BAZE DE DATE MEDICALE - CONSTRUIREA UNOR INTERFEŢE UTILIZATOR EFICIENTE

P. OLAH[1], M. MARUSTERI[2], MONICA SUCIU[3], H. SUCIU[4], DANIELA DOBRU[5]

[1]Doctorand Universitatea de Medicină şi Farmacie Tg. Mureş, [2,3,4,5]Universitatea de Medicină şi Farmacie Tg. Mureş

Cuvinte cheie: *baze de date medicale, sisteme informatice în domeniul medical, interfaţa cu utilizatorul*

Rezumat: *Sistemele Informatice din domeniul Medical (SIM) se bazează în general pe baze de date mari care stochează date medicale. În afară de problemele specifice legate de proiectarea structurilor de date, aceste sisteme prezintă o provocare pentru dezvoltatori în ceea ce priveşte arhitectura nivelului extern al bazei de date, strâns legată de construirea interfeţei cu utilizatorul. Această lucrare prezintă trei aspecte specifice legate de interfaţa cu utilizatorul a unei baze de date medicale (Proprietatea temporală, Procesul de alocare şi Lista deschisă de atribute) folosind o abordare orientată spre date pentru nivelul extern. Abordarea se bazează mai mult pe vederi, construite pe partea de server şi pe partea de aplicaţie client. Acestea pot fi completate de un număr redus de proceduri stocate pentru a implementa funcţionalitatea completă a sistemului. În modul nostru de abordare, complexitatea acestor procese este în mare parte încorporată în interogări care sunt rulate pe serverul bazei de date. Aceste interogări pot deveni însă complexe şi consumatoare de timp. Două probleme majore apar de aici: în primul rând, complexitatea poate duce la prezentarea inconsistentă a aceloraşi date (avem în vedere sisteme informatice de mari dimensiuni) şi în al al doilea rând, interogări complexe pe volume mari de date pot afecta grav performanţa generală. Aceştia sunt factorii pe care designerii unui astfel de sistem trebuie să îi controleze, în scopul de a ajunge la un echilibru acceptabil. Prin utilizarea tehnicilor prezentate, dezvoltatorii de software pot construi cu uşurinţă interfeţe utilizator care prezintă o funcţionalitate de bază sau mediu-complexă, beneficiind în acelaşi timp şi de avantajele standardizării. Pentru o funcţionalitate mai complexă, sunt necesare modele şi tehnologii avansate, care vor fi prezentate într-o lucrare viitoare.*

Keywords: *medical databases, healthcare information systems, user interface*

Abstract: *Healthcare Information Systems (HIS) generally rely on large databases that store medical data. Apart from the specific issues related to the design of the data structures, these systems present a challenge to the developer regarding the architecture of the external level of the database, closely related to the building of the user interface. This paper discusses three specific issues related to the user interface of a medical database (The Temporal Property, The Allocation Process and The Open Attribute List Specialization) using a data-oriented approach for the external level. The approach is mostly based on data views, constructed on the server side and on the client application side. A small number of stored procedures however is necessary to implement the full functionality of the system. In our approach, most of the complexity of these processes is embedded in the queries that are run against the database. These queries can often be complex and time consuming. Two major issues arise from here: first, complexity can lead to inconsistent presentation of the same data (we are considering large information systems) and second, large queries can severely affect overall performance. These are the factors which the designers of such a system must control in order to reach an acceptable balance. By using the presented techniques, software developers can easily build user interfaces that present basic or medium-complex functionality, while having the benefits of standardization. For highly complex functionality, more advanced patterns and technologies are needed, which will be presented in a future paper.*

INTRODUCERE

Sistemele informatice în domeniul medical (SIM) se bazează în general pe baze de date mari care stochează date medicale. În afară de problemele specifice legate de proiectarea structurilor de date, aceste sisteme prezintă o provocare pentru dezvoltatori în ceea ce priveşte arhitectura de nivel extern al bazei de date, strâns legată de construirea interfeţei cu utilizatorul.

