Introduction to PersiMMon

T.R.Barbour (trb@categorical.net)

In the world of operating systems, there has been little progress beyond Unix, which was created around the same time as the C programming language. Since then, there has been much progress in the design of programming languages, and some modern declarative programming languages (e.g. Haskell, Mercury, Curry) are much more advanced than C, yet they have no operating system counterparts (e.g. a viable functional OS).

It is time to seek more advanced operating systems, by analogy with modern declarative programming paradigms. Even a small step in this direction can change the way we use computers and the Internet; however the problem space is large. Efforts so far have not met with resounding success. Some have produced second-class data-types, stuck inside files. Others lack device drivers, developers and users.

What is the most effective way to proceed? One could start by building a good *foundation* for advanced new OS paradigms and bootstrapping from an existing OS (e.g. Linux). A good foundation should overcome fundamental weaknesses of current OSs. A notable weakness of todays operating systems is the filesystem. A file is an abstraction for a deck of punch cards or a reel of tape, both of which have been obsolete (for primary storage) for a long time. Similarly a socket is an abstraction for a serial cable. Surely we can do better than this!

A better abstraction is persistent memory. With persistent memory there is no need for files, and no need for sockets. Persistent memory presents a convincing and practical illusion of a single large, fast non-volatile memory; distributed persistent memory extends this illusion so that all data seems local.

The PersiMMon project will implement persistent memory, *properly*. Persistent memory is a very powerful idea that has never quite become viable. PersiMMon will change that, by exploiting the power of free software.

There are many kludgey "persistence" schemes perched on top of filesystems or relational databases. PersiMMon is different: it will implement persistent memory, at the OS level (persistent memory is related to virtual memory).

There have also been more principled attempts at persistent memory, some of which have been moderately successful. The Multics and KeyKOS operating systems had persistent memory, but suffered from being proprietary. More recently KeyKOS was rewritten and released, under the GPL, as EROS. Although it is a very interesting operating system, no-one is using it, and hence it is short of developers and applications. When a new operating system is short of developers and applications, no-one is likely to use it.

There have also been user-space implementations of persistent memory (e.g. ObjectStore). These typically suffered from the fact that they were sitting on top of a proprietary OS, and hence unable to integrate sufficiently tightly with the OS. In order to handle one page fault, such systems have to cross the user-space to kernel-space boundary several times, making it impossible to achieve good efficiency. These systems also suffered from being proprietary (and typically expensive), limiting their user-base.

If there were any doubt about the desirability of distributed persistent memory, the need for it is shown by the fact that typical forms of enterprise software try to deliver something like it; because of

inadequate OS support, this is almost always an awkward kludge, such as EJB and object-relational mapping layers.

PersiMMon will have at least the following features:

- persistent memory, at OS level
- true orthogonal persistence (with zero burden on application developer)
- ACID properties, including nested transactions
- meta-data
- page-grain locking (optional)
- page-diffing
- online page relocation
- excellent performance
- strong security
- transparent access to remote data, with caching, replication and clustering
- kernel independence
- GNU GPL

although the remote access features may not be in the first release.

PersiMMon is a contraction of Persistent Memory Module, and is available as a module for the Linux kernel. However PersiMMon does not depend on Linux, but only on a kernel abstraction layer (kernabs), which can be implemented on various kernels. Currenly kernabs is implemented for Linux and for user-space (Posix), and PersiMMon runs unchanged in either of these environments. It should be straightforward to port kernabs to other kernels, such as BSD, HURD or L4.

Running PersiMMon entirely in user-space does not provide the best performance, but it is useful for testing and debugging, as well as providing a means for people to try PersiMMon without needing to build the kernel module.

By virtue of being free software, PersiMMon will succeed dramatically, in a way that no similar system has. As free software, it will be available on popular operating systems, to millions of users. The consequences of avoiding filesystems and hiding the network are profound; as people realise this, the way we use computers and the Internet will change.

PersiMMon has been under development since August 2000. During that time it has been through several re-designs, starting with a very conservative design, and ending with the radical Mk5 design, of which a carefully chosen subset is now operating, and nearing completion (see the status diagram for more details).

Future development (after the initial release, and involving other developers) will be aimed towards Internet scale programming, including:

- remote access, with caching, replication and clustering
- stable names (generalisation of pointers, similar to DNS FQDNs)

- pointer swizzling to support stable names
- data swizzling for endianness
- code pointer swizzling to support different CPU architectures
- data diffusion
- unified address space for all the memory on the Internet

In conclusion, PersiMMon will bring a new OS paradigm into the mainstream. Distributed persistent memory hides disk storage and the network, providing a clean abstraction as a foundation for advanced new operating systems and Internet applications. This is technology that will change the computing world.