УДК 61:004.651

Про програмне середовище проектування інтелектуальних медичних баз даних

В.П. Марценюк, Н.О. Кравець
Кафедра медичної інформатики з біофізикою,
Тернопільська державна медична академія ім. І.Я. Горбачевського, Україна

Резюме

У даній роботі представлені результати створення реальної медичної бази даних і побудованої на її основі експертної системи, що оптимізує процес прийняття медичних рішень. Особливо слід зазначити, що результати діалогу з експертом можуть бути опубліковані на Web-сайті.

Ключові слова: медичні бази даних, інформаційна модель, індекси, зв'язки, представлення, експертна система медико-діагностичного призначення.

**Клін. інформат. і Телемед.
2004. Т.1. №1. с.47—53**

Вступ

Прогресивний розвиток медичної науки і практики став можливим завдяки впровадженню сучасних інформаційних технологій. Комп'ютери значною мірою організують і вдосконалюють роботу лікаря, сприяють систематизації медичної інформації і розробці нових технологій у діагностичному та лікувальному процесах, організації діяльності медичних установ. Міжнародні комп'ютерні мережі, система комп'ютерних бібліотек дозволяють лікарю користуватися досягненнями світової медичної науки.

Задача обробки великих обсягів інформації вже давно є однією з найболючіших проблем організації системи охорони здоров'я в Україні. І на допомогу лікареві приходять найрізноманітніші системи управління базами даних та експертні системи.

Матеріали та методи досліджень

Під базою даних мають на увазі деяку уніфіковану сукупність даних, що спільно використовуються персоналом/населенням групи, підприємства, регіону, країни, світу. Задача бази даних полягає у збереженні всіх даних, що представляють інтерес в одному або кількох місцях, причому, таким способом, що наперед виключає непотрібну надлишковість. У добре спроектованій базі даних надлишковість даних виключається та ймовірність зберігання суперечливих даних мінімізується.

Експертна система — це система, здатна сама приймати рішення і давати чіткі вказівки користувачеві. Експертні системи допомагають прийняти рішення лікареві в ситуаціях, які вимагають значного розумового напруження і обмежені в часі. Найчастіше це — постановка діагнозу хвороби, диференціальна діагностика, вибір тактики лікування. Вагоме місце експертні системи займають в діяльності середнього медичного персоналу, коли рішення необхідно приймати в екстремальних умовах при відсутності лікаря. Крім цього, експертні системи допомагають оптимізувати організацію системи охорони здоров'я. Вони будуються для розв'язання конкретних задач за певними попередньо перевіреними правилами.

Сьогоднішній день охорони здоров'я України пов'язується із появою ідей щодо електронного медичного паспорта — проекту, що в кінці призведе до появи розподіленої бази даних, що зберігатиме усю медичну інформацію України. Як зазначається в [1], етап проектування бази даних — найвідповідальніший: від того, якої структури будуть розроблені таблиці, від встановлених між ними зв'язків залежить подальше ефективне використання бази даних і спектр задач, які з її допомогою можна буде розв'язувати. В даній роботі запропоновано один підхід до розв'язування за допомогою медичних баз даних задач підтримки медичних рішень. Даний проект може бути розвинутий до впровадження у реальні клініки чи науково-дослідні медичні заклади.

Результати дослідження та їх обговорення

Питання проектування бази даних

Розробка інформаційної моделі. Медична діагностика – наука неточна. Якщо пацієнт вкаже на певний набір симптомів того чи іншого захворювання, то такий взаємозв'язок не завжди абсолютний. Саме тому визначальним моментом експертних систем в медичній діагностиці є використання ймовірнісного підходу. При цьому, знання, набуті попередніми поколіннями лікарів-експертів про кожне захворювання, можуть бути представлені рис. 1. Тут:

Захворювання – назва захворювання;

p – апріорна ймовірність захворювання (тобто ймовірність захворювання у випадку відсутності додаткової інформації);

Симптом_i – назва симптому (його можна сформулювати як запитання до пацієнта, на яке можна отримати відповідь «Так» або «Ні», наприклад – «Чи є у Вас підвищена температура ?»);

py – імовірність отримати відповідь на запитання для симптому пацієнта «Так» , коли дане захворювання має місце;

pn – імовірність отримати відповідь на запитання для симптому пацієнта «Так», коли дане захворювання не має місця.

Інформаційна модель медичної діагностики повинна містити в своєму складі два об'єкти: захворювання та симптом, між якими панують зв'язки найскладнішого характеру (рис. 2). Так, наприклад, симптом – сильний кашель – може означати, що у пацієнта бронхіт, туберкульоз або просто сильний кашель. Такий тип зв'язку між множинами об'єктів носить умовну назву «багато-до-багатьох».

