ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
«АЛМАТЫ ЭНЕРГЕТИКА ЖӘНЕ БАЙЛАНЫС УНИВЕРСИТЕТІ»
коммерциялық емес акционерлік қоғамы
IT-инжиниринг кафедрасы

ҚОРҒАУҒА ЖІБЕРІЛДІ
Кафедра меңгерушісі
PhD, доцент
_____ Т.С. Картбаев
«____» _____ 2019 ж.

ДИПЛОМДЫҚ ЖОБА

Тақырыбы: Delphi ортасында автоматтандырылған ақпараттық басқару жүйесін құру

Мамандығы: 5В060200 – «Информатика»

Орындаған: Мухатай М.Б. Тобы: ИНФк-15-1
Ғылыми жетекші: аға оқытушы Айткулов Ж.С.

Кеңесшілер:
Экономикалық бөлім: аға оқытушы _____ С.К. Тулегенова
«10» 04 2019 ж.

Өміртіршілік қауіпсіздігі: т.ғ.д., аға оқытушы _____ Ш.Ш. Бекбасаров
«24» 04 2019 ж.

Есептеу техникасын қолдану: аға оқытушы _____ Ж.С. Айткулов
«22» 05 2019 ж.

Норма бақылаушы: аға оқытушы _____ К. Мукапил
«21» 05 2019 ж.

Сын-пікір беруші: т.ғ.к., асс. профессор _____ Ф.Н. Абдолдина
«21» 05 2019 ж.

Алматы 2019

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
«АЛМАТЫ ЭНЕРГЕТИКА ЖӘНЕ БАЙЛАНЫС УНИВЕРСИТЕТІ»
коммерциялық емес акционерлік қоғамы

Басқару жүйелері және ақпараттық технологиялар институты

IT-инжиниринг кафедрасы

Мамандығы 5В060200 – «Информатика»

Дипломдық жобаны орындауға берілген

ТАПСЫРМА

Білім алушы Мухатай Меиржан Биримжанович

Жобаның тақырыбы: Delphi ортасында автоматтандырылған ақпараттық басқару жүйесін құру

2018 жылғы «26» қазан № 124 университет бұйрығымен бекітілген.

Аяқталған жобаны тапсыру мерзімі: «24» мамыр 2019 ж.

Дипломдық жобаның бастапқы мәліметтері (зерттеу нәтижелерінің талап етілген параметрлері мен объектінің бастапқы мәліметтері): Кәсіпорын жарғысы, статистикалық мәліметтер және қарастырылып отырған кәсіпорын туралы мәліметтер Интернет желілерінен алынды.

Дипломдық жобада қарастырылған мәселелер тізімі немесе диплом жобаның қысқаша мазмұны:

- талдау бөлімі;
- жобалау бөлімі;
- жүзеге асыру және тестілеу бөлімі;
- экономикалық бөлім;
- өміртіршілік қауіпсіздігі;
- А қосымшасы. Техникалық тапсырма;
- Ә қосымшасы. Программа листингі;
- Б қосымшасы. Ендіру актісі.

Графикалық материалдар тізімі (міндетті сызбалар дәл көрсетілуі тиіс): 9 кесте, 32 сурет ұсынылған.

Ұсынылатын негізгі әдебиеттер:

1 Климова, Л. М. Delphi 7. Основы программирования. Решение типовых задач. Самоучитель / Л.М. Климова. - М.: КУДИЦ-Образ, 2017. - 480 с.

2 Культин, Никита Основы программирования в Delphi 8 для Microsoft.NET Framework. Самоучитель (+ CD-ROM) / Никита Культин. - М.: БХВ-Петербург, 2013. - 400 с.

3 Культин, Никита Основы программирования в Delphi XE (+ CD-ROM) / Никита Культин. - М.: БХВ-Петербург, 2011. - 416 с.

4 Осипов, Дмитрий Delphi. Программирование для Windows, OS X, iOS и Android / Дмитрий Осипов. - М.: "БХВ-Петербург", 2014. - 464 с.

Дипломдық жобаның бөлімдеріне қатысты белгіленген кеңес берушілер

Бөлімдер	Кеңесшілер	Мерзімі	Қолы
Экономикалық бөлім	С.К. Тулегенова	10.04.2019	
Өміртіршілік қауіпсіздігі	Ш.Ш. Бекбасаров	24.04.2019	
Программалық қамтама	Ж.С.Айткулов	22.05	
Норма бақылау	К. Мукапил	04.04.2019 – 10.05.2019	

Дипломдық жобаны орындау
КЕСТЕСІ

Бөлімдер атауы, қарастырылатын мәселелер тізімі	Ғылыми жетекшіге ұсыну мерзімдері	Ескерту
Талдау бөлімі	02.11.18 – 26.12.18	орындалды
Жобалау бөлімі	05.01.19 – 18.02.19	орындалды
Жүзеге асыру және тестілеу бөлімі	21.02.19 – 11.04.19	орындалды

Тапсырманың берілген күні «30» қазан 2018 ж.

Кафедра меңгерушісі _____ Т.С. Картбаев

Жобаның ғылыми жетекшісі _____ Ж.С. Айткулов

Тапсырманы орындауға алған
білім алушы _____ М.Б. Мухатай

Аңдатпа

Дипломдық жобаның нәтижесінде Delphi программалау тілін қолданып кәсіпорынның автоматтандырылған ақпараттық басқару жүйесі құрылды. Бұл жүйеге мәліметтерді классификациялау және ұйымдастыру, бөлімдері енген. Себебі Delphi программамен жұмыс істеу принциптері Windows жүйесіне негізделген, оның объектілері терезе түрінде ашылады. Сондықтан Windows жүйесімен таныс болған қолданушы Delphi программасында берілгендер қорымен жұмыс істеу қиындық тудырмайды. Дипломдық жоба кәсіпорын, фирма, кез-келген ЖАТ (жаңа ақпараттық технологиясы) қолданысқа алу мүмкіндігі бар ұйымдардың қолданушы кезіндегі тап болатын қиындықтарын шешуге, іс-қағаздармен жұмыс барысында және клиентпен тікелей ақпарат алмасу үшін жұмсалатын жұмыс көлемін барынша азайту проблемаларын шешуге арналған. Клиент-сервер жұмысын автоматтандыру барысында арадағы пайда болатын қателіктер қаупінің мүмкін болатын ең төменгі көрсеткішіне қол жеткізілді.

Дипломдық жобада мәліметтерді енгізетін, өңдейтін, сұрыптайтын, баспаға жібере алатын, берілген мәліметтер негізінде анализ жасай алатын, технологиялық процесстерді кәсіпорын жұмысын автоматтандыру мақсатында қазіргі заманғы технологиялық құралдарды дамыту және жұмыс барысын автоматтандыруға үйренуге негізделген, қолданушыға ыңғайлы интерфейсі бар бағдарлама құрылды.

Аннотация

В результате дипломного проекта была построена автоматизированная информационная система управления предприятия с использованием языка программирования Delphi. Эта система включает в себя разделы, организации и классификации данных. Потому что принцип работы среды Delphi основывается на системе Windows. Поэтому пользователю, знакомый с системой Windows, не составит труда создать базы данных в среде Delphi. В данном дипломном проекте предусматривается решение проблем, с которыми сталкиваются предприятия, фирмы, организации, имеющие возможность применения любой из сторон (новая информационная технология), для решения проблем минимизации объемов работ, затрачиваемых в процессе работы с ценными бумагами и для прямого обмена информацией с клиентом. При автоматизации работы клиента-сервера достигнут минимальный допустимый показатель риска ошибок, возникающих между собой.

В дипломном проекте была создана программа с удобным интерфейсом пользователя, основанная на внедрении, обработке, сортировке, печатании данных, анализе на основе данных, обучении автоматизации технологических процессов в целях автоматизации работы предприятия и развития современных технологических средств.

Annotation

As a result of the degree project was built an automated information management system of enterprise with using the programming language Delphi. This system includes sections, organizations, and classifications of data. Because the principle of the Delphi environment is based on the Windows system. Therefore, the user familiar with the Windows system will not be difficult to create a database in Delphi. This degree project provides a solution to the problems faced by enterprises, firms, organizations that have the ability to use any of the parties (new information technology) to solve the problems of minimizing the amount of work spent in the process of working with securities and for direct exchange of information with the client. When automating the client-server operation, the minimum allowable risk of errors occurring among themselves has been achieved.

In the degree project, a program with a user-friendly interface was created, based on the implementation, processing, sorting, printing data, analysis based on data, training in the automation of technological processes in order to automate the work of the enterprise and the development of modern technological means.

Мазмұны

Кіріспе

Қазіргі заманғы автоматтандырылған басқару жүйесінде проектілеудің дамыған әдіс-тәсілдері жүйелік (систематикалық) проектілеу маңайында шоғырланған, яғни адам мен автоматтандырылған жүйе арасындағы ақпарат алмасудағы «кіріс» пен «шығыс» деңгейіндегі қиындықтарды шешу факторларымен айқындалған. Сонымен қатар автоматтандырылған басқару жүйесінің қолданушыларының қалауын ескере отырып жасалған анализден байкайтынымыз ол жүйелердегі ақпарат алмасудың комплексті біріктірілуінің жоқтығын байкаймыз.

Delphi ортасында автоматтандырылған ақпараттық басқару жүйесін құру бұл кез-келген басқару формасының актуализациясына көмектеседі. Осыған орай бөлім меңгерулерінің жұмысы жүйесі автоматтандыруды өңдеу принциптерін құру және эффективті функциялау мемлекет пен қоғамның қатысына тәуелді маңызды мақсаттың бірі. Өзіміздің елде кез-келген жұмысты автоматтандыруды жобалау және эксплутациялау тәжірбиесі арта түсуде. Ғылыми техникалық прогресінің негізгі бағыты ретінде де өндірістер, кәсіпорындар, тағы басқа әртүрлі ауданда технологиялық процестерді автоматтандыру жұмыстарын комплексті жүргізу болып табылады. Осы айтылғанның барлығы көптеген білім, техника, өндірістік және шаруашылық кәсіптердің, тағы басқалары қоғамның салаларына байланысты күннен-күнге дамып артуда. Негізінен осы салаларда бірнеше ондаған жылдар бойы автоматтандырылған басқару жүйесі белгілі процесстер бойынша эффективті қолданып келді.

Бұл жұмысты Delphi ортасында автоматтандырылған ақпараттық басқару жүйесін құрылу принциптері мен функциялары ретінде клиент-сервер жүйесіне жаңа ақпараттық технология (ЖАТ) енгізуді негізге ала орындадым.

Басқару жүйесін автоматизациялау процесстерінің сәтсіздікке ұшырауы көбінесе техника бөлімінің басқарушыларының осы салада қажетті білімдерінің болмауында. Көптеген жоғары мамандары өзінің кәсібіне ЖАТ – ны қалай және қандай түрін қолдану керек деген сұрақтарына жауап іздеуде. Автоматтандыру әртүрлі техникалық құралдармен қамтамасыз ету, тиімді қолдану, біріншіден жұмыс қарқын тиісте деңгейде бөлу және осымен қатар жұмыс барысында, құжат алмасу кезінде қателік кетудің ең төмен көрсеткішіне жету сұрақтары көрсетілді. Бірақ айтып кететін жағдай – бұларды шешу тек бар ресурстармен программалық – техникалық құралдармен емес, қолданушылардың, яғни кәсіпорын жұмысшыларының ұқыптылығына және олардың квалификацияларына байланысты [1].

Дипломдық жобаның мақсаты – жоғарыда айтылған мәліметтерді енгізетін, өңдейтін, сұрыптайтын, баспаға жібере алатын, берілген мәліметтер негізінде анализ жасай алатын, технологиялық процесстерді кәсіпорын жұмысын автоматтандыру мақсатында қазіргі заманғы технологиялық құралдарды дамыту және жұмыс барысын автоматтандыруға үйренуге негізделген , қолданушыға ыңғайлы интерфейсі бар бағдарлама құру.

Мәліметтер қорын өңдеу жұмыстары серверде атқарылатын, ал клиент қарапайым браузер болатын программалар жасалуда. Delphi-дің мүмкіндігі көп болуына байланысты жоғарыда айтылған мақсатты жүзеге асыру үшін осы ортаны пайдалануға болады.

Дипломдық жобаның өзектілігі:

– есептеуде тап болатын қиындықтарды шешуге, іс-қағаздармен жұмыс барысында ақпарат алмасу үшін жұмсалатын жұмыс көлемін барынша азайту;

– есептеу жұмысын автоматтандыру барысында арадағы пайда болатын қателіктер қаупінің мүмкін болатын ең төменгі көрсеткішіне қол жеткізу.

1 Талдау бөлімі

1.1 Кәсіпорынның автоматтандырылған жұмыс орнын жасаудың негізгі принциптері мен талаптары

Негізінде дербес компьютер болып табылатын автоматтандырылған жұмыс орнының функциясының шарты дұрыс қойылса және ақпаратты өңдеу әдістері адам мен машина арасында дұрыс бөлінсе жұмыс істеу сандық нәтиже беруі мүмкін, сонда ғана автоматтандырылған жұмыс орыны тек ғана жұмыстың өнімділігін ғана емес, сонымен бірге басқару тиімділігін арттыратын әдіс болып табылады.

Енді автоматтандырылған жұмыс орнының қалып күйін және даму көрінісін дербес компьютерде кең ауқымды түрде қарастырайық. Сонымен қатар, автоматтандырылған жұмыс орнының программалық қамсыздандырудың және кейбір техникалық сұрақтарына тоқталайық.

Казіргі уақыттар түрлі мамандық саласында ойлауды және әр түрлі мамаіттық интенсифтендіру жасалып, дербес компьютер негізінде жұмыс істейтін автоматгандырылған жұмыс орны кең ауқымды таралуда.

Оқу жүйесінде қызмет етушілердің және басқару қызметінің автоматтандырылған жұмыс орнының негізгі құрамының элементтерін, олардың дамуын және қолдануын қарастыруына тоқгайық.

Автоматтандырылған жұмыс орнының техникалық негізі болатын, дербес компыотердің жалпы схемасын орта мектептің оқулықтарында кездестіруге болады.

Тәжірбиелерді, ғылыми ізденістерді талдау нәтижесәнде автоматтандырылған жұмыс орнына қойылатын бірқатар талаптар анықталады, яғни:

– машинаның ақпараттық және оның есептеушілік сұранысын жаңаша қанағаттандыру ;

– қолданушының скранысына аз уақытта жауап беру;

– автоматтандырылған жұмыс орнының жұмыс істеу тәсілдерін үйренудің қарапайымдылығы және қарым-қатынасы,сенімділігі, қарапайым жұмыс істеуі;

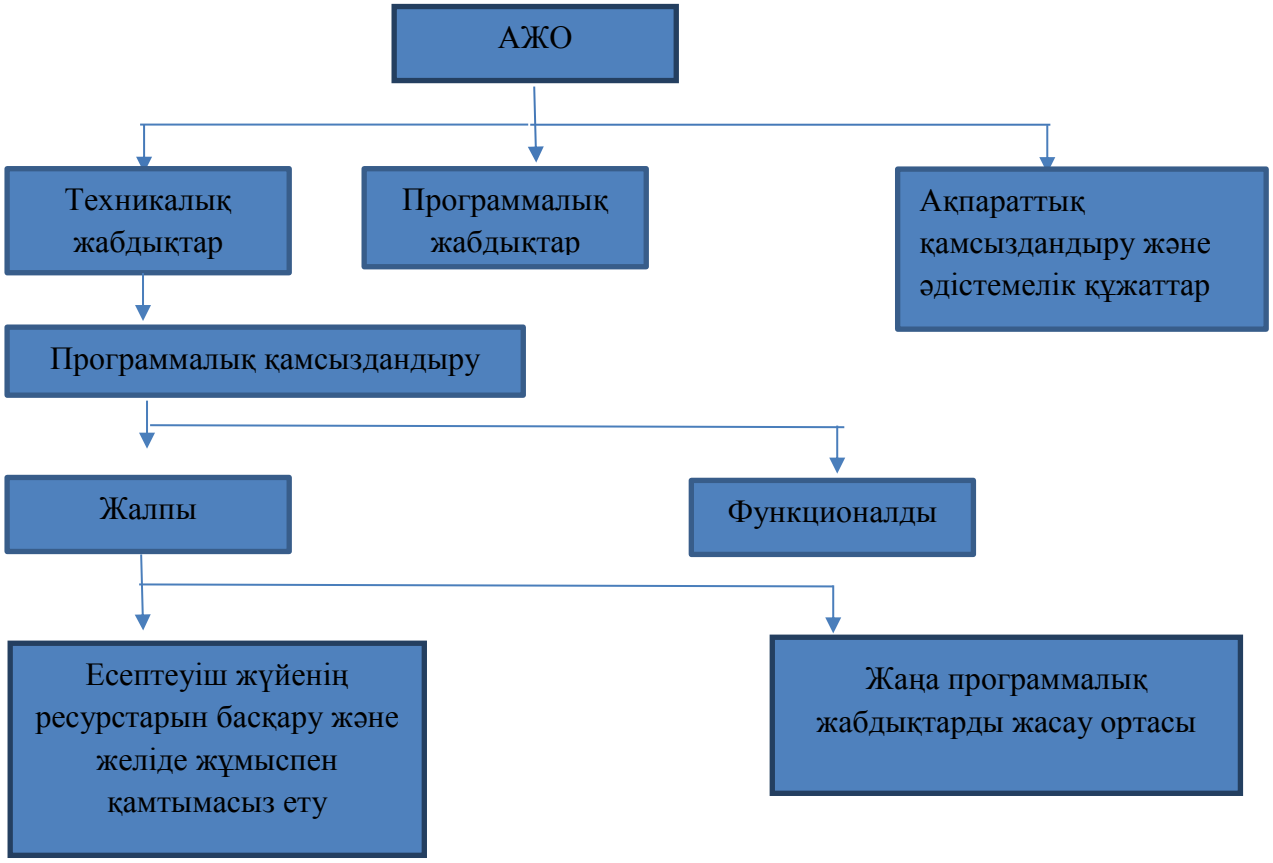
– қолданушының тез үйрену мүмкіншілігі;

– қолданушыға қарағанда төзімділігі;

– қомпьютер желісінің құрамында жұмыс істеу мүмкіншілігі.

Автоматтандырылған жұмыс орнының жалпы схемасын қарастырайық:

Соңғы уақытта толық түрде қалыптастырылмайтын мәселелерді шешу құралдарын жасауға үлкен көңіл бөлінген. Олар семантикалық деп аталады. Бұндай мәселелер экономикалық объектілерді оперативті басқару кезінде және көбінесе толық емес мәлімет шарттарында шешім қабылдау кезінде туындайды.



1.1-сурет – Автоматтандырылған жұмыс орнының схемасы

АЖО тілдері пайдаланушылық – бағытталған және маманды – бағытталған болуы керек. Бұл пайдаланушылардың сұрыпталу ерекшеліктеріне байланысты олар тек маманданған жарамдылық бойынша ғана емес, сондай-ақ қызмет жағдайының иерархтясына оқылғандығына, шешуші мәліметтер түрі бойынша бөлінеді. Бұндай енудің оңайлығына қарамастан, қарапайым тілді пайдалану клавиатура арқылы күрделі емес нәтижелерді алу үшін алып құрлымды енгізу қажеттілігі үшін қандай да болмасын сезілетін артықшылықты бере алмайтындығын ескеру керек.

Кез-келген тілде сияқты АЖО тілдерінің негізін алдын-ала анықталған терминдер және тәсілдерді бейнелеулер (олар арқылы терминдерді ауыстырып немесе толықтырып жаңа түрлерін белгілейді) құрайды. Бұл АЖО-ны жоспарлаған кезде АЖО-ның терминологиялық негізін арнайы түрмен сұрыптау қажеттілігіне әкеледі (тілдің негізгі синтаксистік құрылымын және терминдер мен бірлескен түрлері аралығындағы семантикалық қатынасын анықтау). Осыған байланысты АЖО-ның қарапайым сұрыпталу қажеттілігі туындайды, мысалы, мәліметті өңдеудің кейбір пайдаланушылық режимдерінде көрсету мүмкіндіктері бойынша: сандық, мәтіндік, аралас. Ал өте күрделі жағдайларда АЖО сұрыпталуы мәлімет қорының ұйымдастырылуымен анықталады [2]. Тіл мүмкіндіктері көп жағдайларда пайдаланушы мәліметтік қамтамасыздықты таратуға сәйкес келетін формальді құрылымды құра алатын ережелер тізімін анықтай алады. Мысалы, кейбір

АЖО-ларда барлық мәліметтер мен құрылымдар кесте түрінде (кестелік АЖО) немесе арнайы түрдегі операторлар болып бекітіледі.

Пайдаланушының тілі АЖО-ны диалог түрлері бойынша да бөлінеді. Диалогты қолдау құралдары ақыр соңында пайдаланушыға қажет білімнің тілдік құрылымын анықтайды.

Қамтамасыз етілу жүйелерге біріншіден мыналарды жатқызады: техникалық, мәліметтік, программалық және ұйымдастырушылық. Сонымен қатар кіші жүйе түрлері де бар.

Техникалық қамтамасыздық, маманның делдалдарсыз жұмысын (программисттер, операторлар және т.б) көздейтін мамандanған персоналды компьютер негіз болатын техникалық құралдар қасиетін бейнелейді. Толық АЖО-ларда оңай компьютерлермен 4-6 адам пайдалана алады. Мамандanған персоналды компьютер кешеніне процессор, дисплей, пернелік тақта, мәліметтің магниттік жинағыштары, басып шығару құрылғылары және графо құрылғыштар кіреді.

1.2 SQL Server-ді автоматтандырылған жұмыс орны ретінде пайдаланудың тиімді жолдары

Microsoft SQL Server – дерекқорды басқарудың салыстырмалы жүйесі (DBMS). Реляциялық дерекқорларда деректер кестелерде сақталады. Өзара байланысты деректерді кестелерге топтастыруға болады, сонымен қатар кестелер арасындағы қарым-қатынас орнатылуы мүмкін. Осыған байланысты реляциялық атау – ағылшын тілінен реляциялық (байланысты, өзара байланысты, өзара байланысты). Пайдаланушылар серверде деректерді қосымшалар арқылы қолданады және әкімшілерге дерекқорды теңшеу, басқару және қолдау міндеттерін орындайды, серверге тікелей қол жеткізеді.

Жалпы программалық қамсыздандыру компьютер жұмысын басқаруды, жаңа программаларды қосу және өңдеуді қамтымасыз етеді. Бұған операциялық жүйе және әр түрлі қызмет атқаратынпрограммалар кіреді.

Автоматтандырылған жұмыс орнының мамандық бағдары программалық қамсыздандырудың функционалды бөлігімен анықталады. Аймақтық анықталған пәндік есептерін шешу қамтамасыз етіледі. Функционалдық программалық қамсыздандыру көбінесе «адам-машина» деген сұраққа көңіл бөлінеді.

SQL Server – бұл ауқымды дерекқор, ол деректердің үлкен көлемін сақтай алады және дерекқорға бір мезгілде кіретін көптеген пайдаланушылардың жұмысын қолдайды.

Microsoft SQL Server 6.5 клиент-сервер дерекқорларын басқару жүйелерінің бірі болып табылады. Бұл ДҚБЖ басқарудың және пайдаланудың қарапайымдылығын сақтай отырып, деректерді репликалауды, параллельді өңдеуді, салыстырмалы түрде арзан аппараттық платформалар бойынша ірі дерекқорларға қолдау көрсету сияқты таратылған деректерді өңдеу жүйелеріне қойылатын талаптарды қанағаттандыруға мүмкіндік береді.

МҚБЖ MS SQL Server дерекқорды құрудың ықтимал нұсқаларын ұсынады:

– Management Studio интерфейсін пайдалану;

– SQL пәрмендерін пайдалану.

Дерекқорды жасау – бұл файл атауын анықтау, дерекқор файлдарының өлшемін және орнын анықтау, транзакция журналының параметрлерін анықтау.

MS SQL Server дерекқорларында үш файл түрі бар:

а) алғашқы деректер файлдары. Әдетте, MDF кеңейтімі пайдаланылады. Кез-келген дерекқорда барлық басқа дерекқор файлдарының деректері мен орналасуы бар бір негізгі файл бар;

б) қосымша деректер файлдары. Әдетте, NDF кеңейтімі пайдаланылады. Екінші – негізгі және журнал файлдарынан басқа кез-келген файл. ББ-да бір қайталама файл болмауы мүмкін;

в) журнал файлдары. Әдетте LDF кеңейтімі пайдаланылады. Әр дерекқорда кемінде бір журнал файлы бар. Транзакция журналы дерекқорда болған өзгерістер туралы ақпаратты қамтиды, яғни. транзакция (операция) жүзеге асырылған кезде, бұл журналға ақпарат енгізіледі. Уақыт өте келе, бұл журнал көбірек артып келеді, сондықтан оның өлшемін қадағалау керек. Мәмілелер журналының негізгі мақсаты деректердің тұтастығын қамтамасыз ету болып табылады. Бұл дерекқордағы өзгерістерді болдырмауға мүмкіндік береді.

Әкімшілік және жүктемені бөлу ыңғайлы болу үшін файлдар екі түрге бөлінген файлдық топтарға біріктірілуі мүмкін:

а) негізгі файлдық топтар. Бұл бастапқы файлды және басқа топқа анық орналастырылмаған барлық файлдарды қамтиды;

б) пайдаланушы файл топтары дерекқордағы пайдаланушы жасаған кез келген топ.

Журнал файлдары кез-келген файл тобына кірмейді, олар тұрақты файлдардан бөлек өңделеді.

Жаңа дерекқор – деректер базасының көшірмесі, оның барлық параметрлері жаңа деректер базасына көшіріледі. Әдепкіде, тек sysadmin және dbcreator рөлдеріне тағайындалған пайдаланушылар дерекқорларды жасауға рұқсат етілген.

Деректер базасын құру келесі пәрменді пайдалана отырып жүзеге асырылады:

Деректер базасын құру
[ON [PRIMARY] (NAME = 'logical_file_name',
FILENAME = 'физикалық_файл_аты
[, SIZE = мөлшері]
[, MAXSIXE = {max-size | UNLIMITED}]
[, FILEGROWTH = step_size_size [Mb | Kb | %])
[{FILEGROUP group_name}]
[, ... n]

[КІРІСПЕ (NAME = 'logical_file_name',
FILENAME = 'физикалық_файл_аты
[, SIZE = мөлшері]
[, MAXSIXE = {max-size | UNLIMITED}]
[, FILEGROWTH = step_size_size [Mb | Kb | %])
[, ... n]

Дерекқорды өзгерту. Дерекқор мына оператор арқылы жойылады:

DROP DATABASE database_name [... n]

Нәтижесінде, дерекқор пайдаланатын барлық файлдар жойылады. Дерекқордың иесі және sysadmin рөлінің пайдаланушылары жоюға құқылы, бұл құқық басқа тіркелгілерге ауыстырылмайды.

Дерекқордың иесін өзгерту арнайы сақталған процедура арқылы жасалады. Иеленуші төмендегідей дерекқордың пайдаланушысы болып табылмайтын кез-келген тіркелгіні жасай алады:

sp_changedbowner [[@ loginname =] 'username'

Дерекқордың атауын өзгерту:

sp_renamedb [@ old_name =] 'old_name', [@ new_name =] new_ name '

Дерекқордың атын өзгерту үшін оны бір пайдаланушы режиміне түрлендіру қажет.

Қолданыстағы журнал файлдарын және деректер файлдарын басқару үшін, қосымша деректер файлдарын немесе журнал файлдарын қосыңыз, файлдарды жойыңыз, сондай-ақ файл топтарымен жұмыс істеңіз, мына пәрменді пайдаланыңыз:

ALTER DATABASE дерекқоры
{Файлды қосу <note_on_file> [FILEGROUP атауына]
| ADD LOG FILE <note_on_file>
| FILE logical_file_name файлын алып тастаңыз
| FILEGROUP group_name қосыңыз
| FILEGROUP group_name параметрін алып тастаңыз
| Файлды өзгерту <note_on_file>
| FILEGROUP тобының атау тобының сипатын өзгерту}
мұнда <instruction_file_file> =
(NAME = 'logical_file_name,
FILENAME = 'физикалық_файл_аты
[, SIZE = мөлшері]
[, MAXSIXE = {max-size | UNLIMITED}]
[, FILEGROWTH = step_size_size [Mb | Kb | %])

Бұл пәрмен файлдарды бар файл тобына қосуға, файлдарды жоюға (бұл физикалық файлды жояды), файл топтарын қосу және жоюға, бар файлдардың физикалық параметрлерін өзгертуге, сондай-ақ файл топтарының қасиеттерін өзгертуге мүмкіндік береді: READONLY, READWRITE, DEFAULT (бұл сипаттарында бұл топта топ мүшелігі анықталмаған файлдар болады, әдепкі файл тобы бастапқыда әдепкі ретінде орнатылады.

Дерекқорды қысу. Дерекқорды қысу – файлдың пайдаланылмаған бөліктерін жою арқылы дерекқор файлдарының көлемін азайту процесі. Дерекқорды қысудың үш тәсілі бар:

– дерекқор параметрлерінде тиісті параметрді орнатқанда автоматты түрде қысу;

– MS SQL Server әкімшілік утилиталарын пайдаланатын дерекқор файлдарынан бос орынды жою;

– файлдарды (немесе файл топтарын) азайту, сондай-ақ кейінгі жою үшін файлдардың мазмұнын тазарту.

Автоматты түрде деректер базасының параметрі орнатылған болса, автоматты түрде деректерді қысу белгілі бір аралықта орындалады. Автоматты қысу әрекеттерінде дерекқордың қай бөлігін қысу керек екенін анықтау мүмкін емес. MS SQL Server дербес деректер базасының маңызды бөлігін шығаруға тырысады. Бұл операциялар төменгі пайдаланушы әрекеті кезеңінде орындалады.

DBCC SHRINKDATABASE ('db_name', ['пайыздық'] [, NOTRUNCATE | TRUNCATEONLY])

Параметрлердің сипаттамасы:

– database_name – қысылатын дерекқордың атауы;

– пайыздық – қысудан кейін кетуге болатын бос орынның пайызы;

– NOTRUNCATE – жиынтық кеңістік амалдық жүйеге қайтарылмайды, бірақ файлдарда сақталады, яғни. дерекқордың өлшемін физикалық түрде азайтпайды;

– TRUNCATEONLY – бос орын босатылғаннан кейін, файлда соңғы дәрежеде жойылады, ал деректер өзгермейді және пайыздық параметр еленбейді.

Дерекқорды қысу құқықтары тек sysadmin рөлі мен дерекқор иелеріне беріледі. Дерекқор қысылғаннан кейін, есеп жасалады:

– файл қысылған беттер саны;

– барлық файлдық деректер орналастырылатын беттердің болжамды саны;

– деректерді қамтитын беттер саны;

– файл әлі де қысылатын беттер саны.

Дерекқорды бастапқыдан кішірек өлшемге дейін кішірейту мүмкін емес.

Дерекқорды қысуды сонымен қатар оның әрбір файлын келесі пәрменмен қысу арқылы орындау мүмкін:

DBCC SHRINKFILE ('filename', ['finite_size'])

[EMPTYFILE | NOTRUNCATE | TRUNCATEONLY])

Параметрлердің сипаттамасы:

– файл атауы – қысылатын файлдың логикалық атауы;

– final_size – қысудан кейін файлдың болуы керек қалаған өлшем (мегабайттардағы бүтін сан). Егер бұл параметр көрсетілмесе немесе ең төменгі өлшемнен аз болса, файл ең аз мүмкін өлшемге дейін қысылады;

– EMPTYFILE – деректер файлдан файлдар тобының басқа файлдарына көшіріледі;

– NOTRUNCATE – бос орын бос операциялық жүйеге қайтарылмайды, яғни. файл өлшемі іс жүзінде азаяды. Бұл жағдайда деректер ықшам және файлдың басына жылжытылады;

– TRUNCATEONLY – Файл соңғы бетінен бастап қысқартылады. Деректердің қозғалысы жоқ.

Деректердің сақтық көшірмесі. Пайдаланушы жұмыс істейтін ақпараттың тұтастығына ерекше назар аудару қажет. MS SQL Server резервтік ақпараттың келесі түрлерін ұсынады:

– дерекқордың толық көшірмесі, ол дерекқорды апаттан кейін қалпына келтірудің бастапқы нүктесі болып табылады, бірақ деректердің көлеміне байланысты бұл процесс көп уақыт алады, сондықтан оны жиі орындау ұсынылмайды. Толық көшірме сақтық көшірменің соңында дерекқордағы барлық деректерді қамтиды;

– мәліметтер журналының көшірмесі соңғы сақтық көшірмеден кейін жүйеде болған деректердегі барлық өзгерістерді жазу үшін қажет. Журналдың көшірмесінде транзакциялар туралы ақпарат бар және деректер қорының көшірмесі ғана сәтсіздікке ұшыраған күйге оралуға мүмкіндік береді.

Деректердің дифференциалды көшірмесі дерекқордың соңғы толық сақтық көшірмесі жасалғаннан бері орын алған деректер өзгерістерін қамтиды. Бұл жағдайда тек өзгертілген беттер сақталады. Осылайша, соңғы дифференциалды көшірмесі деректер базасын қалпына келтіру үшін жеткілікті.

Сақтық көшірмені орындау үшін, ортаны таңдау керек, яғни. көшірмелерді жасау үшін пайдаланылатын құрылғыны анықтаңыз. Құрылғыны қосу үшін сақталған процедураны пайдаланыңыз:

sp_addumpdevice уст-у құрылғы_түрі ',' logical_name ',' physical_name '

Параметрлердің сипаттамасы:

– device_type – резервтік құрылғы түрі. Жарамды мәндер: TAPE (магниттік таспа), DISK (магниттік диск);

– logical_name, физикалық_атауы – тиісінше резервтік құрылғының логикалық және физикалық атауы.

Дерекқордың, транзакция журналының, файлдардың және файл топтарының сақтық көшірмесін жасау үшін мына пәрменді пайдаланыңыз:

BACKUP {LOG | DATABASE} db_name
[FILE = 'logical_file_name', ...]
[FILEGROUP = групп group_name ']
Logical_ device_name үшін
[C
[DESCRIPTION = 'түсініктеме']
[Дифференциалды]
[EXPIREDATE = 'күні']
[INIT | NOINIT] ...]

Параметрлердің сипаттамасы:

– DIFFERENTIAL – дерекқордың дифференциалды көшірмесі жасалады;

– EXPIREDATE – резервтік көшірме ескірген деп саналады және қайта жазылуы мүмкін күнді анықтайды;

– INIT | NOINIT – жүйе құрылғыны баптандыруды жүзеге асырады немесе жоқ.

Дерекқорды қалпына келтіру. Дерекқорды сақтық көшірмеден қалпына келтіргенде, қолданыстағы дерекқор қайта жазылады. Дерекқорды қалпына келтіру үшін мына пәрменді пайдаланыңыз:

ҚАЙТАЛАУ {LOG | DATABASE} db_name
'File_or_file_group'
[FROM logical_ device_name]
[DBO_ONLY]
[MOVE 'logical_file_name' TO 'физикалық_атауы'] ...]

Параметрлердің сипаттамасы:

– DBO_ONLY - тек иелеріне қалпына келтірілген дерекқорға кіруге рұқсат етіледі;

– MOVE - қалпына келтірілген файлға қай физикалық атау сәйкес келетінін көрсетеді. Әдепкіде, файл сақтық көшірме жасау кезінде анықталған физикалық атаумен қалпына келтіріледі.

1.3 Delpi мүмкіндіктерін және ерекшеліктерін сипаттау

Дипломдық жобада Delphi программалау тілін пайдалану себебім,қазіргі таңдағы күрделі және жұмыс істеу жағынан мүмкіндігі көп тілдердің бірі.

Программалау тілінде негізгі жұмыстарды істейді, яғни формаларды іске қосып және форма бетіндегі графикалық нәтижелерді шығаруға және есеп нәтижесін беріп отыруда пайдаланды.

Paradox берілгендер қорын пайдалануымның себебі, Delphi программалау тілімен жақсы байланысады және онымен жұмыс істеу кез-келген адамның қолынан келеді.

Берілгендер қорын басқару жүйесі (БҚБЖ) – берілгендер қорымен жұмыс істеуге арналған Delphi ортасындағы программалар қатарында dBase, Paradox, Access, FoxPro т.б. да бар. Database Desktop утилитасының құрамындағы Paradox ең қолайлы, көп тараған программа. Paradox программаның Paradox 1, Paradox 2, Paradox 3, Paradox4, Paradox5, Paradox6, Paradox 7 сияқты бірнеше нұсқалары бар.

Берілгендер қорында екі өлшемді кестенің жолдары жазбалар деп, бағандары өрістер деп аталады. Дәлірек айтқанда берілендер қорында кестедегі әрбір жол жазба болып табылады, ал жазба бірнеше өрістерге бөлінеді.

Windows жүйесінің элементтерімен таныс кез-келген қолданушының Paradox программасын оқып үйренуіне қиындық жоқ десе де болады. Өйткені, Paradox программасымен жұмыс істеу принциптері Windows жүйесіне негізделген, оның объектілері терезе түрінде ашылады.

Процедуралық программалау тілдерінде программаның жұмысы операторларды ретімен орындау бойынша, ал, логикалық программалау тілдерінде ол қатаң логикалық ережелерге сәйкес өзгертулер енгізу ретінде қарастырылған болатын. Объектіге бағдарлы оқиғалық программалау тіліне программаның жұмысы негізінен оқиғалар тізбегінен және түрлі объектілердің осы оқиғаларға жауабынан тұрады. Олардың визуальды түрлері – Visual Basic тілі Qbasic программалау тілі негізінде, ал Delphi (Дельфи) Объектілі Паскаль (Object Pascal) тілі құрылған (visual – көзбен көру, экрандық). Олар, әсіресе, Delphi программалау тілі – кез келген қосымшаны дайындауға болатын жылдамдығы тез, қуатты тіл.

Delphi-дің негізгі ерекшелігі – онда қосымша құруда компоненттік және объектілік тәсілдер пайдаланылды (Windows ортасында пайдаланатындықтан, Delphi-де программаны көбінесе қосымша деп айтады). Бұл программалау технологиясында нағыз революция жасады деуге болады.

