УДК 004.4’232

С. Кулик, В. Овсяк

Українська академія друкарства

МОДЕЛЬ ГРАФІЧНОЇ ЧАСТИНИ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ВІЗУАЛЬНОГО СТВОРЮВАННЯ ОПИСУ ОБЄКТІВ РЕЛЯЦІЙНОЇ МОДЕЛІ

Описано модель графічної частини інформаційної технології візуального створювання опису об’єктів реляційної моделі бази даних алгеброю алгоритмів.

We describe a model of information technology, graphic visual description of the creation of objects the relational database model by algebra of algorithms.

Вступ і формулювання задачі

Самим поширеним стандартом опису реляційних баз даних є стандарти *SQL*-92, *SQL*-99 та мова запитів *SQL(Structured Query Language)* – структурована мова запитів [5]. На основі цього стандарту такими гігантами галузі інформаційних технологій як Microsoft, Oracle, IBM та іншими компаніями були розроблені та впроваджені у програмних продуктах SQL діалекти під назвами Transact-SQL(Microsoft), SQL/DS(IBM), PL/SQL(Oracle), MySQL(Sun Microsystems) та інші. Найбільша різниця між діалектами SQL полягає у процедурних розширеннях мови, тобто стосується можливостей написання процедур, які подібні методам у мовах програмування та зберігаються у файлах баз даних безпосередньо на SQL сервері [7].

Засновники реляційної моделі даних Едгар Кодд і Крістофер Дейт та їх прихильники вказують на те, що SQL не є істинно реляційною мовою, тобто строго не відповідає реляційній моделі [6, 8]. Зокрема, вони вказують на такі недоліки SQL [9]: повторювані рядки, невизначені значення (nulls), явне задання порядку колонок зліва направо, колонки без назв і дубльовані назви колонок, відсутність підтримки властивості «=», використання вказівників, висока надмірність. Крістофером Дейтом запропонований альтернативний опис реляційної моделі мов Tutorial-D та Industrial-D. «D» – набір вимог, запропонованих Крістофером Дейтом і Х'ю Дарвеном у їхній книзі під назвою «Основи майбутніх систем баз даних. Третій маніфест», що висуваються до мов, які є мовами запитів для істинно реляційних СУБД [22]. Tutorial-D є абстрактною навчальною мовою без реалізації. Реальне втілення мови D, програмна реалізація якої припускає практичне використання, називається Industrial-D.

Окремо треба відзначити як недолік часткову відповідність маніпуляційної частини діалектів мови SQL реляційній алгебрі. SQL команди перетворюються ядром СУБД на реляційні операції за власними алгоритмами СУБД, сутність яких означена розробниками системи і користувачам не є доступною. Алгоритми цих перетворень не завжди є оптимальними та досліджувати їхні операції, корегувати або будь-яким іншим чином втручатись у їхнє функціонування не є можливим. Отже загальновідомі засоби опису як SQL діалекти та мова Industrial-D не є формалізованими і навіть не є мовами програмування.

Суттєвими недоліками створення систем методом текстово-символьного опису

є значна трудомісткість та високі вимоги до кваліфікації користувачів. Названі недоліки обумовили впровадження майже усіма провідними постачальниками СУБД світового ринку інтегрованих середовищ розробки та засобів автоматизованого створення SQL коду і безпосередньо об'єктів моделей даних. Клас комп'ютерних систем «Інтегровані середовища розробки» загальновідомі у світі як *IDE – Integrated Development Environment*, а засоби автоматизації розробки моделей та програмного забезпечення мають загальну назву *CASE – Computer–Aided Software Engineering.*

Найбільш відомі програмні продукти представлені фірмами *Oracle, IBM і Microsoft* які контролюють 90% ринку СУБД.

Microsoft пропонує інструменти *ServerExplorer* у складі *VisualStudio 2010* та *Microsoft SQL Server Management Studio*, де поряд із засобами адміністрування є візуальні засоби створення відношень. Відношення у термінології майже усіх СУБД називаються таблицями. Окремо треба відзначити Access – СУБД навчального рівня, яка широко використається в навчальному процесі вищої та середньої освіти. Візуальні засоби створення моделі даних у цій системі найбільш зручні для нефахівців у галузі баз даних та програмування. Прямого доступу до SQL опису моделі даних Access не надає. Діалект Jet SQL, на якому ґрунтується Access, надано користувачеві тільки для побудови запитів.

Система *Oracle SQL Developer Data Modeler* підтримує весь діапазон фізичних описів для баз даних, зокрема логічні розділи, ролі і табличні простори, з урахуванням конкретних версій баз даних в системах з різними СУБД від різних виробників [20]. Система є універсальним, повністю автономним інструментом з підтримкою логічного, реляційного, багатовимірного моделювання і моделювання типів даних. Можливість конструювання моделей даних на різних рівнях забезпечує формування вичерпних концептуальних блок-схем зв'язків між сутностями ERD (Entity Relationship Diagram) і перетворення їх у робочі реляційні моделі даних. Користувачі можуть створювати, розширювати і модифікувати моделі даних, а також порівнювати свої моделі з існуючими. Розділення реляційної і фізичної моделі дає можливість розробникам на базі єдиної реляційної моделі створювати різні фізичні моделі даних для різних баз даних або цільових серверів СУБД, включаючи *Oracle Database, IBM DB2, Microsoft SQL Server* та інші.

Система *Rational Rose Data Modeler* фірми IBM забеспечує розробникам баз даних візуалізацію процесів звертання прикладних систем до баз даних, що надає можливість виявляти і вирішувати проблеми до початку впровадження і використання системи. Забезпечує створення моделей об'єктів, даних і сховищ даних та уможливлює зіставляння логічних і фізичних моделей, гнучко відображаючи в проекті бази даних логіку прикладної системи [21]. Надає можливість прямого і зворотного проектування з синхронізацією моделей даних, моделей об'єктів і файлів мовою опису даних або СУБД.

Аналіз інтерфейсів вище згаданих систем показав що переважують такі візуальні елементи як палітри та діаграми, дерева властивостей та таблиці. У системах фірми *Microsoft* переважають дерева властивостей і таблиці, *IBM* –палітри та діаграми, *Oracle* – табличні описи, палітри та діаграми.

Результатом функціонування усіх наведених систем є модель даних, представлена:

- SQL описом, який у програмах даного класу прийнято *називати DDL – Data Definition Language* (мова опису даних);

·візуальною моделлю окремих діаграм;

·файлом проекту (Project) або рішення (Solution) у внутрішньому форматі комп'ютерної системи.

Разом із Data Control Language (DCL) – мовою контролю даних та Data Manipulation Language (DML) – мовою маніпуляції даних DDL описано у ANSI SQL-92, SQL-99 та усіх наступних стандартах [7].

Результати роботи систем с точки зору доступності згенерованого SQL (DDL) коду є різними. Системи фірм *Oracle* та *IBM* дають доступ до згенерованого SQL коду, системи *Microsoft* створюють об'єкти безпосередньо на сервері, а опису не повертають, але передбачені окремі команди генерації SQL опису для окремих частин баз даних.

Маніпуляція об'єктами та параметрами моделі реалізується тільки неформальними методами – інтерфейсними елементами та програмами – майстрами. Функціональне призначення SQL опису є зберігання результату та експортування моделі до SQL сервера.

Крім згаданих систем, особливе місце займають продукти сторонніх компаній, які створюють візуальні моделі даних для більшості сучасних СУБД. Зокрема системи *Visual Architect* копанії *Visual Paradigm* [23], *Data Architect* копанії «TheKompany» [18], *CA ERwin Data Modeler* компанії *CA technologies* [17], *DBDesigner* 4 компанії *fabFORCE* [19].

Аналіз показав що вони генерують SQL опис використовуючи візуальні засоби представлення об'єктів реляційної моделі як палітр, діаграм, таблиць, дерев та використовують об'єктно – орієнтовані методи керування їхніми властивостями. Використання опису для оптимізації не є можливим, наочність перетворень опису моделі під час роботи з візуальними засобами відсутня. Універсальність систем зумовлює перевантаженість інтерфейсу користувача великою кількістю елементів та налаштувань. Можливості та результати роботи систем *Visual Architect, Data Architect, CA ERwin Data Modeler*, *DBDesigner* аналогічні системі *Oracle SQL Developer Data Modeler* і мають такі самі недоліки.

Недоліками існуючих систем, які треба усунути у новій інформаційній технології є відсутність формалізованого опису моделі даних, неможливість використання опис для оптимізації моделі, відсутність можливості автоматично оптимізувати опис моделі за кількістю операцій, неможливість безпосередньо використовувати операції реляційної алгебри, неможливість використовувати будь - які власні математичні методи для дослідження моделі даних.

Для формалізованого опису завдання на генерування реляційних баз даних створено граматику мови на основі алгебри алгоритмів [1, 2, 11, 12, 13].

