CS514: Intermediate Course in Computer Systems

Lecture 1: Jan 20, 2003

Course Overview and Themes

Professors:
Ken Birman
Paul Francis

TAs: Rimon Barr
Geoff Crew (half time)

Perspectives on Computing Systems and Networks

CS514

Prerequisites:
- CS314: Hardware and architecture
- CS414: Operating Systems with a focus on single-processor and multi-processor systems

Complementary aspects of the same broad area:
- CS513: Security for operating systems and networks
- CS514: Emphasis on "middleware": Web services, distributed computing, reliability, major platforms
- CS519: Network structure and widely used protocols, mobile networking, emerging issues and topics

Aimed a PhD students:
- CS614: A survey of current research frontiers in the operating systems and middleware space

Picking between 514 and 614

CS514

- CS514 is *practical* in emphasis:
  - We study tools used in real products and real systems. "Technology you can buy or build"
  - But looks hard at what goes on beneath the surface
  - Projects build on popular technologies
- CS614 emphasizes research opportunities
  - Mix of "classics" and state of the art papers
  - Tremendous amount of reading
  - Projects are often original research and many hve resulted in publishable papers
  - For systems students in PhD program, often seen as a way to find a good research topic

CS514 has Two Instructors

CS514

- Professor Paul Francis
  - Visiting for spring semester to "check us out"
  - One of the most famous network researchers
- Professor Ken Birman
  - I've been here for twenty years
  - Launched two startups along the way

Paul's background

CS514

- Paul developed Network Address Translation back when the Internet first ran out of address space.
- Co-inventor of IPv6 (aka IPng). But his breakfast ate his lunch: NATs killed IPng!
- Now is working on mobile IP (SIP) and peer-to-peer stuff.
- Throughout his career has been a leader in the IETF community and has worked on all sorts of network-level topics

Ken's background

CS514

- At Cornell since 1982. Also founded and ran two companies
  - One did ok (Isis Distributed Systems).
  - The second is now six feet under!
- Has worked on
  - Replication for fault-tolerance
  - Security of complex distributed systems
  - Scalability challenges
  - Platforms for robust distributed computing
- Emphasis on "critical infrastructure"

Ken's bias

CS514

- Examples of systems that have actually used these technologies:
  - Air traffic control systems in Europe
  - Stock exchanges (NYSE, SWX)
  - Naval AEGIS battleship radar system, NSA
  - Recently interested in needs of the next generation electric power grid
- For systems like these, the question is:

How can we build software that does what we need it to do, reliably, accurately, and securely?

Recent Trends

CS514

- A network rollout of unprecedented scale continues
  - Larger and larger numbers of small devices, web-compatible cell phones
  - Everything is "on the web" – "Web Services"
- Object orientation and components have become a prevailing structural option
  - Promotes productivity, software reuse
  - Widespread use of transactions for reliability and atomicity
- Platform standards are a battleground
  - Java/ Jini *vs* C#/.NET
- Client-server model is fading… what will replace it?

How can we learn about these?

CS514

- Basically two options
  - Study the fundamentals
  - Then apply to specific tools
- Or
  - Study specific tools
  - Extract fundamental insights from examples

Understanding Trends

CS514

- Basically two options
  - Study the fundamentals
  - Then apply to specific tools
- Or
  - Study specific tools
  - Extract fundamental insights from examples

Butler Lampson's Puzzle

CS514

- Why didn't CS researchers invent the web?

Butler Lampson's Puzzle

CS514

- Why didn't CS researchers invent the web?
  - They "would have wanted it to "work"
  - The web doesn't really work
  - But it doesn't really need to!
- But what about Web Services?

Butler Lampson's Puzzle

CS514

- Why didn't CS researchers invent the web?
  - They "would have wanted it to "work"
  - The web doesn't really work
  - But it doesn't really need to!
- But what about Web Services?
  - Suddenly "the Web" will be everywhere
    - Companies will *depend upon it*
    - Computers will talk to each other this way
  - Maybe reliability is about to come back!