Компоненттік тәсілдің мәнісі жеңіл: әр қосымша кітапханасы программалау ортасында дайындалып, арнайы іс-әрекеттерді орындайтын компоненттер элементтерінен жинақталады. Олар жеткіліксіз болса, объектіні өңдеуге арналған үстеме программа құрылады. Delphi-де қолданылатын негізгі кітапхананы визуальды компоненттер кітапханасы (VCL, Visual Component Library) деп атайды. Компоненттер панелінде топ-тобымен жинақталған, жүздеген кластарға тиісті, стандартты компоненттер бар. Пайдаланушы жаңа компонент дайындап, оны осы панельге қосуына да болады.

Delphi Windows жүйесінде праграммалаудың ыңғайлы құралы. Онда көптеген операторларды пайдаланып программа дайындау, программа мәзірін құру, анимация, мультимедиа процестерін ұйымдастыру, OLE технологиясын пайдаланып, басқа офистік қосымшаларды шақыру, олармен жұмыс істеу және т.б. іс-әрекеттерді орындау да мүмкін.

Delphi бағдарламасы Windows 95, Windows 98 немесе Windows NT операциялық жүйесінің басқаруымен жұмыс істейді. Көптеген Delphi-да құрылған программалар негізінен өндіріс және бизнес есептерін шешуге бағытталған. Бұл мәлімметтер қорымен және есеп беру жұмыстары басты шешілу керек есептер болып табылады.

Программалардың сәйкестендірілуі үлкен роль атқарылады. Бұл аппаратпен қамтамассыздандырумен байланысты (HardWare) жекелеген жағдайда мобильді компьютерлерді тарату кең ауқым алуымен байланысты.

Дыбыс, сурет, тексттік және цифрлық типтермен берілген информацияларды беру және оқыту, алу үшін арналған техникалық жабдықтардың әрі қарай дамуы.

Жоғарыда айтылғандай бизнеспен және өндіріспен тығыз байланысты болғандықтан қолданушылар Delphi-дан өздерінің есептерін шешу үшін идеал көмекші құрал тапты. Delphi-дің Visual Basic және C++ сияқты қолданушы интерфейсі бар. Қазіргі кезде көптеген фирмалар өз программа интерфейсінің стандарты ретінде қабылдады. Қолданушы интерфейсі визуалды құрылатын

болғандықтан Delphi ортасында программалауды тез программа құру ортасы делінеді.

Құрудың графикалық ортасынан басқа аспектісі ол – көмекші жүйесінінің күштілігі.

Delphi-да басқа да қазіргі программалау ортасы сияқты, объектілі бағытталған программаларға негізделген. Программа құру барысында дайын компоненттерді, олардың қасиетін, әдістерін және алдын-ала анықталған оқиғаларды пайдалану арқылы аз ғана программа кодымен айналып өтуге болады. Программа құрушыға бұл өзінің программасының қолданушы интерфейсін құру барысында көп уақыт үнемдеуді білдіреді[3].

Есеп беру генераторын және мәліметтер қорын қолданатын бизнес-программаларды құру Delphi-да өте жоғарғы дәрежеде орындайды. Borland фирмасы өткен жылдар бойы жиналған барлық технологиялық білімін пайдаланды. Delphi-да BDE көмегімен DBASE-. PARADOX-. ACCES- және ASSCI мәліметтер қоры кестелерімен тікелей байланыса алады.BORLAND SQL LINKS FOR WINDOWS драйвері SQL-серверімен жұмыс істеуге арналған ORACLE. SYBASE. INFORMIX. INTERBASENT және DB2 көбінесе тестілеу үшін қолданылатын СУБД INTERBASE берілген. Басқа формадағы мәліметтер қорымен байланыс орнату үшін ODDBC-драйверін пайдалану керек. Сонымен қатар Delphi арқылы кез-келген масштабтағы клиент-сервер типіндегі программалар құруға болады, есеп беру үшін TQuickreport интегралданған компонеттері бар.

Delphi-дың аппараттық құрылымына программа құру үшін програмистке ассемблер беріледі. Осының нәтижесінде Intel-Assembler коды Object Pascal программа кодына көше алады.

IDE Delphi құрамы және құрылғысы. Delphi-ді іске қосқан кезде экранда интерграцияланған 4 терезе ашылады: басты терезе, форманы жобалау терезесі, кодты редактілеу және объектілер-инспекторы

Басты терезе. Экранның үстінгі бөлігінде басты терезе орналасқан. Онда басты мәзір жолы, саймандар тақтасы және палитра компоненті орналасқан. Басты терезе Delphi жүктеліп тұрғанда ашық болады. Басты терезе жолында осы уақытты ашық тұрған проктінің аты көрсетіледі.

Саймандар тақтасы. Саймандар тақтасы мәзір командасы көрсетілген батырманы құрайды, мысалы File,View, және т.б. Батырманы шерту нәтижесі бас мәзірдегі командалар таңдауына сәйкес келеді, мысалы файлды ашу үшін File мәзірінде Open командасын орындау керек немесе Standart саймандар тақтасында Open батырмасын шерту керек.

Палитра компоненті. Палитра компонентінде өзіңіздің қосымшаңызда құрылған компоненттер көрсетіледі.Әрбір Delphi элементі үшін негізгі компоненті болып табылады, сонымен қатар Delphi визиуалды компонентіне кітапхана болып табылады. Олар сіздің қолданбалы программаңызды қолданушы интерфейсін құруға көмектеседі.

Диалогты терезе келесі екі түрлі әдіспен ашылады:

а) тышқанның оң жақ батырмасын кез келген жерге шертіп Properties командасын орындау;

б) Component мәзірінде Configure Palette командасын таңдау .

Форманы жобалаушы. Басты терезе – бұл қосымшаны іске қосқан кездегі бірінші көрінетіні. Егер қолданушы бұл терезені жапса, онда ол қосымшаны да жабады. Жоба бірнеше форманы сақтай алады. Оның қайсысы басты екенін білу үшін Project/Options диалогты терезеде Project мәзірінде Options командасын орындағанда білуге болады. Форма Windows – терезесі болып табылады. Ол үлкеюі, кішіреюі, экран бойынша кішіреюі және пиктограммаға айналады.

Редактор кодының терезесі. Редактор кодының терезесі Unit1.pas атымен аталады және жобалау формасының терезесінің артында орналасады. Код редакторы және форма жобалаушісі бір-бірімен тығыз байланысты. Редактор кодында бірнеше файл ашық тұруы мүмкін. Әрбір ашылған файл жеке бетте орналасады, ал оның атауы терезенің үстінгі бөлігінде көрсетеді.

Объект инспекторы. Объект инспекторы объектінің құрамын және формады орналасуын анықтайды, яғни ол формадаға объектіні өзгертуге қолданады. Сонымен қатар, форманың құрамын өзгерту үшін. Объект инспекторы екі беттен тұрады, оның әрқайсысын берілген компоненті анықтау үшін қолданады. Бірінші бет – бұл құрам тізімі, екіншісі – оқиға тізімі. Егер анықталған компонентпен байланыстарды өзгертуге керек болса, оны объект инспекторанда істей аласыз.

1.4 Object Pascal тілі

Object Pascal тілі Delphi бағдарламалау тілі және стандартты Pascal тілінің объекті-ориентирленген кеңейтілуі болып келеді. Delphi жүйесі визуалды бағдарламалау мүмкіндігін VCL визуалды компонеттердің библиотекасы көмегімен қамтамасыз етеді.

Визуалды компоненттердің библиотекасы (Visual Component Library, VCL) қосымшаларды тез өңдеуге арналған көптеген кластарды құрайды. Библиотеканың элементтері Object Pascal - да жазылған, сондықтан ол интегралданған Delphi қосымшасының өңдеу ортасымен тығыз байланысты. VCL де негізінен визуалды емес компененттерден тұрады, және визуалды компоненттер, және де TObject абстракті кластан бастап басқа да кластар бар. Барлық компонентер - бұлар кластар, бірақ барлық кластар компоненттер емес.

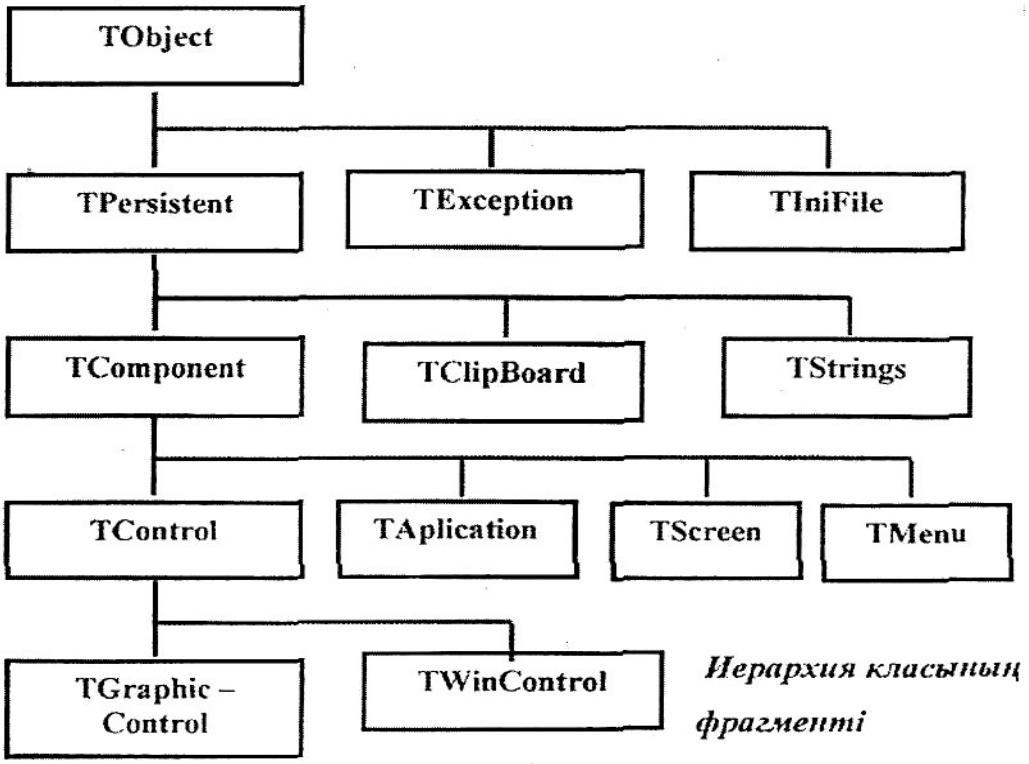
VCL-дің барлық кластары иерархияның анықталған сатысында орналасқан және кластың ағашын құрайды.

Object класы – барлық Object Pascal класының жалпы тегі (атасы) ол иерархияның тамырында (түбірінде) орналасады. Бұл класс абстрактілі болып келеді және барлық ұрпақтар кластары үшін жалпы әдістер қолданылады. Негізгі әдістерін атап өтейік:

– Create – объектіні құру;

– Destroy – объектіні жою (өшіру);

– Free – Create әдісімен құрылған объектіні өшіру, бұл уақытта Destroy әдісі де шақырылады.



1.2-сурет – Иерархия класының фрагменті

Бұл әдістердің көпшілігі ұрпақ класында қайта анықталады. TPersistents, TComponent, TControl көптеген кластардың атасы болып келген. Көрсетілген кластарға негізінде жалпы қасиеттері, әдістері және оқиғалары кіреді.

TPersistents класы ағымнан іске қосылады және ағымға сақталатын қасиеттері бар объектілер үшін абстракты болып келеді. Ағымның механизмі жадымен жұмыс істеу үшін қолданылады. TObject кластың әдісіне қосымша болатын TPersistents класы бір объектіден екінші объектіге алаңды және қасиеттерді беруге мүмкіндігі бар Assign әдісіне ие.

TComponent класы – барлық компоненттерге негіз; өзінің атасының әдісіне қосымша болып, оның басқа компоненттеріне ие болатын құралдарды қамтамасыз етеді. Формаға кез келген компонентті орналастыру нәтижесінде, ол басқа компонентке тиісті болады (негізінен формата). Компонента құру барысында, ол оған тиісті компоненттерді автоматты түрде құрылуын қамтамасыз етеді, ал оны жою барысын оған тиісті барлық компоненттер автоматты түрде жойылады [4].

TComponent класы келесі қасиеттерін белгіленген:

а) Components — тиісті компоненттер тізімі;

б) ComponentCount тиісті компоненттердің саны;

в) Componentlndex тиісті компоненттер тізіміндегі компонент номері;

г) ComponentState - ағынды компоненттің жағдайы;

д) Name - компоненттің аты;

е) Owner - компоненттің иесі;

ж) Tag - компонентпен сақталынатын бүтін мәні.

TComponent компонентінің кейбір әдістері:

а) Destroy - Components барлық тиісті компоненттерді жою (бұзу);

б) Destroying - тиісті компоненттің жойылғаны (бұзылғаны) туралы хабар беру;

в) FindCoraponent - Component тізімінен компонентті табу (іздеу).

Тілдің сөздігі. Сөздер келесі түрлерге бөлінеді:

а) Кілттік сөздер;

б) Стандатты идентификаторлар;

в) Қолданушының идентификаторлары.

Кілттік сөздер тілдің құрама бөлігі болып келеді. Редактор кодында кілттік сөздер қалың (полужирный) шрифтпен бөлінеді.

Мысалы:

And	Exports	mod
Array	File	nil
As	Finalization	not
Asm	For	object
Begin	Function	of
Case	Goto	or
Class	If	out
Const	Implementation	packed
Constructor	In	procedure
Destructor	Inherited	program
Dispinterfac	Initialization	property
Div	Inline	raise
Do	Interface	record
Downto	Is	repeat
Else	Label	resource string
End	Library	set
Except	Finally	shl

Стандартты идентификатор келесі анықталған тілдің өңдеуші конструкциясын көрсету үшін қызмет етеді:

- мәліметтердің типі;
- тұрақты (Константа);
- процедура және функция.

Стандартты идентификаторлары бағдарламада кейбір стандарттармен байланысқан.

Оларға:

Absolute	Name	safecall
Abstract	Near	stdcall
Assembler	Nodefault	strored
Automated	Override	virtual

Cdecl	Packade	write
Contains	Pascal	writeonly
Default	Private	
Dispid	Protected	
Dynamic	Public	
Export	Published	
External	Read	
Far	Readonly	
Forward	Register	
Implements	Reintroduce	
Index	Requires	
Message	Resident	

Қолданушының идентификаторы белгінің атын, тұрақты (константа), айнымалы, процедура, функция және мәліметтер типін белгілеу үшін қолданылады. Бұл аттар бағдарламашының (программист) өзі береді және ол келесі ережелерге сай келу керек: идентификатор әріп немесе санмен (цифр) құрылады; идентификатор барлық уақытта әріппен басталады, тек 0 – 9999 диапазонында орналасқан бүтін сан болатын белгі кірмейді; идентификаторда кіші және бас әріптерді қолдануға болады, компилятор оларды бірдей қабылдайды. Арнайы символдарды пайдалануға болмағандықтан оларды әдемілік үшін басты әріптермен жазуға болады. Мысалы, NumberLmes немесе btnOpen. Бағдарламада екі идентификаторлардың арасына ең кемінде бір бөлгіш болуы керек [5].

1.5 Delphi-де клиент-сервер архитектурасын құру

Көптеген қолданушысы бар үлкен мәліметтер базасы үшін клиент/сервер платформасындағы мәліметтер базасы көп қолданылады. Бұл жағдайда клиенттер тобы үшін мәліметтер базасын қолдану арнайы компьютер - сервер арқылы жүзеге асады. Клиент серверге тапсырма береді, іздеу операциясы немесе мәліметтер базасы жаңалау т.б. Осы күшті сервер, тапсырма операцияларына оптималды әдіспен жауап беруге бейімделген, оларды орындап және өзінің жұмысының нәтижесін хабарлап түрады.

Жұмыстың мұндай ұйымдасуы серверді қолданғанда қосымшаның орындалуының тиімділігін жоғарылатады және мәліметтердің бүтіндігін бақылайды.

Сонымен, «клиент-сервер» сәнді сөзінің артында не бар? Бүгінде әркім осындай жүйелердің біреуін қолданатын немесе дамытатын әсер қалдырады. Осы технологияның негіздерін үйренуге уақыт бөліп көріңіз: соңында Delphi 10 клиенттік-серверлік қосымшаларды жасау ортасы болып табылады. Дегенмен, бұл дегеніміз Delphi-мен жасалған барлық қосымшалар осы түрге жатады дегенді білдірмейді! Сонымен қатар, Oracle, Microsoft SQL немесе Interbase сияқты клиент-сервер дерекқорларында орналасқан деректерге кіретін бағдарламалар клиент-сервер түріне міндетті емес.

Бұл тарау клиенттік-серверлік жүйені құрайтын элементтермен жұмыс істейді және клиент-сервер бағдарламаларын және дәстүрлі жергілікті немесе әмбебап дерекқордың дамуын салыстырады. Клиенттің-сервердің шешімін пайдалану қажеттілігін негіздейтін оңайлатылған мысал келтірілген. Бұдан басқа, ол Delphi 10-да үш деңгейлі клиент-серверлік қосымшаны әзірлеу үдерісін талқылайды және клиенттік-серверлік қосымшаларды әзірлеуге көшкенде жергілікті дерекқор бағдарламаларын әзірлеушіге кедергі келтіретін кейбір тұзақтарды анықтайды.

Клиент-сервер мәліметтер базасында қосымша мәселе (проблема) шығады – проектілеу қосымшасы, ол максималды түрде сервер мүмкіндіктерін пайдаланып және минималды сетьті іске қосады және одан, яғни сетьтен тек минимум мәлімет алып тұру керек.

Көп деңгейлі дерекқордың қолданба архитектурасы көптеген қашықтағы клиенттердің серверлік сұрауларын өңдеу қажеттілігімен өмірге келтіріледі. Клиент-сервер қосымшалары осы тапсырманы оңай жеңе алатын сияқты.

Дегенмен, бұл жағдайда, көптеген клиенттермен, барлық есептеу жүктемесі күрделі іскерлік логиканы (сақталған процедуралар, триггерлер, көріністер және т.б.) іске асыру үшін құралдардың нашар жиынтығы бар дерекқор серверіне түседі. Ал әзірлеушілер клиенттің бағдарламалық жасақтамасының бағдарламалық жасақтамасының кодын айтарлықтай қиындатады, бұл қашықтағы клиенттік компьютерлердің үлкен саны болған кезде өте қажет емес. Шынында да, клиенттің бағдарламалық қамтамасыз етуінің күрделілігімен қателіктердің ықтималдығы жоғарылайды және оның күтімі күрделене түседі.

Көп деңгейлі қолданба дерекқорының архитектурасы аталған кемшіліктерді түзетуге арналған. Осылайша, осы архитектурада жұқа клиенттер – деректерді беруді, олардың жергілікті кэштелуін, пайдаланушы интерфейсі арқылы көрсетілуді, өңдеуді және қарапайым өңдеуді қамтамасыз ететін қарапайым бағдарламалар.

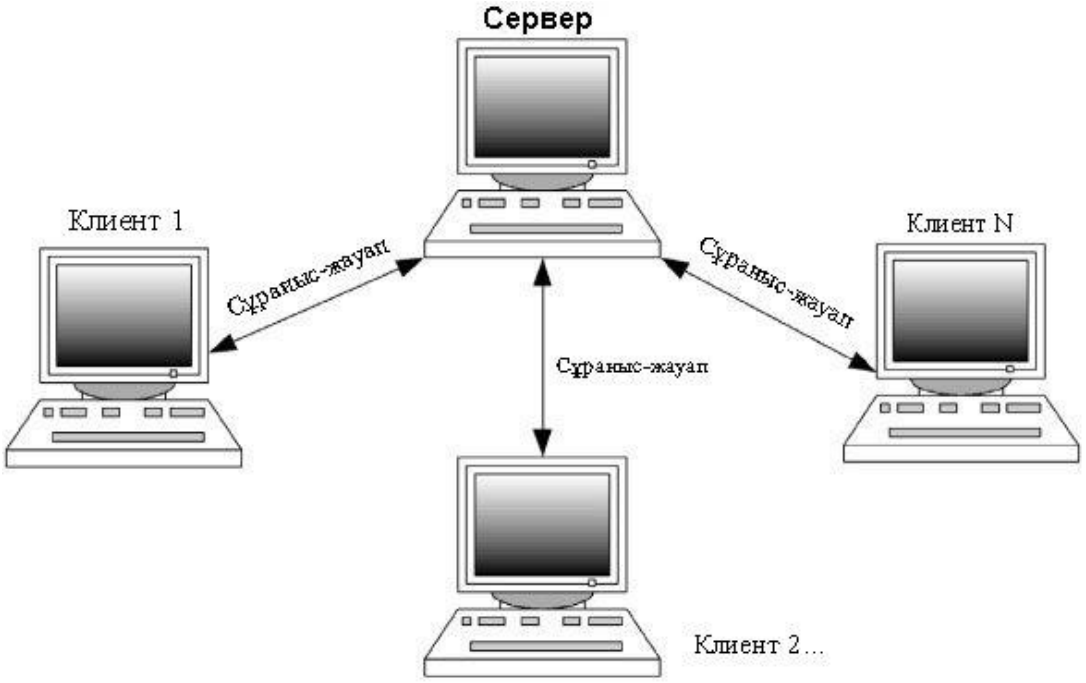
Клиенттік қосымшалар дерекқор серверіне тікелей емес, сонымен қатар мамандандырылған қосымшаларға кірмейді. Бұл бір сілтеме (қарапайым үш деңгейлі үлгі) және күрделі құрылым болуы мүмкін. Қосымшалар қосымшалар сервері деп аталады, клиенттік сұрауларды қабылдайды, оларды бизнес логикасының бағдарламаланған ережелеріне сәйкес өңдейді, қажет болса, оларды дерекқор серверіне ыңғайлы пішінге түрлендіреді және серверге жібереді. Дерекқор сервері алынған сұрауларды орындайды және нәтижелерді клиенттерге деректерді бағыттайтын бағдарлама серверіне жібереді.

Бұл жаңа желідегі мәліметтерді өңдеудің көп жақты жолы. Әдетте бұл әдіс (Delphi-де) мәліметтер базасын ұйымдастыру әдісі multi-tier – көпжіпті деп аталады. Бұл терминді жіп сөзі мәліметтерді бір ағыны ретінде қарастырылады. Қазіргі кезде көбіне үш ярусты түрі тараған:

– компьютердің ең соңғы деңгейінде клиенттер қосымшасы орналасқан, ол қолданушыға интерфейсті пайдалануға мүмкіндік береді;

– екінші деңгейде сервер қосымшасы орналасқан, ол мәліметтер базасымен қолданушы арасындағы мәліметтер алмасуын қамтамасыз етеді;

– үшінші деңгейде мәліметтер базасының алынған сервері орналасқан, ол сервер қосымшасынан мәліметтерді қабылдап, оларды басқарады;



1.3-сурет – Клиент-сервер архитектурасы

Бұл мәліметтер базасының қиын және майысқақ ұйымдасуы. Delphi - де негізінен осы жүйедегі алғашқы екі деңгейін құруды қамтамасыздандырадырылған. Сонымен қатар соңғы деңгейде қолданушының копьютеріне Borland DataBase Engine (BDE) орнату қажет етілмейтіндігін ескеру керек. Осымен мәліметтер базасының көпярустылығының бір қасиеті қорытындыланады. Клиент-сервер интернетке қосылған барлық компьютерлерді екі типке бөледі, олар: серверлер және клиенттер. Бір компьютерде серверді де және клиентті де орнату мағынасында бөлу онша қатаң жүргізілмейді. Жергілікті компьютерде Web-сервер орнатылуы мүмкін және осыған қарамастан, дәл осы компьютерде браузермен және почталық клиентпен де жұмыс жасауға болады. Басқа компьютерлерге анықталған сервис ұсынатын компьютерлерді серверлер деп атайды, ал осы сервисті қолданатындар – клиенттер.

Компьютерлерді серверлер мен клиенттер деп атағаннан гөрі, оларды бағдарламалық қамсыздандыру деңгейіндегі клиенттер немесе серверлер деп атаған жөн. Бір бағдарлама клиент есебінде, ал екіншісі сервер есебінде іске қосылатын қосымшалардың өзара байланысы клиент-сервер архитектурасы болады.

Сервердің басты тапсырмасы – сервиске қайсыбір клиент сұраныс жібермейінше әр кезде жұмыс жасап және күту жағдайында болу болып табылады.

Серверде сұраныстардың көптігінен оның жұмысы баяулап және белгілі бір сұраныстарға қызмет көрсетуді тежейді. Серверге сұраныс белгілі бір протокол шегінде болады – бұл желіде компьютерлер арасында байланысты қамтамасыз ететін стандарттар жиыны. Серверлік бағдарламалар клиенттік бағдарламаларға қызмет көрсету үшін компьютердің аппаратты ресурстарын қолданады. Клиент-бағдарлама сұраныс құрып, оны Желі арқылы белгілі бір адреске жібереді және алдын ала белгіленген протокол арқылы сервер-бағдарламамен өзара байланысады. Сол бір компьютерде бірнеше серверлік бағдарламалар орналаса алады. Клиенттік қосымша серверлік қосымша орналасқан компьютерде де, сонымен қатар, серверден керегінше жойылған компьютерде де орналаса алады, бірақ олар желімен байланысса, бұл айырмашылық тек уақыт бойынша жауаптың кідіруіне сәйкестеледі.

Серверлік бағдарлама әрқашан сұранысты орындауға дайын болу керек және сондықтан да сервер-бағдарлама жұмыс жасайтын компьютерлерге сенімділікке және өнімділігіне байланысты жоғары шарттар қойылады. Клиенттік компьютердің жұмысының тұрақтылығы бір адамның жұмысына әсер ететіндіктен, олардың жұмысына сенімділігіне байланысты аз талаптар қойылады, ал аппаратты сервердің жұмысының сенімділігіне байланысты көптеген клиенттердің жұмысының жүргізілуі тәуелді болады. Жоғарыда көрсетілген тәсіл (клиент-серверлік архитектура) дербес компьютердің қолданушысына миллиондаған серверлердің ресурстарына қол жеткізуге мүмкіндік береді.

1.6 Database Desktop қосымшасы мен SQL байланысы

Бұл бөлімде SQL серверімен жұмыс істеу кезінде қолданылатын негізгі әдістерді сипаттауға тырысамын. Осы тарауды жазып жатқанда мен негізінен MSSQL-тегі тәжірибемді қолданамын, бірақ бұл әдістер басқа SQL дерекқор серверлерімен жұмыс істеу үшін де қолайлы.

Қазіргі уақытта SQL тілінің өзіндік сипаттамалары көп, мен бағдарламаның (DELPHI-де жасалған) және дерекқор серверінің арасындағы өзара іс-қимыл механизміне көбірек назар аударамын. Клиенттің серверлік технологиясының өзі – функцияларды клиенттің бөлігіне бөлу – әдетте, деректерді салыстыру және жаңа деректерді енгізу. Және деректердің өзі, оларды өңдеу, сақтау және мұрағаттаумен айналысатын сервер.

Бұл жерде клиенттік бағдарламаның анықтамасы пайда болады – SQL тілінде сұрауларды жасайтын, оларды серверге жіберетін және осы сұрауды пайдаланушыға өңдеу нәтижелерін беретін бағдарлама.

Әдетте, дерекқорды арнайы программалық қамтамасыз етудің көмегімен бағдарламашы өзі жасайды. Бірақ бағдарламаға қатысты деректермен жұмыс.

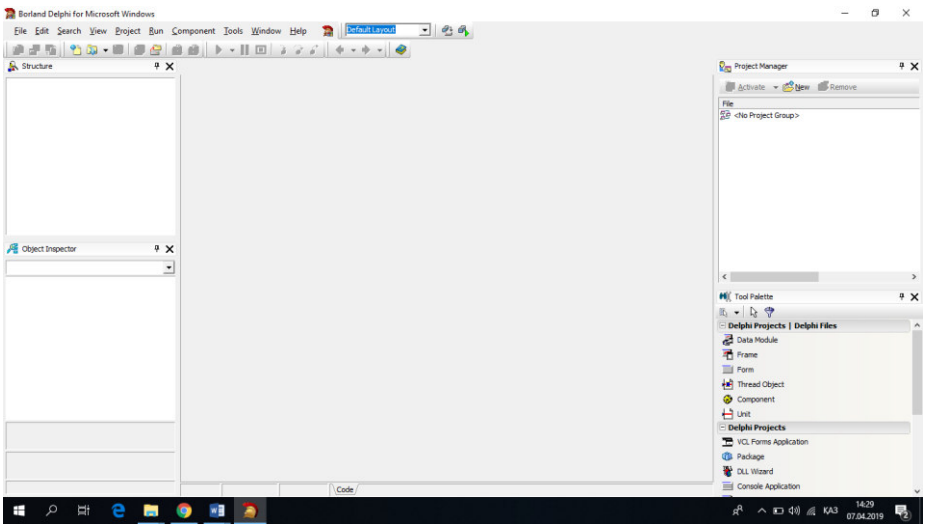
Біріншіден, бағдарлама жаңадан құрылған дерекқорда жасалуы тиіс, ол деректерді дерекқормен толтыруға мүмкіндік береді. Содан кейін бағдарлама пайдаланушыға дерекқорға енгізілген деректерді көруге мүмкіндік беруі керек.

Егер пайдаланушы бірдеңені ұнатпаса, деректерді өзгертіңіз Delphi-де Database Desktop қосымшасы бар және ол арқылы мәліметтер қорын DBase немесе Paradox форматында құрып оны өзгертуге болады және SQL-сұранымдарды орындауға болады. BLOB-алаңдарынан басқасын айтылған қосымшалардың барлық мәліметтер қорының алаңдарын өзгертуге болады .

Database Desktop программасын жүктеу үшін Delphi программасының тобының ішінен Database Desktop элементтерін таңдап, содан кейін DatabaseDesktop экранда пайда болады. Бірінші рет іске қосқанда Database Desktop қосымшасының жұмыс каталогы көрсетіледі. Ол үшін File/Working directory және File / Privite directory қосымшаларын орындаймыз.

Жаңа кесте құру үшін File менюінен New командасын орындап, сосын Table командасын орындасақ нәтижесінде экранға Create Table диалогтық терезесі шығады. Ашылған терезеден кесте типін көрсетеміз де OK кнопкасын басқаннан кейін Create Table терезесі пайда болады. Бұл диалогтық терезеде кесте құрылымын анықтаймыз. Field Name облысында кесте алаңдарының атын береміз, дәлірек айтсақ кесте мәнін Field Name, type, size және кесте бағандарын толтырамыз. Table properties облысында индекс мәнін және кесте түрі драйверлерін таңдауға қолданылады.

Жаңа кесте құрылып сақтау үшін Save As командасын таңдаймыз. Толтырылмаған кестені енгізу үшін File (open)Table командасын орындаймыз және диалогтық OpenTable терезесінен кесте атын таңдаймыз. Енді кестеге мән енгізу үшін саймандар тақтасынан Edit Data кнопкасын басып енгізу режимін активтендіреміз.



1.4-сурет – Database Desktop қосымшасының негізгі терезесі

Айта кетер жағдай Delphi-де мәліметтер қорымен қорымен жұмыс орындаудың негізі Borland Database Engine (BDE)-да құрылады. BDE

мәліметтер қоры мен қосымша арасында делдал қызметін атқарады. Ол мәліметтер қорын реализациялауда қолданушыға мүмкіндік береді. Соған сәйкес МҚ реализациясын өзгерткенде қосымшаны да өзгертуге болады. Delphi-де құрастырылған қосымша ешкашан да мәліметтер қорымен тікелей жұмыс істемей, BDE арқылы ғана қолданыста болады. Жүйе мәліметтер қорымен байланыс орнатқан кезде BDE-ге барады да, мәліметтер қорының лақап атын және ондағы керек кестені хабарлайды. BDE кітапханалары басқа да динамикалық тіркестелген кітапханалар секілді DLE түрінде орналасқан.

Олар басқа кітапханалар секілді IDAPI (Integrated Database Application Program Interface) деп аталған API-мен (Application Program Interface) қолданбалы бағдарламалар интерфейсімен қамтамасыз етілген. BDE SQL сұраныстарының стандарт тілін қабылдайды. Ол syBase,MySQL, Oracle, InterBase сияқты SQL серверімен мәлімет алмасуға мумкіндік береді.Осы мүмкіндік клиент-сервер қосымшасымен жұмыс істеу кезінде кеңінен қолданыста болады.

Қолданбадан дерекқорға сұраныстардың мүмкіндіктерін қарастырыңыз. Сұраныстардың үш түрі бар:

– статикалық сұраныс – сұрау мәтіні бағдарламаны әзірлеу сатысында толығымен қалыптасады;

– параметрлік сұраныс – сұрау мәтіні бағдарламаны әзірлеу сатысында қалыптасады және бірнеше параметрді қамтиды, бағдарламаны орындау уақытында параметр мәндерін орнатуға болады;

– динамикалық сұраныс – сұрау мәтіні бағдарламаны орындау сатысында толығымен қалыптасады.

BDE технологиясын пайдаланған кезде TQuery компоненті сұрауларды орындау үшін пайдаланылады. ADO технологиясын пайдаланған кезде сұрауларды орындау үшін келесі компоненттерді пайдалануға болады:

– TADOQuery компоненті - TQuery компонентінің аналогы. Бұл ең жан-жақты. SELECT, INSERT, UPDATE, DELETE, CREATE TABLE, EXECUTE және басқа да мәлімдемелерді қамтитын кез келген SQL сұрауларын орындауға мүмкіндік береді Сұрау мәтіні SQL сипатына орналастырылады. Деректер жиынтығын қайтаратын сұрауды орындауға мүмкіндік беретін логикалық түрінің Белсенді сипатын қамтиды (SELECT мәлімдемесі). ExecSQL әдісі деректер жиынын қайтармайтын сұрауды орындауға мүмкіндік береді;

– TADODataSet компоненті - SELECT операторы арқылы үлгі сұрауларын ұйымдастырудың арнайы құралы. Компонент деректер жиындарын қайтармайтын мәлімдемелерді орындауға мүмкіндік бермейді, мысалы: INSERT, UPDATE, DELETE, CREATE TABLE және т.б. Үлгі сұраулардың құрылысын жеңілдететін ыңғайлы редактор бар. Редакторға қоңырау шалу үшін CommandText сипатын пайдаланыңыз. Сұрауды орындау үшін Белсенді сипатты пайдаланыңыз;

– TADOCommand құрамдасы: INSERT, UPDATE, DELETE, CREATE TABLE және т.б. сияқты деректер жиынын қайтармайтын сұрауларды

ұйымдастырудың арнайы компоненті. Компонент SELECT операторына рұқсат бермейді. Execute әдісі сұрауды орындауға мүмкіндік береді.

Содан кейін ADO технологиясы негізінде деректер базасының қосымшаларынан сұраныстарды ұйымдастыру тәртібін қарастырамыз.

Деректер қорымен жұмыс істеуге арналған Delphi компоненттері SQL және клиент-сервер архитектурасымен жұмыс істеуге арналған. Олармен жұмыс істеген кезде қашықтағы серверлердің озық қолдау мүмкіндіктерін пайдалана аласыз. Delphi бұл қолдауды екі жолмен ұсынады. Біріншіден, Delphi тікелей командалары әзірлеушіге кестелерді басқаруға, шектеулер қоюға, жоюға, бар жазбаларды кірістіруге және өңдеуге мүмкіндік береді. Екінші әдіс – SQL тілінде сұрауларды сұрау, оны талдау, оңтайландыру, нәтижелерді қайтару және беру үшін серверге жіберілетін сұрау.

ADO технологиясы (ActiveX Data Objects) дерекқорға BDE технологиясына қарағанда қарапайым қосылуды қамтамасыз етеді. Дерекқорға қосылу үшін BDE технологиясындағы сияқты бүркеншік аттарды жасаудың қажеті жоқ. TADOConnection құрамдас бөлігімен қосылымды дұрыс конфигурациялау жеткілікті.

TADOConnection компоненті келесі негізгі қасиеттерді қамтиды:

– ConnectionString - дерекқорға қосылу параметрлерін орнатуға мүмкіндік береді. Параметрлерді орнатудың ыңғайлығы үшін арнайы шеберді пайдалануға болады;

– Connected - мән шын мәніне орнатылғанда дерекқордың қосылымын іске қосады;

– LoginPrompt. Егер «false» мәніне орнатылса, дерекқорға қосылу пайдаланушы атын және паролін енгізуді сұрайтын диалогтық терезені көрсетпейді. Аты мен құпия сөз ConnectionString сипатынан алынады. Егер шын мәніне орнатылса, пайдаланушы аты мен құпиясөз диалогтық терезеде әрқашан сұралады.

Дерекқорды жеткізу жүйесімен байланыс орнатуды қалайсыз ба? TADOConnection компоненті арқылы қосылымды орнатыңыз. Компонентті екі рет басу арқылы қосылым шеберіне өтіңіз. Ашылған терезеде Connection String радиосын пайдалану түймесін таңдап, Build (Құрастыру) түймешігін басыңыз (Сіз деректерді сілтеме файлының радио түймесін пайдалану арқылы арнайы * .udl файлынан байланыс параметрлерін жүктей аласыз).

Одан кейін, Деректерді жеткізуші қойындысындағы тілқатысу терезесінде, дерекқордың провайдер түрін таңдаңыз, біздің жағдайда SQL Server үшін Microsoft OLE DB жеткізушісі (1-сурет). Қосылым қойындысында дерекқордың қосылу параметрлерін орнатамыз: SQL серверінің мекенжайы, пайдаланушы аты және құпия сөзі, дерекқор атауы (2-сурет). Параметрлерді орнатқаннан кейін OK түймешігін басыңыз.

TADOQuery, TADODataSet және TADOCommand компоненттері TADOConnection компонентіне қосылатын Connection сипатын қамтиды.

2 Жобалау бөлімі

2.1 Мәліметтер қорын жобалау

Мәліметтер қорымен жұмыс істейтін ақпараттық жүйелердегі жалпыға арналған программалық қамтамасыз ету мәліметтер қорын басқару жүйесі (МҚБЖ) деп аталады. Мәліметтер қорын басқару жүйелері – бұл мәліметтер қорын құруға, енгізуге және өңдеуден өткізуге арналған тілдік және программалық құралдардың кешені.

