УДК 004.415.2.043

ПУСТОВАРОВ В.І.,
КОВАЛЕНКО С.Ю.

ОРГАНІЗАЦІЯ СХОВИЩ ДАНИХ ДЛЯ МОВ ОПИСУ ТА МАНІПУЛЯЦІЙ З ІНФОРМАЦІЙНИМИ СУТНОСТЯМИ

Проаналізовані поширені структури даних, методи та засоби накопичення інформаційних сутностей в комп'ютерних ресурсах розв'язання задач інженерії знань. Запропоновані універсальні структури зберігання сховищ аналітичних і управляючих даних та узагальнена архітектура баз знань та їх елементів для управління маніпуляціями при виконанні логічного доведення. Обґрунтовані переваги архітектури з аналітичними компонентами в гнучкості і швидкості при розв'язанні зв'язаних комплексів задач на графових моделях різноманітних проблемних галузей з різними рівнями деталізації інформаційних сутностей.

Methods and tools for storage of informational entities are analyzed for computer recourses of computer task decision. Structures for unified warehouses of analytic and control data and generalized knowledge base architecture are proposed for logical deduction manipulation. Analytical component architecture advantages in flexibility and speed are shown for complex task decision for various problem area graph models with different information entity implementation.

Вступ

На сьогоднішній день поширюється використання різноманітних середовищ розробки баз знань та комп'ютерних ресурсів обробки, що забезпечують розв'язання комплексів задач проектування та аналізу для заданої проблемної галузі. Разом з тим, сучасні засоби розробки програм [1] спрямовуються на розв'язання задач, які стосуються не лише вузько спеціалізованої проблемної галузі, а також торкаються суміжних галузей і задач. До них зазвичай відносять базові напрями синтезу і аналізу елементів технічних проектів, наприклад, розробку апаратних і програмних ресурсів розв'язання комплексів задач (РРКЗ) в заданій проблемній галузі. Виходячи з перспективного використання таких ресурсів та їх специфікацій [5] в суміжних галузях, доцільним є еволюційний розвиток узагальненого подання інформаційних сутностей об'єктів в комп'ютерних сховищах [4], що передбачає можливість модифікації методів створення, керування та відображення об'єктів.

Для вирішення цього питання потрібно подолати проблему різноманіття форматів представлення та зберігання платформенонезалежних даних, які входять до складу створюваних проектів [2, 3]. На сьогоднішній день практично кожна фірма-розробник РРКЗ має низку власних форматів даних навіть у деяких випадках в рамках однієї проблемної галузі. Таке різноманіття форматів приводить користувачів до вибору між двома шляхами застосування пакетів РРКЗ. Перший варіант полягає у використанні деякого базового пакету РРКЗ з усіма його перевагами і недоліками; інший – у використанні декількох програмних пакетів та часто повторюваних процедурах конвертування різних форматів даних. Крім того, взагалі відсутні механізми взаємодії між середовищами розробки, які використовуються для різних проблемних галузей, не зважаючи на те, що розробки комплексних та узагальнених рішень стають все більш популярними. До того ж, значна різноманітність апаратних та програмних платформ і використовуваних базових продуктів є причиною виникнення величезної кількості програм, необхідних для роботи на базі різних платформ реалізації програм та обладнання. За умов досить швидкого розвитку апаратних і програмних платформ очевидно, що кількість таких продуктів буде збільшуватися і надалі [4].

Все це обґрунтовує необхідність розробки єдиного стандартного або уніфікованого механізму для збереження маніпулювання і представлення інформаційних сутностей, що мають можливість відобразити все різноманіття сутностей в найбільш універсальному і найчастіше неповному, розрідженому поданні. Необхідні уніфіковані подання конструкцій комп'ютерних мов повинні відпові-

дати наступному комплексу вимог, що забезпечують:

- абстрагування скороченням та/або деталізацію додаванням інформаційних сутностей в моделях предметної галузі та їх об'єктах;
- створення узагальненої схеми збереження даних про інформаційні сутності, їх зв'язки, типізацію та іменування їх атрибутів для довільної проблемної галузі;
- полегшення перетворень зв'язків, вхідних та цільових змінних і критеріїв задач на обчислювальні формули та оператори програм;
- розв'язання кінцевої множини задач аналізу, настройки або оптимізації та синтезу комплексів об'єктів проблемної галузі для досягнення поставленої прагматичної мети;
- визначення баз даних і знань для зберігання інформаційних сутностей РРКЗ та результатів їх обґрунтування для будь-яких конфігурацій і комбінацій платформ проектування.