Опис поданий у вигляді формул набуває такі нові можливості, які мають сучасні математичні методи дослідження та оптимізації систем. Це надає такому опису безумовні переваги перед будь якими нематематичними методами опису реляційної моделі. А саме, застосування математичного апарату дає можливості виконувати оптимізацію формул за кількістю унітермів, дослідження коректності побудованої моделі, в автоматизованому режимі виконувати декомпозицію на підсистеми засобами комп'ютерних систем [3, 15, 16]. Практично створювати формалізований опис можливо за допомогою комп'ютерної системи, яка буде його реалізацією та засобом верифікації.

Створення математичної моделі інтерфейсу користувача комп'ютерної системи аналітичного опису реляційної моделі бази даних є науковою задачею даної роботи.

Принцип побудови графічної підсистеми інформаційної технології

Модель інформаційної технології описується графічною та функціональною частинами.

Аналіз досвіду використання реляційної моделі даних та реалізуючих цю модель комп'ютерних систем провідними виробниками СУБД показує що для ефективного використання мови опису, успішного впровадження інформаційної технологій необхідною умовою є створення комп'ютерної системи візуального синтезу формул моделей.

Найбільш зрозумілою та універсальною формою представлення та впорядкування інформаційних структур баз даних є таблиці.

Концепція автоматизованої комп'ютерної системи синтезу формул полягає на використанні таблиць у якості базового елементу інтерфейсу користувача. Базовими об'єктами які утворюють структуру бази даних є заголовок відношення, домени та тіло відношення[10].

Заголовок відношення утворюється атрибутами, які мають певні властивості. Кожен атрибут представлений рядком таблиці, а його властивості впорядковано записуються у комірки таблиці.

Зміст відношення утворене секвентними послідовностями компонентів відношення, які є парами утвореними назвами атрибутів та їхніми значеннями [11]. Секвентні послідовності компонентів відношення у загально прийнятій термінології Едгара Кодда називають кортежами [9], тому в роботі використана ця назва в інтерфейсі системи. Заголовок таблиці містить назви атрибутів, а комірки – значення.

Домени мають шість властивостей, назви яких розміщено у заголовку таблиці, а значення у комірках[13].

Кількість рядків та стовпчиків не є постійною, тому їх потрібно динамічно формувати у методах класу вікна. Для того, у нижній частині діалогового вікна необхідно передбачити елементи керування розмірами таблиць та кількістю стовпців та рядків.

Попередній перегляд формули під час візуального проектування відношень, доменів, секвентних послідовностей компонентів прискорює освоєння користувачем інформаційної технології та граматики мови опису завдань на генерування баз даних. В розробленій моделі у функціональній частині розроблені засоби попереднього перегляду формул на основі спливаючої підказки (відома властивість графічних елементів ToolTip).

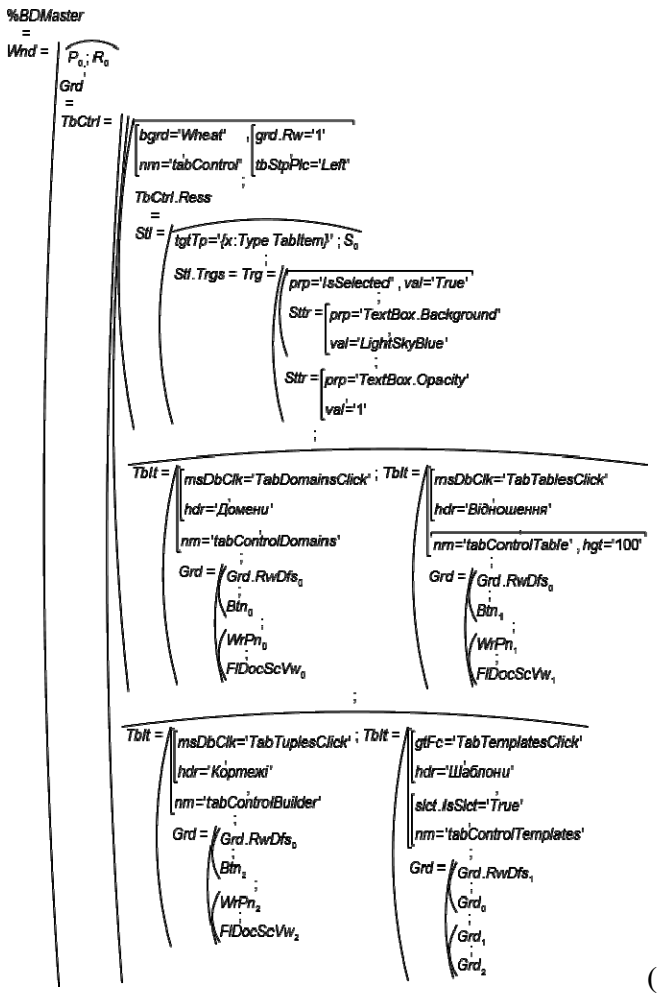
Модель дизайну вікна має вміщувати чотири області: описи доменів, відношень, кортежів і бібліотеки шаблонів. Розміщення усіх чотирьох частин, три з яких схожі одна на одну в одному вікні приводить до перевантаження уваги користувача. Ефективним рішенням цієї проблеми є застосування вкладок «Домени», «Відношення», «Кортежі», «Шаблони», які в один момент часу показують тільки одну частину, при мінімальному використанні площі вікна.

Для ефективного проектування систем є доцільним мати бібліотеку шаблонів проектування, яка зберігає генеровані формули у локальному сховищі формул алгоритмів, виконує маніпулювання збереженими шаблонами, а також забезпечує попередній перегляд формул до вставлення їх у редактор. Бібліотека має вміщувати сховище вбудованих шаблонів різної складності, та сховище шаблонів, які створюються користувачем. Здійснення перемикання між сховищами має вимагати від користувача мінімальну кількість дій. Також є необхідним забезпечити

користувача зручними, зрозумілими і швидкими у засвоєнні та використанні засобами інтерфейсу до сховища, з альтернативою виконання першорядних дій не тільки з панелей вікна, а і за допомогою клавіатурних комбінацій, контекстного меню миші, подвійних цоків мишкою.

Загальна модель опису графічного вікна

Загальна модель описується формулою (1).



(1)

У ній *%BDMaster* - назва графічної підсистеми; *Wnd* – кореневий елемент опису, який має бути тільки один і реалізується відомим класом *Window* [4, 14], та надає можливість створювати, налаштовувати, показувати звичайні і діалогові вікна, а також керувати часом їх існування; *Grd* – графічний елемент, яким описує розбиття графічного вікна на стовпці і рядки, реалізується відомим класом *Grd* [4, 14]; *TbCtrl* – графічний елемент типу контейнера, призначений для розміщення на ньому інших елементів і реалізується відомим класом *TabControl* [4, 14]; *bgrd* – властивість, якою відомим об'єктам (*TabControl*, *Button*, *TextBox*, *WrapPanel*, *local*:*AACanvas*, *Grid*, *Label*, *DockPanel*) [4, 14] задається колір фону, реалізується відомим елементом *Background* [4, 14]; *nm* – типовий унітерм задання ідентифікаційних назв об'єктам (*Window*, *TabControl*, *TabItem*, *Button*, *TextBox*,

