CERCETĂRI PRIVIND MODALITĂȚILE DE INTERFAȚARE A APLICAȚIILOR SOFTWARE REALIZATE ÎN C++ CU BAZELE DE DATE

NEGREA Cătălin
Conducători ştiinţifici: Conf. dr. ing. **George ENCIU**, Asist. Univ. dr. ing. **Adrian POPESCU**

REZUMAT: Lucrarea prezintă paşii necesari pentru a putea creea o punte între o aplicație software realizată in C++ şi o bază de date oferind aplicației capacitatea de a citi şi de a modifica datele din baza de date în cadrul limitelor impuse pentru a asigura integritatea bazei de date. Astfle aplicația nu are dreptul la acțiuni care ar duce la blocarea bazei de date.

CUVINTE CHEIE: Aplicație C++, Baza de date, Interfață

1 INTRODUCERE:

O bază de date este constituită dintr-un ansamblu structurat de date evolutive, organizate pentru a fi exploatate de diferite programe (aplicaţii).

Deşi orice întreprindere face apel la baze de date pentru păstrarea şi gestionarea informaţiilor, câteva dintre aplicaţiile acestora sunt deosebit de spectaculoase:

• bazele de date ale liniilor aeriene care sunt accesate simultan din sute de agenţii pentru a realiza rezervări şi vânzări de locuri pentru date şi zboruri diferite;

• bazele de date ale băncilor care permit realizarea a mii de tranzacţii zilnic;

• bazele de date ale supermagazinelor care sunt accesate atât de la casele de marcaj cât şi de la echipamentele de inventariere;

• bazele de date ale bibliotecilor care păstrază milioane de titluri şi permit localizarea unei lucrări folosind diferite criterii (cuvinte cheie, titlu, autori, domeniu);

2 STADIUL ACTUAL:

Pentru realizarea unei aplicaţii care foloseşte baze de date se poate proceda în două moduri:

1. Se crează baza de date cu ajutorul unei aplicaţii de tip server de baze de date şi se scriu apoi aplicaţiile care accesează baza de date într-un limbaj care posedă funcţiile necesare accesării serverului (frecvent se folosesc limbajele C#, Java, Visual C++ sau Visual Basic).

2. Se foloseşte o aplicaţie de tip sistem de gestiune de baze de date (S.G.B.D. sau D.B.M.S. - database management system).

[1] Specializarea Logistică Industrială, Facultatea IMST;
E-mail: catalin.negrea95@gmail.com;

3 CREAREA UNEI BAZE DE DATE

Primul pas în crearea unei aplicaţii client pentru gestiunea unei baze de date este crearea bazei de date. Deschidem un nou proiect de tip Windows Application, numit patru. Urmează adăugarea proiectului o bază de date. Pentru aceasta se apasă click dreapta pe rădăcina proiectului în Solution Explorer şi în meniul contextual se alege Add şi New Item…. Va apare dialogbox-ul Add New Item – patru, în care se va selecta SQL Database. În textbox-ul Name: se va alege numele bazei de date care va fi adăugată serverului, fie Persoane.mdf acest nume. Se apasă apoi butonul Add (fig. 1).

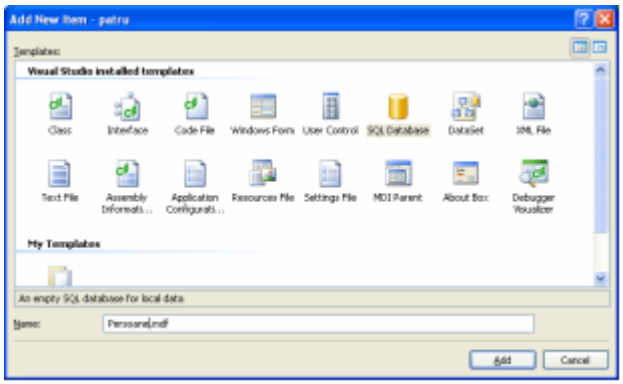

Fig 1.