Постановка задачі

При створенні ефективних сховищ необхідно узагальнити вхідні комп'ютерні мови проектування РРКЗ і внутрішнє подання їх конструкцій та інформаційних сутностей. Більш чітке окреслення цілей та вимог до вдосконалених комп'ютерних мов полягає в розробці універсальних та зручних засобів для подання і перетворення різноманітних даних.

Основними задачами при розширеннях існуючих мов є спрощення сприйняття створюваного сценарію та перебудова орієнтуючої концепції елементів опису мови і керування даними таким чином, щоб включити в систему об'єкт-відображення як універсальний елемент взаємозв'язків. Такі описові елементи мови забезпечать компактне подання моделей та використання спеціальних вбудованих пакетів РРКЗ для зберігання та маніпуляцій даними. Крім того, платформенно-незалежний спосіб взаємодії між внутрішніми частинами сховища та зовнішніми клієнтами забезпечить зменшення кількості програм цільової обробки і спростить механізми взаємодії з даними.

Виходячи з існуючої ситуації, поставимо за мету обґрунтування доцільності реалізації абстрактної мови опису та маніпуляції інформаційними сутностями та програмного пакету для обробки інформаційних сутностей будь-якого типу і виду, в тому числі, управляючих і доказових сутностей. Наявність такого пакету РРКЗ дозволить як більш ефективно використовувати існуючі на сьогодні пакети проектування, так і створювати нові універсальні пакети для розробки програм, моделей, будь-яких інформаційних сутностей, які мають можливість взаємодіяти між собою, використовуючи уніфіковану організацію сховищ.

Пропозиції щодо розв'язання задачі

При розробці спеціалізованих галузевих програмних комплексів для моделювання об'єктів, середовищ їх існування, тощо детально розглядаються різноманітні об'єкти, їх проблемно-спрямовані описи різних рівнів деталізації, які набувають все більш спеціалізованих форм з ускладненням комплексу. Найбільш логічним шляхом розв'язання цієї проблеми є використання графових структур, поданих в форматі реляційних баз даних.

Для графових структур таке узагальнення не несе в собі суттєвих ускладнень механізму обробки даних, в той час як для реляційних (табличних) баз даних воно призводить до значного ускладнення структур даних або до зменшення ефективності роботи системи. Використання виключно графового подання даних призводить до ускладнення сприйняття та зменшення ефективності роботи сховища при доступі до даних. Тому доцільно знайти компроміс між швидкодією при використанні таблиць та універсальністю графового подання і деталізації даних.

Концепція сховищ подання інформаційних сутностей. Концепцію розробки та розвитку будь-якої мови має формувати саме цільове спрямування мови. В нашому випадку основна мета створення узагальненого внутрішнього представлення – це універсалізація описів, реалізація та настроювання РРКЗ. Виходячи з цього, запропоновані сховища мають забезпечувати максимальну гнучкість та багатоваріантність їх використання. Їх організація має бути найпростішою для будь-яких комп'ютерних мов.

Крім того, форма внутрішнього представлення має забезпечити можливість написан-

ня послідовностей дій над інформаційними сутностями та відображеннями. Послідовності кодів, які являють собою цільові обробники, матимуть можливість працювати з даними в середині сховища, не використовуючи зовнішніх ресурсів та сторонніх засобів програмування.

Базові виконавчі оператори комп'ютерних мов включають в себе блочні оператори структурного програмування зі специфічними щодо мов модифікаціями синтаксису і можуть відображуватися графами [2]. Приклад графа підлеглості операторів і операцій для оператора циклу з передумовою мови C++ наведено на рис. 1:

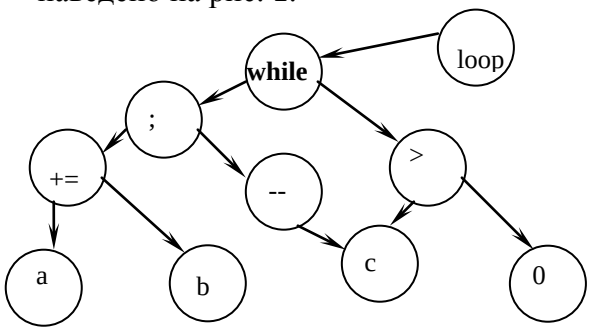

Рис. 1. Граф підлеглості оператора **do**{a+=b-c; --c;}**while**(c>0);

Як видно з рис. 1 узагальнення ациклічного графу підлеглості для побудови структур умовних операторів та операторів циклів повинно включати єдиний кореневий вузол для представлення кожного оператора. Зворотні семантичні зв'язки в графах операторів структурного програмування умовно опускаються на період синтаксичної та пошукової обробки, а послідовності операторів організуються так, щоб представлення першого з них знаходилося в найвіддаленішій лівій гілці графа.

