makroblokun daxilində sıra ilə nömrələnir.

3. Birinci səhifə monitor tərəfindən çağırılır. Digər səhifələrin çağırılmasını makroblok həyata keçirilir. Əsas yaddaşa eyni zamanda dördə qədər səhifə saxlamağa icazə verilir.

4. Səhifənin həcmi ilkin dilinin 500 operatorunu aşa bilməz.

5. Makroblokun hər bir səhifəsi eyni bir standart prosedurların köməyi ilə avtonom şəkildə translyasiya və redaktə olunur. İdarənin və verilənlərin makroblokun ayrı-ayrı səhifələri arasında ötürülməsi «Vilnüs» sisteminin xüsusi alt proqramları vasitəsi ilə yerinə yetirilir.

6. Səhifələrin verilənlərin və digər proqramm obyektlərinin identifikasiyası «Vilnüs» sisteminin standart razılaşmaları əsasında nizama salınır.

7. Makroblok tətbiqi proqramlar paketlərinin yerinə yetmə rejimlərinə reaksiya göstərir.

8. Makroblok müəyyən iyerarxik struktura malikdir. Bu struktur «Vilnüs» sistemindəqəbul olunmuş, problem modullarının proqramlaşdırılması texnologiyası ilə nizama salınır.

9. Translyasiya zamanı makroblok kompüterin və tətbiqi proqramlar paketinin konfiqurasiyasını müəyyən edən qlobal parametrlərin qiymətlərinə görə sazlanır.

İxtiyari makroblok xarici mühitlə dəqiq təyin olunmuş interfeys vasitəsi ilə qarşılıqlı əlaqədədir. Bu interfeysə aşağıdakılar daxildir:

- makroblokun konfiqurasiyasını müəyyən edən qlobal parametrlər;
- ilkin və nəticə verilənlər topluları;
- problemin xarakteristikalarının daimi cədvəlinin istifadə olunan komponentləri;
- sorğuya görə verilən məlumatların kodu;
- makroblokun verdiyi qadağanedici kodlar;
- makroblokun yerinə yetməsiniqadağan edən kodlar;
- makroblokun yerinə yetmə rejimləri;
- makroblokların istifadə etdiyi tətbiqi proqramlar paketinin yerinə yetmə rejimləri.

İnterfeys makroblokun xassələrini tamalilə müəyyən edir və onun layihələndirilən prosessorun tərkibinə daxil etmək üçün kifayətdir. Makroblokun reallaşdırılması haqqında bu zaman heç bir məlumat tələb olunmur.

Prosessorlar istifadəçinin müəyyən məqsədlərinin reallaşdırılması-na istiqamətlənmiş makroblokların təşkil olunmuş toplusudur. Hər bir prosessor giriş verilənləri toplusuna, çıxış verilənləri toplusuna və yerinə yetmə rejiminə malikdir. Prosessorun strukturu və tərkibi tətbiq sahəsinin alqoritmik modelinin onun proqramm modelinə inikası üsullarını qeydə alan cədvəllər vasitəsi ilə müəyyən olunur.

Prosessorun tərkibinə makrobloklar və altproqramlar daxildirlər. Onlar makroblokun konkret yerinə yetmə anında prosessorun giriş və çıxış verilənlərinin seçilməsi üsulunu müəyyən edirlər. Prosessorun idarəedici hissəsi və yaxud digər bir xüsusi proqram komponenti yoxdur. Prosessorun bir neçə giriş nöqtəsi, həm də bir neçə çıxış nöqtəsi ola bilər. Prosessor konkret yerinə yetirdiyi anda bu nöqtələrdən hansı birinin seçiləcəyi məsələsinin qoyuluşu ilə şərtləndirilir. Hər bir giriş nöqtəsi ilə hər hansı bir giriş verilənləri toplusu əlaqələndirilir, uyğun olaraq, hər bir çıxış nöqtəsi ilə də hər hansı bir çıxış verilənləri toplusu əlaqələndirilir. Bu toplular verilənlərin manipulyasiyası dilinin köməyi ilə formalaşdırılır. Topluların formalaşdırılması üçün sorğunu «Vilnüs» sisteminin monitoru verir. Prosessor yerinə yetdikdən sonra monitor hesablamaların nəticəsinin verilənlər bazasına əlavə olunması üçün sorğu verir. Giriş, həm də çıxış nöqtəsi xüsusi altproqramlarla müəyyən olunur. Ayrı-ayrı prosessorlar öz aralarında bilavasitə heç bir əlaqəyə malik deyillər. Onlar yalnız verilənlər bazası vasitəsi ilə əlaqə saxlayırlar.

Prosessorun yerinə yetməsinin üç rejimi nəzərdə tutulmuşdur: ardıcıl, paralel, qarışıq.prosessorun ardıcıl yerinə yetməsi zamanı verilənlər seqmentləri ardıcıl yerinə yetirilirlər. Paralel rejimdə isə verilənlər seqmentləriprosessorun bir yerinə yetməsi zamanı paralel icra olunurlar. Qarışıq rejim hər iki əvvəlki rejimin xüsusiyyətlərini nəzərə alır. Prosessor qarışıq rejimdə yerinə yetən zaman verilənlərin seqmentlərə bölünməsi qadağan edilir və prosessor öz verilənləri toplusu üzərində bir dəfə yerinə yetirilir. Nəticələr verilənlə bazasına əlavə olunarkən yenə seqment-ləşdirilirlər.

Tətbiq sahəsinin proqramm modelinin qurulması vasitələri bütünlükdə paketin yerinə yetməsi rejimlərini nəzərdə tutur.

FƏSİL III. «VILNÜS» İNSTRUMENTAL PROQRAMLAŞDIRMA SİSTEMİNİN TƏRKİBİ VƏ ƏSAS FUNKSİYALARI

3.1 «Vilnüs» sisteminin əsas komponentləri

3.1.1 Generasiya vasitələri

«Vilnüs» sisteminin əsas komponentləri aşağıdakılardır: generasiya vasitələri; konstruktor; idarəedici proqram; verilənlərin translyasiyası vasitələri. Bu komponentlər sistemin baza nüvəsini əmələ gətirirlər.

Generasiya vasitələri. Proqram sistemlərinin tətbiq olunması, bir qayda olaraq, kifayət qədər çətin hazırlıq işlərinin görülməsini tələb edir. Verilmiş istifadəçi üçün sistem kompüterdə generasiya olunmazadn əvvəl müəyyən hazırlıq işləri görülməlidir. Bu aşağıdakı tələblərlə şərtləndirilir:

- sistemin konkret kompüterin və istifadəçinin konkret məsələlərinin parametrlərinə sazlanması;

- yaddaşın sistem verilənlər toplularına görə bölüşdürülməsi;

- sistemin nüvəsini əmələ gətirən standart vasitələr toplusu ilə yana-şı istifadəçinin arzusuna görə, sistemə daxil ediləcək əlavə vasitələrin seçilməsi və komplektləşdirilməsi.