Va apare dialogbox-ul Data Source Configuration Wizard, care permite configurarea bazei de date şi conectarea ei la aplicaţie. Pentru moment, apăsăm butonul **Cancel**. În partea stângă a ecranului, au apărut mai multe tab-uri, nu doar Toolbox-ul. Unul dintre aceste tab-uri este Database Explorer (fig. 2) Cu ajutorul acestuia, se poate configura baza de date Persoane. De asemenea, în Solution Explorer a apărut o nouă intrare (fig. 3), care reprezintă tocmai baza de date nou creeată. Vom putea naviga înapoi în formă (cod sau interfaţă) în aceeaşi manieră ca şi până acum, sau vom putea reveni la baza de date pentru a realiza configurări

suplimentare. Navigarea între Toolbox şi Database Explorer se poate face cu ajutorul tab-urilor din partea de jos a fig. 2.
În acest moment, baza de date este goală. Pasul următor ar fi construirea şi să adăugarea de tabele. În mod normal, o bază de date este constituită din mai multe tabele, între acestea putând exista diferite relaţii de legătură, motiv pentru care baza de date poartă denumirea de *relaţională*. În acest moment, ne vom limita la o bază de date cu un singur tabel.

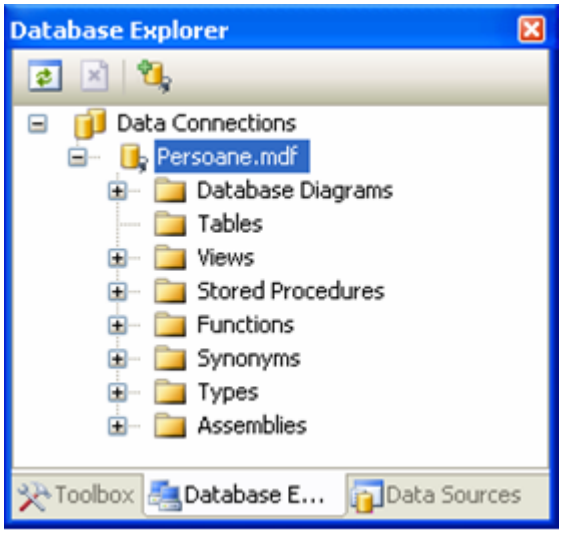

Fig 2

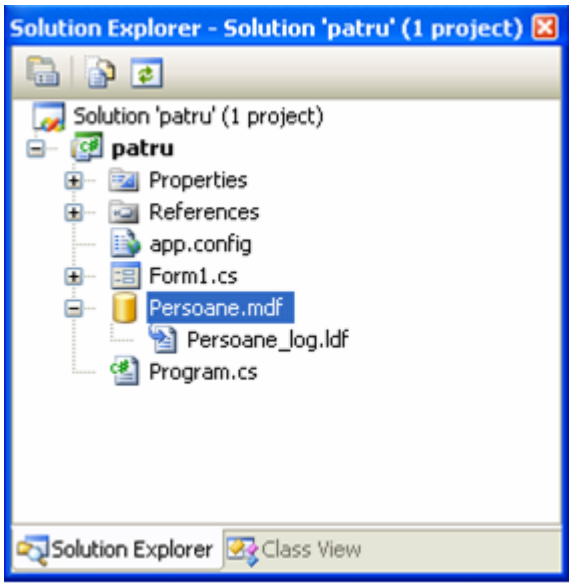

Fig 3

Urmează să adăugăm un tabel şi să definim structura tabelului (structura unui articol). Pentru aceasta, în Database Explorer apăsăm click dreapta pe Tables şi în meniul contextual alegem **Add New Table**. Windows Form Designer ne va permite să configurăm tabelul.

Vom începe prin a defini numele câmpurilor şi tipul acestora. Pentru aceasta, se face click pe prima linie a coloanei Column Name şi se inserează numele primului câmp. Apoi, în coloana Data Type şi în combobox-ul apărut se alege tipul câmpului respectiv. Desselectăm apoi ceckbox-ul Allow Nulls şi continuăm pe următoarea linie, până la definirea completă a structurii. Dacă aş lăsa checkbox-ul Allow Nulls marcat, aş preciza infrastructurii că acel câmp poate fi gol. Există situaţii în care tabelul se construieşte astfel încât să permită existenţa câmpurilor necompletate, însă în exemplul nostru nu vom permite acest lucru.

