Кирик Т.А.

Візуальні мови програмування

Сьогодні відома досить велика кількість вдалих інструментальних засобів візуалізації програмування. Насамперед це стосується візуальних засобів розробки екранних форм, меню й інших елементів програми (Visual BASIC, Delphi, Visual C++, Builder C++ і т.д.), засобів автоматизації проектування програмного забезпечення (CASE-засобів), засобів швидкої розробки додатків для інформаційних систем (Visual FoxPro), текстових і графічних редакторів, видавничих систем і т.д.

Існують різні тлумачення терміну "візуальні мови програмування". Зазвичай, коли мова заходить про візуальне програмування, то насамперед під цим розуміють програмування в середовищах типу Delphi чи Visual Basic. Тим часом, подібні середовища використовують візуальну технологію проектування, а код записується за допомогою текстових мов програмування. Візуальні мови програмування (ВМП) поділяють на три категорії:

- Для опрацювання візуальної інформації;
- Для підтримки візуальної взаємодії з користувачем;
- Для програмування за допомогою візуальних виразів(конструкцій).

Призначення мов першої групи, як правило, керування графічними об’єктами у засобах моделювання, САПР, графічних редакторах. Прикладами можуть служити AutoCAD, 3D Studio MAX, Corel Draw. У другу групу потрапляють середовища програмування, які підтримують візуальний дизайн інтерфейсу користувача. Візуальне програмування у такому контексті надає засобів зображення об’єктів до початку виконання самої програми. Можливість бачити об’єкти у вигляді, який вони будуть мати під час виконання програми, знімає необхідність кодування багатьох операцій вручну, що характерно для середовища, якому не притаманні засоби візуалізації. Прикладами таких систем можуть служити середовища програмування, що підтримують можливості візуального дизайну прототипів форм введення - виведення інформації. Це вже названі вище середовища Delphi, C++ Builder, Visual C++, Visual Basic, Visual J. Також у цю групу попадають мови, які призначені для опрацюванняоб'єктів, що не є візуальними споконвічно, а подаються в такому виді для зручності користувачів. Як правило, це розширення існуючих текстових мов програмування, таких як Lisp, Ada, Pascal. Такі мови називають мовами , що підтримують візуальну взаємодію.

Третя група мов поєднує власне візуальні мови програмування. Візуальною мовою програмування [1] називають сукупність візуальних речень, побудованих з іконок, наданих системою, за визначеними семантичними і синтаксичних правилами. При аналізі просторового розташування таких іконок виявляється синтаксична структура, що лежить у їх основі . При семантичному аналізі візуальне речення інтерпретується для визначення його змісту. Перевагою візуальної мови програмування у такому контексті є наочне подання даних і структур програм, що розробляються, а також надана користувачам можливість працювати в термінах своїх уявлень. У силу специфічних обмежень використання такої мови не завжди може принести відчутну користь. Однією з областей, де користь від їхнього використання безсумнівна, є навчання, зокрема, навчання алгоритмізації й основам програмування.

Зокрема, цікавим прикладом навчальної оболонки є система Blue J, розроблена компанією Sun MicroSystems. На початку розробки проект Blue J базувався на J2SDK v1.2/1.3 та був призначений для навчання студентів основам об'єктно – орієнтованого програмування і програмуванню мовою Java. Проект Blue J мав на меті розв'язання кількох проблем. Часто під візуалізацією розробки розуміють можливість розміщення на екрані тих чи інших об'єктів на етапі проектування задачі. Але жодне із середовищ візуальної розробки не дозволяє побачити ієрархію об'єктів у графічному вигляді з відображенням зв'язків між об'єктами. Крім того, як правило, велика кількість інструментальних засобів у професійних середовищах відволікає увагу від вивчення об'єктно – орієнтованого програмування. Тобто у результаті очікуване мислення категоріями ООП замінюється мисленням, що характерне традиційному процедурному програмуванню.

Вікно менеджера проекту оболонки Blue J містить:

- панель інструментів, на якій розміщені кнопки для створення нового класу, визначення зв'язків між класами, компіляції класу;
- головне вікно, у якому розміщений граф класів;
- панель об'єктів, яка містить створені екземпляри класу.

Вікно менеджера містить тільки найбільш необхідні елементи. Класи проекту зображуються графічному вигляді прямокутників. Система дозволяє слідкувати за виконаними над класами проекту діями. Так, колір і стиль зображення вказують на те, чи клас модифікований, скомпільований, чи відбувається компіляція класу. Оболонка має у своєму складі також текстовий редактор і відлагоджувач. Враховуючи вільне розповсюдження проекту і можливість встановлення середовища на платформах різних операційних систем(Windows 2000, Linux, Solaris), Blue J може цілком скласти

конкуренцію багатьом сучасним інтегрованим середовищам розробки. На платформі Linux засобами оболонки Blue J створюються повністю легальні програми.

Класи та відношення між ними можна визначати за допомогою CASE – засобів. При реалізації складних проектів системним аналітикам і менеджерам проекту прийде на допомогу об'єктно – орієнтований засіб проектування Rational Rose. Rational Rose є одним з прикладів реалізації CASE – технології. CASE – технологія, про яку вже говорилося раніше, є сукупністю методологій аналізу, проектування, розробки і супроводження складних систем програмного забезпечення. ЇЇ основне призначення – це проектування програми, абстраговано від її реалізації у коді. Така технологія дозволяю розділити процес створення програми на етапи проектування, кодування, тестування, впровадження. Для опису діаграм проекту використовується мова UML(Unified Modeling Language). У Rational Rose модель проекту подається у кількох виглядах(View). Основними з них є

- *вигляд використання*(Use Case View), який описує як проект виглядає з точку зору його використання(хто і куди вводить дані, які дії після цього виконуються програмою, хто отримує результат);
- *вигляд логіки*(Logical View), у якій описується логіка(поведінка) програми;
- *вигляд компонент*(Component View) вказує на компоненти, з яких складається проект, на їх змістовну частину, переходи від одного модуля до іншого;
- *вигляд пристроїв*(Deployment View) допомагає при проектуванні розміщення фізичних пристроїв та зв'язків між ними.

Кожен вигляд по замовчуванню має головну діаграму *Main*, у якій відображена відповідна точка зору моделі. У виглядах можна створювати додаткові діаграми: діаграму станів і переходів між ними, діаграму стану системи та її об'єктів, діаграму взаємодії об'єктів.

Після проектування класів на діаграмах, Rational Rose дозволяє прив'язати їх до конкретної мови програмування.(створити відповідний код). Rational Rose підтримує роботу з мовами C++, Ada, Java/J2ET, Visual C++, Visual Basic, XML; має надбудову для платформ Windows, Unix(Solaris, Linux).

Серед CASE – засобів виділяють такі, що підтримують UML – методологію я у чистому вигляді(Visual UML, Rational Rose) та у сукупності(комбінаціях) з іншими(Real, ModelMaker, WithClass99 та ін.).