Bəzi sistemlər üçün onun işə salınması zamanı hazırlıq işləri ta-mamilə sistem proqramçısının üzərinə düşür. Əksər hallarda bu sistem-lərin, konkret istifadəçi üçün sazlanarkən, əlavə vasitələrə ehtiyacı qlamır. Onlar funksional nüvədən ibarət olurlar. Mümkündür ki, nüvə bir neçə variantda proqramlaşdırılsın.

Nisbətən mürəkkəb sistemlər üçün hazırlıq işləri, onların sisteminin uyğun generasiya vasitələri ilə sazlanmasını tələb edir. Generasiya vasi-tələrinin əsas funksiyası sistem kitabxanalarının formalaşdırılmasıdır. Ge-nerasiya iki mərhələdə yerinə yetirilir. Birinci mərhələdə sistem proqramçı-sının göstərişlərinə əsasən sistemin modullarının translyasiyası, redaktə olunması və kitabxanalara yazılması üçün tapşırıqlar axını formalaşdırılır. Tapşırığın formalaşdırılması üçün Assambler dilini makrovasitələrindən istifadə olunur. İkinci mərhələdə tapşırıq yerinə yetirilir.

Generasiya olunan sistemin parametrləri adətən hər hansı bir makrodildə şərh olunur. Bu metod bir çox sistemlərin generasiyası zamanı tətbiq olunur.

«Vilnüs» sisteminin generasiya vasitələrinin əsasını elə yuxarıda şərh olunan metod təşkil edir. Onun əsas fərqli cəhətləri isə aşağıdakı-lardır:

- generasiya olunan sistemin parametrlərinin şərhi üçün cədvəl dilindən istifadə olunması;

- yaddaşın bölüşdürülməsi, kitabxanaların surətlərinin çıxarılması və digər hazırlıq işləri üçün tapşırığın hazırlanmasının avtomatlaşdırılması;

- əlavə vasitələrin seçilməsi üçün dialoq rejiminin olması.

«Vilnüs» sisteminin generasiya olunan variantının şərhi üçün gene-rasiya blankından istifadə olunur. Blank üç hissədən ibarətdir. Bu hissə-lərdə kompüterin konfiqurasiyası, generasiya olunan sistemin konfiqurasi-yası, tətbiq sahəsinin problem parametrləri şərh olunurlar. Bundan başqa blankda generasiya olunmuş sistemə müraciət etmək üçün gələcəkdə istifadə olunacaq parollar da verilir.

«Vilnüs» sistemi Assambler dilinin vasitələri ilə reallaşdırılmışdır. Nüvənin bütün moduları, mümkün yaddaşın həcmini, verilənlərin daxil edilməsi üsulunu, qurğuların tipini və hesablama sisteminin digər xüsusiyyətlərini müəyyən edən qlobal dəyişənlərə bağlanmışdır. Qlobal parametrlərə qiymətlər, generasiya blankı əsasında formalaşdırılan xüsusi makroəmrlər vasitəsi ilə mənimsədilir. Qlobal dəyişənlərin istifadə olunması modulların spesifikasiyaları ilə nizama salınır.

Nüvədən başqa, sistemin ilkin variantı, «Vilnüs» sisteminin köməyi ilə yaradılmış bir neçə baza tətbiqi proqramlar paketindən ibarət olur. Elə bu da, istifadəçinin arzusu ilə sistemin konkret generasiyasına qoşula bilən əlavə vasitələrdirlər. Bu vasitələrin verilməsi üçün dialoq rejimindən istifadə olunur. Sistemin generasiyası iki mərhələdə yerinə yetirilir. Birinci mərhələdə generasiya blankı analiz olunur və generasiya olunan sistemin şərhi makrodildə formalaşdırılır. Makrodil aralıq dil kimi istifadə olunur. Bu şərh Assembler tərəfindən emal olunur və aşağıdakılar üçün tapşırıq axını formalaşdırılır:

- «Vilnüs» sisteminin rezident diskinin inisalizasiyası;

- sistem verilənlər toplularının katoqlaşdırılması;

- prosedurlar kitabxanasının və ilkin mətnlərin kodlaşdırılması;

- sistemin translyasiyası və redaktə olunması;

- əlavə vasitələrin qoşulması.

Formalaşdırılan tapşırıqların sayı və strukturu, əsasən, generasiya olunan sistemin konfiqurasiyasını şərh edən generasiya blankının parametrləri vasitəsi ilə müəyyən olunur. Sistemin generasiyasının birinci mərhələsinin reallaşdırılması aşağıdakıardıcıllıqlı həyata keçirilir:

- generasiya blankı daxil edilir və analiz olunur;

- generasiya olunan sistemin şərhi makrodilə translyasiya olunur və translyasiyanın nəticəsi işçi maqnit diskinə yazılır;

- makroassamblerin köməyi ilə tapşırıqlar axını formalaşdırılır və distrubutiv maqnit diskinə yazılır;

- qlobal dəyişənlərə qiymətlərin mənimsədilməsi makrotəyinləri formalaşdırılır və distrubutiv maqnit diskinə yazılır;

- istifadəçidən sistemə qoşulacaq əlavə vasitələr toplusu haqqında məlumat soruşulur;

- distributiv maqnit diskinə ilkin modulların mətnləri köçürülür.

İkinci mərhələdə distrubutiv maqnit diskindən sistemin konkret variantı formalaşdırılır. Nəticədə sistemin rezidenti, sistem cədvəllərini saxlamaq üçün sistem verilənlər topluları və verilənlər bazası yaranır. Sistemin generasiyasının ikinci mərhələsinin reallaşdırılması aşağıdakı ardıcıllıqla həyata keçirilir:

- diskləri inisalizasiya edən, kitabxanaları köçürən, (surətini çıxaran) tapşırıqlar və digər hazırlıq işləri yerinə yetirilir;

- ilkin mətnlər translyasiya olunur və yükləmə modulları kitabxanası formalaşdırılır;

- sistemin verilənlər topluları yaradılır;

- generasiya olunmuş sistemin işi misal vasitəsi ilə nümayiş etdirilir.