World Wide Web

CS514

- A seductive pass-time, but it never really made it as a serious business model
  - Some big successes: Amazon, eBay…
  - The Web is rapidly replacing paper for many purposes
  - An effective tool for sharing knowledge
- But the Web just doesn't work well enough to "depend" upon it!
  - Web sites are often unavailable for lots of reasons
  - And they are easily attacked
  - The whole architecture seems wrong for mission-critical uses, and even for most corporate uses

Web Services

CS514

- Web Services
  - These let computers to talk to computers in a client-server "style"
  - Builds on the same standards popularized by Web browsers – XML encodings, etc
  - A new degree of language independence, at cost of "bloat" (XML representations are BIG)
- Systems used to be structured as clients and servers
  - The client use remote method invocation to access functionality implemented by the server
  - Web browsers download documents, but when they are generated on the fly, it looks like a form of RMI…
- The match is good enough to have triggered a trend: *rebuild everything as a Web Service*

Relying on Web Services: Banking

CS514

- Imagine a bank that buys into this model, big-time:
  - All the customer accounts will be on databases accessed as Web Services, often through Web browsers
  - Broker will have Web access to up-to-the minute stock quotes and investment data and advice
  - Back office will use the Web to trade stocks based on what the broker currently wants
  - Criminals will try and violate security/privacy to steal funds or manipulate trades

Relying on Web Services: Banking

CS514

- Notice the mixture of:
  - Benefits – standards, everything can potentially talk to everything else
  - Components have a “well understood” structure and behavior
- But also new needs
  - We want 24x7 availability, security
  - A degree of robustness unmatched in the Web on the usual Internet

Relying on Web Services: Medicine

CS514

- Web-style interface in a hospital
  - In fact there are already many that work this way
  - “Global grid assisted telesurgery on a brain cancer patient”
- Doctor relies on accuracy of patient status records to make treatment decisions
- Nurse relies on accuracy of drug dosage and frequency data to administer treatment
- Hospital legally obligated to provide for security and privacy of the data

Relying on Web Services: Publishers

CS514

- More and more publications are going electronic (movies, music, MTV-style videos, etc)
  - Right now, the battle is to keep the bits from escaping from the box
  - In long term, media companies will "sell" information. Publisher's edge: quality of authors, quality of material, complete corpus.
- But for this to work, need reliable ways to charge for access and to limit access

Critical Computing on Web Services: Air Traffic Control

CS514

- Web interface could easily show planes, natural for controller interactions
- The hard part is to make this incredibly robust:
  - Need to know that trajectory and flight data is current and consistent
  - Also need help with routing options
  - Continuous availability is vital. Security and privacy also needed
- Could we get to a point of building ATC systems as Web Services?

It isn't easy!

CS514

- Focus on Air Traffic Control (ATC) for a moment
  - Suppose we want to use Web technologies to build a new ATC system
  - Clearly it needs to guarantee continuous availability
  - How would we approach such an issue?

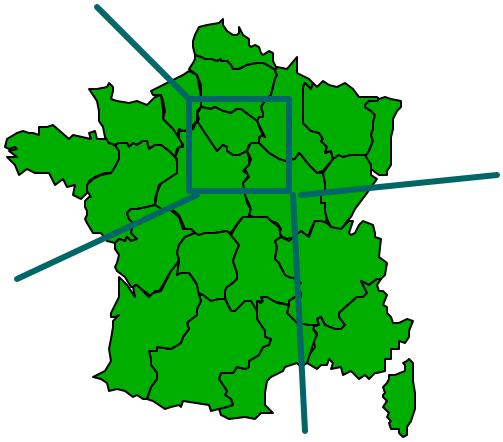
An ATC Fiasco: Advanced Automation System (AAS)

CS514

- Started by FAA in 1989 to replace existing ATC system
- Current system has video display of radar for controllers to use
- Database has information about each flight
- Telephones to talk to the planes

ATC systems divide country up. Here's France

CS514



More details on ATC

CS514

- Each sector has a control center
- Centers may have few or many (50) controllers
- Data comes from a radar system that broadcasts updates every 10 seconds
- Database keeps other flight data
- Controllers each "own" smaller sub-sectors

Why build a new one?

CS514

- Much technology still dates to 1960's
  - Overloaded computers that often crash
  - Getting slow as volume of air traffic rises
  - Inconsistent displays a problem: phantom planes, missing planes, stale information
- Many major issues prior to 9/11/2001
  - Long downtimes triggered by small failures
  - Near-misses were becoming common
  - Lost ATC for 12 minutes over mid-West
- So the pressure to upgrade is huge