Principalele aspecte pe care le-am abordat pe parcursul dezvoltării acestui SIM sunt următoarele:

- Filtrarea - filtrarea datelor după diferite criterii;
- Ordonarea - sortarea datelor pentru a se potrivi nevoilor utilizatorului;
- Procesul de Sincronizare - filtrarea subseturilor de date în funcţie de o valoare curentă selectată;

[1]Autor corespondent: P. Olah, Str. Ghe. Marinescu, Nr. 50, Cod 540136, Târgu Mureş, România, E-mail: olah_peter@yahoo.com, Tel: +40265 212111

Articol intrat în redacţie în 20.08.2012 şi acceptat spre publicare în 28.09.2012
ACTA MEDICA TRANSILVANICA Decembrie 2012;2(4):12-15

- Proprietatea temporală - gestionarea datelor care au o valabilitate limitată în timp;
- Procesul de alocare - asocierea seturilor de date la o înregistrare dată în baza de date;
- Lista deschisă de atribute - gestionarea datelor care au asociate o listă de caracteristici ce evoluează în timp;
- Prelucrarea structurilor de date complexe medicale - arbori şi arbori multipli;(1)
- Procesul de raportare - extragerea de informaţii relevante pentru utilizator.

Abordarea se bazează mai mult pe vederi de date, construite pe partea de server şi pe partea de aplicaţie client. Cu toate acestea este necesară completarea acestora cu un număr redus de proceduri stocate pentru a implementa funcţionalitatea completă a sistemului.

SCOP

În această lucrare vom discuta despre trei aspecte specifice legate de interfaţa cu utilizatorul a unei baze de date medicale, utilizând o abordare orientată spre date (2) pentru nivelul extern (Proprietatea Temporală, Procesul de Alocare şi Lista Deschisă de Atribute).

MATERIAL ŞI METODĂ DE LUCRU

Tehnicile de dezvoltare de software pe care le propunem în această lucrare sunt derivate dintr-un proiect real, care implică utilizarea datelor medicale ale pacienţilor investigaţi şi trataţi în cadrul unei secţii de gastroenterologie. Cerinţele pentru acest sistem au fost complexe şi au avut o evoluţie în timp pe parcursul proiectării şi dezvoltării software-ului. Prin abordarea noastră ne-am dorit nu doar rezolvarea unor probleme curente, ci dezvoltarea unei soluţii pentru o întreagă categorie de probleme similare. Astfel, folosind concepte teoretice găsite în literatura de specialitate, am identificat problemele menţionate mai sus şi am dezvoltat o serie de soluţii care implică şabloane de proiectare de baze de date, soluţii de arhitectură a aplicaţiilor şi module tip pentru construirea interfeţei utilizator.

Arhitectura sistemului este una de tip client-server. Am folosit un sistem relaţional de gestionare de baze de date (DBMS) pe partea de server. Pentru partea de client am folosit un mediu de dezvoltare de software, care oferă facilităţi de programare orientată spre obiecte, dar care dispune, de asemenea, de propriul motor de baze de date relaţionale, permiţând astfel utilizarea de vederi de date pe partea de client.

