Тема 6. Візуальна розробка додатків баз даних

1. Механізми доступу до даних із засобів розробки

Існує кілька способів доступу до даних із засобів розробки і клієнтських додатків [22].

Більшість систем керування базами даних містить у своєму складі бібліотеки, що надають спеціальний ***прикладний програмний інтерфейс****(Application Programming Interface, API)* для доступу до даних цієї СКБД.

У цьому випадку створений додаток зможе використовувати дані тільки СКБД цього виробника, і заміна її на іншу (наприклад, з метою розширення сховища даних або переходу в архітектуру “клієнт-сервер”) призведе до переписування значної частини коду клієнтського додатка. Клієнтські API і об'єктні моделі не підкоряються ніяким стандартам і різні для різних СКБД.

Інший спосіб маніпуляції даними в додатку базується на застосуванні ***універсальних механізмів*** доступу до даних. Універсальний механізм доступу до даних звичайно реалізований у вигляді бібліотек і додаткових модулів, називаних ***драйверами*** або ***провайдерами***, що нерідко підкоряється тій або іншої специфікації. Додаткові модулі, специфічні для тієї або іншої СКБД, реалізують безпосереднє звертання до функцій клієнтського API конкретних СКБД.

Найбільш популярними серед універсальних механізмів доступу до даних є наступні:

- Open Database Connectivity (ODBC).
- OLE DB.
- Active Data Objects (ADO).
- Borland Database Engine (BDE).

Універсальні механізми ODBC, OLE DB і ADO фірми Microsoft являють собою власне кажучи промислові стандарти. Що стосується механізму доступу до даних BDE фірми Borland, то він так і не став промисловим стандартом, однак донедавна застосовувався досить широко, тому що до виходу Delphi 5 і C++Builder 5 був практично єдиним універсальним механізмом доступу до даних, підтримуваним засобами розробки Borland на рівні компонентів і класів.

У загальному випадку додаток, що використовує бази даних, може застосовувати, наприклад, наступні механізми доступу до них:

- безпосередній виклик функцій клієнтського API (або звертання до COM-об'єктів клієнтських бібліотек);
- виклик функцій ODBC API (або застосування класів, що інкапсулюють подібні виклики);
- безпосереднє звертання до інтерфейсів OLE DB;
- застосування ADO (або застосування класів, що інкапсулюють звертання до об'єктів ADO);
- застосування ADO + OLE DB + ODBC;
- застосування BDE + SQL Links (або застосування класів, що інкапсулюють звертання до функцій BDE);
- застосування BDE + ODBC Link + ODBC.

Крім цих існують і інші способи доступу до даних.

Використовуючи C++Builder, можна створювати додатки, які працюють як з ***автономними*** (автономні локальні бази даних зберігають свої дані в локальній файловій системі на тому же комп'ютері, що і СКБД, мережа не використовується), так і з ***серверними*** СКБД, такими як Oracle, Sybase, Interbase, MS SQL, Informix, DB2, а також з ODCB-джерелами (Open Database Connectivity).

Робота з даними в C++Builder в основному виконується через **Borland Database Engine** (BDE) процесор баз даних фірми Borland. Відповідна програма має бути встановлена на комп'ютері користувача. BDE є посередником між додатком та базами даних. Він надає користувачу єдиний інтерфейс для роботи, що звільняє від необхідності знати конкретну реалізації бази даних. Завдяки цьому, не треба змінювати додаток при зміні реалізації бази даних. Додаток C++Builder звертається до бази даних через BDE.

Починаючи з C++Builder 5, введена інша альтернативна можливість роботи з базами даних, минаючи BDE. Це – розроблена в Microsoft технологія **ActiveX Data Objects** (ADO). ADO – це користувацький інтерфейс до будь-яких типів даних, включаючи бази даних, електронну пошту системні, текстові і графічні файли, Зв'язок з даними здійснюється засобами так званої технології OLE DB.

