УДК 681.3.06

©1999 р. Д.В. Режко, М.В. Остапович

Чернівецький державний університет ім.Ю.Федьковича, Чернівці

ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ У СИСТЕМАХ АВТОМАТИЧНОГО ПРОЕКТУВАННЯ РАДІОЕЛЕКТРОННОЇ АПАРАТУРИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Методами об'єктно-орієнтованого програмування розроблена навчальна інструментальна система ELECTRO для аналізу електричних кіл.

Educating tool system ELECTRO for electric circuits analysis is developed with the methods of object-oriented programming.

Методи структурного програмування, які використовуються для розробки програмного забезпечення автоматизованих систем проектування, управління, тощо, мають великі недоліки:

- непридатність для розробки складних програмних систем (до яких можна віднести САПР);
- складність внесення у систему наступних змін;
- обов'язкове повторне виконання усіх етапів розробки.

Усіх цих недоліків не має об'єктно-орієнтоване програмування (ООП). Воно представляє собою послідовний ітеративний процес, який дозволяє безболісно вносити будь-які зміни у налагоджений програмний продукт, а також сприяє повторному використанню написаного раніше коду. Завдяки ООП розв'язання проблем розробки складного програмного забезпечення суттєво спрощується. Прийшовши на зміну структурному програмуванню, ООП у даний час є найбільш перспективним і займає провідне місце у розробці складних проектів. Застосування об'єктів (у розумінні ООП) дозволяє якнайкраще моделювати структуру та поведінку предметів реального світу [1].

Мета даної статті – показати основні принципи застосування ООП при розробці САПР РЕА на прикладі навчальної інструментальної системи аналізу електричних кіл ELECTRO.

Ця система побудована за допомогою Borland Delphi – засобу швидкої розробки програмного забезпечення для Windows NT/Windows 95(98). Delphi пропонує об'єктно-орієнтовану структуру на базі Windows і забезпечує потужну бібліотеку об'єктно-орієнтованих компонент (Visual Component Library – VCL). Інтерфейс користувача будується на основі візуального проектування, що дозволяє створювати складні програмні проекти за відносно короткий час.

Система ELECTRO призначена для розрахунку електричних кіл, представлених у еквівалентному вигляді за допомогою базових схемних елементів (індуктивності, ємності, опору, тощо), а також багатополюсників – складних схемних елементів. Засоби програми дозволяють користувачу побудувати електричну схему у графічному вигляді, вводити та редактувати параметри елементів схеми, зберігати та зчитувати побудовані схеми у файлах, виводити на друк, а також проводити аналіз електричних кіл таких типів:

- лінійних і нелінійних кіл постійного струму;
- лінійних кіл синусоїдального струму;
- лінійних і нелінійних динамічних кіл.

Функціональні можливості програми дозволяють використовувати її у навчальних цілях при проведенні практичних або лабораторних робіт.

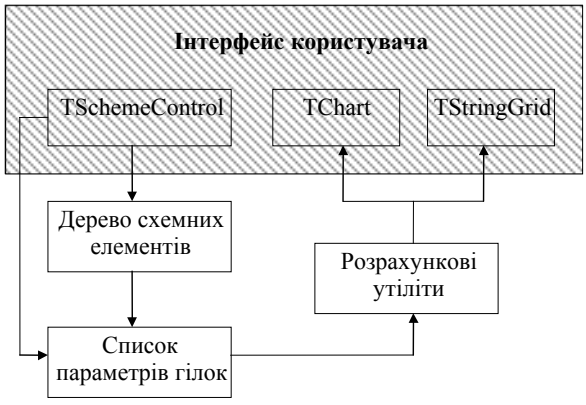

Рис. 1. Схема функціонування програми.