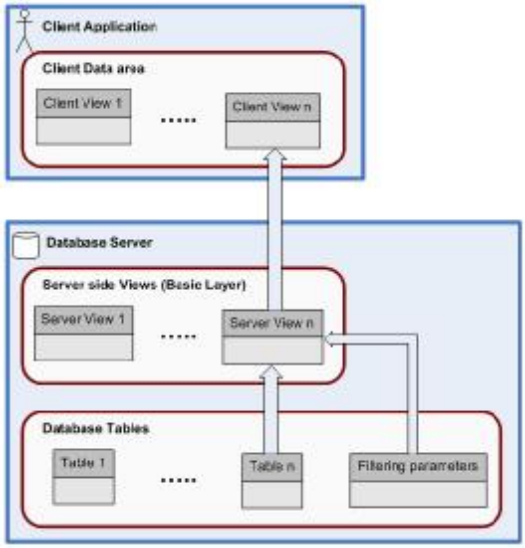
REZULTATE

A. Modelarea Timpului – Proprietatea Temporală

Timpul este o dimensiune omniprezentă care influenţează toate procesele din lumea noastră. Aceasta include şi evoluţia datelor medicale, fie că este vorba de datele clinice sau cele legate de cercetare. Acestea fiind spuse, o cerinţă de bază a unui SIM va fi întotdeauna capacitatea de a înregistra datele cronologice sau evoluţia datelor în timp. Am implementat un astfel de mecanism în baza noastră de date prin utilizarea unei ştampile duble de timp, care include o dată de început şi o dată de sfârşit. Acest lucru ne-a permis să modelăm intervale de timp, mai degrabă decât doar puncte singulare în timp, cum ar fi cazul atunci când se utilizează o ştampilă simplă de timp. Când sunt necesare momente punctuale de timp, intervalul va fi îngustat până la un punct, prin înregistrarea aceleiaşi date şi ore la ambele capete ale acestuia. Această abordare oferă un model complet de timp, dar necesită o putere de procesare mai mare pentru interogarea bazei de date. Fiecare interogare care accesează date temporale trebuie să fie filtrată folosind un interval de timp dat sau predefinit. Pentru a nu supraîncărca interfaţa cu utilizatorul cu controale de filtrare specifice, am implementat un tabel special în baza de date care conţine astfel de intervale de timp pentru fiecare utilizator.

Aşa cum am ilustrat în Figura 1, aceşti parametri de filtrare pentru intervalele de timp sunt apoi utilizati într-un nivel intermediar de vederi, stocat pe server. Programatorii care implementează diferite părţi ale interfeţei vor folosi aceste vederi în interogări în locul tabelelor de bază. În acest fel, în multe cazuri, dezvoltatorii nu mai trebuie să ţină cont de faptul că datele interogate au o componentă temporală. Această abordare s-a dovedit a creşte eficienţa de dezvoltare în mod semnificativ.

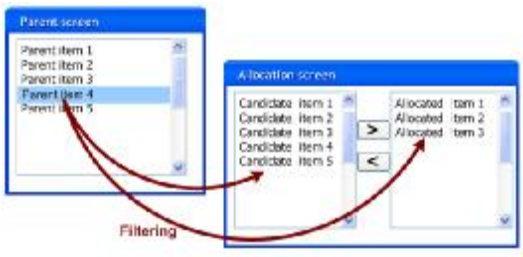
Figura nr. 1. Reprezentarea grafică a utilizării parametrilor de filtrare



B. Modelarea Submulţimilor – Procesul de Alocare

Procese ca atribuirea unui element la un grup sau o categorie sunt adesea regăsite ca o cerinţă la nivel de interfaţă utilizator a SIM. De obicei, utilizatorul va atribui un număr de elemente dintr-un set la un anumit grup sau categorie. La rândul său, grupul sau categoria este reprezentat de obicei de o înregistrare într-un tabel, sau un element într-un al doilea set. Dacă ne uităm la acest proces din punct de vedere al teoriei mulţimilor, trebuie să asociem un număr de elemente ale mulţimii A (o submulţime a lui A) cu un singur element al mulţimii B. Utilizatorul va naviga de obicei prin elementele mulţimii B şi fiecăruia îi va aloca o submulţime de elemente ale lui A. Vom numi mulţimea B mulţimea elementelor „părinte”. O submulţime a lui A, deja alocată unui părinte dat va constitui lista elementelor „alocate” iar restul mulţimii A va reprezenta elemente care pot fi în mod potenţial alocate, sau "elemente candidate".