Основу ідентифікації будь-якої інформаційної сутності складає заголовок її опису. В загальному випадку інформаційну сутність складають будь-які об'єкти даних, специфікацій [5] та управління обчисленнями. Ациклічний граф, наведений на рис.1, можна узагальнити для відображення типів та примірників даних, виділивши в них ідентифікаційний заголовок та конкретизацію сутності. На рис. 2 показано приклад опису класу мови C++ з метапозначеннями проміжних конструкцій та ідентифікАцією РРКЗ через вузол *ім'я*.

Організація описів та зв'язків даних у вигляді ациклічних графів дає перевагу у вигляді максимальної універсальності опису структур даних. Разом з тим вона породжує ряд складнощів у забезпеченні цілісності динамічних даних. Вони вирішуються шляхом створення графів спеціальних списків власників об'єктів та їх відображень.

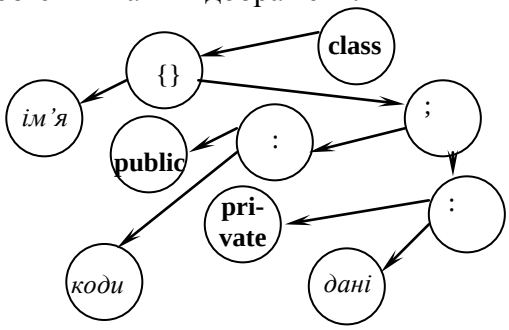

Рис. 2. Приклад дерева визначення класу об'єктів мови C++ **class** *ім'я*{**public**:*коди*;**private**:*дані*;}

Подібні структури надають можливість формувати комплексні (складені або складні) групи об'єктів. При цьому як нетермінальні вузли графів використовуються дужки ("{}") та роздільники (":") групування, а також роздільники переліку ("," і ";"). Для конкретизації об'єктів використовуються спеціальні графи описів, що посилаються на попередньо визначений тип користувача. Приклад такої структури показаний на рис. 3, має вигляд ациклічного графа, в якому термінальні вузли вміщують ідентифікацію та характеристики примірників об'єктів.

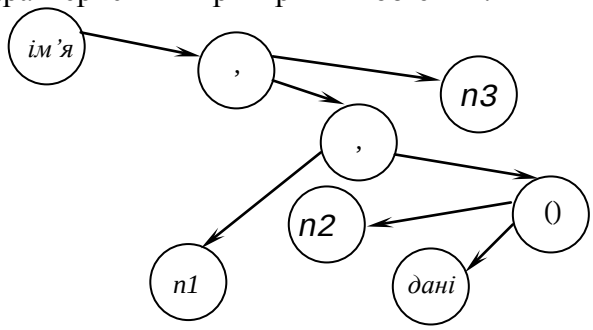

Рис. 3. Приклад дерева визначення примірників об'єктів в мові C++ *ім'я* *n1*,*n2*(*дані*),*n3*;

За аналогією з описом класів та примірників об'єктів можна будувати і такі ж складні графи описів типів даних користувача з переліком полів та примірників даних. Для того щоб довести суперечність або надлишковість описаних даних, подібні представлення слід використати для специфікацій [5] зв'язків типів і примірників об'єктів.

Графові представлення специфікацій на додаток до елементів групування і переліку

[3, 5] включають теоретико-множинні і звичайні математичні унарні та бінарні операції для визначення доменів, структур і зв'язків. Це дозволяє також зберігати відношення або обчислювальні формули, виклики функцій та коди для комп'ютерних обчислень.

Послідовності перетворень [3], що використовуються для доведень і формування розрахункових формул, включають ланцюжки операцій еквівалентних та прямих або зворотних імплікативних перетворень. Такі доведення та обґрунтування доцільно зберігати в базах знань, що управляють роботою зі створення, використання і супроводження бібліотек РРКЗ для заданої проблемної галузі. Вони зберігають базові інформаційні сутності для наступних доведень і формальної верифікації розроблюваних РРКЗ.

Пошук елементів бібліотек РРКЗ з синтаксичними та синтаксичними ознаками і зберігати послідовності доведення коректності. Заголовки елементів бібліотек повинні включати ідентифікацію функцій та версій елементів, покажчик на записи специфікації і про доведення відповідності специфікації, а також про оцінки критеріїв продуктивності та необхідних витрат для експлуатації РРКЗ.

