Лабораторна робота № 8

(4 години)

Тема: Засоби взаємодії між процесами у Windows.

Мета: Ознайомитися із засобами організації обміну даними між процесами, навчитися використовувати іменовані канали для міжпроцесної взаємодії.

Короткі теоретичні відомості

Під обміном даними між паралельними процесами (Interprocess Communication або IPC) розуміють пересилку даних від одного потоку до іншого, припускаючи, що ці потоки виконуються в контекстах різних процесів. Потік, який посилає дані іншому потоку, називається відправником. Потік, що отримує дані від іншого потоку, називається адресатом або одержувачем.

Є кілька механізмів, які використовуються для обміну даними між процесами:

1. Локальні засоби IPC.
   а) передача повідомлень (через чергу повідомлень, безпосередній обмін повідомленнями, за допомогою спеціального повідомлення **WM_COPYDATA**);
   б) використання поділюваної пам’яті – shared memory (за допомогою проекцій файлу, через поділювані засоби DLL);
   в) використання анонімних каналів.
2. Мережеві засоби IPC.
   а) іменовані канали;
   б) поштові скриньки;
   в) сокети;
   г) віддалений виклик процедур (RPC).

Розглянемо засоби обміну даними на прикладі іменованих каналів.

Іменованим каналом називається об'єкт ядра операційної системи, який забезпечує передачу даних між процесами, що виконуються на комп'ютерах в одній локальній мережі. Процес, який створює іменований канал, називається *сервером* іменованого каналу. Процеси, які зв'язуються з іменованим каналом, називаються *клієнтами* іменованого каналу.

Відзначимо характеристики іменованих каналів:

- мають ім'я, яке використовується клієнтами для зв'язку з іменованим каналом;
- можуть бути як напівдуплексними, так і дуплексними;
- передача даних може здійснюватися як потоком, так і повідомленнями;
- обмін даними може бути як синхронним, так і асинхронним;

Обмін даними між процесами через іменований канал може бути організована таким чином:

- створення іменованого каналу сервером:
- з'єднання сервера з примірником іменованого каналу;
- з'єднання клієнта з примірником іменованого каналу;
- обмін даними через іменований канал;
- від'єднання сервера від примірника іменованого каналу;
- закриття іменованого каналу клієнтом і сервером.

Схематично взаємодію клієнта і сервера з використанням іменованого каналу можна відобразити так:

