What is *Good Systems Research?*

Liviu Iftode
Rutgers University
What is Systems Research?

- **Broad definition**: all core components of a computer system
  - computer architecture, operating system, languages and compilers, networking, fault-tolerance, systems security

- **Narrow definition**
  - operating systems and distributed systems

- **Challenging research field**
  - Prototype implementation and evaluation
  - Large teams (see SOSP’09)
  - A lot of “hidden work”
  - Boring sometime

Is this computer science or engineering?
A Changing Research Field

- Two driving forces
  - new technologies
  - new applications
- Perspectives shift
  - performance & scalability
  - reliability, availability and survivability
  - maintainability
  - manageability
  - security
  - usability
Systems Research Questions

- What drives systems research towards success or failure?
- Where do novel ideas come from and when?
- What makes systems research results last longer?
- How to anticipate next “fashion” in systems?
Evolution of Research Themes

- UNIX
- Parallel systems
- Distributed and cluster-based systems
- Internet systems
- Storage systems
- Peer-to-Peer systems
- Embedded/Sensor systems
- Virtual machines
- Transactional memory
- Systems for multi-core processors
- Pervasive/Ubiquitous/Cyber-Physical systems
- Cloud computing systems
Talk Roadmap

- Retrospective view of past 10 editions of the Symposium on Operating Systems Principles (SOSP)
- Discuss
  - main topics
  - selected papers
  - lessons to take away
  - examples of good/bad systems work
  - they are personal opinions
Ten SOSP Conferences

- SOSP’91, Pacific Grove, CA
- SOSP’93, Asheville, NC
- SOSP’95, Copper Mountain, CO
- SOSP’97, Saint-Malo, France
- SOSP’99, Kiawah Island, SC
- SOSP’01, Chateau Lake Louise, Canada
- SOSP’03, Bolton Landing, NY
- SOSP’05, Brighton, UK
- SOSP’07, Stevenson, WA
- SOSP’09, Big Sky, MT
Main topic: File systems
  ◦ Log-Structured File System
  ◦ Semantic File Systems
  ◦ Coda

On the rise:
  ◦ Multi-threading: Scheduler Activations
  ◦ Multiprocessors: Munin (software DSM)
  ◦ Real time/Multimedia

New problems:
  ◦ Networking: Automated Reconfiguration
  ◦ Security: Authentication
  ◦ Reliability: Replication in File Systems
Log-Structured File System (LFS)

- **Problem:**
  - More memory -> larger caches -> disk traffic dominated by writes
  - Writes are synchronous and dominated by small accesses (metadata) -> slow

- **Solution:**
  - Buffer file changes together (data and metadata) and write them all in a single large disk write (log)

- **Advantages:**
  - Fast (practically eliminates the seek operation)
  - Simple recovery

- **Secondary issues:**
  - How to locate file blocks
  - How to manage the free space
LFS -
An Example of Good Systems Research

- Driven by awareness of the technology trend
- A radical depart from traditional approach
- Maintains compatibility with traditional approach
- Good model analysis
- Solid implementation and evaluation
- Long lasting impact (see SOSP’09)
Problem:
- Mobile computing causes disconnections
- File system should remain usable in the presence of disconnections

Basic Idea
- Availability more important than consistency

Solution
- Cache whole files in advance (best-effort hoarding)
- Emulate server at client when disconnected
- Reintegration upon reconnection: resolve conflicts

Advantage
- Simple, feasible, usable
- Addresses the common case (no file sharing)
Real problems but too complex solutions

- **Scheduler Activations**
  - Problem: both user-level and kernel-level threads have limitations
  - Solution: a new kernel interface for user-level threads

- **Munin**: the first release-consistency software DSM
  - Problem: too much false sharing due to large coherence granularity
  - Solution: convince programmer to accept a relaxed consistency model in order to get acceptable performance
SOSP’91 Summary

• New directions triggered by new technologies:
  ◦ memory capacity is increasing
  ◦ mobility

• Main application domain (high-performance computing) is offered too complex solutions
  ◦ scheduler activations
  ◦ release-consistency software DSM
Interesting new problems
- Make networking fast using *Fbufs*
- Models for distributed systems: *Limits of causality and total ordered communication*
- *Location information in ubiquitous computing*
Main topic: OS structure and performance
- Exokernel and SPIN
- Impact of Architecture on OS Performance

On the rise:
- Reliability: Hypervisor, Hive, Logged VM
- Mobility: Bayou

Still of interest
- File systems: xFS, Informed Prefetching and Caching
- Distributed memory: Global Memory, CRL