Әлемде МҚБЖ-ның 1000-нан астам түрі бар. Мәліметтер қорын басқарудың әр алуан жүйелері мәліметтердің әр түрлі модельдерін пайдаланады. Мәліметтер қорының әр түрлі модельдеріне негізделген МҚБЖ-лардың мысалдарына төмендегілер жатады:

а) иерархиялық модель – IMS PC/Focus, Team-UP, Data Edge;

б) желілік модель – IDMS, db-Vista III;

в) реляциялық модель – Dbase, DB2(IBM), FoxBase және FoxPro (Fox Software), Paradox, Dbase for Windows жәнеInterBase (Borland), Visual FoxPro және Access (Microsoft), Clarion (Clarion Software), Ingress (ASK Computer Systems), Oracle (Oracle), Informix;

г) постреляциялық модель – Bubba, DasDb;

д) көпөлшемді модель – EssBase (Ardor Software), Media Multi-matrix (Speedware), Oracle Express Server (Oracle), Cache (InterSystems);

е) объектілі бағытталған модель – POET(POET Sofyware),Jasmine (Computer Associates), Versant (Versant Technologies), 02 (Ardent software), Iris, Orion, Postgress,ODB-Jupiter [6].

Мәліметтер қорының сервері ретінде келесі МҚБЖ-ларболуы мүмкін: NetWare SQL (Novell), MS SQL Server (Microsoft), InterBase (Borland), SQLBase Serve (Gupta), Intelligent DataBase (Ingress), Oracle, Informix көптеген пайдаланушыларға арналған МҚБЖ болып табылады, ол текті емес ортада жұмыс істейді (әртүрлі ЭЕМ мен ЖЖ).

МҚБЖ-ның негізгі қызметіне келесілер жатады:

- мәліметтерді құру және бақылау;
- мәліметтерді қорғау және бақылау;
- көптеген пайдаланушылардың мәліметтерге қол жеткізуі;
- пайдаланушыға мәліметтерді манипуляциялау мүмкіндігін ұсыну;
- қолданбалы программаларды құру мүмкіндігі.

Мәліметтерді құру мен бақылау мәліметтер сөздігінің көмегімен орындалады. Мәліметтердің сөздік/каталогы мәліметтердің құрылымы, мәліметтердің типтері, оларды ұсыну форматтары, мәліметтердің өзара байланысу сұлбалары, пайдаланушылар, мәліметтерді қорғау мен оларға қол жеткізу кодтары туралы ақпаратты орталықтандырылған түрде сақтауға арналған [7]. Көбінесе сөздіктің қызметін МҚБЖ-лар орындайды және олар жүйелік мәзірден шақыртылады немесе утилиттері арқылы орындалады.

Мысалы, Orion атты оқу мәліметтер қорының келесі түрдегі сөздігі болуы мүмкін:

SYSTABLES	NAMECREATORCOLCOUNT		
	Salespeople	AGU	4
	Customers	AGU	3
	Orders	AGU	5
SYSCOLUMNS	NAMETBNAME COLTYPE		
	Snum	Salespeople	Integer
	Sname	Salespeople	Char
	City	Salespeople	Char
	Manager	Salespeople	Integer
	Cnum	Customers	Integer
	Cname	Customers	Char

2.1-сурет – Orion операторының сөздік кестесі

Біз ақпараттық жүйенің ерекшеліктерін: заттар облысы туралы пайдаланушылардың әртүрлі көзқарасын қолдауды белгілеп өтеміз. Қазіргі заманда ақпараттар жүйесі бірнеше көрініс деңгейлерін қолдайды. Олардың түрлерін ақпараттар жүйесі архитетурасы түсінігімен қабылданған.

Дерекқорды жобалау - дерекқор жобасын жасау үдерісі. экономикалық объектінің жұмыс істеуін қамтамасыз етуге бағытталған және оның мақсаттарына қол жеткізуге ықпал етеді. Бұл өте қиын процесс. бірлескен сарапшының күш-жігерін талап етеді. дизайнерлер мен пайдаланушылар. Деректер базасын жобалау кезінде фактіні қарастырған жөн. дерекқор күрделі талаптарға сәйкес болуы керек. Бұл талаптар келесідей:

а) дерекқордың тұтастығы - деректердің толықтығы мен дәйектілігі туралы талап;

б) деректерді қайта пайдалану;

в) пайдаланушы сұраулары туралы ақпаратты жылдам іздеу және алу;

г) деректерді жаңартудың қарапайымдылығы;

д) шамадан тыс деректердің артық болуын төмендету;

е) деректерді рұқсатсыз кіруден қорғау. бұрмалау және жою.

Дерекқордың өмірлік циклі (WDCB) - бұл дизайн процесі, деректер базасын енгізу және қолдау. WCRD жеті кезеңнен тұрады:

а) алдын-ала жоспарлау;

б) техникалық-экономикалық негіздемені тексеру;

в) талаптарды анықтау;

г) тұжырымдамалық жобалау;

д) логикалық дизайн;

е) физикалық дизайн;

ж) өнімділікті бағалау және деректер базасын қолдау.

Біз әрбір кезеңнің негізгі міндеттерін сипаттаймыз.

Кез-келген экономикалық нысанды дамытуда бұл іске бір сәтте келеді. даму деректеріне одан әрі жету үшін ақпарат қажет. қызметкерлерге тиесілі. дерекқорға ортақ пайдалануды біріктіру және оларды корпоративтік ресурс ретінде қабылдау.

а) дерекқорды алдын-ала өңдеу – бұл шашыраңқыдан интеграцияланған деректерге көшу процесінің маңызды кезеңі. Бұл кезеңде ақпарат жиналады. Файлдарды өңдеу процесінде қолданылатын қолданбалы бағдарламалар олармен байланысты. Ол қолданыстағы бағдарламалар арасындағы байланыстарды орнатуға көмектеседі. олардың ақпараты қалай пайдаланылатындығын көрсетеді. Сонымен қатар. болашақ дерекқордың талаптарын анықтауға мүмкіндік береді. Ақпарат жалпыланған тұжырымдамалық деректер моделі түрінде ресімделеді;

б) техникалық-экономикалық негіздеме үш мәселе бойынша есептерді дайындауды қамтиды:

1) жоспарланған деректер қорын (технологиялық-техникалық) жүзеге асыру үшін қажетті жабдықтар мен бағдарламалық қамтамасыз ету бар ма?;

2) персонал бар ма. деректер қорын құру жоспарын (операциялық-техникалық) табысты іске асыру үшін қаражат және сарапшылар;

3) жоспарланған деректер қоры төленетінін (экономикалық тиімділік) тиімділік;

4) талаптарды анықтау (деректер қорының мақсаттары, әртүрлі құрылымдық бөлімшелер мен олардың басшыларының ақпараттық қажеттіліктері, жабдықтарға қойылатын талаптар, бағдарламалық жасақтама талаптары).

в) тұжырымдамалық дизайн. Бұл кезеңде доменнің қолданушы деректерінің егжей-тегжейлі үлгілері жасалады. Сонда олар тұжырымдамалық үлгіде біріктіреді. ол корпоративтік деректердің барлық элементтерін жинақтайды. дерекқорға жүктелетін болады. Бұл модель тұжырымдамалық дерекқор схемасы деп те аталады;

г) логикалық дизайн. Бұл кезеңде деректер үлгісінің түрін таңдау. Тұжырымдамалық модель логикалық модельге сәйкес келеді. қазірдің өзінде негізделген құрылымдар. таңдалған модельге тән;

д) физикалық дизайн. Бұл кезеңде логикалық модель сипаттамалары бойынша кеңейтіледі. дерекқордың физикалық сақтауын анықтау үшін қажет. сақтау құрылғыларының түрі. дерекқор деректеріне қол жеткізу әдістері. қажетті жады. деректер базасын сақтау ережелері және т.б.

Тұжырымдамалық жобалау сатысында доменді модельдеу құралы – субъект-қатынас моделі. Жиі оны ER-моделі деп атайды. Онда деректер

доменінің құрылымын модельдеу графикалық құралдарды – ER диаграммаларын («байланыс-байланыс-диаграммалар») пайдалануға негізделген. Көрнекі түрде олар субъектілер арасындағы қарым-қатынасты көрсетеді.

Ұйым нақты объект болып табылады, олар дербес болуы мүмкін. Ұйымның атрибут мәндерінде бір-бірінен ерекшеленеді және бір мәнді сәйкестендіруге мүмкіндік береді.

Атрибут – бұл ұйымның сипаты. Мысалы, Кітаптың мәні осындай атрибуттармен сипатталады. автор ретінде. атауы. баға баспа үйі. айналымы беттер саны. Арнайы кітаптар BOOK субъектісінің даналары болып табылады. Олар көрсетілген төлсипаттардың мәндерімен ерекшеленеді және «ат» төлсипаты арқылы бірегей түрде анықталады.

Атрибут. ол ұйымның мысалдарын бірегей түрде анықтайды. кілтті шақырды. Күрделі кілт болуы мүмкін. бірнеше атрибуттардың комбинациясын білдіретін.

2.2 Деректер қорын жобалау рәсімдері

Тұжырымдамалық жобалау рәсімдері. Концептуалды дизайн фазасының мақсаты - субъектінің субъектісі туралы пайдаланушылардың қабылдауына негізделген тұжырымдамалық деректер үлгісін жасау. Оған қол жеткізу үшін бірқатар дәйекті рәсімдер орындалады.

а) субъектілерді анықтау және оларды құжаттау. Нысандар объектілерді анықтау үшін анықталған. олар өзгелерден тәуелсіз. Мұндай объектілер - субъектілер. Әрбір субъектке мағыналы ат тағайындалады. пайдаланушыларға түсінікті. Ұйымдардың есімдері мен сипаттамалары деректер сөздігіне енгізіледі. Егер мүмкін болса. Бұл әрбір ұйымның күтілетін санын көрсетеді;

б) медицуд субъектілерімен қарым-қатынастарды анықтау және оларды құжаттау. Тек субъектілер арасындағы байланыстар анықталған. дерекқордың дизайны талаптарына сай болу үшін қажет. Олардың әрқайсысының түрін белгілейді. Кәсіпорындардың меншік класын анықтайды. Қатынастарға мағыналы атаулар беріледі. етістіктермен көрсетілген. Әр қосылыстың түрін және субъектілер құрамын көрсете отырып, кеңейтілген сипаттамасы. байланысқа тартылған. деректер сөздігін енгізді;

в) тақырыптық аймақтың ЕО үлгісін құру. ЕС диаграммалары олардың арасындағы байланыстарды және олардың арасындағы байланыстарды бейнелеу үшін пайдаланылады. Олардың негізінде модельдендірілген тақырыптық аймақтың бірыңғай визуалды көрінісі - тақырыптық аймақтың ЕС моделі жасалады;

г) атрибуттарды және олардың құжаттамасын анықтау. Барлық атрибуттар анықталды. құрылған ЭК-моделінің мәнін сипаттайтын. Әрбір төлсипатқа мағыналы ат тағайындалады. пайдаланушыларға түсінікті. Деректер сөздігі сөзіндегі әрбір төлсипат туралы келесі ақпаратты орналастырды:

1) төлсипат атауы мен сипаттамасы;

2) мәннің түрі мен өлшемі;

3) мән. әдепкі төлсипат үшін қабылданады.

Төлсипаттың типі:

– атрибуттың құрамдас бөлігі. және солай болса. онда ол қандай қарапайым атрибуттар тұрады. Мысалы. «Клиенттің аты» атрибуты «Тегі» қарапайым атрибуттарынан тұруы мүмкін. «Аты». «Әкесінің аты». немесе қарапайым болуы мүмкін. жалпы мәндері бар. қандай да бір түрде «Мухатай Меиржан Биримжанович». Егер пайдаланушыға «Толық аты» жеке элементтеріне кіру қажет болмаса. Бұл атрибут қарапайым болып табылады:

– атрибут есептелген болып табыла ма. және солай болса. оның мәндері қалай есептеледі.

д) атрибут мәндерін және олардың құжаттамасын анықтау. Әрбір жеке төлсипат үшін. ЕО моделіне қатысады. жарамды мәндердің жиынтығы анықталады және атау тағайындалады. Мысалы. «Тіркелгі түрі» атрибуты «депозит» мәніне ғана ие болады. «ағымдағы», «сұраныс бойынша», «карт-шот». Деректердің сөздік жазбалары жаңартылды. атрибуттарға қатысты. - олар атрибут мәндерінің жиынынан тұрады;

е) субъектілер үшін бастапқы кілттерді және олардың құжаттамасын анықтау. Бұл қадамда біз бастапқы кілт анықтамасын - субъектінің атрибуттары немесе атрибуттары ретінде басшылыққа аламыз. оның даналарын бірегейлендіруге мүмкіндік береді. Бірінші кілттер туралы ақпарат деректер сөздігінде орналастырылады;

ж) тұжырымдамалық модельдерді соңғы пайдаланушылармен талқылау. Тұжырымдамалық деректер моделі ER үлгісімен ілеспе құжаттармен ұсынылған. дамыған деректер үлгісінің сипаттамасы бар. Егер пәндік салада сәйкессіздіктер болса. онда сол уақытқа дейін модельге өзгерістер енгізіледі. пайдаланушылар растағанша. ұсынылған модель олардың жеке көзқарастарын барабар көрсетеді.

Логикалық жобалау рәсімдері. Логикалық дизайн кезеңінің мақсаты - деректер базасын физикалық іске асыру үшін келесі ДҚБЖ-да пайдаланылатын ерекшеліктерге қарамастан таңдалған деректер моделіне негізделген тұжырымдамалық модельді логикалық моделге айналдыру. Оған қол жеткізу үшін келесі рәсімдер орындалады.

а) деректер үлгісін таңдау. Көбінесе, реляциялық деректер моделі деректерді кестелік көрсетілімнің көрінуіне және олардың жетілуіне байланысты таңдалады;

б) ER-моделіне негізделген кестелер жиынтығын және олардың құжаттамасын анықтау. Әрбір ЭК-үлгі субъектісі үшін кесте жасалады. Ұйым атауы - кестенің аты. Кестелердің құрылымы 1.4-тармақта баяндалған ережелер негізінде құрылады. Алғашқы және шетелдік кілттер механизмі арқылы қатынастар кестелер арасында белгіленеді. Кесте құрылымдары мен олардың арасында орнатылған қатынастар құжатталған;

в) кестелерді қалыпқа келтіру. Нормализацияны дұрыс орындау үшін дизайнер деректерді пайдаланудың семантикасы мен ерекшеліктерін терең зерттеуі керек. Бұл қадамда ол кесте құрылымының дұрыстығын тексереді. алдыңғы қадамда жасалған. оларға қалыпқа келтіру рәсімін қолдану арқылы. Бұл рәсім 1.5-тармақта сипатталған. Ол әрбір кестені келтіреді. кем дегенде. ZNF-ке. Нормалдау өте икемді дерекқор жобасында нәтиже береді. қажетті кеңейтулерді оңай жасауға мүмкіндік береді;

г) пайдаланушылар ұсынатын барлық транзакцияларды орындау мүмкіндігі үшін логикалық деректер үлгісін тексеріңіз. Мәміле - бұл әрекеттер жиынтығы. дерекқордың мазмұнын өзгерту үшін бір пайдаланушы немесе бағдарлама арқылы орындалады. Мәселен BANK жобасында транзакцияның мысалы, белгілі бір клиенттің шоттарын басқа клиентке басқару құқығын беру болуы мүмкін. Бұл жағдайда дерекқор бірден бірнеше өзгерісті жасауы керек. Егер транзакция кезінде компьютердің құлауы орын алса. онда дерекқор сәйкессіз күйде болады. өйткені кейбір өзгерістер енгізілді. ал қалғандары әлі. Осылайша, дерекқорды бұрынғы дәйекті күйіне қайтару үшін барлық ішінара өзгерістерді қалпына келтіру керек. Мәмілелердің тізімі субъектінің қолданушылары әрекеттерімен анықталады. ЕС моделін пайдалану. деректер сөздігі және негізгі және шетелдік кілттер арасындағы байланыстар орнатылған. барлық деректерге қол жеткізу әрекеттерін қолмен орындауға әрекет жасалды. Кез келген операция орындалса, қолмен орындалмайды. онда құрастырылған логикалық деректер моделі жеткіліксіз және қателер бар. ол жойылуы керек. Мүмкіндігінше. олар субъект үлгісіндегі өтуге байланысты. сілтеме немесе атрибут;

д) деректердің тұтастығын қолдау талаптарын және олардың құжаттамасын анықтау. Бұл талаптар шектеулер. олар дерекқорда қайшылықты деректердің алдын-алу үшін енгізілген. Бұл қадамда деректердің тұтастығы мәселелері оны жүзеге асырудың нақты аспектілеріне қарамастан орындалады. Төмендегі шектеулерді қарастырған жөн:

1) қажетті деректер. Көрсетіледі. кез-келген атрибуттар бар ма? олар n-мәндерден тұруы мүмкін емес;

2) төлсипат мәндері бойынша шектеулер. Төлсипаттар үшін жарамды мәндер анықталады;

3) объектінің тұтастығы. Қол жеткізілді, егер ұйымның бастапқы кілтінде QGII-мәндері болмаса;

4) сілтеме тұтастығы. Ол мұны түсінді, шетел кілтінің мәні міндетті түрде кестенің жолдарының бірінің бастапқы кілтінде болуы керек ата-ана ұйымы;

5) шектеулер іскерлік ережелермен белгіленген. Мысалы, банк жобасы бойынша жағдайда ереже қабылдануы мүмкін, клиенттің билік етуіне тыйым салады. Айталық үшден астам шоттар.

Деректер сөздігінде барлық бекітілген деректер тұтастығы шектеулері туралы ақпарат орналастырылған;

е) логикалық деректер моделінің түпкілікті баяны жасау және оны пайдаланушылармен талқылау. Осы қадамда логикалық деректер моделін білдіретін, ЕС үлгісінің соңғы нұсқасы дайындалды. Модель және құжаттама өзі. соның ішінде деректер сөздігі және реляциялық кестені байланыстыру схемасы пайдаланушылардың қарауына және талдауына ұсынылған. ол сенімді болуы керек. ол тақырыпты дұрыс көрсететінін білдіреді.

Физикалық дизайн рәсімдері. Физикалық дизайн фазасының мақсаты компьютердің сыртқы жадында орналасқан дерекқордың нақты іске асырылуын мақсатты ету болып табылады. Деректерді сақтау құрылымының сипаттамасы және дерекқор деректеріне қол жеткізудің тиімді әдістері. Логикалық дизайн жағдайында, сұраққа жауап «не істеу керек, ал физикалық жағдайда әдіс таңдалады. бұны қалай істейді. Физикалық дизайн рәсімдері келесідей.

а) таңдалған ДҚБЖ көмегімен дерекқор кестелерін жобалау. Реляциялық ДҚБЖ таңдау жүзеге асырылады, ол деректер базасын құру үшін пайдаланылатын болады. машиналық тасымалдаушыларға орналастырылған. Кестелерді жобалаудағы функционалдылығы терең зерттеледі. Содан кейін кестелер ДББЖ ортасында байланыс схемаларын әзірлейді. Деректер базасының дайындалғаны ілеспе құжаттамада сипатталған;

б) таңдалған ДҚБЖ ортасында іскерлік ережелерді енгізу. Кестелердегі ақпаратты жаңарту бизнес ережелерімен шектелуі мүмкін. Оларды жүзеге асыру әдісі таңдалған ДҚБЖ-ға байланысты. Домен талаптарын іске асырудың кейбір жүйелеріне көп мүмкіндіктер ұсынылады, басқалары – menype. Кейбір жүйелерде іскерлік ережелерді іске асыруға ешқандай қолдау жоқ. Бұл жағдайда олардың шектеулерін жүзеге асыру үшін қосымшалар әзірленеді. Барлық шешімдер осы саладағы іскерлік ережелерді іске асыруға байланысты қабылданған. ілеспе құжаттамада егжей-тегжейлі сипатталған;

в) дерекқорды ұйымдастыруды жобалау. Бұл қадамда кестелер үшін ең жақсы файлдық ұйым таңдалады. Анықталған операциялар. ол жобаланған дерекқорда орындалатын болады. және олардың ең маңыздысын ерекшелендіреді. Транзакция өткізу қабілеттілігі талданады – мәмілелер саны. ол белгілі бір уақыт интервалында өңделуі мүмкін. жауап беру уақыты - бұл уақыт аралығы, бір мәмілені аяқтау үшін талап етіледі. Мәміле өткізу қабілетін арттыру және жауап уақытын азайту. Осы көрсеткіштерге сүйене отырып, кестелердегі көрсеткіштерді анықтау арқылы дерекқордың жұмысын оңтайландыру бойынша шешімдер қабылданады. дерекқордан деректерді таңдауды жеделдету немесе кестелерді қалыпқа келтіру деңгейіне қойылатын талаптарды азайтады. Дискіні сақтауды бағалау жүргізілуде. жасалатын дерекқорды орналастыру үшін қажет. Оны азайту үшін ұмтылыңыз. Анықталған мәселелер бойынша қабылданған шешімдер құжатталған;

г) стратегия дерекқорының дерекқорын жасау. Дерекқор - құнды корпоративтік ресурс, ал оны қорғауды ұйымдастыру үлкен көңіл бөледі. Ол үшін дизайнерлер барлық құралдардың толық және нақты түсінуіне ие болуы керек. Таңдалған ДҚБЖ ұсынған;

д) дерекқор жұмысының мониторингін ұйымдастыру және оны орнату. Деректер базасының жобасын жасағаннан кейін оның жұмысын үздіксіз бақылау жүргізіледі. Деректер базасының орындалу деңгейі туралы алынған ақпарат оны теңшеу үшін пайдаланылады. Бұл үшін таңдалған ДҚБЖ құралдары де қатысады.

Операциялық дерекқорға кез-келген өзгерістер енгізу туралы шешімдер қасақана және мұқият өлшенуі керек.

Ақпараттық жүйенің құрылымын ішкі жүйелер деп аталатын жекеленген бөліктер құрайды. Қолдану саласына тәуелсіз ақпараттық жүйенің жалпы құрылымын ішкі жүйенің жиынтығы ретінде қарастыруға болады. Бұл жағдайда классификацияның құрылымдық белгісін қарастырып, ал ішкі жүйелерді қамсыздандырушы ішкі жүйелер деп аталады. Жүйенің жалпы құрылымын мынадай қамсыздандырушы ішкі жүйелер құрайды: ақпараттық, техникалық, математикалық, программалық, ұйымдастырушылық және құқықтық.Ақпараттық қамсыздандыру ішкі жүйесінің негізгі міндеті басқарудағы шешімдерді қабылдау үшін шынайы ақпаратты дер кезінде ұсыну. Көптеген ұйымдар жұмысын талдау кезінде ондағы құжаттарды жүргізу ісінде бірқатар кемшіліктер байқалды: қолмен өңдеуге арналған құжаттың шамадан тыс үлкен көлемі; бірдей көрсеткіштердің түрлі құжаттарда қайталануы; құжаттың үлкен көлемімен жұмыс істеу мамандардың негізгі жұмысын орындауына кедергі болуы; жасалған көрсеткіштің қолданылуы; Аталған кемшіліктерді жою ақпараттық қамсыздандыру алдында тұрған мәселелердің бірі болып табылады. Ақпарат ағымдарының схемасын жасау ақпараттың қозғалыс маршруттарын сипаттайды. Осындай схемаларды талдау нәтижесінде басқару жүйесін толығымен жетілдірудің іс шараларын жасап шығаруға болады. Ақпараттың көлемін анықтау және оны бөлшектеп талдау мүмкіндіктерін беретін ақпараттық ағымдардың схемалары қайталанатын және қолданылмайтын ақпаратты жоюды ақпаратты тиімді түрде сипаттау және жіктеуді қамсыздандырады. Сонымен қатар ақпараттың басқару деңгейлеріне сәйкес өзара байланыс қозғалысын қарастырған жөн. Басқару шешімін қабылдау үшін қандай көрсеткіштің қажет немесе қажет еместігін анықтау қажет. Орындаушының әрқайсысына оның қолданатын ақпараты ғана келіп тұруы тиіс. Деректер қоймасын құру методологиясы оны жобалаудың теориялық деңгейіне негізделеді. Методологияның негіздерін тәжірибеде тізбектей жүзеге асырылатын екі кезең түрінде атап өтейік.

а) 1-кезең: ұйымның барлық функционалды бөлімшелерін зерттеу мақсаттары: оның іс әрекетінің құрылымы мен спецификасын түсіну; ақпарат ағымдарының схемасын жасау; ондағы құжат айналымы жүйесіне талдау жасау; ақпараттық объектілерді және олардың қасиеттері мен міндеттерін сипаттайтын реквизиттің құрамын анықтау;

б) 2-кезең: 1 кезеңде зерттелген іс әрекет саласына арналған концептуалды ақпараттық логикалық деректер моделін жасау. Бқл модельде объектілер мен олардың реквизиттерінің арасындағы барлық байланыстар

орнатылып, оңтайландырылуы тиіс. Ақпараттық логикалық модель деректер қорын жасаудың негізі болып табылады.

Абстракцияның алғашқы деңгейі негізгі қолданушылардың заттар облысы туралы көрінісіне сәйкес келеді – оларды пайдаланушылардың локальды көрінісі деп атайды.

Екінші деңгейді инфологиялық түсінік деп атаймыз; ол өз бетінше пайдаланушылардың локальді көрінісін көрсетеді және ақпараттар объектісінің барлық жиынын көре алады.

Абстракцияның концептуалды деңгейін мәліметтер базасының администраторы мәліметтердің логикалық ұйымдасу бейнесіне сәйкес болады. Бұл абстракция деңгейі инфологиялық деңгейге ұқсас, бірақ оның өзгешелігі МБЖҚ - ның әдістерін іске асыруында. Концептуалды деңгейде мәліметтер базасын сипаттау МБЖҚ-ның осы жүйеде қолданылатын терминдер мен шектеулер арқылы сипатталынады.Абстракцияның әрбір деңгейінде заттық объекттің өзіндік моделі анықталады. Бұл модельдерді сипаттау схема деп аталынады. Қазіргі кездегі МБЖҚ архитектурасы мәліметтер базасыны сипаттауындағы концептуалды және ішкі тәуелсіз тапсырма болып табылады.

2.3 Мәліметтер қорын программалық жабдықтау

Қазіргі уақыттағы МҚБЖ-лар мәліметтер қорымен жұмыс істеу бойынша ауқымды есептерді қосымшаны жасамай-ақ шешуге мүмкіндік береді. Дегенмен де, қосымшаны жасау дұрыс болатын жағдайлар да бар. Мысалы, егер мәліметтермен манипуляциялауды автоматтандыру қажет етілсе, МҚБЖ-ның терминалдық интерфейсі жеткілікті түрде дамымаған болса, немесе МҚБЖ-дағы ақпаратты өңдеуден өткізу жөніндегі стандарттық қызметтер пайдаланушының қажеттерін қанағаттандырмайтын болса.

Мәліметтер қорын қолданбалы программалық қамтамасыз ету дегеніміз – бұл нақты мәселелерді шешу үшін пайдаланушылардың өздері құрастыратын программалар мен жүйелер. Қазіргі уақыттағы программалаудың алгоритмдік тілдерінде – Delphi, C++, Visual Basic және т.б. – SQL тілін пайдалана отырып мәліметтер қорына қол жеткізетін программалау құралдары бар. Сонымен қатар, көптеген МҚБЖ-ларда жүйе ішіне орнатылған программалау тілі бар. Dbase форматындағы мәліметтер қорын программалауға арналған арнайы Clipper алгоритмдік тілі бар.

Қосымшалады жасау үшін МҚБЖ –ның программалық интерфейсі болуы тиіс, оның негізін сәйкес программалау тілінің функциялары және/немесе процедуралар құрайды. Қолданыстағы МҚБЖ –лар қосымшаларды жасаудың келесі технологияларына (және олардың комбинацияларын) қолдау жасайды:

– программаларды қолмен кодтау (Clipper, FoxPro, Paradox);

– генераторлардың көмегімен қосымшалардың мәтіндерін құрастыру (FoxPro-де FoxApp, Paradox-та Personal Programmer);

– визуалды программалау әдістерімен дайын қосымшаны автоматты түрде генерациялау (Delphi, Access, Paradox for Windows).

Қолмен кодтау кезінде программалаушылар қосымшалар программаларын мәтінін қолмен тереді, сосын олардың жұмысын жүргізіп баптайды.

Генераторларды пайдалану қосымшаларды жасауды жеңілдетеді, себебі бұл жағдайда программалық кодты қолмен термей-ақ алуға болады. Қосымшалардың генераторлары қосымшалардың негізгі элементтерін (мәзір, экрандық форма, сұраныстар және т.б.) жасауды жеңілдетеді, алайда көбінесе қолмен кодтауды жоққа шығара алмайды.

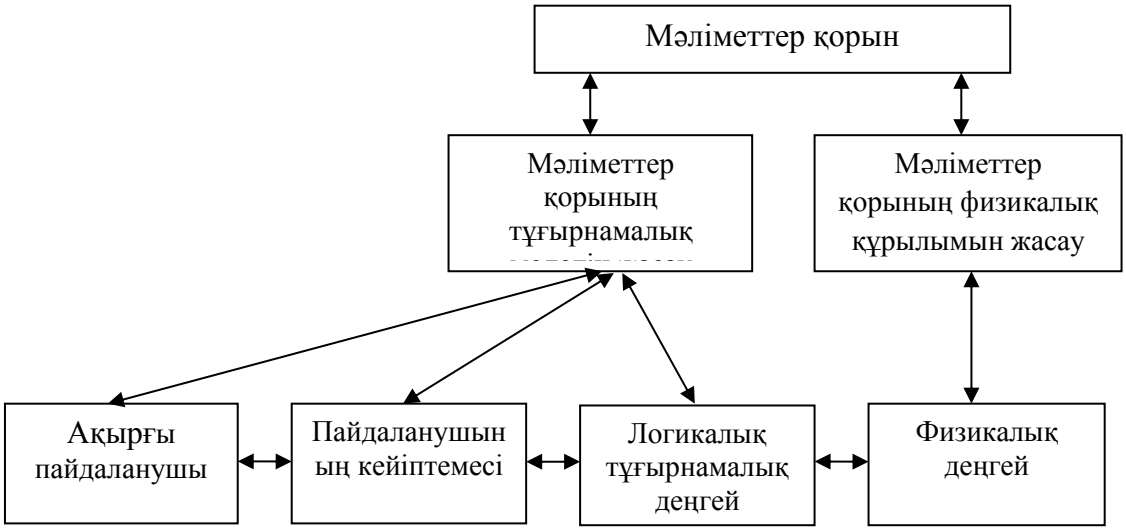
Қосымшаларды визуалды программалау құралдары қосымшалардың генераторларын қолдану идеясының дамуы боып табылады. Бұл ретте қосымша ыңғайлы біріктірілген ортаның көмегімен дайын түрдегі «құрылыс блоктарынан» құрылады. Қажет болған жағдайда жасаушы қосымшаға өзінің кодын оңай енгізеді. Біріктірілген орта, әдетте, қосымшаларды құрастыру, баптау мен өзгертудің күшті құралдарын ұсынады. Визуалды программалау құралдарын пайдалану қысқа мерзім ішінде алғашқы екі әдіспен алынған қосымшалардан да сенімді, тартымды және тиімді қосымшаларды құрастыруға мүмкіндік береді.

Жасалған қосымша әдетте амалдар жүйесінің бір немесе бірнеше файлдарынан құралады.

Егер де қосымшаның негізгі файлы орындалатын файлы (мысалы, exe-файл) болса, онда бұл қосымша МҚБЖ ортасынан дербес түрде орындалатын тәуелсіз қосымша болып табылады. Іс-жүзінде тәуелсіз қосымшаны алу, мәтінді қолмен теру сияқты, сондай-ақ қосымшаның генераторының немесе визуалды программалау ортасының көмегі сияқты тәсілдерімен алынған программаның бастапқы мәндерін компиляциялау жолымен жүзеге асырылады.

Тәуелсіз қосымшаларды мысалы, FoxPro МҚБЖ-лары және Delphi визуальді программалау жүйесін алуға мүмкіндік береді. Айта кететін нәрсе, Delphi құралдарының көмегімен әдетте, тәуелсіз қосымшаларды жасамайды, себебі бұл өте көп еңбекті қажет ететін үдеріс, оның орнына МҚБЖ ядросының рөлін атқаратын BDE (Borland DataBase Engine) мәліметтер қорының процессорын пайдаланады. Дербес ЭЕМ-дарға арналған қосымшаларды жасаудың ең алғашқы құралының бірі «таза түрдегі компилятор» болып саналатын Clipper жүйесі болып табылады [8].

Көп жағдайларда қосымша МҚБЖ ортасы болмағанда орындалмайды. Қосымшаны орындау МҚБЖ қосымша файлдарының құрамдас бөлігін (жеке жағдайда бұл бастапқы программаның мәтіні) талдайды және орындалатын қажетті машиналық командаларды автоматты түрде құруына негізделген. Былайша айтқанда, қосымша интерпретациялау әдісі арқылы орындалады. Сонымен қатар, компиляция мен интерпретацияның аралық нұсқасын – былайша айтқанда (псевдокомпиляция) жалған компиляцияны пайдаланатын жүйелер болады. Мұндай жүйелерде бастапқы программа компиляция арқылы аралық кодқа (псевдокодқа) өзгереді және дискіде жазылады.



2.2-сурет – Мәліметтер қорының үш деңгейлік архитектурасы

Интерпретациялау режимі қазіргі заманғы көптеген МҚБЖ-ларда, мысалы, Access, Visual FoxPro және Paradox сияқты сондай-ақ, FoxBase және FoxPro тәрізді бұрынғы МҚБЖ-ларда жүзеге асырылған.

Бұл трүде оны кейбір жүйелерде тіпті редакциялауға рұқсат етіледі, алайда жалған компиляцияның басты мақсаты – программаны оны интерпретациялау үдерісін жылдамдатуға мүмкіндік беретін түрге өзгерту. Мұндай тәсіл DOS басқаруымен жұмыс істейтін МҚБЖ-ларда, мысалы, Foxbase+ және Paradox 4.0/4.5 for DOS-та кеңіннен қолданылған.

Windows басқаруымен жұмыс істейтін МҚБЖ-ларда (псевдокод) жалған код көбінесе қосымшаны модификациялауға тыйым салу үшін қолданылады. Бұл жұмыс істеп жатқан программаны кездейсоқ немесе әдейі бұзудан сақтау үшін пайдалы болады. Мысалы, мұндай тәсіл Paradox for Windows МҚБЖ-да қолданылған, онда әзірленген экрандық формалар мен есеп берулерді редакциялауға сәйкес нысандарға өзгертуге жол беріледі.

Кейбір МҚБЖ-лар пайдаланушыға қосымшаны жасау нұсқасын таңдап алу мүмкіндігін ұсынады: МҚБЖ-мен интерпретацияланатын программалық код ретінде немесе тәуелсіз программа ретінде.

Тәуелсіз қосымшаларда қолданудың артықшылығы – машиналық программаны орындау уақыты, әдетте интерпретация кезіндегі уақытпен салыстырғанда қысқалау болады. Мұндай қосымшаларды әлсіз машиналарда және қосымшаны пайдаланушылардың тарапынан өзгертуге қарсы жабу қажет болатын, жүйелерді, «қолдануға даяр» етіп орнатқан жағдайда пайдаланған дұрыс.

Интерпретацияланатын жүйелердің тағы бір үлкен артықшылығы – жақсы МҚБЖ-ларды әдетте мәліметтердің бүтіндігін бақылау мен оларға рұқсатсыз қол жеткізуден қорғауға арналған күшті құралдар болады, ал компиляциялаушы түрдегі жүйелер туралы олай деп айта алмаймыз.

Сонғыларында аталған қызметтерді қолмен программалауға, немесе әкімшелердің құзырына қалдыруға болады.

Қосымша жасау құралдарын таңдау кезінде келесі негізгі үш факторды: компьютердің ресурстарын, қосымшаның ерекшеліктерін (программаның қызметтерін модификациялаудың қажеттігі, жасауға жұмсалатын уақыт, қол жеткізуді бақылаудың қажеттілігі және ақпараттың тұтастығын қамтамасыз ету) және жасаудың мақсатын (шеттетілетін программалық өнім немесе өзінің күнделікті іс-әрекетін автоматтандыру жүйесі) ескерген дұрыс.

Қазіргі заманғы компьютері бар және онша күрделі емес қосымшаны құруды жоспарлап отырған пайдаланушы үшін интерпретациялау типіндегі МҚБЖ көбірек сай болады. Еске сала кейік, мұндай жүйелер жеткілікті дәрежеде күшті болып келеді, олардың жоғары деңгейлі құралы бар, олар жасау мен баптауға ыңғайлы, жасау жұмысын жылдам түрде орындауға мүмкіндік береді және қосымшаға еріп жүру мен оны модификациялаудың ыңғайлы болуын қамтамасыз етеді.

Сипаттамалары нашар компьютерді пайдаланған кезде тәуелсіз қосымшаларды жасау құралдары бар жүйені таңдаған дұрыс.

Бұл ретте ескертетін нәрсе, қосымшадағы кішігірім өзгеріс программалау, программаны компиляциялау және баптау кезеңдерін циклдік түрде қайталануына әкеледі. Тәуелсіз қосымшаны орындау мен қосымшаны интерпретация режиміндегі орындаудан айырмашылығы тәуелсіз қосымшаның пайдасына миллисекунд айналасында тербелуінде болады. Сонымен қатар, қосымшаны оны пайдалануға дайындау уақытындағы айырмашылық әдетте интерпретациялау жүйелерінің пайдасына минуттар-сағаттар дәрежесіндегі шама құрайды.

2.4 SQL және мәліметтер қорының қауіпсіздігі

МҚБЖ-нің қызметтерінің біріне мәліметтер қорының қауіпсіздігін қамтамасыз ету жатады. Реляциялық МҚБЖ-лардың қауіпсіздік жүйесінің негізі SQL тілі болып табылады. Мәліметтерді қорғаудың негізгі үш принципі қолданылады:

а) мәліметтерді оқуға, енгізуге және жаңартуға арналған кез келген SQL-команда қандай да бір пайдаланушының атынан орындалады;

б) қорғау объектілері негізінен кестелер мен бейнелеу болады;

в) әрбір пайдаланушыға қандай да бір әрекеттерді жасауға арналған белгілі бір артықшылықтар (құқықтар) беріледі.