Таблиці ідентифікації інформаційних сутностей доцільно побудувати як індекси з використанням покажчиків на інформаційні поля графів. Таким чином, загальне використання графово-табличного представлення обґрунтовується універсальністю логічного представлення, яку вони забезпечують.

Набір структур даних для представлення сутностей, який використовується в узагальненому форматі сховища, являє собою поєднання послідовного представлення даних на нижньому рівні та представлення структурованих груп даних у вигляді ациклічних графів на верхньому. Ациклічні графи, приклади яких показані на попередніх прикладах, обираються як інструмент структуризації даних, тому що вони надають максимальну гнучкість та універсальність у поданні інформаційних та управляючих сутностей. Поля вузлів графа дозволяють мінімізувати зібрану інформацію, яка може бути опущена при відсутності необхідності її використання.

Для обґрунтування такого вибору використані наступні міркування. Найбільш популярним варіантом структур даних на сьогоднішній день являються реляційні табличні бази даних. Вони є найбільш зручними та ефективними для описання порівняно простих та одноманітних об'єктів. Але об'єкти, що використовуються у програмних пакетах для моделювання та розробки не являються простими та одноманітними. це призводить до виникнення значної кількості таблиць для всеохоплюючого подання інформації, як наслідок, зниження ефективності їх використання.

Натомість при використанні ациклічних графів введення нових об'єктів, зміна структури інформаційних сутностей вже існуючих об'єктів та створення додаткових відображень (найбільш поширена операція в системах проектування) не потребують значних додаткових витрат ресурсів. В той самий час нерідко виникає ситуація, коли при зміні структур даних інформаційних сутностей доводиться заново проектувати табличну базу даних.

Структура даних, що пропонується для використання в удосконаленому сховищі має дуальний вигляд і складається з 2 частин: декларативної та інформаційної, що поєднуються спільним кореневим вузлом графа.

Декларативна частина структури даних представлена окремою структурою даних, в якій зібрані описи для відображень, що використовуються в програмному пакеті. В ній міститься інформація про тип і зміщення кожного поля даних у відображенні відносно адреси початку області пам'яті що займає відображення. Це необхідно для надання можливості оперувати окремими полями даних та збереження інформації про наявні види відображень. Декларативна частина створюється як універсальна для всіх об'єктів РРКЗ, представлених у системі.

Також окремо визначається група узагальнених описів, призначених для створення визначень, об'єктів та відображень, їх редагування та видалення. Її основна задача – дозволяти редагувати об'єкти та структуру інформаційних сутностей засобами мови. До неї входять наступні команди визначення даних і графів описів. За аналогією з мовою SQL це – Create Instance <Ім'я об'єкта> is <Ім'я визначення>: <Визначення об'єкта з присвоєнням попередньо заданих значень>; Create Reflection <Ім'я відображення> : <Ви-

значення відображення з присвоєнням попередньо визначених значень> – команди для створення відображення.

Набір процедур обробників для перетворення сутностей мають містити арифметичні та логічні оператори, оператори зчитування та модифікації полів відображень, деякі стандартні оператори виводення та оператори для управління відображеннями та об'єктами.

Кожне відображення містить дані для певної проблемної галузі і певного рівня деталізації у вигляді таблиці. При цьому в ній не міститься даних, що мають обов'язкові відношення до інших галузей або рівнів деталізації. Відношення між даними реалізуються зв'язками відображень або спільними відображеннями. Таке представлення даних відповідає асоціативності та проблемній орієнтованості сприйняття і проектування структур даних і забезпечує оптимальне універсальне представлення даних будь-якої структури.

Крім того, запропоноване узагальнене представлення має надавати можливість описати будь-яку кількість відображень інформаційної сутності об'єкта. Відображення доцільно орієнтувати на задану проблемну галузь. Відображення в свою чергу можуть включати в себе відображення наступного рівня, орієнтовані на роботу з даними, що відносяться до певного більш вузького напряму заданої проблемної області. Вони також зберігаються як ациклічний граф, що забезпечує можливість створювати сутності даних з інформацією, відповідальною за зв'язок даних з різних проблемних областей.

Набір операцій для роботи з даними графів. Всі операції, необхідні для забезпечення вищевказаних функцій, можна розділити на декілька функціональних груп. Відповідно до них сховище деталізується як база для реалізації інтерпретатора запропонованих розширень мови SQL і швидкісних перетворень настроювання.

