Тема 1. Системне програмування. Основні поняття

План
1. Поняття про системне програмування
2. Концепції програмування
   Імперативне програмування
   Програмування на основі подій
3. Засоби створення Windows-програм

1. Поняття про системне програмування

Програмне забезпечення комп'ютера за функціональним призначенням поділяють на системне, прикладне та інструментальне.

Системні програми – програми, призначені для забезпечення працездатності та обслуговування обчислювальної системи, керування її ресурсами, підтримки діалогу з користувачем.

До системного програмного забезпечення відносять системні керуючі програми та системні обслуговуючі програми.

Керуюча програма – системна програма, що реалізує набір функцій керування, який передбачає керування ресурсами та взаємодію із зовнішнім середовищем обчислювальної системи, відновлення роботи системи після виявлення несправностей в технічних засобах. Серед керуючих програм виділяють драйвери – програми, призначені для керування апаратними компонентами комп'ютера.

Програма обслуговування (утиліта) – програма, призначена для надання послуг загального характеру користувачам та обслуговуючому персоналу обчислювальної системи.

Керуюча програма разом з набором необхідних для експлуатації системи утиліт складає операційну систему (ОС).

Крім утиліт, що входять до складу ОС можуть існувати і інші утиліти, які виконують додаткове (опційне) обслуговування.

Прикладні програми – програми, призначені для розв'язування задач у певній галузі фахової діяльності людини.

Область застосування прикладних програм надзвичайно широка: від найпростіших програм для домашнього вжитку до величезних корпоративних програмних систем. Сюди відносять текстові та графічні редактори, видавничі системи, системи бухгалтерського та складського обліку, навчальні програми, математичні пакети та багато інших.

Інструментальні програми (системи програмування) – програми призначені для розробки нових комп'ютерних програм.

Система програмування (Software Development Kit, SDK) – набір програмних засобів, який включає мову програмування, компілятор (або інтерпретатор) програм, документацію та інші засоби (бібліотеки, шаблони), і призначений для створення та переведення програм у форму, придатну для виконання.

Сьогодні все частіше виділяють третій, проміжний рівень програмного забезпечення. Його можна визначити як сукупність програм, що керують своїми вторинними ресурсами. Такий рівень найчастіше називають сервісами. До цього рівня відносять менеджери транзакцій, сервери баз даних, мережеві сервери та ін.

Системне програмування – це процес розробки системних програм (в т.ч., керуючих та обслуговуючих). Значна частина системного і практично все прикладне ПЗ пишеться на мовах високого рівня, що забезпечує скорочення витрат на їх розробку (модифікацію) та можливість перенесення. Окремі програмні модулі, до яких висуваються підвищені вимоги споживання ресурсів та швидкодії, пишуться на мовах низького рівня – асемблерах.

В контексті нашого розгляду під системним програмуванням буде розумітися програмування з використанням системних викликів ядра операційної системи, яке в багатьох випадках виявляється більш ефективним, ніж стандартні засоби популярних мов програмування.

2. Концепції програмування

Парадигма програмування – це загальний підхід до програмування, філософія, принцип побудови і функціонування програми. Існує кілька парадигм програмування, як-то: імперативне програмування, функціональне програмування, логічне програмування, об'єктно-орієнтоване програмування, аспектно-орієнтоване програмування та багато інших.

Системне програмування у Windows використовує переважно дві парадигми – імперативне програмування (структурне та процедурне) та програмування, кероване подіями.

Потік керування в імперативній програмі явно визначений: команди виконуються послідовно, одна за одною, і вказують, які обчислення слід виконати. Кожна команда змінює стан програми, представлений як множина значень змінних. Імперативне програмування тісно пов'язане з архітектурою фон Неймана. Різновидністю імперативного програмування є структурне, коли потік керування складається з кількох типових блоків.

Програмування, кероване подіями (Event-Driven Programming) передбачає, що потік керування визначається асинхронними діями (від людини або пристроїв), головна діяльність програми – це реакція на події. Джерела подій можуть бути різні, найчастіше це введення даних користувачем (за допомогою миші та клавіатури), таймери, зовнішні пристрої.

Імперативне програмування

Відносно нескладні програми зазвичай розробляються з використанням імперативного програмування та його різновидностей – структурного та процедурного.

Процедурна програма являє собою послідовність операторів та викликів підпрограм (процедур) обробки тих або інших даних.