Реляциялық мәліметтер қорының әрбір пайдаланушысына идентификатор пайдаланушыны бірмәнді анықтайтын қысқаша атау меншіктеледі. Коммерциялық, өнеркәсіптік мәліметтер қорларында пайдаланушыға атауды мәліметтер қорының әкімшісі береді. Дербес ЭЕМ-дарда мәліметтер қорында мәліметтер қорын құрған пайдаланушының тек бір ғана идентификаторы болуы мүмкін. Көптеген МҚБЖ-лар үшін идентификатор ретінде операциялық жүйеде тіркелген пайдаланушылардың атаулары қолданылады. Әдетте, пайдаланушы жұмыс алдында идентификаторды және

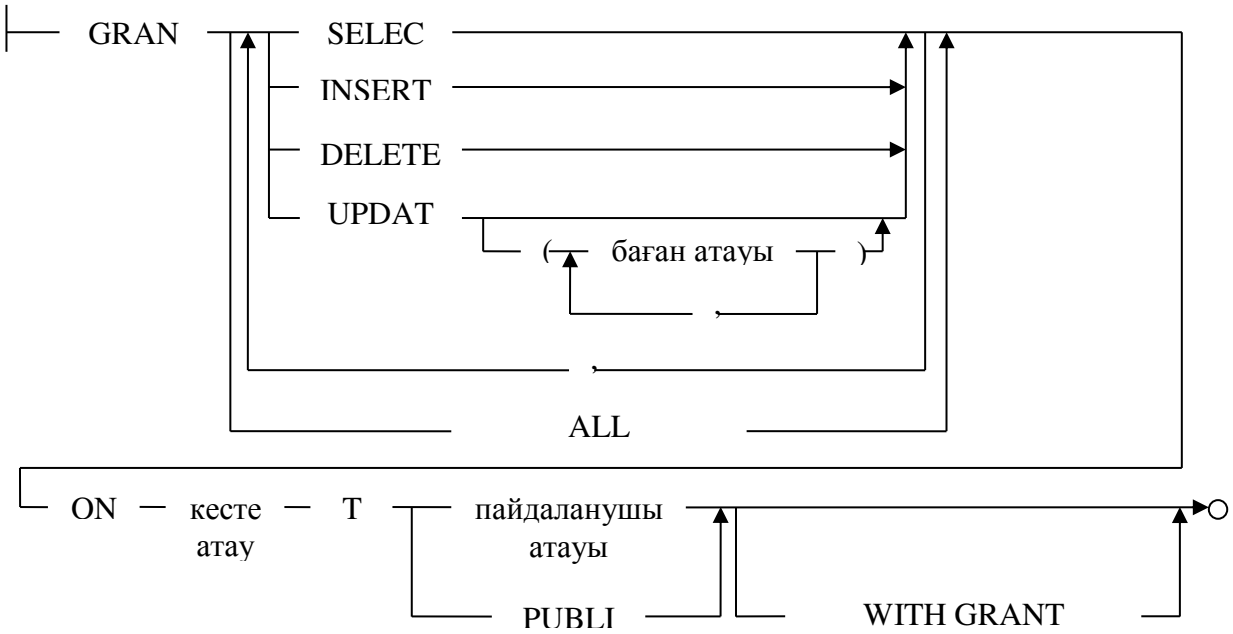
сонымен байланысты парольді енгізуі тиіс. Әртүрлі пайдаланушылардың бірдей идентификаторлары болуы мүмкін.

Артықшылықтар. SQL стандарттарында төмендегідей артықшылықтар анықталған:

- оқуға (SELECT);
- жазбаларды енгізуге (INSERT);
- жазбаларды жоюға (DELETE);
- жазбаларды өзгертуге (UPDATE).

CREATE TABLE және CREATE VIEW операторлары көмегімен кестені немесе бейнелеуді құрған пайдаланушы автоматты түрде осы объектілердің иесіне айналады және осыларға қатысты жоғарыда аталған барлық артықшылықтарға ие болады. Кестенің немесе көрсетімнің иесі басқа пайдаланушыларға осы мәліметтерге қол жеткізуге рұқсат ете алады.

АЖ құрудың бастапқы этаптарында біз автоматандырғымыз келіп жатқан ұйым қалай жұмыс істейтіндігін ұғынып алған жөн. Ол ұйымда жұмыс істеушінің бірі де оны толық білмейді, сол үшін АЖ құрар кезде соны анықтап білу керек. Ұйым жетекшісі оны толық біледі деуге болады, бірақ ол әрбір жұмысшының ісін ашып айта алмауы мүмкін. Ал қатардағы жұмысшы өз жұмысына жауапты ол да басқалар жайында ақпараттарды бірмейді. Сондықтан ұйым жұмысын сипаттау үшін модель құру керек.



2.3-сурет – GRANT операторының синтаксистік диаграммасы

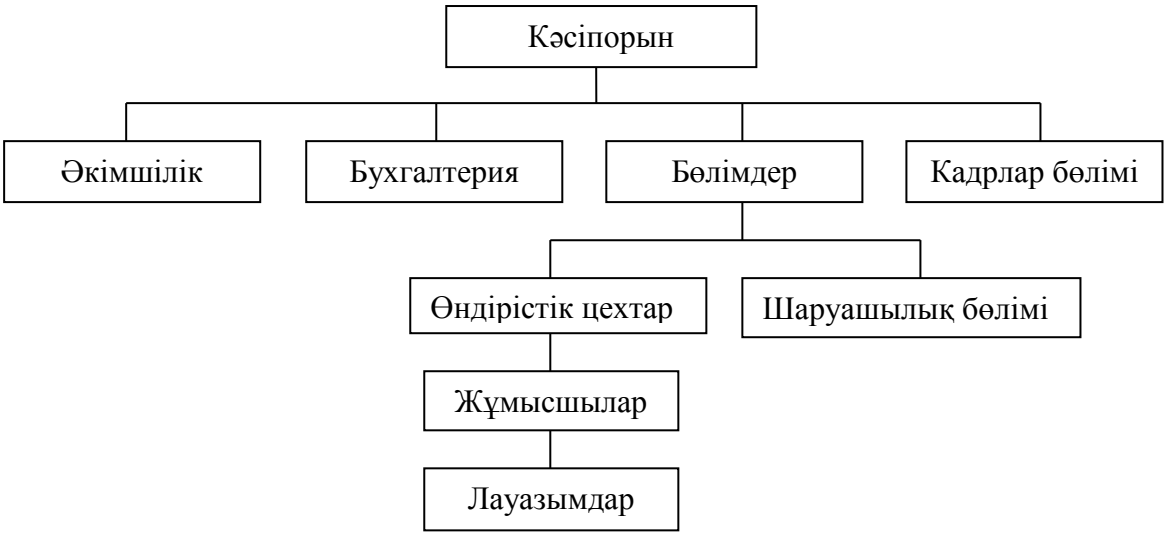
Иерархиялық модель. Алғашқыда мәліметтер қорының иерархиялық модельдері пайда болды. Иерархиялық модельде мәліметтер арасындағы байланыстар реттелген граф (немесе бұтақ) түрінде берілген. Иерархиялық

МҚ-ның құрылымын (сұлбасын) сипаттау үшін кейбір программалау тілінде «бұтақ» мәліметтер типі қолданылады.

«Бұтақ» типі ПЛ/1 және Си программалау тілдерінің «құрылым» атты мәліметтер типімен және Паскаль тілінің «жазба» типімен ұқсас. Оларда типтердің бір-біріне салынуына жол беріледі, олардың әрқайсысы белгілі бір деңгейде орналасқан. «Бұтақ» типі құрамдас болып келеді. Оған ішкі типтер («кіші бұтақтар») кіреді, олардың әрқайсысы өз кезегінде «бұтақ» типі болып табылады. «Бұтақ» типінің әрқайсысы бір «түбірлік» типтен және бағыныңқы типтердің реттелген жиынтығынан (бос болуы да мүмкін) тұрады. «Бұтақ» типіне енгізілген қарапайым типтердің әрқайсысы «жазба» атты жай немесе құрама тип болып табылады. Қарапайым «жазба» бір типтен тұрады, мысалы, сандық типтегі, ал құрама «жазба» типтердің белгілі бір жиынтығын, мысалы, бүтін, символдар жолы және көрсеткіштерді (сілтеме) біріктіреді.

Түбірлік деп бағыныңқы типтері бар және өзі ішкі тип болмайтын тип аталады. Бағыныңқы тип (ішкі тип) өзі үшін ата тегі (ата-ана) рөлін атқаратын типке қатысты ұрпақ болып табылады. Бір типтің ұрпақтары бір-біріне қатысты егіздер болып табылады. Тұтастай алғанда «бұтақ» типі «жазба» типтерінің иерархиялық тұрғыдан ұйымдастырылған жиынтығын сипаттайды[9].

Иерархиялық модельдің мәліметтер қорлары бұтақ түріндегі құрылымы бар нысандарды сипаттауға ыңғайлы. Мысалы, белгілі бір кәсіпорынның құрылымын келесі түрде келітруге болады:



2.4-сурет – Кәсіпорынның иерархиялық құрылымы

Иерархиялық МҚ құрамында «жазба» (жазбалар) типіндегі мәліметтер даналарын (экземплярын) қамтитын «бұтақ» типіндегі мәліметтер даналарының реттелген жиынтығы болып табылады. Көп жағдайда типтердің арасындағы туыстық қатынастарды жазбалардың арасындағы қатынастарға көшіреді. Жазбалар өрістері шындығында МҚ-ның негізгі мазмұнын құрайтын сандық немесе символдық мәндерді сақтайды. Иерархиялық МҚ-ғы барлық

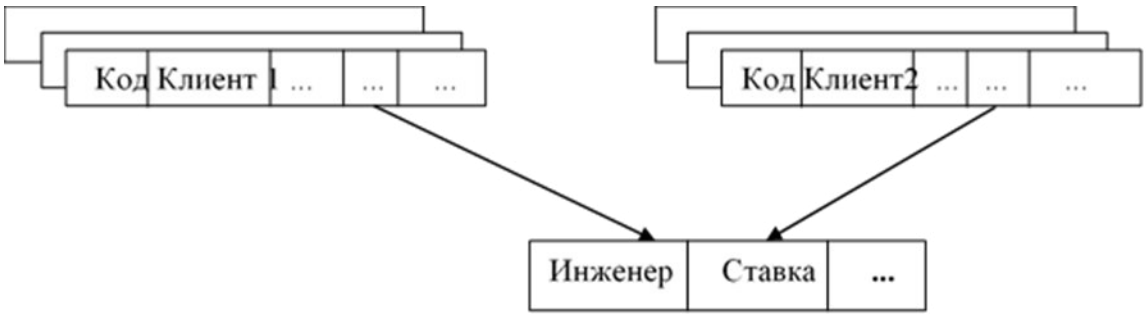
элементтерді қарап шығу әдетте жоғарыдан төмен қарай және солдан оңға қарай жүргізіледі.

Желілік модель. Желілік модельдерде мәліметтердің арасындағы өзара байланыстар еркін граф түрінде болады, ондағы әрбір ұрпақтың екі және одан да көп ата тегі болуы мүмкін. Желілік модельдің мәліметтер қоры жазбалар жинағы мен байланыстар жинағынан құралады. Байланыстардың жинағы физикалық көрсеткіштері бар өрістен тұрады.

Осылайша, желілік МҚ сұлбасын сипаттау үшін типтердің екі тобы: «жазба» және «байланыс» қолданылады. «Байланыс» типі «жазба» типінің ата тегі және ұрпақ типтері үшін анықталады. «Байланыс» типінің айнымалылары байланыстардың даналары болып табылады.

Желілік МҚ жазбалар жинағы мен сәйкес байланыстар жинағынан тұрады. Байланыстарды қалыптастыруға ерекше шектеулер қойылмайды. Егер иерархиялық құрылымдарда ұрпақ-жазбаның бір ғана ата тегі –жазбасы болса, ал мәліметтердің желілік моделінде ұрпақ-жазба ата тегі-жазбалардың кез келген санына ие бола алады.

Келесі суретте инженер лауазымына сәйкес келетін жазбаға екі жазба сілтеме жасайды.



2.5-сурет – Желілік модельдегі жазбаларды байланыстыру

Объектілі-бағытталған модель. Бұл модельдің негізін объектілі-бағытталған программалаудың идеялары мен принциптері (ұстанымдары) құрайды. Объектілі-бағытталған МҚ-ның логикалық құрылымы сырттай иерархиялық МҚ-ның құрылымына ұқсас болып келеді. Олардың бір-бірінен негізгі айырмашылығы – мәліметтерді манипуляциялау әдістерінде. Мәліметтер қорының объектілі-бағытталған моделі өзара күштің байланысқан мәліметтермен жұмыс істеуге мүмкіндік береді.

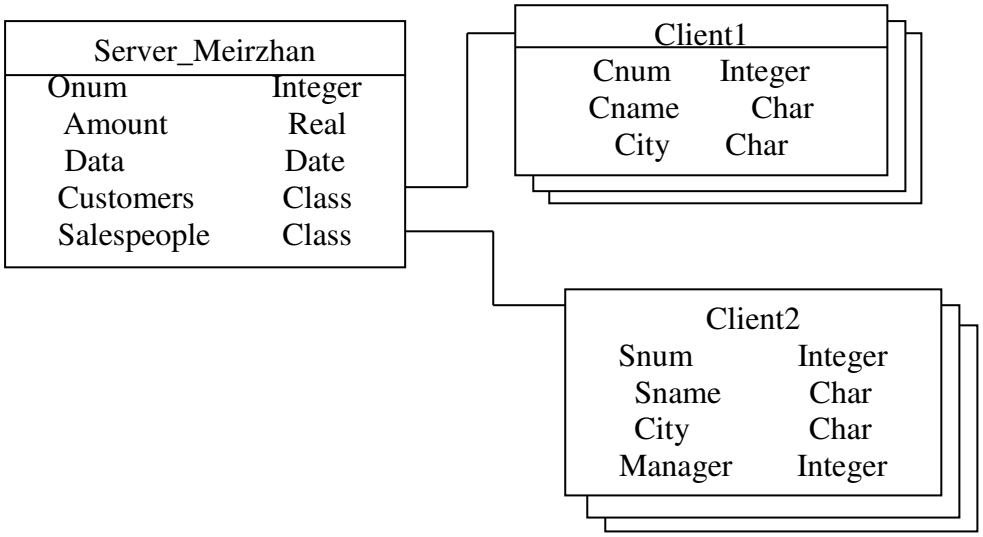
Мәліметтердің объектілі-бағытталған моделінің реляциялық модельмен салыстырғандағы негізгі артықшылығы – объектілердің күрделі өзара байланыстары туралы ақпаратты бейнелеу мүмкіндігі. Мәліметтердің объектілі-бағытталған моделі мәліметтер қорының жеке жазбасын идентификациялап, оларды өңдеуден өткізудің қызметтерін анықтауға мүмкіндік береді. Объектілі-бағытталған модельдің кемшілігі – түсінудің жоғары деңгейдегі қиындығы, мәліметтерді өңдеуден өткізудің ыңғайсыздығы және сұраныстарды орындау жылдамдығының төмендігі.

Мәліметтер мен оларды өңдеуден өткізу әдістерінің арасында программалаудың объектілі-бағытталған тілдеріндегі сәйкес құралдарға ұқсас механизмдердің көмегімен өзара байланыстар орнайды. Объектілердің қасиеттері тек real, char және сол сияқты стандартты типтермен ғана емес, сондай-ақ пайдаланушылардың өздері жасайтын типтермен де сипатталуы мүмкін.

Класстар – деп аталатын бұл типтер көптеген объектілерді құруға арналған шаблон (қимаүлгі) болып табылады. Class типінің мәні болып табылатын объектілер осы класстың даналары деп аталады, мысалы, 2.6-суретте көрсетілгендей:

Класстардың негізінде инкапсуляция, мұра етіп алу және полиморфизм деген үш іргелі ұстаным жатыр.

Инкапсуляция нысанды өзге класстардың объектілерінен оқшаулауы, оның қасиеттерінің көрінуін шектейді.Қасиеттің мәні сол инкапсуляцияланған нысанмен анықталады. Мұра етіп алу, керісінше, аталық объектінің қасиеттерін оның барлық ұрпақтарына таратады.



2.6-сурет – Объектілі-бағытталған модельдегі жазбаларды байланыстыру

Полиморфизм әртүрлі кластағы объектілерде атаулары бірдей қызметтер мен өңдеу процедураларын (әдістерін) қолдануға жол ашады [10].

Мәліметттермен әрекеттерді орындау үшін қарастырылып отырған МҚ моделінде инкапсуляция, мұра ету және полиморфизмнің объектілі-бағытталған механизмдерімен күшейтілген логикалық амалдар қолданылады. SQL командаларына (мысалы, МҚ құру үшін) ұқсас амалдар шектелген түрде қолданылуы мүмкін. Модельдеу тілі жобаның сипаттамасын беру үшін қолданылатын нотация. Нотация – бұл модельдерде қолданылатын графиктік объектілердің жиынтығы. Модельдеу тілінің синтаксисі де нотациямен анықталады.

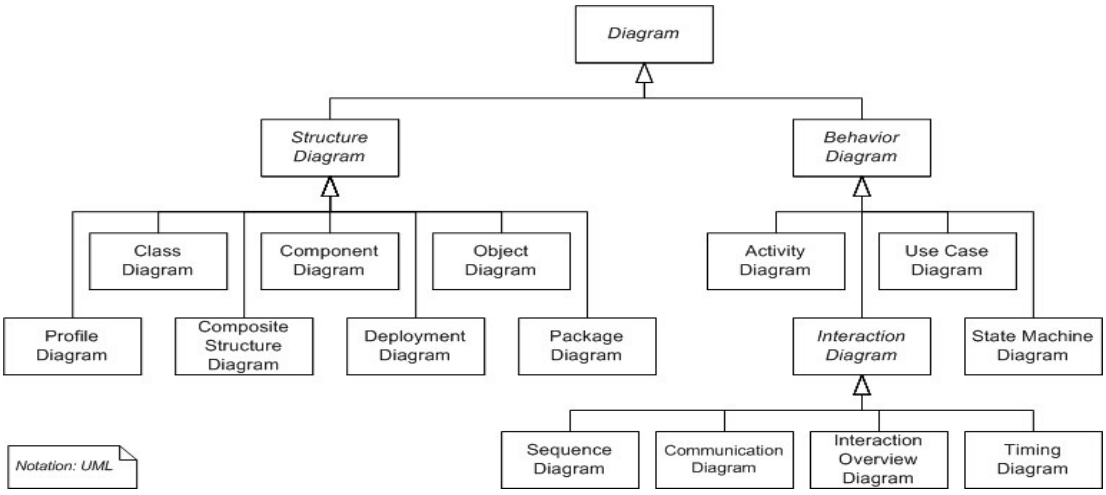
2.5 UML Диаграммалар негізі

UML тілінің негізгі мақсаттары мен мүмкіндіктері:
– қолданушыға түсінікті болатын визуальды модельді құру;
– модельдегі базалық концепциялардың кеңейтуге бейім болуы;
– программалау тілдеріне, құру процессіне тәуелсіз болуы;
– модельдеу тілінің формальды негізде болуын қамтамасыз етеді;
– объектілік бағдарланған жабдықтар нарығына стимуляция жасайды;
– практикалық тәжірибелердің ең жақсысын біріктіру және тарату;

UML –де қолданылатын негізгі диаграммаларды келесі топтарға бөліп қарастырады:

2.7-сурет – UML-де қолданылатын негізгі диаграммалар

Диаграммалар құрылымы (2.8-сурет) сызба-нұсқада көрсетілген:



2.8-сурет – Диаграммалар құрылымы

2.6 Rational Rose аспабында құрылған диаграммалар

Класстар диаграммасы – жүйедегі класстардың статикалық құрылымын модельдеу үшін және класстар арасындағы байланысты көрсету үшін жасалады. Класстар диаграммасы объектіге бағдарланған ұстанымдағы негізгі диаграмма болып табылады.

Класстар диаграммасының қызметі: жүйедегі объектілердің типін анықтау және олардың арасындағы байланысты көрсету болып келеді. Байланыстың статикалық екі түрі қолданылады: ассосация және подтиптер (тума типтер). Бұлардан басқа класстар диаграммасының элементтеріне атрибуттар, операциялар және объектілер арасындағы шектеулер жатады. Класстар диаграммасын жобалаудан бұрын, ол диаграмманың қандай мақсатта қолданылатынын анықтап алу керек. Класстар диаграммасын жобалаушы үш түрлі мақсатта қолдануы мүмкін:

– концептуалдық аспект – мұнда класстар даграммасы зерттелетін пәндік облыстағы негізі ұғымдарды анықтайды. Бұл ұғымдар болашақта құрылатын класстарға сәйкес болу керек, бірақ іс жүзінде ол барлық уақытта бірдей орындалмайды. Сондықтан концептуалдық модель болашақ ақпараттық жүйемен әлсіз байланыста болады және ол программалау тіліне тәуелсіз болады;

– спецификациялық аспект – мұнда құрылатын диаграмма ақпараттық жүйенің (программалық жабдықтың) интерфейсі деңгейінде жасалады. Класстың өзінің ішкі құрылымы қарастырылмайды;

– жүзеге асыру аспектісі (реализация) – мұнда класстар диаграммасы ақпараттық жүйеге (программалық жабдыққа) қатысатын класстарды ішкі құрылымдарымен қоса анықтайды. Бұл аспекті программистер үшін негізгі диаграмма болып табылады.

Диаграмма тұрғызуда жалғыз аспектіні таңдап алу қажет. Диаграмманы оқу кезінде оның қандай аспектіге сәйкес тұрғызылғанын білу қажет. Яғни бұл білім бізге диаграмманы дұрыс интерпретациялауда қажет болады.

Кластың атауы үшін пән облысына сәйкестендіріліп қабылданған терминдерді қолданған дұрыс. Класс аты ретінде жобаланатын түсінікті толығымен сипаттай алатын зат есіміңіз жекеше түрі қолданылады. Кейде қысқартылған атауларда қолданылып, класты құжаттандырғанда міндетті түрде мағынасын ашып көрсету керек. Егер аббревиатура бірдей мағыналы емес интерпретация жіберсе, онда сәйкесінше айтылудың толық түрін қолданамыз.

UML-да класс аймақтарға бөлінген тіктөртбұрыш түрінде кескінделеді. Жоғары аймақта класс аты, ортасында оның құрылымы (атрибуттар тізімі), оның астындағысында тәртібі сипаттамаларын анықтайтын функциялар беріледі.

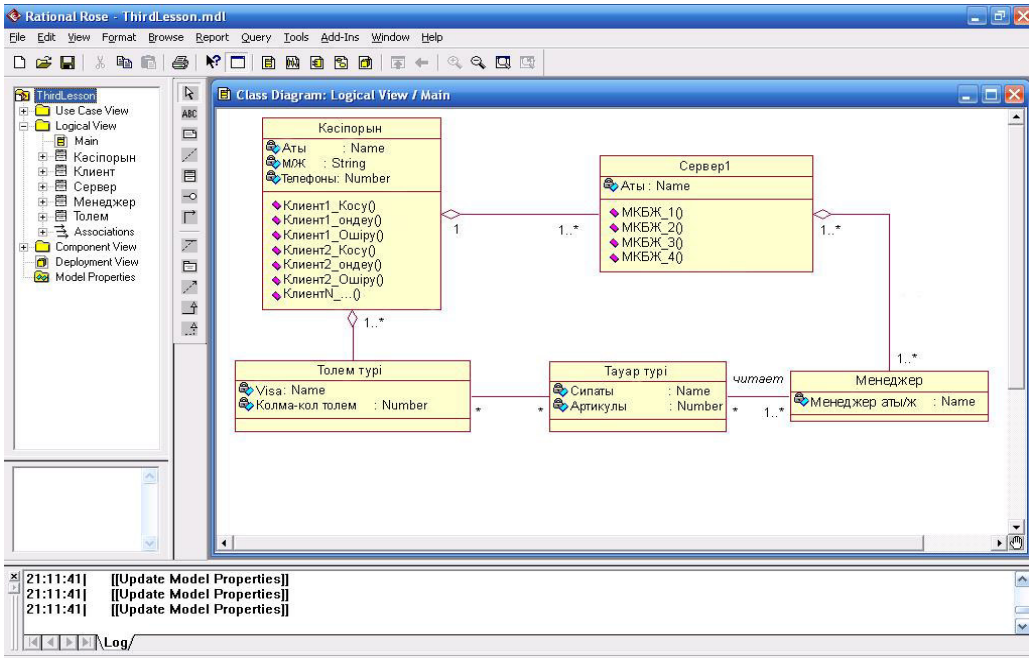
Кластар диаграммасы UML-дың ресми бөлігіне жатпағанымен модельдерді тұрғызу мен анализдеуде өте пайдалы болады. UML конструкциясын көрудің кез-келген үш нүктесінде қолдануға болады.

Кластар диаграммасы объектілі-бағдарланған тәсілдің орталық буыны болып табылады. Кластар диаграммасы жүйе объектілерінің типтерін және олардың арасындағы болатын түрлі статикалық байланыстарды анықтайды.

Кластар диаграммасын сипаттаудан бұрын, осы диаграммаларды құрастырушының қолдану сипатына байланысты сәтке назар салған жөн. Бұл сәт құжаттандырылмайды, бірақ ол диаграммаларды интерпретациясы тәсіліне әсер етеді, сондықтан моделдер көмегімен сипаттауға оның қатысы маңызды.

Класс ортақ қасиетттері (атрибуттары), тәртібі (функциялары), семантикасы және басқа объектермен байланысы бар объектер тобын анықтайды. Кластың объектіні құруға арналған шаблон ретінде қарауға болады. Әрбір объект қандайда бір ғана кластың нұсқасы.Дұрыс құрылған класс тек бір ғана абстракцияны бере алады. Мысалы, студент туралы мәлімет сақталған, сонымен қатар барлық оқу барлығында студент өткен курстар тізімі функциясы көрсетілген класты сәтті құрылған деп айта алмаймыз, өйткені ол екі әртүрлі операциялар тобын қамтиды. Аспект түсінігі кластар диаграммасын тұрғызумен қатар, оны оқуда да үлкен маңызы бар. Бірақ аспектілер арасындағы айырмашылық айқын емес, сондықтан көптеген құрастырушылар диаграмма тұрғызуда олардың ығысу мүмкіндігіне рұқсат етеді.

Кластар диаграммасы 2.9-суретте көрсетілген.



2.9-сурет – Класстар диаграммасы

Прецеденттер диаграммасы. Прецеденттерді модельдеу талаптарды орнатумен тығыз байланысты. Талаптарды сипаттау құжатында мәтіндік түрде берілетін талаптарды прецеденттерге дейін жеткізу керек, бірақ бұл тек функционалды талаптарға ғана қатысты. Егер өмірлік циклда қолданушы талаптарының өндірісі өзгерсе, онда бұл өзгерістерде талаптарды сипаттау құжатында сияқты прецеденттер моделінде де белгілеп өткен жөн.

Әрбір прецедент құжатта тіркелген оқиғалар ағыны көмегімен сипатталған болу керек. Сәйкес мәтіндік құжат актер прецеденті инициялизацияланған кезде жүйе не істеу керектігін анықтайды. Прецеденті сипаттайтын құжаттың құрлымы әр түрлі болады, бірақ қарапайым сипаттама келесі бөлімдерден тұру керек:

– қысқаша сипаттама;

– алдын ала шарттар;

– оқиғалардың ағынын бөлшектеп сипаттау: негізгі ағын және альтернативті ағындар.

Қолдану вариантындағы адам фигуралары орындаушылар, сопақ шеңберлер қолдану варианты, ал сызықтар мен бағыттауыш сызықтар – орындаушы мен қолдану варианты арасындағы байланыстарды көрсетеді. Қолдану варианты деп сыртқы объект тудырған оқиғаға жүйе жауап ретінде орындаған әрекеттердің тізбегін айтады. Ол қолданушыға таныс белгілі бір функцияны қамтуы мүмкін, сондай-ақ кіші немесе керегінше үлкен де бола алады және қолданушы үшін белгілі бір дискретті есепті шеше алады. Қолданушының қандай да бір функцияларды іске асырғысы келетінін талқылау процесінде қолдану варианты анықталады.

Орындаушы – қолданушының жүйеге қатысты ойынындағы рөлі. Орындаушы рөлді береді, яғни ол нақты адам немесе жұмыс атын көрсетпейді. Қолдану варианты диаграммаларында орындаушы адам фигурасы түрінде келтірілсе де, ол осы жүйеден белгілі бір ақпарат алғысы келетін сыртқы жүйе болуы мүмкін. Орындаушыға белгілі бір орындау варианты керек болғанда ғана, оны диаграммада көрсету қажет.

Кез-келген қолдану варианты жүйенің функциялығына қойылған сыртқы талаптармен байланыста болады. Айталық, есептеу жүйесі файлды талап етсе, онда ол талап қанағаттандырылуы қажет. Қолдану вариантарын әр уақытта қолданушының нақты есептерін анықтап және оларды шешудің альтернативті әдістерін қарастыра отырып, жүйенің орындаушыларымен бірге анализдеу қажет [12].

Қолдану вариантын иденфикациялаудың көзі болып сыртқы оқиғалар қызмет атқарады. Алдымен жүйе реакция беретін сыртқы әлемде болып жатқан оқиғаларды тізбектеуден бастаған дұрыс. Кейбір нақты оқиғалар қолданушыны араластырмай жүйенің реакциясын тудыруы немесе қолданушының өз реакциясын тудыруы мүмкін. Реакция беруді қажет ететін оқиғаларды идентификациялау қолдану вариантарын айқындауға көмектеседі.

Жүйе тәртібінің бір фрагменті бірнеше қолдану нұсқасында қайталанса, онда оның сипаттамасын әр нұсқаға көшірмеу үшін «қолдану» байланысы қолданылады. «Кеңейту» және «қолдану» байланыстарының арасындағы ұқсастық пен айырмашылықтар. Бұл екеуі де бірнеше қолдану нұсқаларының ортақ тәртіптер фрагменттерін жеке нұсқаға бөлектеп, оны басқада нұсқаларға «қолданады» немесе оларды «кеңейтеді». Екінші жағынан, бұл жағдайлар оны әртүрлі мақсатта жасайды.

Қолдану нұсқалары программалық қамтамаға талаптарды құру

сатысындағы қажетті құрал болады. Әрбір қолдану нұсқасы – бұл жүйеге қойылатын потенциал талаптар және оларды айқындамай жүйені іске асыруды жоспарлау мүмкін емес.

Жүйені құру үшін зат облысын білу қажет. Сонымен қатар жүйенің қалай жұмыс істейтіні туралы хабарлар болу керек. Диаграмма жүйесі орындайтын операция тізімін құру үшін арналған. Ол берілген функциядағы жүйе объектісі анықталған тізімдер жүйесін орындайды. Осындай түрде орындалатын жүйенің функция жүйесі құрылады және жүйемен қарым-қатынастағы объектілер әрекетінің сценарийлары жазылады. Прецедент процесін модельдеу үшін қолданушы іс-әрекетін толық көрсетуіне және толық реакциясына бағытталған. Себебі, іс-әрекеті олардың шарттарына негізделген. Жүйенің жұмысы онымен қалай айналысады және не қажет ететіне байланысты.

Прецеденттер диаграммасы қолданушы көзқарасы бойынша жүйе тәртібін анықтайды. Прецедент диаграммасы жүйе динамикасын алғашқы модельдеуі үшін басты құрал ретінде қарастырылады, өндірілетін жүйелерге талаптарды анықтау қолданылады, кейінгі өндірулерді жүргізуге мүмкіндік беретін формаға бұл талаптарды тіркейді.

Қосылатын байланыс – сол функцияның орындалғанынан басқа прецеденттің анықталғандығын және осы прецедентке басқа да бір прецеденттердің біріккенін көрсететін байланыс.

Кеңейтілген байланыс – шартты емес жағдайда туындайтын міндетті емес функцияның жиынтығынан туындайтын байланыс.

Біздің мақсат талаптарды орналастыру сатысын толық қарастыру – яғни бірінші сатыны. Бұл сатыда программалық жүйеге, қолданушылардың ұсынған талаптары қарастырылады:

– жүйе ақпаратты іздеуге және өңдеуге кеткен уақыт мөлшерін азайту қажет;

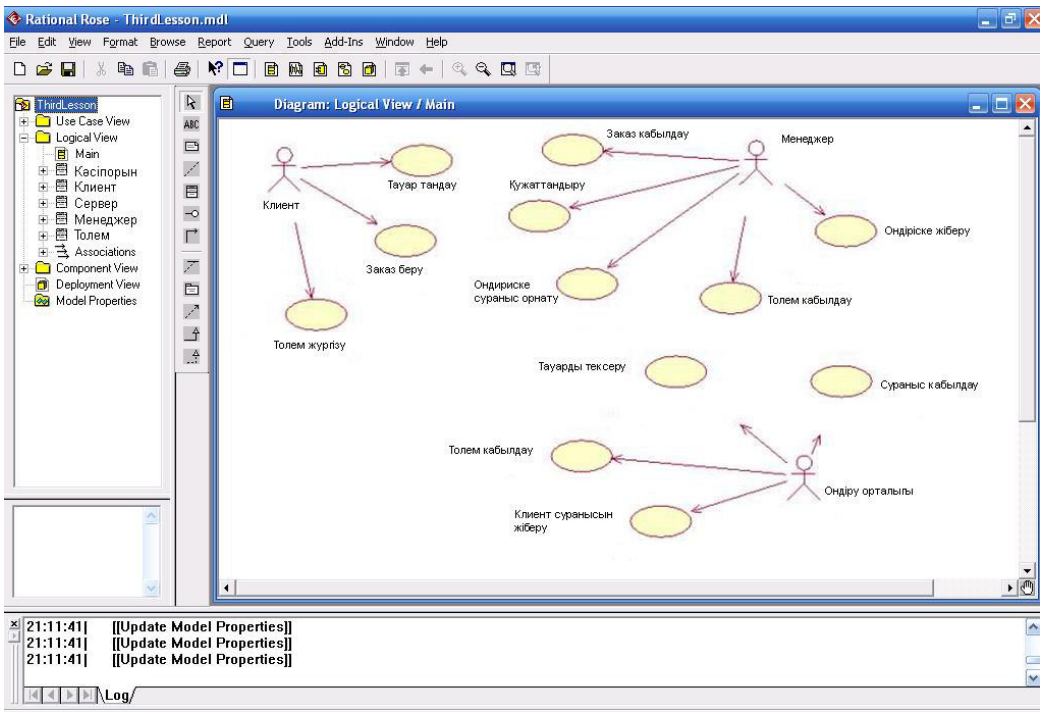
– қолданушы компьютерді минимальді түрде білгендіктен программалық жүйені оңай қолдана алу мүмкіндігі;

– деректер қорын мәліметтермен толықтыру, өзгерту, өшіру және таңдаған мәліметтің есебін алу мүмкіндіктерінің болуы;

– бірнеше қолданушы бір мезгілде жұмыс істей алатын клиент-серверлік, желі арқылы байланыста болу мүмкіндігі. Прецеденттер диаграммасы 2.10-суретте көрсетілген.

Қолдану вариантты (Use Case) жүйе мен активті субьект арасындағы диалогты модельдеуге мүмкіндік береді және функцияны соңында бейнелейді. Жүйені қолдану вариантты жиыны оны қолданудың көптеген тәсілдер назарына еңбегі сіңген.

Қолдану вариантты – транзакция жүйесімен орында латын тізбектілік, бұл арқылы анықталған активті субьект қызыққан нәтижені алуға болады. Прецеденттер диаграммасы қолданушы көзқарасы бойынша жүйе тәртібін анықтайды.



2.10-сурет – Прецеденттер диаграммасы

3 Жүзеге асыру және тестілеу бөлімі

3.1 Қолданушыға мәлімет

Delphi ортасында автоматтандырылған ақпараттық басқару жүйесін құру кезінде қойылатын ең бірінші негізгі талап – ең көп таралаған платформа – Windows платформасымен сәйкестігі. Екіншісі–қолдану кезінде пайдаланушыға ыңғайлы болуы және артық ақпарат еңгізу көрсеткішін азайту. Программа интерфейсі кез келген деңгейдегі қолданушы үшін программаны басқару кезінде қиындықтар тудырмайтындай ұйымдастырылу керек. Мәліметтер қорының операциялық жүйеге сәйкес болатын форматта берілу керек және дұрыс жұмыс жасау үшін қосымша элементтерді (кітапханаларды, драйверлерді, кодектерді) қажет етпейтін болу керек.

Айтылған талаптарды ескере отырып, қойылған есепті шешудің келесі инструменталдық құралдары мен әдістері анықталды:

– инструменталдық жүйе ретінде Borland Delphi таңдалынды, себебі бұл программалық пакет Windows үшін программалар құруға мүмкіндік береді. Компиляция кезінде дайын программа орындалатын бір файлға орналастырылады да, басқа дербес компьютерлерде негізгі компоненттреді пайдалану кезінде қосымша кітапханаларды талап етпейді. Borland Delphi программалық пакеті біздің техникалық тапсырманың барлық пункттеріне толық сәйкес келеді;

– деректер қорын сақтау форматы ретінде SQL Server таңдалынды. Себебі бұл формат универсалды болып табылады және барлық аппараттық және программалық платформалармен қолданысқа келе алады, ал қондыру барысында клиент-сервер жүйесімен қондырылып, басқарылады. Бұл формат та техникалық тапсырманың барлық пункттеріне сәйкес келеді – Windows-пен қолданылады, оның көмегімен кез-келген интерфейс мәселесін шешуге болады, сонымен қатар, оны қарау үшін қосымша программалық қамсыздандыруды талап етпейді.

Сонымен, қойылған талаптарды орындау үшін бізге келесі программалық қамсыздандыру қажет: Borland Delphi инструменталдық жүйесі. Қазіргі кезде программистерге Delphi пакетінің келесі нұсқасын - Delphi 10 Studio пайдалану мүмкін болып отыр. Алдыңғы нұсқалары сияқты Borland Delphi 10 Studio әр түрлі программалар құруға мүмкіндік береді: қарапайым бір терезелі қосымшадан бастап, таратылған мәліметтер қорын басқаруға болатын программаға дейін. Пакет құрамына мәліметтер қорымен, XML-құжаттармен жұмыс жасауға мүмкіндік беретін, анықтамалық жүйе құруға мүмкіндік беретін, басқа да есептер шешуге болатын әр түрлі утилиталар кіреді. Жетінші нұсқаның ерекшелігі .NET технологиясын қолдау болып табылады.SQL клиент-сервер мәліметтер қорымен байланыс орнатылуы да қарапайым болып келеді. DbExpress қосымшасы арқылы кестелердегі кілттік сөздердің сақталуы да жұмысты көпшілік қолданушыға ыңғайлатады.

Файл кеңейтімдері:

– .pas - модульдің бастапқы коды (pascal);
– .dpr - жобаның бастапқы коды (pascal);
– .dproj - жобаның бастапқы коды (xml);
– .dproj.local - жобаның бастапқы коды (xml);
– .dfm - бастапқы кодын қалыптастыру;
– .dpk - жинақталған бума;
– .dcu - құрастырылған модуль;
– .exe - құрастырылған бағдарлама;
– .res – ресурстар;
– .dsk - файлдарға байлау;
– .identcache - кэштелген файл байланысы.