Ще один альтернативний доступ до баз даних Interbase був введений у C++Builder 5 на основі технології **InterBaseExpress** (IBX). У бібліотеці компонентів C++Builder 5 з'явилася сторінка *InterBase*, що містить компоненти роботи з InterBase, минаючи BDE. Ці компоненти забезпечують підвищену працездатність і дозволяють використовувати нові можливості сервера InterBase, недоступні звичайним компонентам BDE.

У C++Builder 6 введена ще одна нова технологія доступу до баз даних – **dbExpress.** Це драйвери, що забезпечують доступ до серверів SQL на основі єдиного інтерфейсу. При використанні dbExpress можна поставляти свій додаток разом з DLL відповідного драйвера й, якщо треба, з додатковими файлами, що містять інформацію про з'єднання.

У C++Builder 6 також розширені можливості побудови клієнтських додатків за рахунок появи нових компонентів BDEClientDataSet, SQLClientDataSet, IBClientDataSet і розширення можливостей компонента ClientDataSet. Нові компоненти SharedConnection і LocalConnection дозволяють створювати істотно більш гнучкі багатопоточні додатки із клієнтськими наборами даних.

У C++Builder 6 є окремі сторінки BDE, ADO, InterBase, dbExpress, які містять компоненти, що забезпечують організацію відповідних технологій.

При створенні додатків баз даних зручно не просто вказувати шлях доступу до таблиць бази даних, а використовувати для цього так званий ***псевдонім (аліас)***. Він зберігається в окремому конфігураційному файлі у довільному місці на диску і дозволяє виключити з програми пряму вказівку шляхів доступу до бази даних. Такий підхід дає можливість розташовувати дані в будь-якому місці, не перекомпільовуючи при цьому програму. Крім шляху доступу, в аліасі вказуються тип бази даних, драйвер і багато іншої керуючої

інформації. Тому використання аліасів дозволяє легко переходити, наприклад, від локальних баз даних до SQL-серверних баз при виконанні вимог поділу додатка на клієнтську і серверну частини.

1.1. Універсальний механізм доступу ODBC

ODBC (Open Database Connectivity) – широко розповсюджений програмний інтерфейс фірми Microsoft, що задовольняє стандартам ANSI і ISO для інтерфейсів звертань до баз даних (Call Level Interface, CLI) [22]. Для доступу до даних конкретної СКБД за допомогою ODBC, крім власне клієнтської частини цієї СКБД потрібний ODBC Administrator (додаток, що дозволяє визначити, які джерела даних доступні для даного комп'ютера за допомогою ODBC, і описати нові джерела даних), і ODBC-драйвер для доступу до цієї СКБД. ODBC-драйвер являє собою бібліотеку, що завантажується динамічно (DLL), яку клієнтський додаток може завантажити у свій адресний простір і використовувати для доступу до джерела даних. Для кожної використовуваної СКБД потрібний власний ODBC-драйвер, тому що ODBC-драйвери використовують функції клієнтських API, різні для різних СКБД.

За допомогою ODBC можна маніпулювати даними будь-якої СКБД (і навіть даними, що не мають прямого відношення до баз даних, наприклад, даними у файлах електронних таблиць або в текстових файлах), якщо для них є ODBC-драйвер. Для маніпуляції даними можна використовувати як безпосередні виклики ODBC API, так і інші універсальні механізми доступу до даних, наприклад OLE DB, ADO, BDE, що реалізують стандартні функції або класи на основі викликів ODBC API у драйверах або провайдерах, спеціально призначених для роботи з будь-якими ODBC-джерелами.

1.2. Доступ до баз даних через BDE

BDE (Borland Database Engine) – універсальний механізм доступу до даних, що застосовується у засобах розробки фірми Borland (а саме – Delphi і C++Builder), а також у деяких інших продуктах, наприклад Corel Paradox, Corel Quattro Pro, Seagate Software Crystal Reports [22].

BDE – це спадкоємець бібліотеки Paradox Engine, створеної для Borland Pascal і Borland C++ з метою надати додаткам, розробленим за їхньою допомогою, доступ до таблиць СКБД Paradox. Незабаром після створення Paradox Engine компанією Borland було розроблено кілька бібліотек-драйверів під загальною назвою SQL Links. Ці бібліотеки розширили функціональність BDE, дозволивши застосовувати набір функцій, який був в Paradox Engine, для доступу до даних dBase, ODBC-джерел, а також найбільш популярних серверних СКБД. Пізніше до цього набору були додані бібліотеки для доступу до Access і FoxPro.