Структурована програма складається з окремих блоків, які мають одну точку входу і одну точку виходу. Кожний блок може виконувати всього три функції: слідування, вибір, повторення. Слідування означає, що вихід першого блоку з'єднується із входом наступного. Вибір здійснюється одним із трьох способів: структурою **if** (одиничний вибір), структурою **if / else** (альтернативний вибір), структурою **switch** (множинний вибір). Повторення реалізується теж трьома способами: структурою **while** (з передумовою), структурою **do / while** (з післяумовою), структурою **for** . Структурована програма складається з окремих блоків за двома правилами:
пакетування (за одним блоком ставиться наступний) і вкладення (будь-який блок замінюється на керуючу структуру вибору або повторення). У структурованих програмах немає потреби в операторі **goto**.

Після запуску імперативна програма послідовно виконує свої оператори. Операції введення/виведення контролюються програмним кодом, тому що здійснюються всередині одного із блоків, тобто відбуваються у моменти, передбачені програмістом під час розробки. Після завершення програми керування передається ОС.

В операційній системі DOS операції введення-виведення реалізуються через програмні переривання DOS, переривання BIOS або шляхом прямого доступу до портів зовнішніх пристроїв. В операційній системі Windows імперативні програми представлені консольними додатками, у яких операції введення-виведення реалізуються за допомогою системних викликів операційної системи – функції прикладного програмного інтерфейсу (API).

Програмування на основі подій (Event-Driven Programming)

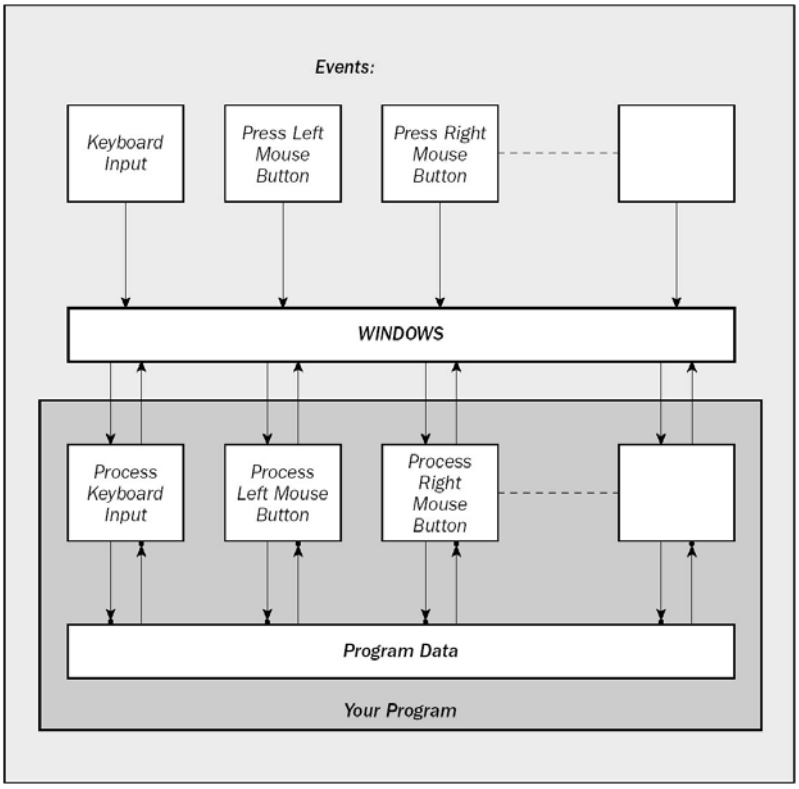
У міру зростання обсягу і складності створюваних програмних продуктів все більш привабливою виявляється парадигма, що отримала назву об'єктно-орієнтованого програмування (ООП). Об'єкти – це компоненти програмного забезпечення, які моделюють елементи реального світу та інші сутності. Їх внутрішній склад і правила поведінки визначені наперед в описах класів цих об'єктів.

Суть ООП полягає в тому, що програма являє собою систему об'єктів, що взаємодіють між собою шляхом обміну повідомленнями. Повідомлення формуються як реакція на події. Природа інтерфейсу між користувачем і програмою Windows така, що певний набір вхідних подій можливий у будь-який час. Користувач може вибирати будь-який із пунктів меню, клацнути кнопку панелі інструментів, або клацнути кнопкою миші де-небудь у вікні програми. Добре розроблена програма Windows має бути готовою, щоб мати справу з будь-яким із можливих вхідних сигналів у будь-який час, тому що неможливо знати наперед який вхідний сигнал у який час збирається надійти. Ці дії користувача в першу чергу отримує операційна система і сприймає їх як події.