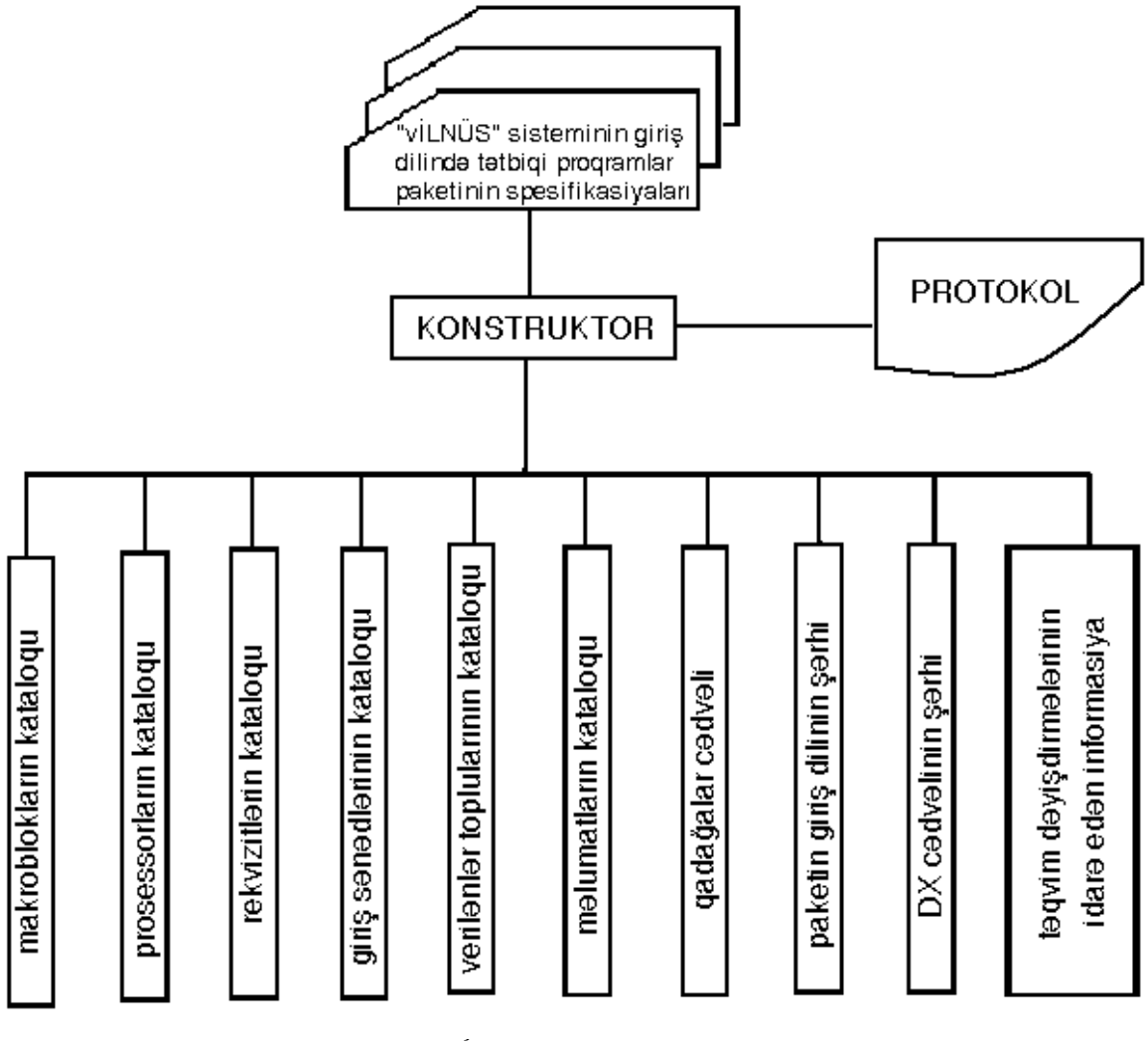
3.1.2 Konstruktor.

Konstruktorun əsas vəzifəsi «Vilnüs» sistemin giriş dilində yaradılacaq paketin şərhinə əsasən sistemin cədvəllərinin qurulmasıdır. Bu cədvəllər idarəedici prqramın və verilənlərin translyasiyası proqramının işini idarə edir. Bundan başqa, konstruktora paketin şərhinin düzgünlüyünün yoxlanması tapşırığı də verilmişdir. Bu məqsədlə ilkin mətnin giriş dilinin sintaksisinə uyğunluğunun və qurulmuş sistem cədvəllərinin qarşılıqlı uyğunluğunun yoxlanılması zəruridir, başqa sözlə, dil, alqoritmik, informasiya vəproqram modellərinin düzgünlüyü şərtlərinin yoxlanılması zəruridir. Konstruktorun işi sxemi şəkil 5-də verilmişdir.

Konstruktor tətbiqi proqramlar paketinin işini idarə edən sistem cədvəllərini yaradır. Bu cədvəlləri ətraflı nəzərdən keçirək.

Makroblokların kataloqu. Makroblokların - paketinin əsas funksional modullarının kataloqlaşdırılması üçün təyin olunmuşdur. Kataloq tətbiq sahəsinin proqramm modelinin şərhidir və həm istifadəçinin məsələsinin həlli planının interpretasiyası zamanı, həm də soraq kitabçası kimi istifadə olunur.

Prosessorların kataloqu. Prosessorların - yuxarı səviyyənin funksional modullarının kataloqlaşdırılması üçün təyin olunmuşdur. Hər bir prosessorun şərhi bir neçə cədvəl şəklində saxlanılır. Bu cədvəllər tətbiq sahəsinin alqoritmik modelinin şərhini, bu modelin proqram modelinə inikası üsulunu, prosessorun giriş/çıxış verilənlərini və onun digər xarakteristikalarını əks etdirirlər. Kataloq istifadəçinin məsələsinin həlli planının formallaşdırılmasızamanı istifadə olunur.



İlkin sənədlərin, rekvizitlərin və verilənlər toplularının kataloqları tətbiq sahəsinin informasiya modelinin şərhinin saxlanılması üçün təyin olunmuşlar. Onlar verilənlər bazasının strukturunu, ilkin sənədlərin daxil edilməsi şablonunu, hər hansı bir prosessor üçün giriş və yaxud çıxış işçi verilənləri toplularının strukturunu müəyyən edirlər. Bundan başqa, verilənlər topluları kataloqunda çıxış sənədlərinin şərhləri kataloqlaşdırılırlar. Çıxış sənədlərinin şərhi çıxış generatoru vasitəsilə generasiya olunur. Hər üç kataloq verilənlərin translyasiyası vasitələri tərəfindən istifadə olunurlar.

Məlumatların kataloqu paketininproblem və sistem modullarının verdiyi məlumatların mətnlərininsaxlanması üçün təyin olunmuşdur. Məsələnin həlli protokolunun aparılması üçün sistemin monitoru tərəfindən istifadə olunur. Bundan başqa, kataloqda seqmentin vəziyyətləri vektorun zəruri və təcridedici

komponentlərihaqqında informasiya saxlanılır. Bu informasiyalar makroblokların kataloqu əsasında formalaşır və monitor funksional modulları çağırarkən istifadə olunur.

Qadağalar cədvəli - məsələnin həlli planında düzəlişlərin aparılması tələb olunan vəziyyətləri yadda saxlamaq üçün təyin olunmuşdur. Hər bir vəziyyət müəyyən şifrlə identifikasiya olunur və qadağan edici kod adlanır. Hər bir qadağan edici kodla elementar problem-orientasiyalı əməllər əlaqələndirilir və vəziyyətlər əmələ gəldikdə bu əməllərin yerinə yetməsi qeyri mümkün və yaxud mənasız olur. Qadağan edici kodlar aralıq hesablamaların nəticələrini analiz edən problem modulları tərəfindən verilirlər. Cədvəl verilmin qadağan edici koda əsasən məsələnin cari həlli planının dəyişdirilməsi, yəni modifikasiya olunması üçün istifadə olunur.