Складові частини системи

Спрощена схему функціонування програми Electro показана на рис.1. Інтерфейс користувача включає усі компоненти для керування системою (вікна, меню і т.д.). Окремо виділена компонента TSchemeControl, яка створена для організації візуальної побудови електричного кола. За її допомогою під час побудови схеми створюється дерево схемних елементів – ієрархічна структура даних, яка містить інформацію про топологію схеми (взаєморозташування елементів). Одночасно формується список розрахункових параметрів гілок кола. Цей список містить вхідні дані, необхідні для аналізу кола. За допомогою розрахункових утіліт (звичайних, не об'єктно-орієнтованих процедур та функцій) відбувається знаходження напруг та струмів у кожній гілці кола для заданого діапазону частот або часу. Отримані результати виводяться у графічному вигляді за допомогою стандартної компоненти Delphi TChart або у табличному вигляді (стандартна компонента TStringGrid).

Отже, можна виділити такі етапи у функціонуванні програми:

1. Формування електричного кола за допомогою TStringGrid.
2. Розрахунок (виконується за допомогою комплексу алгоритмів, описаних у [2]).
3. Виведення даних (у графічному або табличному вигляді).

Розглянемо детальніше реалізацію першого етапу, оскільки саме тут у повній мірі застосовується об'єктно-орієнтований підхід. (З основними принципами ООП та особливостями його реалізації у Delphi можна ознайомитись відповідно у роботах [3] та [4]). Електрична схема складається з окремих складових частин – схемних елементів, кожен з яких характеризується

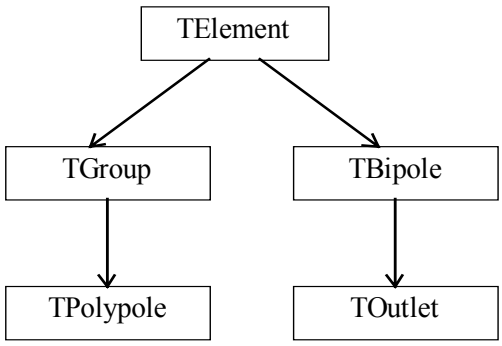

Рис. 2. Діаграма наслідування класів схемних елементів

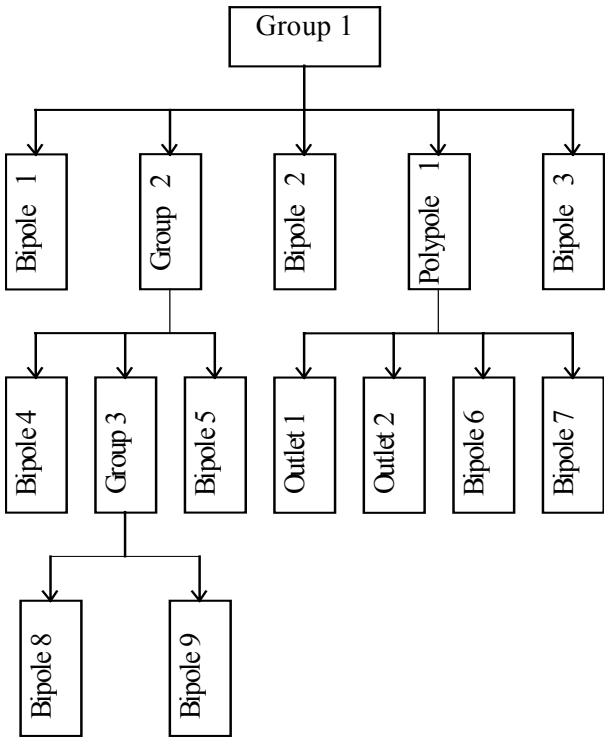

Рис. 3. Приклад дерева схемних елементів.

певними параметрами. Для візуальної побудови електричного кола за допомогою спеціалізованого редактора повинна існувати можливість переміщення, зміни розмірів та інших параметрів схемних елементів; створення груп елементів для одночасного керування ними. Отже, для кожного елемента у схемі, що проектується, створюється об'єкт – екземпляр відповідного класу. Для цього у програмі розроблена ієрархія класів схемних елементів (рис.2). При їх використанні сформована схема представляється у вигляді дерева, коренем котрого завжди буде екземпляр класу TGroup, внутрішніми вузлами – екземпляри TGroup або TPolypole, а листками – екземпляри класу TBipole або TOutlet (рис.3).

Нижче наведено короткий опис кожного з класів схемних елементів.

