Lessons Learned from Isis

Ken Birman

Outline

- History of Isis
- How we presented the system
- How did people really use it?
  - What worked? What didn’t?
- What to take away

A quick summary

- Use process groups as a distributed systems programming construct
  - They “structure” your system
  - And create a new way to think about distributed state
- Fault-tolerance through “toolkit”
- And Isis will make money as a middleware player

Lesson #1: People like groups

- The Isis programming model was very popular
- People liked group communication primitives. Isis has an easily-used API
  - This lesson has been lost for subsequent systems: Spread, Ensemble, Horus were all far harder to use!
  - …. and Eternal, trying to be even simpler (in a CORBA setting) is too constraining

Lesson #2: It works

- Productivity was actually very high
- And fault-tolerance in the Isis model is quite good
  - Although we deliver many events in the same order….
  - … there are enough differences to avoid correlated failures
Lesson #3: They don’t follow advice

- Isis recommended:
  - Groups, but not “too many of them”
  - And not “too many members”
- People are people. They don’t easily accept such advice

Swiss Stock Exchange Problem: Vsync. multicast is “fragile”

- Most members are healthy…
  - … but one is slow

Performance degrades as the system scales up

- SWX was forced to limit fanout to about 50 receivers.
  - Up to 50 users per "repeater"
  - Replicate for availability
  - All data filters through core system
  - One or a few repeaters at each bank or brokerage
  - 50x50 = 2500 users, maximum

New York Stock Exchange

- A far more ambitious system!
- They have
  - Far higher loads
  - Much more demanding requirements
- But… they didn’t want to replicate the entire exchange (whew!)
  - In fact, they backed into their use of Isis

NYSE goals?

- At first, goal was to link clearing systems (completed trades and certain quoted prices) to SEC monitoring
- Later: used Isis to operate the overhead display systems
Use of multicast

- In this architecture, the role of multicast is to keep the states of the forwarding nodes synchronized.
- Both trees should look the same.
- So inner nodes need to use the same filtering policy.
- There is also a need to replicate the overall “shape” of the tree.
- And multicast is employed, internally, to report failures so that the tree can be healed in a clean, consistent manner.

Non-use of multicast

- Multicast does not carry high data volumes – the TCP connections handle that.
- This works (in large part) because NYSE has a private, clean network.
- Even so, need to support “dual IP addressing” was very tough.
- Today would probably use a hardware solution (an off-the-shelf CISCO router...)
- On the WAN Internet, same architecture would have tough problems overcoming network congestion... but this gets to a topic we'll visit later in the lecture.
Other similar “problems”

- With too many groups, Isis gets slow
  - Often they overlap heavily
  - But each runs its own version of the membership protocol... redundantly
  - And Isis tries to preserve properties, like causal order, over group boundaries
- Is the Isis model “scalable”?

French ATC problem

- They wanted to use Isis for
  - Fault-tolerant console clusters
  - Radar data comes in separately. Display as a sort of “background” image
  - Use Isis to replicate data for state of planes in the air - tracks, flight id, “owner”, etc
  - System management
    - Track state of all the workstations in Toulouse
    - Inter-airport flight plan updates

Observations

- One center might have
  - 150-250 machines, in
  - 50 clusters of perhaps 3-5 each
  - Flight plan updates relevant to 3-5 control centers at most

“Fear of Isis”

- Without Isis, those 200 workstations are very loosely coupled
  - Isis couples groups of 3-5 at a time...
  - ... But the management application also links them into massive groups
  - And data rates
    - Are low for the small groups
    - But would be high for the management app.
Solution?
- Run Isis in a partitioned mode
  - Each group of 3-5 systems is a separate Isis instance
  - Loads light... membership small... an easy Isis application
  - Don’t use Isis for the system-wide membership mechanism!

Isis success stories
- The two French ATC systems (console cluster or “Phidias” and the inter-airport flight plan system - I don’t know name)
- New York and Swiss Stock Exchanges
- Florida Electric and Gas
- US Naval AEGIS warship, various black applications in military

Lesson #4: You don’t get rich
- Sadly “Middleware doesn't sell”
- And because of this, economic forces conspire ferociously against Isis and similar technologies!
- Ultimately, any middleware company
  - Either grows into a services company
  - Or dies

Terminology
- A company can sell
  - Iron. Like Dell: Priced by the pound
  - Platforms. Like Microsoft, Red Hat
  - Applications. They do stuff. Like Adobe
  - Solutions. Integrated applications with the “whole story” for some customer category
  - Itself. Some companies exist primarily to be acquired by other companies.

Middleware?
- A term used for a company that is neither selling platforms nor applications
- Often, they sell bandages! We’ll fix defects in your Linux and Windows platforms...
  - And you might even pay for this... a fraction of what you spent on Windows!

Services
- Companies like IBM, HP... and in fact, Isis, make much of their money on services
  - At IBM, roughly six to one relative to licenses for middleware
  - “Our crack team of hackers, using our superior middleware, can solve your problems better.”
When Stratus bought Isis...

- They wanted to greatly increase the flat license revenue with less services revenue
- And this was a disaster.
- In fact, it can't be done.
- Middleware gets sold to the wrong kind of customer. Not a scalable revenue stream.

Summary of lessons?

- Groups are way cool
  - But in fact, building a really scalable group mechanism remains a tough challenge even today!
  - And even if you pull it off... nobody will want to pay enough to make a profit on it
  - This is just not a good business to go into!
- But the underlying technology was deep and that was fun