the globus alliance
www.globus.org

Overview of GT4 Data Services

Bill Allcock, ANL

Ann Chervenak, USC-ISI

Neil P. Chue Hong

GlobusWORLD 2005

Globus Data Services Talk Outline

Summarize capabilities of the following data services in the Globus Toolkit Version 4.0

- GridFTP
- The Reliable File Transfer Service (RFT)
  - Data movement services for GT4
- The Replica Location Service (RLS)
  - Distributed registry that records locations of data copies
- The Data Access and Integration Service (DAIS)
  - Service to access relational and XML databases

Vision for data services in WS-RF and plans for 2005

the globus alliance
www.globus.org

GridFTP and Reliable File Transfer Service (RFT)

Bill Allcock, ANL

What is GridFTP?

- A secure, robust, fast, efficient, standards based, widely accepted data transfer protocol
- A Protocol
  - Multiple Independent implementation can interoperate
    - This works. Both the Condor Project at Uwis and Fermi Lab have home grown servers that work with ours.
    - Lots of people have developed clients independent of the Globus Project.
- The Globus Toolkit supplies a reference implementation:
  - Server
  - Client tools (globus-url-copy)
  - Development Libraries

GridFTP: The Protocol

- FTP protocol is defined by several IETF RFCs
- Start with most commonly used subset
  - Standard FTP: get/put etc., 3rd-party transfer
- Implement standard but often unused features
  - GSS binding, extended directory listing, simple restart
- Extend in various ways, while preserving interoperability with existing servers
  - Striped/parallel data channels, partial file, automatic & manual TCP buffer setting, progress monitoring, extended restart

GridFTP: The Protocol (cont)

- Existing standards
  - RFC 959: File Transfer Protocol
  - RFC 2228: FTP Security Extensions
  - RFC 2389: Feature Negotiation for the File Transfer Protocol
  - Draft: FTP Extensions
  - GridFTP: Protocol Extensions to FTP for the Grid
    - Grid Forum Recommendation
    - GFD.20
    - http://www.ggf.org/documents/GWD-R/GFD-R.020.pdf

wuftpd based GridFTP

Existing Functionality

- Security
- Reliability / Restart
- Parallel Streams
- Third Party Transfers
- Manual TCP Buffer Size
- Server Side Processing
- Partial File Transfer
- Large File Support
- Data Channel Caching
- Integrated Instrumentation
- De facto standard on the Grid

New Functionality in 3.2

- Server Improvements
  - Structured File Info
    - MLST, MLSD
  - checksum support
  - chmod support
- globus-url-copy changes
  - File globbing support
  - Recursive dir moves
  - RFC 1738 support
  - Control of restart
  - Control of DC security

GT4 GridFTP Implementation

- 100% Globus code. No licensing issues.
- GT3.2 Alpha had a very minimal implementation
- The latest development release has a very solid alpha.
- wuftpd specific functionality, such as virtual domains, is NOT present
- Has IPV6 support included (EPRT, EPSV), but we have limited environment for testing.
- Based on XIO
- Extremely modular to allow integration with a variety of data sources (files, mass stores, etc.)
- Striping support is provided in 4.0

Striped Server Mode

- Multiple nodes work together *on a single file* and act as a single GridFTP server
- An underlying parallel file system allows all nodes to see the same file system and must deliver good performance (usually the limiting factor in transfer speed)
  - I.e., NFS does not cut it
- Each node then moves (reads or writes) only the pieces of the file that it is responsible for.
- This allows multiple levels of parallelism, CPU, bus, NIC, disk, etc.
  - Critical if you want to achieve better than 1 Gbs without breaking the bank