Фізично BDE являє собою набір бібліотек доступу до даних, що реалізують BDE API – набір функцій для маніпуляції даними, що викликаються з додатка. Ці функції, у свою чергу, можуть звертатися до функцій клієнтського API (у випадку, наприклад, Oracle, Informix, IB Database) або ODBC API (Access 2000, Microsoft SQL Server 7.0, будь-які ODBC-джерела), а також безпосередньо маніпулювати файлами деяких СКБД (dBase, Paradox).

Для доступу до бази даних за допомогою BDE на комп'ютері, що містить клієнтський додаток, повинні бути встановлені бібліотеки BDE загального призначення, а також BDE-драйвер для даної СКБД. У випадку серверних СКБД такі драйвери звуться SQL-Links. Ці драйвери містять BDE API – стандартний набір функцій, створених на основі функцій клієнтських API відповідних СКБД.

Серед BDE-драйверів є драйвер, створений з використанням ODBC API, – так званий ODBC Link, що застосовується разом з ODBC-драйвером для обраної СКБД.

Для доступу за допомогою BDE до Access 2000 можна використовувати відповідний ODBC-драйвер і ODBC Link, при цьому на комп'ютері, де експлуатується додаток, що їх використовує, потрібна наявність Microsoft Jet Engine 4.0. Він входить до складу Microsoft Access 2000, а також до складу Microsoft Data Access Components (MDAC).

У C++Builder доступ до баз даних за допомогою BDE можна організувати, наприклад, через компонент TDatabase, вказавши в ньому псевдонім, а доступ до кожної таблиці відбувається через компоненту типу TTable (їх може бути багато, вони можуть бути пов’язані). В свою чергу зв’язок між таблицями і елементами керування (наприклад, типу TDBGrid) здійснюється через компонент TDataSource). Ці три компонента знаходяться на сторінках Data Access (TDataSource), BDE, Data Control. Їх можна помістити в один модуль даних – компонент TDataModule.

Зазначимо, що використання BDE – не найефективніший спосіб доступу до даних Access. Більш доцільно здійснювати доступ до даних Access за допомогою ADO і OLE DB, тому що ці механізми надають у порівнянні з BDE набагато більше функціональних можливостей.

1.3. Доступ до баз даних через ADO

Як було сказано вище, починаючи з C++Builder 5, з’явилася можливість роботи з базами даних засобом розробленої в Microsoft технології ActiveX Data Object (ADO). ADO – це користувацький інтерфейс до любих типів даних, включаючи реляційні та нереляційні бази даних, електронну пошту, системні, текстові і графічні файли. Зв'язок з даними здійснюється засобом так званої технології OLE DB.

Використання ADO являється альтернативою Borland Database Engine (BDE), яка забезпечує більш ефективне роботу з даними. Для використання цієї можливості на комп’ютері повинна бути встановлена система ADO (міститься в останніх версіях Windows). Крім того, повинна бути встановлена клієнтська система доступу до даних, наприклад, Microsoft SQL Server, а в ODBC повинен міститися драйвер OLE DB для того типу баз даних, з яким іде робота.

Для роботи з ADO в C++Builder передбачені компоненти, розміщенні на сторінці бібліотеки – ADO. Вони інкапсулюють такі об’єкти ADO, як Connection, Command і Recordset. Їм відповідають компоненти C++Builder TADOConnection, TADOCommand і TADODataSet.

Зв’язок з базою даних в технології ADO здійснюється звичайним ланцюжком: набір даних => джерело даних (компонент TDataSourse) => компоненти управління і відображення даних (TDBGrid, TDBEdit та ін.).

Перелічимо далі основні компоненти зі сторінки ADO палітри компонентів.