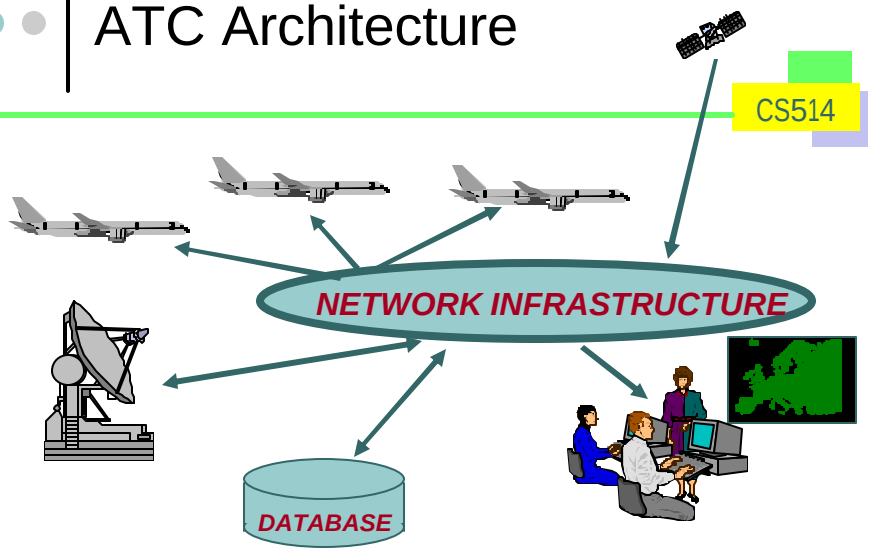
Concept behind the AAS

CS514

- Replace video terminals with workstations
- Build a highly available real-time system guaranteeing no more than 3 seconds downtime per year
- Offer much better user interface to ATC controllers, with intelligent course recommendations and warnings about future course changes that will be needed

ATC Architecture

CS514



Technologies Used

CS514

- AAS proposal: Build using standard, off-the-shelf workstations
  - A version of Unix
  - Goal was that this should be easier to maintain, upgrade, manage
  - IBM proposed a software scheme for fault-tolerance and a very modular architecture
- Fancy graphical user interface much like the Web, pop-up menus for control decisions…

Sample “tough problems”

CS514

- How to guarantee high availability?
  - IBM proposal: use “replication” to make spare copies of critical services
  - Use reliable multicast to keep replicas in sync, doing this in real-time
- If a critical server fails, the clients “fail over” to the backup

Sample tough problems

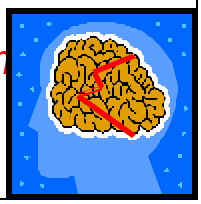
CS514

- How to detect a failure?
  - Needed in the reliable multicast
    - You should “give up” on sending an update to a failed replica, but not one that is merely slow or temporarily flakey
  - Needed for fail-over too
    - Clients will “fail over” if server crashes
- Notice how a *consistency requirement* has snuck into our problem!

Consistency issues

CS514

- Sample questions:
  - Who's up? Who crashed? When?
  - What is the current state of a replicated data item, such as "the flight plan for flight US Airways 271"?
  - Suppose the server handling some critical task *t* fails. Who should take over?
- Unless we have a general approach to solving such problems, our system could exhibit *split brain syndrom*



split brain syndrome

CS514

- Happens when something that needs a single answer from a single place ends up with multiple programs independently formulating answers
- E.g. "is it safe to route a plane into this part of the sky?"
- Don't want *two* programs in charge of *one* chunk of sky!

Anyhow… AAS was a fiasco!!

CS514

- IBM unable to implement their fault-tolerant software architecture!
  - Problem was much harder than they expected.
  - From day one schedule "derailed"
- IBM scaled back by focusing purely on the controller's workstation, yet even with this limited objective, project failed
  - Resulting system was unsatisfactory
  - US abandoned it. British accepted delivery

French Project

CS514

- This was much more of a success
- They used Ken's "Isis Toolkit"…
  - It solved the split brain and replication / high availability problems…but that wasn't *really* why they succeeded
  - The key was that they were more systematic about robustness issues
- In CS514 we'll look at the kinds of technology and also the *ways of thinking* that made the French ATC system a success while the IBM effort fumbled

Mission critical applications are common

CS514

- Banking, stock markets, stock brokers
- Heath care, hospital automation
- Control of power plants, electric grid
- Telecommunications infrastructure
- Electronic commerce and electronic cash on the Web (very important emerging area)
- Corporate “information” base: a company’s memory of decisions, technologies, strategy
- Military command, control, intelligence systems

Bottom line: people depend on distributed systems!

CS514

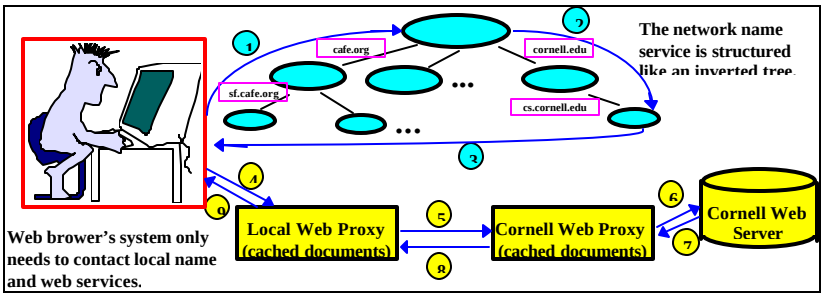
- If these critical systems don’t work
  - When we need them
  - Correctly
  - Fast enough
  - Securely and privately
- ... then revenue, health and safety, and national security may be at risk!
- Goal of CS514: Learn to approach such problems in a “mature”, smart way

But what *really* makes it hard?

