Architectural ideas behind OS/2 personality

Compatibility with OS/2 (Intel)

osFree combines OS/2 (Intel) ideas with those from OS/2 (PowerPC). It must run programs using OS/2 API, both console and PM ones. So, on Intel architecture, it must be binary compatible with IBM's OS/2. But on other architectures it will have different ABI [1)], but the same portable API [2)].

Also, like it was in OS/2 (PowerPC), the binary compatibility with OS/2 (Intel) is possible (even on non-Intel architectures). It includes the translator of PowerPC instructions to Intel ones. The Instruction Set Translator (IST) is a special DLL. It could be made on the base of QEMU binary translation.

LX executable format is unlikely to be used on other architectures, because it is tied to the Intel one. So, most possibly, the ELF format will be used. It is cross-platform, extensible and simple. It supports both a 32-bit and a 64-bit variant. IBM Microkernel used ELF too. And more, IBM used an extended ELF format with special binary sections semantics. It supported inline resources, imports and exports, for example. So, it was possible to use UNIX-style shared objects, as well as OS/2-style DLL's, both in ELF format.

So, the userland must be compatible with OS/2 (Intel). This includes some support of 16-bit applications. Although there is a very little of pure 16-bits apps (like cmd.exe and hiew.exe), but some 16-bit API's are widely used today. For example, Kbd/Mou/Vio API's (I.e., Console API) are still 16-bits in OS/2 (Intel), though on OS/2 (PowerPC) they are made 32-bits, because binary compatibility makes no sense on other arch's (only source one makes sense).

TODO: Thunking. Getting rid of thunks in 32-bits executables calling the 16-bit API's.

Kernel-related parts of OS/2

The most of 'kernel' area was decided to be moved to userland of L4 microkernel. So, some services, commonly meant as kernel parts, like filesystems, drivers, memory management, thread scheduling etc. are moved outside the kernel. So, they are not significantly differ from usual applications, and confined inside their address spaces.

The OS/2 personality main server (or simply "OS/2 Server")

Filesystem server

Installlable file systems (IFS)

Filesystem router

IFS helpers

Exec server

Virtual Memory Arenas management

Instalable eXecutable Formats (IXF)

Shared memory management

Multiple Virtual Machines (MVM) server

A historical note

The term “MVM” goes from OS/2 (PowerPC). It replaces the term “MVDM” (Multiple Virtual **DOS** Machines). The PowerPC platform could not so easily emulate the Intel processor real mode, so it required more emulation. Though, we must say that it worked just fine, fast, and theoretically may be used not for DOS only. (IBM had plans to provide a binary compatibility with OS/2 (Intel), on the base of the same component which was used to run DOS Apps).

Read more on MVM personality

Other OS's personalities

The microkernel is a 'core' of the system implementing the very minimal set of mechanisms and abstractions. It has no policies in it, only pure mechanisms. The OS's are implemented on top. There can be multiple separate OS'es running on top of a common microkernel. In general, the OS's can have only microkernel as a common part.

But it is not a clever idea to create each OS from scratch. So, some higher level abstractions needed to be created on top of the microkernel, and then the OS'es need to be based on them. These higher level abstractions are the set of servers and libs, called the Personality Neutral Services (or Neutral Personality). If compare with WinNT design, it is simiar to 'Native' NT API, which is a base for Win32 and other subsystems

See osFree Whitepaper for general system design, and proposed personalities.

"The one Ring to Bind Them all" (OS/2 as an integration platform for different types of applications)

OS/2 personality is proposed to be a dominant personality. It governs all Personalities and Virtual Machines via the WPS objects. It has API's to communicate other VM's and Personalities. And the Desktop is an OS/2 desktop, using the GRADD Video driver. This Desktop is shared between all 'native' and 'alien' apps.

I heard that IBM registered a trade mark “The Integration Platform (TM)”, and it is a general concept and ideology under OS/2 design.

It advertised as “DOS better than DOS” and “Windows better than Windows”, and it's true. The continuation of this idea was the Workplace OS, which was never finished, because a voluntary desision of IBM management. It is failed also because no other companies were willing to port their OS'es to IBM's microkernel. But we care much only about OS/2 personality as a base and our OS. The other personalities could be reused as results or other projects, like L4linux or Reactos.

The current OS/2 personality prototype

osFree PM

The osFree PM is an osFree version of FreePM, which was began by Evgeny Kotsuba, and then abandoned. After that, we did some changes to it, and called it “osFree PM”, to avoid name collision in case Evgeny will continue his initial version.

- Docs about FreePM, by Evgeny, now inaccessible
- FreePM on SourceForge Forum and code repository
- Documentation on PM implementation based on FreePM
- Some parts of FreePM docs written by Evgeny.

Graphical Program Interface (GPI)

The GPI [3] is the graphical engine of Presentation Manager. It is based on PM GRE [4]. The GPI/GRE is the counterparts of Windows GDI. The GPI/GRE pair is designed as an enhanced version of Windows Graphics Engine. Contrary to Windows, they are decomposed to two layers. The GPI is the high-level layer.

The Windows programs API's operate directly on DC [5]. OS/2 PM is redesigned, so Programs operate on PS [6], not the DC. The DC is something related to the instance of graphics device (the video screen, a window or printer).

The PS is a higher level abstraction. It can be treated as a canvas in other graphics libraries. It maintain such things as current background/foreground color, a palette, a brush shape, a current font etc. Also, the PS can be associated/deassociated to/from the DC, or migrate from one DC to another. This gives a flexibility to graphics API.

The GPI functions have the “Gpi” prefix and reside in PMGPI.DLL. They are buillt on top of GRE functions.

Graphics Runtime Environment (OS/2 PM GRE) and Presentation Drivers

The GRE is a low-level graphics API which operates DC's. It handles all kinds of graphics deviices like video cards and printers.

The GPI contains an array of pointers called the Dispatch Table. This table stores pointers to the most lowlevel GRE functrions.

The graphics device drivers are called also “presentation drivers”. The presentation drivers get an

instance of Dispatch Table, and hook in GRE by installing their own functions pointers in the Dispatch Table, and preserving the old ones.

The GRADD model

Video Protected Mode Interface (PMI)

VIO/KBD/MOU (Console API)

VIO and BVH's (Base Video Handlers)

Framebuffer Interface

Virtual Framebuffer over GRADD driver

BVH and PM GRE on top of a Framebuffer interface

The current osFree PM prototype

1)
Application Binary Interface
2)
Application Program Interface
3)
Graphical Program Interface
4)
Graphics Runtime Engine
5)
Device Context
6)
Presentation Space