TADOConnection	використовується для зв’язку з набором даних ADO. Може працювати з кількома компонентами наборів даних як диспетчер виконання їх команд
TADODataSet	універсальний компонент зв’язку з наборами даних, який може працювати в різних режимах, заміняючи зв’язані з BDE компоненти TTable, TQuery, TStoredProc
TADOTable	використовується для роботи з одною таблицею. Може зв’язуватись з нею безпосередньо або через TADOConnection
TADOQuery	використовується для роботи з набором даних за допомогою запитів SQL, включаючи такі запити мови DDL, як CREATE TABLE
TADOStoredProc	використовується для виконання процедур, які зберігаються на сервері
TADOCommand	використовується в основному для виконання

	команд SQL, які не повертають множину результатів

Доступ до бази даних здійснюється або за допомогою рядка з'єднання – властивості ConnectionString, або за допомогою окремого компонента TADOConnection, ім'я якого задається у властивості Connection інших компонентів.

2. Взаємодія компонентів доступу до даних з інтерфейсними компонентами

Компоненти доступу до даних (data access control) містяться на сторінці палітри компонентів Data Access, BDE, ADO, InterBase, dbExpress. Вони забезпечують додатку доступ до баз даних, невидимі під час виконання програми (невізуальні). До них відносяться, наприклад, TDatabase, TADOTable, TTable,TDataSource, TQuery, TADOQuery.

Інтерфейсний елемент (data-aware control) – візуальний (видимий під час виконання програми) компонент, який дозволяє користувачу переглядати і змінювати дані в базі, використовуючи компоненти доступу до даних. Інтерфейсні компоненти знаходяться на вкладинці Data Controls палітри компонентів.

TDBGrid	використовується для табличного відображення даних
TDBNavigator	це набір зі звичайних кнопок розташова-них на єдиній панелі, дозволяють маніпулювати положенням курсору, додавання рядків, видалення й т.д.
TDBText	є аналогом компонента TLabel, але дозволяє встановлювати текст із певного стовпця БД
TDBEdit	є аналогом компонента TEdit (дозволяє демонструвати і вводити дані стовпця БД вручну)
TDBMemo	аналог багаторядкового редактора тексту
TDBImage	дозволяє відображати графічні зображен-ня, що утримуються у файлі бази даних
TDBListBox	дозволяє вибирати значення зі списку наявних значень
TDBComboBox	є поєднанням TDBEdit і TDBListBox, тіль-ки використовується спадаючий список
TDBCheckBox	активізує/деактивізує одне значення
TDBRadioGroup	сукупність компонентів (аналог TRadioButton)
TDBLookUpList	список значень, підібраних з іншого стовпця або іншої таблиці
TDBLookUpComboBox	аналогічно TDBLookUpList
TDBRichEdit	аналогічно TDBMemo, але має можли-вість більше детального відображення й редагування (колір, розмір)
TDBCtrlGrid	відображає записи таблиці
TDBChart	компонент використовується для побудови одно- і багато- серійних графіків

C++Builder підтримує "триступінчасту" модель розробки додатка баз даних. У цій моделі компонент керування зв'язаний з компонентом джерела, а той, у свою чергу, одержує фактичні дані таблиці або запиту.

Набір даних у C++ Builder – це об'єкт, що складається з набору записів, кожен з яких, у свою чергу, складається з полів, і покажчика поточного запису. Набір даних може мати повну відповідність з реально існуючою таблицею або бути результатом запиту, він може бути частиною таблиці або поєднувати між собою кілька таблиць.

Набір даних у C++Builder є нащадком абстрактного класу TDataSet Наприклад, класи TQuery, TTable і TStoredProc, що містяться на сторінці палітри компонентів Data Access, – спадкоємці TDBDataSet, що, у свою чергу, є спадкоємцем TDataSet. TDataSet містить абстракції, необхідні для безпосереднього керування таблицями або запитами, забезпечуючи засоби для того, щоб відкрити таблицю або виконати запит і переміщатися по рядках.

TTable, TADOTable – компоненти C++Builder, які забезпечують доступ до таблиць баз даних. Властивість TableName цих компонентів використовується для вибору конкретної таблиці, до якої потрібно звернутися.

TField – клас C++Builder, який забезпечує доступ до полів таблиці бази даних.

