OceanStore
Status and Directions
ROC/OceanStore Retreat 1/13/03

John Kubiatowicz
University of California at Berkeley
Everyone’s Data, One Utility

- **Millions of servers, billions of clients ...**
  - 1000-YEAR durability (excepting fall of society)
  - Maintains Privacy, Access Control, Authenticity
  - Incrementally Scalable ("Evolvable")
  - Self Maintaining!

- Not quite peer-to-peer:
  - Utilizing servers in infrastructure
  - Some computational nodes more equal than others
OceanStore Data Model

• Versioned Objects
  - Every update generates a new version
  - Can always go back in time (Time Travel)
• Each Version is Read-Only
  - Can have permanent name
  - Much easier to repair
• An Object is a signed mapping between permanent name and latest version
  - Write access control/integrity involves managing these mappings
Self-Verifying Objects

$AGUID = \text{hash\{name+keys\}}$

Heartbeat: \{\text{AGUID, VGUID, Timestamp}\}_{\text{signed}}

Patrick Eaton: Discussions of the Future formats
The Path of an OceanStore Update

Second-Tier Caches

Multicast trees

Inner-Ring Servers

Clients
OceanStore Goes Global!

• Planet Lab global network
  - 98 machines at 42 institutions, in North America, Europe, Australia (~ 60 machines utilized)
  - 1.26Ghz PIII (1GB RAM), 1.8Ghz PIV (2GB RAM)
  - North American machines (2/3) on Internet2
  OceanStore components running “globally:”

• Word on the street: it was “straightfoward.”
  - Basic architecture scales
  - Lots of Communications issues (NAT, Timeouts, etc)
  - Locality really important

• Challenge: Stability and fault tolerance!

• Dennis Geels: Analysis (FAST 2003 paper)

• Steve Czerwinski/B. Hoon Kang: Tentative Updates
Enabling Technology: DOLR (Decentralized Object Location and Routing) "TAPESTRY"
Self-Organizing second-tier

- Have simple algorithms for placing replicas on nodes in the interior
  - Intuition: locality properties of network help place replicas
  - DOLR helps associate parents and children to build multicast tree
- Preliminary results show that this is effective
  - Dennis will talk about effectiveness for streaming updates
Tapestry Stability under Faults

• Instability is the common case....!
  - Small half-life for P2P apps (1 hour????)
  - Congestion, flash crowds, misconfiguration, faults

• **Must Use DOLR under instability!**
  - The right thing must just happen

• Tapestry is natural framework to exploit redundant elements and connections
  - Multiple Roots, Links, etc.
  - Easy to reconstruct routing and location information
  - Stable, repairable layer

• **Thermodynamic analogies:**
  - Heat Capacity of DOLR network
  - Entropy of Links (decay of underlying order)
Single Node Tapestry

<table>
<thead>
<tr>
<th>Application-Level Multicast</th>
<th>OceanStore</th>
<th>Other Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Interface / Upcall API</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dynamic Node Management</td>
<td>Routing Table &amp; Object Pointer DB</td>
<td>Router</td>
</tr>
<tr>
<td>Network Link Management</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transport Protocols</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
It’s Alive On Planetlab!

• Tapestry Java deployment
  - 6-7 nodes on each physical machine
  - IBM Java JDK 1.30
  - Node virtualization inside JVM and SEDA
  - Scheduling between virtual nodes increases latency

• Dynamic insertion algorithms mostly working
  - Experiments with many simultaneous insertions
  - Node deletion getting there

• Tomorrow: Ben Zhao on Tapestry Deployment
Object Location

![Graph showing object location](image)

- **RDP (min, median, 90%)**
- **Client to Obj RTT Ping time (1ms buckets)**

ROC/OceanStore Jan’02
Tradeoff: Storage vs Locality

Tapestry locality optimization - Opt #1 - 1 hop

Tapestry locality optimization - Opt #1 & #2 - 1 hop

Tomorrow: Jeremy Stribling on Locality
Fraction of Blocks Lost per Year (FBLPY)

- Exploit law of large numbers for durability!
- 6 month repair, FBLPY:
  - Replication: 0.03
  - Fragmentation: $10^{-35}$
The Dissemination Process: Achieving Failure Independence