CS514

- Web Services grow out of existing Web
  - Except that people behind browsers are replaced by applications running on “client computers”
- So: how hard would it be to build a “reliable Web site”?
- Key is to realize that these run on Internet and the Internet has “issues”

Confusing behavior… a timeout



- A few representative possible causes (not exhaustive):
  - Domain name service (DNS) can overload (1-3)
  - Server or proxies can overload, crash (4-9)
  - Communication lines can overload or break
  - DNS or proxy can return “stale” data

Web Server's perspective

CS514

- Process *a* was talking to process *b*
- Suddenly *b* became unresponsive
  - If *b* failed, *a* should fail-over to *c*
  - If *network* failed, *a* should wait a while and perhaps report something to the user
  - If *a* itself failed, it may need to go offline for repair…
- Which happened? How can we tell?

Overcoming Web problems

CS514

- Doesn't look easy!
  - Some stem from the Internet
    - We can't really fix this right now!
    - Can we work around its limitations?
  - Some are from dependence on things that could fail
    - Can we replicate critical components?
  - Some are security issues
  - Even diagnosis of a problem can be hard!
- Unless we can solve such problems, how can Web Services ever be "robust"?

Major components of a typical modern "platform"

CS514

- Network layer
  - Connections to the outside world… includes firewalls, determines the ways that objects communicate, imposes limitations such as constraints on movement of code or objects

Major components of a typical modern "platform"

CS514

- Network layer
- Communication tools
  - Built in mechanisms such as the Web Services protocols (UDDI, SOAP, XML, etc), Java RMI or C# method invocation
  - Mechanisms for finding important objects or other resources that may be needed
  - Technologies like publish-subscribe message buses or JMS for standardizing rendezvous…

Major components of a typical modern “platform”

CS514

- Network layer
- Communication tools
- Object orientation
  - Modern systems provide ways to create “packages” of various kinds
  - These are treated as objects, even if large
  - Standards determine ways of talking to objects, events objects may see while running, persistence features, etc

Major components of a typical modern “platform”

CS514

- Network layer
- Communication tools
- Object orientation
- Naming
  - There are many standards by which components can be “named”
  - How can “a” find “b”, which it needs for such and such a task?
  - Mobility is creating major new challenges

Major components of a typical modern “platform”

CS514

- Network layer
- Communication tools
- Object orientation
- Naming
- Caching and Replication
  - Often, locality of data is key to performance
  - But if data is copied to keep it local, how can updates be handled?
  - Modern platforms have elaborate caching or replication subsystems…

A course like this could be about…

CS514

- What all the acronyms mean
- The specific building blocks you might need to use when building a commercial quality application with .NET
- How to get reasonable performance in a networked, componentized world

What this course is *actually* about

CS514

- We will try to appreciate what the various acronyms mean and what the standards "do", but…
  - …distributed computing is rapidly transforming the way we work, live, the way that companies do business.
  - Increasingly, distributed computing systems are the only ones you can buy.
- Our challenge: devise ways to build distributed systems that can be relied upon in critical settings

What's the Story Today?

CS514

- If you build on .NET or J2EE you'll end up with failure-prone solutions
- But the methods that promote
  - Fault-tolerance
  - Security
  - Predictability
  - Scalability
- … simply don't exist as options within .NET or J2EE!

Honest goals for this course?

CS514

- Understand the basic technologies from which distributed systems are constructed
- Maintain a degree of emphasis on reliability issues throughout: how reliable are the standard technologies? Can they be used reliably despite their limitations?
- Look at advanced technologies in context of real systems built in standard ways
- But not stretch so far that we're doing research on the fly…

Trends are changing

CS514

- More and more pressure on industry
  - When the network is down, your company won't make money
  - Clients want tools they can rely on
- This is creating pressure on vendors who offer middleware
  - Result is a new emphasis on scalability and reliability
  - Indeed, Bill Gates has a major thrust in this area underway at Microsoft, and one on security
- Hopefully, the insights we take away will help us all be better voices pushing for change and will help us make smart choices when faced with competing options