Daimi xarakteristikalar cədvəlinin şərhi bu cədvəlin strukturunu müəyyən etmək üçün istifadə olunur. Həm də DX cədvəli məsələnin həlli prosesininprotokollaşdırılması üçün istifadə edilir.

Tətbiqi proqramlar paketinin giriş dilinin şərhi tətbiq sahəsinin dil modelini şərh edən cədvəlləri saxlayır. Bu şərh həm də dil modelinin funksional modelə, funksional modelin alqoritmik modelə inikası üsullarını əks etdirən cədvəlləri saxlayır.

Təqvim dəyişdirmələrini idarə edən informasiya. Burada zaman intervalının ölçü vahidləri, bayramlar, işçi şənbə günləri haqqında məlumatlar saxlanılır. Bu informasiya «КАЛЕНДАРЬ» alt proqramının işini idarə etmək üçün istifadə olunur. «КАЛЕНДАРЬ» alt proqrammı şəbəkənin hesablanması, planın formalaşdırılması və uyğun digər məsələlər üçün istifadə edilir. Əgər layihələndirilən paketdə «КАЛЕНДАРЬ» alt proqramından istifadə olunmursa, onda idarəedici informasiya lazım gəlmir.

Giriş dilinin hər bir blankı konstruktor üçün ayrıca tapşırıq kimi tərtib olunur. Bu isə paketi addımlarla layihələndirməyə imkan verir. Bundan başqa konstruktorun bu xassəsindən paket istismar olunan zaman onun modifikasiyası və genişləndirilməsi üçün istifadə olunur.

Konstruktorun bütün fəaliyyəti protokollaşdırılır. Protokolda tapşırıqların mətni, səhvlərin diaqnostikası və konstruktorun yerinə yetirdiyi işlərin şərhi göstərilir (qeydə alınır). Protokol sistemin monitoru tərəfindən aparılır. Konstruktor monitorun idarəsi altında işləyir. Paket layihələndirilən zaman konstruktor özü paketin sistem hissəsinə qoşulur ki, bu da paketin dəyişən ətraf mühütin tələblərinə uyğunlaşmasına imkan verir.

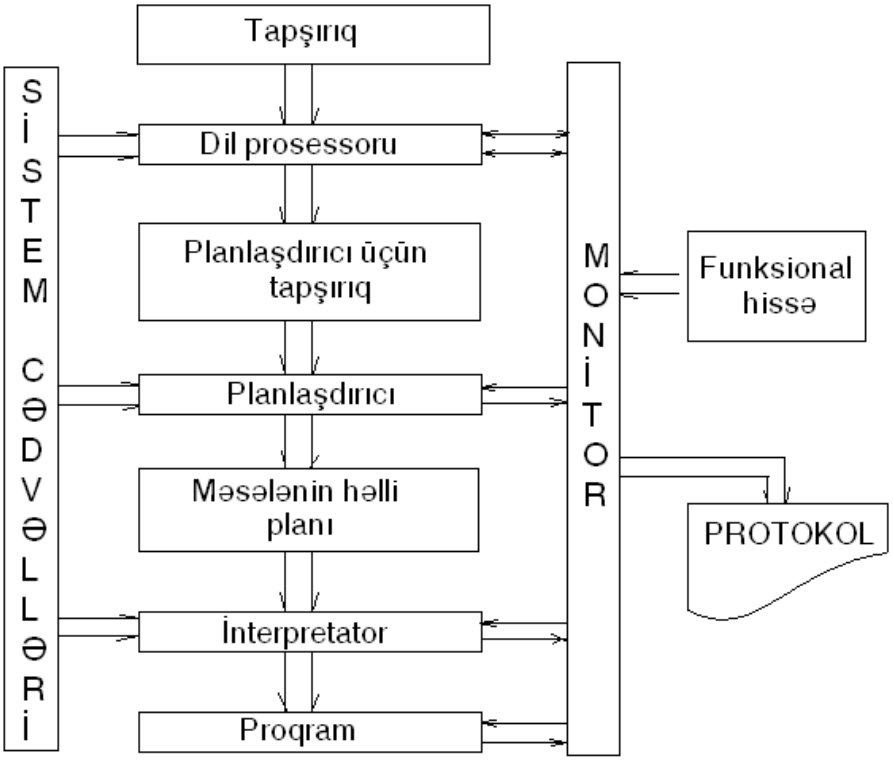
3.2 İdarəedici proqram və onun əsas komponentləri

İdarəedici proqram «Vilnüs» sisteminin nüvəsinin əsas komponenti hesab olunur və istifadəçinin tapşırığının translyasiyasını yerinə yetirir, həm də translyasiya nəticəsində alınan proqramın yerinə yetməsini idarə edir. İdarəedici proqramın tərkibinə aşağıdakılar aiddir:

- dil prosessoru;
- planlaşdırıcı;
- məsələnin həlli planının interpretatoru;
- monitor.

Bu komponentləri ətraflı nəzərdən keçirək. Onların qarşılıqlı fəaliyyət sxemi aşağıdakı şəkil 3.3-də göstərilmişdir.

Dil prosessoru, yaxud, onu məsələlər translyatoru da adlandırırlar, tətbiqi proqramlar paketinin giriş dilində tərtib olunmuş məsələlərin şərhinin translyasiyası üçün təyin olunmuşdur.



Şəkil 3.3

«Vilnüs» sistemində bir cür aralıq dil rolunu tətbiq sahəsinin funksional modelini şərh edən VƏ / VƏ YAXUD qrafı oynayır. Dil prosessoru «Vilnüs» sistemində tətbiq sahəsinin dil modelinin onun funksional modelinə inikası vasitəsidir. Dil prosessorunun işinin nəticəsində paketin giriş dilində tapşırığın semantikasının şərhi kimi baxmaq olar. Nəzərdə tutulur ki, əgər məqsədlər VƏ / VƏ YAXUD qrafının anlayışları vasitəsi ilə aşkar şəkildə formalaşdırılıbsa, tapşırığın semantikası təmamilə müəyyən olunur. Dil prosessorunun reallaşdırılması metodikasının əsasını aşağıdakı kimi şərh etmək olar. VƏ / VƏ YAXUD qrafında tətbiq sahəsinin funksional modelini müəyyən edən VƏ YAXUD təpə nöqtələrindən çıxan hər bir tilə, aşağı səviyyənin uyğun alt məqsədinin seçilməsi kriteriyasını müəyyən edən bir proedikat qarşı qoyulur. Proedikatın tipii və onun arqumentləri paketin giriş dilinin xüsusiyyətləri ilə müəyyən olunur və problemlərin şərhi blankının köməyi ilə şərh olunur. Proedikatlar etalon qiymətlərlə istifadəçinin tapşırığının mətnindən formalaşdırılan

qiymətlər arasında verilmiş münasibətlərdən qurulur. Münasibətlər $=, \neq, \rangle, \langle, \in$ və sair tipli ola bilərlər. Predikatların aldığı qiymətlərin əsasında, altağacların kök təpə nöqtəsi olan, yalnız VƏ tipli təpə nöqtələri saxlayan yaxud aralıq hesablaşmaların nəticəsindən asılı olan, dinamik seçilən, alternativləri müəyyən edən VƏ YAXUD tipli təpə nöqtələrindən ibarət, terminal təpə nöqtələri seçilir. Yuxarıda şərh olunan, dil prosessorunun reallaşdırılması metodikası, təkcə cədvəl tipli dillər üçün deyil, həm də digər əmr tipli dillər üçün də tətbiq oluna bilər. Bu o hallarda mümkündür ki, aralıq dil kimi VƏ / VƏ YAXUD qraflarının köməyi ilə formalaşdırılmış tətbiq sahəsinin funksional modeli istifadə olunsun. Xüsusi halda bü metodika direktiv dillər və «menyu» tipli dillər üçün də tətbiq oluna bilər.

