IPC – inter process communication. Механизм межпроцессного взаимодействия при использовании разделенного адресного пространства.

1. Через **сигналы** – программный механизм, аналогичный прерыванию(без использования приоритетов), который информирует процесс о наступлении синхронного события.
   Минус – скорость выполнения. Плюс – зарезервированные константы.
2. Через **разделяемые файлы**.
3. Через **каналы** – именованная (неименованная) очередь сообщения по принципу FIFO. Записывает один поток - считывает другой.

способности и задержки взаимодействия между потоками и типа передаваемых данных.

Буфер обмена (clipboard)

Это одна из самых примитивных и хорошо известных форм IPC. Он появился еще в самых ранних версиях Windows. Основная его задача - обеспечивать обмен данными между программами по желанию и под контролем пользователя.

Сообщение WM_COPYDATA

Стандартное сообщение для передачи участка памяти другому процессу. Работает однонаправленно, принимающий процесс должен расценивать полученные данные как read only. Посылать это сообщение необходимо только с помощью SendMessage, которая (напомню) в отличие от PostMessage ждет завершения операции. Таким образом, посылающий поток "подвисает" на время передачи данных. Вы сами должны решить, насколько это приемлемо для вас. Это не имеет значения для небольших кусков данных, но для больших объемов данных или для real-time приложений этот способ вряд ли подходит.

Разделяемая память (shared memory)

Этот способ взаимодействия реализуется не совсем напрямую, а через технологию File Mapping - отображения файлов на оперативную память. Вкратце, этот механизм позволяет осуществлять доступ к файлу таким образом, как будто это обыкновенный массив, хранящийся в памяти (не загружая файл в память явно). "Побочным эффектом" этой технологии является возможность работать с таким отображенным файлом сразу нескольким процессам. Таким образом, можно создать объект file mapping, но не ассоциировать его с каким-то конкретным файлом. Получаемая область памяти как раз и будет общей между процессами. Работая с этой памятью, потоки обязательно должны согласовывать свои действия с помощью объектов синхронизации.

Библиотеки динамической компоновки (DLL)

Библиотеки динамической компоновки также имеют способность обеспечивать обмен данными между процессами. Когда в рамках DLL объявляется переменная, ее можно сделать разделяемой (shared). Все процессы, обращающиеся к библиотеке, для таких переменных будут использовать одно и то же место в физической памяти. (Здесь также важно не забыть о синхронизации.)

Протокол динамического обмена данными (Dynamic Data Exchange, DDE)

Этот протокол выполняет все основные функции для обмена данными между приложениями. Он очень широко использовался до тех пор, пока для этих целей не стали применять OLE (впоследствии ActiveX). На данный момент DDE используется достаточно редко, в основном для обратной совместимости. Больше всего этот протокол подходит для задач, не требующих продолжительного взаимодействия с пользователем. Пользователю в некоторых случаях нужно только установить соединение между программами, а обмен данными происходит без его участия. Замечу, что все это в равной степени относится и к технологии OLE/ActiveX.

OLE/ActiveX

Это действительно универсальная технология, и одно из многих ее применений - межпроцессный обмен данными. Хотя стоит думаю отметить, что OLE как раз для этой цели и создавалась (на смену DDE), и только потом была расширена настолько, что пришлось поменять название ;-). Специально для обмена данными существует интерфейс IDataObject. А

для обмена данными по сети используется DCOM, которую под некоторым углом можно рассматривать как объединение ActiveX и RPC.

Каналы (pipes)

Каналы - это очень мощная технология обмена данными. Наверное, именно поэтому в полной мере они поддерживаются только в Windows NT/2000. В общем случае канал можно представить в виде трубы, соединяющей два процесса. Что попадает в трубу на одном конце, мгновенно появляется на другом. Чаще всего каналы используются для передачи непрерывного потока данных. Каналы делятся на анонимные (anonymous pipes) и именованные (named pipes). Анонимные каналы используются достаточно редко, они просто передают поток вывода одного процесса на поток ввода другого. Именованные каналы передают произвольные данные и могут работать через сеть. (Именованные каналы поддерживаются только в WinNT/2000.)

Сокеты (sockets)

Это очень важная технология, т.к. именно она отвечает за обмен данными в Интернет. Сокеты также часто используются в крупных ЛВС. Взаимодействие происходит через т.н. разъемы-"сокеты", которые представляют собой абстракцию конечных точек коммуникационной линии, соединяющей два приложения. С этими объектами программа и должна работать, например, ждать соединения, посылать данные и т.д. В Windows входит достаточно мощный API для работы с сокетами.

Почтовые слоты (mailslots)

Почтовые слоты - это механизм однонаправленного IPC. Если приложению известно имя слота, оно может помещать туда сообщения, а приложение-хозяин этого слота (приемник) может их оттуда извлекать и соответствующим образом обрабатывать. Основное преимущество этого способа - возможность передавать сообщения по локальной сети сразу нескольким компьютерам за одну операцию. Для этого приложения-приемники создают почтовые слоты с одним и тем же именем. Когда в дальнейшем какое-либо приложение помещает сообщение в этот слот, приложения-приемники получают его одновременно.

Объекты синхронизации

Как ни странно, объекты синхронизации тоже можно отнести к механизмам IPC.

Microsoft Message Queue (MSMQ)

Этот протокол действительно оправдывает свое название - он обеспечивает посылку сообщений между приложениями с помощью очереди сообщений. Основное его отличие от стандартной очереди сообщений Windows в том, что он может работать с удаленными процессами и даже с процессами, которые на данный момент недоступны (например, не запущены). Доставка сообщения по адресу гарантируется. Оно ставится в специальную очередь сообщений и находится там до тех пор, пока не появляется возможность его доставить.

Удаленный вызов процедур (Remote Procedure Call, RPC)

Строго говоря, это не совсем технология IPC, а скорее способ значительно расширить возможности других механизмов IPC. С помощью этой технологии общение через сеть становится совешенно прозрачным как для сервера, так и для клиента. Им обоим начинает казаться, что их "собеседник" расположен локально по отношению к ним .