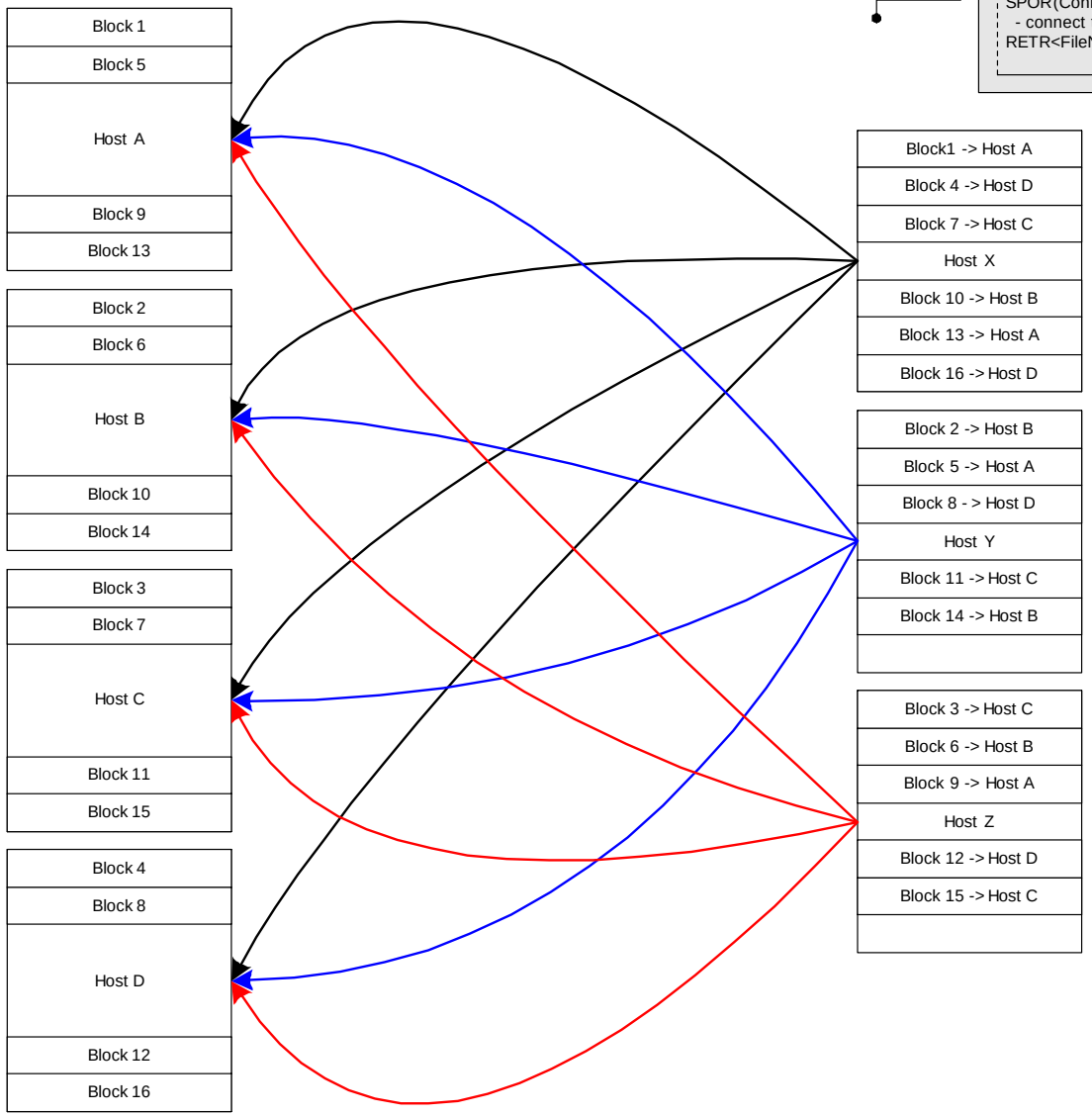
the globus alliance
www.globus.org

18-Nov-03

GridFTP Striped Transfer

MODE E
SPAS (Listen)
