ittersweet Symphony

Jambo casts his cynical eye over Apple's beleaguered Operating System plans, past, present and future. Oh, and he gets a bit sentimental about a dog, too...

I used to own a puppy. He was a lovely little guy who had a way of making you do whatever he wanted. I specifically remember one day when a few of my friends came round and little Steve, as I called him, wandered into the living room, much to the delight of my friends. As soon as he came in, he was the centre of attention with all my mates crowding round like he was the only reason they came (*sniff*). And then all of a sudden everything went wrong. Steve sat down in the middle of the carpet and took the biggest dump of his life. I was left to clean up the mess, while my friends sat and laughed at me.

Now, your starter for 10: Which multinational computer corporation does this

remind you of?

Over the last 15 or so years, Apple has had a fairly turbulent time, and no part of its history has stunk more than the repeated restructuring of the Apple OS plan.

First of all I'd like you to cast your mind back to 1995, when Choco Crispies were Coco Pops and Starburst were Opal Fruits. Apple was still reeling from the fact that it had spectacularly failed in its assessment of the number of people wanting to buy Performas, and to cap it all off it was being slagged off for the poor build quality of its recent line of PowerBook laptops. There was no doubt that Apple needed something to bring back enthusiasm to the workforce and pep up the diminishing support from the all-important purchasing public.

Enter Copland

Copland was Apple's first attempt at a major rewrite of the ubiquitous macintosh operating system. This new generation operating system had some impressive specifications. For a start Copland promised to give the MacOS a bold new look, moving away from the flat, boring System 7 look to a more sophisticated 3D platinum effect. This bold new interface idea spawned some very innovative extensions, most notably Aaron (a play on the new MacOS' development name. Aaron Copland was once a very famous musician, preceding even the likes of Sting and Steve Tyler, hence Aaron came before copland) which later became Kaleidoscope. Under the bonnet, however, was were the real advantage lay. As well as a fast new search system, nicknamed 'V-Twin', this bold new OS would be fully PowerPC native, enabling Apple to finally get the most out of their new baby, the PowerMac.

Copland also promised Pre-emptive multitasking, a feature common in modern OSes and long-promised for the MacOS. In a nutshell, pre-emptive multitasking allows you to run more than one application at the same time, without a noticeable drop in performance. While it has been possible, through a technique called 'Cooperative Multitasking', since the days of the switcher and multi-finder to run miltiple applications at the same time, this has always meant that one application would take all the CPU time while the others sat and waited for it to finish. With pre-emptive multitasking the MacOS acts as a supervisor, giving out access to the CPU. Each program gets to use the processor, but they are never allowed to take all the processing time, meaning that other applications can carry on their work uninterrupted.

Another feature of Copland was Protected Memory, which means that each program only uses its own assigned area of the memory, so that if a program crashes it only takes down its own bit of memory, leaving you to simply force quit the app and carry on with your work. No restart needed!

Exit Copland, stage left

All these features are great, but there was a drawback. A drawback that turned out, in fact, to be big enough to kill the Copland project. Compatability. Despite having a truckload of engineers working on Copland, there seemed to be no way that Apple could get traditional mac programs to work with the new OS. This was a major problem for Apple because it meant that developers would have to re-write from scratch all their programs in order for them to work with Copland, a fact that did not go down well with the mac developer community. Imagine your company had spent thousands of man-hours and millions of Dollars on a program, only to find that Apple had moved the goalposts and forced you to re-write the whole application. Not a good deal.

And so, dealing with resentment from every corner, along with constant delays in its development, Apple was forced to terminate the copland project and put engineers back in the department that was dealing with development of the current OS - a department that was, at one point, working with only two engineers.

Be Pre-emptive, Be multithreaded, just Be

After the obliteration of the Copland project, Apple was once again on the prowl, looking for an operating system worthy of filling system 7's boots. At this point it became apparent that apple were not going to be able to re-write a whole new OS, and it seemed increasingly likely that a third-party operating system would be purchased and further developed. The press' hot tip was Be, a company founded by former Apple employee Jean Louis Gassée. The BeOS, which could be run on either the company's own BeBox computer or - crucially - PowerMacs with a PCI bus. The BeOS was causing a stir in the computer industry, with its high performance and support for multiple processors, allowing it to perform as a top-notch machine for digital media. However Apple obviously didn't think Be was the right OS for them...

The NeXTStep

Instead, Apple plumped for NeXT. This move was a significant moment in Apple history because it meant the return of Apple's founder Steve Jobs, whom Apple fans had a strong affinity with, seeing him as something of a cult hero.

Jobs aside, this purchase meant that Apple had access to all of NeXT's resources - including their NeXTStep operating system.

NeXTStep, according to Apple's plan, would be developed into a new

Operating System codenamed 'Rhapsody'. The OpenStep system, part of NeXTStep, would become the 'Yellow Box' which would on top of another NeXT technology, the Mach kernel, handling between them tasks of varying complexity.

This layering system was to be extended to such an extent that Yellow Box would work on top of other OS'es so that one piece of software would work almost universally accross all the major operating systems. Yellow Box was to be developed for Rhapsody, Windows 95, and its industrial-strength sibling Windows NT

It doesn't stop there, though. Another layer called the 'Blue Box' was to lie on top of the Yellow Box. The blue box was essentially a MacOS emulator (a kind of Mac9X if you like) that would let Rhapsody run old Mac programs, albeit without Rhapsody's advanced features like Protected Memory. This fact meant that Apple were able to rescue some of their best technologies; V-Twin resurfaced as sherlock, for example.

So this time we had partial compatibility - programs could be run on the new OS but if software companies wanted to take advantage of Rhapsody's new features, they would still have to re-write their programs from scratch.

Once again, developers kicked up a fuss - after all, they were not going to spend their time and money re-writing their programs for a new, untested operating system.

The last straw

And so, once again, Apple re-thought their OS plan. They eventually decided upon a compromise, taking the best features from the traditional MacOS and Rhapsody, and combining them to make a new operating system that promised both advanced features and total Mac compatability.

MacOS X

Apple received a lot of flak for canning Rhapsody, as it meant that they had spent an incredible amount of money on NeXTStep, a technology that would not be used. This however, proved to be untrue as MacOS X retains a lot of Rhapsody's features. Indeed, MacOS X is in essence Rhapsody with native MacOS support.

Carbon Copy

So how come Apple can do this? Well, it's all to do with a new set of API's (Application Programming Interfaces). Apple engineers looked at the toolbox (a collection of over 8,000 API's) and removed all the toolbox calls that were

causing problems. The result: Carbon - the 5,000 or so remaining API's and some new ones.

Introducing Carbon meant that software would not need to be re-written for the new OS, just tweaked. Finally Apple had come up with an OS strategy that the developers liked.

Command-W

So, what have we learned from this brief history of Apple's OS strategies? Well, first of all we have learned not to trust Apple unless we have actual, physical evidence that their OS is going to be released.

But more than this, we have learned never, ever, to let your dog in the house unless it is toilet-trained.

Jambo@yewclark.demon.co.uk