TElement – базовий клас схемного елемента. Він містить властивості та методи, характерні для усіх елементів. Клас TElement – абстрактний, його екземпляри ніколи не створюються. Він застосовується для утворення класів-нащадків, у яких реалізуються властивості та методи конкретних схемних елементів [5]. TElement володіє основними властивостями для відображення елемента, а також для підтримки деревоподібної структури, в яку об'єднуються елементи. Більшість методів у ньому абстрактні і повинні бути

перекриті у класах-нащадках. Оголошення їх у цьому класі необхідне лише для організації у програмі поліморфних викликів.

Клас TElement успадковується від класу TComponent. Це зроблено тому, що TComponent містить базові властивості поведінки компонент Delphi, серед яких є можливість зберігання даних компоненти у потоках введення-виведення [5]. Завдяки цьому існує можливість зберігати нащадки TElement у файлі, буфері обміну Windows та копіювати між собою, використовуючи одні й ті ж самі методи.

TBipole – клас двополюсника. На основі цього класу створюються двополюсні схемні елементи, такі як лінійні та нелінійні опір, провідність, ємність, індуктивність; ідеальні джерела струму і напруги; керовані джерела струму.

TGroup – клас, який об'єднує групу двополюсників. За допомогою цього компонента створюється можливість маніпулювати сукупністю двополюсників (а також інших груп) як єдиним цілим.

TPolypole – клас багатополюсника, нащадок класу TGroup. Керує сукупністю двополюсників та виводів, але відображається як єдиний елемент. За його допомогою створюються складні схемні елементи, такі як транзистор, операційний підсилювач, трансформатор, тощо.

TOutlet – нащадок класу TBipole. Використовується для створення елементів, які позначають вхідні й вихідні виводи (контакти) багатополюсного елемента (див. клас TPolypole). Завдяки екземплярам цього класу, відбувається сполучення багатополюсних схемних елементів з іншими

TBranch – клас, який описує сукупність розрахункових параметрів схемного елемента (такі, як величина опору або ємності, початкове значення напруги або струму у схемному елементі, тощо). Ці параметри відокремлені в окремий клас. Реалізовувати їх у межах класу TBipole було б недоцільно, оскільки існують двополюсники, які не мають розрахункових параметрів: "з'єднання" та "заземлення" (позначення нульового вузла). Замість цього двополюсник містить вказівник на об'єкт TBranch. Крім того, під час побудови схеми екземпляри класу TBranch об'єднуються у список, який потім зручно обробляти за допомогою алгоритмів розрахунку.

TSchemeControl – компонента редактора схеми. Разом з декількома стандартними компонентами Delphi вона формує у програмі графічний

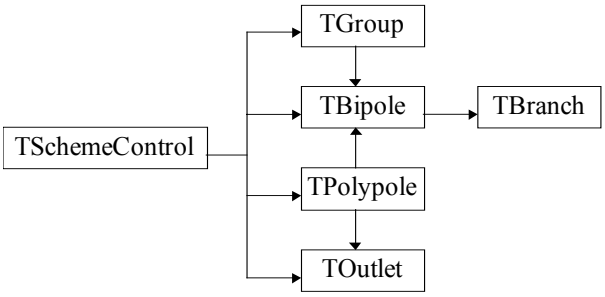

Рис. 4. Діаграма залежностей класів схемних елементів

редактор для побудови електричного кола. TschemeControl, утворена як нащадок від стандартної компоненти Delphi TGraphicControl, є візуальною, та керує екземплярами усіх вищенаведених класів схемних елементів.

Вона сприймає зовнішні події від "миші", клавіатури та викликає необхідні методи. Кінцева мета цієї компоненти – сформувати дерево схемних елементів та список розрахункових параметрів. У таблицях 1, 2 перераховані основні властивості та методи TSchemeControl. На рис.4 представлена діаграма залежностей між Tscheme Control та класами схемних елементів. Стрілками позначені виклики загальнодоступних методів одних класів іншими.