Figura nr. 2. Reprezentarea grafică a teoriei mulţimilor



În cadrul Interfeţei Grafice Utilizator (IGU), am reprezentat setul părinte folosind un ecran care oferă o

funcţionalitate completă pentru a crea, interoga, actualiza şi respectiv şterge înregistrări. Aceasta conţine un set de date pe care îl numim vedere Părinte. Alocarea va avea loc folosind un al doilea ecran sincronizat cu primul. Aici sunt prezentate cele două submulţimi ale lui A, elementele deja alocate şi elementele candidate, împreună cu butoanele funcţionale pentru alocare/dealocare (Figura 2). Cele două submulţimi ale lui A sunt implementate cu ajutorul a două vederi speciale construite în nivelul extern al bazei de date.

Vederea elementelor alocate

Această vedere selectează înregistrările din tabela A, care sunt deja alocate la înregistrarea curentă din B. Caracteristica cea mai importantă a acestei vederi este faptul că aceasta este filtrată în funcţie de înregistrarea curentă a vederii părinte, folosind o cheie din tabela părinte.

Vederea elementelor candidate

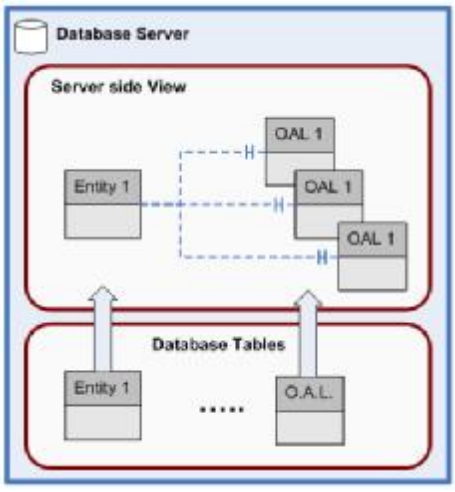
Această vedere trebuie să identifice lista elementelor candidate în conformitate cu înregistrarea curentă din vederea părinte. Va trebui deci să returneze un subset al lui A eliminând elementele deja alocate din întregul set. Implementarea unei astfel de vederi cu ajutorul unei interogări SQL presupune utilizarea unei clauze NOT IN, care poate crea probleme de performanţă. În cazul în care tabela care reprezintă mulţimea A conţine un volum mare de date, mai multe opţiuni de filtrare pot fi implementate. Acestea vor fi accesibile prin intermediul IGU folosind controale special concepute.

C. Lista deschisă de atribute

Una dintre provocările cu care se confruntă proiectantul bazei de date atunci când construieşte un SIM care va fi folosit în cercetarea medicală este faptul că nu toate cerinţele pot fi formulate într-un stadiu incipient. Noi şi noi cerinţe vor fi formulate pe măsură ce sistemul este utilizat şi multe dintre acestea vor viza structura bazei de date. În scopul de a preveni reproiectarea constantă a bazei de date ca urmare a acestui proces, am folosit un şablon de proiectare care modelează extensional unele dintre atributele unei entităţi în loc ca acestea să fie modelate intensional.

Am implementat acest lucru prin intermediul unui tabel special, care stochează atributele ca şi înregistrări, nu ca şi coloane. Acest tabel poate fi apoi legat cu un alt tabel care reprezintă o entitate prin intermediul unei relaţii de tip m la n (care poate stoca şi valorile atributelor dacă este necesar). Am numit acest şablon de proiectare Lista Deschisă de Atribute (LDA).

Figura nr. 3. Reprezentarea grafică a şablonului de proiectare LDA



O problemă specifică pe care am întâlnit-o la acest şablon de proiectare a fost utilizarea pe scară largă a legăturilor de tip „outer-join” atunci când am convertit aceste atribute stocate ca şi rânduri în coloane în cadrul diferitelor vederi (Figura 3). Aceste legături sunt impuse de o convenţie pe care am stabilit-o, prin care fiecare element al unei entităţi poate avea zero sau mai multe atribute înregistrate în LDA. Acest lucru oferă un mare grad de flexibilitate, dar poate avea un impact negativ asupra performanţei.