Trebuie definită cheia primară, adică acel câmp care este unic definit pentru fiecare articol în parte şi pe baza căruia acel articol este identificat. În acest caz, va fi câmpul marca. Este foarte simplu. Click pe linia marca şi apoi apăsăm butonul cu cheiţă din partea stângă sus. În acest moment, în dreptul liniei marca, va apare o cheiţă, ceea ce va specifica faptul că aceasta este cheia primară în tabel. Tot ce mai rămâne de făcut este să salvăm tabelul în baza de date, sub un nume. Fie Angajat acest nume. Vom apăsa butonul dischetă şi vom da tabelului numele dorit (fig. 4). În acest moment, tabelul va fi afişat în baza de date, în Database Explorer (fig. 5).

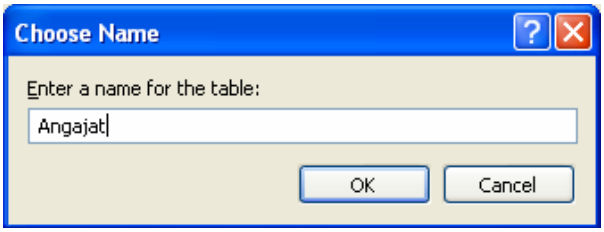

Fig 4

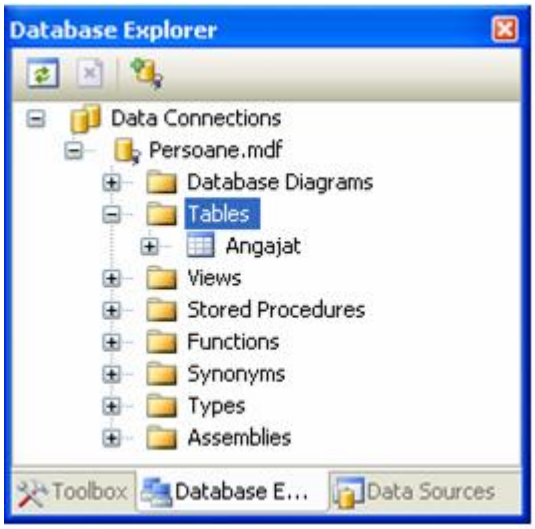

Fig 5

Urmează popularea tabelul cu date iniţiale. În mod normal, acest lucru se face utilizând un program client, dar în această faza e bine să vedem cum se poate face acest lucru şi manual. Pentru aceasta, facem click dreapta pe tabelul Angajat şi în meniul contextual alegem Show Table Data. Va apare un grid care ne permite să completăm date în tabel.

4 CREAREA UNUI PROGRAM CLIENT PENTRU BAZA DE DATE

Un program de interfaţă pentru bazele de date va utiliza un mecanism de *databinding*. Ce înseamnă databinding? Prin acest mecanism, un control de interfaţă (TextBox, ListControl, etc) este conectat direct pe înregistrările unui câmp dintr-un tabel, astfel încât primeşte, afişează şi permite modificarea valorilor acestuia, fără ca programatorul să fie nevoit să implementeze accesul la baza de date în mod explicit. În principiu, fiecare control va implementa un mecanism propriu de conectare la date. Vor exista controale care vor prezenta date dintr-un singur câmp şi controale care vor putea administra date din mai multe câmpuri simultan.
Mecanismul de databinding utilizează tehnologia ADO.NET, care este o dezvoltare a unei tehnologii Microsoft mai vechi, numită ADO (ActiveX Data Objects). Această tehnologie defineşte o serie de clase care permit manipularea datelor dintr-o bază de date, prin intermediul unor obiecte de interfaţă uzuale.