FlowDocumentScrollViewer, *Table*, *ColumnDefinition*, *WrapPanel*, *ListBox*, *Grid*, *RowDefinition*, *RadioButton*, *CheckBox*, *Label*, *ComboBox*), реалізується відомою властивістю *Name* [4, 14]; *tbStpPlc* – типовий унітерм повернення або задання способу вирівнювання заголовків вкладок щодо вмісту вкладок, реалізується відомою властивістю *TabStripPlacement* [4, 14]; *Stl* – унітерм задання режиму сумісного використання властивостей, ресурсів і обробників подій екземплярами типу, реалізується відомим класом *Style* [4, 14]; *tgtTp* – унітерм для повернення або задання типу, відносно якого призначений даний стиль, реалізується відомою властивістю *TargetType* [4, 14]; *Stl.Trgs* – унітерм який вказує на відому колекцію стилів Style.Triggers, властивості яких керуються тригерами [4, 14];; *Trg* – унітерм застосування значення властивостей або виконання дій залежно від умов, реалізується відомим класом *Trigger* [4, 14]; *prp* – унітерм повертання або задання властивості, якою повертається значення, що порівнюється з властивістю *Value* тригера, а порівняння є перевіркою рівності посилань, реалізується відомою властивістю *Property* [4, 14]; *val* - значення елементу пари *KeyValuePair* колекції–словника, реалізується відомою властивістю *Value* [4, 14]; *Sttr* – типовий унітерм задання значення властивості, реалізується відомим класом *Setter* [4, 14]; *TbIt* – унітерм вибору елементу, який знаходиться у середині елементу керування *TabControl*, реалізується відомим класом *TabItem* [4, 14]; *msDbClk* – типовий унітерм події, яка відбувається при натисненні кнопки маніпулятора миші два або більше разів і відбувається на відомих графічних елементах *TabItem*, *TextBox*, *ListBox* та реалізується відомою подією *MouseDoubleClick* [4, 14]; *hdr* – типовий унітерм повертання або задання даних відомому об'єкту *TabItem*, використовуваний для заголовка кожного елементу керування і реалізується відомою властивістю *Header* [4, 14]; *slct.lsSlct* – унітерм встановлюючий відому вбудовану властивість Selector.IsSelected, що дана вкладка є активною по замоченню після запуску вікна [4, 14]; *RwDf* - визначає властивості конкретного рядка, які застосовуються до відомих елементів *System.Windows.Controls.Grid* і реалізується відомим класом *RowDefinition*; *hgt* – унітерм задання висоти елементу і реалізується відомою властивістю *Height* [4, 14]; *gtFc* – типовий унітерм-подія, який відбувається при отриманні фокусу елементом і реалізується відомою подією *GotFocus* [4, 14]; P_0 – формула параметрів унітерму *Wnd*; R_0 – формула ресурсів унітерму *Wnd*; S_0 – формула опису стилю унітермів типу *TabItem* [4, 14]; $RwDfs_0$, $RwDfs_1$ – формули графічних елементів, які є типу відомого класу *RowDefinitions* [4, 14] і отримують колекцію RowDefinitionCollection, визначену для даного екземпляра елемента керування Grid; Btn_0, Btn_1, Btn_2 – формули графічних елементів, які реалізуються відомим класом Button [4, 14] та є елементом керування "кнопка", що реагує на подію Click; $WrPn_0$, $WrPn_1$, $WrPn_2$ – формули графічних елементів, які реалізуються відомим класом WrapPanel, що забезпечує засоби для розміщення дочірніх елементів послідовно зліва направо або зверху вниз, залежно від значення властивості Orientation; Bs_0 – секвенція трьох формул, унітерми яких реалізуються елементами відомого класу *Button*; $FlDocScVw_0$, $FlDocScVw_1$, $FlDocScVw_2$ – формули опису перегляду вмісту нефіксованого формату в режимі безперервної прокрутки, які реалізуються відомим класлом *FlowDocumentScrollViewer* [4, 14]; Grd_0, Grd_1, Grd_2 – формули опису графічних елементів розбиття графічного вікна на рядки і колонки, які реалізуються відомим класом *Grid* [4, 14]; *True*, *False* – логічні значення Істинно, Хибно; *Wheat*, *LightSkyBlue* – назви кольорів із відомої системної колекції *Colors* [4, 14]; *tabControl* – назва елементу *TabControl*, який

розміщує чотири вкладки *TabItem* з основними компонентами вікна, які мають назви: *tabControlDomains, tabControlTuples, tabControlTables, tabControlTemlates.* За цими назвами до них звертаються та виконують над ними дії у функціональній частині моделі.

Формула опису параметрів унітерму *Wnd* (P_0) описується виразом (2). У ній *x:Class* – опис параметрів компіляції розмітки XAML для об'єднання класів розмітки і класів з виділеним кодом, що розділяються. Клас коду, що розділяється, визначається в окремому файлі коду на мові Common Language Specification (CLS). Клас розмітки, що розділяється, зазвичай створюється при створенні коду під час компіляції *XAML*; *xmlns* – атрибут, який задає простір назв XAML за замовчуванням об'єкту *Window*, яке містить визначення типів для резервних типів, на які посилатиметься розмітка як на елементи.

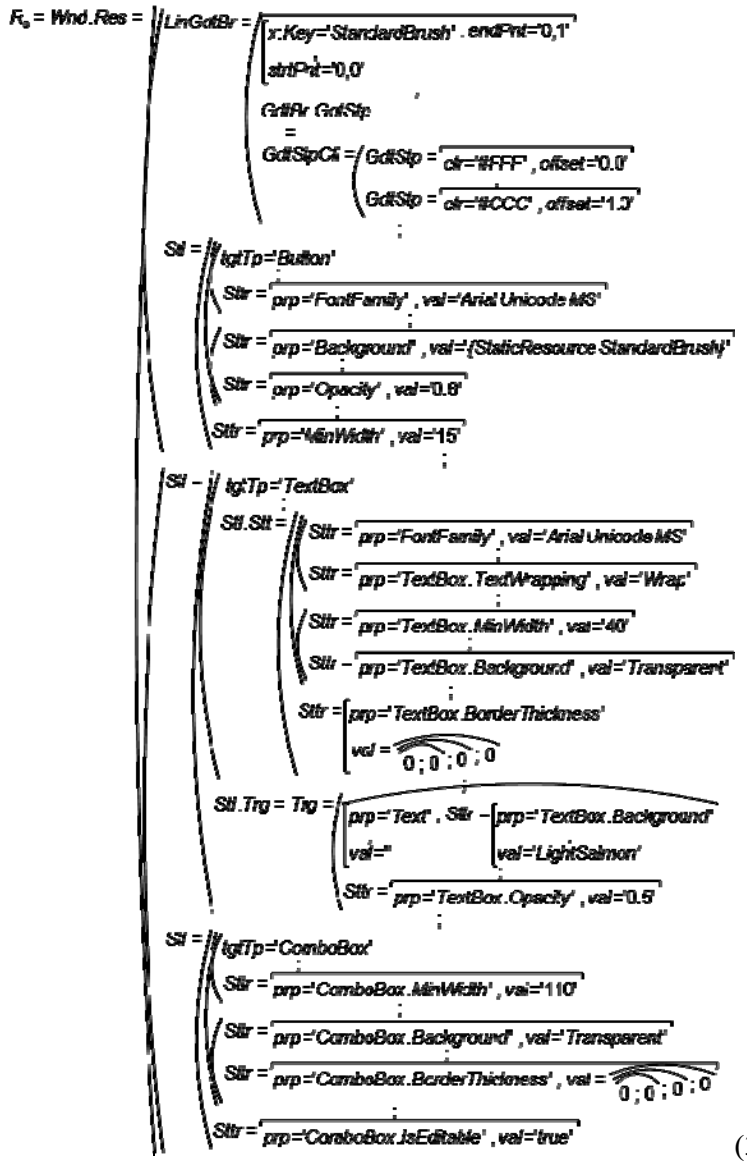
$P_0 =$
```
[ x:Class='TomEdit.BDMaster'
  xmlns='http://schemas.microsoft.com/winfx/2006/xaml/presentation'
  xmlns:x='http://schemas.microsoft.com/winfx/2006/xaml'
  xmlns:local='clr-namespace:TomEdit
  [ ttl='Майстер БазДаних. Візуальний створювач.', [ hgt='380'
    mnHgt='186'                                      wdt='736'
  [ nm='Master'                                    , [ ico='pack://application:,,,/Images/BDMI.ico'
    kbKDwn='Master_KeyDown'                          clsd='Master_Closed'
    szCngd='Master_SizeChanged'
```
(2)

Простір назв XAML за замовчуванням використовує відомий системний простір назв WPF *http://schemas.microsoft.com/winfx/*$_{2006}$*/xaml/presentation*; *xmlns*:*local* – атрибут доступу з XAML розмітки до елементів визначених у довільному просторі назв проекту; *xmlns*:*x* – атрибут кореневого елементу *xmlns*:*x*, який вказує обробникові XAML додатковий простір назв *XAML*, що містить визначення типів для резервних типів, на які посилатиметься розмітка як на елементи. Співставлено простору назв мови XAML *http://schemas.microsoft.com/winfx/*2006/*xaml*; *ttl* – типовий унітерм задання заголовку графічного вікна, реалізується відомою властивістю *Title* [4, 14]; *mnHgt* – унітерм що повертає або задає мінімальне обмеження висоти елементу, який реалізується відомою властивістю *MinHeight* [4, 14]; *wdt* – унітерм задання ширини елементу, реалізується відомою властивістю *Width* [4, 14]; *ico* – типовий унітерм повернення або задання піктограми, реалізується відомою властивістю *Icon* [4, 14]; *clsd* – унітерм-подія на елементі *Wnd*, який реалізується відомою подією *MinHeight* [4, 14]; *szCngd* – типовий унітерм-подія на відомих елементах *Window*, *TextBox*, яка відбувається при зміні значення будь-якої з відомих властивостей *ActualHeight* або *ActualWidth* даного елементу [4, 14].

