Inter-Process Communication

Application Level

Finally we look briefly at IPC at the application level, namely high level mechanisms for passing data between processes

Inter-Process Communication

Application Level

Finally we look briefly at IPC at the application level, namely high level mechanisms for passing data between processes

Again, at base, this goes via the kernel (often using a mechanism we have already mentioned, e.g., pipes or shared memory, assisted by signals or semaphores), but the idea is to provide high level constructs so we (as programmers) don't have to be bothered with details

Inter-Process Communication

Application Level

Finally we look briefly at IPC at the application level, namely high level mechanisms for passing data between processes

Again, at base, this goes via the kernel (often using a mechanism we have already mentioned, e.g., pipes or shared memory, assisted by signals or semaphores), but the idea is to provide high level constructs so we (as programmers) don't have to be bothered with details

These are always implemented by system libraries and a fixed interface presented to the programmer regardless of the underlying implementation

Inter-Process Communication

Application Level

These came back into prominence with windowing GUIs where it was found necessary for applications to communicate with each other and with the system

Inter-Process Communication

Application Level

These came back into prominence with windowing GUIs where it was found necessary for applications to communicate with each other and with the system

Cut-and-paste and drag-and-drop are basic examples, where structured information needs to pass between components (processes)

Inter-Process Communication

Application Level

These came back into prominence with windowing GUIs where it was found necessary for applications to communicate with each other and with the system

Cut-and-paste and drag-and-drop are basic examples, where structured information needs to pass between components (processes)

The idea is much older than GUIs, of course: originally this was called a *software bus* in analogy with hardware buses that connect hardware components

Inter-Process Communication

Application Level

Popular implementations include

Inter-Process Communication

Application Level

Popular implementations include

- CORBA (Common Object Request Broker Architecture)
- DCOP (Desktop COmmunication Protocol)
- Bonobo (based on CORBA)
- D-Bus
- COM (Component Object Model) and variants, including .NET

Inter-Process Communication

Application Level

These try to be language independent with each language having a set of *bindings* (standard functions) to access them

Inter-Process Communication

Application Level

These try to be language independent with each language having a set of *bindings* (standard functions) to access them

They focus on passing objects between components

Inter-Process Communication

Application Level

These try to be language independent with each language having a set of *bindings* (standard functions) to access them

They focus on passing objects between components

So they need a standardised method of representing the data/objects in a *message*

Inter-Process Communication

Application Level

These try to be language independent with each language having a set of *bindings* (standard functions) to access them

They focus on passing objects between components

So they need a standardised method of representing the data/objects in a *message*

This is called *message passing*, another important paradigm for IPC

Inter-Process Communication

Application Level

These kinds of framework tend to be very complicated as they need to support a wide variety of communications between a wide variety of components

Inter-Process Communication

Application Level

These kinds of framework tend to be very complicated as they need to support a wide variety of communications between a wide variety of components

For example, passing a picture from a program written in C to one written in Java

Inter-Process Communication

Application Level

These kinds of framework tend to be very complicated as they need to support a wide variety of communications between a wide variety of components

For example, passing a picture from a program written in C to one written in Java

Exercise. Read up on some of these

Inter-Process Communication

So there are many IPC mechanisms: they are not mutually exclusive

Inter-Process Communication

So there are many IPC mechanisms: they are not mutually exclusive

Any of these mechanisms can be used in tandem

Inter-Process Communication

So there are many IPC mechanisms: they are not mutually exclusive

Any of these mechanisms can be used in tandem

- Our program might employ D-Bus to pass data from one application to another (e.g., cut and paste)

Inter-Process Communication

So there are many IPC mechanisms: they are not mutually exclusive

Any of these mechanisms can be used in tandem

- Our program might employ D-Bus to pass data from one application to another (e.g., cut and paste)
- D-Bus might use pipes to communicate between processes

Inter-Process Communication

So there are many IPC mechanisms: they are not mutually exclusive

Any of these mechanisms can be used in tandem

- Our program might employ D-Bus to pass data from one application to another (e.g., cut and paste)
- D-Bus might use pipes to communicate between processes
- And pass a filename between them

Inter-Process Communication

So there are many IPC mechanisms: they are not mutually exclusive

Any of these mechanisms can be used in tandem

- Our program might employ D-Bus to pass data from one application to another (e.g., cut and paste)
- D-Bus might use pipes to communicate between processes
- And pass a filename between them
- and the data is communicated in the file

Inter-Process Communication

So, which IPC mechanism to choose?

Inter-Process Communication

So, which IPC mechanism to choose?

As always, it depends on the application

Inter-Process Communication

So, which IPC mechanism to choose?

As always, it depends on the application

The best way to choose is to have lots of experience of using them

Inter-Process Communication

So, which IPC mechanism to choose?

As always, it depends on the application

The best way to choose is to have lots of experience of using them

- The level your program is at: low or high?

Inter-Process Communication

So, which IPC mechanism to choose?

As always, it depends on the application

The best way to choose is to have lots of experience of using them

- The level your program is at: low or high?
- The amount of data to be communicated: just a bit or a huge datafile?

Inter-Process Communication

So, which IPC mechanism to choose?

As always, it depends on the application

The best way to choose is to have lots of experience of using them

- The level your program is at: low or high?
- The amount of data to be communicated: just a bit or a huge datafile?
- What is available?

Inter-Process Communication

So, which IPC mechanism to choose?

As always, it depends on the application

The best way to choose is to have lots of experience of using them

- The level your program is at: low or high?
- The amount of data to be communicated: just a bit or a huge datafile?
- What is available?
- What your boss tells you to use

Inter-Process Communication

So, which IPC mechanism to choose?

As always, it depends on the application

The best way to choose is to have lots of experience of using them

- The level your program is at: low or high?
- The amount of data to be communicated: just a bit or a huge datafile?
- What is available?
- What your boss tells you to use
- and so on