Олардың мақсаттарына байланысты қасиеттер әртүрлі нысандарда қасиеттер терезесінде ұсынылуы мүмкін. Мысалы, мәндерінің диапазоны кіші болып табылатын сипаттар барлық осы мәндердің тізімделетін ашылмалы тізім түрінде жиі ұсынылады және олардың ішінен ең қолайлы таңдауды жеткілікті.

Бұл сипаттар, атап айтқанда, Түсті қамтиды. Ол пішіндегі кез келген басқару элементіне және пішіннің өзі үшін орнатылуы мүмкін. Сонымен қатар, ықтимал мәндер тізімінде, түстер атауына қосымша, қасиеттер терезесінің ашылмалы тізімінің оң жағында тиісті түске толтырылған шағын тіктөртбұрыш көрсетіледі.

BorderStyle сипаты (шекара мәнері) пішіндегі (немесе пішіннің өзіндік) элемент жиектерінің көрінісін орнатуға мүмкіндік береді. Мұндай шекаралар элементтен элементтерге әр түрлі болуы мүмкін, ал кейбір элементтер (мысалы, түймелер) олар жай ғана жоқ.

PopupMenu сипаты (мәтінмәндік мәзір) әзірлеуші жасаған жергілікті мәзірді белгілі бір элементке байланыстыруға мүмкіндік береді - оны тінтуірдің оң жақ түймешігімен басқанда шақырылады. Курсордың сипаты (курсор) курсордың түрін таңдауға мүмкіндік береді, ол өзгерте алатын кезде оған тиісті элемент. Бұл өзгеріс тек қана жұмыс істеп тұрған бағдарламада ғана пайда болады, дизайнерде әзірленген формамен емес. Қол жетімді курсор пішіндері ашылмалы тізімде графикалық түрде көрсетіледі.

Қосылған сипат (қосылған) пайдаланушыға әр түрлі элементтерді уақытша қолжетімді етуге мүмкіндік береді. Бұл сипат Boolean Boolean түріне ие және, тиісінше, екі True немесе False мәндерінің бірі бола алады. Жалған таңдалса, пайдаланушы осы басқару элементімен өзара әрекеттесе алмайды - түймені баса алмайды, қосқышты ауыстырады, мәзір элементін және т.с.с. Бірақ нысандағы элемент көрінетін болады, тек кейбір «пассивті» түрде болуы мүмкін.

Көрінетін сипат экрандағы элементтің көрінуін анықтайды. Егер оның мәні True болса, онда мұндай элемент дизайнерде көрінетін болса да, ол жұмыс жасалынған бағдарламада көрінбейді. Бірақ бұл сипатты элемент әсер етуі үшін пайдаланушыға қол жетімді ете бермей, жалған күйге аударылған Enabled сипатын бөліп алу керек, дегенмен оны терезеде көрінеді. Көрінетін сипат элементті жай көрінбейді.

AutoSize сипатын пайдаланып, элементтің өлшемін автоматты түрде баптау оның ішіндегі мәтіннің ұзындығына орнатуға болады. Бұл сипат, бірінші кезекте, жапсырма өрісі сияқты элементтерде талап етіледі, бірақ барлық элементтер жоқ, бірақ көбінесе мәтінді өңдеу және көрсету үшін пайдаланылады.

Кейбір сипаттар салынған құрылымға ие болуы және өздері кірістірілуі мүмкін. Мұндай сипаттар тізімнің сол жағындағы шағын плюс белгісі бар қасиеттер терезесінде бөлектеледі. Бұл белгішені басқанда, осы сипатқа қатысты қосымша сипаттардың кірістірілген тізімінде көрсетіледі. Әзірлеуші оларды ағымдағы элементтің жеке қасиеттерімен шатастырмауы үшін ғана олар оңға қарай біршама ауысады.

Кірістірілген сипаттар басқа нысандар мен басқару элементтері болып табылады. Ендірілген нысанды пайдаланған кезде, осы нысанның сипаттары қасиеттер терезесінде қосымша деңгейде көрсетіледі. Жиі басқа нысандарға салынған осындай сипаттың үлгі мысалы - Қаріп сипаты (қаріп). Дизайнердегі өріс жазуын таңдап, осы сипатты ашсаңыз, онда ол қандай қасиеттердің шын мәнінде болатындығын көре аласыз.

Шрифтпен байланысты Аты өрісінде, сәйкес қаріп профилін таңдауға болады. Size сипатын пайдалану - Bold және Italic сипаттарын (олар, өз кезегінде, Style сипатына кірістірілген) пайдаланып, өлшемін орнатыңыз, қаріптің қалың немесе көлбеу болуын және т.б. Бұл жағдайда көрініп тұрғандай, әртүрлі қасиеттердің атаулары әдетте өз мақсаттарын нақты көрсетеді және мүмкіндігінше, әр түрлі танымал бағдарламаларда қабылданған атаулармен стандартты белгілер ретінде сәйкес келеді.

Түс сипаты (шрифт түсі) басқаруға түсірілімдердің қандай түсі анықталатынын анықтайды. Сол және жоғарғы сипаттар пішіндегі элементтің орнын анықтайды (элементтің сол жақ жоғарғы бұрышының координаттары). Ені (ені) және биіктігі (биіктігі) элементтердің өлшемін орнатады. Орналасу сипаты пішін үшін орындалады. Ол ашылған кезде экранның қай жерде көрсетілетінін анықтауға мүмкіндік береді (компиляцияланған бағдарламаны іске қосқаннан кейін). Орналасу келесі мәндердің бірі болуы мүмкін:

– Тағайындалған. Пішін экранда оны дизайнерде дайындаған кезде экранда болатын дәл орнында көрсетіледі;

– PoDesktopCenter. Пішін жұмыс үстелінің шеттеріне қатысты ортақ болады;

– PoScreenCenter. Пішін экранның шеттеріне қатысты ортақ;

– PoDefaultPosOnly. Пішін Windows жүйесінде әдепкі жерде көрсетіледі;

– PoDefault. Пішін Windows жүйесінде әдепкіде көрсетіледі және терезе өлшемдері әдепкі бойынша Windows жүйесімен тең болады;

– PoOwnerFormCenter. Пішін ата-аналық терезеге қатысты ортаға келтіріледі.

Сіз Delphi-де бағдарламаны құруға кіріспес бұрын, қағаз парағына қарапайым нобай жасаңыз, сіздің тапсырмаңыздың қандай міндеттері шешілетінін және нәтиже болуы керек. Егер ең басында бағдарламаның негізгі

функцияларын түсінбейтін болсақ, онда сіз көп уақыт пен күш жұмсай аласыз, бірнеше рет қайталанатын нәрсе жасай аласыз. Клиенттің немесе басқа соңғы пайдаланушылардың пікірлерін ескеру қажет.

Бағдарламалық кодты теру үшін, құрылымдық кодты жазу үшін өзіңізді үйрету керек. Бағдарламаны құрастыру немесе іске қосу үшін кодты құрылымдау қажет емес екеніне қарамастан. Құрылымдық кодтың бар болуы бағдарламаны оқуды айтарлықтай жеңілдетеді.

Командада бағдарламашы ретінде жұмыс жасағанда және бірнеше әзірлеушілердің бағдарламаларын жасау кезінде жалпы ережелерді сақтау керек. Бір бағдарламашы кодты жазса, екіншісі отладтануды орындайды. Немесе алты ай жұмыс істеген бағдарламашы бағдарлама кодын жазған жағдайды елестетіп көрейік, ол өте маңызды бөлігі, оны алып тастап, кетіп қалады. Сондай-ақ, басқа кодты түсінуге қарағанда, пішімделмеген, аңғартпастан басқа кодты қайта жазу оңай болады.

Келесі кодты қарастырайық:

```
TForm1.Button1Click (жіберуші: TObject);
const TemperaturaKipenia = 100;
var TemperaturaNastoiascha: бүтін;
басы
TemperaturaNastoiascha: = 65;
TemperaturaNastoiascha <TemperaturaKipenia then ShowMessage ('Су піспеген!') else ShowMessage («Шайнекті пісірілген, газды өшіріңіз.»);
соңы;
```

Жазу кодын түсіну күрделі. Код үздіксіз мәтінмен жазылған және мәтінді шарлау қиын. Енді төмендегі коды қарастырайық. Мәтін таза және әдемі болды.

```
рәсім TForm1.Button1Click (жіберуші: TObject);
const
  TemperaturaKipenia = 100;
var
  TemperaturaNastoiascha: бүтін;
бастаңыз
  TemperaturaNastoiascha: = 65;
  егер ТемператураНастояска <TemperaturaKipenia болса
    ShowMessage ('Су әлі қайнатылған жоқ!')
  басқа
    ShowMessage ('Шайнек пісірілген, газды өшіріңіз.');
соңы;
```

Кодты құрылымдау ережелерін қалыптастыруға тырысамыз.

Кілтсөздер кодты жазғаннан кейін. Бұл жағдайда const, var, start, end, and else. Декларацияның const және var бөлімдерінде әр типті бөлек жолмен жазылады.

```
номер, кисло: бүтін;
itogo: нақты;
n1, n2: логикалық;
```

Кілт сөзге сілтеме жасайтын код оң жиектен шығарылады. Шегінісінің саны еркін таңдап алынады. Сіз бір бос орынды шектей аласыз немесе Tab пернесін пайдаланыңыз. Қосымша сызықтарды енгізу бағдарламаның көлемін арттырмайды және бағдарламаның көрінуін жақсартады. Белгілердің арасында айқындылық үшін бос орындар қою керек.

```
var
  i, j, k: бүтін;
бастаңыз
  // Бастапқы мәндерді беріңіз
  i: = 1;
  j: = 0;
  k: = 3;
// есептеу блогы
  j: = i + k +3;
....
// Барлығы
....
соңы;
```

Код бос жолдармен үш бөлікке бөлінеді. Әрбір бөлім түсініктемеден басталады. Түсіндірмелер өзіңіздің және тіпті біреуіңіздің кодын оқуға көмектеседі, себебі уақыт өте көп ұмытып кетеді және ескертулердің бірнеше жолы есептеулердің мәнін есте сақтауға көмектеседі.

Өзім үйреткен ең бастысы - құрылымдық кодты жазу болды. Үлгі кодын қарастырайық. Ол қатты мәтінде тураланбастан жазылған құрылымдық емес кодты ұсынады. Қазір бұл жерде не болып жатқанын түсінуге тырысыңыз. Түсіну қиын.

Құрылыссыз код:

```
TForml.ButtonlClick (жіберуші: TObject);
const tt = 12;
var i-Integer;
boolean;
бастаңыз
i: = 1212 * 12;
егер> 10,000 болса
бастаңыз
b. = шын;
i: = 1212 * 2;
егер i <12 болса
b: = false;
соңы басқа
b: = false;
соңы;
```

Енді жаңа коды қараңыз, онда барлық ережелер ережелерге сәйкес реттеледі.

Құрылымдық код:

TForml.ButtonlClickCSender: TObject): const

tt = 12:

var

i: Integer;

b: boolean: бастау

i 1212 * 12; егер> 10000 болса, онда бастаңыз

b: = шын:

i: = 1212 * 2;

егер i <12 болса

b: = false;

соңы

b: = false;

соңы;

Мәтін әдемі және түсінікті болды. Нормаланған нәрсеге қарап көрейік.

Кілт сөздерден кейін код жоқ. Осылайша, const және var сөздерден кейін, жарнамалар келесі жолға жылжытылады. Кодексті жасау кезінде осы ережені ұстану керек.

Бар және const бөлімдерінде әрбір айнымалы түрі өз сызығында жарияланады. Көптеген элементтер бар болғанда, сізге қажет нәрселерді табу оңай болады, егер айнымалы бірнеше декларациялар болса, олар келесідей реттелуі мүмкін:

Индекс. Бүтін сан: b. t boolean;

Нақты;

Бұл мысалда, тип атаулары бірдей деңгейде, ал жарнамалар жағымды көрінеді. Бірақ сол түрдегі тым көп айнымалы мәндер болған кезде код қалай бұзылады:

Индекс бүтіндігі: аты, тегі. Телефон. Мекенжай жолдары;

b. t boolean;

нақты real;

Айнымалы атаулар мен түрлер арасында үлкен бос орындар әдемі көрінеді. Мұндай жағдайларда айнымалы атаулар келесідей бірнеше жолға бөлінеді:

Индекс бүтіндігі;

Атауы. Тегі.

Телефон. Мекенжай жолдары;

b, t boolean;

нақты real;

Бос орындардың саны кішірек, бірақ оқылу мүмкіндігі енді бірдей емес, сондықтан мен сирек айнымалы типті аттарды бір деңгейде орналастырады. Егер сіз басқа біреудің кодымен немесе кодты ұзақ уақыт ұмытқан болсаңыз, онда соңғы сәтте айнымалы мәндер есепке алынады. Көбінесе олардың түрлерін тікелей код немесе префикс арқылы түсінуге болады

(айнымалылардың атаулары дұрыс деп аталса), бірақ кейінірек атау туралы айтуға болады.

Сонымен қатар, кілт сөзге қатысты барлық код шетінен шығарылады. Сіз өзіңіздің шегінісін таңдауыңыз керек. Кейбіреулер екі немесе үш бос орын қояды немесе Tab пернесін шегініс үшін пайдаланады және мен бір ғана бос орынды қолданады. Себебі, егер кілт сөздерді тым көп орналастыру басталса, егер басқалар болса, онда оңға қарай өту өте үлкен және ыңғайсыз, себебі бөлік терезенің оң жақ жиегінен жасырылады. Бұл жағдайда терезеге кірмейтін кодты көру үшін әрқашан айналдыруды солға және оңға жылжыту керек.

Кілт сөзді және тиісті соңынан кейін бүкіл код оң жаққа бір бос орынға ауысады. Осылайша, сіз қай оператордың қайсысына сәйкес келетінін көре аласыз.

Кілт операторларынан кейін, егер қайталап және қайталаса, код тек жаңа жолда және офсетпен жазылады. Барлығы бір жолға сәйкес келуге тырыспаңыз. Алынған бағдарлама аз болмайды, бірақ отладтану кезінде не болып жатқанын түсіну қиын болады.

Менің дизайнерлік әдісімнің баламасы ретінде тағы бір нәрсе бар.

Құрылымдық код

```
TForml.ButtonlClick (жіберуші: TObject). const
tt = 12;
var
i. Бүтін сан;
b: логикалық;
бастаңыз
i: = 1212 * 12; i> 10000 болса, b бастаңыз: = true;
i: = 1212 * 2, егер i <12 болса, b: = false;
соңы
b: = false;
соңы;
```

Бұл жағдайда Tab пернесі шегініс үшін пайдаланылады. Код кішкентай болғанда, ол тіпті анық болады, бірақ егер бір жолда әртүрлі жазбалар (айнымалылардың ұзын есімдері немесе бірнеше салыстыру) болса, онда, жоғарыда айтылғандай, мәтіннің экранның оң жақ жиегінен тыс «кетеді» және бүкіл сызықты толық көру мүмкін емес.

Сонымен қатар, бастауыш кілт сөз бірдей жолда жазылғаннан кейін, егер, егер, уақытша және т.б. Кілт сөздің соңы осы бастау операторымен бірдей деңгейге қойылады. Бұл дизайн Java бағдарламалау тілінде өте жиі пайдаланылады, бірақ бір айырмашылық бар - егер if операторларының басталуы мен аяқталуы әрдайым қойылмаса.

Операторлардың n және end операторлары i f сияқты деңгейде орналасуы мүмкін:

```
егер> 10000 болса, онда бастаңыз
b: = шын:
i: = 1212 * 2;
```

i <12 болса, b: = false;

Соңында, менің ойымша, кем дегенде бір кеңістікті ауыстыру анық

Тіпті бір жолға бәріне сай келуге тырыспаңыз. Кодты көлденең қарау уақытты талап етеді және бұл шығындар мүлдем пайдасыз. Мысалы, егер функцияның 10 параметрі болса, оның қоңырауы келесідей жазылуы мүмкін:

Function_Name (Параметр1, Параметр2.

Параметр 3

...):

Егер сіз бәрін бір жолда жазсаңыз, функционалды шақыру экранның көрінетін бөлігіне сәйкес келмеуі мүмкін және сіз соңғы параметрлерді көру үшін айналдыруыңыз керек.

Сіздің кодыңызға бос жолдарды салуға тырысыңыз. Бұл кодтың өлшеміне әсер етпейді, сондықтан кеңістіктер мен бос жолдардың саны ешқандай жанама әсер етпестен көрінуді жақсартады. Кодты толық мәтінді бірыңғай кеңістікте жазылмаған оқу өте қиын.

Құрылымдық код:

var

i: бүтін сан: j: бүтін;

бастаңыз

// Бастапқы мәндер i: = 0; j: = 1;

// есептеуді i: = 10 * j + 5: j: = 20-I;

// Нәтижені көрсету

соңы;

Код бос жолдар бойынша үш бөлікке бөлінеді: баптандыру, нәтижені есептеу және шығару.

Әрбір бөлік мәтінді түсінуге көмектесетін түсініктеме. Егер сіз өзіңіздің кодты өзіңіз жазған болсаңыз да, бірнеше айда не істелетінін және неге екенін есте сақтау өте қиын болады. Бағдарлама мүмкіндігі әрдайым жетілдірілуде және алты ай ішінде, мүмкін, кодты басқа жолмен жазасыз, өйткені айнымалыларды жобалауға немесе атауға ыңғайлы әдісті табасыз. Ия, және Windows API (қосымшалар бағдарламасының интерфейсі - қолданбалы программалау интерфейсі) немесе VCL (Visual Compo nent Library - компоненттердің көрнекі кітапханасы) барлық функцияларын есте сақтау мүмкін емес. Әрине, ағылшын тілін жақсы меңгерген болсаңыз, мақсатыңызды атаумен анықтауға болады, бірақ Funcl, Func2, т.б. функцияларыңызды шақыратын болсаңыз, онда мұндай код түсініктеме берілмейді.

Қалай болғанда да, алынған кодқа қатысты түсініктемелер әсер етпейді, сондықтан олар артық болмайды. Егер менің кейбір процедураларым 10-нан астам кодтық жолдан тұрса, мен міндетті түрде түсініктеме жазамын. Бұл код ұзақ уақыт өткеннен кейін де оқуға көмектеседі және қателерді жылдамырақ табуға көмектеседі. Бағдарламаның даму кезеңінде болашақты ойлап көріңіз.

Бос жолдар немесе бос жолдар сияқты бос орындар жинақтың нәтижесі нәтижесінде алынған файлдың өлшеміне әсер етпейді, бірақ оқылуды арттырады. Келесі кодты қараңыз:

Парам: = Paraml + NetComonent.Count-Form.Wi dth;

Парам2: = Form.Hei ght + Screen.Width / 2 * MAX_SELECT:

Бұл кодтың мәнін түсінуге тырыспаңыз, себебі ол жоқ. Мағынасы - оқу өте қиын. Меніңше, бәрі дұрыс жазылған, бірақ операторлар көрінбейді. Әр айнымалы және әрбір таңба кеңістікпен бөлінуі керек:

Парам: = Paraml + NetComonent Count - Form.width;

Парам2: = Form.Height + экран ені / 2 * MAX_SELECT:

Мұндай жазбаларды түсіну оңайырақ деп келісемін.

Код жасау - дизайнерлік жұмыс. Дизайнер, мысалы, әдемі сайтты жасайды, ол пайдаланушымен жұмыс істеуге ыңғайлы болады. Бағдарламашы бағдарлама кодын онымен жұмыс істеу ыңғайлы және жағымды етіп реттеуге тиіс. Дизайнерлік ережелердің бірі - бос кеңістіктің болуы. Егер үстелде тек қалам мен кітап болса, олар бірден көрінеді. Бірақ, бұған қоса, компакт-дискілердің және басқа да заттардың «қабырғасы» болса, онда тұтқаны табу қиын болады. Бос орындар мен бос жолдар кодта бос бос орындар жасайды және элементтерді бір-бірінен бөліп, элементтердің жақсы көрінуіне қол жеткізеді.

Егер сіз жұмыс үстелінде жұмыс істеуге ыңғайлы болса, онда бәрі демпингтік және шашыраңқы болса, онда код бірдей болады. Дегенмен, дұрыс салынбаған кестеде қанша уақыт жұмсап жатқанын санай аласыз. Осыдан кейін тазалау сөзсіз. Кодқа тапсырыс беру қиын, сондықтан бағдарламашыны «араласпау» және «қоқыс» жасау керек. Сіз әдемі бастапқы кодты жазуыңыз керек және ол шынымен сапасына әсер етеді.

Мен басқа біреудің кодымен жұмыс істей бастадым (жұмысымда жиі жасауым керек), ең алдымен коды қайта қолмен қайта жасаймын. Форматтау кезінде бағдарламашы менің араласуымның алдында не айтқысы келетінін қорыту өте оңай. Егер бәрі менің ережелерім бойынша жасалса, онда оны оқу өте оңай және кодты түсіну үшін әлдеқайда аз уақыт кетеді.

Ең үлкен қиындық түрлі стилде безендірілген модульдермен байланысты. Сіз үнемі қайта құруыңыз керек, сондықтан сіз бәрін толықтай қайта пішімдеуіңіз керек немесе жағдайды шатастырмау үшін бірдей стильде жазыңыз.

Жақында Windows үшін бастапқы кодтың көбі ұрланған, мен кейбір бөліктерді көре алдым. Мен көрген нәрселер сол заңдарға сәйкес жасалған, бұл кез келген бағдарламашы жұмысқа тез араласуға мүмкіндік береді. Мен бағдарламашылардың дұрыс емес дизайны үшін қаржылық айыппұлдар алғанын естідім, өйткені бұл әзіл емес, себебі, тіпті ресейлік кәсіпорындарда бірнеше адамның қатысуымен үлкен бағдарламаларды әзірлеу кезінде ұқсас ережелер бар. Бұл дұрыс шешім, ал бағдарламашыларды кодты дұрыс жазуға үйретеді.

Жаңашыл бағдарламашылар жазған бастапқы кодта көптеген қателер мен оңтайландырылмаған бөлімдер болды. Бұған қарамастан, барлығы құрылымдалған көріністі (бірнеше адам бағдарламада жұмыс істеген кезде ғана ескі бөліктер ғана ережелерсіз безендірілген болатын). Белгілі дизайн

ережелерінің арқасында, бағдарламашылардың үлкен тобы жоба бойынша табысты жұмыс жасайды, тіпті өте нашар жазылған кодты оңай жөндеуге және қажет болғанда жұмыс істеуге болады.

Кейбіреулер фирма кейбір жақсы бағдарламашыларды жалдауға болатынын және кодты ешқандай дизайнсыз жақсартқанын айтады, бірақ бұл қате. Біріншіден, осындай бағдарламашыларды табу өте қиын, олардың жалақысы әлдеқайда жоғары. Екіншіден, барлық «бір өлшемге» сай келмеу мүмкін емес. Екі өте жақсы бағдарламашылар әр түрлі жаза алады, өйткені әрбір мәселе он жолмен шешілуі мүмкін. Сондықтан, қызметкерлердің біліктілігіне қарамастан, кез-келген үлкен жобада айқын жобалау ережелері қажет.

Барлық жады өндірушілер әртүрлі өлшемдер мен пішіндерді еске түсіру жолын бастаған жағдайды елестетіңіз. Әрқайсысы оны тиімді және әдемі етеді, бірақ бірдей коннекторға түрлі басқарушыларды кірістіру мүмкін емес. Бұған жол бермеу үшін негізгі бағыттар бәрін бірдей етіп жасаған кезде стандарттау ойлап тапты. Стандарттау - өткеннің қалдықтары емес, бүкіл әлемнің болашағы. Егер бәрі өз бағыттарын тартып алса, онда біз «аққу, қатерлі ісік және шортан» әсер етпейміз.

Мен әртүрлі дизайнерлік әдістерді көрсетуге тырыстым, сіз өзіңіздің таңдауыңызды таңдауыңыз керек. Мүмкін, ол тағы бір нәрсе болады, бірақ ең бастысы, кодты дәл солай жасауға және оқуға оңай болуы керек.

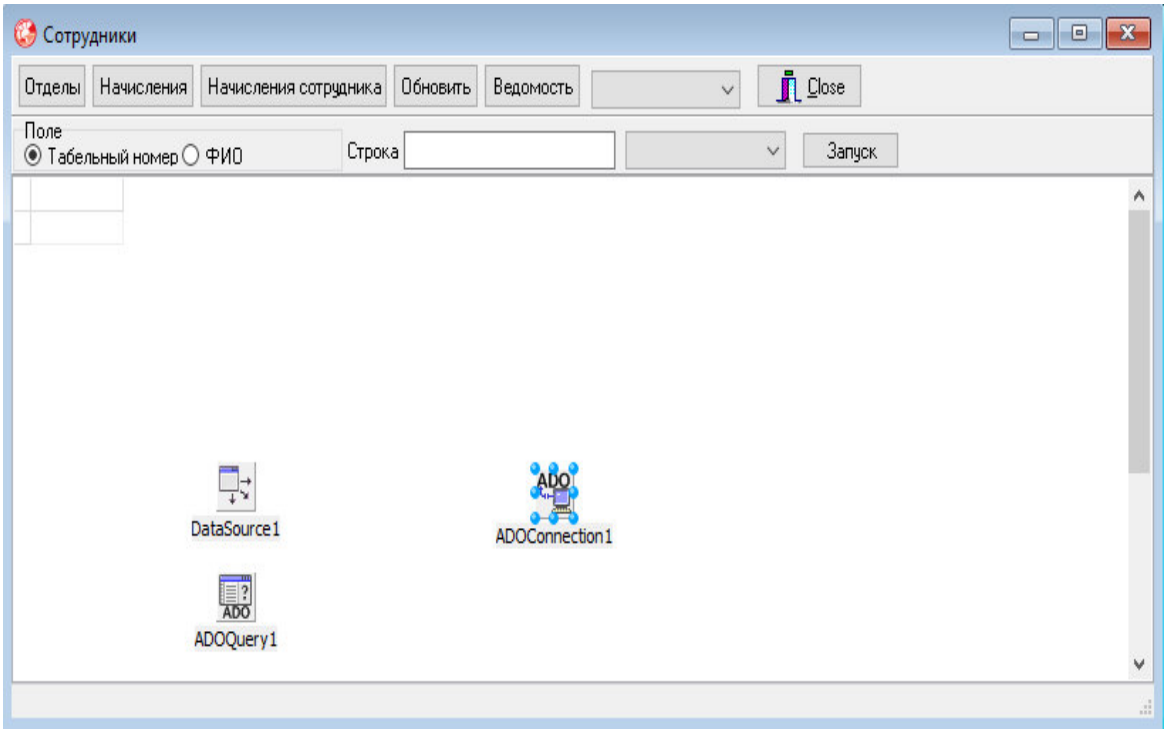
3.2 Тапсырма қойылымы

Бағдарламалық өнімді жасамас бұрын бағдарламалау ортасын таңдауымыз қажет. Сондықтан жасалынатын ақпараттық жүйені жетік тексере білуіміз керек.

Бағдарлама бірнеше терезеден тұрады, олардың әр қайсысының өз қызметтері бар. Windows-тың кез-келген операциялық жүйесімен ашылатын .exe файл арқылы іске қосамыз. Ең бірінші бағдарламаны ашқандағы қосылған терезе басты терезе болып табылады, ол терезеде бағдарламаға тіркелген мәліметтер қоры ашылған.

Деректер қорымен жұмыс істеуге арналған Delphi компоненттері SQL және клиент-сервер архитектурасымен жұмыс істеуге арналған. Олармен жұмыс істеген кезде қашықтағы серверлердің озық қолдау мүмкіндіктерін пайдалана аласыз. Delphi бұл қолдауды екі жолмен ұсынады. Біріншіден, Delphi тікелей командалары әзірлеушіге кестелерді басқаруға, шектеулер қоюға, жоюға, бар жазбаларды кірістіруге және өңдеуге мүмкіндік береді. Екінші әдіс - SQL тілінде сұрауларды сұрау, оны талдау, оңтайландыру, нәтижелерді қайтару және беру үшін серверге жіберілетін сұрау.

Дерекқорға қосылу үшін BDE технологиясындағы сияқты бүркеншік аттарды жасаудың қажеті жоқ - TADOConnection құрамдас бөлігімен қосылымды дұрыс конфигурациялау жеткілікті.



3.1-сурет – Бағдарламаның басты терезесі

TADOConnection компоненті келесі негізгі қасиеттерді қамтиды:

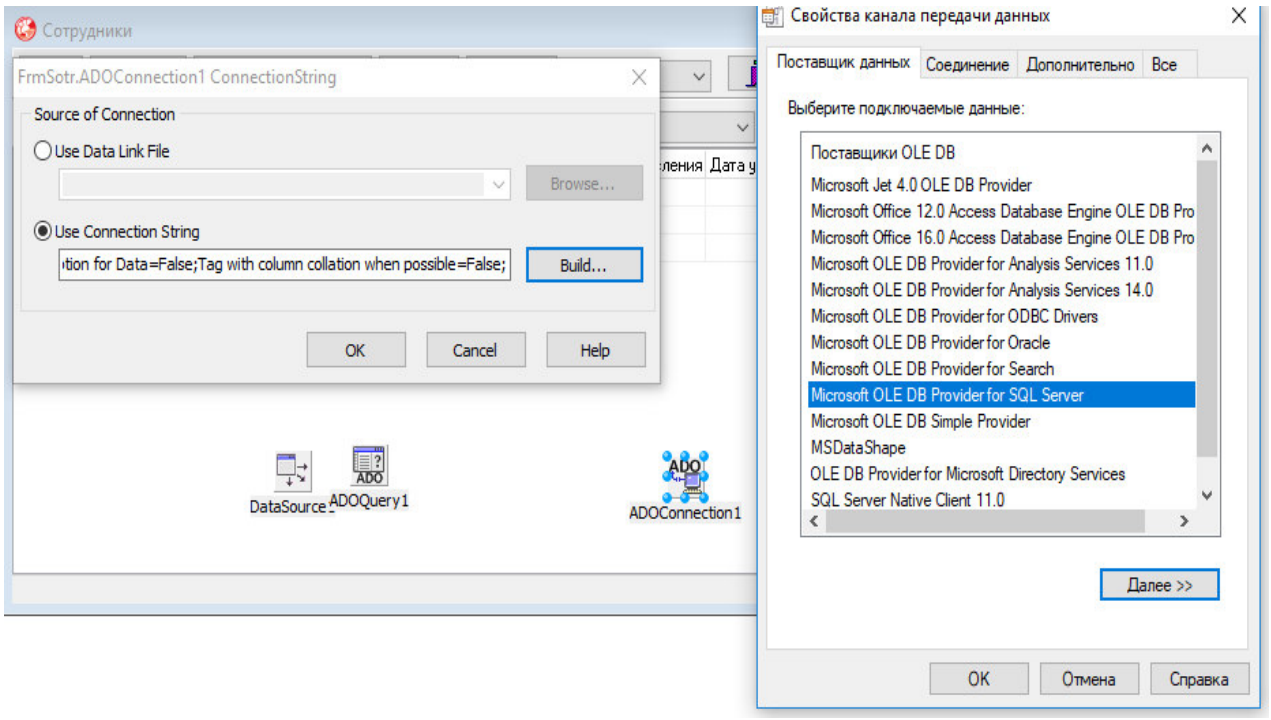
– ConnectionString – дерекқорға қосылу параметрлерін орнатуға мүмкіндік береді. Параметрлерді орнатудың ыңғайлығы үшін арнайы шеберді пайдалануға болады;

– Connected – мән шын мәніне орнатылғанда дерекқордың қосылымын іске қосады;

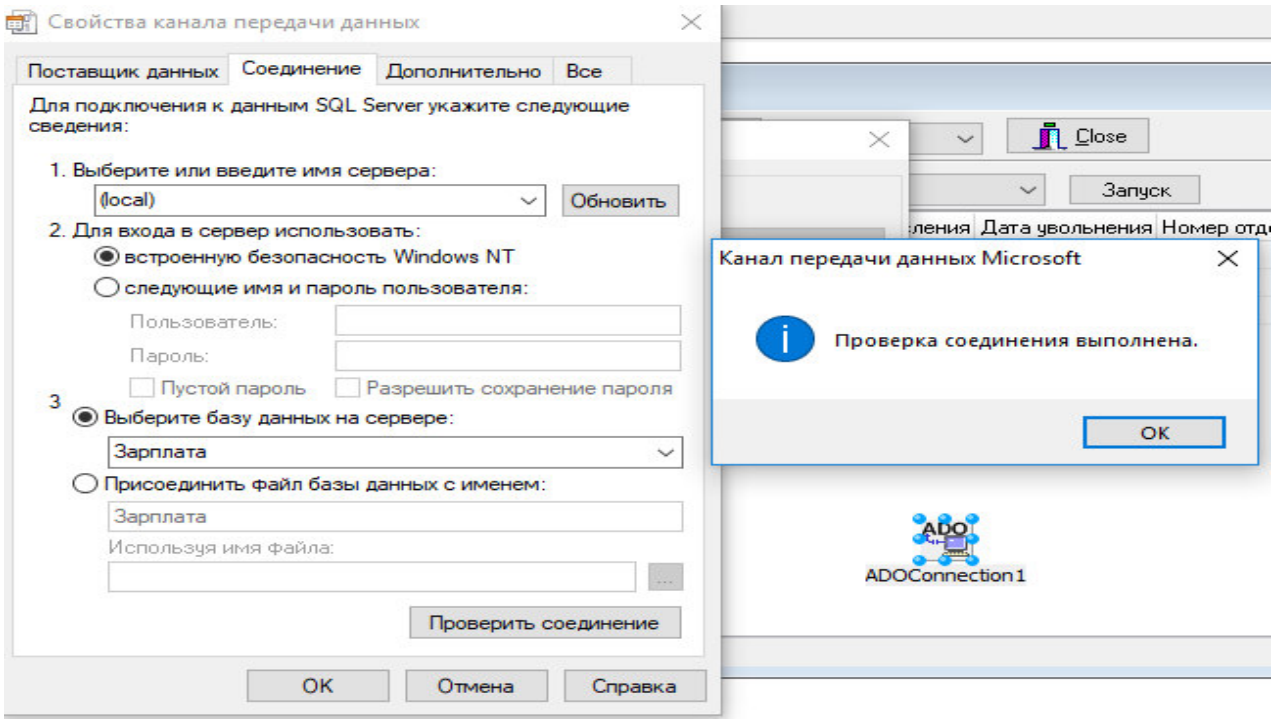
– LoginPrompt. Егер «false» мәніне орнатылса, дерекқорға қосылу пайдаланушы атын және паролін енгізуді сұрайтын диалогтық терезені көрсетпейді. Аты мен құпия сөз ConnectionString сипатынан алынады. Егер шын мәніне орнатылса, пайдаланушы аты мен құпиясөз диалогтық терезеде әрқашан сұралады.

Дерекқорды жеткізу жүйесімен байланыс орнатуды қалайсыз ба? TADOConnection компоненті арқылы қосылымды орнатыңыз. Компонентті екі рет басу арқылы қосылым шеберіне өтіңіз. Ашылған терезеде Connection String радиосын пайдалану түймесін таңдап, Build (Құрастыру) түймешігін басыңыз (Сіз деректерді сілтеме файлының радио түймесін пайдалану арқылы арнайы * .udl файлынан байланыс параметрлерін жүктей аласыз).

Одан кейін, Деректерді жеткізуші қойындысындағы тілқатысу терезесінде, дерекқордың провайдер түрін таңдаңыз, біздің жағдайда SQL Server үшін Microsoft OLE DB жеткізушісі. Қосылым қойындысында дерекқордың қосылу параметрлерін орнатамыз: SQL серверінің мекенжайы, пайдаланушы аты және құпия сөзі, дерекқор атауы. Параметрлерді орнатқаннан кейін OK түймешігін басыңыз.



3.2-сурет – ADOConnection компоненті арқылы SQL Server-ге қосылу



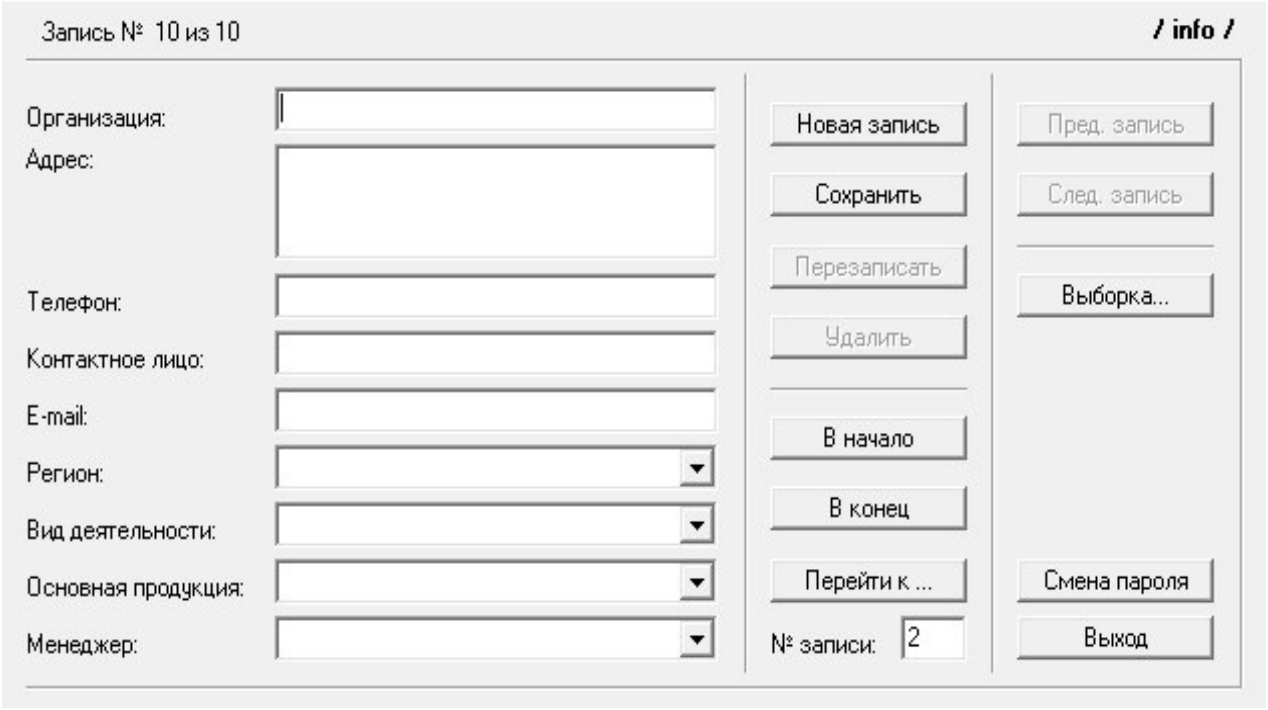
3.3-сурет – Delphi-ді SQL Server-ге қосу және оны тексеру

Егер сіздің қосымшаларыңызда SQL қолданатын болсаңыз, онда TQuery компонентімен танысуға міндетті боласыз. TQuery және TTable компоненттері TDataset-дан мұраланған. TDataset дерекқорларға қол жеткізу үшін қажетті функционалдылықты қамтамасыз етеді. Осылайша, TQuery және TTable

компоненттері көптеген жалпы сипаттамаларға ие. Сол TDatasource визуалды компоненттерде көрсету үшін деректерді дайындау үшін пайдаланылады. Сондай-ақ, қай серверге және дерекқорға кіруіңіз керектігін анықтау үшін бүркеншік атты көрсетіңіз. Бұл TQuery нысанының aliasName сипатын орнату арқылы орындалуы керек. Басты терезеде келітірілген мәліметтер қорымен жұмыстың жалпы мүмкіндіктері көрсетілген. Яғни, терезеде «анықтама», «өзгерту», «толығырақ», «фильтрациялау» сияқты батырмалар бар.