«Vilnüs» sisteminin dil prosessorunda VƏ / VƏ YA qrafının analizi üçün həll cədvəlləri aparatında istifadə olunur. Həll cədvəli dörd hissədən ibarətdir. Birinci hissədə şərtlər sadalanır, başqa sözlə

$$C = \{c_1, c_2, ..., c_n\},$$

məqsəd ağacında alternativ məqsədin seçilməsi kriteriyasını müəyyən edən predikartlar sadalanır. Burada m –predikatların ümumi sayıdır. Cədvəlin ikinci hissəsində fəaliyyətlər (işlər) çoxluğu göstərilir, başqa sözlə

$$A = \{a_1, a_2, ..., a_к\},$$

aşağı səviyyə məqsədlərinə uyğün olan VƏ / VƏ YA qrafının təpə nöqtələri sadalanır; burada k- istifadəçinin məqsədlərinin ümumi sayıdır. Cədvəlin üçuncü və dördüncü hissələrində siste-min hansı rejimdə, dialoq yaxud paket rejimində işlədiyi və predikatların qiymətlərinin müxtəlif kombinasiyalarına uyğun VƏ / VƏ YA qrafının təpə nöqtələrinin adları göstərilir. Həll cədvəli apparatı cədvəl dilləri haqqında həm də ilkin mətnin sintaktik analizi üçün tətbiq oluna bilir. Bu məqsədlə giriş dilinin cədvə-linin hər bir sütununa predikat qarşı qoyulur. Bu predikat uyğun sütuna mümkün qiymətlər çoxluğunda «doğru» yaxud «yalan» qiymətlərindən birini ala bilər. Uyğun qayda ilə həll cədvəli ge-nişləndirilir. Birinci hissəyə yuxarıda xatırlanan predikatlar, dördüncü hissəyə ilkin mətndə səhvlər aşkar olunduqda verilən məlumatlarının kodu əlavə olunur.

İstifadəçinin məqsədlərinin realaşdırılması üsulunu dəqiqləşdirən məhdudiyyətlər bir qrup parametrlərlə müəyyən olunur. Bu parametrlər giriş dili səviyyəsinə çıxarıla bilər və yaxud dil prosessoru vasitəsilə susmaya görə formalaşdırıla bi-lər. Məhdudiyyətləri müəyyən edən parametrlər hər iki halda informasiya kartı şəklində təqdim olunurlar. Hər bir informasiya kartı 80 –nə qədər simvol saxlaya bilən sətir tipli dəyişəndir. Kartlar sıra nömrəsi ilə nömrələnirlər. Hər bir kartın strukturu paketin layihə sənədlərində qeydə alınır və problem modulları tərəfindən istifadə olunur. Sistemdə informasiya kartlarını onların nömrələrinə görə çağırmaq üçün vasitələr nəzərdə tutulmuşdur. Əslində informasiya kartları mexanizmi istifadə-çinin verdiyi parametrlərin problem modullarına ötürülməsi mexanizmidir. Bu mexanizm COMMON tipli sahə topluları şəklində realaşdırılır.

İnformasiya kartlarının formalaşdırılması üsulunu prob-lemin şərhi blankı ilə müəyyən olunur. Kartlar istifadəçinin tapşırığından seçilən parametrlərdən formalaşdırılır.

Dill prossesorunun işi aşağıdakı sxem vasitəsilə şərh oluna bilər. İlkin mətnin növbəti simvolu nəzərdən keçirilir. Bu simvol leksik analizator vasitəsilə hər hansı bir dəyişənin qiymətinə çevrilir. Bu qiymət verilmiş predikatın hesablanmasın-da istifadə olunur, yaxud informasiya kartının formalaşdırılmasında istifadə olunur. Bütün mətn nəzərdən keçirildikdən sonra verilmiş predikatların qiymətləri hesablanır və həll cədvəllərinə görə ilkin mətnin sintaktik analizi yerinə yetirilir. Qeydə alınan səhvlərin kodları monitora verilir. Səhv olmadıqda həll cədvəlinə görə, elementləri istifadəçinin məqsədlərinə uyğun olan, cədvəlin ikinci hissəsindən sadalanan, bul vektoru formalaşdırılır. Bu vektor planlaşdırıcıya tapşırıq adlanır. Sonra şərhlərə görə infor-masiya kartları formalaşdırılır.

Planlaşdırıcı dil prosessorunun formalaşdırdığı planlaşdırıcıya tapşırığı məsələnin həlli planına çevirmək üçün təyin olunmuşdur. Plan məsələnin həlli zamanı istifadə olunan verilənləri müəyyən edir, həm də plan bu verilənlər üzərində yerinə yetirilən, paketin tətbiq sahəsində elementar hesab olunan problem-oriyentasiyalı əmrlərin ardıcıllığını müəyyənləşdirir. Planlaşdırıcı

«Vilnüs» sistemində tətbiq sahəsinin funksional modelinin, semantik şəbəkə aparatının köməyi ilə müəyyən olunmuş, onun alqoritmik modelinə inikası vasitəsidir.

Məqsədlər ağacı ilə verilən və planlaşdırıcıya tapşırıqda sadalanan, istifadəçinin məqsədləri paketə nəzərən xarici fak-tordurlar və paketin daxili məqsədlərinə çevrilmək zərurətindədirlər. Daxili məqsədlər dedikdə semantik şəbəkənin əməllərə uyğun məqsəd təpə nöqtələri nəzərdə tutulur. Bu əməllərin yerinə yetirilməsi nəticəsində istifadəçinin məqsədlərinin real-laşmasını təmin edən verilənlər topluları formalaşdırılır. Bir verilənlər toplusu bir neçə məqsədin reallaşmasını təmin edə bilər və əksinə, hər hansı bir məqsədin reallaşması üçün bir neçə verilənlər toplusu tələb oluna bilər. Planlaşdırıcıya tapşı-rığın paketin daxili məqsədlərinə çevrilməsi üçün həll cədvəlləri aparatı tətbiq olunur. Bu halda şərt kimi, tapşırığı şərh edən, bul vektorunun komponentləri istifadə olunur. Yerinə yetiriləcək əməllər kimi isə semantik şəbəkənin məqsəd təpə nöqtələri istifadə edilir. Hər bir təpə nöqtəsi üçün semantik şəbəkənin identifikatoru verilir. Bu identifikator hər hansı bir prosessorun alqoritmik modelini və bu şəbəkədə təpə nöqtəsinin kodunu şərh edir. Şəbəkələr uyğun prosessorların adları ilə identifikasiya olunurlar.

