Exokernel (Aegis + ExOS)

- **Motivation:** fixed-interface between applications and hardware resources does not allow applications with specific needs to use resources effectively
  - Interface is **fixed**, and cannot be replaced by user-level applications
  - In current OSes, Protection and Management of resources live in the kernel

- **Objective:** separate protection of resources from management
  - Exokernel ensures protection of resources
  - “library operating systems” manage them

- **End-to-end** design, like in networking
  - Exokernel is simple; main responsibility is safe multiplexing of resources
  - **Less switching** between kernel- and user-mode since most complex functionality is found in the library OS

- **Exokernel Design**
  - **Secure bindings:** fine-grained access to all hardware; manage authorizations to use resource, not control; use a Software TLB to cache secure bindings
  - **Visible revocation:** library OS is notified (and takes part) in resource revocation; **slower**, done even for CPU time; uses exported physical names to speed up process and avoid ambiguity
  - **Abort protocol:** revoke resource, use a “repossession vector” to notify library OS of lost resources (small number of resources is protected from revocation)
• **Downloadable Code**
  - VCODE: create executable code at runtime, to be run inside the Exokernel without requiring a context switch
  - **Fast Networking**: Dynamic Packet Filter (DPF) – packets can begin to be processed in the same buffer where they are received
  - **Application Specific Handlers**: untrusted code checked at time of download; high-speed messaging possible in Exokernel, allowing

• **IPC**
  - IPC primitives coexist in the same library OS; very fast communication between processes since no trip to the kernel code is necessary

• **Advantages**
  - Benchmarks comparing Aegis/ExOS to UNIX usually favor the former by considerable margins
  - Different library OSes can coexist easily

• **Weaknesses of this solution**
  - Both the Exokernel and the Library OS are architecture dependent! Portability of applications is no longer straightforward
  - Even within the same architecture, changes to the hardware require rewrite of Exokernel and Library OS to take advantage of new features or just to guarantee basic compatibility