Лабораторная работа № 7
Использование модели WSAAsyncSelect

7.1 Цель работы

Целью лабораторной работы является ознакомление студентов с возможностью организации параллельного обслуживания клиентов с использованием модели WSAAsyncSelect.

7.2 Задание на лабораторную работу

В лабораторной работе необходимо разработать сервер, использующий модель WSAAsyncSelect. Сервер должен принимать или отправлять файлы по запросу клиентов.

7.3 Методические указания

7.3.1 Системная очередь сообщений

Большинство сообщений создают драйверы устройств ввода/вывода, таких, как клавиатура, мышь, таймер или сетевая карта. Драйверы создают сообщения при поступлении аппаратных прерываний. Например, когда вы нажимаете и затем отпускаете клавишу, драйвер обрабатывает прерывания от клавиатуры и создает несколько сообщений. Аналогично сообщения создаются при перемещении мыши или в том случае, когда вы нажимаете кнопки на корпусе мыши. Можно сказать, что драйверы устройств ввода/вывода транслируют аппаратные прерывания в сообщения.

Созданные драйверами сообщения, прежде всего, попадают в системную очередь сообщений Windows. Системная очередь сообщений одна. Далее из нее сообщения распределяются в очереди сообщений приложений. Для каждого приложения создается своя очередь сообщений (см. рисунок 7.1).

Рисунок 7.1 Системная очередь сообщений

7.3.2 WSAAsyncSelect

Функция select введена в библиотеку WinSock для совместимости с аналогичными библиотеками других платформ. Для программирования в Windows более мощной является функция **WSAAsyncSelect**, которая позволяет отслеживать состояние сокетов с помощью сообщений Windows. Таким образом, нет необходимости замораживать работу программы для ожидания доступности сокетов.

Функция выглядит следующим образом:

```
int WSAAsyncSelect(
                SOCKET s,
                HWND hWnd,
                unsigned int wMsg,
                long lEvent
                    );
```

Рассмотрим каждый параметр:

- ***s*** - сокет, события которого необходимо ловить;
- ***hWnd*** - окно, которому будут посылаться события при возникновении сетевых сообщений.

- ***wMsg*** - сообщение, которое будет отсылаться окну. По его типу можно определить, что это событие сети;
- ***lEvents*** - битовая маска сетевых событий, которые нас интересуют. Этот параметр может принимать любую комбинацию из следующих значений:
    1) **FD_READ** — готовность к чтению;
    2) **FD_WRITE** — готовность к записи;
    3) **FD_OOB** — получение срочных данных;
    4) **FD_ACCEPT** — подключение клиентов;
    5) **FD_CONNECT** — соединение с сервером;
    6) **FD_CLOSE** — закрытие соединения;
    7) **FD_QOS** — изменения сервиса QoS (Quality of Service);
    8) **FD_GROUP_QOS** — изменение группы QoS.

Если функция отработала успешно, то она вернет значение больше нуля, если произошла ошибка — **SOCKET_ERROR**.

Функция автоматически переводит сокет в неблокирующий режим, и нет смысла вызывать функцию ioctlsocket.

Вот простой пример использования WSAAsyncSelect:

```
WSAAsyncSelect(s, hWnd, wMsg, FD_READ|FD_WRITE);
```

После выполнения этой строчки кода окно **hWnd** будет получать событие **wMsg** каждый раз, когда сокет **s** будет готов принимать и отправлять данные. Чтобы отменить работу события, необходимо вызвать эту же функцию, но в качестве четвертого параметра указать 0:

```
WSAAsyncSelect(s, hWnd, 0, 0);
```

В данном случае необходимо правильно указать первые два параметра и обнулить последний. Содержимое третьего параметра не имеет значения, потому что событие не будет отправляться, и можно указать ноль. Если вам нужно просто изменить типы событий, то можете вызвать функцию с новыми значениями четвертого параметра. Нет смысла сначала обнулять, а потом устанавливать заново.

Для каждого сокета можно назначить только одно сообщение на разные события. Это означает, что нельзя по событию FD_READ окну посылать одно сообщение, а по FD_WRITE — другое.

В функцию **WndProc** при возникновении события передаются параметры **wParam** и **lParam**, содержащие вспомогательную информацию(в

зависимомти от события). Для событий сети в параметре wParam хранится дескриптор сокета, на котором произошло событие. Таким образом, не надо хранить массив созданных сокетов, их всегда можно получить в событии. Параметр lParam состоит из двух слов: младшее определяет событие, а старшее — код ошибки.