Follow-ups
- Weak connectivity

New problems
- User-level communication: U-Net
- 64-bit address spaces
New OS structures and OS performance

- **Problem**
  - OS has become too big, rigid and hard to manage
  - Performance does not increase with raw hardware performance

- **SOSP’95 debate**
  - Liedtke: Micro-kernel is fine, just needs a good implementation
  - Kashooek: *Exokernel* - Nano-kernels with application-controlled resource management
  - Bernshad: *SPIN* - Safe extensibility through downloadable modules written in a type-safe language

- **What happened after**
  - Complex solutions with hardly feasible assumptions
  - Influence on future OS design happened but not clear how much
Research on OS reliability on the rise

- **Problems**
  - OS often fail, how to make applications survive?
- **Papers anticipated the VMM research (a decade later)**
- **Hypervisor-based fault-tolerance**
  - Interpose a VM software layer between hardware and OS
  - Log non-deterministic events to mirror state of a primary computer onto a backup
  - Continued a decade later with the ReVirt but applied to intrusion analysis
- **Hive**
  - Fault-containment using a cellular OS
  - Better resource allocation
  - A decade later: VMware Inc
OS Research for Cluster Computing

- OS research influenced by cluster computing demands for performance and programmability
  - TCP/IP: too heavy for high-performance interconnects
  - Message-based programming: too difficult for large scale

- User-level communication: U-Net
  - Simple and efficient but requires RDMA support
  - Today: Infiniband for network storage

- CRL software DSM and global memory
  - Make it simple: no relaxed consistency
  - Still did not make it
When it is too much complexity

- **xFS**
  - A completely decentralized distributed file system
  - Wonderful engineering work

- **Bayou**
  - Eventual consistency using an anti-entropy protocol for update propagation
  - Automatic resolution of update conflicts
  - Why did Coda and not Bayou survive?
Concern about OS structure and performance

Research in OS reliability anticipates new challenges: fault isolation and containment (SOSP’07, SOSP’09)

Cluster computing only half-way successful in OS research

Research in mobility and file systems propose too complex solutions

Follow-up on your prior research may not be a bad idea
SOSP’97

- On the rise
  - OS support for Internet services: BASE instead of ACID, Security in Java
  - Real-time scheduling for multimedia
  - Application Adaptation for Mobility: Agile

- Still around
  - Running commodity OSes on multiprocessors: Disco
  - Software DSM: Shasta, Cashmere
  - Distributed file systems: Frangipani
  - OS performance profiling

- New problems:
  - Decentralized information flow-control
  - Dynamic data race detection: Eraser

- Follow-up
  - Scalability in Exokernel
  - Flexible update propagation in Bayou
New Research Problems

- Decentralized Information Flow Control
  - Privacy becomes a concern in client-server interaction
  - How to share information among systems with mutual distrust
- Detection of data races in multithreaded programs
  - Bugs caused by programming error to follow a locking discipline
  - Exposure depends on scheduling non-determinism
  - Notorious hard to detect, reproduce, locate and eliminate
  - Race-freedom not enough for correctness: atomicity also necessary (see ASPLOS’06)
- Both papers prevent programming errors to cause damages, hot field today
The Internet changes the OS

- Internet services require OS changes
  - Problem:
    - ACID (atomicity, consistency, isolation, durability) data semantics is hard to support and not always required
    - Availability is more important
  - Solution
    - Relax data semantics for better scalability
    - Optimistic approach, supports partial failures
    - BASE (basically available, soft state, eventual consistency)
- Internet Services: significant impact on OS research
  - LRP, Scout-OS, LARD, IO-Lite, Resource Containers
  - Security
SOSP’97 Summary

- Internet service applications are about to become the next main application domain
- High-performance computing makes its last strong appearance in OS research (two DSM papers!)
- A slowly but steadily emerging field: detection of software bug (still present at SOSP’09)
Main topic: Internet Services
- Availability & Scalability: Porcupine
- Negative result: Cooperative web proxy caching
- Distributed VM for networked computers
- Soft-timers for network processing

On the rise
- Security: Separating key management from FS security
- Networking: Intentional Naming System, Click

Still there
- User errors: When to forget in Elephant FS
- Real time OS issues

New
- Energy: Adaptation for mobile applications

Follow-up
- Resource management in Cellular Disco
New Research Problems

- The email is important
- Battery lifetime is also important: adapt applications
- User mistakes must be addressed by the OS
  - *Elephant*: Let system decide when/what to delete
- Interrupts are expensive for network servers
  - *Use soft-timers*
- Security for global file systems
  - *SFS*: Self-certifying pathnames eliminate need for key management
- Naming in dynamic and mobile networks
  - *INS*: route messages by names
- Flexible network routers
  - *Click*: implement routers in software
Why researching the email?