На сьогоднішній день такий зв'язок не реалізовано безпосередньо в СУБД для персональних комп'ютерів, а для його моделювання створюють спеціальну проміжкову таблицю, в якій зберігають інформацію про симптоми кожного захворювання, як зображено на рисунку 3.

При цьому, таблиця СИМПТОМИ_ЗАХВОРЮВАННЯ містить такі поля-атрибути:

- ***ІД_Захворювання*** – зовнішнє посилання на визначальний код захворювання;

Рис. 1. Інформація про захворювання, якою володіє експертна система медико-діагностичного призначення.

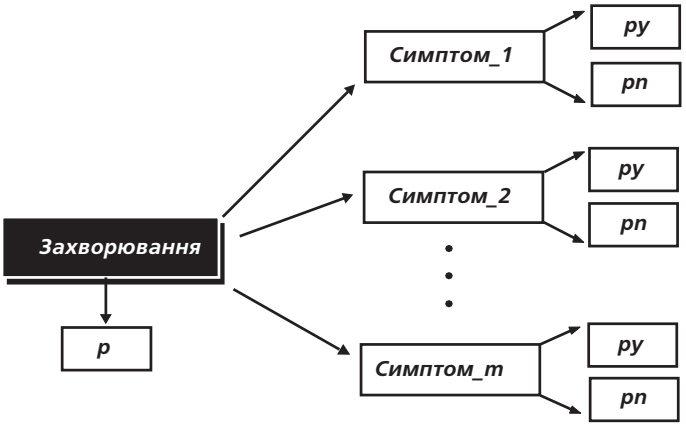


Рис. 2. Між об'єктами «захворювання» та «симптом» існує зв'язок «багато-до-багатьох».

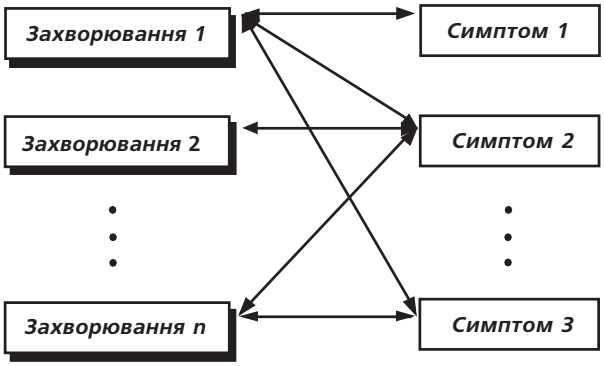


Рис. 3. Зв'язок «багато-до-багатьох» реалізується створенням проміжкової таблиці.

• ***ІД_Симптом*** – зовнішнє посилання на визначальний код симптому;
• ***ру*** – імовірність отримати відповідь на запитання для симптому «Так», коли дане захворювання має місце;
• ***рп*** – імовірність отримати відповідь на запитання для симптому «Так», коли дане захворювання не має місця.

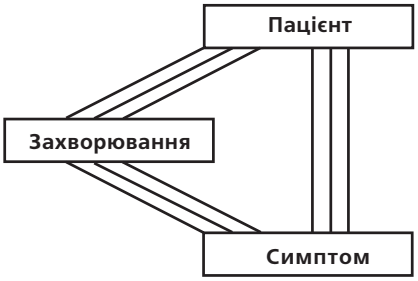
Створення структур даних для збереження медико-діагностичної інформації: таблиці.

При роботі в СУБД таблиці є одним з головних об'єктів, на їх основі здійснюється побудова усіх інших елементів: для баз даних – це локальні та зовнішні представлення, запити; для всього проекту – це форми та звіти. У таблицях збираються дані з конкретної теми, наприклад, уся відома інформація про захворювання. Кожен блок даних такої таблиці включає інформацію про певне захворювання. Інформація ця може бути неоднорідною, і тому блок складається із кількох полів різного типу, що містять назву захворювання, його опис, фотокартки, посилання на таблиці, де зберігаються історії хвороби пацієнтів і т. ін.

База даних може складатися із кількох таблиць, в кожній з яких зберігається однотипна інформація. У першій таблиці може зберігатися інформація про захворювання Diseases, у другій – про симптоми Symptom, в третій – про симптоми Disease_Symptoms, притаманні окремому захворюванню, в четвертій – про пацієнтів та їх лікування Patient. Об'єкти інформаційної моделі медичної діагностики та зв'язки між ними показані на рисунку 4.