Әрбір батырманың қандай класс пен қандай формада қолданылғаны туралы мәлімет бағдарламаның листінгісінде, А қосымшада көрсетілген.

Толығырақ батырмансын басу барысында шығатын терезе:



3.4-сурет – Жаңа мәлімет енгізу терезесі

Жұмысшы туралы мәлімет терезесі DataBase қосымшасымен тіркелген кесте мен бірнеше форма-элементтен құралған. Бір студенттен келесі студентке ауысу батырмалары қайта шығып, басқаны таңдау сияқты іс-әрекеттен арылып, қолданушының жұмысын жылдамдатады. Толығырақ терезесіндегі мәліметті оңай өзгертуге болады. Ол үшін басты бет терезесіне көшіп, өзгерту батырмасын басу керек. Сондағы ашылған терезе жаңа ақпарат еңгізуге, келтірілген ақпаратты өңдеп реттеуге мүмкіндік береді.

Әрбір бөлімшелер туралы, соның ішінде мамандық-группа-пән сияқты мәліметті алу үшін анықтама батырмасын басуға болады. Ол кезде келесі терезе ашылады. DataBase қосымшасының мүмкіндіктері арқылы да бұл терезеде келтірілген мәліметтер жалпы қорға сақталады және мәліметті қосуға, өзгертуге, жоюға болады.

name	address	phone	e-mail	tovary
Asia_mebel	Bokei_109	2394565	asia_mebel	DSP_DVP
Art_mebel	Brus_103	2349900	art_mebel	Furnitura
Paks_metal	Tolebi_92	2348745	paks_metal	Metal
Mustafa	Saina_201	2335879	iMusi	stellazh
DIA	Seiful_84	2347845	DIAvitrin	vitriny
Teksan	Ryskul_25	2345500	iTeksan	Svet
TMC	Makat_124	2346688	TMCplus	torg_ob
Admart	Satpaeva	2347777	Admart	reklama
RSadver	Satpaeva	2944631	RSA_kz	orakal
Softgroup	Moskva	9652546	Soft_G	kassa_ap
EKO	Moskva	9657845	ICE_G	holod_ap
MXM	loskhkar	9648822	MXM.ru	holod_ap
Neitral	loshkar	9651245	neitral	metal
Mebel_cent	Egizbai	2746254	mebel_c	mebel
Ar_jan	Ratush_12	2384565	ar_jan	raspil
Ulzhan	Toktybai	2343654	mir_stekla	steklo

3.5-сурет – SQL Server-де құрылған серіктестер туралы мәліметтер қоры

Column Name	Data Type	Allow Nulls
id_postavshik	int	☐
name	nchar(20)	☐
address	nchar(10)	☐
phone	nchar(10)	☐
[e-mail]	nchar(10)	☐
tovary	nchar(10)	☐

3.6-сурет – Серіктестер туралы анықтама терезесі

Column Name	Data Type	Allow Nulls
id_contract	int	☐
klient_id	int	☐
postavshik_id	int	☐
date	date	☐
time	int	☐

3.7-сурет – Келісім-шарт туралы анықтама терезесі

Тұрақты сұрауды орындауға мүмкіндік беретін бағдарлама жасадым. Мұны істеу үшін келесі әрекеттерді орындадым:

а) Жаңа каталогқа жаңа жоба ашу;

б) Біз статикалық сұрауды ұйымдастырамыз. Ол үшін:

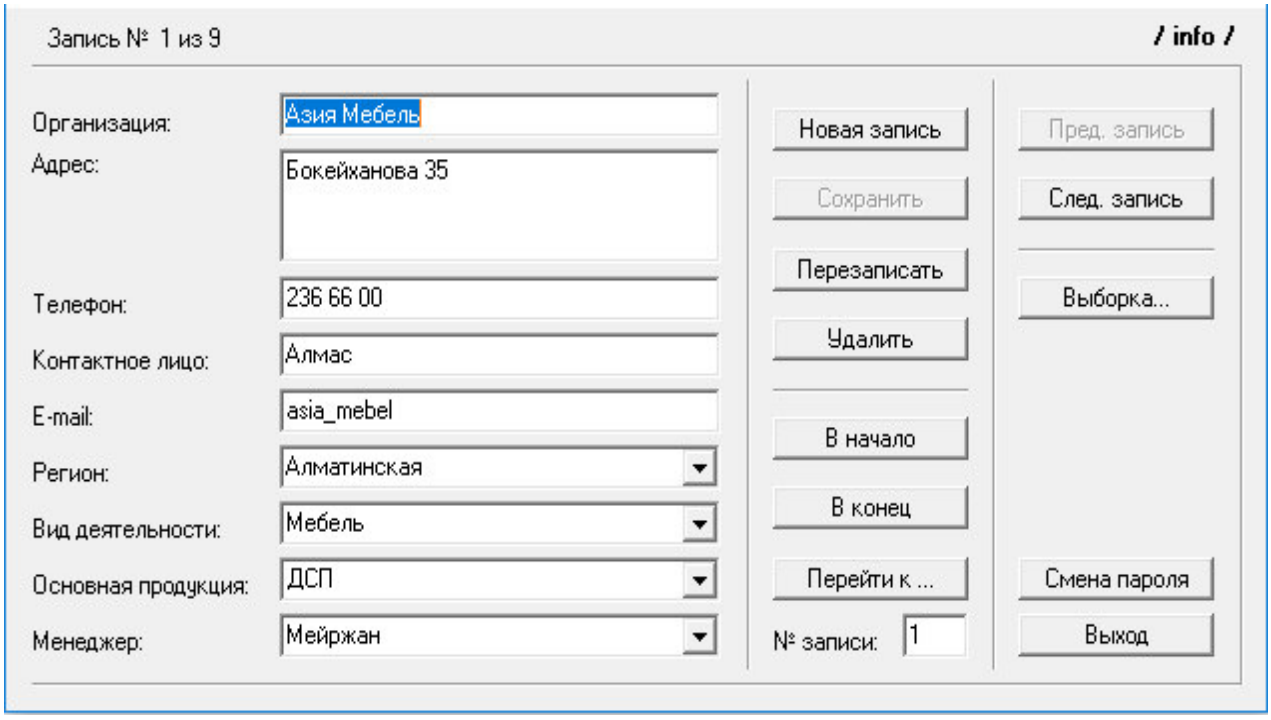
1) TADOConnection, TADOQuery, TDataSource және TDBGrid компоненттерін пішінге орналастыру, оларға тиісті атауларды бердім;

2) оларды біріктірдім;

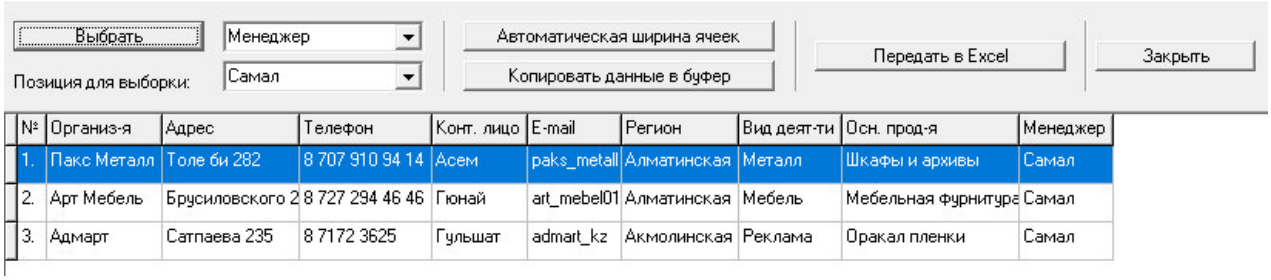
3) ADOConnection компонентін Supply DB-мен байланыстырдым;

4) QueryStatic компонентінің SQL сипатын сұрау мәтінін оған енгіздім;

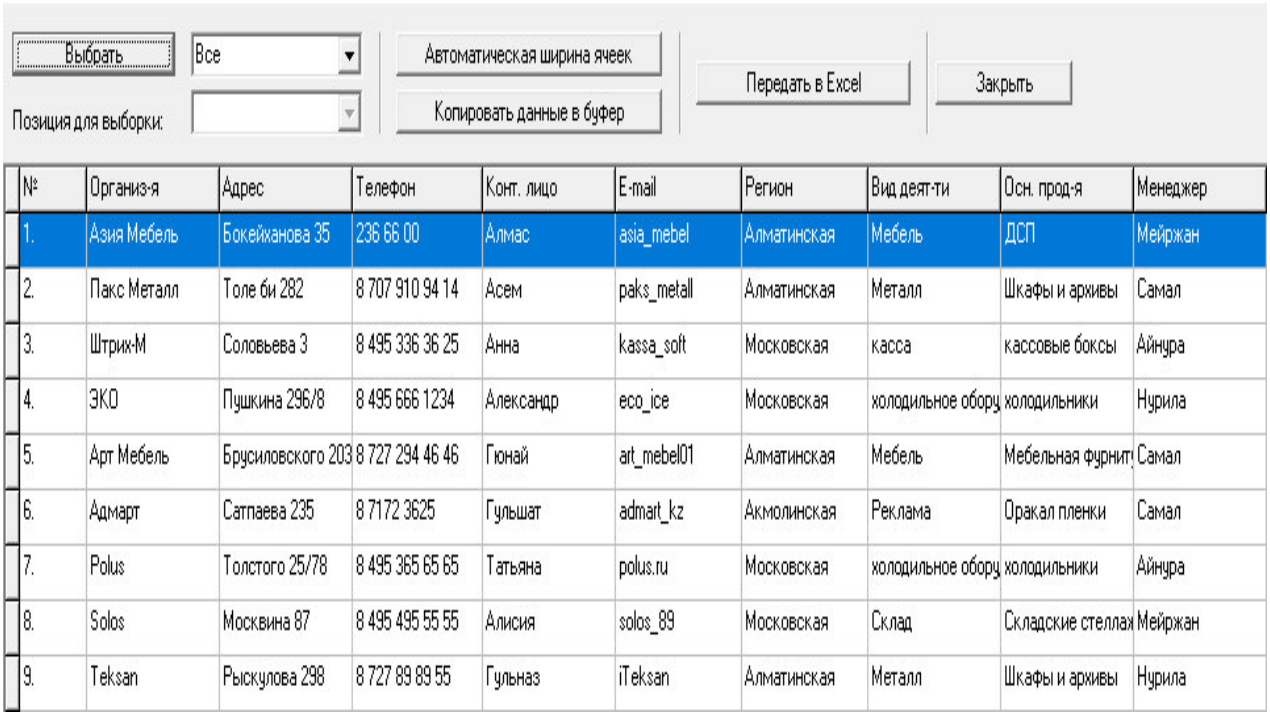
5) Әрбір бағанға жаңа элементтер қосуға, өңдеуге және сұрыптауға болады.



3.8-сурет – Жаңа мәлімет қосу бөлімі туралы анықтама терезесі



3.9-сурет – Таңдау арқылы ашылған анықтама терезесі



№	Организ-я	Адрес	Телефон	Конт. лицо	E-mail	Регион	Вид деят-ти	Осн. прод-я	Менеджер
1.	Азия Мебель	Бокейханова 35	236 66 00	Алмас	asia_mebel	Алматинская	Мебель	ДСП	Мейржан
2.	Пакс Металл	Толе би 282	8 707 910 94 14	Асем	paks_metall	Алматинская	Металл	Шкафы и архивы	Самал
3.	Штрих-М	Соловьева 3	8 495 336 36 25	Анна	kassa_soft	Московская	касса	кассовые боксы	Айнура
4.	ЭКО	Пушкина 296/8	8 495 666 1234	Александр	eco_ice	Московская	холодильное обору	холодильники	Нурила
5.	Арт Мебель	Брусиловского 203	8 727 294 46 46	Гюнай	art_mebel01	Алматинская	Мебель	Мебельная фурнит	Самал
6.	Адмарт	Сатпаева 235	8 7172 3625	Гульшат	admart_kz	Акмолинская	Реклама	Оракал пленки	Самал
7.	Polus	Толстого 25/78	8 495 365 65 65	Татьяна	polus.ru	Московская	холодильное обору	холодильники	Айнура
8.	Solos	Москвина 87	8 495 495 55 55	Алисия	solos_89	Московская	Склад	Складские стеллаж	Мейржан
9.	Teksan	Рыскулова 298	8 727 89 89 55	Гульназ	iTeksan	Алматинская	Металл	Шкафы и архивы	Нурила

3.9-сурет – Серіктестер туралы мәліметтер терезесі

ТОО iDIA Market - База менеджеров за 22.05.2019

№	Организ-я	Адрес	Телефон	Конт. лицо	E-mail	Регион	Вид деят-ти	Осн. прод-я	Менеджер
1.	Азия Мебель	Бокейханова 35	236 66 00	Алмас	asia_mebel	Алматинская	Мебель	ДСП	Мейржан
2.	Пакс Металл	Толе би 282	8 707 910 94 14	Асем	paks_metall	Алматинская	Металл	Шкафы и архивы	Самал
3.	Штрих-М	Соловьева 3	8 495 336 36 25	Анна	kassa_soft	Московская	касса	кассовые боксы	Айнура
4.	ЭКО	Пушкина 296/8	8 495 666 1234	Александр	eco_ice	Московская	холодильное оборудование	холодильники	Нурила
5.	Арт Мебель	Брусиловского 203	8 727 294 46 46	Гюнай	art_mebel01	Алматинская	Мебель	Мебельная фурнитура	Самал
6.	Адмарт	Сатпаева 235	8 7172 3625	Гульшат	admart_kz	Акмолинская	Реклама	Оракал пленки	Самал
7.	Polus	Толстого 25/78	8 495 365 65 65	Татьяна	polus.ru	Московская	холодильное оборудование	холодильники	Айнура
8.	Solos	Москвина 87	8 495 495 55 55	Алисия	solos_89	Московская	Склад	Складские стеллажи	Мейржан
9.	Teksan	Рыскулова 298	8 727 89 89 55	Гульназ	iTeksan	Алматинская	Металл	Шкафы и архивы	Нурила

3.10-сурет – Программа нәтижесінің MS Excel-ге шақырылуы

Программалық бөлім мен аппаратураға қойылатын талаптар: қазіргі заман талабына сай дамып келетін кез-келген дербес компьютерлер, айта кетсек қолданыста азайған Pentium 3 процессорлы дербес компьютерден бастап Intel Core QUAD процессорлары ; ОЕҚ 128-мбайт жадыдан бастау алып 8-12 ГБ дейін; Windows 98- Windows 8; 16-128 разрядты видеоадаптер;

Программалық өнімнің дистрибютивті қосымшасы жасалынған. Берілген комплекс ашық түрдегі программалық өнім болып табылады, яғни программаға өзгертулерді енгізуге және пайда болған сұрақтарға байланысты адаптациялауға мүмкіндік береді [13].

Программаның дұрыс жұмыс істеуі үшін қойлатын талаптар:

– қажетті процессор;
– оперативті жады көлемі;
– сәйкес жұмыс жасайтын операциялық жүйелердің тізімі;
– қосымшалар (бағдарламалық кешендер) мысалы, Macromedia Flash, Java Runtime 2.0 немесе басқа виртаулды машина, IZET немесе SanaSoft қазақша шрифттер драйверлері, Internet Explorer 5.0 одан да жоғары, немесе сол класстағы басқа браузер, мысалы, Mozilla, FireFox, Opera, т.с.с.;
– қатты дисідегі жүйенің кэш-файлдар үшін талап ететін орын;
– түрлі-түсті монитор және т.б.;
– бағдарламалау ортасын таңдау;

Дипломдық жобаны құру барысында жалпыға ортақ Delphi 10 бағдарламалау ортасы қолданылады. Және сонымен қатар бағдарлама құру барысында қолданылған бағдарламалық кешендер тізімі мен олардың қызметін ашып көрсететін түсініктеме беріледі.

4 Экономикалық бөлім

4.1. Бағдарламаның енгізілуінің экономикалық тиімділігі

Дипломдық жобаның мақсаты «Delphi ортасында автоматтандырылған ақпараттық басқару жүйесін құру» болып табылады. Бұл бағдарлама осы жүйені қолданушылар жұмысын жеңілдетеді және мәліметтер қорын сақтау барысында керек мәліметтерді тез және қолдануға ыңғайлы етіп шығарады.

Жобаның экономикалық тиімділігін анықтау бағдарламалық құралды құру процессінің ажырамас бөлігі болып табылады. Автоматтандыру құралдарының дамуы мекемені басқару облысындағы қажеттіліктерге сай жүзеге асырылуы тиіс. Сонымен қатар, аталған қажеттіліктерді тек технологиялық жаңартпаларды енгізу қажеттілігімен байланыстырып керек жоқ. Бағдарлама әмбебап және оны орналастырудың, қолданудың еш қиындығы жоқ. Бағдарлама өңдеушісіне бизнес-бөлімшелердің нақты қажеттіліктерін түсіну керек, ал тапсырыс берушілер өңдеушілердің мүмкіндіктерін анық бағалауы тиіс. Жаңа жүйені құру жайында соңғы шешімге келу үшін мәліметтерді өңдеудің қазіргі әдістері мен енгізілуге тиіс әдістерді салыстыру қажет. Дегенмен, ақпараттық жүйені енгізудің экономикалық тиімділігі оның сипатына тәуелсіз, қойылған мақсатқа жету дәрежесімен анықталады. Автоматтандырылған жобалау жүйесі – жабдықтардан, программа мен оған керек құжаттамалардан, компьютерге енгізілетін мәліметтер жинағынан, жүйені пайдалану туралы құжаттамадан және жұмыс атқаруға қажетті жабдықтар мен жұмыс орындарымен қамтамасыз етілгеп ұжымнан тұратын үйымдастырушы-техникалық жүйе. Автоматтындырылған жобалау жүйесінің негізгі міндеті – бұйым мен оның құрама бөліктерін жобалаудың барлық немесе жеке сатыларында автоматтандырылған жобалау жұмысын жүргізу

Қазірде қоғам бұрын болып көрмеген ақпараттар ағынының көбеюі кезеңінде өмір сүруде. Бұл экономика, элеуметтік және басқару саласында айқын байқалады. Нарықтық қатынас ақпараттың уақтылы берілуіне, шынайылығына, толықтығына жоғары талаптар қояды, мұнсыз маркетингтік, қаржы – кредиттік, инвистициялық іс әрекеттер тиімді жүргізілмейді. Ақпаратқа түрлендіруші, анықтаушы қасиет тән. Информатика индустриясын құру және ақпараттық өнімнің тауарға айналуы – қоғамда терең әлеуметтік өзгерістерге алып келеді. Ақпарат материалдық өндірістен әлеуметтік салаға дейінгі қоғамның барлық салаларын қамтиды.

Нарықты өтімді талдау. Автоматтанған жұмыс орны нгізінен есептегіш техниканы пайдалану бойынша арнайы дайындығы жоқ пайдаланушыға бағытталған. Қазіргі уақытта көптеген кәсіпорындарда ауыл шаруашылығын басқарудың таратылған жүйе концепциясы жасалуда. Оларда иерархияның әртүрлі деңгейлерінде мәліметтің локальді және көп жағынан аяқталған өңдеушілігі көзделеді. Бұл жүйелерде төменнен жоғарыға мәліметті жоғары деңгейде қажет етілгендерін жіберу ұйымдастырылады. Қазіргі уақытта

осындай «гибридті» интеллектіні құру өте үлкен мәселе. Бірақ автоматтандырылған жұмыс орнын ұйымдастыру мен қызметі кезіндегі осы енгізілімнің таратылымы жақсы нәтижелерді әкеле алады. Автоматтандырылған жұмыс орны тек еңбек өнімділігін және басқару нәтижелілігінің өсу құралы ғана емес, сондай-ақ мамандардың сайлануы ыңғайлылық құралы да болады. Осы жерде адам автоматтандырылған жұмыс орны жүйесінде басқарушы буын болып қала береді.

Өндірістік кәсіпорындарда автоматтандырылған жұмыс орны мәліметті жоспарлау, басқару, өңдеу және шешімді қабылдаудың өзіндік құрылымы ретінде автоматтандыру жүйесін басқару (АЖБ) маңызды құрылымдық біріктірілуі болып табылады. Автоматтандырылған жұмыс орны – бұл әр уақытта арнайы жүйе, техникалық құрылғы мен бағдарламалық қамтамасыздандыру тобы және ол нақты шамаларға бағытталады-әкімшілік, экономист, инженер, бақылаушы, жоспарлаушы, архитектор, дизайнер, дәрігер, кітапханашы, ұйымдастырушы, зерттеуші мұражай қызметкері.

Автоматтандырылған жұмыс орнының ұйымдастырушылық қамтамасыздануның мақсаты олардың қызметін ұйымдастыру, дамыту, кадрді даярлау және әкімшіліктендіру. Соңғысына жұмысты жоспарлау, жіктеу, бақылау, талдау, реттеу, оны пайдаланушының құқықтары мен міндеттерін құжатты түрде рәсімдеу. Егер жүйе құрылғысы өте қиын болса, ол пайдаланушы да арнайы дағдысы болмаса, онда пайдаланушыны оның негізгі автоматтанған жұмыс орнына бірте-бірте енгізе алатын арнайы оқу құралдарын қолдану мүмкін. Автоматтандырылған жұмыс орны қызметін жүзеге асырғанда ағымдағы қызметінің мақсатын, мәліметтік қажеттілігін, оның тарату процесстерін бейнелейтін әртүрлі шараларын анықтау тәсілдері қажет.

Маркетинг стратегиясы. Автоматтандырылған жүйе орны жоспарлау тәсілдері оның қызмет ету тәсілдерімен байланыса алмайды, өйткені оның қызметінің дамуы пайдаланушылардың өзі мен оның даму мүмкіндігін көздейді. Автоматтандырылған жүйенңң тілдік құралдары ақырғы (соңғы) пайдаланушы көз қарасы жағынан тәсілдік құралдарын тарату болып табылады, ол бағдарламалықтар пайдаланушының тілдік құралдарын таратуды және ақырғы пайдаланушыға барлық қажет етілген әрекеттерді орындауға мүмкіндік береді. Автоматтандырылған жүйе орнының тілдік құралдары пайдаланушы әрекетімен ДЭЕМ реакциясының бір мағыналық сәйкестігі үшін қажет. Оларсыз оқу процессі, диалогты жасау, қатені табу мен дұрыстау мүмкін емес. Бұндай тілдерді жасауқиындығы олар көбіне процедуралық емес болу керектігіне негізделеді. Егер процедуралық тіл берілген әрекет орындалуын көрсетсе, процедуралық емес-бөлшектенусіз бұған қажет етілген әрекетті орындау керектігі көрсетіледі. Ақырғы пайдаланушылар мәліметтік қамтамасыздықты тарату процессін толық білмеу керек болғандықтан, АЖО интеллектуалдығының жоғарылығына қарай оның тілінде процедуралық мүмкіндіктің көлемдігі көзделген болуы керек.

3.6-сурет – Маркетинг топтамасы

Автоматтандырылған жүйе орны тілдері пайдаланушылық – бағытталған және маманды – бағытталған болуы керек. Бұл пайдаланушылардың сұрыпталу ерекшеліктеріне байланысты олар тек мамандanған жарамдылық бойынша ғана емес, сондай-ақ қызмет жағдайының иерархиясына оқылғандығына, шешуші мәліметтер түрі бойынша бөлінеді [14].

Бұндай енудің оңайлығына қарамастан, қарапайым тілді пайдалану клавиатура арқылы күрделі емес нәтижелерді алу үшін алып құрлымды енгізу қажеттілігі үшін қандай да болмасын сезілетін артықшылықты бере алмайтындығын ескеру керек.

4.2 Атқарылған жұмысты бағалау

Жүйені құруға кеткен шығынның сметасын құрамыз.

Электронды құжат айналымы жүйесін құру барысында жасалған шығындардың сметасы негізгі, қосымша және басқа шығындардан тұрады.

Кез-келген бағдарламалық өнімнің өзіндік құны келесі шығындардан құралады:

– негізгі бағдарламалық өңдеушілердің жалақысы (Зп);
– қосымша жалақы (бонус ретінде төленеді) (Здоп);
– еңбек ақы төлеу қоры (ФОТ = Зп + Здоп);
– әлеуметтік салық (Ос);
– амортизациялық төлемдер (А);

– шығын материалдары (қағаз, картридж, канцелярилық құрал-жабдықтар және т.б.) (М);

– интернет шығындары (Ри);

– арендалық төлемдер (Ра);

– сертификация мен лицензиялау шығындары (Рл);

– басқа өндірістік шығындар (барлық шығындардың 20%-ы) (Рпр).

Сонда шығындардың жалпы сомасы (1.1) формуламен анықталады:

$$С=ФОТ + Ос +А + М + Ри+ Ра +Рл+ Рпр, \quad (1.1)$$

Әрбір жұмыстың түрі бойынша бір күндік циклдың ұзақтығын (1.2) формуламен береміз:

$$t_n = \frac{T}{q_n \cdot z \cdot K}, \quad (1.2)$$

мұндағы, T – этаптың еңбек сыйымдылығы, норма-сағат;

q_n – этап бойынша атқарушылардың саны;

Z – жұмыс күнінің ұзақтығы, z=8 сағат;

K – уақыт нормасының орындалу коэффициенті, K=1,1.

Алынған t_n шамасын бүтін санға дейін жуықтаймыз.

Жүйе құру процессіне қатысушы қызметкерлердің саны мен олардың жалақы көлемі 1.1-кестеде көрсетілген.

1.1-кесте – Қызметкерлердің жалақы мөлшері

Атқарушылар	Саны, адам	Айлық жалақы мөлшері, тенге
Инженер-техник	1	100 000
Оператор	1	80 000
Барлығы	2	180 000

Жалақыны есептеу

Бір сағаттық жалақыны (1.3) формула бойынша есептейміз:

$$D = \frac{ЗП_м}{Д_р \cdot Ч_р}, \quad (1.3)$$

мұндағы, $ЗП_м$ – айлық жалақы мөлшері;

$Д_р$ – бір айдағы жұмыс күнінің саны;

$Ч_р$ – бір күннің жұмыс сағаты саны (8 сағаттық жұмыс күнінде).

Әр жұмысшының бір сағаттық жалақы мөлшері:

– инженер-техник үшін:

$$D = \frac{100000}{21 \cdot 8} = 595{,}238 \text{ теңге/сағат;}$$

– оператор үшін:

$$D = \frac{80000}{21 \cdot 8} = 476{,}190 \text{ теңге/сағат.}$$

1.2-кесте – Есептеулер нәтижесі

Жұмыс этаптарының атауы мен құрылымы		Атқарушы	Цикл ұзақтығы, күн	Еңбек сыйымдылығы, норма-сағат	Бір сағаттық жалақы мөлшері	Жалақы сомасы
Банк сферасын және оның электронды құжат айналымы жүйесін зерттеу		Оператор	2	10	476,190	4761,90
Деректер қорын құру	Негізгі кестелерді анықтау, құру және олардың арасындағы байланысты көрсету	Оператор	2	16	476,190	7619,04
	Бағдарламалау	Инженер-техник	7	50	595,238	29761,9
Пайдаланушы интерфейсін құру	Дизайнды анықтау	Оператор	2	10	476,190	4761,90
	Бағдарламалау	Инженер-техник	4	25	595,238	14880,9
Алынған нәтижелерді тестілеу		Инженер-техник, оператор	1	5	595,238	2976,19
Жүйені эксплуатациялау жөнінде нұсқаулық жазу		Оператор	2	10	476,190	4761,90
Барлығы:			20	126	3690,47	69523,7

Негізі жалақы мөлшері жүйені құру процессі кезінде қатысқан барлық қызметкерлердің жалақысының сомасы ретінде анықталады.

$$\text{Зп} = 69523{,}78 \text{ теңге.}$$

Қосымша жалақы негізгі жалақы көлемінің 10 %-н құрайды:

$$З_{\text{доп}} = З_{\text{осн}} \cdot 10\%, \qquad (1.4)$$

$$З_{\text{доп}} = 69523{,}78 \cdot 10\% = 6952{,}38 \text{ теңге.}$$

Бір айлық еңбек ақы төлеу қоры (ФОТ) негізгі және қосымша жалақылардан тұрады:

$$\text{ФОТ} = З_{\text{ОСН}} + З_{\text{ДОП}}, \quad (1.5)$$

$$\text{ФОТ} = 69523{,}78 + 6952{,}378 = 76476{,}16 \text{ теңге.}$$

Әлеуметтік салық қызметкерлер табысының көлеміне сәйкес есептеледі. Әлеуметтік салықты айлық еңбек ақы төлеу қорының 11 %-н құрайды (зейнетақы қорына түсетін 10 %-дық салықты алып тастағанда):

$$О_{\text{П}} = \text{ФОТ} \cdot 10\%, \quad (1.6)$$

$$О_{\text{П}} = 76476{,}158 \cdot 10\% = 7647{,}616 \text{ теңге.}$$

$$О_{\text{СН}} = (\text{ФОТ} - О_{\text{П}}) \cdot 11\%, \quad (1.7)$$

$$О_{\text{СН}} = (76476{,}158 - 7647{,}6158) \cdot 11\% = 7571{,}14 \text{ теңге.}$$

1.3-кесте – Шығын материалдарын есептеу

Материал атауы	Өлшем бірлігі	Саны	Бағасы, теңге	Барлығы, теңге
Басып шығаруға арналған қағаз	бір қорап	1	500	500
Картридж	бір қорап	1	2 000	2 000
Канцелярриялық жиынтық	бір дана	2	500	1 000
А4 форматты дәптер	бір дана	2	300	600
Барлығы:				4 100

Техникалық құралдарға жұмсалған шығындарды есептеу

Техникалық құралдардың амортизациялық төлемдерін есептеу:

$$А = \frac{N_{\text{АМ}} \times C_{ПЕР}}{100 \cdot 360} \cdot t_n, \quad (1.8)$$

мұндағы, $N_{\text{АМ}}$ – амортизация нормасы;

$C_{\text{ПЕР}}$ – құрылғының бастапқы құны.

Амортизациялық төлемдер:

$$\text{Компьютер} = \frac{0{,}35 \cdot 100000}{360} \cdot 20 = 1944{,}44 \text{ теңге;}$$

$$\text{Принтер} = \frac{0{,}35 \cdot 30000}{360} \cdot 20 = 583{,}33 \text{ теңге;}$$

$$\text{Сыртқы қатты диск} = \frac{0{,}40 \cdot 50000}{360} \cdot 20 = 1111{,}11 \text{ теңге.}$$

1.4-кесте – Техникалық құрылғылар мен бағдарламалық қамтамаға амортизациялық төлемдер

Құрылғы атауы	Бастапқы құны, тг	Амортизация нормасы, %	Амортизация сомасы, тг
Компьютер	100 000	35	1944,44
Принтер	30 000	35	583,33
Сыртқы қатты диск	50 000	40	1111,11
Бағдарламалық қамтама	тегін	-	-
Барлығы:			3638,88

Өндіріс процессі кезінде электрлік құрылғылар қолданылатындықтан электр энергиясына жұмсалатын шығында есептеу қажет.

Жұмыс орнын жалдауға кеткен шығынды есептеу

Алматы қаласы бойынша бір бөлмелі пәтерді жалдау бағасы айына шамамен 75 000 теңгені құрайды. Бағдарламалық жүйені құру процессі 20 күн болғандықтан, жұмыс орнын жалдауға кететін төлем (1.9) формуламен есептеледі:

$$P_a = \frac{A_м \cdot T}{30} = \frac{75000 \cdot 20}{30} = 50000 \text{ теңге,} \qquad (1.9)$$

мұндағы, Т – жалға беру мерзімі (біздің жағдайда ол t_n-ге тең).

Интернет шығынын есептеу

Модем компьютер құнына кіретіндіктен, интернетке жұмсалатын абоненттік төлемді ғана есептейміз. Абоненттік төлем таңдалған тарифке байланысты болады (мысалы, ID Net HIT):

$$P_И = 4260 \text{ теңге}$$

Жүйені енгізуге жұмсалатын шығындар

Электр энергиясына жұмсалатын шығын:

$$Э = W \cdot T \cdot S \cdot K_{ИМ} = \sum W \cdot S, \quad (1.10)$$

мұндағы, W – электр энергиясын пайдаланатын құрылғылардың тағайындалған қуаты, кВт;

S – электр энергиясының килловат-сағатының құны (13,45/кВт·сағ);

$K_{ИМ}$ – қуат пайдалану коэффициенті (0,8…0,9);

T – құрылғылардың жұмыс істеу уақыты, сағат;

$$Э = 89 \cdot 13{,}45 = 1197{,}05 \text{ тенге.}$$

Электр энергиясына жұмсалған шығындардың нәтижесі 1.5-кестеде келтірілген.

1.5-кесте – Электр энергия шығындары

Құрылғы атауы	W кВт	Жұмыс істеу күні	$K_{ИМ}$	Құрылғы жұмыс уақыты, сағ	$\sum W$, кВт×ч
Компьютер (ноутбук)	0,6	20	0,9	140	84
Принтер лазерлі	0,5	2	0,9	10	5
Барлығы	1,1	22	1,8	150	89

Эксплуатациялық шығындарды есептеу

Эксплуатациялық шығындр (1.11) формула бойынша есептеледі:

$$Э_Р = ФОТ + О_{СН} + Р_а + Э_Э, \quad (1.11)$$

$$Э_Р = 76476{,}16 + 7571{,}14 + 50000 + 1197{,}05 = 135244{,}35 \text{ теңге.}$$

Жүйені құруға кеткен қосымша шығындар барлық шығындардың 20 %-н құрайды:

$$Н_Р = Э_Р \cdot 20\%, \quad (1.12)$$

$$Н_Р = 135244{,}35 \cdot 20\% = 27048{,}87 \text{ теңге}$$

Сертификациялауға кететін шығындар:

$$Р_С = 5\% \times Э_Р, \quad (1.13)$$

$$P_C = 5\% \times 135244{,}35 = 6762{,}22 \text{ теңге.}$$

Жүйе тиімділігі мен өзіндік құнын анықтау

Бағдарламалық жүйенің өзіндік құнының құраушыларын 1.6-кестеде келтірдім.

1.6-кесте – Жүйе құнының құраушылары

Шығын түрі	Сомасы, тг	Жалпы құндық үлесі, %
Негізгі жалақы	69 523,78	38,84
Қосымша жалақы	6 952,38	3,88
Әлеуметтік салық	7 571,14	4,23
Амортизациялық төлемдер	3638,88	0,88
Шығын материалдары	4 100,00	2,29
Интернет шығыны	4 260,00	2,38
Жұмыс орнын жалдау шығыны	50 000,00	27,93
Электр энергиясы шығыны	1 197,05	0,67
Сертификациялау шығыны	6 762,22	3,78
Қосымша шығындар (20% от эксплуатационных затрат)	27 048,87	15,11
Барлығы:	181 054,32	100

Құрылған автоматтандырылған жүйенің құнын (1.14) формула бойынша есептейміз:

$$Ц = 1{,}12 \cdot (С + П), \qquad (1.14)$$

мұндағы, С – жүйенің өзіндік құны;

П – пайда.

$$Ц = С \cdot 30\% = 178998{,}77 \cdot 30\% = 53699{,}63 \text{ теңге;}$$

$$Ц = 1{,}12 \cdot (С + П) = 1{,}12 \cdot (181\,054{,}32 + 53699{,}63) = 262924{,}42 \text{ теңге.}$$

Технико - экономикалық тиімділігі

Жаңа бағдарламалық өнімді қолдану кезіндегі шығындар мен қазіргі уақыттағы шығындармен салыстырайық. Ол үшін жоғарыда есептелген еңбек шығындары, құжат айналым жүйесімен жұмыс істейтін қызметкердің орташа жалақы мөлшері (100 000 теңге көлемінде), бір айдағы жұмыс сағаты саны (176 сағат), компьютер құны, оның қызмет ету уақыты және амортизация төлемдері қажет.

Бұл аталған көрсеткіштер құрылған бағдарламалық қамтаманы және ақпараттық жүйені қолдана отырып атқарылатын операцияларға кететін шығындарды есептеуге жеткілікті:

$$Ei = Ti \cdot (W / D + \frac{S \cdot (1+R)}{Pt}), \qquad (1.15)$$

мұндағы, Ti – базалық және жобалық нұсқалар бойынша еңбек шығындары;

W – құжат айналым жүйесімен жұмыс істейтін қызметкердің орташа жалақы мөлшері;

D – жұмыс күні саны;

S – ДК құны;

R – компьютер үшін амортизациялық төлемдер;

Pt – ДК қызмет ету мерзімі.

1.7-кесте – Негізгі бизнес-процестерді орындау уақыты

Процесс	Базалық нұсқа бойынша орындау уақыты, мин.	Жобалық нұсқа бойынша орындау уақыты, мин.
Дайын құжатты енгізу	15	10
Хабарламалармен алмасу	5	2
Құжатқа қол қою	7	3
Құжатты растау	10	3
Құжатты іздеу	15	5
Жаңа пайдаланушыларды енгізу	15	10
Барлығы	67	33

Қазіргі қолданылатын жүйеде операцияларды орындауға кететін шығындар:

$$Eo = \frac{67}{60} \cdot (100000/176 + \frac{100000 \cdot (1+15)}{17500}) \approx 737 \quad теңге$$

Құрылған жүйеде операцияларды орындауға кететін шығындар:

$$Ep = \frac{33}{60} \cdot (100000/176 + \frac{100000 \cdot (1+15)}{17500}) \approx 363 \quad теңге$$

Экономикалық тиімділікті E_p және E_o айырмасы ретінде есептейміз. Бұл айырма 737 – 363 = 374 теңгеге тең. Бір пайдаланушы күніне осындай операциялардың 40 жуығы орындай алады. Сонымен, берілген жүйені енгізу ең кемінде $\frac{260622,21}{374 \cdot 40} = 18$ күннен кейін пайда әкеле бастайды.