Bütün R təpə nöqtələri çoxluğu R_1, R_2, …,R_n alt çox-luqlarına bölünür, burada n- prosessorların sayıdır, belə ki, hər bir alt çoxluğa bir şəbəkənin təpə nöqtələri aiddirlər. Bu alt çoxluqların bəziləri boş çoxluq da ola bilərlər. Hər bir semantik şəbəkədən R_i-yə görə G_i (V_i, L_i), i = 1,2,…,n alt şəbəkəsi ayrılır.

G={G_1 (V_1, L_1),…,G_n(V_n, L_n)} altşəbəkələr çoxluğu məsələnin həlli planını qurmaq üçün əsasdır.

Məsələnin həlli planı aşağıdakı şəkildə formalaşdırılır: başlanğıc markeri; prosessorun başlanğıcı markeri; prosessorun adı; prosessorun yerinə yetmə rejimi; prosessorun reallaşdırılması planı; prosessorun sonu markeri; prosessorun başlanğıcı markeri; prosessorun adı;…;prosessorun sonu markeri; məsələnin həlli planının sonu markeri.

Prosesorun reallaşdırılması planının strukturu prosessorun rejimindən asılıdır. Prosessorlar üç rejimdə işləyə bilirlər ardıcıl, paralel və qarışıq. Prosessor ardıcıl rejimdə yerinə yetirərkən, bütün sadalanan verilənlər seqmentləri ardıcıl emal olunurlar, başqa sözlə, əvvəlcə bütün əməllər ardıcıllığı birinci seqmentdə, sonra ikinci seqmentdə və i. yerinə yetirilir. Prosessor paralel rejimdə yerinə yetirərkən sadalanan verilənlər seqmentləri paralel yerinə yetirilirlər,başqa sözlə, bütün seqmentlərin birinci əməli, sonra bütün seqmentlərinikinci əməli və i. yerinə yetirilirlər. Prosessor qarışıq rejimdə yerinə yetərkən bütün verilənlər seqmentləri birgə emal olunurlar.

3.3 Məsələnin həlli planının interpretatoru və monitor

Məsələnin həlli planının interpretatoru planının funksional modullarının anlayışlarında interpretasiya üçün təyin olunmuşdur. Bu tətbiq sahəsinin alqoritmik modelinin onun proqram modelinə inikası vasitəsidir. İnterpretasiya metodu iki səbəbə görə seçilmişdir. Əvvələ, «Vilnüs» sisteminin istiqamətləndiril-diyi tətbiq sahələrində problem modullar kifayət qədər böyükdürlər və hətta onların həcmi bir neçə min maşın əmri qədərdir (bu modullar öz növbəsində modul struktura malik olmaqla daha kiçik modullardan ibarətdir). Məsələnin həlli planı yüzə qədər belə modulları əhatə edə bilər. Hər bir yeni hesablama zamanı dəyişə bilən belə plana əsasən kompilyasiya metodu ilə proqramın yaradılması əlbəttə məqsədə uyğun deyildir. İkincisi isə, sistemdə bir sıra daxili dillərdən (verilənlərin manipulyasiyası dilindən, problem modulların sorğu dilindən və s.) istifadə olunur. Bu dillərin reallaşdırılması üçün isə hər hansı bir rezident sistem modulu zəruridir. Belə modulun işini məsələnin həlli planını reallaşdırmaq üçün interpretasiya metodundan istifadə etməklə təşkil etmək daha sadədir.

İnterpretatorun işi aşağıdakı sxemlə şərh oluna bilər. Məsələnin həlli planından növbəti simvol oxunur. İnterpretatorun vəziyyətindən və oxunan simvolun tipindən asılı olaraq monitor üçün müəyyən bir tapşırıq formalaşdırılır. Belə tapşırıq makroblokun çağırılması, prosessorun çağırılması, verilənlər

seqmentinin çağırılması və s. ola bilər. Makroblokun çağırılması üçün tapşırıq hazırlanarkən makroblokun adı və onun yerinə yetmə rejimi uyğun sistem cədvəli üzrə hər hansı bir semantik şəbəkədə təpə nöqtələrinin kodlarından formalaşdırılır. Şəbəkənin təpə nöqtələri ilə makrobloklar arasında uyğunluq prosessorların şərhi blankının köməyi ilə müəyyən olunur. Tapşırıq yerinə yetdikdən sonra monitor yenidən interpretatoru çağırır və dövr yenidən təkrarlanır. Proses, monitor «Hesablama prosesini sona çatdırmalı» tapşırığı alana qədər davam etdirilir.

«Vilnüs» sisteminin monitoru idarəedici proqramın digər komponentlərini idarə etmək üçün təyin olunmuşdur. Bundan başqa «Vilnüs» sisteminin konstruktoru da monitorun idarəsi altında işləyir. Monitorun əsas funksiyaları aşağıdakılardır:

- tapşırıqların tanınması;
- istifadəçinin səlahiyyətinin yoxlanması;
- problem modullarının çağırılması və bu modullara olan tələblərin reallaşdırılması;
- hesablama prosesinin kəsilməsi və bərpa olunması;
- məsələnin həlli protokolunun aparılması.

Müəyyən işlərin yerinə yetirilməsi üçün istifadəçiyə tapşırıq hesablamaya tapşırıq blankının (HTB) köməyi ilə təqdim olunur. Sistem cədvəllərinin qurulması yaxud modifikasiya olunması üçün konstruktora tapşırıq «Vilnüs» sisteminin giriş dilinin blankları vasitəsilə təqdim edilir. Bundan başqa sistemə problem proqramçıları tərəfindən yolanılmaq üçün problem modulları daxil edilə bilər. Monitor tapşırığın tipini «tanıyır» və onun reallaşdırılmasını təşkil edir.

İcazəsiz müraciətlərdən qorunma parollar sistemi vasitəsi ilə həyata keçirilir. Hər bir makroblok üçün, onun yerinə yetməsinə icizə verən, ona qədər parol verilə bilər. Məsələnin həlli planını reallaşdırarkən, monitor, istifadəçinin daxil etdiyi parolla, bu plana daxil edilmiş makrobloklardan istifadə etməklə hesablamaya icazə verən parollar arasındakı uyğunluğu yoxlayır. Əgər istifadəçinin səlahiyyəti qoyulmuş məsələni həll etməyə ona icazə vermirsə, onda sistem məsələnin yerinə yetməsinə imkan vermir. Parol kimi təşkilatı bölmələrin yaxud vəzifəli şəxslərin

kodları istifadə oluna bilər. Bütün sistem cədvəlləri də parollarla müdafiə olunurlar.