Добре продумана структура і формат записів бази даних є неодмінною передумовою ефективної роботи з нею, оскільки, саме в розумно побудованій базі можна отримати швидкий доступ до всієї необхідної інформації. Перед тим, як приступити до розробки великої бази даних з багатьма таблицями, її потрібно ретельно спланувати, щоб в подальшому не довелося займатися великими змінами структури. Ця вимога виражає загальновідому закономірність, що полягає в тому, що в процесі розробки довільного виробу найважливіші та найцінніші рішення приймаються на самому початку проектування.

Рис. 4. Об'єкти інформаційної моделі медичної діагностики пов'язані зв'язками «багато-до-багатьох».



В таблицях може зберігатися інформація найрізноманітнішого вигляду (символьного, числового, текстового).

Індекси, зв'язки, локальні представлення.

Для формування зв'язків необхідно в таблицях створити індекси.

Індекс можна розглядати як результат деяких операцій над полями таблиці з метою отримання ідентифікуючих значень для кожного запису. Тобто, призначення або ж походження індексів можна розглядати з наступної точки зору. Таблиці, що моделюють об'єкти реального світу, як правило, містять поля, між якими існують складні функціональні зв'язки. Тому рідко вдається ідентифікувати кожен запис таблиці (тобто, окремий об'єкт із усієї множини об'єктів), користуючись одним полем (атрибутом). Як правило, ідентифікуюче значення для об'єкту можна отримати лише в результаті композиції кількох атрибутів. Отже, індекси задають правила знаходження ідентифікуючого значення записів у таблицях.

Після проектування основи бази даних потрібно перейти до встановлення зв'язків між таблицями, що здійснюється за допомогою механізму індексування.

Представлення (View) можна розглядати як своєрідний буфер між даними та додатком споживача. Річ у тім, що розміщення полів у таблицях не завжди відповідає тому способу сприйняття про об'єкти інформаційної моделі, який притаманний споживачу. Як правило, споживача не дуже цікавлять ідентифікаційні позначення та ключі, що були введені лише з технічних міркувань і які є невід'ємними атрибутами кожної таблиці. Натомість людині зручніше працювати з такими описовими атрибутами, як Назва_хвороби та Визначальне_питання_симптому. Практично усі наступні компоненти СУБД спрямовані на розв'язування цієї проблеми. Представлення серед них є першим.

Проілюструємо можливості представлень задачею, яка стосується одночасного заповнення таблиць DISEASES, SYMPTOMS та DISEASE_SYMPTOMS. Для створення такого представлення потрібно скористатися підпрограмою Конструктор. Вона дозволяє відкрити вікно вибору таблиць або іншого представлення в якості джерел даних. При потребі, можна використати інші джерела даних. Для того, щоб включити таблиці DISEASES, SYMPTOMS та DISEASE_SYMPTOMS у процес створення представлення, вони почергово помічаються та натискається кнопка Add. Графічні зображення таблиць повинні з'явитися у вікні View Designer. Далі, у вкладці Fields конструктора представлень, за допомогою кнопки Add переноситься групу полів із списку Available Fields у список Selected Fields. У вкладці Order by поле diseases з таблиці Diseases переноситься у список Ordering criteria. Після збереження і запуску на виконання з'являється вікно, яке дозволяє переглянути одночасно потрібні дані із трьох таблиць. Деякі з полів у представленні можна редагувати.

Проте, у представленні немає можливості змінювати значення умовних імовірностей *ру* та *рп* із вхідної таблиці. Механізм модифікації табличних значень побудований так, що змінюваний запис розшукується в таблиці (а насправді в пам'яті комп'ютера) через унікальний код запису, який є одним із полів таблиці, яке не допускає дублювання своїх значень. На жаль, використання індексу тут не допомагає. Розглянемо проблему розробки таких полів.

Взагалі, існує два способи створення унікальних кодів записів. Обидва вони грунтуються на використанні поля, якому присвоюється значення, що отримується, за припущенням від деякої збереженої процедури.

Збережені процедури (Stored Procedures) є невід'ємною частиною бази даних. Тут зберігаються тексти підпрограм-функцій і підпрограм-процедур, що можуть використовуватися під час роботи з базою даних.

Перший спосіб грунтується на автоматичному збільшенні ключа на одиничку та на використанні спеціальної таблиці у складі бази даних. Така таблиця (назвемо її Sys_key) складається із двох полів: поле table_name містить назву таблиці бази даних, поле key_val містить значення ключа останнього створюваного запису. При цьому, наступне значення ключа обчислюється за функцією – збереженою процедурою NextKey.