Формула ресурсів унітерму *Wnd* (R_0) описується виразом (3), де

LinGdtBr – типова підсистема яка інкапсулює відомий об'єкт *Brush* з градієнтом і реалізується відомим класом *LinearGradientBrush* [4, 14];

strtPnt – типовий унітерм-властивість відомого об'єкту *LinearGradientBrush*, яка отримує або задає початкові двомірні координати лінійного градієнта і реалізується відомою властивістю *StartPoint* [4,14]; *endPnt* – типовий унітерм-властивість об'єкту *LinearGradientBrush*, що отримує або задає кінцеві двомірні координати лінійного градієнта і реалізується відомою властивістю *EndPoint* [4,14];

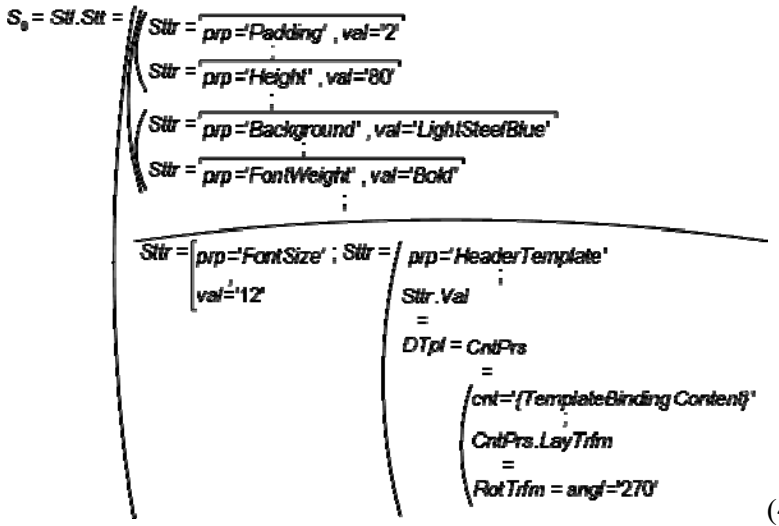


(3)

<u>GdtStpCll</u> – унітерм-властивість колекції відомих об'єктів *GradientStop*, доступ до яких можна дістати індивідуально за індексом, унітерм реалізується відомою властивістю *GradientStopCollection* [4, 14]; <u>GdtStp</u> – типова підсистема, яка описує місцеположення і колір точок переходу в градієнті, реалізується відомим класом *GradientStop* [4, 14]; <u>*clr*</u> – типовий унітерм-властивість відомого об'єкту *GradientStop* для отримання або задання кольору обмежувача градієнта, яка реалізується відомою властивістю *Color* [4, 14]; <u>*offset*</u> – типовий унітерм-властивість об'єкту *GradientStop*, яка призначена для отримання місцеположення обмежувача градієнта у векторі градієнта і реалізується відомою властивістю *Offset* [4, 14].

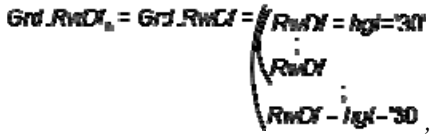
Формула опису стилю унітермів (S_0) подана виразом (4), де <u>*DTpl*</u> – типова підсистема опису візуальної структури об'єкту даних, яка реалізується відомою підсистемою *DataTemplate* [4, 14]; <u>*CntPrs*</u> – типова підсистема відображення вмісту елементу керування *ContentControl*, яка реалізується відомою підсистемою

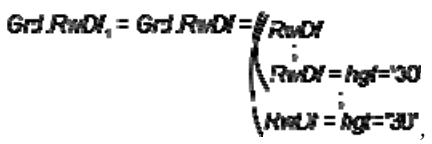
ContentPresenter [4, 14]; *cnt* – типовий унітерм-властивість об'єкту *ContentPresenter* для отримання або задання вмістимого об'єкту *ContentControl*, яка реалізується відомою властивістю *Content* (успадкованою від *ContentControl*) [4, 14]; *RotTrfm* – типовий унітерм-властивість для поворотів об'єктів за годинниковою стрілкою щодо заданої точки в двомірній системі координат x-y, яка реалізується відомою властивістю *RotateTransform* [4, 14]; *angl* – типовий унітерм-властивість об'єкту *RotateTransform*, для отримання або задання кута повороту за годинниковою стрілкою в градусах, яка реалізується відомою властивістю *Angle* [4, 14].



(4)

Формули графічних елементів $\underline{RwDfs}_0$ і $\underline{RwDfs}_1$ описуються такими виразами



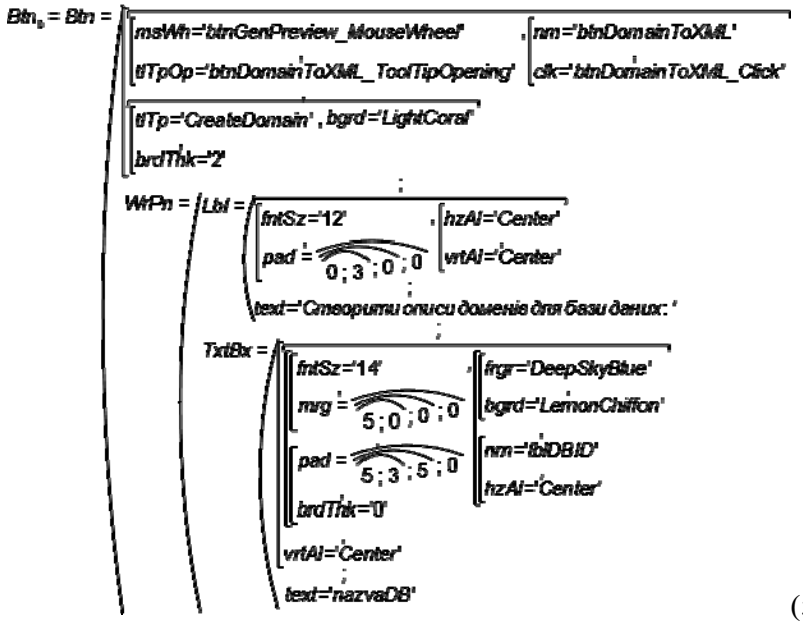


де, *RwDf*– типова підсистема визначення властивості конкретного рядка, який застосовуються до елементів *System.Windows.Controls.Grid* та реалізується відомим класом *RowDefinition* [4, 14].

$\underline{Btn}_0$ наведено формулою (5), де *msWh* – типовий унітерм-подія відомих об'єктів Button, TextBox, яка відбувається у випадку руху мишкою, якщо елемент керування має фокус і реалізується відомою подією *MouseWheel* (успадковано від Control) [4, 14]; *tlTpOp* – типовий унітерм-подія об'єктів Button, Label, відбувається при відкритті якої-небудь підказки в елементі, який реалізується відомою подією ToolTipOpening (успадковано від FrameworkContentElement) [4, 14];

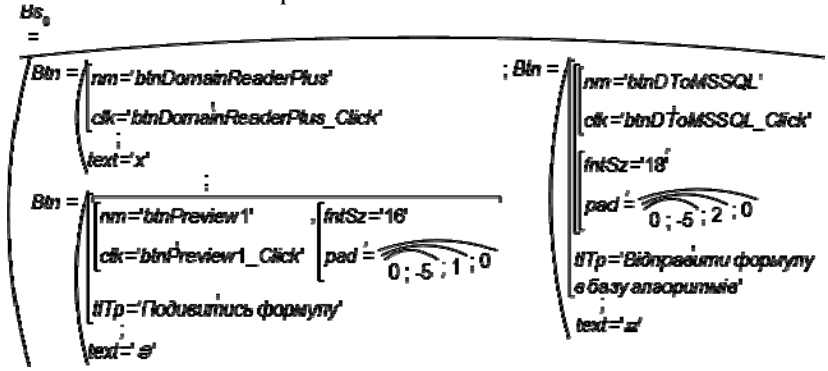
clk – типова подія Click відомого об'єкту Button, яка відбувається при натисненні кнопки, розміщеної на відомому контейнері MenuItem [4, 14]; *tlTp* – підсистема типових об'єктів Button, TextBox, FlowDocumentScrollViewer, Label і TextBlock, представляє елемент керування, який створює спливаюче вікно, що відображає відомості для елементу в інтерфейсі і реалізується відомим класом *ToolTip* [4, 14]; *brdThk* – типова властивість *BorderThickness* відомих об'єктів Button і TextBox, яка повертає або задає ширину межі елементу керування (успадковано від Control) [4, 14]; *WrPn* – підсистема-контейнер для розміщення дочірних елементів послідовно зверху вниз або справа наліво, залежно від значення властивості Orientation, підсистема реалізується відомим класом *WrapPanel* [4, 14]; *Lbl* – типова підсистема для створення текстової мітки і забезпечує підтримку клавіш доступу та реалізується відомим класом *Label* [4, 14]; *fntSz* – типова властивість об'єктів *Label, TextBox, Button, FlowDocumentScrollViewer, ListBox, TextBlock* для отримання або задання розміру шрифта, яка реалізується відомою властивістю *FontSize* (успадковану від *Control*) [4, 14]; *pad*– типова властивість відомих об'єктів *Label, TextBox* і *Button* для повернення або задання заповнення елементів керування, яка реалізується відомою властивістю *Padding* (успадковану від *Control*) [4, 14]; *hzAl* – типова відома підсистема об'єктів *Label, TextBox* і *TextBlock*, якою вказується де на горизонтальній осі відобрається елемент відносно виділеного розділу макету батьківського елементу та реалізується відомим класом *HorizontalAlignment* [4, 14]; *vrtAl* – типова відома підсистема об'єктів Label і TextBox для вертикального позиціонування або розтягування в розділі структури батьківського елементу дочірного елементу, описується відомим класом *VerticalAlignment* [4, 14]; *TxtBx* – типова підсистема, яка забезпечує спрощене керування відображенням невеликої кількості потоку даних, реалізується відомим класом *TextBox* [4, 14]; *mrg* – типова відома властивість об'єктів *TextBox, WrapPanel, Button, FlowDocumentScrollViewer, local:AACanvas* і *TextBlock*, яка задає зовнішні поля розміщення елемента та реалізується відомою властивістю *Margin* (успадковано від *FrameworkElement*) [4, 14]; *frgr* – типова відома властивість об'єкту *TextBox* для задання пензля та описує основний колір, реалізується відомою властивістю *Foreground* (*успадковано* від *Control*) [4, 14]; формула Btn_0 описує елемент який запускає процес створення формул через подію *btnDomainToXML_Click*, та у себі містить текстове поле *lblDBID* назви бази даних, яке використовуються при побудові формули.

