Extensibility, Safety and Performance in the SPIN Operating System

Bershad et al

Presentation by norm

Slides shamelessly “borrowed” from Stefan Savage’s SOSP talk
Goals

• Extensibility
  – Applications can dynamically extend system to provide specialized services
• Safety
  – Kernel is protected from actions of extensions
• Performance
  – Extensibility and safety have low cost
Why is this hard?

Can we have all three in a single operating system?
Approach

• Put extension code in the kernel
  – Cheap communication
• Use language protection features
  – Static safety
• Dynamically interpose on any service
  – Fine-grained extensibility
A SPIN extension
SPIN structure

User

Kernel

Applications

OSF/1 Unix server

Unix Apps

Video Server

Web Server

Mach API

Unix API

Net Video

HTTP

Threads

Syscall

Process

Network

File Sys

Execution State

Memory

Devices

Extension Services

Application Extensions

Shared Extensions

SPIN Core Services
Safety

• Language-based protection based on Modula-3
  – Memory safe
  – Interfaces for hiding resources
  – Cheap capabilities
Extensibility

• Events defined as procedures within interfaces in Modula-3

• Dispatcher for finding handlers
  – Guards for selective execution of handlers
Dispatcher
Other services

• Almost all “system” services are extensions
  – Network protocols
  – File systems
  – System call interface
• SPIN only implements services which cannot be safely implemented as extensions
  – Processor execution state
  – Basic interface to MMU and physical memory
  – Device IO / DMA
  – Dynamic linker and Dispatcher
A protocol graph in SPIN

UDP recv
  Active messages
  RPC
  Video
  TCP recv
  ICMP count
  Ping

UDP packet arrived
  IP packet arrived
  Ethernet count
  Ether packet arrived
  Ethernet driver

TCP packet arrived
TCP

HTTP

IP

ICMP packet arrived
ICMP
Performance

• SPIN runs on DEC Alpha platforms
• Measurements
  – DEC AXP 3000/400 @ 133 Mhz
• Comparison systems
  – DEC OSF/1 V2.1
  – Mach 3.0
SPIN performance advantages

• Extensions provide specialized service
  – Don’t execute unnecessary code
• Extensions execute close to kernel services
  – Low latency response to faults/interrupts
  – Invoking services is cheap
Video service

![Graph showing the relationship between percent CPU utilized and number of video streams for DEC OSF/1 and SPIN.](image)
Other basic system services
Discussion

• Do user level programs need to be written in Modula-3 like extensions do?
Discussion

• How can you force extension writers to use Modula-3?
Discussion

- Can SPIN can be efficiently used in resource constrained situations? Most memory safe languages, including Modula-3, are more ‘heavy weight’ than C in which most OS kernels are written. Isn’t performance the reason that we are still using C in most OSes today?
Modula-3 vs C

• Most operations are compiled equivalently whether written in Modula-3 or C
• Modula-3 can sometimes introduce runtime checks to guarantee type safety
Discussion

• In Spin, can multiple event guards be true at the same time? If so, how will the dispatcher decide the order of activating different event handlers?
Discussion

• The authors compare the performance of SPIN with Mach, however, Mach is slow. Do you think SPIN will have good performance compared to L4?
Discussion

• What happens if an extension raises an exception?
Discussion

• Can a buggy extension crash the system, perhaps the dispatcher?
Discussion

• What is the essential difference between SPIN and L4?
Discussion

• Is it really secure to trust the type system of a language for OS safety?
Discussion

• Why don’t we see any extensible OSes today?