Другий спосіб. Дуже цікаве вирішення задачі запропоноване програмістом Кевіном О'Коннером із штату Вашингтон. В якості значення для ключа нового запису пропонується брати значення, що повертає функція – збережена процедура, яка в якості своїх аргументів фактично використовує поточну дату і час.

```
FUNCTION CasualKey
RETURN SYS(1) +
PADL(ALLTRIM(STR(SECONDS()*1000)),8,'0')
```

Для проектування такого ключа використовується той же підхід, що й у першому способі, лише слід врахувати, що значення ключового поля

тепер належить символьному типу Character, під який слід виділити 15 символів. Даний підхід має ще одну суттєву перевагу. А саме, можна розробити функції, обернені до CasualKey, які за значенням ключового поля повертатимуть дату і час. В такий спосіб можна, не виділяючи додаткової пам'яті, довідатись про дату і час створення кожного запису таблиці.

Існують представлення, які потрібні споживачу регулярно при незмінних критеріях відбору. Наприклад, представлення, що дозволяють вибрати із вказаної бази медичних знань назви усіх захворювань. Але часто використовуються представлення, які є незначно видозміненими варіантами одного разу підготовленого базового представлення. Мова може йти, наприклад, про запит, який дозволяє вибрати з бази даних симптоми певного захворювання, причому, назва самого захворювання задається окремо, в діалозі. Таке представлення (параметричне) змінюється від випадку до випадку, але незначно. Для їх реалізації проектується параметричне представлення, у якому вказується той критерій (критерії), який може змінюватися на замовлення споживача.

Життєвою є задача, коли для вказаного захворювання необхідно отримати список усіх відомих його симптомів із вказуванням їх умовних імовірностей.

Для відбору даних з таблиць поряд з представленнями існує механізм запитів. З'ясуємо сутність цих понять.

Запит – це прохання на одержання інформації з бази даних, сформоване на основі використання критеріїв для записів, що вилучаються.

Представлення – це означення віртуальної таблиці, спроектованої за вибором споживача. Воно може бути локальним, віддаленим або параметричним. Представлення посилаються на одну або більше таблиць або інші представлення. Їх можна модифіковувати і вони можуть посилатися на віддалені таблиці. З першого погляду може здатися, що мова йде про одну і ту ж річ. Насправді це не так. Отже, існують такі головні відмінності.

1. Відмінність щодо фізичного розміщення на диску. Як сказано в означенні, представлення – це віртуальна таблиця. Під цим мається на увазі, що представлення не є звичайною таблицею у повному розумінні цього слова. Насправді, інформація про представлення зберігається у файлі бази даних (з розширенням .DBC) і не існує окремого файлу, пов'язаного із представленням. Для запиту все навпаки. Інформація про структуру запитів зберігається в окремих файлах із розширенням .QPR.

2. Підпорядкованість до баз даних. Представлення (локальні або ж віддалені) є компонентами баз даних і не існують поза ними. Запити – це незалежні функціональні одиниці і з цього боку своєю поведінкою нагадують вільні таблиці.

3. Можливість впливу на дані, розміщені у вхідних таблицях. На відміну від представлень, через які можна змінювати дані у полях базових таблиць, запити такої можливості не мають. Вони працюють лише в одному напрямку, передаючи дані від базових таблиць до додатку споживача.

4. Джерела вхідних даних. Представлення грунтуються на даних, що можуть поступати з таблиць або ж інших представлень. Запити також будуються на основі полів таблиць і представлень. І в той же час, слід підкреслити, що запити не можуть будуватися на інших запитах.

В силу вищенаведених міркувань, складно надати перевагу використанню запитів або ж представлень. Спосіб отримання вибірки даних повинен визначатися специфікою задачі та характером даних, що отримуються.

Щодо інструментів побудови запитів, то вони мало чим відрізняються від представлень. Замість підпрограми View Designer використовується Query Designer, яка має дуже подібний інтерфейс.

Існує два способи введення (і попередньої обробки) інформації в СУБД: через таблиці (представлення) і за допомогою форм. Вибрати той чи інший спосіб можна, грунтуючись на такому простому правилі.

Якщо дані у таблиці змінюються рідко або в неї рідко додаються нові записи, то для введення, змін та індикації даних потрібно використовувати таблицю. Крім того, режим таблиці рекомендується використовувати тоді, коли потрібно отримати найповніший огляд даних.

Але, якщо дані часто змінюються або база постійно поповнюється новими записами, потрібно користуватися формою, оскільки в режимі форми можна сконцентрувати увагу на даних, що відносяться до певного запису, наприклад, до певного захворювання.

