УДК 681.3.06

КОНЦЕПЦІЯ КУРСУ ПОДІЄ-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ В СИСТЕМІ X WINDOW

Поліщук О.П., Семеріков С.О.

(Криворізький державний педагогічний університет)

В статті розглядається концепцію курсу подіє-орієнтованого програмування в системі X Window у вищій школі.

Будь-яка комп'ютерна програма виконується у такий спосіб: після розміщення в пам'яті глобальних змінних і конструювання глобальних об'єктів класів починається виконання команд, що містяться в головній підпрограмі. Як правило, спочатку виконується ряд ініціалізаціонних дій – розбір параметрів командного рядка, формування інтерфейсу користувача тощо. Після виконання таких робіт можливі два принципово різних методи роботи програми – або вона виконується далі послідовно оператор за оператором (команда за командою), викликаючи в заданій послідовності необхідні підпрограми, аж до завершення, або після виконання ініціалізаціонних дій переходить у режим очікування подій, що вимагають виконання визначених дій, і після того, як подія відбудеться, викликає на виконання підпрограму – обробник цієї події, продовжуючи одночасно аналізувати надходження інших подій і т.д. у нескінченному циклі, поки не надійде подія, що вимагає залишити цикл і завершити програму. При цьому момент настання тієї чи іншої події і їхня послідовність не регламентовані, але події можуть бути ранговані по пріоритетах їхнього обслуговування і більш пріоритетна подія може перервати обслуговування менш пріоритетної на час свого обслуговування. Перервана менш пріоритетна подія (стосовно, тієї, що обслуговується в поточний момент) повинна стати в чергу на обслуговування і її обробник виконається, коли надійде його черга. Методи складання такого типу програм ми і називаємо *подіє-орієнтованим програмуванням* [1].

Кожна прикладна програма може реагувати одну й ту саму подію (наприклад, натискання комбінації клавіш чи кнопок миші) по різному; крім того, вона повинна мати можливість сама імітувати події для виконання відповідних дій (для виклику потрібного обробника події), тому одна з основних задач операційної системи полягає в перетворенні сигналу, що надійшов, в інформаційне повідомлення (це, як правило, структура з необхідним набором полів), визначення адресата – обробника події і виклик цього обробника з передачею йому в якості одного з аргументів повідомлення, що відповідає події. Щоб операційна система могла виконати виклик необхідного обробника, структура прикладної програми повинна відповідати вимогам тієї операційної системи, під керуванням якої вона буде виконуватися.

Одним з найбільш яскравих прикладів реалізації подіє-орієнтованого програмування є задача побудови інтерфейсу користувача, елементи якого взаємодіють один з одним через механізм повідомлень. Набір функцій (класів) для створення інтерфейсу утворює бібліотеку. В операційній системі MS-DOS прикладами таких бібліотек є Turbo Vision та Graphics Vision (перша створює текстовий інтерфейс, друга – графічний). В операційних системах сімейства Windows є власні механізми обробки подій, об’єднані в програмний інтерфейс Win32 (Win64), над яким надбудовуються об’єктно-орієнтовані бібліотеки (OWL, MFC, VCL). На жаль, застосування цих бібліотек в процесі навчання подіє-орієнтованого програмування сьогодні утруднене, адже всі вони (так само, як і згадані операційні системи) вимагають значних ліцензійних виплат фірмам-розробникам, що для більшості навчальних закладів України є непідйомним тягарем.

В зв’язку з вищевикладеним виникає актуальна проблема: *створення курсу подіє-орієнтованого програмування графічного інтерфейсу користувача на основі операційних систем та бібліотек, що не вимагають ліцензійних відшкодувань*.

Для розв’язання поставленої проблеми було необхідно, перш за все,

обрати операційну систему. На думку М.І. Жалдака, найбільш привабливою з вільно поширюваних операційних систем є Linux – UNIX-подібна операційна система. UNIX – класична вузівська операційна система, яка за більш ніж тридцять років свого існування пройшла у своєму розвитку кілька стадій, і в даний час представляє, мабуть, найбільш розвинену, але разом з тим просту й елегантну операційну систему. У UNIX є усі: паралельне виконання багатьох програм, одночасна робота декількох користувачів, віртуальна пам'ять, підтримка великої кількості зовнішніх пристроїв і мереж, розвинені засоби обробки текстів, потужні інструментальні засоби для створення програмного забезпечення. Система працює в усьому світі на мільйонах комп'ютерів різних типів [2].

Графічні інтерфейси UNIX мають давню історію. Стандартом стала розподілена система X Window, що дозволяє малювати на екрані дисплея графічні зображення, підтримує концепцію вікон і уніфікує роботу з різними пристроями введення-виведення на основі бібліотеки Xlib [3]. Для того, щоб полегшити програмування з застосуванням Xlib і спростити створення інтерфейсів користувача, існує кілька пакетів, з яких найбільш широко поширені X Toolkit Intrinsics, Athena і Motif [4]. В останні роки з'явилися два нових пакети: GTK+ і Qt, що покладені в основу популярних графічних інтерфейсів GNOME і KDE.

X Window (чи просто X) – це система для створення графічного інтерфейсу користувача на комп'ютерах, що працюють під керуванням операційної системи UNIX. X була створена в Масачусетському Технологічному Інституті (США).

Особливістю системи є те, що вона підтримує роботу як на окремій ЕОМ, так і в мережі. Це означає, що програма, яка «живе» на одному комп'ютері, може за допомогою X Window взаємодіяти з користувачем, що працює за іншою машиною: система забезпечує виведення графічної інформації на екран його машини, сприймає сигнали від зовнішніх пристроїв, таких як клавіатура і

миша, і передає їх програмам.

