

CS514: Intermediate Course in Operating Systems

Professor Ken Birman
Krzysz Ostrowski: TA

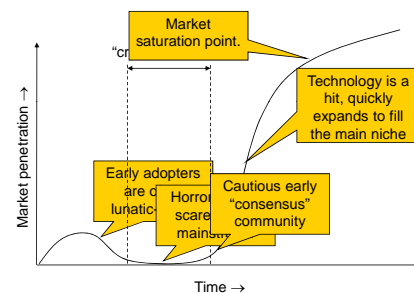
Putting it all together

- Today is our last lecture!
 - Wednesday was originally used as an in-class final by Professor Schneider, but we don't have an exam this year
 - People interested in doing an early demo are encouraged to do so, Wednesday or any time in the next two weeks
 - All group members must be there!

Today's topic: "Synthesis"

- Let's look back over the semester
 - What's the big picture to take away?
 - Where will complex systems of systems go next?
 - What kinds of bets on the future are starting to emerge right now?

Technology adoption curve



The world we live in?

- We're seeing Web 1.0 reaching that saturation situation
 - For desktop uses, the web is probably doing much of what it "will do"
 - For wireless and mobile, of course, the situation is very different
 - And we're using Web to mean "web sites with relatively static content"

The world we live in?

- Meanwhile Web 2.0 is taking off
 - Technologies that leverage and support social networking
 - Google mashups, RSS feeds, search
- Arguably Web 2.0 is already hitting its own saturation point

The world we live in?

- Web Services
 - Basically, can recognize these in terms of a set of (simplistic) steps
 - Let's allow programs to do what browsers do
 - Let's use Web Services standards to build systems of systems
 - Let's make it easier to construct these solutions and interconnect them
 - Call this a Web 2.0 technology area

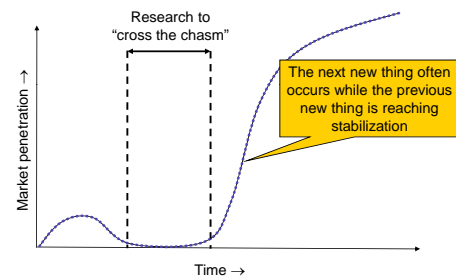
The world we live in?

- Web 3.0
 - Makes for a fun homework topic (someday you'll thank us... ☺)
 - But really just a distant glimmer right now
- The real Second Life system is just your basic datacenter, very much a Web 2.0 construct!
 - Technology to support social networks

Could Live Objects be "it"?

- Sure, if...
 - We can get them to run in a stable way in the Internet WAN, with firewalls etc.
 - And can "manage" the resulting system
 - And can figure out how to help people find the stuff they would want to pull in
 - And can get a critical mass of content... without some sort of scalability nightmare
- No problem!

Technology adoption curve




A multi-layered picture

- Over time, a technology "area" such as web services ends up having wave after wave of major technologies
- Each follows a similar curve
- (Assumes that there is a larger and larger aggregate market to pursue)


CS514 emphasis was on reliability, mostly via replication

- We looked, superficially, at the technology backdrop against which all this is happening
- Client-server interaction models
 - CORBA (we skipped this "epoch")
 - Web Services (the current new thing)
 - Systems of systems (SoS or SOAs)




CS514 emphasis was on reliability, mostly via replication

- Gossip technologies
 - Very scalable and robust, at least in some ways. Predictable, low load
 - But sluggish; poor choice if we want snappy response
- Other P2P technologies
 - BitTorrent, RON, DHTs
 - Some combine P2P ideas with gossip




CS514 emphasis was on reliability, mostly via replication

- Group communication
 - Multicast, but normally in support of replication or event notification
 - Many “types”, which leads towards a perspective that multicast “type” is a type much like any other “type”
 - Object-oriented multicast would probably look like “live distributed objects”
 - Multicast type extends the component type



CS514 emphasis was on reliability, mostly via replication

- Against this backdrop we looked at
 - What can and cannot be done (FLP)
 - Scalable best-effort multicast (QSM)
 - Virtual synchrony model
 - Consensus (Paxos model)
 - Quorums and static membership
 - Transactional replication (1SR)
 - Time-critical and real-time multicast
- Can view all of these as “types” of multicast and in fact QS/2 will do just this



CS514 emphasis was on reliability, mostly via replication

- Byzantine Agreement
 - Strongest property of all
 - Basically subsumes all the others!
- Not impossibly slow anymore (PRACTI, BASE, other BFT schemes)
- But use only for “sensitive” purposes



Giving rise to a “vision”

- Today, Web Services focuses on how to connect clients to datacenters
 - ... and more and more, how to create complex SoS structures with datacenters that talk to one-another
 - But existing platforms offer relatively little autonomic support and forces us to build datacenters more or less entirely by hand



The vision?

- Systems that are
 - Easy to build: Better tools
 - Autonomic by construction: The tools lead us to robust solutions that can manage themselves in large, complex deployments
 - The tools themselves are better integrated into environments like .NET
- Unlike cs513 we didn't look at security... but even so, add “secure” to this list

Approaching that vision

- Cornell approach:
 - We need better technology
 - Then show how it can integrate seamlessly into major platforms
 - Then hope the world will imitate us
- The problem?
 - The world is drowning in a sea of noise, technologies, buzz...

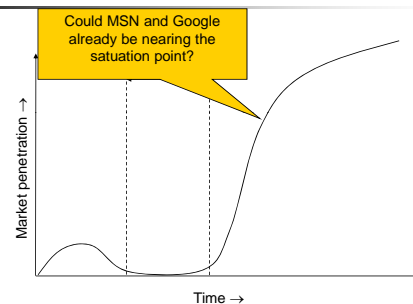
Approaching that vision

- Corporate players?
 - Google is driven mostly by search and social networking opportunities
 - Which for them, are opportunities to leverage their role by helping you find their partner's sales sites, or posting just the right ad at the right moment
 - Many betting that Google is dead on.

Approaching that vision

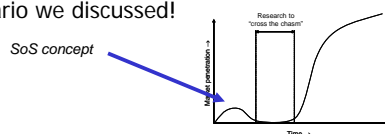
- What about Microsoft?
 - "MSN Live" intended to enter same space
 - But unclear, so far, just what the Live concept will really do
- Could "Live" be "Live distributed objects"? Cornell thinks so, but MS hasn't shown much sign of believing this
- Yet big success of .NET is its clean integration of components, clean use of type system

So... where are we?



Betting that "our time is up"

- If we bet that the datacenter/search paradigm is already close to its peak...
 - Microsoft's next bet is on systems of systems, but the technology is full of holes
 - Looks much like that "early adopter" scenario we discussed!



Betting that "our time is up"

- Google is aimed at cell phones
 - Building a national "free" network (lure in the marks with a loss-leader)
 - Faustian bargain: Just agree to run Google on your cell phone
- Then they use GPS, voice recognition, etc. to somehow get you into "their" hotels, restaurants, nightclubs, stores...

Google's problem?

- Cell phone screens are just too small
 - Already need to squint to see anything on them
 - And voice recognition doesn't work very well yet – an A/I challenge for decades with progress, but rather slowly
- Will Google pull it off?

What about us?

- We're the crowd that ends up dealing with today's challenges
- These are basically
 - Building datacenters with inadequate tools
 - Making systems self-managed even though Web Services is constantly "in our face" making the job harder than it should be
 - Creating SoS without proper standards

This is a good and a bad thing

- The good news:
 - We do have technologies that can help
- The bad news:
 - Never underestimate how hard it can be to deploy them into your app!
 - They aren't going to look very "standard" to your boss...

Good luck!