Довільна форма будується на основі таблиці або представлення. Уся інформація форми міститься в керуючих елементах (полях) цієї форми, деякі з них безпосередньо пов'язані з полями базової таблиці. У таких елементах можна показати вміст відповідних полів таблиці і внести у них зміни. Інші елементи форми служать для оформлення, наприклад, надпису, дозволяють позначити ті чи інші об'єкти у формі, лінії і прямокутники, щоб сконструювати форму і позначити групу даних.

Найлегше спроектувати екранну форму Visual FoxPro, користуючись підпрограмою – Майстер автоформи (Autoform Wizard). Майстер самостійно створить усі елементи керування для відображення даних, панель інструментів для мандрування базою даних (тут розміщені кнопки для переміщення записами, створення нових записів, знищення існуючих записів, переходу в режим редагування запису), заголовок-мітку екранної форми, що співпадає з назвою таблиці або представлення. Найсуттєвішим недоліком, що обмежує використання таких форм є неможливість використання даних з різних таблиць або представлень.

Суттєвим недоліком автоформи є те, що її не можна зберігати окремим файлом, а необхідно щоразу здійснювати виклик.

Найпрактичнішим способом розробки форм є спосіб, коли користувач може вмішуватися у процес створення форми від самого початку. Для додавання елементів керування потрібна панель інструментів Form Controls. Кожна піктограма цієї панелі інструментів є стилізованим зображенням елементу керування, який можна вмонтовувати в форму.

На рисунку 5 зображено блок-схему реальної бази даних медичного призначення.

Експертні системи медико-діагностичного призначення байєсівського виводу

Основні положення байєсівської системи логічного виводу.

Запропонована в подальшому експертна система грунтується на алгоритмі, що одержав в застосуванні до експертних систем назву байєсівська система логічного виводу. Ось його суть.

1. Програма визначає кількість доступних захворювань та симптомів.

2. Програма ініціалізує апріорні імовірності P(H). Вона також виробляє деякі значення масиву правил RULEVALUE. Для кожного питання обчислюється RULEVALUE(I) = RULEVALUE(I) + ABS(P(H:E) - P(H:неE)), що відповідає значенню можливих змін імовірностей усіх хвороб, до яких вони відносяться. Це робиться для того, щоб визначити, які питання (симптоми) є найважливішими і з'ясувати, про що запитувати в першу чергу.

3. Програма знаходить найважливіше запитання і задає його. Існує ряд варіантів, що робити з відповіддю: ви можете просто сказати: «Так» або «Ні». Можете спробувати сказати: «Не знаю», – змін

при цьому не відбудеться. Набагато складніше використати шкалу від -5 до +5, щоб виразити степінь впевненості у відповіді.

4. Апріорні імовірності заміняються новими значеннями при одержанні нових підтверджуючих доказів.

5. Підраховуються нові значення правил. Визначаються також мінімальне і максимальне значення для кожного захворювання, що грунтуються на існуючих на даний час апріорних імовірностях і припущеннях, що докази, які залишилися, будуть на користь гіпотези або суперечити їй. Важливо з'ясувати: чи варто дану гіпотезу продовжувати розглядати, чи ні ? Гіпотези, які не мають сенсу, просто відкидаються. Ті ж з них, чиї мінімальні значення вищі певного рівня, можуть вважатися можливими наслідками. Після цього програма повертається до третього кроку і продовжує свою роботу.

Рис. 5. Блок-схема реальної бази даних медичного призначення.