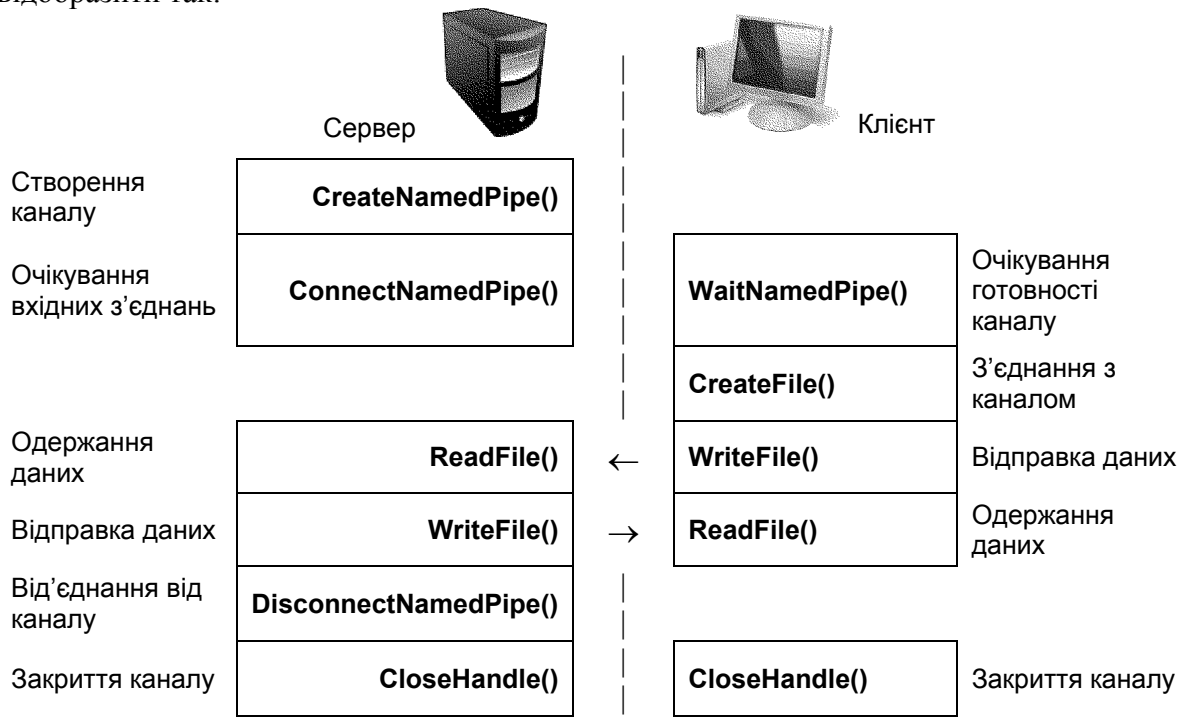

Рис. 3. Взаємодія процесів через іменований канал.

Створення іменованого каналу

HANDLE CreateNamedPipe(

```
  LPCTSTR lpName,                  // ім'я каналу
  DWORD dwOpenMode,                // режим відкриття каналу
  DWORD dwPipeMode,                // спосіб передачі даних
  DWORD nMaxInstances,             // максимальне число примірників каналу
  DWORD nOutBufferSize,            // розмір вихідного буфера в байтах
  DWORD nInBufferSize,             // розмір вхідного буфера в байтах
  DWORD nDefaultTimeOut,           // тайм-аут за умовчанням в мілісекундах
  LPSECURITY_ATTRIBUTES lpSecurityAttributes     // структура атрибутів безпеки
);
```

Параметри функції **CreateNamedPipe()**:

lpName вказує на рядок з ім'ям каналу у форматі: `\\.\pipe\pipename`

Тут "\\." позначає локальну машину, тому що іменований канал завжди створюється на локальній машині, слово `pipe` фіксоване, а `pipename` позначає ім'я каналу, яке задається користувачем і нечутливе до регістру.

dwOpenMode – режим відкриття каналу:

PIPE_ACCESS_DUPLEX	Дуплексний режим для сервера. Клієнт визначає режим доступу при виклику функції **CreateFile()**.
PIPE_ACCESS_INBOUND	Напрям передачі від клієнта до сервера. При виклику функції **CreateFile()** клієнт повинен встановити прапорець GENERIC_WRITE.
PIPE_ACCESS_OUTBOUND	Напрям передачі від сервера до клієнта. При виклику функції **CreateFile()** клієнт повинен встановити прапорець GENERIC_READ.
FILE_FLAG_WRITE_THROUGH	Забороняє буферизацію даних (наскрізний запис). Запис у канал чекає завершення передачі даних.
FILE_FLAG_OVERLAPPED	Дозволяє асинхронну передачу даних (введення-виведення із перекриттям). Функції читання-запису

повертаються негайно, не чекаючи завершення виконання операції.

dwPipeMode визначає спосіб передачі даних через канал.

PIPE_TYPE_BYTE	Запис даних потоком.
PIPE_TYPE_MESSAGE	Запис даних повідомленнями.
PIPE_READMODE_BYTE	Читання даних потоком.
PIPE_READMODE_MESSAGE	Читання даних повідомленнями.
PIPE_WAIT	Синхронний режим роботи (очікування завершення операцій **ReadFile()**, **WriteFile()** або **ConnectNamedPipe()**).
PIPE_NOWAIT	Асинхронний режим роботи (вищезазначені операції повертаються негайно).

nDefaultTimeOut визначає час очікування функції **WaitNamedPipe()** за умовчанням (якщо задано NMPWAIT_USE_DEFAULT_WAIT).