TQuery, TADOQuery – компоненти C++Builder, які дозволяють створювати і обробляти власні запити SQL.

TStoredProc – компонент C++Builder, який дозволяє запускати відкомпільовані процедури SQL, які знаходяться на сервері баз даних. Такі процедури називаються процедурами, що зберігаються (stored procedures).

TDataSource (сторінка Data Access) – компонент C++Builder, який забезпечує зв’язок компонентів, нащадків TDataSet, з інтерфейсними компонентами. Інтерфейсні елементи звертаються до компонентів TDataSource, які в свою чергу звертаються до компонентів доступу до даних (data access control) – нащадків TDataSet.

У додатку інтерфейсні компоненти звертаються до компонентів типу TDataSource. Як правило, в формі міститься невелика кількість компонентів типу TDataSource, проте інтерфейсних компонентів може бути багато. Компоненти TDataSource, у свою чергу, звертаються до одного або декількох компонентів, нащадків TDataSet.

У додатку не обов’язково використовувати компонент TDatabase для звертання до бази даних. Він надає деякі додаткові можливості, які не обов’язково використовувати, але він не є невід’ємною частиною додатків C++Builder, які працюють з базами даних. Нижче показаний можливий взаємозв’язок названих компонентів у випадку використання компонентів доступу до баз даних з вкладники BDE.

Рис. 2.

3. Етапи проектування додатка СКБД у найпростішому випадку

Нехай треба створити простий додаток СКБД з використанням для доступу до бази даних псевдоніму (доступ здійснювати через ADO або BDE), для компонентів доступу до даних – модуля даних, для інтерфейсної частини проекту – окремої форми з елементами відображення даних.

Процес створення такого додатку можна умовно поділити на п’ять етапів.

1. Налагодження джерел даних ODBC з використанням для цього *псевдоніма (аліаса)*.

Псевдонім (alias) містить всю інформацію, необхідну для забезпечення доступу до бази даних. Ця інформація повідомляється тільки один раз при створенні псевдоніма. А додаток для зв'язку з базою даних використовує псевдонім. У цьому випадку додатку байдуже, де фізично міститься та чи інша база даних, а часто байдужа і СКБД, що створила і обслуговує цю базу даних. При зміні системи каталогів, сервера тощо. в додатку нічого переробляти не треба. Досить, щоб адміністратор ввів відповідну інформацію у псевдонім.

Створити псевдонім до бази даних можна через адміністратор джерел даних ODBC. Для цього у панелі управління треба вибрати ***Администрирование | Источники данных ODBC*** (для Windows XP), далі вибрати сторінку ***Пользовательский (User DSN)*** і додати нове джерело даних, натиснувши кнопку***Добавить (Add)*** і вибравши у вікні, що відкриється, потрібний драйвер, наприклад, ***Microsoft Access Drives***. Після цього з’явиться наступне вікно, наприклад, ***ODBC Microsoft Access Setup*** з полем вводу ***Data Source Name***. Тут можна ввести ім’я псевдоніму, а за допомогою кнопки ***Select*** – вибрати базу даних, яка буде відповідати створеному псевдоніму.

2. Перенесення на спеціалізовану форму (модуль даних) компонента набору даних з вкладинки BDE (TTable або TQuery) або з вкладинки ADO (TADOTable або TADOQuery) із сторіки палітри Data Access і встановлення його властивостей у відповідності зі специфічними вимогами обраної бази даних (вказується, наприклад, псевдонім).

3. Перенесення на форму (модуль даних) компонента джерела даних TDataSource з вкладинки Data Access. При цьому у властивості DataSet указується властивість Name об'єкта набору даних з п.2 (наприклад, Table1 або Query1).

4. Перенесення на форму (інтерфейсна частина проекту) потрібних компонентів відображення і редагування даних із вкладки Data Controls. При цьому у властивості DataSource задається джерело даних (наприклад, DataSourcel). Відображуване поле набору даних вказується у властивості DataField.

5. Якщо на попередньому кроці обрано компонент сітка TDBGrid (тоді властивість DataField не потрібна), то треба додати на форму навігатор TDBNavigator, який використовується разом з компонентом TDBGrid.