Вирази $WrPn_0$ і $FlDocScVw_0$ формули (1) описуються формулами (6) і (7), відповідно. У формулі (6): *txtCng* – типова подія об′єкту *TextBox,* яка відбувається при зміні значення властивості *Text* і реалізується відомою подією *TextChanged* (успадкована від *Control*) [4, 14]; *txtAl* – типовий унітерм-властивість об′єктів *TextBox, TextBlock,* яка повертає або задає спосіб горизонтального вирівнювання текстового та реалізується відомою властивістю *TextAlignment* (успадкована від *Block*) [4, 14]; *mnWdt* – типовий унітерм-властивість об′єкту *TextBox,* яка повертає або задає мінімальне обмеження ширини елементу і реалізується відомою властивістю *MinWidth* (успадковано від FrameworkElement) [4, 14].



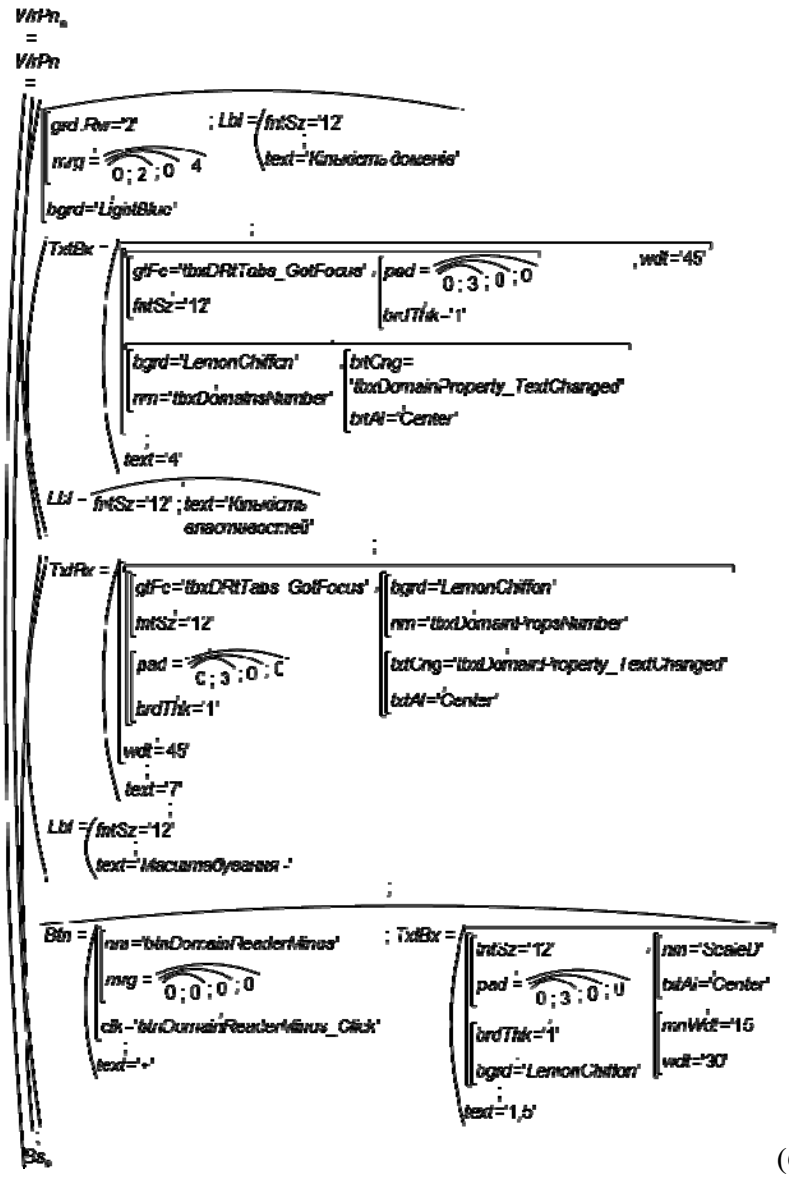
(5)

Секвенція трьох формул (Bs_0) у формулі (6) є описом панелі керування розміром основної таблиці. Унітерми Bs_0 реалізуються елементами відомого класу *Button* та описані таким виразом:

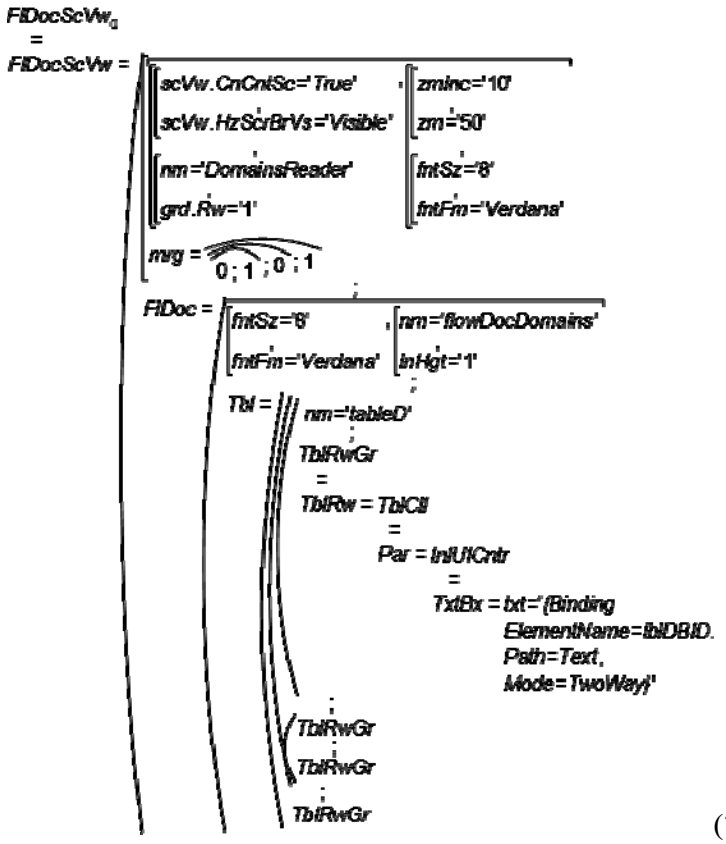


У формулі (7): *FlDocScVw* – типова підсистема перегляду вмісту нефіксованого формату в режимі безперервної прокрутки, яка реалізується відомим класом *FlowDocumentScrollViewer* [4, 14]; *zmInc* – типова властивість відомого об′єкту *FlowDocumentScrollViewer,* яка отримує або задає крок зміни масштабу і реалізується відомою властивістю *ZoomIncrement* [4, 14]; *zm* – типова властивість відомого об′єкту *FlowDocumentScrollViewer,* отримує або задає поточний масштаб і реалізується відомою властивістю *Zoom* [4, 14]; *fntFm* – типова підсистема об′єктів *FlowDocumentScrollViewer, TextBox, ListBox,* яка представляє сімейство зв′язаних між собою шрифтів і реалізується відомим класом *FontFamily* [4, 14]; *FlDoc* – типова підсистема розміщення і форматування вмісту потоку з такими додатковими властивостями документів, як розбиття на сторінки і стовпці, реалізується відомим класом *FlowDocument* [4, 14]; *lnHgt* – типова властивість об′єкта *FlowDocument,* яка отримує або задає висоту кожного рядка вмісту і реалізована відомою властивістю *LineHeight* (успадкована від *Block*) [4, 14]; *Tbl* – типова підсистема вмісту потоку на рівні блоку, яка забезпечує керування у вигляді сітки, впорядкованої в рядки і стовпці та реалізується відомим класом *Table* [4, 14];

TblRwGr – типова підсистема вмісту потоку, яка використовується для групування елементів *System.Windows.Documents.TableRow* у *System.Windows.Documents.Table*, яка реалізується відомим класом *TableRowGroup* [4, 14]; *TblRw* – типова підсистема вмісту потоку, яка визначає рядок з *System.Windows.Documents.Table* і реалізується відомим класом *TableRow* [4, 14];



(6)



(7)

TblCll – типова підсистема вмісту потоку, який визначає осередок вмісту в рамках *System.Windows.Documents.Table* і реалізується відомим класом *TableCell* [4, 14]; *Par* – типова підсистема вмісту потоку на рівні блоку, якавикористовувана для угрупування вмісту в абзаці і реалізується відомим класом *Paragraph* [4, 14]; *InlUICntr* – типова підсистема вмісту потоку на вбудованому рівні, якою упроваджуються елементи типу System.Windows.UIElement (наприклад, System.Windows.Controls.Button) у вміст потоку, реалізується відомим класом *InlineUIContainer* [4, 14]; *txt* – типова властивість, яка отримує або задає текстовий вміст елемента та реалізується відомою властивістю *Text* об'єкта *TextBox* [4, 14].

