FutureGRID: A Program for long term research into GRID Systems Architecture

Jon Crowcroft, Steve Hand, Tim Harris, Ian Pratt
The Computer Laboratory, University of Cambridge
Andrew Herbert, Director, Microsoft Research Cambridge

0. Introduction
- Program of work between the Computer Lab, and Microsoft Research
- Builds on existing collaborations
- Designed as a set of loosely coupled basic research projects
- Common elements to projects, which lead to understanding
- Later, full systems architecture will emerge for a Future GRID.
- PhD studentships efficient use of funds (and to be honest, we have more good applicants than money).

1. Who, where, how, what
- Collaborative tools based on Scribe and Pastry instead (or as well as) IP multicast (P2P CSCW) (existing RFC on PGM etc)
- Search based on locality and on partial content matching (publications this month)
- Computation based on large scale systems and massively redundant partition of computational problems (a.k.a. spread spectrum)
- Extension of Pasta work on mutable, persistent P2P storage (publications)

3. Peer-peer networking == GRIDng

IP Multicast - Project 1
- No duplicate packets
- Highly efficient bandwidth usage
Key Architectural Decision: Add support for multicast in IP layer

P2P-GRID networking Focus at the application level

Routers with multicast support
Concerns with IP Multicast

- Scalability with number of groups
  - Routers maintain per-group state
  - Analogous to per-flow state for QoS guarantees
  - Aggregation of multicast addresses is complicated
- Supporting higher level functionality is difficult
  - IP Multicast: best-effort multi-point delivery service
  - End systems responsible for handling higher level functionality
  - Reliability and congestion control for IP Multicast complicated
- Inter-domain routing is hard.
- No management of flat address space.
- Deployment is difficult and slow
  - ISP’s reluctant to turn on IP Multicast

End System P2P Multicast

Why is self-organization hard?

- Dynamic changes in group membership
  - Members join and leave dynamically
  - Members may die
- Limited knowledge of network conditions
  - Members do not know delay to each other when they join
  - Members probe each other to learn network related information
  - Overlay must self-improve as more information available
- Dynamic changes in network conditions
  - Delay between members may vary over time due to congestion
  - Use Pastry/Scribe P2P system as it provides precisely these characteristics...

P2P Search: basics - Project 2

Vector Space Search

- Existing systems use flat unstructured keys
  - Let’s extend this to a virtual multi-dimensional space
- Entire space is partitioned amongst all the nodes
  - Every node “owns” a zone in the overall space
  - Self-stabilizing mechanisms manage nodes entering and exiting from the system
- Abstraction:
  - Keys can be represented as “points” in the space (perhaps with associated values)
  - Messages can be routed for a particular key to the node that owns that “point”
Vector Space Search: applications

- Resource discovery:
  - Points represent resource requirements of jobs and resource availability of machines
  - Nodes act as brokers between jobs and systems that can host them
- Network position could be reflected in the broker’s co-ordinates
  - Promote scalability through disjoint operation of user communities when requests are satisfied by local facilities

Spread Spectrum Computing - Project 3

- Use redundancy coding ideas
- For code and data,
- Dissemination uses high degrees of replication
- Collection of responses is
  - Distributed (P2P)
  - Fault tolerant (like SETI@Home) and the set of ideas in a lot of cryptanalysis work recently
- Highly Optimised Tolerance (c.f. John Doyle’s work at CalTech).

Global Storage - Project 4

- Available anywhere, anytime - and fast!
- Must cope with node and network failures
  - Use replication, information dispersal codes
- Must cope with ‘flash crowds’
  - Automatic load balancing and distribution
- Must allow local caching for performance
  - Challenge of maintaining consistency
- Must provide ‘hands free’ administration
  - Self-organizing system

Global Storage with Pasta

- Uses P2P Distributed Hash Table techniques
  - More complex structures necessary? B*trees?
- Aims to provide traditional file-system like semantics (incl. efficient mutability, quotas)
- Also, wider look at shared workspaces to support ad-hoc collaboration
  - Not all participants fully trusted...
  - Need versioning, ‘views’ and ‘overlaying’
  - Object-specific locking and atomicity enforced by storage system