Məsələnin həlli planı reallaşdırılarkən, monitor, hesablanan verilənlər seqmentləri haqqında informasiyanı yaddaşa yükləyir, tələb olunan prosessorları və makroblokları çağırır və idarəni onlara verir. İş sona yetdikdən sonra hər bir makroblok idarəni monitora qaytarır.

İxtiyari prossesor yerinə yetməzdən əvvəl monitor prosessorun ilkin verilənlərinin formalaşdırılması üçün «Vilnüs» sisteminin Verilənlər Bazasının İdarə olunması Sisteminə sifariş verir, prosessorun işi sonna çatdıqdan sonra isə hesablamaların nəticələri verilənlər bazasında yadda saxlanılır. İlkin verilənlərin və nəticələrin tərkibi, məsələnin həlli planı ilə reallaşdırılan alqoritmdən və hesablanan verilənlər seqmentinin vəziyyətindən asılıdır. Hər hansı bir alqoritm yerinə yetərkən müəyyən qrup verilənlər tələb olunur və müəyyən nəticələr alınır, başqa bir alqoritmin icrası zamanı isə digər bir qrup verilənlər tələb olunur və təmamilə yeni nəticələr alınır. Bundan başqa aralıq nəticələrdən bəziləri artıq digər bir alqoritmin reallaşdırılması zamanı hesablana və verilənlər bazasında saxlanıla bilər. Belə olan halda tələb olunan alqoritmin icrası əvvəldən deyil, yəni qrafın kök təpə nöqtəsindən deyil, hər hansı bir aralıq təpə nöqtəsin-dən başlaya bilər, və buna uyğun olaraq ilkin verilənlər deyil, müəyyən aralıq verilənlər tələb oluna bilər. Verilmiş prosessor üçün potensial ilkin yaxud nəticə informasiyası ola biləcək bütün verilənlər topluları prosessor layihələndirərkən müəyyən olunurlar və prosessorun şərhi blankında şərh edilirlər. Hər bir belə toplu ilə müəyyən altproqram əlaqələndirilir və bu alt proqram verilmiş verilənlər toplusunun zəruri olduğunu myəyyən edir. Monitor növbə ilə bu alt proqramları çağırır. Onlar prosessorun xüsusi şərtləri adlanırlar.

Monitor problem modullarla qarşılıqlı fəaliyyəti xüsusi sorğu dili vasitəsi ilə təmin olunur. Monitorun reallaşdıracağı sorğular bu dildə formalaşdırılırlar. Problem modulların sorğular dilinin bütün operatorları aşağıdakı formada verilirlər:

PP$\prec$ operatorun nömrəsi $\succ \cup \prec$ operatorun identifikatoru $\succ$,

≺ operandların siyahısı ≻.

Bu dil Assembler dilinin makrogenişlənməsi şəklində realizə olunmuşdur.

Monitorun bütün fəaliyyəti protokollaşdırılır. Məsələnin həlli protokoluna aşağıdakılar daxildir: istifadəçinin tapşırığı; məsələnin həlli planı; səhvlərin diaqnostikası; məsələnin həlli reallaşdırarkən baş verən əsas vəziyyətlərin şərhi.

NƏTİCƏ VƏ TƏKLİFLƏR

İxtiyari problem-oriyentasiyalı proqramlaşdırma sisteminin, o cümlədən tətbiqi proqramlar paketinin, imkanları bu sistemin giriş dilinin semantikası ilə müəyyən olunur.

Tətbiqi proqramlar paketləri istifadəçinin tətbiqi fəaliyyətinin modelləşdirilməsi vasitəsi hesab olunur. Giriş dilinin semantikası onlar üçün tətbiq sahəsinin modelləşdirilməsi yolu ilə müəyyən oluna bilər, belə ki, modelləşdirmə müxtəlif səviyyələrdə – informasiya, funksional, alqoritmik, proqramm səviyyələrində aparıla bilər. Giriş dilinin tipi bu səviyyələrdən hansının istifadə olunacağının müəyyən edir.

Giriş dilinin qeyri – prosedur olması tətbiq sahəsinin alqoritmik və proqram səviyyələrində modelləşdirilməsini tələb edir. İstifadəçini verilənlərin giriş strukturunu müəyyən etmək zərurətindən qurtarmaq üçün tətbiq sahəsinin informasiya səviyyəsində modelləşdirilməsini tələb edir. Paketin sistem hissəsi-nin tipikləşdirilməsi üçün daha bir səviyyə modelləşdirmə, funksional modelləşdirmə daha lazımdır. Tətbiq sahəsinin funksional modeli dil prosessorları ilə sistemin qalan hissəsi arasında tətbiq interfeys rolunu oynayır. O, paketin qeyri-prose-dur semantik ekvivalent olan giriş dillərinin translyasiyası zamanı aralıq dil kimi istifadə oluna bilər, məsələn direktiv dil yaxud «menyu» tipli dil üçün.

Tətbiqi proqramlar paketinin sistem hissəsinin strukturu, əsasən, paketin tətbiq sahəsinin modelləşdirilməsinin seçilmiş səviyyələri ilə, bu hissənin ayrı-ayrı komponentlərinin strukturu isə seçilmiş modelləşdirmə üsulları ilə müəyyən olunur.

Tətbiqi proqramlar paketinin bir sıra keyfiyyət xarakteristikalarının (etibarlılıq, kommunikativliyini və s.) təmin etmək üçün bir sıra funksiyaların mərkəzləşdirilməsi tələb olunur. Mərkəzləşdirmə paketin xüsusi komponenti – paketin monitoru – vasitəsi ilə təmin olunur. Bu məqsədlə də həmçinin paketin daxili dilləri-nin olması zəruridir. (verilənlərin manipuliyasiyası dili, monitora verilən sorğular, dili və s.).

Məlumdur ki, tətbiqi proqramlaşdırma sahəsində ən mühüm problemlərdən biri də tətbiq sahəsinin proqram modelləşdirilməsi üsullarının təplərə ayrılmasıdır, yəni tipləşdirilməsidir. Bu zaman funksional modulların strukturu, parametrlərin ötürülməsi üsulları, daxili dillər üzrə standart razılaşmaların işlənib hazırlanması nəzərdə tutulur. Bu problemin həll olunması paketin sistem hissəsi ilə funksional hissəsi arasında dəqiq interfeysin yaradılmasına bu isə öz növbəsində eyni bir funksional hissəsin müxtəlif sistem hissələri ilə birlikdə və əksinə, eyni bir sistem hissənin müxtəlif funksional hissələrlə birlikdə tətbiq olunma-sına imkan verərdi.