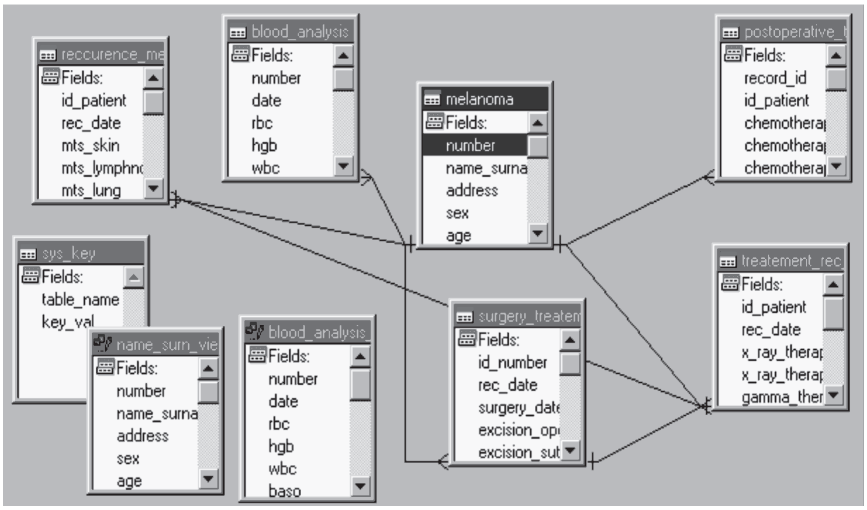
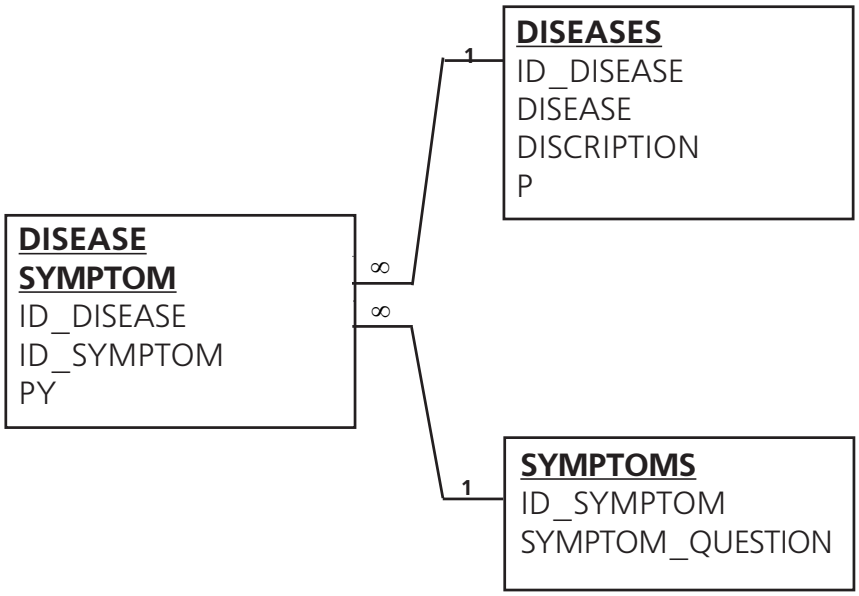


Рис. 6. Структура таблиць, що використовуються у базі знань для компоненту TExpertDialog.



Компонент TExpertDialog

Даний компонент дозволяє набути навичок у конструюванні експертних систем медико-діагностичного призначення. Експертні системи такого роду грунтуються на байєсівській системі логічного виводу. Крім того, використовується методика «гнучких» відповідей. Тобто, відповідаючи на запитання «Чи була останнім часом простуда, чи подібне інфекційне захворювання?», вибираємо значення з імовірнісного інтервалу [0,1]. Результати діалогу з експертом зберігаються в автоматично створеному HTML-документі, тобто можуть бути опубліковані на WWW.

База даних для збереження даних про захворювання та їх симптоми складається з трьох таблиць, як показано на рисунку 6 .

Декларація компоненту має вигляд

```
TQuestionForm = class(TForm)
{declaration is ommited here}
end;
TExpertDialog = class(TSpeedButton)
private
  { Private declarations }
  FDataLinkGT: TDataLink;
  FDataLinkHT: TDataLink;
  FDataLinkET: TDataLink;
  FGeneralTable: TDataSource;
  FHypothesesTable: TDataSource;
  FEvidencesTable: TDataSource;
  function GetGeneralTable: DataSource;
  function GetHypothesesTable:
  TDataSource;
  function GetEvidencesTable:
  TDataSource;
  procedure SetGeneralTable
  (Value: TDataSource);
  procedure SetHypothesesTable
  (Value: TDataSource);
  procedure SetEvidencesTable
  (Value: TDataSource);
protected
  { Protected declarations }
  property DataLinkGT:
  TDataLink read FDataLinkGT;
  property DataLinkHT:
  TDataLink read FDataLinkHT;
  property DataLinkET:
  TDataLink read FDataLinkET;
public
  { Public declarations }
  constructor
  Create(AOwner:TComponent);
  override;
  destructor Destroy; override;
  procedure Click; override;
published
  { Published declarations }
  property  GeneralTable: TDataSource
  read GetGeneralTable
  write SetGeneralTable;
  property HypothesesTable:
  TDataSource
  read GetHypothesesTable
  write SetHypothesesTable;
  property EvidencesTable: TDataSource
  read GetEvidencesTable
  write SetEvidencesTable;
end;
```

Тут TQuestionForm – це компонент модальної форми для діалогу з експертною системою; найважливішими властивостями компоненту TExpertDialog є GeneralTable (таблиця для збереження правил для заключень експертної системи), HypothesesTable (таблиця для збереження імовірних діагнозів), EvidencesTable (таблиця для збереження даних про симптоми).