Использование сообщений Windows очень удобно, но вы теряете совместимость с UNIX-системами, где сообщения реализованы по-другому и нет функции WSAAsyncSelect. Поэтому при переносе такой программы на другую платформу возникнут большие проблемы, и придется переписать слишком много кода. Но если перенос не планируется, то применение WSAAsyncSelect позволяет добиться максимальной производительности и удобства программирования.

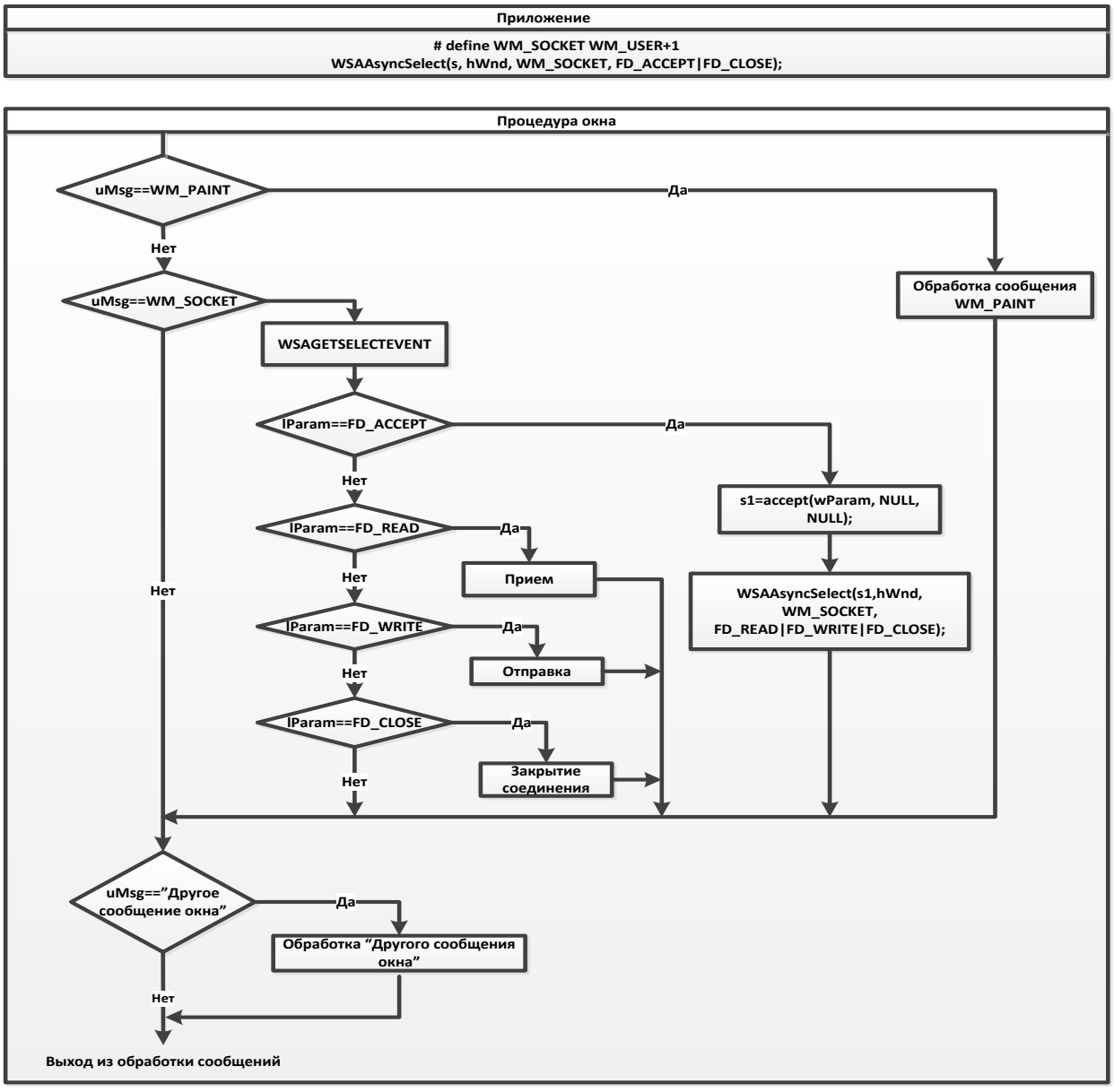

Рисунок 7.2 Типичный алгоритм применения модели WSAAsyncSelect