Baxılan dissertasiya işində tətbiqi proqramlar paketinin kifayət qədər sadə şəkildə reallaşdırıla bilən sistem hissənin konfiqurasiyası təklif olunmuşdur. O sadə əmr dillərinin tətbiq olunmasına əsaslanmışdır: funksional modelə VƏ / VƏ YA qrafı vasitəsilə verilən məqsədlər ağacı kimi baxılması; şəbəkə vasitəsilə verilən alqoritmik modelə, nəzərdə tutulmuş məqsədlərin reallaşdırılması üzrə fəaliyyətlər toplusu kimi baxılması; informasiya modelinə verilənlərin şərhi dili vasitəsilə müəyyən olunan verilənlərin təqdimatı dilində mətnlər toplusu kimi baxılması. Belə sadə olmasına baxmayaraq, paketin uyğun sistem hissəsi çox böyük müvəffəqiyyətlə xeyli geniş tətbiq sahələri üçün tətbiq oluna bilər.

Tətbiq sahəsinin obyektlərinin informasiya modellərinin hər hansı bir formal dillərdə mətnlər kimi şərh olunması verilənlər bazasının formalaşdırılmasına və nəticələrin istifadəçiyə verilməsinə translyasiya prosesi kimi baxmağa imkan verir. Giriş generatoruna, həm də nəticə generatoruna bu halda çoxdilli sistem kimi baxıla bilər. Bu sistemlər sintaktiki darə olunan translyasiya metodları yaxud kompilyatorlar kompilyatoru qurulması metodları əsasında reallaşdırırlar.

Mürəkkəb struktura malik tətbiqi proqramlar paketi sistem proqramlaşdırılmasının müxtəlif metod və üsullarından istifadə olunmağa yaradılırlar. Tətbiqi proqramlar paketi yaradan instrumental sistemlərə yaradılmanın qəbul olunmuş prosedur qaydalarına riayyət edən vasitə kimi baxmaq olar. Paketin sistem hissəsinə nəzərən intsrumental sistem metasistemdir və o translyatorları yaradan sistemlərdə tətbiq olunan metodlara oxşar metodlarla yaradıla bilər.

İnstrumental sistemin istifadə olunması texnologiyalarını sistemin yaradıcılarından uzaqlaşdırmaq üçün bu texnologiyaların strukturlaşdırılması zəruridir. Strukturlaşdırma üsullarından biri texnologiyaların prosedur modeli ola bilər.

Formalaşdırmanın müəyyən həddi olduğundan, texnologiya layihələndirərkən bu həddi nəzərə almaq zəruridir. Bu hədd nəzərə alınmadıqda sistemi yaradan kollektivin işi xeyli çətinləşir.

İSTİFADƏ OLUNMUŞ ƏDƏBİYYAT

1. Карминский А. М., Черников Б. В. Применение информационных систем в экономике. Учебное пособие, изд-во Форум Инфра-М, 2012. - 320 стр.
2. Информационные системы в экономике. Под ред. проф. Романова А. Н., Одинцов Б. Е. Учебное пособие, 2-е издание, изд-во Вузовский учебник, 2010, - 410 стр.
3. Терехов А. Н. Технология программирования. Учебное пособие, изд- во Бином, 2011. – 148 стр.
4. Анализ опыта реализации диалоговых систем. /Л.В. Кокарева, И.И. Малашинин, О.Л. Перевозчикова, Е.Л. Ющенко // Упр. системы и машины. —1987.—№4. -с. 63-69.
5. Аннотированный библиографический указатель по диалоговым системам. - Пущино: НИВЦ АН СССР, 1986.—220 с.
6. Бабаев И.О., Лавров С.С. Развитие автоматизированной системы решения задач СПОРА // Пакеты прикладных программ. Инструментальные системы. —М.: Наука, 2007.—с. 5-18.
7. Диалоговая информационная система управления пакетами прикладных программ ДИСУППП / О.Л. Перевозчикова, И.В. Криштопа, Г,Э. Микаилов, М.Н. Мусаев и др.; Ин-т кибернетики им. В.М.Глушкова АН УССР—Киев, 1985 —290с. —деп. В Гос ФАП СССР. 24.12.85, №50860000937.
8. Ещенко В.Г. О реализации на языке высокого уровня математического обеспечения пакета программ РАЗМЕЩЕНИЕ.— Программирование, 1983, № 2, с.12—16.
9. Каталог диалоговых систем: материалы по математическому обеспечению—Киев: Ин-т кибернетики АН УССР, 2006, —232 с.

10. Кахро М.М., Калья Л.П., Тыугу Э.Х. Инструментальная система программирования ЕС ЭВМ (ПРИЗ).-М.: Финансы и статистика, 1988.- 181 с.
11. Компьютерные технологии обработки информации: Учебное пособие /С.В. Назаров, В.И. Першков, В.А. Гафинцев и др.; Пол. ред. С.В. Назарова.—М.: Финансы и статистика, 1995—248 с.
12. Крижановский В.В., Мусаев М.Н., Перевозчикова О.Л. Создание схем вычислений маршрутных систем.—Киев, 1984.—30с.—(Препринт/АН УССР, Институт Кибернетики; 84-27).
13. Мусаев М.Н. Об одной проблеме проектирования проблемно-ориентированных систем. Труды конф. «Новые информационные технологии и проблемы прикладной математики», Баку, 1997.
14. Мусаев М.Н. Об одном способе построения моделей предметных областей.—Изв. АН Азербайджана, сер. Физ.-техн. и матем. Наук, XYI том, № 5-6, 1995.
15. Пакеты программ офисного назначения: Учеб. Пособие /С. В. Назаров. Л.П. Смольников, В.А. Тафинцев и др.;Под, ред. проф. С.В. Назарова.-М.: Финансы и статистика, 2007—320 с.
16. Перевозчикова О.Л. Модели общения при решении задач на ЭВМ. //Упр. системы и машины.—1987.—№5.—с.61-68.
17. Перевозчикова О.Л., Ющенко Е.Л. Диалоговые системы.—Киев: Наук. думка, 1990.—184 с.
19. Перевозчикова О.Л., Ющенко Е.Л. Системы диалогового решения задач на ЭВМ—Киев: Наук. думка, 1986.—264 с.
20. Чаплинскас А.А., Матулис. В.А. Система «Вильнюс»: конпепции, структура и технологии ее использования. — Вильнюс: Ин-т математики и кибернетики АН Лит. ССР, 1981 .—146 с.
21. Петрушин В.А. Экспертно-обучающие системы.-Киев: Наук думка, 1992.-с.196.

22. Петрушин В.А., Третьяк В.А., Лисюк Т.А. Инструментальные средства для создания экспертно- обучающих систем. / Интеллектуальные системы в задачах проектирования, планирования и управления в условиях неполноты информации. Матер. всесоюз. науч.- техн. совещ. (январь 1990 г.)-Казань: НПО ”Волга“, 1990.-с.112-115.
23. Ездаков А. Л. Функциональное и логическое программирование. Учебное пособие. Изд-во Бином, 2011. – 119 стр.
24. Фуфаев Э. В., Фуфаева Л. И. Пакеты прикладных программ. Учебное пособие, изд-во Academia, 2012. – 352 стр.