Призначення інших параметрів очевидне.

З'єднання сервера з каналом

З'єднання сервера з каналом здійснюється функцією **ConnectNamedPipe()**, яка очікує на підключення клієнта до примірника каналу.

```
BOOL ConnectNamedPipe(

  HANDLE hNamedPipe,            // дескриптор каналу
  LPOVERLAPPED lpOverlapped     // вказівник на структуру OVERLAPPED
);
```

Якщо параметр *lpOverlapped* = NULL, використовується синхронний режим роботи: функція не повертається до тих пір, поки клієнт з'єднається з каналом або станеться помилка. Якщо клієнт успішно з'єднується з каналом після виклику **ConnectNamedPipe()** – функція повертає значення TRUE. Якщо клієнт з'єднається з каналом до виклику **ConnectNamedPipe()** – функція повертає FALSE, хоча з'єднання встановлюється успішно.

З'єднання клієнта з каналом

Перед з'єднанням клієнт повинен перевірити доступність одного з примірників каналу. Для цього використовують функцію **WaitNamedPipe()**:

```
BOOL WaitNamedPipe(

  LPCTSTR lpNamedPipeName,      // вказівник на ім'я каналу
  DWORD nTimeOut                // інтервал очікування в мілісекундах
);
```

Ця функція чекає, поки з'явиться вільний примірник каналу (за умови, що серверний процес викликав функцію **ConnectNamedPipe()**) або вичерпається інтервал очікування. Параметри функції:

lpNamedPipeName ім'я каналу у форматі: `\\servername\pipe\pipename`

nTimeOut визначає час очікування на з'єднання з каналом в мілісекундах або приймає одне із значень:

NMPWAIT_USE_DEFAULT_WAIT	Тайм-аут визначається функцією сервера **CreateNamedPipe()**.
NMPWAIT_WAIT_FOREVER	Функція не повертається, поки не стане доступним примірник каналу.

Після цього клієнт може викликати функцію **CreateFile()**, якій передається ім'я іменованого каналу в одному із вищенаведених форматів. При роботі з каналом слід звернути увагу на наступні параметри:

dwDesiredAccess може бути одним із (або комбінацією) таких значень:

GENERIC_READ	дозволяє читання з каналу;
GENERIC_WRITE	дозволяє запис в канал.

dwShareMode визначає режим спільного використання каналу і дорівнює нулю для заборони спільного доступу або комбінації таких значень:

FILE_SHARE_READ	дозволяє спільне читання з каналу;
FILE_SHARE_WRITE	дозволяє спільний запис в канал.

Якщо клієнт і сервер запущені на різних комп'ютерах і параметри безпеки задаються за умовчанням, то повинні бути задані однакові імена і паролі користувачів, оскільки канал належить користувачу, який його створив. Для загального доступу до каналу потрібно відповідним чином задати атрибути безпеки.

Обмін даними через іменований канал

Для обміну даними через іменований канал сервер і клієнт зазвичай використовують функції **WriteFile()**, **ReadFile()**.

Крім цього є деякі специфічні функції для роботи з іменованим каналом. Зокрема, можна використовувати функцію **TransactNamedPipe()**, яка поєднує в одній мережевій операції функції відправки запиту і читання відповіді (фактично замінює дві функції **WriteFile()** і **ReadFile()**).

Функція **CallNamedPipe()** призначена для одноразової транзакції із використанням іменованого каналу і зручна для використання клієнтом. Вона поєднує в одній операції з'єднання з каналом, запис в канал, читання з каналу і закриття каналу. Таким чином, функція обслуговує один запит клієнта до сервера і звільняє канал для інших клієнтів.

Функція **PeekNamedPipe()** дає можливість перевірити наявність даних в каналі і визначити їх кількість.