Функціональний унітерм Grd_0 описується формулою (8). У ній *ClDf* – типова підсистема, яка визначає властивості стовпця, що застосовуються до елементів *Grid*, реалізується відомим класом *ColumnDefinition* [4, 14]; *txtWrp* – типова підсистема об′єктів *TextBox* і *TextBlock,* якою визначається чи переноситься текст при досягнені краю поля, реалізується відомим класом *TextWrapping* [4, 14]; *fntWgt = FontWeight* – клас об′єкту *TextBox,* відноситься до щільності гарнітури (в плані легких або важких контурів); *brdBr* – унітерм-властивість об′єкту *TextBox,* яка повертає або задає пензель і описує фоновий колір межі елементу керування та реалізується відомою властивістю *BorderBrush* (успадкована від *Control*) [4, 14]; *mxHgt* – типова властивість об′єкта *FlowDocumentScrollViewer,* – повертає або задає максимальне обмеження висоти елемента, реалізується відомою властивістю *MaxHeight* (успадкована від *FrameworkElement*) [4, 14]; *BlkUICnt* – типова підсистема вмісту потоку на рівні блоку, якою описується впровадження елементів типу *System.Windows.UIElement* (наприклад, *System.Windows.Controls.Button*) у вміст потоку, реалізується відомим класом *BlockUIContainer* [4, 14]; *LstBx* – типова

підсистема, яка містить список елементів для вибору, реалізується відомим класом *ListBox* [4, 14]; *slctCng* – унітерм-подія об′єктів *ListBox* і *ComboBox,* яка виникає при зміні поточного виділення в *Selector,* реалізується відомою подією *SelectionChanged* (успадкована від *Selector*) [4, 14].

Унітерм Gs_0 описується формулою (9), де *GrdSpl* – типова підсистема, яка поширює простір між стовпцями або рядками елемента керування *Grid*, реалізується відомим класом *GridSplitter* [4, 14]; *RdBtn* – типова підсистема - перемикач, який користувач може вибирати, але не відміняти вибір, реалізується відомим класом *RadioButton* [4, 14]. Властивість *System.Windows.Controls.Primitives. ToggleButton.IsChecked* елемента *System.Windows.Controls.RadioButton* встановлюється, коли користувач цоком клавіші мишки вибирає цей елемент, але очистити властивість можна тільки програмним способом; *isChkd* – типовий унітерм- властивість об′єктів *RadioButton* і *CheckBox,* яка вказує чи є *System.Windows.Controls.Primitives.ToggleButton* у включеному стані, реалізується відомою властивості *IsChecked* [4, 14]; *grpNm* – типовий унітерм-властивість об′єкту *RadioButton,* яка визначає котрі елементи керування *RadioButton* є взаємовиключними, реалізується відомою властивості *GroupName* [4, 14]; *ChBx* – типова підсистема встановлення і знімання прапорця-ознаки, яка реалізується відомим класом *CheckBox* [4, 14]; *ScVw* – типова підсистема представлення прокручованої області з видимими елементи, яка реалізується відомим класом *ScrollViewer* [4, 14]; *cnCntScrl* – типовий унітерм-властивість об'єкту ScrollViewer, яка повертає або задає значення дозволу прокрутки для елементів, що підтримують інтерфейс *IScrollInfo*, реалізується відомою властивістю *CanContentScroll* [4, 14]; *hzScrlBVs* – типовий унітерм-властивість об'єкту *ScrollViewer*, яка вказує, чи потрібно відображати горизонтальну смугу прокрутки *ScrollBar*, реалізується відомою властивістю *HorizontalScrollBarVisibility* [4, 14];

flDrc – типовий унітерм-властивість об'єкта local:AACanvas, яка повертає або задає напрям потоку тексту та інших елементів user interface (UI) у середині будь-якого батьківського елемента, керуючого їхньою структурою, реалізується відомою властивістю *FlowDirection* (успадкована від FrameworkElement) [4, 14]; *isHtTstVsb* – типовий унітерм-властивість об'єкта local:AACanvas, яка повертає або задає значення, що оголошує, чи може даний елемент бути повернений як результат перевірки вибору в деякій частині його візуалізованого вмісту, реалізується відомою властивістю *IsHitTestVisible* (це властивість залежностей успадкована від UIElement) [4, 14]; *clpTBnd* – типовий унітерм-властивість об'єкта local:AACanvas, яка повертає або задає значення про відсікання вмісту даного елемента або вмісту дочірніх елементів даного елемента, щоб він поміщався у розмір батьківського елементу, реалізується відомою властивістю *ClipToBounds* (це властивість залежностей успадкована від UIElement) [4, 14]; *fcsb* – типовий унітерм-властивість об'єкта local:*AACanvas*, яка повертає або задає значення про можливість отримувати елементом фокусу, реалізується відомою властивістю *Focusable* (успадкована від *ContentElement*) [4, 14].

Формули Grd_1, Grd_2 містять модель панелі керування операціями з базою даних, вибору назви бази даних та джерела формул для зберігання.

Унітерм Grd_1 описується формулою (10), де *TxtBlk* – типова підсистема, яка забезпечує спрощене керування відображенням невеликої кількості потоку вмісту, реалізується відомим класом *TextBlock* [4, 14]; *DkPn* – типова підсистема, яка визначає область, у якій можна упорядкувати дочірні елементи горизонтально або

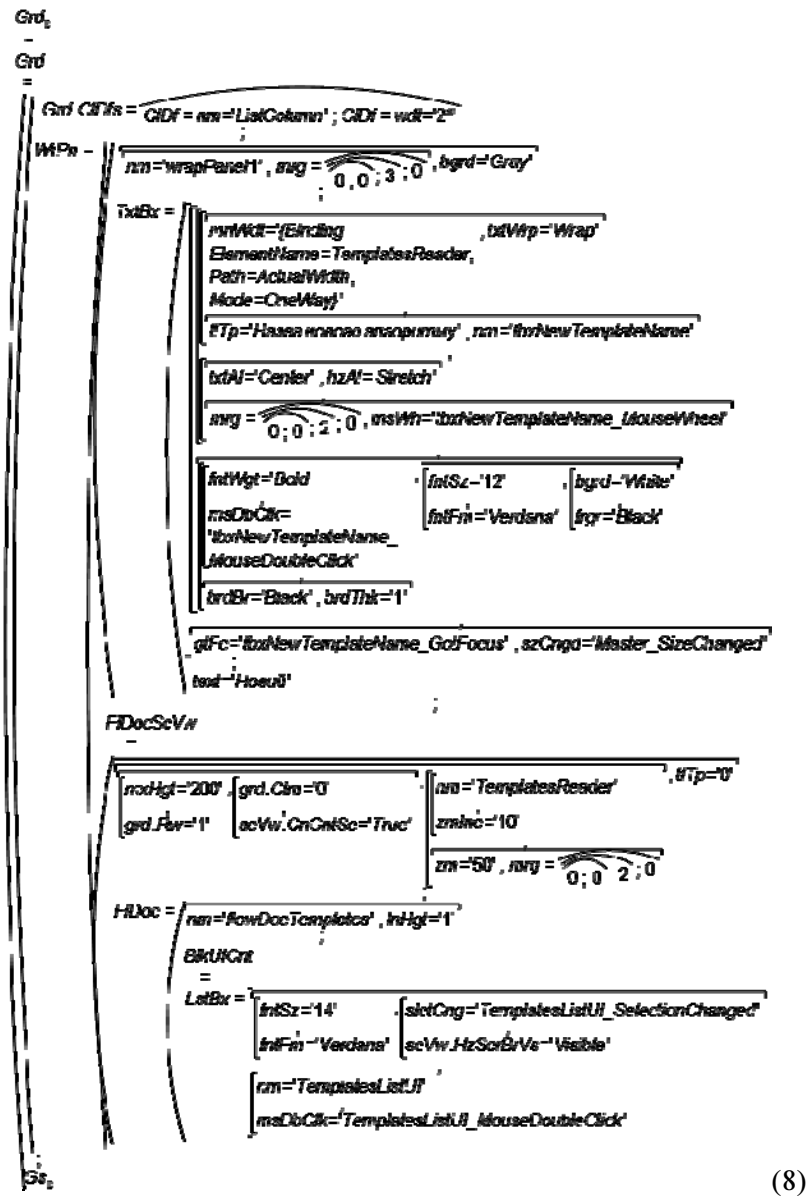
вертикально щодо один одного, реалізується відомим класом *DockPanel* [4, 14]; *CbBx* – типова підсистема, яка призначена для вибору значення із списку що розкривається, який можна відображати і приховувати натисненням кнопки із стрілкою, реалізується відомим класом *ComboBox* [4, 14]; *isEdtb* – типовий унітерм-властивість об'єкта ComboBox, яка повертає або задає значення дозволу або заборони редагування тексту в текстовому полі елемента керування ComboBox, реалізується відомою властивістю *IsEditable* [4, 14]; *isRdOy* – типовий унітерм-властивість ComboBox включення режиму тільки для читання, у якому вміст поля із списком можна вибирати, але не редагувати, реалізується відомою властивістю *IsReadOnly* [4, 14]; *CbBxIt* – типова підсистема вибору значення із *System.Windows.Controls.ComboBox*, реалізується відомим класом *ComboBoxItem* [4, 14]; *isSlct* – типовий унітерм-властивість *ComboBoxItem* виброру *ListBoxItem*, реалізується відомою властивістю *IsSelected* [4, 14].