- **Email is important**
  - Real demand
- **Email is hard**
  - Write intensive
  - Low locality
- **Email is easy**
  - Well defined API
  - Large parallelism
  - Weak consistency
Main topic: Peer-to-Peer and Overlay Networks
- P2P storage systems: PAST, CFS
- Resilient Overlay Networks (RON)

On the rise
- Software Bugs: Bugs as Inconsistent Behavior, OS Errors

Still around
- Internet Services: Continuous Consistency, Event-Driven Programming
- OS adaptation: Gray-Box
- File systems and networking: Low-bandwidth Network FS

New problems:
- Energy conservation in hosting centers
- Sensor Networks: Low Level Naming

Follow-ups
- Privacy among untrusted hosts: Secure Programming Partitioning
P2P “hijacked” the OS research

- Decentralized storage systems
  - Main properties: scalable, highly available
  - Use scalable routing and lookup substrates (*Pastry, Chord*)
  - What is the OS problem they solve?

- **Resilient Overlay Networking (RON)**
  - Problem: Internet routing problem
  - Solution: Application-layer overlay on top of IP
  - Advantage claim: more resilient than IP routing
  - What is the OS problem?

- Are sensor networks becoming the next “hijacker”?
Practical Intellectual Challenges

- Concurrency programming debate:
  - *Seda*: Threads or Staged Event-Driven?
  - Event-driven manages load better
  - Practical

- How to transfer file over a low-bandwidth connection?
  - Exploit similarities between files and file versions
  - Avoid sending data blocks over the network if they are already in cache at server/client
...and Less Practical Intellectual Challenges

- **Continuous consistency**
  - Replication for availability makes consistency hard to strictly maintain
  - Trade consistency for availability
  - Metric: max deviation from strong consistency on replica-basis

- **Gray Box**
  - How to acquire OS internal state info and control without modifying the OS?
  - Interpose information&control layers between client and the OS to exploit knowledge of the algorithms used by the box
  - Successfully used for controlling file caching, disk layout, etc
Software Bugs

- **Problem**
  - How to determine the correctness rules when programmers do not specify them

- **Solution**
  - Infer rules as “programmer beliefs” from static analysis
  - Cross-check them for contradiction

- **Evaluation**
  - Hundreds bugs are found in Linux: better than manual

- **Question**
  - Dynamic monitoring (*Eraser*) or static analysis?

- **Anticipate more papers on OS errors at next SOSP**
  - Device drivers have error rate 3-7 times higher than the rest of the kernel (see *Nooks* at next SOSP)
New Problems

- **Conserving Energy in Hosting Centers**
  - Problem: Energy becomes the driving resource management issue
  - Solution: Adaptation to load by dynamically resizing the active server set with a certain degradation of service

- **Sensor Networks**
  - Low-level naming based on attributes relevant to the application and external to the network topology (like INS but not over IP)
  - In-network processing of data: directed diffusion
  - What is the OS problem?
Main topic: **OS Robustness**
- Execute untrusted code: *Model-Carrying Code*
- VMM: *Xen, Untrusted on XOM, Terra*
- Handle bugs in OS drivers: *Nooks*
- Race condition detection: *RacerX*

Still around
- File Systems: *Google File System*

Follow-ups
- Policies into Mechanisms using *Infokernel*
- Overlay networks and P2P
New and old hardware inspires OS research

- Problem
  - OS not trusted

- Three solutions
  - Virtual Machine Monitors: *Xen*
  - OS over XOM processor architecture
    - HW trusted to execute tamper-resistant SW
  - OS over a trusted VMM (*Terra*)
    - Tamper-resistant HW partitioned in multiple isolated VMs
    - Applications can cryptographically authenticate the software stack to remote parties (attestation)
Growing interest on software bugs

- **Nooks**
  - Problem
    - faulty drivers
  - Solution
    - Fault resistance (survivability) not fault tolerance (reliability)
    - Isolate driver failures with lightweight protection domain to prevent kernel corruption

- **Backtracker**
  - Problem: analyze intrusions is hard
  - Solution: VM to log events and objects in dependency graphs
  - Remember Hypervisor at SOSP’95?