Найпростіший додаток – експертна система з використанням TExpertDialog може бути розроблена в результаті наступних дій.

Необхідне програмне забезпечення:

1. Інструментальна система Delphi із встановленим компонентом TExpertDialog.
2. Демонстраційна база даних медико-діагностичного призначення, встановлена в системі під псевдонімом MKBPardx.

Необхідні дії:

1. Створіть каталог для збереження проекту.
2. Запустіть Delphi і створіть проект, вибравши з головного меню File – New Application
3. Виберіть на сторінці Data Access, Палітри Компонент, компонент Table і розмістіть три його примірники у вікні форми. За припущенням вони отримають імена Table1, Table2, Table3.
4. Виберіть на сторінці Data Access, Палітри Компонент, компонент DataSource і розмістіть три його примірники у вікні форми. За припущенням вони отримають імена DataSource1, DataSource2, DataSource3.
5. Виберіть на сторінці MedInfTraining, Палітри Компонент, компонент ExpertDialog і розмістіть його у вікні форми.
6. Встановіть властивості у такі значення:

для компоненту Table1
DatabaseName MKBPardx
TableName diseases
Active True
для компоненту Table2
DatabaseName MKBPardx
TableName symptoms
Active True
для компоненту Table3
DatabaseName MKBPardx
TableName disease_symptom
Active True
для компоненту DataSource1
DataSet Table1
для компоненту DataSource2
DataSet Table2
для компоненту DataSource3
DataSet Table3
для компоненту ExpertDialog1
HypothesesTable DataSet1
EvidencesTable DataSet2
GeneralTable DataSet3

7. Збережіть проект у каталозі, створеному на кроці 1.

Рис. 7. Головне вікно однієї із розроблених експертних систем.

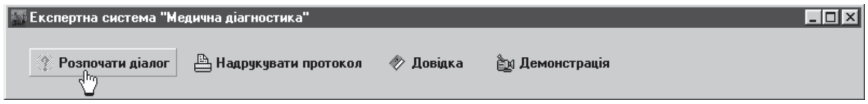


Рис. 8. Вікно запитання розробленої експертної системи.

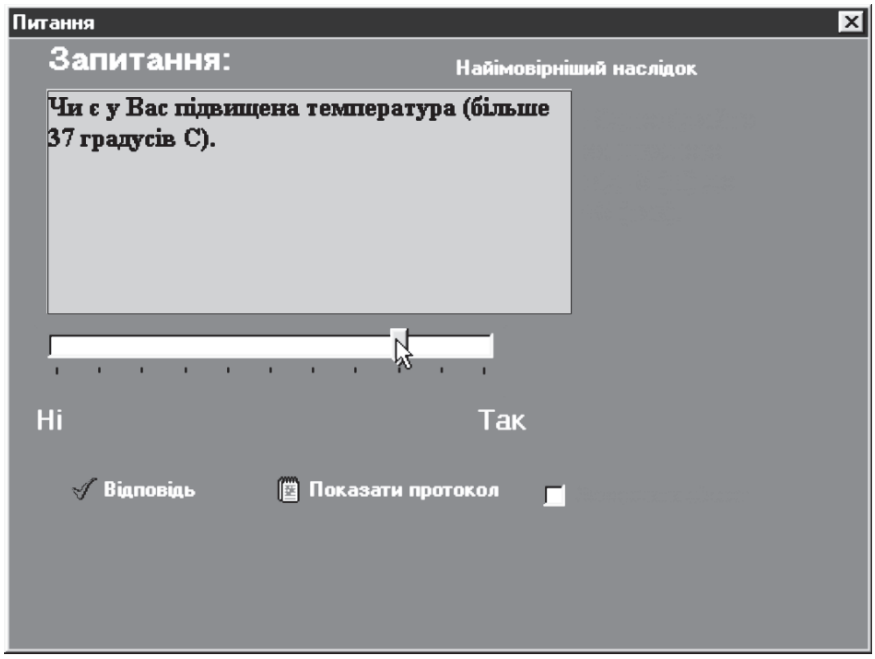
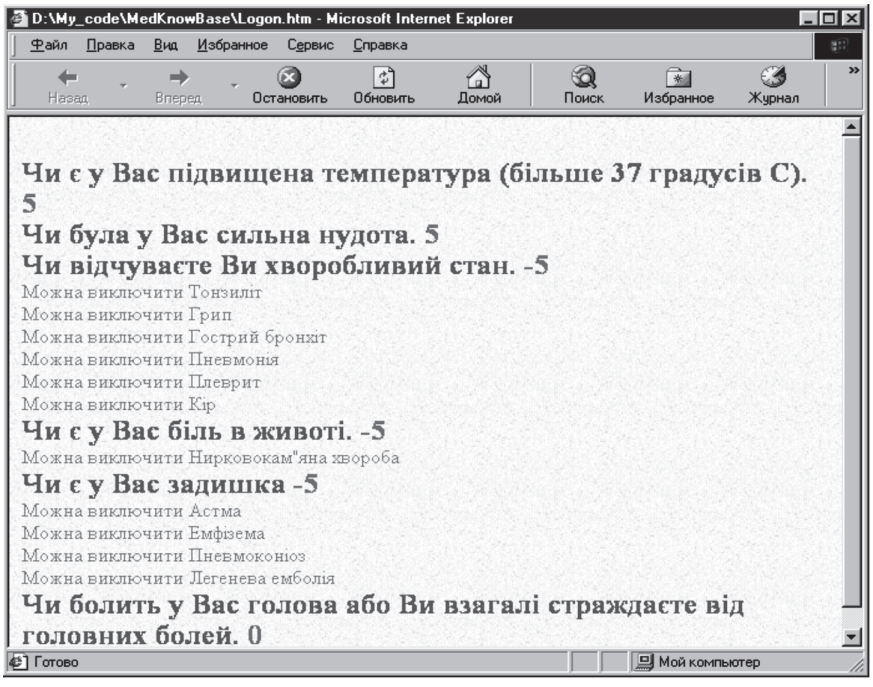