Унітерм Grd_2 описується формулою (11), де *opct* – типовий унітерм-властивість об'єкта *Button*, яка задає коефіцієнт непрозорості (властивість залежностей успадкована від *UIElement*), реалізується відомою властивістю *Opacity* [4, 14].

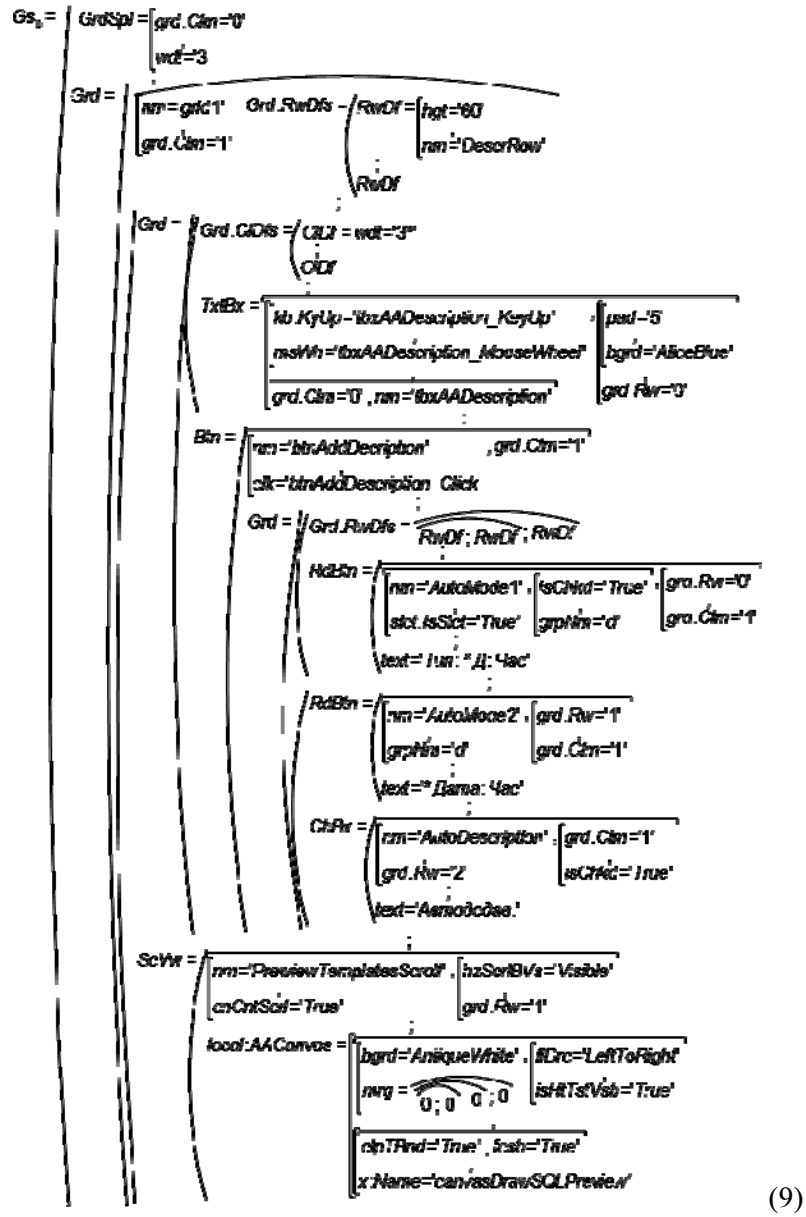
Декомпозовані формули $WrPn_1$, $WrPn_2$, $FlDocScVw_1$, $FlDocScVw_2$, Btn_1, Btn_2, які містить формула (1), описують графічну частину моделі формування описів відношень та секвентних послідовностей компонентів відношення. За виключенням назв елементів (властивість nm) та обробників подій, наведені формули повністю аналогічні формулам $WrPn_0$, $FlDocScVw_0$, Btn_0, тому їх опис у даній роботі опущено.

Реалізація побудованої моделі

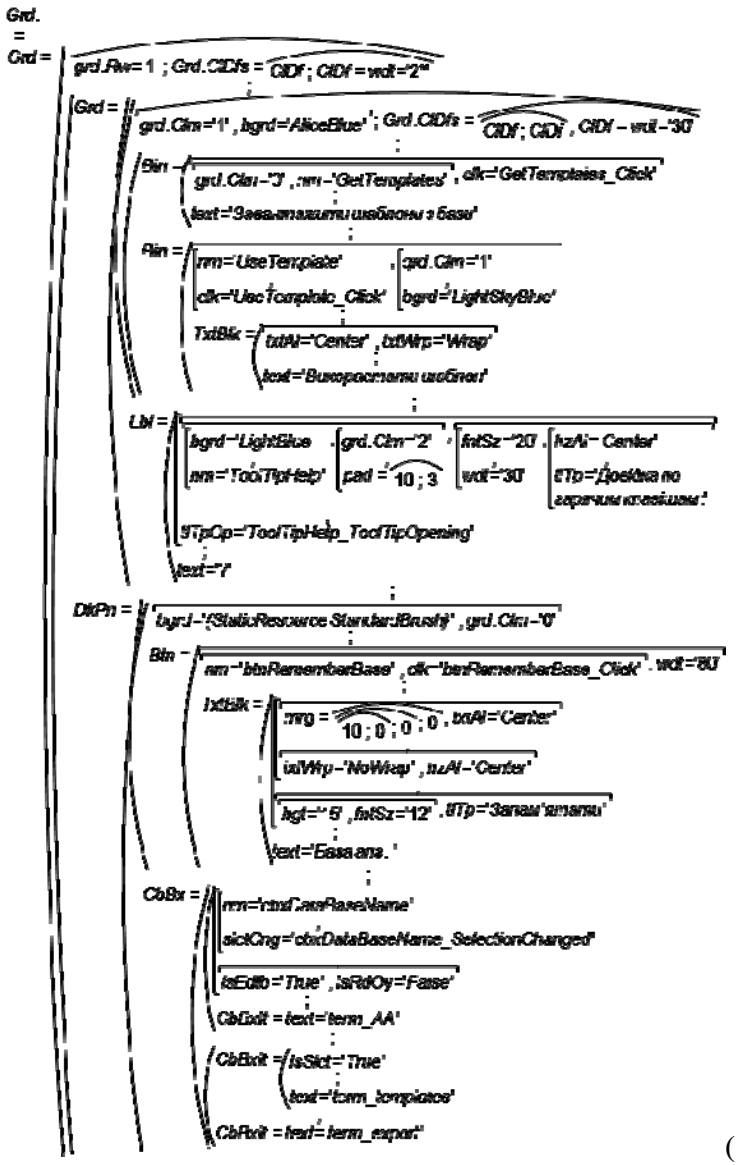
Реалізація моделі представлена вікном *BDMaster*, описаним мовою розмітки XAML у вигляді дерева елементів класу *Control*, елементів–контейнерів *TableRowGroups*, які вміщують такі основні частини: *TableRowGroup0* – контейнер назви таблиці, *TableRowGroup1* – контейнер заголовку таблиці, *TableRowGroup2* – зміст таблиці, *TableRowGroup3*– рядок формування змісту нового елементу з кнопкою його додавання до таблиці.