- Introspection
- Model Builder
- Human Input
- Network Monitoring
- Set Creator
- Probe
- Type
- Inner Ring
- Set
- Fragments
- Inner Ring
- Fragments
Active Data Maintenance

- Tapestry enables “data-driven multicast”
  - Mechanism for local servers to watch each other
  - Efficient use of bandwidth (locality)
Project Seagull

• Push for long-term stable archive
  - Fault Tolerant Networking
  - Periodic restart of servers
  - Correlation analysis for fragment placement
  - Efficient heart-beats for fragment tracking
  - Repair mechanisms

• Use for Backup system
  - Conversion of dump to use OceanStore
  - With versioning: yields first-class archival system

• Use for Web browsing
  - Versioning yields long-term history of web sites
PondStore Prototype
First Implementation [Java]:

- Event-driven state-machine model
  - 150,000 lines of Java code and growing
- Included Components
  - DOLR Network (Tapestry)
    - Object location with Locality
    - Self Configuring, Self Repairing
  - Full Write path
    - Conflict resolution and Byzantine agreement
  - Self-Organizing Second Tier
    - Replica Placement and Multicast Tree Construction
  - Introspective gathering of tacit info and adaptation
    - Clustering, prefetching, adaptation of network routing
  - Archival facilities
    - Interleaved Reed-Solomon codes for fragmentation
    - Independence Monitoring
    - Data-Driven Repair
- Downloads available from [www.oceanstore.org](http://www.oceanstore.org)
Event-Driven Architecture of an OceanStore Node

- Data-flow style
  - Arrows Indicate flow of messages
- Potential to exploit small multiprocessors at each physical node
Working Applications
MINO: Wide-Area E-Mail Service

- Complete mail solution
  - Email inbox
  - Imap folders

OceanStore Objects
Riptide: Caching the Web with OceanStore
Other Apps

• Long-running archive
  - Project Segull

• File system support
  - NFS with time travel (like VMS)
  - Windows Installable file system (soon)

• Anonymous file storage:
  - Nemosyne uses Tapestry by itself

• Palm-pilot synchronization
  - Palm data base as an OceanStore DB

• Come see OceanStore demo at Poster Session: IMAP on OceanStore/Versioned NFS
Future Challenges

• Fault Tolerance
  - Network/Tapestry layer
  - Inner Ring
• Repair
  - Continuous monitoring/restart of components
• Online/offline validation
  - What mechanisms can be used to increase confidence and reliability in systems like OceanStore?
• More intelligent replica management
• Security
  - Data Level security
  - Tapestry-level admission control
• “Eat our Own Dogfood”
  - Continuous deployment of OceanStore components
• Large-Scale Thermodynamic Design
  - Is there a science of aggregate systems design?
OceanStore Sessions

http://10.0.0.1/

- **ROC**: Monday (3:30pm – 5:00pm)
  - OceanStore Pond Deployment
  - Evolution of Data Format and Structure
  - Tentative Updates

- **Shared**: Monday (5:30pm – 6:00pm)
  - OceanStore Long-Term Archival Storage

- **Sahara**: Tuesday (8:30am-9:10am)
  - Tapestry status and deployment
  - Peer-to-peer Benchmarking (Chord/Tapestry)
  - Tapestry Locality Enhancement

- **Sahara**: Tuesday (11:35-12:00am)
  - Peer-to-peer APIs
For more info:
http://oceanstore.org

- OceanStore vision paper for ASPLOS 2000
  “OceanStore: An Architecture for Global-Scale Persistent Storage”

- OceanStore Prototype (FAST 2003):
  “Pond: the OceanStore Prototype”

- Tapestry algorithms paper (SPAA 2002):
  “Distributed Object Location in a Dynamic Network”

- Upcoming Tapestry Deployment Paper (JSAC)
  “Tapestry: a Global-Scale Overlay for Rapid Service Deployment”

- Probabilistic Routing (INFOCOM 2002):
  “Probabilistic Location and Routing”

- Upcoming CACM paper (not until February):
  - “Extracting Guarantees from Chaos”