Подія, яка надходить від інтерфейсу користувача для вашого додатка, зазвичай закінчується тим, що буде виконуватись певна ділянка коду. Тому послідовність виконання програми визначається послідовністю дій користувача. Про програми, які діють таким чином, говорять, як про **керовані подіями програми** (Event-Driven Programs)**,** в цьому їх головна відмінність від імперативних програм, які виконуються згідно із заданим алгоритмом.

Windows-програми

Програма Windows складається із набору процедур, які відповідають на події, викликані дією користувача, або самої Windows. Події (натиснута клавіша, натиснута ліва кнопка миші, натиснута права кнопка миші та ін..) сприймаються операційною системою. За кожною подією закріплено окрему процедуру – обробник подій (обробити натискання клавіші, обробити натискання лівої кнопки і т. д.). Операційна система визначає, кому призначається повідомлення про подію і викликає відповідний обробник. Програма виглядить так, ніби складається із відокремлених фрагментів, насправді, їх об'єднує диспетчер подій операційної системи.



Windows-програма повністю підвладна операційній системі. Вона не має справи безпосередньо з технічними засобами, а всі зв'язки із зовнішнім середовищем проходять через Windows. Після запуску Windows-програми керування одержує не сама програма, а диспетчер подій ОС. Він приймає сигнали від пристроїв введення-виведення, формує системні повідомлення, які призводять до виклику підпрограм – обробників подій. Якщо надходить повідомлення про завершення програми, диспетчер запускає процедуру деініціалізації.

ОС Windows багатозадачна, тому керування ресурсами системи і їх розподіл між програмами завжди в руках операційної системи. Коли ми використовуємо Windows-програму, ми взаємодіємо передусім із Windows, яка зв'язується з прикладною програмою і запускає певні ділянки коду. Так відбувається тому, що кожна програма потенційно поділяє комп'ютер з іншими програмами, які, можливо, виконуються в той же час.

3. Засоби створення Windows-програм

Програми для Windows можна писати на різних мовах програмування. Природно, ні внутрішні можливості Windows, ні тим більше концепції цієї системи ніяк не залежать від використовуваної мови, проте практичне керівництво по програмуванню, спосіб створення програми, склад програмних засобів повинні спиратися на певну мову програмування і навіть на певний варіант компілятора

В даний час існує цілий ряд мов програмування: C, C++, C#, Java, Python, Delphi Object Pascal, Visual Basic та багато інших. Для кожного класу задач більше підходить та чи інша мова програмування. Прикладне програмне забезпечення частіше пишуть на Object Pascal, C++, C#, Java, Visual Basic, у Web-програмуванні найбільш поширені мови PHP, JavaScript, Perl, додатки з інтенсивним використанням ресурсів, такі як графічні програми або ігри часто написані на C++ або C#.

Не дивлячись на широкий вибір сучасних мов програмування, одним із основних засобів в руках системних програмістів залишається мова C. Скоріше за все вона не буде повністю витіснена іншими мовами програмування. Якщо ви хочете розробити невелику системну програму, драйвер, керуючу програму для вбудованої системи, мова C, очевидно, буде кращим вибором. При розробці драйверів та інших критичних до швидкодії та системних ресурсів модулів використовують мову асемблера. Не дивлячись на заяви скептиків, асемблер буде існувати до тих пір, поки існують процесори. Мови C, C++, асемблер можуть безпосередньо викликати функції Windows API. Інші мови використовують спеціально розроблені бібліотеки, що викликають системні функції непрямо.

Для того, щоб створювати програми з використанням системних функцій, ми повинні мати комплект розробника програмного забезпечення (Software Development Kit або SDK). Такий комплект містить середовище розробки програм (Integrated Development Environment або IDE), компілятор, компонувальник, редактор ресурсів, відлагоджувальник та ін. Крім цього слід мати заголовкові файли, бібліотеки підпрограм, зразки програм та документацію.

Прикладами інтегрованих середовищ розробки можуть бути Microsoft Visual Studio (6.0, 2005, 2008, 2010 та ін.), Embarcadero C++ Builder, а також вільні програмні продукти Eclipse, NetBeans, Code::Blocks. Сучасні IDE – надзвичайно складні системи, вивчення їх широких можливостей може стати окремою задачею і вимагати багато часу. Тому, початківцям можна обрати одне із найпростіших (до того ж, безкоштовних) середовищ типу Dev-C++ або Relo 2.0, які, у більшості випадків, справляються із задачами системного програмування.

Сьогодні популярним є багатоплатформне (крос-платформне) програмування. Є декілька бібліотек, які надають можливість програмістам створювати програми, які працюватимуть у Windows, Linux, BSD і Mac OS. Це добре відомі бібліотеки: Qt, Swing або wxWidgets.