- Static detection of race conditions and deadlocks: RacerX
Main topic: OS Security and Robustness
- OS integrity without HW: Pionner
- Intrusion Detection and Containment: Vulnerability-Predicates, Vigilante
- Software bugs: Asbestos, Rx

Still around
- Declarative overlays
- Byzantine fault tolerance
- Semantics in File Systems: Connections (remember SOSP’91?)
- Race detection with adaptive tracking: RaceTrack

Follow-ups
- IRON File Systems
Pionner: No hardware inspires OS

- The Intellectual Problem:
  - Verify code integrity
  - No trusted hardware support

- Solution
  - All-software based code attestation using integrity measurements
  - Expected time of checksum code execution

- Assumption
  - Client (dispatcher) knows the configuration of the untrusted hardware
Software bugs: New Approaches

- **Labels & Events**
  - Problem
    - Current OS abstractions do not provide sufficient flexible isolation between different users
  - Solution
    - OS support for information flow control

- **Rx: treating bugs as allergies**
  - Problem
    - How to survive software failures safely
  - Idea
    - Bug exposure depends on execution environment
  - Solution
    - Rollback, modify environment and re-execute
A New OS Problem: Intrusion Detection

- Detecting intrusions using vulnerabilities predicates
  - Problem
    - Prevent software bugs to be exploited by the attacker until they are fixed
  - Solution
    - Define predicates to monitor intrusions that trigger the vulnerability
    - Use VMM: IntroVirt

- End-to-end containment of internet worms
  - Problem
    - Internet worms containment must be done automatically because they spread too fast
  - Solution
    - Collaborative worm detection and containment
    - Self-certifying alerts using proof of vulnerability (since hosts do not trust each other)
    - Use SCA to generate filters to block infection
Main topic: OS Robustness
  - Survive with vulnerabilities by blocking bad input: Bouncer

“Back to the Future”: Byzantine Fault Tolerance

Still around
  - Information flow-control: for the OS, for web applications
  - Energy
  - Virtual machine for commodity OS security: SecVisor – A Tiny Hypervisor
  - File systems: Dynamo (Amazon’s key-value store)

New problems:
  - Web and OS: MashupOS
  - Transaction Memory and the OS: TxLinux
  - OS maintenance: upgrade and configuration management
  - Storage of interaction: DejaView

Follow-ups
  - Software bugs: MUVI
Interesting novel projects

- **MashupOS**
  - Problem:
    - Web 2.0 applications are becoming complex
    - Browsers are left behind
  - Solution:
    - browser-based OS with support for protection and communication

- **TxLinux**
  - A new technology: hardware transactional memory
  - How to make OS locks work with transactions?
  - How to integrate transactions with OS scheduler
  - A well-done “traditional” OS work
“Rethinking” existing solutions in a different light
- FAWN: clusters of power-efficient well-balanced nodes
- Multi-kernel: OS structure for multicore systems
- BPFS: byte-addressable file system using BPRAM

Important result
- Formal verification of an OS kernel

Large-scale real experience
- Debugging at Microsoft

Survivability
- Drivers: tolerate hardware failures in software
- Surviving sensor networks software failures

Practical research:
- Drivers, debugging, scalable software routers
- Solutions for emerging hardware
- Machine learning and data mining techniques applied to OS
- Large authorship
- Strong presence of Microsoft Research
Some general observations

- A favorite theme
  - Created by a new technology, a new application domain
  - Sometimes its importance grows slowly
  - Dominates 1-2 SOSP cycles
  - After that, proposed solutions become complex and less influential

- Several permanent themes
  - File systems
  - Real time
  - Byzantine fault tolerance (BFT)
  - Software bugs?

- OS research hijacking?
  - Networking and sensor networks
  - Software engineering and compilers
  - Intrusion detection

- Model analysis becomes a necessary part of OS research
Some final tips

- Have a problem before having a solution
- Make sure that it is a real problem
  - “Your research must be a painkiller not a vitamin”
- Prior work
  - Read papers before you write
- Build prototype
- Evaluate with realistic load
  - Negative results are as valuable as positive results
- Work in a team
- Spend time on paper writing and presentation
- Evaluate the surprise factor (the “wow factor”) of your contribution
“Suggested” areas

- OS survivability, manageability and security
- Cyber-physical systems ("outdoor computing")
- Cloud computing systems
- Systems for smart phones
- Systems research opportunities in online social networking?
- Watch Google: Android, Chrome OS
Thank you!