Рис. 9. Протокол зберігає діалог з експертною системою. На кольоровому рисунку протокол виглядає так: червоний колір позначає запитання, синій — відповіді, зелений — поради експерта.



8. Відкомпілюйте і запустіть проект. Клацаючи кнопку ExpertDialog, ви розпочнете діалог з експертною системою.

На рисунках 7, 8, 9 можна побачити додаток, побудований за допомогою TExpertDialog. Зауважте, що він містить можливість голосового опитування. Для цього запитання у звуковій формі зберігаються у базі даних.

Висновки

Таким чином, накопичені знання та досвід дозволяють розробляти експертну систему на основі логічної схеми байєсівського виводу, користуючись базою даних медико-діагностичного призначення або на основі генетичного алгоритму та Internet-програмування. Застосування експертних систем такого типу відкриває широкі можливості для оптимізації роботи лікаря та молодшого медичного персоналу і дозволяє швидко приймати ефективні рішення.

Література

1. В.П. Марценюк. Медична інформатика. Проектування та використання баз даних. — Тернопіль.: «Укрмедкнига», 2001. — 177с.
2. В.П. Марценюк, Н.О. Кравець. Медична інформатика. Методи системного аналізу. — Тернопіль.: «Укрмедкнига», 2002. — 175с.
3. О.Ю. Майоров, В.М. Пономаренко, М.І. Хвісюк, В.В. Кальниш. Інформаційні технології в охороні здоров'я // Медична освіта. №2, 2002. — с.60 — 68.

About the software environment of intellectual databases designing

V.P. Marceniuk, N.O. Kravetz
Faculty of medical informatics with biophysics,
Ternopol state medical academy named I.Ya. Gorbachevsky ,
Ukraine

Abstract
In this work there are presented results of real medical database creation and building expert system based on it. Note, that results of dialog with expert are able to be published through Web-site.
Keywords: medical database, information model, indexes, link, representation, expert system of diagnostic assignment.

О программной среде проектирования интеллектуальных медицинских баз данных

В.П. Марценюк, Н.О. Кравец
Кафедра медицинской информатики с биофизикой,
Тернопольская государственная медицинская академия им. И.Я. Горбачевского, Украина

Резюме
В данной работе представлены результаты создания реальной медицинской базы данных и построенной на ее основе экспертной системы, которая оптимизирует процесс принятия медицинских решений. Особо следует отметить, что результаты диалога с экспертом могут быть опубликованы на Web-сайте.

Ключевые слова: медицинские базы данных, информационная модель, индексы, связи, представления, экспертная система медицинско-диагностического назначения.

Переписка
к.т.н. **В.П. Марценюк**
Кафедра медицинской информатики с биофизикой,
Тернопольская государственная медицинская академия им. И.Я. Горбачевского,
площадь Свободы, 1
Тернополь, 46001, Украина
e-mail: marceniuk@yahoo.com
тел.: (0352) 431168