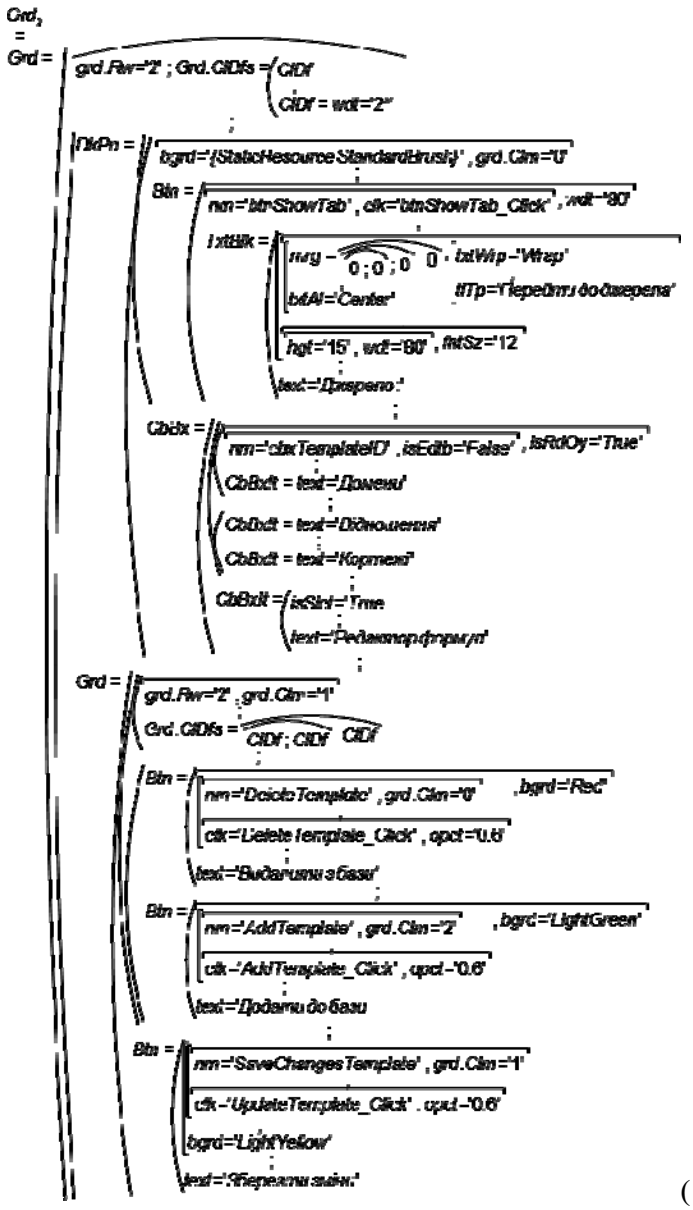
(8)



(9)


Grd.
=
Grd =
grd.Rw=1 ; Grd.ClDfs = ClDf ; ClDf = wdt="2"
Grd =
grd.Cln='1' , bgrd='AliceBlue' ; Grd.ClDfs = ClDf ; ClDf , ClDf – wdt–'30'
Btn –
grd.Cln–'3' , nm–'GetTemplates' , clk='GetTemplates_Click'
text='Завантажити шаблони з бази'
Pln =
nm='UseTemplate'
clk='UseTemplate_Click'
grd.Cln='1'
bgrd='LightSkyBlue'
TxtBlk =
txtAl='Center' , txtWrp='Wrap'
text='Використати шаблон'
Lbl =
bgrd–'LightBlue
nm='ToolTipHelp'
grd.Cln–'2'
pad = 10 ; 3
fntSz–'20'
wdt='30'
hzAl– Center'
tTp='Довідка по гарячим клавішам:'
tTpOp='ToolTipHelp_ToolTipOpening'
text='i'
DtPn =
bgrd–'{StaticResource StandardBrush}' , grd.Cln–'0'
Btn –
nm–'btnRememberBase' , clk–'btnRememberBase_Click' . wdt='80'
txtBlk =
mrg = 10 ; 0 ; 0 ; 0 , txtAl='Center'
txtWrp–'NoWrap' , hzAl–'Center'
fgt='5' , fntSz='12' . tTp='Запам'ятати'
text='База ап. '
CbBx =
nm='cmbDataBaseName'
selChng='cbxDataBaseName_SelectionChanged'
isEdtb='True' , isRdOy='False'
CbExlt = text='term_AA'
CbExlt = isSlct='True'
text='term_template'
CbExlt = text='term_expert'

(10)



(11)

Висновки

Створена математична модель графічної підсистеми описує можливості автоматизованого синтезу формул, якими, у вигляді формул алгоритмів, описуються моделі реляційних баз даних.

Моделі реляційних баз даних можуть бути оптимізовані за кількістю унітермів і використані для автоматичного генерування програмного коду баз даних.

1. Ovsyak V.K. Computation models and algebra of algorithms /Ovsyak V.K.// Інформаційні системи та мережі. Вісник Національного університету «Львівська політехніка». 2008. − №621. − с. 3-18.

2. Owsiak W., Owsiak A.. Rozszrzerzenie algebry algorytmów //Pomiary, automatyka, kontrola 2, 2010. –S.184-188.

3. Owsiak W., Teoria algorytmov abstrakcyjnych i modeowanie matematyczne systemow informacyjnych./ Qwsiak A., Owsiak J.// −Opole: Studia i monografie. z. 176. ”Politechnika opolska”, 2005. − 275 s.

4. Petzold C. Programowanie Windows w języku C#. –Warszawa: „RM”, 2002. – 1161 s.

5. Алекс Кригель, Борис Трухнов SQL. Библия пользователя. Язык запросов SQL, 2-е издание = SQL Bible, 2nd edition — М.: «Диалектика», 2009. — 752 с. — ISBN 978-5-8459-1546-7.

6. Дейт К. Дж. Введение в системы баз данных, 8-е издание.: Пер. с англ. — М.: Издательский дом "Вильямс", 2005. — 1328 с.

7. Джеймс Р. Грофф, Пол Н. Вайнберг, Ендрю Дж. Оппель. SQL: полный справочник, 3-е издание = SQL: The Complete Reference, Third Edition — М.: «Вильямс», 0. — 960 с. — ISBN 978-5-8459-1654-9.

8. Кодд Є. Ф. Расширение реляционной модели для лучшего отражения семантики. /Є. Ф. Кодд //Системы управления базами данных 5/1996. — Переклад російською з оригіналу E.F. Codd. Extending the Database Relational Model to Capture More Meaning. //ACM Transactions on Database Systems, Vol. 4, # 4, December 1979.

9. Кодд Є. Ф. Реляционная модель данных для больших совместно используемых банков данных/Є. Ф. Кодд // журнал Системы Управления Базами Данных # 1/1995. — Переклад російською з оригіналу //E.F. Codd. A Relational Model of Data for Large Shared Data Banks. Communications of the ACM, Volume 13, Number 6, June, 1970.

10.Коннолли Т., Бегг К. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. 3-е издание. : Пер. с англ. — М. : Издательский дом "Вильямс", 2003. — 1440 с.

11.Кулик С., Овсяк В. Синтаксис і семантика базових понять реляційної моделі та операції. Комп'ютерні технології друкарства № 23/2010, Львів. - с. 70-79

12.Кулик С.О. Овсяк В.К. Опис операцій реляційної алгебри засобами алгебри алгоритмів. Поліграфія і видавнича справа №1(51)/2010, Львів. - с. 68-79

13.Кулик С.О. Овсяк В.К. Опис структурної частини реляційної моделі баз даних засобами алгебри алгоритмів. Квалілогія книги №2(16)/2009, Львів. - с. 51-60

14.Мак-Дональд Мэтью. Windows presentation foundation в .NET 3.5 с примерами на C# 2008. – Москва, Санкт-Петербург, Киев: “Apress”, 2008. – 922 с.

15.Овсяк В.К. АЛГОРИТМИ: аналіз методів, алгебра впорядкувань, моделі, моделювання. - Львів: 1996. – 132 с.

16.Овсяк В.К. Засоби еквівалентних перетворень алгоритмів інформаційно-технологічних систем. / Овсяк В.К.//Доповіді Національної академії наук України, 1996, №9. -С. 83-89.

17.CA ERwin Data Modeler, CA technologies, http://erwin.com/products/detail/ca_erwin_data_modeler_standard_edition/

18.Data Architect, TheKompany, http:///thekompany.com/products/dataarchitect/

19.DBDesigner 4, fabFORCE, http://www.fabforce.net/dbdesigner4/

20.Oracle Corporation, www.oracle.com/products/

21.Rational Products IBM Corporation, www.ibm.com.

22.The Third Manifesto by Hugh Darwen and C.J. Date, http://www.thethirdmanifesto.com/

23.Visual Architect, Visual Paradigm, http://www.visual-paradigm.com/product/?favor=dbva