Realizarea mecanismului de databinding se face în mai mulţi paşi:
• ADO.NET începe prin a realiza o conectare la baza de date, utilizând sistemul de gestiune al fişierelor implementat de sistemul de operare, sau o conexiune prin intermediul unui protocol de comunicaţie, dacă serverul de baze de date se găseşte la distanţă.
• ADO.NET gestionează conversaţia aplicaţiei (cerere şi răspuns) cu baza de date.
• ADO.NET gestionează datele primite în urma interogării bazei de date. Prin intermediul acestui mecanism operaţiile ce pot fi efectuate asupra articolelor din baza de date sunt: căutarea unui articol pe baza informaţiei dintr-un câmp, actualizarea unui articol, ştergerea unui articol respectiv adăugarea unui nou articol.
• ADO.NET gestionează mecanismul de legătură dintre obiectele de interfaţă şi informaţiile recepţionate de la baza de date, stocate într-un obiect de tip *dataset*.

Aici apare o întrebare. De ce e nevoie să construim programe de aplicaţie care să lucreze cu baza de date şi nu lăsăm utilizatorul să lucreze direct cu aceasta? Există mai multe motive pentru aceasta.
• În primul rând, utilizatorul final al bazei de date nu este în general un specialist în calculatoare. De aceea, este necesar ca aplicaţia pe care o utilizează să necesite un efort de învăţare cât mai mic din partea acestuia şi să fie cât mai uşor de utilizat, nepermiţând acţiuni ale utilizatorului care pot afecta corectitudinea şi integritatea bazei de date.
• În al doilea rând, în acest mod, accesul la baza de date poate fi restricţionat, din motive de securitate. Utilizatorul final va avea acces doar la acele informaţii disponibile prin intermediul interfeţei şi în plus, accesul la unele informaţii poate fi restricţionat pe bază de parole.
• În al treilea rând, o aplicaţie de interfaţă va putea controla şi ordona modul de prezentare a datelor. Prin acces direct, datele pot fi accesate oricum, fără a exista o logicăp evidentă în modul lor de prezentare. Prin implementarea unui program de interfaţă, utilizatorul final va fi silit să urmeze paşii impuşi de logica programatorului.

• În al patrulea rând, prin acces direct, un utilizator final nespecialist poate strica integritatea datelor. Un program de interfaţă va evita în mare măsură acest lucru şi, în situaţia pierderii integrităţii datelor, va putea lua măsuri, transparent pentru utilizator, de refacere a acesteia.

Să începem să construim aplicaţia de interfaţă. Pentru aceasta, să revenim la formă (click pe Form1.cs în Solution Explorer şi apoi pe butonul care afişează forma). Se adauga acum proiectului o sursă de date. Pentru aceasta selectăm meniu Data şi alegem intrarea de meniu Add New Data Source. Astfel, va fi lansată o aplicaţie numită Data Source Configuration Wizard care ne permit să configurăm sursa de date pentru proiectul nostru.
Se poate observa că implicit, ca sursă de date este selectat Database. Dar, se poate alege şi o altă sursă de date, furnizată de un serviciu web, sau de un alt obiect care permite conectarea la date. Vom lăsa aursa de date implicită şi vom apăsa butonul **Next>**.
La pasul următor, vom selecta sursa de date pentru proiectul nostru. Putem alege o bază de date din cele recunoscute de wizard, prin selctarea ei în controlul ComboBox, sau putem creea o nouă conexiune, prin apăsarea butonului **New Connection…**. De asemnea, în această fază, poate fi vizualizată fraza SQL de conectare la sursa de date, prin deschiderea casetei Connection String. Despre aceste lucruri vom discuta pe larg în cursul de programare avansată. Ăn acest moment vom apăsa doar butonul **Next>**.