Сәйкесінше жылдық (240 жұмыс күні) экономикалық тиімділік:

$$Y = 374 \cdot 40 \cdot 240 - 262924,42 = 3590400 - 262924,42 = 3327475,58 \text{ теңге.}$$

Содан шығатын экономикалық тиімділік көрсеткіші:

$$\frac{3327475,58}{262924,42} = 12,66\ \%$$

Алынған нәтижелер бойынша көріп тұрғанымыздай, экономикалық тиімділік өте жоғары, бұл мекеменің құжат айналым жүйесіне кететін шығынды азайтуға жағдай жасайды. Бұл құрылған өнімнің бағасын арттыруға бірден бір себеп.

Бағдарламаны қызметкерлер саны 100-ден асатын компанияларда қолдануға ұсынылады. Нарық бағалары бойынша ұқсас бағдарламалық құралдар орташа бағамен 400 000 теңгеге бағаланады.

5 Өміртіршілік қауіпсіздігі

5.1 Дербес компьютер мен бағдарламаны қолдану барысындағы қауіпсіздік шаралары

Дербес компьютер (ДК) заманауи қоғамның өміріне пайдасы тиетін шексіз функцияларға толы. Бұл есептеуіш машиналар күрделі техника болып табылады және кез келген басқа құрылғылар тәрізді белгілі бір электр және техника қауіпсіздік ережелерін сақтауды талап етеді. Оларды білмеу немесе сақтамау біршама келеңсіз жағдайларға душар етуі мүмкін[15].

Дегенмен, электронды-есептеуіш техниканың негативті факторлары әлі де жойылған жоқ. Дербес компьютерден және кеңселік перифериялық қондырғылардан келетін негативті факторларға келесілер жатады:

- электромагниттік өрістің жоғары деңгейі;
- акустикалық шу;
- ауадағы қауіпті бөлшектердің концентрациясы;
- жұмсақ рентгендік сәуле шығару;
- видеодисплейлердің қауіпті визуалды көрсеткіштері.

Дербес компьютерлермен жұмыс істейтін адамдарға жұмыс күнінің аяғында бас ауруы, көз ашуы, мойын, қол, жауырын ауыруы, бет терісінің ашуы және т.с.с. тән. Оларды күнде сезіну мигренге, көз ауруына, сколизға және де басқа да ауруларға әкеліп соғуы мүмкін. Адамдар жетекші технологиялардың зиянды әрекетінен ешқашан құтылмауы да мүмкін, бірақ, басқа да жағдайлар сияқты дербес компьютер қолданушылары олардың әсерән минимумға түсіруі мүмкін. Көп проблемалар жұмыс орнының дұрыс ұйыдастырылуы, техника қауіпсіздігі ережелерін сақтау және жұмыс уақытын дұрыс бөлу болса, өзінен - өзі шешіледі.

Өз жұмысында дербес компьютер негізіндегі автоматтандырылған ақпараттық жүйелерді қолданатын адамдардың денсаулығына қатысты негізгі проблема көзі монитор болып табылады, әсіресе электронды- сәулелік мониторлар. Мониторлар оператор денсаулығына зиянысдсәулелер шығарады.

Зерттеулер нәтижесі әзірше, тек статистикалық характерде және адекватты түсінік бермейді. Монитор шығаратын жиілік спектрі рентген, ультра-күлгін, ифра-қызыл сәулелермен және басқа да электро-магнитті тербелістермен мінезделеді.

Рентген және басқа да сәулелерден туатын қауіпті көптеген ғалымдар өте кішкене дейді, өйткені олардың деңгейі өте төмен және олар негізінен экран қабықшасымен жұтылады. Үлкенірек қауіп өте төмен жиілік шығаратын өрістермен байланысты. Олар тірі организмге әсер еткенде биологиялық эффектілер тудыратыны белгілі болды. 60 Гц- ті жиілікті электро-магнитті өріс жануарлар клеткасында өзгерістер (ДНК синтезін бұзуға дейін) тударытыны белгілі болды. Зерттеушілер үшін таңқаларлық факт болып, электромагниттік толқындардың рентгендік сәулелерден айырмашылығы электро магниттік толқындар ерекше қасиеті бар, олар шығару интенсивтілігін кеміткенмен

әсерінің қауіптілігі кемімейді, тіпті кейбір өрістер дене клеткаларына тек кіші интенисивтілікте немесе нақты бір жиілікте әсер етеді. Американдық ғалымдардың түсіндіруі бойынша айнымалы 60 Гц жилікте тербелетін электромагнитті өріс өзімен бірге кезкелген дене немесе ми молекулаларын тербеліске түсіреді.

Айтылғандардың негізінде дербес компьютер қолданушылары үшін олардың еңбегін қорғау тұрғысынан белгілі бір кеңестер қорытып шығаруға болады. Бұл проблеманы шешу негізінен аппараттық және программалық жабдықтарын қатаң бақылау, сонымен қатар, оларды қолданудың эргономикалық талаптарын қатаң сақтау арқылы шешуге болады.

Олардың кейбірін келтіріп кетейік:

– медициналық көрсеткіштер бойынша шектеулерді сақтау;

– дисплей характеристикаларына аса көңіл бөлу;

– оператордың жұмыс орнын дұрыс ұйымдастыру;

– оператордың жұмыс уақытын дұрыс ұйымдастыру;

Осылардың әрқайсысын жеке қарастырып, кеңестер қорытып шығаруға болады:

– дербес компьютерлермен жұмысты қозғалыс аппараттарының ауруымен ауыратындар, көз, тері, сонымен қатар, екіқабат әйелдер үшін шектеу қажет.

– Жоғары өлшемді және көлемі 15”-тен (Hi-Resolution, Non-Interlaced, Low-Radiation) кем емес дисплей қолданған жөн;

– 70-72 Гц –ті және жоғары өлшемді видеоадаптер болғаны жөн;

– дисплейден 70 см қашықтыұта отыру;

– дисплей экраны жарық көздерінен шағылыспауы керек;

– дисплейді тікелей жарық көздерінің астына немесе өте жақын орналаспауы тиіс;

– оператордың жұмыс орны бөлменің қалыпты жарықтығынан 2/3 нормасынан аспағаны дұрыс;

– дисплей экранының артындағы жар оның экранындай жарықтануы тиіс;

– бір бөлмеде бірнеше компьютер орналасқан жағдайда әрбір оператордың жұмыс орны көршісінен қашықтығы 1,2 м- ден кем болмауы тиіс;

– жұмыс орны қолайсыз болмауы керек;

– дисплеймен жалпы жұмыс көлемі барлықжұмыс уақытының 50%-нан аспауы тиіс;

– сағатына 10 мың (1500 сөздей) клаваишті басу жылдамдығынан аспаған жөн;

– компьютермен жұмыс кезінде әрбір 2 сағат саын демалған жөн;

Қолданушы қауіпсіздігімен қатар, компьютер қауіпсіздігін де қамтамасыз ету қажет, әсіресе онда сақталған ақпараттарды. Ерекше маңызды ақпараттар сақталған компьютерді электр желісіндегі авариялық жағдайлардан

белгілі бір уақыт қоректендіретін тоқтаусыз қоректендіру құрылғысымен жабдықтау қажет [16].

Жүйелік блоктың артқы жағын қабырғаға тақап қоюға болмайды, бұл жүйелік блоктың салқындауына қиындық туғызады да, оның қызып кетуіне алып келеді. Дисплейге де соны айтуға болады, оның үстіне қағаз, кітап, жалпы ешнәрсе қоюға болмайды. Электроника шаңға төзе алмайды, сондықтан шаң минимальды болуы тиіс.

5.2 Кондиционерлеу және вентиляция жүйесін есептеу

2.1-Кесте – Алғашқы мәндер

Қала		Алматы
Бөлме параметрлері, м	Ұзындығы	12
	Ені	6
	Биіктігі	4
Қондырғы бойынша мәліметтер	саны, дана	10
	жұмыс қуаты, кВт/ч	1,5
	ПӘК, η	0,8
Жасанды жарық бойынша мәліметтер	жарық құрылғ-ң қуаты, Вт/м^2	60
Қызметкерлер саны, олардың ішінде	Ерлер	4
	Әйелдер	6
Терезелер	Саны	8
	1 терезенің ауданы, м^2	2
	Орналасуы	ОШ
	Түрі	жалюзи, ағаш түптеу, ластануы елеусіз
Тәуліктің есептік уақыты, сағ.		11-12
Бөлмедегі температура, ^{0}С	Жазда	25
	Қыста	18
Жұмыс істеу қалпы		Отырып

Бөлмедегі жылулық жүктелімдерді есептеу

Әр түрлі мақсатта пайдаланылатын бөлмелерде негізінен жылулық жүктемелер әрекет етеді, олар бөлме сыртында да, ішінде де пайда болады[18].

Сыртқы жылулық жүктемелер

Берілген жүктемелер келесі құрауыштардан тұрады:

– бөлме сыртындағы және ішіндегі температуралардың айырмашылығынан туатын қабырғалар, төбе, терезелер мен есік арқылы жылудың енуі немесе жылудың жоғалуы;

– әнектелген аудандар арқылы күннің сәуле шығаруынан пайда болатын жылу енуі;

– инфильтрация салдарынан жылу енуі.

Жыл мезгілі мен тәулік мезгіліне сәйкес сыртқы жылулық жүктемелер оң болуы мүмкін.

Температуралардың айырмашылығы әсерінен болатын жылу енулері мен жылу жоғалтулары (1.20) формула бойынша анықталады:

$$Q_{огр} = V_{пом} \cdot X_0 \cdot (t_{Врасч} - t_{Нрасч}), \quad \text{Вт} \qquad (1.20)$$

мұндағы, $V_{пом}$ – бөлме ауданы, м^3:

$$V_{\text{пом}} = 12 \cdot 6 \cdot 4 = 288 \text{ м}^3$$

X_0 – меншікті жылулық сипаттама, Вт/м^3 ^{0}С:

$$X_0 = 0{,}42 \quad Вт / м^3 \, ^0C;$$

$t_{Нрасч}$ – сыртқы температура (Б параметрі), суық период үшін – 13 сағаттағы ең суық айдың орташа температурасы. Мәліметті 1.2-кестеден аламыз: жылы период үшін $t_{Нрасч}$ = 26 ^{0}С және суық период үшін $t_{Нрасч}$ = -18^{0}С;

$t_{Врасч}$ – ішкі температура, жайлы жағдайларды немесе өндірістік процестерге қойылатын технологиялық талаптарға сәйкес алынады. Жылы период үшін $t_{Врасч}$ = 25 ^{0}С және суық период үшін $t_{Врасч}$ = 18 ^{0}С.

$$Q_{\text{огр}} = 288 \cdot 0{,}42 \cdot 1 = 120{,}96 \text{ Вт}$$

$$Q_{\text{огр}} = 288 \cdot 0{,}42 \cdot 36 = 4354{,}6 \text{ Вт}$$

Жылы период кезіндегі жылу енуі 120,96 Вт болса, суық период кезіндегі жылу жоғалуы 4354,6 Вт.

Күн сәулесінен болатын артық жылу әйнек типіне байланысты 90 %-ға дейін бөлме ортасымен жұтылады, ал қалған бөлігі шағылады. Максималды жылулық жүктеме сәулеленудің максималды кезінде болады. Сәулелену интенсивтілігі өңірдің еніне, жыл мерзіміне және тәулік уақытына байланысты болады.

Күн сәулесінен жылудың әйнек арқылы енуі (1.21) формуласымен анықталады:

$$Q_p = q^1 \cdot F_0 \cdot \beta_{с.з} = (q_{вп} + q_{вр}) \cdot K_1^C \cdot K_2 \cdot \beta_{с.з} \cdot n \cdot S_0, \; Вт \, , \qquad (1.21)$$

мұндағы, $q_{вп}$, $q_{вр}$ – тура және шашыраңқы радиацияның жылулық ағындары, Вт/м^2;

$F_{0Ю} = n \cdot S_0 = 8 \cdot 2 = 16 \;\; м^2$ – терезе ауданы, О (оңтүстік) бағыт үшін (n – терезелер саны; S_0 – 1 терезенің ауданы);

$\beta_{с.з}$ – жылу ену коэффициенті, Жалюзи үшін $\beta_{с.з}$ = 0,20;

$К_1$ – түптеулермен қараңғылану коэффициенті ($К^С_1$ – сәулеленген терезелер үшін). $К^С_1 = 0,72$;

$К_2$ – әйнек ластануының коэффициенті, $К_2 = 0,9$. 44^0 ендік үшін О түс уақытынан кейін (11-12 сағ.):

$$q_{вп} = 288 \ \ Вт/м^2; \ \ q_{вр} = 85 \ \ Вт/м^2$$

Сонда:

$$Q_P = (288 + 85) \cdot 0,72 \cdot 0,9 \cdot 0,2 \cdot 16 = 773,5 \text{ Вт}$$

44^0 ендік үшін ОШ түс уақытынан кейін (11-12 сағ.):

$$q_{вп} = 214 \ \ Вт/м^2; \ \ q_{вр} = 79 \ \ Вт/м^2$$

Сонда:

$$Q_P = (214 + 79) \cdot 0,72 \cdot 0,9 \cdot 0,20 \cdot 16 = 607,5 \text{ Вт}$$

Күн сәулесінен түсетін жалпы жылу:

$$Q_P = 773,5 + 607,5 = 1381 \text{ Вт}$$

Ішкі жылулық жүктемелер

Ішкі жүктемелердің негізгі құраушылары:

– адамдардан бөлінетін жылу;

– шамдар, жарықтандыру және электр тұрмыстық қондырғылардан бөлінетін жылулар;

– компьютерлер, басып шығарушы қондырғылардан және т.б. бөлінетін жылулар.

Өндірістік және технологиялық бөлмелерде жылу бөлінудің қосымша көздері болып келесілер саналады: қыздырылған өндірістік қондырғы, ыстық материалдар, соның ішінде сұйықтықтар және әр түрлі жартылай фабрикаттар, жану және химиялық реакция заттары[19].

Адамдардан жылу бөліну атқарылатын жұмыстың интенсивтілігіне және қоршаған ауаның параметрлеріне байланысты болады.

Есептеулер үшін бөлмедегі адамдар санын бір адам үшін көрсеткішке көбейту қажет. Осылай біз бөлмедегі барлық адамдардан бөлінетін жылуды таба аламыз [20].

Жаз периоды үшін: 1.3-кесте бойынша 25 ^{0}С температурада бір ер адам 51 Вт айқын жылу бөледі, ал жалпы – 102 Вт. Әйел адам ер адамның жылу бөлу нормасының 85%-н құрайтын жылу бөледі.

Сонда бөлмедегі айқын жылу бөлу:

$$Q_{л}^{я} = 51 \cdot 4 + 51 \cdot 6 \cdot 0{,}85 = 464 \text{ Вт}$$

Жалпы жылу бөлу:

$$Q_{л}^{о} = 102 \cdot 4 + 102 \cdot 6 \cdot 0{,}85 = 928 \text{ Вт}$$

Жаз периоды үшін: 1.3-кесте бойынша 18 ^{0}С температурада бір ер адам 82 Вт айқын жылу бөледі, ал жалпы – 103 Вт. Әйел адам ер адамның жылу бөлу нормасының 85%-н құрайтын жылу бөледі.

Сонда бөлмедегі айқын жылу бөлу:

$$Q_{л}^{я} = 82 \cdot 4 + 82 \cdot 6 \cdot 0{,}85 = 746 \text{ Вт}$$

Жалпы жылу бөлу:

$$Q_{л}^{о} = 103 \cdot 4 + 103 \cdot 6 \cdot 0{,}85 = 937 \text{ Вт}$$

Жарықтандыру құралдарынан, оргтехника мен қондырғылардан бөлінетін жылу (1.22) формуласымен есептеледі. Шамдардан бөлінетін жылу:

$$Q_{осв} = \eta \cdot N_{осв}, \ \ Вт, \tag{1.22}$$

мұндағы, η – электр энергиясының жылу энергиясына айналу коэффициенті (қыздыру шамдар үшін η = 0,92-0,97);

$N_{осв}$ – шамдардың орнатылған қуаты (N = 60 Вт/м^2).

Жарықтандыру құралдарынн бөлінетін жылуды есептеу үшін еден ауданын ескеру қажет.

Сонда:

$$Q_{осв} = \eta \cdot N_{осв} \cdot F_{пол}, Вт, \tag{1.23}$$

$F_{пол}$ – бөлме еденінің ауданы:

$$F_{пол} = 12 \cdot 6 = 72 \ \ м^2$$

$$Q_{осв} = 0{,}8 \cdot 60 \cdot 72 = 3456 \ \ Вт$$

Өндірістік қондырғылармен бөлінетін жылу:

$$Q_{об} = N_{уст} \cdot K \tag{1.24}$$

$$Q_{\text{об}} = 1{,}5 \cdot 10 \cdot 0{,}8 = 12 \text{ кВт}$$

Оргтехника арқылы бөлінетін жалпы жылуды анықтау үшін қондырғы қуатының 20 %-ына көбейту керек.

Сонда:

$$Q_{\text{орг}} = 1{,}5 \cdot 10 \cdot 0{,}2 = 3 \text{ кВт}$$

Бөлменің жылулық балансын есептеу

Жүргізілген есептеулердің негізінде бөлмедегі жылу балансын құрамыз.

Жылдың суық периоды үшін:

$$Q^{З}_{изб} = Q_{Р} + Q^{Я}_{З} + Q_{осв} + Q_{об} + Q_{орг\_х}$$

$$Q_{\text{изб}} = 4354{,}6 + 773{,}5 + 937 + 3456 + 12000 + 3000 = 24\,521{,}1 \text{ Дж}$$

Жылдың жылы периоды үшін:

$$Q^{Л}_{изб} = Q_{Р} + Q^{Я}_{Л} + Q_{осв} + Q_{об} + Q_{орг\_т}$$

$$Q_{\text{изб}} = 120{,}96 + 773{,}5 + 928 + 3456 + 12000 + 3000 = 20\,278{,}46 \text{ Дж}$$

$Q^{Л}_{изб} > Q^{З}_{изб}$ болғандықтан, кейінгі есептеулер үшін:

$$Q_{изб} = Q^{Л}_{изб} = 9412{,}5676 \quad Вт.$$

Ауаның жылу кернеулігін есептейміз:

$$Q_{\text{н}} = \frac{Q_{\text{изб}} \cdot 860}{V_{\text{пом}}}$$

$$Q_{\text{н}} = \frac{20\,278{,}46 \cdot 860}{288} = 60{,}55 \approx 61 \text{ ккал/м}^3$$

$$Q_{Н} \succ 20 \quad ккал/м^3, \ \Delta t = 8^0 C,$$

$$Q_{Н} \prec 20 \quad ккал/м^3, \ \Delta t = 6^0 C$$

Бөлмеге қажетті ауа мөлшерін анықтау:

$$L = \frac{Q_{изб} \cdot 860}{C \cdot \Delta t \cdot \gamma},\ м^3/сағ, \quad (1.25)$$

мұндағы, С = 0,24 ккал/(кг*^{0}С) – ауаның жылу сыйымдылығы;
γ = 1,206 кг/м^3 – құйылатын ауаның меншікті массасы;
Δt = 6 ^{0}С.

Сонда:

$$L = \frac{20\ 278{,}46 \cdot 860}{0{,}24 \cdot 10^4 \cdot 8 \cdot 1{,}206} = 753{,}16\ м^3/час$$

Ауа алмасудың еселігін анықтау:

$$n = \frac{L}{V_{пом}}$$

$$n = \frac{753{,}16}{288} = 2{,}61\ час^{-1}$$

Кондиционерді таңдау

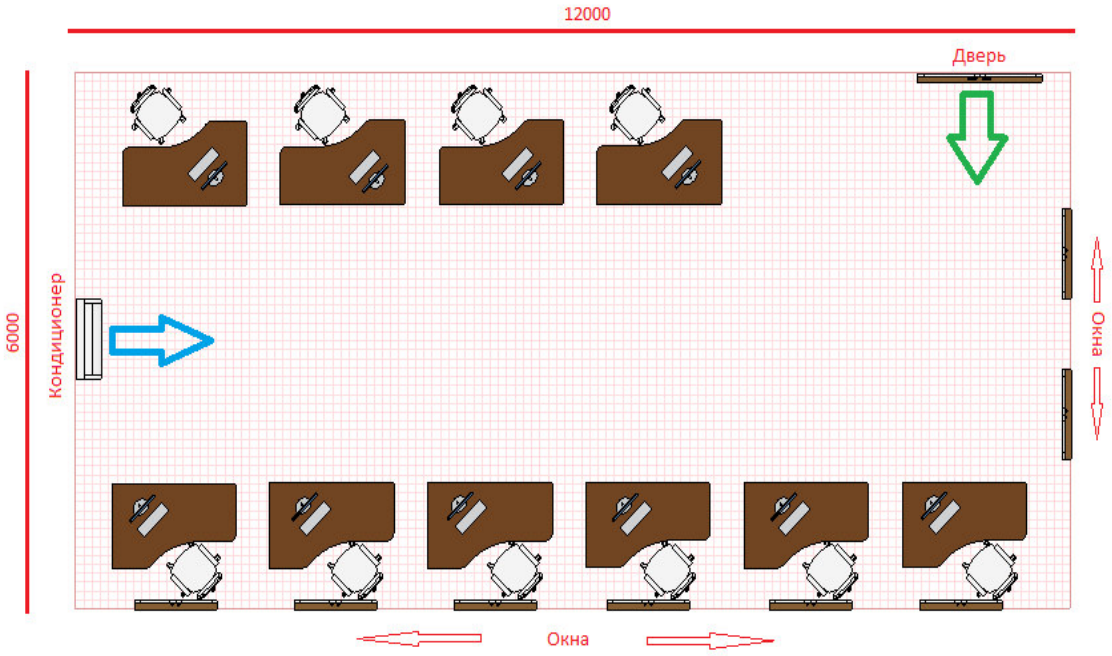
Алынған мәліметтерге жүгіне отырып жоғарыдан ауа жіберетін SUA 0331 кондиционерін таңдаймыз. Оның техникалық сипаттамалары 1.4-кестеде берілген.

2.2-кесте – UNIFLAIR фирмасының SUA 0501 модельді кондиционерінің негізгі техникалық сипаттамалары

Жоғарыдан жіберу моделі		SUA 0501
Ауа шығыны, м^3/сағ	макс.	4720
	мин.	2950
Суық өндірушілігі, Вт	24 ^{0}С, 50%	16,7
	26 ^{0}С, 50%	17,7
Компрессор қуаты, кВт		4,5
Электр қыздырғыштығы, кВт		6,6
Ылғалдатқыштығы, кВт		2,3
Бу шығыны, кг/ч		3,0
Өлшемдері, мм	биіктігі	1740
	ені	1200
	тереңдігі	450
Массасы, кг		260

Ауа салқындатқышы бар кондиционер екі блоктан тұрады: ішкі блок компрессордан, буландырғыштан, вентилятор мен автоматикадан тұрады; сыртқы блок алып шығарылатын конденсатордан және жылу алмастырғыштан

тұрады. Ауа төмен бірден бөлмеге жіберіледі, беттік панель арқылы ішке беріледі.



4.1-сурет – Офиста кондиционердің орналасу сызбасы

Қорытынды

Delphi ортасында автоматтандырылған ақпараттық басқару жүйесін құру тақырыбындағы дипломдық жобамды кәсіпорын, фирма, кез-келген ЖАТ (жаңа ақпараттық технологиясы) қолданысқа алу мүмкіндігі бар ұйымдардың қолданушы кезіндегі тап болатын қиындықтарын шешуге, іс-қағаздармен жұмыс барысында және клиентпен тікелей ақпарат алмасу үшін жұмсалатын жұмыс көлемін барынша азайту проблемаларын шешуге арнадым. Клиент-сервер жұмысын автоматтандыру барысында арадағы пайда болатын қателіктер қаупінің мүмкін болатын ең төменгі көрсеткішіне қол жеткізе алдым.

Дипломдық жобаны орындауға Delphi бағдарламалау ортасы таңдап алынды. Себебі, бұл бағдарламалау ортасының мүмкіндіктері өте көп және пайдаланушы интерфесі өте ыңғайлы болып табылады. Бұл жұмысты зерттей келе өзіме Delphi ортасының көптеген мүмкіндіктерін аштым, қолданыста пайдалана алатын қосымшалар жұмысымен таныстым. Класстар негізін, инкаплусяция, полиморфизм деңгейімен жұмыс барысында класстар,прецеденттер диаграммасы қолданылды. Серверлік бағдарлама ретінде реляциондық мәліметтер қорын қолданысқа алатын SQL Server бағдарламасын қолдандым.

Жұмыстың нәтижесі ретінде ТОО «IDIA Market» орталығында қолданысқа енген Delphi ортасында құралған, автоматтандырылған ақпараттық басқару жүйесін көруімізге болады. Тапсырма, талаптарға сәйкес бағдарлама құрылды және жалпы жүйеде қолданысқа енгізілді.

Әдебиеттер тізімі

1 Климова, Л. М. Delphi 7. Основы программирования. Решение типовых задач. Самоучитель / Л.М. Климова. - М.: КУДИЦ-Образ, 2017. - 480 с.

2 Культин, Никита Основы программирования в Delphi 8 для Microsoft.NET Framework. Самоучитель (+ CD-ROM) / Никита Культин. - М.: БХВ-Петербург, 2013. - 400 с.

3 Культин, Никита Основы программирования в Delphi XE (+ CD-ROM) / Никита Культин. - М.: БХВ-Петербург, 2011. - 416 с.

4 Осипов, Дмитрий Delphi. Программирование для Windows, OS X, iOS и Android / Дмитрий Осипов. - М.: "БХВ-Петербург", 2014. - 464 с.

5 Ревич, Ю. Нестандартные приемы программирования на Delphi / Ю. Ревич. - М.: БХВ-Петербург, 2016. - 560 с.

6 Санников, Е. В. Курс практического программирования в Delphi. Объектно-ориентированное программирование / Е.В. Санников. - М.: Солон-Пресс, 2013. - 188 с.

7 Фленов, М.Е. DirectX и Delphi. Искусство программирования (+ CD-ROM) / М.Е. Фленов. - М.: БХВ-Петербург, 2010. - 482 с.

8 Архангельский А. Я. Программирование в Delphi для Windows. Версии 2006, 2007, Turbo Delphi (+ CD-ROM); Бином-Пресс - Москва, 2010. - 716 с.

9 Бобровский, Сергей Delphi 7. Учебный курс; СПб: Питер - М., 2018. - 736 с.

10 Григорьев, А.Б. О чем не пишут в книгах по Delphi; БХВ-Петербург - М., 2016. - 576 с.

11 Дарахвелидзе, П.Г.; Марков, Е.П. Delphi 2005 для Win32 наиболее полное руководство; БХВ-Петербург - М., 2018. - 234 с.

12 Калверт, Ч. Базы данных в Delphi 4; Киев: ДиаСофт - М., 2013. - 464 с.

13 Марков, Е.П.; Никифоров, В.В. Delphi 2005 для .NET; БХВ-Петербург - М., 2017. - 896 с.

14 Понамарев, В. Базы данных в Delphi 7. Самоучитель; СПб: Питер - М., 2015. - 224 с.

15 Сван, Том Секреты 32-разрядного программирования в Delphi (+ дискета); Диалектика - М., 2015. - 480 с.

16 Сухарев, М.В. Основы Delphi. Профессиональный подход; Наука и техника - М., 2018. - 600 с.

17 Федоров, А. Delphi 2.0 для всех; Компьютер-пресс - М., 2013. - 464 с.

18 Шумаков, П.В. Delphi 3 и разработка приложений баз данных; Нолидж - М., 2018. - 704 с.

19 О.Камардинов, Х.Жантелі Delphi 5-6. Шымкент, 2002 ж.

20 Белов, В.В. Программирование в Delphi: процедурное, объектно-ориентированное, визуальное: Учебное пособие для вузов / В.В. Белов, В.И. Чистякова. - М.: ГЛТ , 2014. - 240 с.

А қосымшасы
(міндетті)

Техникалық тапсырма

А.1 Техникалық тапсырманың сипаттамасы

Бұл бөлімде Delphi ортасында орындалған қолданушыға ыңғайлы интерфейсі бар және іске асыру ерекшеліктері қарастырылады.

А.1.1 Жобаның мақсаты мен бағыты

Delphi ортасында автоматтандырылған ақпараттық басқару жүйесін құру бұл кез-келген басқару формасының программаның техникалық сипаттамалары актуализациясына көмектеседі. Осыған орай бөлім меңгерулерінің жұмысы жүйесі автоматтандыруды өңдеу принциптерін құру және эффективті функциялау мемлекет пен қоғамның қатысына тәуелді маңызды мақсаттың бірі. Өзіміздің елде кез-келген жұмысты автоматтандыруды жобалау және эксплутациялау тәжірбиесі арта түсуде. Ғылыми техникалық прогресінің негізгі бағыты ретінде де өндірістер, кәсіпорындар, тағы басқа әртүрлі ауданда технологиялық процестерді автоматтандыру жұмыстарын комплексті жүргізу болып табылады.

Дипломдық жұмысымның мақсаты – жоғарыда айтылған мәліметтерді енгізетін, өңдейтін, сұрыптайтын, баспаға жібере алатын, берілген мәліметтер негізінде анализ жасай алатын, технологиялық процесстерді кәсіпорын жұмысын автоматтандыру мақсатында қазіргі заманғы технологиялық құралдарды дамыту және жұмыс барысын автоматтандыруға үйренуге негізделген , қолданушыға ыңғайлы интерфейсі бар бағдарлама құру.

Мәліметтер қорын өңдеу жұмыстары серверде атқарылатын, ал клиент қарапайым браузер болатын программалар жасалуда. Delphi-дің мүмкіндігі көп болуына байланысты жоғарыда айтылған мақсатты жүзеге асыру үшін осы ортаны пайдалануға болады.

Дипломдық жұмыстың өзектілігі:

– есептеуде тап болатын қиындықтарды шешуге, іс-қағаздармен жұмыс барысында ақпарат алмасу үшін жұмсалатын жұмыс көлемін барынша азайту;

– есептеу жұмысын автоматтандыру барысында арадағы пайда болатын қателіктер қаупінің мүмкін болатын ең төменгі көрсеткішіне қол жеткізу;

А.1.2 Программаның техникалық сипаттамасы

Компьютерлерге қойылатын жалпы талаптар:

– процессор – жоғары;

– оперативті жады – 256 МВ жоғары;

– диск тұлғалы кеңістік – ~ 52 MB + қолданушылар файлдарын сақтауға

А қосымшасының жалғасы

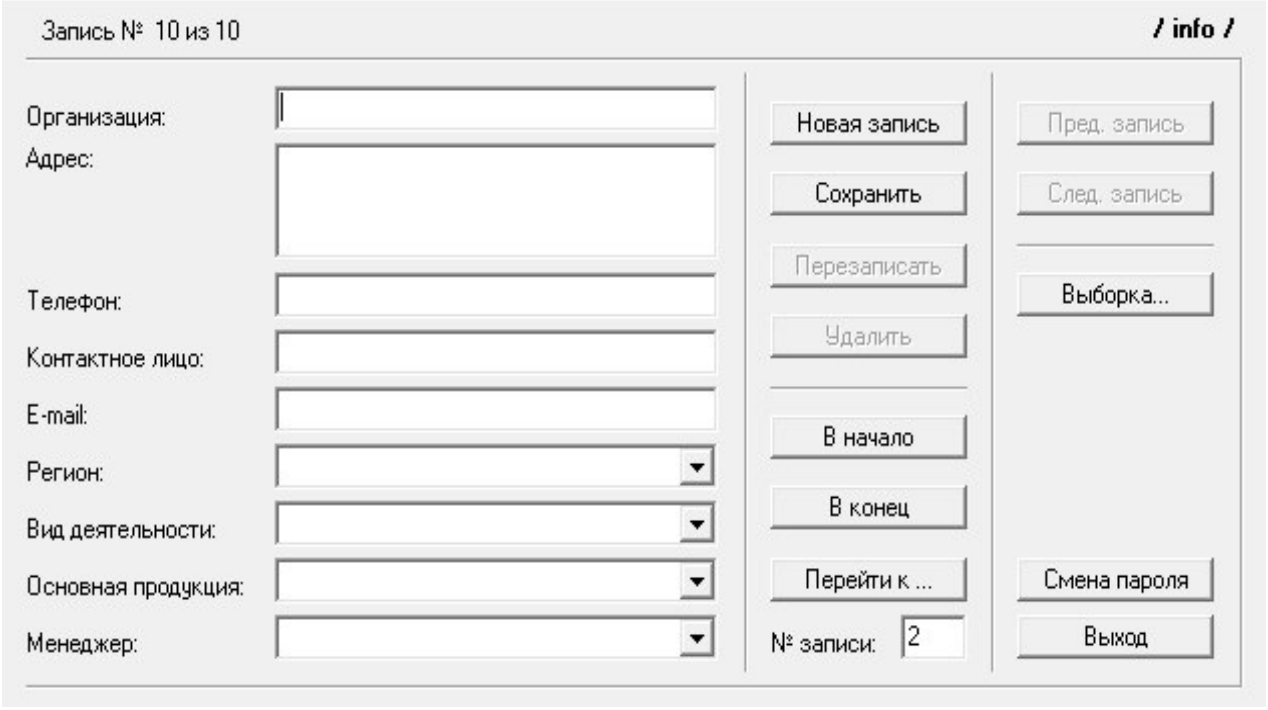
– қажет орын;
– Internet желісіне қатынау;
– жұмысты қолдайтын видеокарта;
– пернетақта, манипулятор тышқан.

А.1.3 Пайдаланылатын техникалық құралдар

Серверлік программалы компоненттер:
– Windows 2010 Server операциялық жүйесі;
– ДҚБЖ SQL Server Management Studio 12.0;
– Embarcadero Rad Studio Delphi 10 XE;
– Клиенттік программалы компоненттер:
– Windows 2010/XP/Vista операциялық жүйесі;
– Internet Explorer, Google Chrome, Opera.

А.1.4 Кіріс деректер

Бағдарлама бірнеше терезеден тұрады, олардың әр қайсысының өз қызметтері бар. Windows-тың кез-келген операциялық жүйесімен ашылатын .exe файл арқылы іске қосамыз. Ең бірінші бағдарламаны ашқандағы қосылған терезе басты терезе болып табылады, ол терезеде бағдарламаға тіркелген мәліметтер қоры ашылған.



А.1.1-сурет – Бағдарламаның басты терезесі
А қосымшасының жалғасы

А.1.5 Шығыс деректер

Тұрақты сұрауды орындауға мүмкіндік беретін бағдарлама жасадым. Мұны істеу үшін келесі әрекеттерді орындадым:

– жаңа каталогқа жаңа жоба ашу;
– біз статикалық сұрауды ұйымдастырамыз.

ТОО iDIA Market - База менеджеров за 22.05.2019

№	Организ-я	Адрес	Телефон	Конт. лицо	E-mail	Регион	Вид деят-ти	Осн. прод-я	Менеджер
1.	Азия Мебель	Бокейханова 35	236 66 00	Алмас	asia_mebel	Алматинская	Мебель	ДСП	Мейржан
2.	Пакс Металл	Толе би 282	8 707 910 94 14	Асем	paks_metall	Алматинская	Металл	Шкафы и архивы	Самал
3.	Штрих-М	Соловьева 3	8 495 336 36 25	Анна	kassa_soft	Московская	касса	кассовые боксы	Айнура
4.	ЭКО	Пушкина 296/8	8 495 666 1234	Александр	eco_ice	Московская	холодильное оборудование	холодильники	Нурила
5.	Арт Мебель	Брусиловского 203	8 727 294 46 46	Гюнай	art_mebel01	Алматинская	Мебель	Мебельная фурнитура	Самал
6.	Адмарт	Сатпаева 235	8 7172 3625	Гульшат	admart_kz	Акмолинская	Реклама	Оракал пленки	Самал
7.	Polus	Толстого 25/78	8 495 365 65 65	Татьяна	polus.ru	Московская	холодильное оборудование	холодильники	Айнура
8.	Solos	Москвина 87	8 495 495 55 55	Алисия	solos_89	Московская	Склад	Складские стеллажи	Мейржан
9.	Teksan	Рыскулова 298	8 727 89 89 55	Гульназ	iTeksan	Алматинская	Металл	Шкафы и архивы	Нурила

А.1.2-сурет – Программа нәтижесінің MS Excel-ге шақырылуы

А.2 Программалық қамтаманы жобалау кезеңі

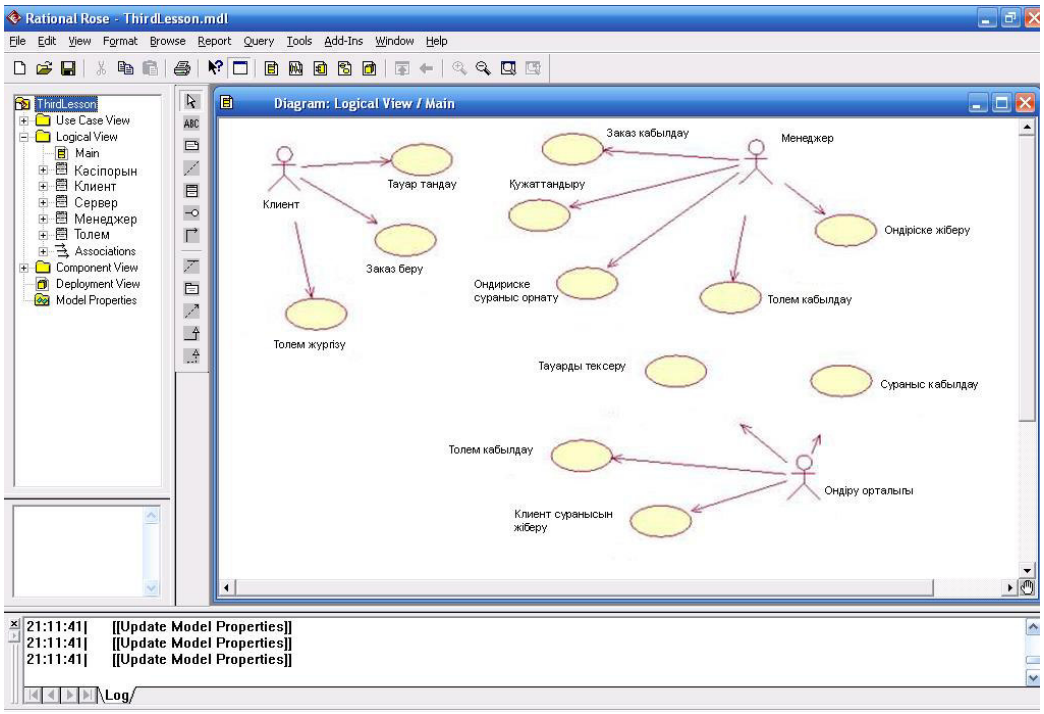
Прецеденттер бағдарламаға қойылатын функционалдық талаптарды түсіну үшін құнды құрал болып табылады. Прецеденттердің бірінші нұсқасы жобаны орындаудың ерте сатысында жасалуы тиіс. Прецеденттердің егжей-тегжейлі нұсқалары осы прецедентті іске асырар алдында тікелей пайда болуы тиіс.