Від'єднання сервера від каналу

Від'єднання сервера від каналу здійснюється за допомогою функції **DisconnectNamedPipe()**, яка відключає примірник іменованого каналу від клієнтського процесу.

```
BOOL DisconnectNamedPipe(
  HANDLE hNamedPipe                  // дескриптор каналу
);
```

Після цього сервер може з'єднатися з іншим клієнтським процесом, використовуючи функцію **ConnectNamedPipe()** і наявний у нього дескриптор.

Завдання для виконання

1. Написати консольну програму клієнт-сервер із використанням іменованого каналу. Сервер створює іменований канал і переходить в режим прийому запитів від клієнта. Одержаний у вигляді текстового рядка запит виводить на екран. Клієнт з'єднується з іменованим каналом на заданому хості, приймає з клавіатури текстовий запит і відправляє його серверу. Ім'я хоста задається в командному рядку при запуску клієнта.
2. Змінити програму з п. 1 таким чином, щоб сервер після прийому запиту клієнта відправляв відповідь клієнту, а клієнт виводив його на екран.
3. Змінити програму з п. 2 таким чином, щоб клієнт і сервер могли обмінюватися повідомленнями багаторазово. Після одержання кожного повідомлення клієнт (сервер) виводить його на екран і переходить в режим введення з клавіатури нового повідомлення. Робота клієнта (сервера) завершується при введенні з клавіатури команди `$quit`.

Контрольні питання

1. В чому суть механізму міжпроцесної взаємодії (IPC)?
2. Які способи міжпроцесної взаємодії ви знаєте?
3. Як відправити повідомлення іншому процесу?
4. В чому різниця між анонімним та іменованим каналом?
5. Яку роль виконують сервер та клієнт іменованого каналу?
6. Який формат назви іменованого каналу для сервера і клієнта?
7. Як створити іменований канал? Які він має режими роботи?
8. Яка послідовність обміну даними через іменований канал?
9. Що таке «сокет»? Як він ідентифікується? Що таке Winsock?
10. Як ви розумієте термін «порт» сокету?
11. Яка послідовність дій для передачі даних через сокети?
12. Які бібліотеки та функції використовують при роботі з сокетами?

Додаток. Приклад використання іменованого каналу

/* Програма-сервер для читання повідомлення з іменованого каналу */

```
#include <windows.h>
int main (void) {
HANDLE hPipe;
DWORD dwRead;
char c = -1;
   hPipe = CreateNamedPipe("\\\\.\\pipe\\hello", PIPE_ACCESS_INBOUND, PIPE_WAIT,
                           PIPE_UNLIMITED_INSTANCES, 256, 256, 1000, NULL);
   ConnectNamedPipe (hPipe, NULL);
   while (c != '\0')
   {
      ReadFile(hPipe, &c, 1, &dwRead, NULL);
      if (dwRead > 0 && c != 0) putchar(c) ;
   }
   DisconnectNamedPipe (hPipe);
   CloseHandle (hPipe);
   return 0;
}
```

/* Програма-клієнт для відправки запиту через іменований канал */

```
#include <windows.h>
#include <stdio.h>

int main (int argc, char *argv[]) {
HANDLE hPipe;
DWORD dwWritten;
char lpPipe[80];
char lpText[] = "Client Massage";
   if (argc != 2)
   {
      printf ("Usage: %s hostname\n", argv[0]);
      return 1;
   }
   sprintf (lpPipe, "\\\\%s\\pipe\\hello", argv[1]);
   WaitNamedPipe (lpPipe, NMPWAIT_WAIT_FOREVER);
   hPipe = CreateFile (lpPipe, GENERIC_WRITE, 0, NULL, OPEN_EXISTING,
                           FILE_ATTRIBUTE_NORMAL, NULL);

   WriteFile (hPipe, lpText, strlen(lpText)+1, &dwWritten, NULL);
   CloseHandle (hPipe);
   return 0;
}
```