Acum urmează să bifăm **Tables** şi să apăsăm **Finish**. Cu aceasta, sursa de date este adăugată proiectului nostru. Să vedem acum care este rezultatul muncii noastre. Putem observa că în Solution Explorer a fost adăugată o intrare, PersoaneDataSet.xsd, care la rândul ei conţine 3 intrări. Extensia .xsd arată faptul că acest fieşier conţine o schemă XML a proiectului .NET. Despre semnificaţia detaliată a acestui fişier vom discuta pe larg în cursul de programare avansată. Ce este demn de reţinut în acest moment, este că baza de date reală este stocată de Persoane.mdf, în timp ce PersoaneDataSet.xmf conţine descrierea internă a unei o copii de lucru a bazei de date, utilizată de proiectul nostru, copie care urmăreşte ca un cache înregistrările bazei de date şi poartă denumirea de ***dataset*** sau ***cursor***. Toate modificările din baza de date se vor oglindi în cursor, respectiv toate modificările din cursor vor fi transmise bazei de date de infrastructura .NET.
Putem vedea într-un format visual structura fişierului .xsd, apăsind dublu click asupra lui. Se poate observa că este prezentată exact structura tabelului Angajat creeat anterior. De fapt, ceea ce vedem, este copia locală a structurii bazei de date Persoane, adică structura cursorului asociat tabelului Angajat.

Să adăugăm formei un TabControl şi să modificăm proprietatea Text a primei pagini în DataGrid. Să apăsăm din nou meniul Data şi să alegem Show Data Sources. Se observă că în partea stângă, în Data Sources, avem o altă reprezentare a cursorului. Dacă vom deschide combobox-ul asociat tabelului Angajat, vom obţine un meniu care ne permite să alegem modul de reprezentare a datelor pe interfaţă. Haideţi să alegem DataGrid şi să tragem tabelul cu mousul în interiorul controlului Tab. Să dimensionăm apoi controalele în aşa fel încât în DataGrid să fie disponibile toate coloanele. Obţinem o interfaţă care ne afişează conţinutul tabelului şi care ne permite să modificăm, adăugăm şi să ştergem înregistrări.

Puteţi naviga în interiorul grid-ului cu butoanele de navigare din partea de sus a formei, puteţi adăuga o nouă înregistrare apăsând butonul +, puteţi modifica conţinutul unei înregistrări prin simpla tastare a noului conţinut în caseta corespunzătoare, sau puteţi şterge o înregistrare prin slectarea ei în coloana din stânga coloanei marca şi apăsarea butonului **x** marcat cu roşu sau puteţi naviga direct la o înregistrare prin tastarea numărului în textboxul dintre butoanele de navigare şi apăsarea tastei Enter. Modificările făcute sunt salvate prin apăsarea butonului cu pictogramă dischetă.

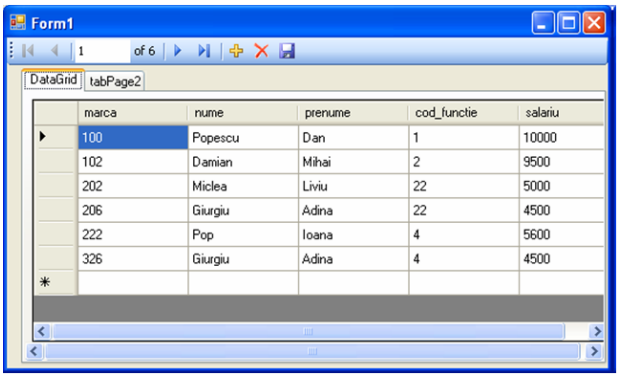

Fig. 6

5 CONCLUZIE

Scopul final al acestui tip de interfețe este acela de a optimiza procesele de producție la nivelul fabricilor prin eliminarea factorului uman. Astfel datele despre cantitatea produsa sau materialele utilizate vor putea fi citite direct de la aparate iar baza de date actualizata in timp real scăzând drastic numărul momentelor în care producția trebuie sa fie oprită.

În concluzie interfațarea bazelor de date folosind aplicații software realizate in C++ oferă posibilitatea creării unui mediu prietenos pentru clienți oferindu-le informațiile necesare fară a compromite siguranța celorlalte date sau integritatea serverului și a bazei de date.

6 BIBLIOGRAFIE

[1]. http://www.infoap.utcluj.ro/ - curs accesarea bazelor de date – Accesat la data: 12.05.2015
[2]. http://philip.greenspun.com/teaching - curs RDBMS - Accesat la data: 12.05.2015
[3]. https://docs.oracle.com/database - Introducere în bazele de date ORACLE
Accesat la data: 12.05.2015