X дозволяє користувачу спілкуватися з багатьма програмами одночасно. Щоб вивід з них не змішувався, система створює на екрані дисплея «віртуальні» підекрани – *вікна*. Кожна програма, як правило, малює лише у своєму вікні чи вікнах. X надає набір засобів для створення вікон, їхнього переміщення по екрану і зміни їхніх розмірів.

Особливістю X Window є те, що вона організує спілкування між самими програмами і між програмами і зовнішнім середовищем шляхом розсилання подій. *Подія* – це одиниця інформації, що ідентифікує дії, які відбуваються в системі, і містить додаткові дані про них.

Система X Window представляє сукупність програм і бібліотек. Серцем її є окремий UNIX-процес, що існує на комп'ютері, до якого приєднаний дисплей. Саме *сервер* знає особливості конкретної апаратури, знає, що треба зробити, щоб зафарбувати піксель на екрані, намалювати лінію чи інший графічний об'єкт. Він також вміє сприймати сигнали, що приходять від клавіатури і миші.

Сервер спілкується з програмами-клієнтами, посилаючи чи приймаючи від них порції (пакети) даних. Якщо сервер і клієнт працюють на різних машинах, то дані посилаються по мережі, якщо ж комп'ютер один, то для передачі даних використовується внутрішній канал. Наприклад, якщо сервер виявляє, що натиснуто кнопку миші, то він готує відповідний пакет і посилає його тому клієнту, у чиїм вікні знаходиться курсор миші. І навпаки, якщо програмі треба що-небудь вивести на екран дисплея, то вона створює необхідний пакет даних і надсилає його серверу.

Склад пакетів і їхня послідовність визначаються спеціальним протоколом. Але щоб програмувати для X, зовсім не обов'язково знати деталі реалізації сервера і протоколу обміну. Система надає бібліотеку процедур Xlib, за допомогою яких програми здійснюють доступ до послуг X на високому рівні.

На жаль, кількість як перекладених, так і оригінальних видань,

присвячених подіє-орієнтованому програмуванню в X Window з використанням Xlib, дуже мала, що не в останню чергу зумовлено комерційною орієнтацією на застосування засобів швидкої розробки програм (Kylix, KDevelop, Eclipse) та відповідних об'єктно-орієнтованих бібліотек візуальних компонентів. Застосування таких засобів спрощує процес програмування, проте ховає саму основу подіє-орієнтованого програмування – безпосередню обробку подій, що негативно вливає на рівень розуміння студентами відповідних механізмів та суттєво звужує коло розв'язуваних ними задач.

Це змусило авторів вдатися до розробки навчального посібника та лабораторного практикуму з програмування в X Window, побудованого за принципом поступового зростання рівня абстракцій [5]. В першому розділі посібника описується X протокол та основи Xlib – віконна структура, вивід тексту та графіки, робота з зовнішніми пристроями та програмними ресурсами, засоби міжклієнтської взаємодії. В другому розділі розглядається бібліотека Xt як найпростіший засіб автоматизації обробки повідомлень. Третій розділ присвячено бібліотеці найпростіших віконних елементів Athena, що дає можливість працювати з абстракціями "кнопка", "список", "меню" тощо. В четвертому розділі розглядається найкраща процедурна бібліотека для програмування візуального інтерфейсу програми – Motif. Останні два розділи поки що знаходяться у стадії розробки і будуть містити основи програмування в GTK+ та Qt.

Апробація курсу подіє-орієнтованого програмування в Криворізькому державному педагогічному університеті дозволила зробити такі висновки:

1. Застосування операційної системи Linux та її графічної підсистеми X Window дозволяє організувати процес навчання на основі локалізованого, безоплатного, ліцензійно чистого програмного забезпечення.

2. З метою кращого розуміння студентами основ подіє-орієнтованого програмування вивчення методів побудови інтерфейсу користувача доцільно починати не з об'єктно-орієнтованих бібліотек візуальних компонентів та

засобів швидкої розробки додатків, а з розгляду базових механізмів обробки повідомлень та клієнт-серверної взаємодії на рівні X протоколу.

3. Процедурна природа Xlib, Xt, Athena та Motif дозволяє розпочати опанування подіє-орієнтованого програмування користувацького інтерфейсу одразу після вивчення мови процедурного програмування C, що дає додаткову мотивацію студентам, роблячи їх програми більш наочними без суттєвого ускладнення структури та необхідності раннього переходу до об'єктно-орієнтованої технології. В той же час об'єктно-орієнтована бібліотека Qt (стандарт де-факто в програмуванні графічного інтерфейсу вільно поширюваних UNIX-систем) може бути використана в якості ілюстрації в процесі вивчення програмування мовою C++.

4. Послідовне опанування засобів бібліотек Xlib, Xt, Athena, Motif та Qt дозволяє організувати процес навчання за принципом поступового зростання рівня абстракцій з поверненням на кожному новому рівні до засвоєного на попередньому матеріалу.

Література:

1. Полищук А.П., Семериков С.А. Событийно-ориентированное программирование. – Кривой Рог: КГПУ, 2001. – 336 с.
2. Робачевский А.М. Операционная система UNIX. – К.: БХВ, 2000. – 518 с.
3. Adrian Nye. Volume 1: Xlib Programming Manual, 3rd Edition. – O'Reilly & Associates, 1992. – 821 p.
4. Antony Fountain, Jeremy Huxtable, Paula Ferguson and Dan Heller. Volume 6A: Motif Programming Manual for Motif 2.1, Open Source Edition. – O'Reilly & Associates, 2001. – 975 p.
5. http://kdpu.narod.ru/ipm/index.htm

The article is devoted to methodic of teaching of event-oriented programming in X Window at high school.