Technologies we will cover

CS514

- Component structure of modern computing systems
- Java and J2EE. C# and .NET. How do they differ? How are they similar?
- Internet technologies (email, news, msg. bus) and trends
- Naming things, finding things, caching things, replication
- Web Service technologies, scalability issues, clusters and data centers
- Network-level trends: Behavior of the Internet WAN, firewalls and NATs and issues they raise, mobility
- Event-driven architectures. Scalable event routing using multicast and content-based routing mechanisms. Publish-subscribe message buses
- Transactions and reliability
- Just a Taste of Security
- Hot new trends: Autonomic and Grid Computing

Reliability, Scalability

CS514

- Becoming hot issues for industry
- Basically, customers expect solutions that
  - Can be developed on a small scale
  - Continue to work during prime-time
  - Scalability and stability: can be considered from many dimensions
- Web Services just aren't giving the necessary properties, yet are being rolled out at an accelerating pace

The Prevailing Mindset

CS514

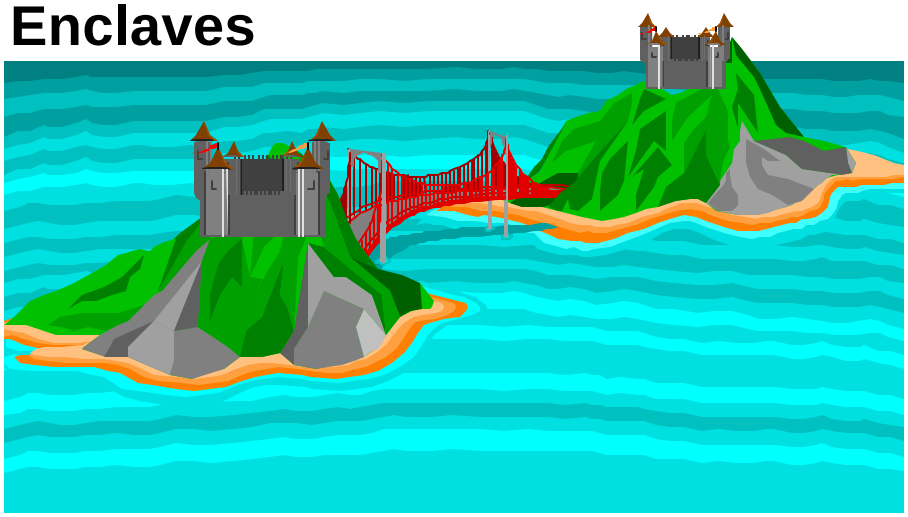
- Many developers believe that reliable systems are clumsy, overengineered, slow
- Image: a “robust bridge”. Sounds like some sort of ugly, heavy eyesore
- The Web and the Net are about elegant, light-weight, fast systems: “antithesis” of robust ones
- Reliability is also at odds with using standard components and packages

Insights From Course?

CS514

- Reliability techniques are often very elegant
- Complexity is a challenge; modularity used to control these costs
- Can achieve high performance in reliable distributed systems
- ... but they sometimes are hard to combine with standard technologies

Lightweight but Resilient Bridges, Secure Computing Enclaves



Course Summary: four major chunks of material

CS514

1. A broad overview of the technology options
2. Drilling down on platform issues seen in data centers, other large systems
3. Pushing into the network. Trends, tools, challenges. Fancy options like reliable multicast
4. New developments – tools for large-scale system monitoring and management, autonomic computing and global grid computing, etc.

Work we expect you to do

CS514

- Homeworks, most weeks… but short. 10% *total*
- Prelims: two, take-homes, 20% of your grade *each*
- A reasonably ambitious software project (can be used to satisfy your MEng project requirement). 50%
  - Projects can be done in groups
  - This semester we're suggesting two options
    - A Web Services project that adds a fault-tolerance mechanism. Visual C# for .NET using ASP.NET "template" for a basic Web Service and client
    - A network-level file-system problem with an emphasis on performance issues. C on Linux

You'll teach yourself how to use the necessary tools and you'll independently read documentation. We use powerful tools, but we won't hold your hands!

Major Themes Projects Should Demonstrate

CS514

- Modularity. Better structured systems are more reliable.
- Performance. Technologies need to be fast to be perceived as working well
- Fault-tolerance. If companies will depend on systems to give 24x7 availability, you need to know how to make it happen
- Rigor. We want to know why a technique works: ad-hoc solutions often break under stress

Recommended Reading

CS514

- Our Web page has readings for each lecture. Keep up on your reading!
- We'll sometimes use additional papers on topics the book doesn't tackle
- And you need to get familiar with the online resources from MSFT and Linux