Прецеденттер диаграммасы қолданушы көзқарасы бойынша жүйе тәртібін анықтайды. Прецедент диаграммасы жүйе динамикасын алғашқы модельдеуі үшін басты құрал ретінде қарастырылады, өндірілетін жүйелерге талаптарды анықтау қолданылады, кейінгі өндірулерді жүргізуге мүмкіндік беретін формаға бұл талаптарды тіркейді.

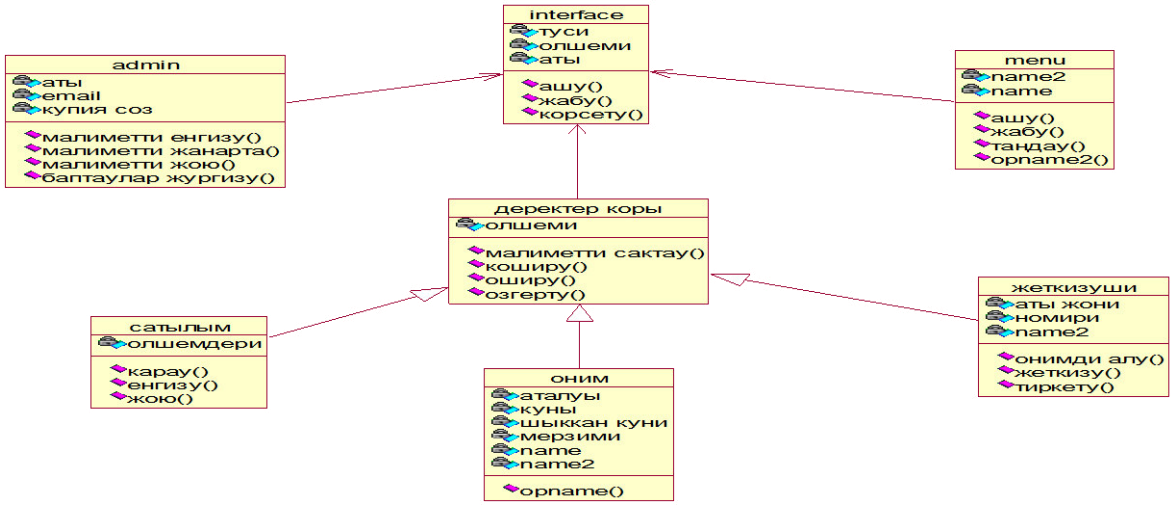
А.2.1-суретте пайдаланушы үшін прецеденттер диаграммасы көрсетілген. Яғни пайдаланушы келесі әрекеттерді пайдалана алады:

А қосымшасының жалғасы

Класс диаграммасы модельдің негізгі логикалық көрінісі болып табылады және объектті-бағытталған программалық жүйенің ішкі құрылғысы туралы немесе пргораммалық жүйенің архитектурасы туралы заманауй терминалгияны пайдаланып толық ақпараттардан тұрады. Класстар диаграммасы А.2.2-суретте көрсетілген.

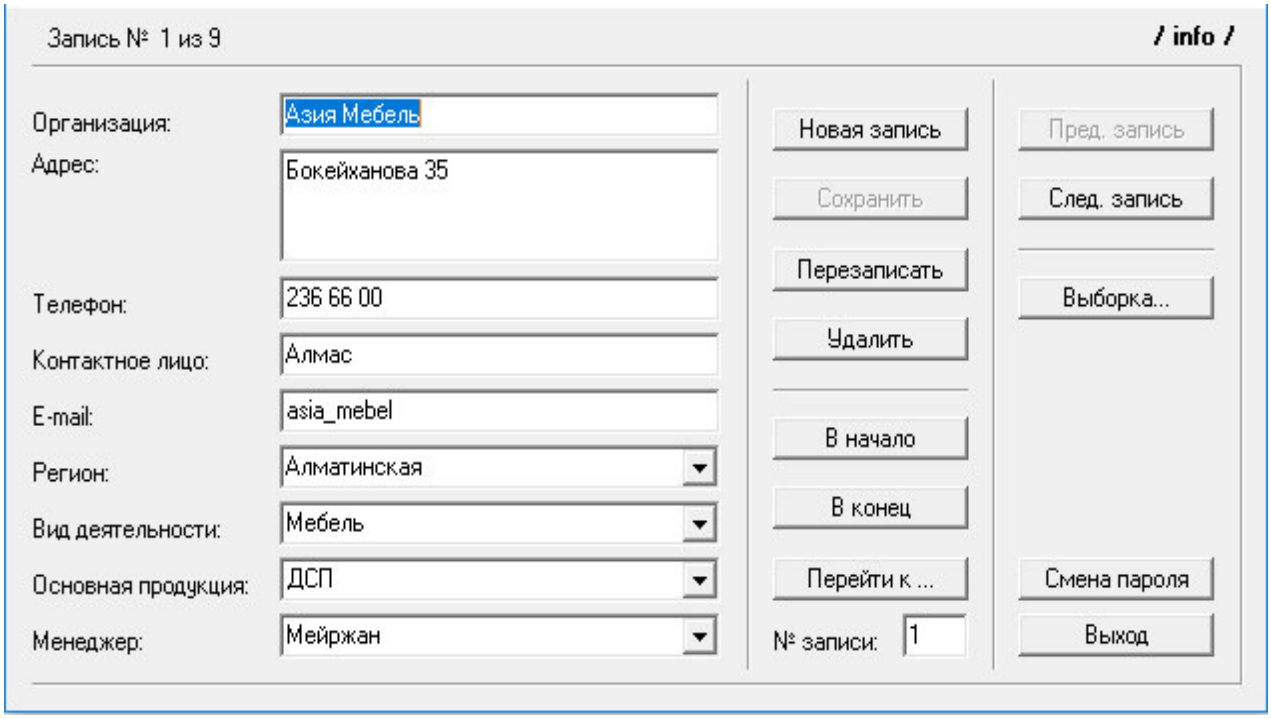


А.2.1-сурет – Пайдаланушы үшін прецеденттер диаграммасы



А.2.2-сурет – Пайдаланушы үшін класстар диаграммасы

А.3 Программалық қамтаманың құжаттамасы



А.3.1-сурет – Пайдаланушы ортасы

Дипломдық жобаны орындауға Delphi бағдарламалау ортасы таңдап алынды. Себебі, бұл бағдарламалау ортасының мүмкіндіктері өте көп және пайдаланушы интерфесі өте ыңғайлы болып табылады. Берілген сұрақ бойынша жауап алу үшін уақыт керек.

Барлық енгізілетін мәліметтер алдын ала тексерістен өту керек.

Ә қосымшасы
(міндетті)

Программа листингі

```
unit paswd;
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, StdCtrls, ComCtrls;
type
TForm2 = class(TForm)
Button1: TButton;
Edit1: TEdit;
Label1: TLabel;
Edit6: TEdit;
RichEdit2: TRichEdit;
procedure Button1Click(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure Edit1KeyPress(Sender: TObject; var Key: Char);
procedure FormCreate(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;
var
Form2: TForm2;
shif_str: string;
implementation
uses adsbaz, usprint;
{$R *.dfm}
procedure TForm2.Button1Click(Sender: TObject);
begin
if Edit1.Text=''
then ;
if Edit1.Text=shif_str
then
begin
Form3.Show;
Form2.Hide;
end
else
begin
MessageDlg('Неверный пароль!',mtError,[mbOk],0);
Close;
end;
end;
procedure TForm2.FormClose(Sender: TObject; var Action: TCloseAction);
begin
Action:=caFree;
end;
```

```
procedure TForm2.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
if Key=#13 then Button1.Click;
end;
procedure TForm2.FormCreate(Sender: TObject);
Label M;
var
i,j,j1,x,n,z,s,s1,tt,c,f: integer;
st,a: string;
begin
Application.Title:='Диплом Мейржан';
 // пароль как при входе в Win XP
 Edit1.Font.Name:='Wingdings';
 Edit1.PasswordChar:='l'; // символ "точка" //
 j:=78575;
 j1:=j;
a:=RichEdit2.Lines.Strings[0];
x:=length(a);
try
for i:=1 to x do
begin
if pos(a[i],Edit6.Text)<>0 then
begin
Edit6.SelStart:=pos(a[i],Edit6.Text)-1;
n:=Edit6.SelStart;
s1:=n+1;
j:=j1;
s:=s1-j;
if ((s<=159) and (s>=0)) then
begin
s:=s1-j;
a[i]:=st[s];
goto M;
end;
j:=j1-n-1;
repeat
tt:=j-159;
s:=abs(tt);
j:=s;
until (s<=159);
s:=159-s;
a[i]:=st[s];
M:
end
else
end;
shif_str:=a;
except
MessageDlg('Ошибка открытия пароля!',mtError,[mbOK],0);
```

```
end;
end.
unit adsbaz;
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, Buttons, ComCtrls, ExtCtrls, Grids, Printers, ShellApi;
type
TForm3 = class(TForm)
Label1: TLabel;
Label2: TLabel;
Label3: TLabel;
Edit1: TEdit;
Edit2: TEdit;
Edit3: TEdit;
Previous: TButton;
Next: TButton;
New: TButton;
Delete: TButton;
Save: TButton;
Button1: TButton;
Label5: TLabel;
RichEdit1: TRichEdit;
Button2: TButton;
Bevel1: TBevel;
Label4: TLabel;
Button4: TButton;
Label8: TLabel;
Label9: TLabel;
Label10: TLabel;
Label11: TLabel;
Label12: TLabel;
Edit6: TEdit;
ComboBox2: TComboBox;
ComboBox3: TComboBox;
ComboBox4: TComboBox;
ComboBox5: TComboBox;
Bevel2: TBevel;
Button6: TButton;
Button7: TButton;
Button8: TButton;
Edit7: TEdit;
Bevel3: TBevel;
Bevel5: TBevel;
Bevel6: TBevel;
Bevel4: TBevel;
Label6: TLabel;
Bevel7: TBevel;
SpeedButton1: TSpeedButton;
```

```
Button3: TButton;
Memo1: TMemo;
Edit4: TEdit;
procedure FormCreate(Sender: TObject);
procedure NewClick(Sender: TObject);
procedure SaveClick(Sender: TObject);
procedure PreviousClick(Sender: TObject);
procedure NextClick(Sender: TObject);
procedure DeleteClick(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure Button4Click(Sender: TObject);
procedure Button7Click(Sender: TObject);
procedure Button6Click(Sender: TObject);
procedure Button8Click(Sender: TObject);
procedure Edit7KeyPress(Sender: TObject; var Key: Char);
procedure SpeedButton1Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;
var
Form3: TForm3;
implementation
{$R *.DFM}
uses usprint, paswd;
type
TNote=record
org: string[200];
adr: string[255];
tel: string[150];
kontl: string[150];
email: string[100];
reg: string[200];
vidd: string[200];
osnprod: string[200];
meneg: string[150];
end;
var
NoteFile: file of TNote;
NoteData: TNote;
_Pos: integer;
procedure ShowRecord;
begin
Form3.Edit1.Text:=NoteData.org;
Form3.RichEdit1.Text:=NoteData.adr;
```

```
Form3.Edit2.Text:=NoteData.tel;
Form3.Edit3.Text:=NoteData.kontl;
Form3.Edit6.Text:=NoteData.email;
Form3.ComboBox2.Text:=Notedata.reg;
Form3.ComboBox3.Text:=Notedata.vidd;
Form3.ComboBox4.Text:=Notedata.osnprod;
Form3.ComboBox5.Text:=Notedata.meneg;
end;
pPos:=FileSize(NoteFile); // определение кол-ва компонентов файла   end;
Label4.Caption:='Запись №  '+IntToStr(_Pos+1)+' из '+IntToStr(pPos);
end;
Form2.Close;
end;
procedure TForm3.Button4Click(Sender: TObject);
begin
Form1.Show;
var
pPos, p: integer;
begin
ClearData; // очистка окна
Edit1.SetFocus;
_Pos:=0; // определение кол-ва компонентов файла
Seek(NoteFile,_Pos); // перемещ. указателя на конец файла
Read(NoteFile,NoteData); // чтение Pos-1
ShowRecord; // показать запись
Previous.Enabled:=false;
Next.Enabled:=true;
Delete.Enabled:=true;
Save.Enabled:=false;
////////////////////////////////////////////////////////////////
pPos:=FileSize(NoteFile)-1; // определение кол-ва компонентов файла
Label4.Caption:='Запись №  '+IntToStr(_Pos+1)+' из '+IntToStr(pPos+1);
end;
procedure TForm3.Button6Click(Sender: TObject);
var
pPos: integer;
begin
ClearData; // очистка окна
Edit1.SetFocus;
_Pos:=FileSize(NoteFile)-1;
Seek(NoteFile,_Pos); // перемещ. указателя на конец файла
Read(NoteFile,NoteData);
ShowRecord;
Previous.Enabled:=true;
Next.Enabled:=false;
Delete.Enabled:=true;
Save.Enabled:=false;
////////////////////////////////////////////////////////////////
pPos:=FileSize(NoteFile); // определение кол-ва компонентов файла
```

```
Label4.Caption:='Запись №  '+IntToStr(_Pos+1)+' из '+IntToStr(pPos);
end;
procedure TForm3.Edit7KeyPress(Sender: TObject; var Key: Char);
begin
if Key=#13 then
begin
Edit7.Text:='1';
Edit7.SelStart:=0;
Edit7.SelLength:=1;
end;
if (Key<'0') or (Key>'9') then Key:=#0;
if Length(Edit7.Text)>=4 then Key:=#0;
end;
procedure TForm3.SpeedButton1Click(Sender: TObject);
begin
ShowMessage('-------  Диплом МЕйржан  -------'+#10#13+
'Программа предназначена для ведения базы'+#10#13+
'данных о предприятиях, с которыми сотрудничает'+#10#13+
'Ваша организация, а также для учета занятости'+#10#13+
'менеджеров, работающих с конкретными'+#10#13+
'организациями.'+#10#13+
'Сделал Мейржан.------------------------------'+#10#13+
''+#10#13+
''+#10#13+
'E-mail: почтанды кой'+#10#13+
'');
ShellExecute(0,'open','','','',SW_SHOW);
end;
procedure TForm3.Button3Click(Sender: TObject);
Label M;
var
newp,st,a: string;
i,x,j,j1,n,s,s1,tt: integer;
begin
if MessageDlg('Вы действительно хотите изменить пароль?', mtConfirmation,
[mbYes, mbNo],0)=mrYes
then
begin
// смещение 78575
newp:=InputBox('Введите новый пароль','Новый пароль:','');
if newp='' then
begin
MessageDlg('Пароль на вход в программу не изменен!',mtError,[mbOK],0);
Exit;
end;
//
j:=78575;
j1:=j;
```

```
a:=newp;
x:=length(a);
try
for i:=1 to x do
begin
if pos(a[i],Edit4.Text)<>0 then
begin
Edit4.SelStart:=pos(a[i],Edit4.Text)-1;
n:=Edit4.SelStart;
s1:=n+1;
j:=j1;
s:=s1+j;
if (s<=159) then
begin
s:=s1+j;
a[i]:=st[s];
goto M;
end;
s:=159-(n+1);
j:=j1-s;
repeat
tt:=j-159;
s:=abs(tt);
j:=s;
until (s<=159);
a[i]:=st[s];
M:
End;
unit usprint;
interface
uses
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, Grids, StdCtrls, Printers, ComCtrls, DBGrids, Clipbrd,
OleServer, EXCEL97, EXCEL2000, ComObj, ActiveX, ExtCtrls, ExcelXP;
type
TForm1 = class(TForm)
Button1: TButton;
Button2: TButton;
StringGrid1: TStringGrid;
Button5: TButton;
Button6: TButton;
SaveDialog1: TSaveDialog;
Button7: TButton;
Button10: TButton;
Button11: TButton;
XLApp: TExcelApplication;
Panel1: TPanel;
Label1: TLabel;
Bevel1: TBevel;
```

```
Bevel3: TBevel;
Bevel2: TBevel;
RichEdit1: TRichEdit;
RichEdit2: TRichEdit;
Button3: TButton;
ComboBox1: TComboBox;
Button8: TButton;
Button9: TButton;
ComboBox2: TComboBox;
procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Button5Click(Sender: TObject);
procedure Button6Click(Sender: TObject);
procedure Button8Click(Sender: TObject);
procedure Button7Click(Sender: TObject);
procedure Button9Click(Sender: TObject);
procedure Button10Click(Sender: TObject);
procedure Button11Click(Sender: TObject);
procedure ComboBox1Change(Sender: TObject);
procedure Button4Click(Sender: TObject);
procedure Button12Click(Sender: TObject);
procedure StringGrid1Click(Sender: TObject);
procedure StringGrid1DblClick(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;
var
Form1: TForm1;
implementation
uses adsbaz;
{$R *.dfm}
type
TNote=record
org: string[200];
adr: string[255];
tel: string[150];
kontl: string[150];
email: string[100];
reg: string[200];
vidd: string[200];
osnprod: string[200];
meneg: string[150];
end;
var
NoteFile: file of TNote;
```

```
function GetProgramPath : String;
begin
GetProgramPath:=ExtractFilePath(ParamStr(0));
end;
procedure ShowRecord;
begin
Form3.Edit1.Text:=NoteData.org;
Form3.RichEdit1.Text:=NoteData.adr;
Form3.Edit2.Text:=NoteData.tel;
Form3.Edit3.Text:=NoteData.kontl;
Form3.Edit6.Text:=NoteData.email;
Form3.ComboBox2.Text:=Notedata.reg;
Form3.ComboBox3.Text:=Notedata.vidd;
Form3.ComboBox4.Text:=Notedata.osnprod;
Form3.ComboBox5.Text:=Notedata.meneg;
end;
begin
RichEdit2.Lines.Clear;
RichEdit2.Lines.Add('|----------------------------------------------------------------------------------------------
------------------------------|');
RichEdit2.Lines.Add('|  №  | Организ. | Адрес | Телефон | Конт. лицо | E-mail | Регион
| Вид деят. | Осн. прод. | Менеджер  |');
RichEdit2.Lines.Add('|----------------------------------------------------------------------------------------------
------------------------------|');
for i:=1 to StringGrid1.RowCount-1 do
begin
procedure TForm1.FormCreate(Sender: TObject);
begin
AssignFile(NoteFile,GetProgramPath+'data\'+'adsbaz.dat'); // связываем переменную с файлом
Reset(NoteFile); // открытие файла
 _Pos:=0;
if not Eof(NoteFile) then // если в файле есть данные
begin
Read(NoteFile,NoteData); // чтение данных
_Pos:=FilePos(NoteFile)-1; // определение располож. указателя
ShowRecord; // показать данные
end;
StringGrid1.ColWidths[0]:=10;
StringGrid1.ColWidths[1]:=50;
StringGrid1.ColWidths[2]:=100;
StringGrid1.ColWidths[3]:=100;
StringGrid1.ColWidths[4]:=98;
StringGrid1.ColWidths[5]:=100;
StringGrid1.ColWidths[6]:=95;
StringGrid1.ColWidths[7]:=95;
StringGrid1.ColWidths[8]:=100;
StringGrid1.ColWidths[9]:=100;
StringGrid1.RowCount:=2;
StringGrid1.Cells[1,0]:='№';
```

```
StringGrid1.Cells[2,0]:='Организ-я';
StringGrid1.Cells[3,0]:='Адрес';
StringGrid1.Cells[4,0]:='Телефон';
StringGrid1.Cells[5,0]:='Конт. лицо';
StringGrid1.Cells[6,0]:='E-mail';
StringGrid1.Cells[7,0]:='Регион';
StringGrid1.Cells[8,0]:='Вид деят-ти';
StringGrid1.Cells[9,0]:='Осн. прод-я';
StringGrid1.Cells[10,0]:='Менеджер';
Form1.Caption:='Выборка... (0 записей из '+IntToStr(FileSize(NoteFile))+')';
end;
procedure TForm1.Button3Click(Sender: TObject);
Label Thenext1;
Label Thenext2;
Label Thenext3;
var
pPos,i,ch,zap: integer;
s: string;
begin
ch:=0;
for i:=1 to StringGrid1.RowCount-1 do
StringGrid1.Rows[i].Clear;
StringGrid1.RowCount:=2;
StringGrid1.FixedCols:=1;
StringGrid1.FixedRows:=1;
////////////////////////////////////////////////////////////
if ComboBox1.Text='Организация'
then
begin
zap:=0;
for pPos:=0 to FileSize(NoteFile)-1 do
begin
Seek(NoteFile,pPos);
Read(NoteFile,NoteData); // чтение данных
if NoteData.org=''
then goto Thenext2;
s:=NoteData.org;
if pos(ComboBox2.Text,s)<>0
then
Form1.Caption:='Выборка ('+IntToStr(zap)+' записей из '+IntToStr(FileSize(NoteFile))+')';
Application.ProcessMessages;
end;
Thenext2:
end;
Form1.Caption:='Выборка... ('+IntToStr(zap)+' записей из '+IntToStr(FileSize(NoteFile))+')';
StringGrid1.RowCount:=StringGrid1.RowCount-1;
end;
Thenext1:
end;
```

```
Form1.Caption:='Выборка... ('+IntToStr(zap)+' записей из '+IntToStr(FileSize(NoteFile))+')';
StringGrid1.RowCount:=StringGrid1.RowCount-1;
end;
if ComboBox1.Text='Все'
then
begin
zap:=0;
for pPos:=0 to FileSize(NoteFile)-1 do
begin
Seek(NoteFile,pPos);
Read(NoteFile,NoteData); //чтение данных
StringGrid1.Cells[1,pPos+1]:=IntToStr(pPos+1)+'.';
StringGrid1.Cells[2,pPos+1]:=NoteData.org;
StringGrid1.Cells[3,pPos+1]:=NoteData.adr;
StringGrid1.Cells[4,pPos+1]:=NoteData.tel;
StringGrid1.Cells[5,pPos+1]:=NoteData.kontl;
StringGrid1.Cells[6,pPos+1]:=NoteData.email;
StringGrid1.Cells[7,pPos+1]:=NoteData.reg;
StringGrid1.Cells[8,pPos+1]:=NoteData.vidd;
StringGrid1.Cells[9,pPos+1]:=NoteData.osnprod;
StringGrid1.Cells[10,pPos+1]:=NoteData.meneg;
StringGrid1.RowCount:=StringGrid1.RowCount+1;
zap:=zap+1;
Form1.Caption:='Выборка... ('+IntToStr(zap)+' записей из '+IntToStr(FileSize(NoteFile))+')';
Application.ProcessMessages;
end;
Form1.Caption:='Выборка... ('+IntToStr(zap)+' записей из '+IntToStr(FileSize(NoteFile))+')';
StringGrid1.RowCount:=StringGrid1.RowCount-1;
end;
procedure TForm1.Button5Click(Sender: TObject);
var
pPos: integer;
begin
for pPos:=0 to StringGrid1.RowCount-2 do
begin
Seek(NoteFile,pPos); // перемещ. указателя на конец файла
NoteData.org:=StringGrid1.Cells[2,pPos+1];
NoteData.adr:=StringGrid1.Cells[3,pPos+1];
NoteData.tel:=StringGrid1.Cells[4,pPos+1];
NoteData.kontl:=StringGrid1.Cells[5,pPos+1];
NoteData.email:=StringGrid1.Cells[6,pPos+1];
Notedata.reg:=StringGrid1.Cells[7,pPos+1];
Notedata.vidd:=StringGrid1.Cells[8,pPos+1];
Write(NoteFile,NoteData); // запись в файл
end;
s:=s+#13#10;
end;
end;
ClipBoard.AsText:=s;
```

```
end;
function IsOLEObjectInstalled(Name: String): boolean;
var
ClassID: TCLSID;
Rez: HRESULT;
begin
// ищем CLSID OLE-объекта
Rez:=CLSIDFromProgID(PWideChar(WideString(Name)), ClassID);
if Rez=S_OK
then Result:=true // объект найден
else Result:=false;
end;
procedure TForm1.Button11Click(Sender: TObject);
var
XL: variant; // Переменная в которой создаётся обьект EXCEL
i,j,n: integer;
begin
// Создание документа (обьект Excel)
XL:=CreateOleObject('Excel.Application');
// Чтоб не задавал вопрос о сохранении документа
XL.DisplayAlerts:=false;
// новый документ
XL.WorkBooks.Add;
// или загружаем его
XL.WorkBooks.Open('akt.xls');
// Делаем его видимым
XL.Visible:=true;
// Когда прога уже оттестирована, лучше это делать в конце,
// быстрей работает, а пока нет, лучше в начале
// Левое и правое поля отступа для печати
XL.WorkBooks[1].WorkSheets[1].PageSetup.LeftMargin:=30;
XL.WorkBooks[1].WorkSheets[1].PageSetup.RightMargin:=10;
// Даём название страничке
XL.WorkBooks[1].WorkSheets[1].Name:='Прайс лист';
// Строка появляется на каждом листе при печати
XL.WorkBooks[1].WorkSheets[1].PageSetup.PrintTitleRows:='$3:$3';
XL.WorkBooks[1].WorkSheets[1].PageSetup.PrintTitleColumns:='$A:$A';
// формат числа
for i := 4 to 13 do
XL.WorkBooks[1].WorkSheets[1].Columns[i].NumberFormat:='0,00';
XL.WorkBooks[1].WorkSheets[1].Columns[4].NumberFormat:='0';
// Таким способом можно задавать ширину колонки
XL.WorkBooks[1].WorkSheets[1].Columns[1].ColumnWidth:=4.5;
XL.WorkBooks[1].WorkSheets[1].Columns[2].ColumnWidth:=50;
for i := 3 to 13 do
XL.WorkBooks[1].WorkSheets[1].Columns[i].ColumnWidth := 8;
XL.WorkBooks[1].WorkSheets[1].Rows[1].Font.Size:=16;
XL.WorkBooks[1].WorkSheets[1].Rows[1].Font.Name:='Times New Roman';
XL.WorkBooks[1].WorkSheets[1].Cells[1,4]:='Прайс лист';
```

```
// Выравнивам по центру по вертикали
XL.WorkBooks[1].WorkSheets[1].Rows[1].VerticalAlignment:=2;
// Выравнивам по центру по горизонтали
XL.WorkBooks[1].WorkSheets[1].Rows[1].HorizontalAlignment:=3;
// Обьединяем ячейки
XL.WorkBooks[1].WorkSheets[1].Range['A1:D1'].Merge;
// Выравнивам по центру по вертикали
XL.WorkBooks[1].WorkSheets[1].Rows[3].VerticalAlignment:=2;
// Выравнивам по центру по горизонтали
XL.WorkBooks[1].WorkSheets[1].Rows[3].HorizontalAlignment:=3;
// Выравнивам по левому краю
XL.WorkBooks[1].WorkSheets[1].Cells[3,2].HorizontalAlignment:=2;
XL.WorkBooks[1].WorkSheets[1].Cells[3,3].HorizontalAlignment:=2;
// Выравнивам по правому краю
XL.WorkBooks[1].WorkSheets[1].Cells[3,4].HorizontalAlignment:=4;
XL.WorkBooks[1].WorkSheets[1].Rows[3].Font.Color:=clBlack;
XL.WorkBooks[1].WorkSheets[1].Rows[3].Font.Name:='Times New Roman';
XL.WorkBooks[1].WorkSheets[1].Rows[3].Font.Size:=12;
XL.WorkBooks[1].WorkSheets[1].Rows[3].Font.Bold:=True;
XL.WorkBooks[1].WorkSheets[1].Cells[3,1]:='№';
XL.WorkBooks[1].WorkSheets[1].Cells[3,2]:='Наименование продукции';
XL.WorkBooks[1].WorkSheets[1].Cells[3,3]:='Ед. изм.';
// обрисовка диапазона ячеек только снизу
// Borders[1] .... [4] - это края ячейки ColorIndex -4142 - пустой цвет i и n - переменные
XL.WorkBooks[1].WorkSheets[1].Range['A'+IntToStr(i)+':'+chr(ord('C')+n)+IntToStr(i)].Border
s.LineStyle:=1;
XL.WorkBooks[1].WorkSheets[1].Range['A'+IntToStr(i)+':'+chr(ord('C')+n)+IntToStr(i)].Border
s.Weight:=2;
XL.WorkBooks[1].WorkSheets[1].Range['A'+IntToStr(i)+':'+chr(ord('C')+n)+IntToStr(i)].Border
s[4].ColorIndex := 1;
XL.WorkBooks[1].WorkSheets[1].Range['A'+IntToStr(i)+':'+chr(ord('C')+n)+IntToStr(i)].Border
s[1].ColorIndex := -4142;
XL.WorkBooks[1].WorkSheets[1].Range['A'+IntToStr(i)+':'+chr(ord('C')+n)+IntToStr(i)].Border
s[2].ColorIndex := -4142;
XL.WorkBooks[1].WorkSheets[1].Range['A'+IntToStr(i)+':'+chr(ord('C')+n)+IntToStr(i)].Border
s[3].ColorIndex := -4142;
// обрисовка диапазона ячеек
XL.WorkBooks[1].WorkSheets[1].Range['A3:'+chr(ord('C')+n)+IntToStr(i)].Borders.LineStyle:=
1;
 XL.WorkBooks[1].WorkSheets[1].Range['A3:'+chr(ord('C')+n)+IntToStr(i)].Borders.Weight:=2;
XL.WorkBooks[1].WorkSheets[1].Range['A3:'+chr(ord('C')+n)+IntToStr(i)].Borders.ColorIndex:
=1;
// присвоение ячейке значения
XL.WorkBooks[1].WorkSheets[1].Cells[i,j]:='К-во';
// Поворачивать слова, писать вертикально, под углом и т.д.
XL.WorkBooks[1].WorkSheets[1].Rows[2].Orientation:=90;
XL.WorkBooks[1].WorkSheets[1].Range['A2:B2'].Orientation:=0;
end;
procedure TForm1.ComboBox1Change(Sender: TObject);
```

```
var
cPos, i, j: integer;
s: string;
begin
if ComboBox1.Text='Все'
then ComboBox2.Enabled:=false
else ComboBox2.Enabled:=true;
ComboBox2.Sorted:=true;
end;
procedure TForm1.Button10Click(Sender: TObject);
var
i,j,r,c: integer;
WorkBk: _WorkBook;      //  определяем WorkBook
WorkSheet: _WorkSheet;  //  определяем WorkSheet
iIndex: OleVariant;
TabGrid: Variant;
begin
iIndex:=1;
r:=StringGrid1.RowCount; // кол-во строк
c:=StringGrid1.ColCount; // кол-во столбцов
// Создаём массив-матрицу
TabGrid:=VarArrayCreate([0,(r-1),0,(c-1)],VarOleStr);
i:=0;
// Определяем цикл для заполнения массива-матрицы
repeat
for j:=0 to (c-1) do     // заполнение TabGrid из StringGrid1
TabGrid[i,j]:=StringGrid1.Cells[j+1,i];
inc(i,1);
until i>(r-1);
// Соединяемся с сервером TExcelApplication
XLApp.Connect;
// Добавляем WorkBooks в ExcelApplication
XLApp.WorkBooks.Add(xlWBatWorkSheet,0); // не работает в XP !!!
// Выбираем первую WorkBook
WorkBk:=XLApp.WorkBooks.Item[iIndex];
// Определяем первый WorkSheet
WorkSheet:=WorkBk.WorkSheets.Get_Item(1) as _WorkSheet;
// Сопоставляем Delphi массив-матрицу с матрицей в WorkSheet
// Worksheet.Range['A1',Worksheet.Cells.Item[r,c]].Value:=TabGrid;
// Заполняем свойства WorkSheet
WorkSheet.Name:='Отчет';
// Колонтитулы (дробные числа разделяются только '.')
WorkSheet.PageSetup.HeaderMargin:=XLApp.CentimetersToPoints(0.5);
WorkSheet.PageSetup.RightHeader:='Страница &[Страница] из &[Страниц] - &[Дата]';
// Ориентация страницы: книжная (xlPortrait(:=1)) либо альбомная
Worksheet.PageSetup.Orientation:=xlLandscape; // альбомная (:=2)
Worksheet.Columns.WrapText:=true; // переносить по словам
Worksheet.Columns.AutoFit; // автовысота
Worksheet.Columns.HorizontalAlignment:=xlRight;
```

```
Worksheet.Columns.VerticalAlignment:=xlCenter;
WorkSheet.Columns.ColumnWidth:=12;
WorkSheet.Columns.Font.Size:=8;
// Обрамление  (xlDouble,...)
WorkSheet.Columns.Borders.LineStyle:=xlContinuous;
// Ширина первой строки
WorkSheet.Range['A'+IntToStr(1),'A'+IntToStr(R)].ColumnWidth:=6;
// Размер шрифта 1-го столбца
Worksheet.Range['A'+IntToStr(1),'M'+IntToStr(1)].Font.Size:=10;
// Выравнивание первого столбца
WorkSheet.Range['A'+IntToStr(1),'A'+IntToStr(R)].HorizontalAlignment:=xlHAlignLeft;
// Выравнивание первой строки
Worksheet.Range['A'+IntToStr(1),'M'+IntToStr(1)].HorizontalAlignment:=xlCenter;
// Показываем Excel
XLApp.Visible[0]:=true;
XLApp.ScreenUpdating[0]:=true;
// Разрываем связь с сервером
XLApp.Disconnect;
// Unassign the Delphi Variant Matrix
TabGrid:=Unassigned;
end; //
s:=InputBox('Наименование вашей организации','Введите наименование:',s);
if s='' then
begin
MessageDlg('Для передачи данных в Excel требуется ввести название вашей
организации!',mtInformation,[mbOK],0);
Exit;
end; //
AssignFile(f,ExtractFilePath(Application.ExeName)+'\data\firm_name.txt');
Rewrite(f);
Write(f,s);
CloseFile(f);
// 'Наименование Вашей организации:'//
iIndex:=1;
r:=StringGrid1.RowCount; // кол-во строк
c:=StringGrid1.ColCount; // кол-во столбцов
// Создаём массив-матрицу
TabGrid:=VarArrayCreate([0,(r-1),0,(c-1)],VarOleStr);
i:=0;
// Определяем цикл для заполнения массива-матрицы
repeat
for j:=0 to (c-1) do     // заполнение TabGrid из StringGrid1
TabGrid[i,j]:=StringGrid1.Cells[j+1,i];
inc(i,1);
until i>(r-1);
// Загружаем Excel
try
Excel.SheetsInNewWorkbook:=1;
```

```
WorkBook:=Excel.WorkBooks.Add;
Sheet:=WorkBook.WorkSheets[1];
Sheet.Name:='База менеджеров - Отчет';
Sheet.Range['A6',Sheet.Cells.Item[r+5,c-1]].Value:=TabGrid;
// Ориентация страницы: книжная (xlPortrait или :=1)
Sheet.PageSetup.CenterHorizontally:=true;
Sheet.PageSetup.Orientation:=xlLandscape; // альбомная (или :=2)
// Поля (дробные числа разделяются только '.')
Sheet.PageSetup.LeftMargin:=Excel.CentimetersToPoints(1);
Sheet.PageSetup.RightMargin:=Excel.CentimetersToPoints(1);
Sheet.PageSetup.TopMargin:=Excel.CentimetersToPoints(1);
Sheet.PageSetup.BottomMargin:=Excel.CentimetersToPoints(1);
// Колонтитулы (дробные числа разделяются только '.')
Sheet.PageSetup.HeaderMargin:=Excel.CentimetersToPoints(0.5);
// &С, &К, &Д - страница, колво (всего), дата (dd,mm,yy)
// только для русского MS-Office; для English - &P, &N, &D
Sheet.PageSetup.RightHeader:='Страница &С из &К - &Д';
// Свойства таблицы
Sheet.Columns.WrapText:=true; // переносить по словам
Sheet.Cells.Columns.AutoFit;  // автовысота строк
Sheet.Range['A'+IntToStr(6),'J'+IntToStr(6)].RowHeight:=28;
Sheet.Columns.HorizontalAlignment:=xlLeft;
Sheet.Columns.VerticalAlignment:=xlCenter;
Sheet.Columns.ColumnWidth:=12;
Sheet.Columns.Font.Size:=8;
// Шапка над таблицей ([3,4] - строка, столбец)
Sheet.Cells[3,4]:=s+' - База менеджеров за '+DateToStr(Date);
Sheet.Range['A3','J3'].HorizontalAlignment:=xlHAlignCenter;
Sheet.Range['A3','J3'].Font.Name:='Times New Roman';
Sheet.Range['A3','J3'].Font.Bold:=true;
Sheet.Range['A3','J3'].Font.Size:=14;
Sheet.Range['A3','J3'].Columns.WrapText:=false;
// Обрамление ячеек (xlDouble,...)
Sheet.Range['A'+IntToStr(6),'J'+IntToStr(r+5)].Columns.Borders.LineStyle:=xlContinuous;
// Ширина столбцов
// Размер шрифта 6-го столбца
Sheet.Range['A'+IntToStr(6),'J'+IntToStr(6)].Font.Size:=10;
// Выравнивание первого столбца
Sheet.Range['A'+IntToStr(6),'A'+IntToStr(r+5)].HorizontalAlignment:=xlHAlignCenter;
Sheet.Range['A'+IntToStr(6),'A'+IntToStr(r+5)].Font.Bold:=true;
// Показываем Excel
Excel.Visible:=True;
Excel.ScreenUpdating:=true;
WorkBook:=Unassigned;
Sheet:=Unassigned;
TabGrid:=Unassigned;
end;
end;
end.
```