Перший тип операцій – операції доступу до даних та обробників. Вміщує в себе операції для отримання інформації з відображення предмету та запуску обробників. Також в цю групу доцільно включити операції встановлення фільтрів для відображень. На додаток до команд мови маніпуляції даними SQL група додаткових команд вибірки включає команди: встановлення фільтру для відображення (SetFilter); встановлення фільтру на певне відображення певного предмета (SetIndividualFilter); одержання інформації про об'єкт або його частину (Get (Sub)Object); пошук схожого підграфа за заданою мірою схожості (FindSimilar); одержання інформації про відображення відповідно встановленого фільтра (GetDefinition); старт обробника відображень (RunReflector).

Команди корекції, що додаються до мови SQL, включають додавання додаткового підграфа до заданого вузла (AddSubTree); вилучення підграфа, починаючи з заданого вузла (AddSubTree); перетворення графа для розрахунку заданого цільового вузла (ConvToGoal).

Відповідно до функцій сховища даних, друга група команд має дозволяти працювати з обробки правил операції занесення корекцій графів. Наразі для роботи з постійними даними використовуються функції файлової системи. Тому для збереження та перенесення даних на постійних носіях потрібно забезпечити можливість створювати та видаляти папки та зберігати сутності та групи сутностей в певних файлах, а також відновлювати їх в процесі завантаження файлу.

Висновки

Створення сховищ для середовищ проектування РРКЗ, орієнтованих на збереження даних забезпечує гнучко та універсально інфраструктуру для необмежених структур даних. Вона орієнтована на роботу з інформаційними сутностями, що мають різноманітну структуру даних. Її методи та оператори дозволяють виконувати дії з даними як з допомогою зовнішніх скрипкових сценаріїв, так і на стороні сервера.

Архітектура сховища даних, описана в статті, дозволяє зв'язати робочий простір сховища з іншими робочими просторами для інших програм. Дана взаємодія на файловому рівні доволі сильно спрощує деякі ключові моменти роботи середовищ розробки.

Додатковою можливістю взаємодії є варіант реалізації сховища як надбудови на існуючі варіанти реляційних СУБД. Це дозволяє змінювати дані в інтерактивному

режимі. Крім того, за допомогою трансакцій в базах даних можлива взаємодія на рівні безконфліктного зчитування даних для будь-якої кількості програм.

Ще одним варіантом перетворень, при якому атомарною є взаємодія з записами бази знань, є обробка специфікацій програм та сумісна обробка специфікацій та програмних кодів. Вони дозволяють досить зручно відображати структури даних, які використовуються у сховищі. Проте робота з ними потребує гігантських обсягів обчислень, і тому допускається лише при невеликому робочому завантаженні.

Виходячи з цього найбільш ефективною є реалізація сховища у складі окремого програмного пакету з можливостями управління паралельною обробкою даних. Описаний в статті варіант графової архітектури та методів перетворень сховища є доволі гнучким для забезпечення обробки на рівні перетворення даних для різних форматів, а прикладний програмний інтерфейс забезпечує можливість безпосереднього звертання програм до сховища даних. Такий метод взаємодії з базою знань про бібліотеки РРКЗ є найбільш безпечним і ефективним при настроюванні програм з усіх методів збереження інформаційних сутностей, що використовуються у вбудованих базах даних і знань.

Список посилань

1. Ноутон П., Шилдт Г. Java™ 2: Пер. с англ. – СПб.: БХВ-Петербург, 2007. – 1072 с.
2. Пустоваров В.І. Побудова баз знань для сумісного створення програм і обладнання та їх формальної верифікації.: Вісник Національного технічного університету “Київський політехнічний інститут”. Інформатика, управління та обчислювальна техніка. К.: «Век+». – 2008.– 49.– с. 41 – 46.
3. Буханцов С.Н., Пустоваров В.І., Стіренко С.Г., Желудкова Т.В. Доказове управління контролю генерацією та модифікацією програмних модулів для інформаційно-аналітичних систем.: Вісник Національного технічного університету “Київський політехнічний інститут”. Інформатика, управління та обчислювальна техніка. К.: «Век+». – 2008.– 48.– с. 49 – 53.
4. Inmon W. H. Building the Data Warehouse. Wiley Publishing, Inc. 2005 – 543p.
5. Woodcock J., Davies J. Using Z. Specification, Refinement, and Proof. C.A.R. Hoare Series editor, 1995 – 390 p.

Поступила в редакцию 8.12.2009