Висновки

Як підсумок, можна сформулювати основні етапи, які передбачає запропонований підхід до створення програмного забезпечення систем автоматизованого проектування:

1. Розробка ієрархії класів, кожен з яких моделює певний об'єкт заданої предметної області, а також реалізація взаємодій між цими класами.
2. Об'єднання створених екземплярів класів у деревоподібну структуру, яка передбачає можливість додавання та вилучення елементів і виконання пошуку за певною ознакою. Динамічне створення екземплярів класів сприяє раціональному використанню оперативної пам'яті ЕОМ.
3. Розробка візуальної компоненти для керування процесами створення, зміни, відображення, видалення вищезгаданих об'єктів.

Усе це дозволяє досягти високого ступеня гнучкості при розробці системи та внесенні у неї змін.

З більш докладним описом системи Electro можна ознайомитись на сайті фізичного факультету ЧДУ (http://www.phys.chsu.cv.ua).

Таблиця 1. Основні властивості компоненти TSchemeControl.

Властивість	Тип	Опис
ActiveBranch	TBranch	Гілка кола, яка відповідає обраному двополюснику
ActiveElement	TElement	Активний (обраний) схемний елемент
ActiveBipole	TBipole	Активний (обраний) двополюсник
BranchList	TBranchList	Список гілок (розрахункових параметрів) електричного кола
DynaParams	TDynaParams	Сукупність початкових умов для розрахунку динамічного кола
Locking	Boolean	Вказує, чи дозволено змінювати схему
Modified	Boolean	Вказує, чи була схема змінена після останнього збереження у файлі
SchemeMode	TSchemeMode	Тип електричного кола, що будується: лінійне, синусоїдальне або динамічне
SchemeTree	TGroup	Корінь дерева схемних елементів
SinusParams	TSinusParams	Сукупність початкових умов для розрахунку синусоїдального кола
Tool	TDrawType	Вказує тип двополюсника, який створюватиметься
AllowMoving	Boolean	Вказує, чи дозволено переміщення схемних елементів
GridSize	Byte	Розмір допоміжної сітки
ShowBrunchSymbol	Boolean	Вказує, чи відображати позначення схемних елементів
ShowGrid	Boolean	Вказує, чи відображати допоміжну сітку
SnapToGrid	Boolean	Вказує, чи прив'язувати схемні елементи до сітки
PageHeight	Integer	Висота сторінки у міліметрах (для виводу на друк)
PageWidth	Integer	Ширина сторінки у міліметрах (для виводу на друк)

Таблиця 2. Основні методи компоненти TSchemeControl.

Метод	Опис
Clear	Знищення усіх схемних елементів
CopyToClipboard	Копіювання фрагмента схеми у буфер обміну
CutToClipboard	Переміщення фрагмента схеми у буфер обміну
CreateKnotList	Створення списку вузлів схеми (необхідно перед початком розрахунку)
Grouping	Утворення групи з обраних схемних елементів
HasSelection	Вказує, чи є у схемі хоча б один виділений елемент
Import	Копіює у схему заданий схемний елемент
KeyDown	Виконується при натиску клавіш на клавіатурі
PasteFromClipboard	Вставка фрагмента схеми з буфера обміну
Print	Вивід схеми на друк
ReadFromFile	Зчитування схеми з файлу
RemoveElements	Знищення виділених схемних елементів
SelectAll	Виділення усіх схемних елементів
UnGrouping	Розформувати виділені групи елементів
WriteToFile	Зберегти схему у бінарному файлі
WriteToTextFile	Зберегти схему у текстовому файлі

СПИСОК ЛІТЕРАТУРИ

1. *Шлеер С.,Меллор С.* Объектно-ориентированный анализ: моделирование мира в состояниях. - Киев: Диалектика, 1993.
2. *Нерретер В.* Расчет электрических цепей на персональной ЭВМ. - М.: Энергоатомиздат, 1991.
3. *Буч Г.* Объектно-ориентированное проектирование с примерами применения. - Киев: Диалектика, 1992.
4. *Калверт Ч.* Delphi2. Энциклопедия пользователя. - Киев: НИПФ "ДиаСофт Лтд.", 1996.
5. *Сван Т.* Секреты 32-разрядного программирования в Delphi. - Киев: Диалектика, 1997.