SDIMS: A Scalable Distributed Information Management System

Praveen Yalagandula and Mike Dahlin
Laboratory for Advanced Systems Research, The University of Texas at Austin

SDIMS: A Basic Building Block

SDIMS : Fundamental distributed systems basic building block that gathers information on a large-scale networked system.

Core components of Distributed Applications:

Monitoring, Querying, and Reacting to changes in the state

Example applications :

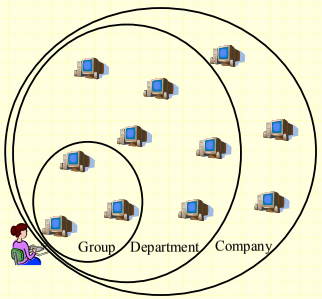
- System administration and management
- File location service
- Service location and placement
- Multicast tree construction
- Sensor monitoring and control
- Naming and request routing
- Leader election
- Barrier synchronization
- Voting
- …

SDIMS : A "distributed systems backbone" and facilitate development and deployment of distributed services

Hierarchical Aggregation

Fundamental abstraction for scalability

- Detailed views of nearby information
- Summary views at different levels of a hierarchical tree



Challenge 1: Scalability

SDIMS should be scalable with both nodes and attributes

SDIMS basic building block ➔

- Scale to enterprise sizes : large number of nodes
- Scale to several applications : large number of attributes

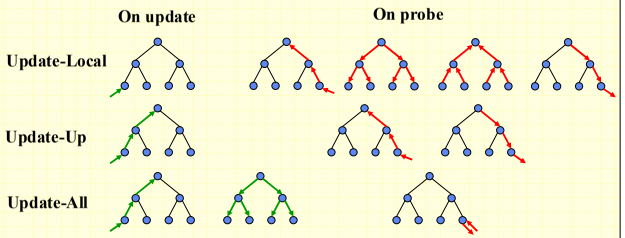
Our Approach: Use Distributed Hash Tables

- Aggregate an attribute along the tree for key = hash(*attrName*)

Challenge 2: Flexibility

SDIMS should enable applications to tradeoff update cost vs. access cost

- SDIMS general building block ➔ support different applications
- Support different workloads with different read-to-write ratios: e.g., CPU Load – low, file location – moderate, *numMachines* – high
- Support flexible aggregate propagation methods



Our Approach: Flexible API – Install, Update and Probe

Challenge 3: Autonomy

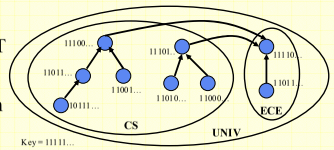
SDIMS should provide administrative autonomy

Why we want administrative autonomy?

- Security – from malicious behavior of nodes outside domain
- Isolation – from failing nodes outside domain

Our Approach:

- Simple augmentations to DHT construction for autonomy
- Satisfy Path Locality and Path Convergence



Challenge 4: Robustness

SDIMS should handle reconfigurations

- Handle node joins, leaves and failures
- Provide eventual consistency guarantees
- Enable application to tradeoff bandwidth for consistency

Our Approach:

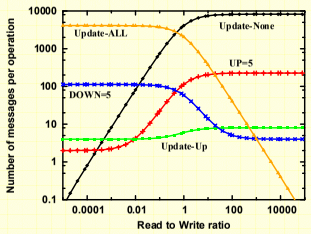
- Default lazy re-aggregation ➔ eventual consistency
- Application controlled fast re-aggregation

Initial Results

Prototype implementation based on Pastry

Scalability and Flexibility ➔

Autonomy and Robustness (see web site)



Future Directions

- Robustness against frequent reconfigurations
- Handling composite queries
- Self-tuning adaptation
- Caching support
- Bounding accuracy of the response
- Auditing the aggregated value
- Query and install policies : capabilities, etc.

Further Information

URL: http://www.cs.utexas.edu/users/ypraveen/sdims

Email: ypraveen@cs.utexas.edu