DISCUŢII

Sistemele informatice din domeniul medical necesită soluţii înalt specializate, din perspectiva Tehnologiei Informaţiei.(3) Cu toate acestea, acest lucru nu exclude beneficiile date de generalizare, ceea ce produce elemente reutilizabile în procesul de dezvoltare. Aceste elemente pot fi şabloane de proiectare de baze de date, soluţii de arhitectură a aplicaţiilor şi module tip pentru construirea interfeţei utilizator. În abordarea prezentată mai sus ne-am concentrat pe primele două din această listă.

Cea mai mare parte a complexităţii acestui sistem este modelată în vederile utilizate pentru interogarea bazei de date. În scopul de a păstra gestionabil setul tot mai mare de vederi, recomandăm structurarea acestora pe nivele. Un nivel de bază, care ar trebui să fie stocat pe serverul de baze de date, precum şi unul sau mai multe nivele "utilizator", care vor fi dezvoltate folosind vederile din nivelul de bază. Nivelul de bază va oferi unele denormalizări des utilizate, precum şi unele funcţionalităţi specifice, de exemplu filtrare folosind intervalele de timp predeterminate. Vederile utilizator vor fi parte a nivelului extern al bazei de date în conformitate cu cele trei niveluri de arhitectură ANSI/SPARC.(4) Acestea vor fi construite folosind nivelul de bază şi/sau tabelele bazei de date.

Această abordare centralizează mentenanţa părţilor sensibile ale bazei de date, îmbunătăţind astfel consistenţa şi eficienţa în dezvoltare. Cu toate acestea, pentru baze de date mari, pot apărea unele probleme de performanţă ca urmare a prelucrării suplimentare necesare pentru interogarea nivelului de bază al vederilor.

CONCLUZII

Prin utilizarea tehnicilor prezentate mai sus, dezvoltatorii de software pot construi cu uşurinţă interfeţe utilizator care oferă o funcţionalitate de bază sau mediu-complexă, beneficiind în acelaşi timp şi de avantajele standardizării. Acest lucru va reduce timpul de dezvoltare şi, mai important, va reduce numărul de erori şi/sau inconsistenţe în sistem. Utilizarea Listei Deschise de Atribute reduce nevoia intervenţiilor ulterioare efectuate de echipa de dezvoltare de software în cazul unor solicitări noi, pentru că acest tip de structură de date permite utilizatorului final să definească şi să configureze atributele noi.

Pentru o funcţionalitate mai complexă, cum ar fi modelarea arborilor sau a arborilor multipli (de exemplu, foile de observaţie ale pacientilor), sunt necesare modele şi tehnologii mai avansate. Soluţii pentru aceste probleme vor fi prezentate într-o lucrare viitoare.

Notă:

*Această lucrare este parţial susţinută de Programul Operaţional Sectorial Dezvoltarea Resurselor Umane (POS DRU), finanţat din Fondul Social European şi de către Guvernul României, în cadrul proiectului cu numărul de contract **POSDRU 64331 şi POSDRU 60782**.*

REFERINŢE

1. Furnas GW, Zacks J. Multitrees: Enriching and Reusing Hierarchical Structure. Proceedings of the ACM CHI 94

Human Factors in Computing Systems Conference; 1994. p. 330-336.
2. Lewis B. Data-Oriented Application Engineering: An Idea Whose Time Has Returned. The Data Administration Newsletter - TDAN.com. January 1; 2007.
3. Muji M, Ciupa RV, et al. Database Design Patterns for Healthcare Information Systems. MEDITECH 2009, IFMBE Proceedings 26, p. 63-66.
4. Date CJ. An Introduction to Database Systems (8th edition). Addison-Wesley; 2004.