Brief Technical History of Unix

Dr. D. M. Ritchie

Rapporteur: Mr. A.J. Linton
Mr. G.N. Dixon

Abstract:

The Unix operating system has a long history, and although its flavour and style have been maintained since it began, it has undergone much change. This talk will discuss the major milestones in its technical development as it outgrew the PDP-7, the PDP-11, and the VAX, and eventually found a home on the Cray 2. Instead of listing system releases, I will try to describe the real improvements in capability: a hierarchical file system, multiprogramming, networking, interprocess communication.

A useful reference for the topics covered in this talk is: Ritchie, D., "The Evolution of the UNIX Time-sharing System", AT&T Bell Laboratories Technical Journal, Vol. 63, No. 8, October 1984.

Discussion

Brian Randell asked was reusability of software using pipes an important concept in the development of UNIX®†

Dennis Ritchie said that Doug McIlroy had the idea of pipes early on in the development of the system and pushed to have pipes included in the system. A lot of thrashing out of the syntax of pipelines occurred (see the paper referenced above). Reusablity has been an important concept.

Harry Whitfield asked has any work been done on providing full duplex pipes e.g. ‖ ?

Dennis Ritchie replied that Doug has tried this but has been less successful - no satisfactory notation has yet appeared. Some work has been done on providing multiple inputs to a pipe e.g.

```
(a, b, c)| d

3 output    3 input
streams -->  streams
```

This won't block - no deadlocks but it gets messy because of the multiplicity of the standard input, output and error streams (stdin, stdout and stderr).

Harry Whitfield asked do we need new abstractions for interprocess communication, such as the socket mechansism in the Berkeley 4.2BSD system?

Dennis Ritchie stated I am not necessarily against this.

Herman Kopetz asked what about UNIX for real time applications?

Dennis Ritchie replied Hewlett Packard have done it. The problems lie in the critical regions where interrupts are locked out.
i.e.

```
s = spl5();
...
...
(void) splx(s);
```

Clean up these areas and things are much better. I don't know how good it can be made.

† Registered trademark of AT&T in the USA and other countries