- returns list of host:port pairs
STOR<FileName>

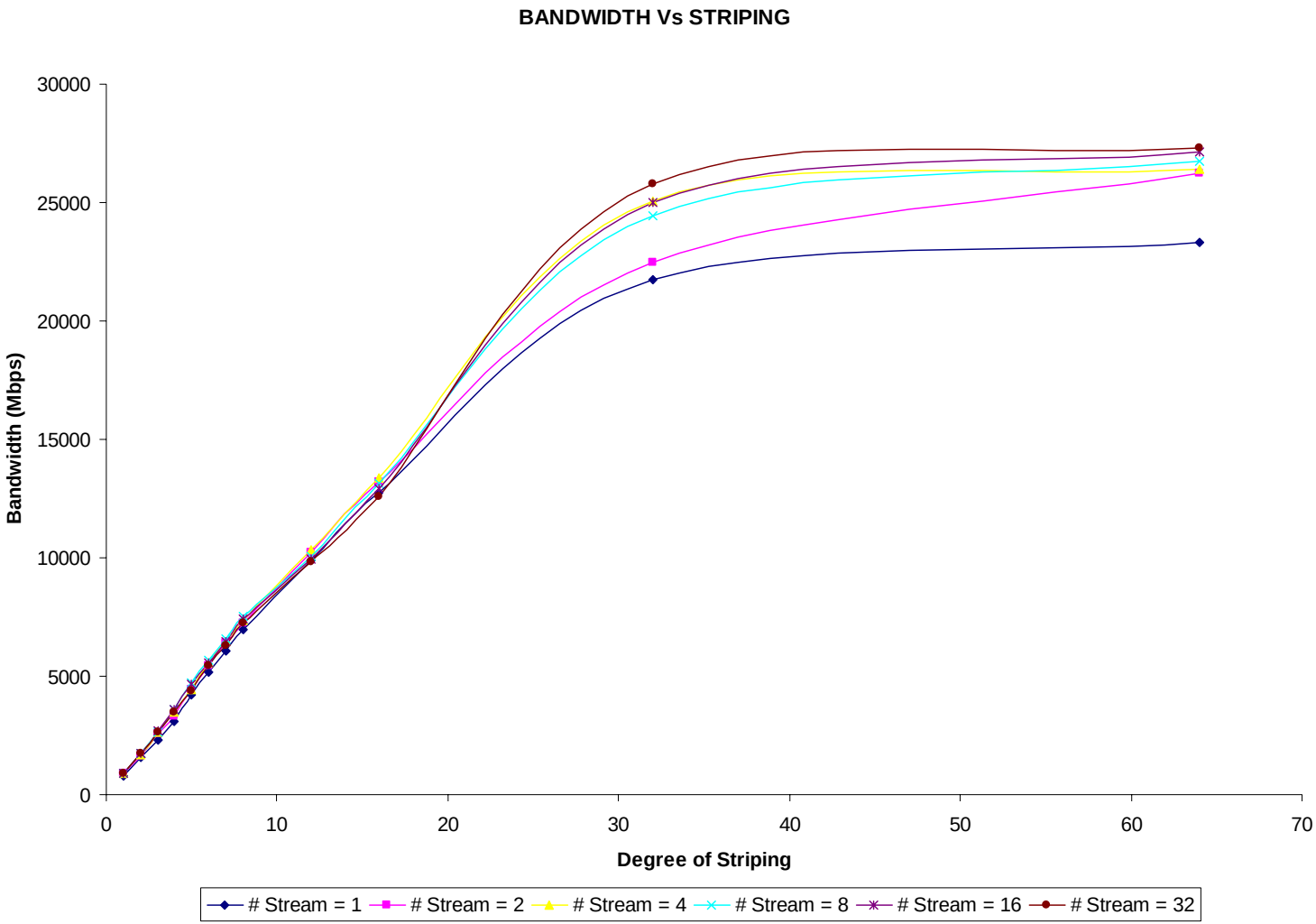
MODE E
SPOR(Connect)
- connect to the host-port pairs
RETR<FileName>



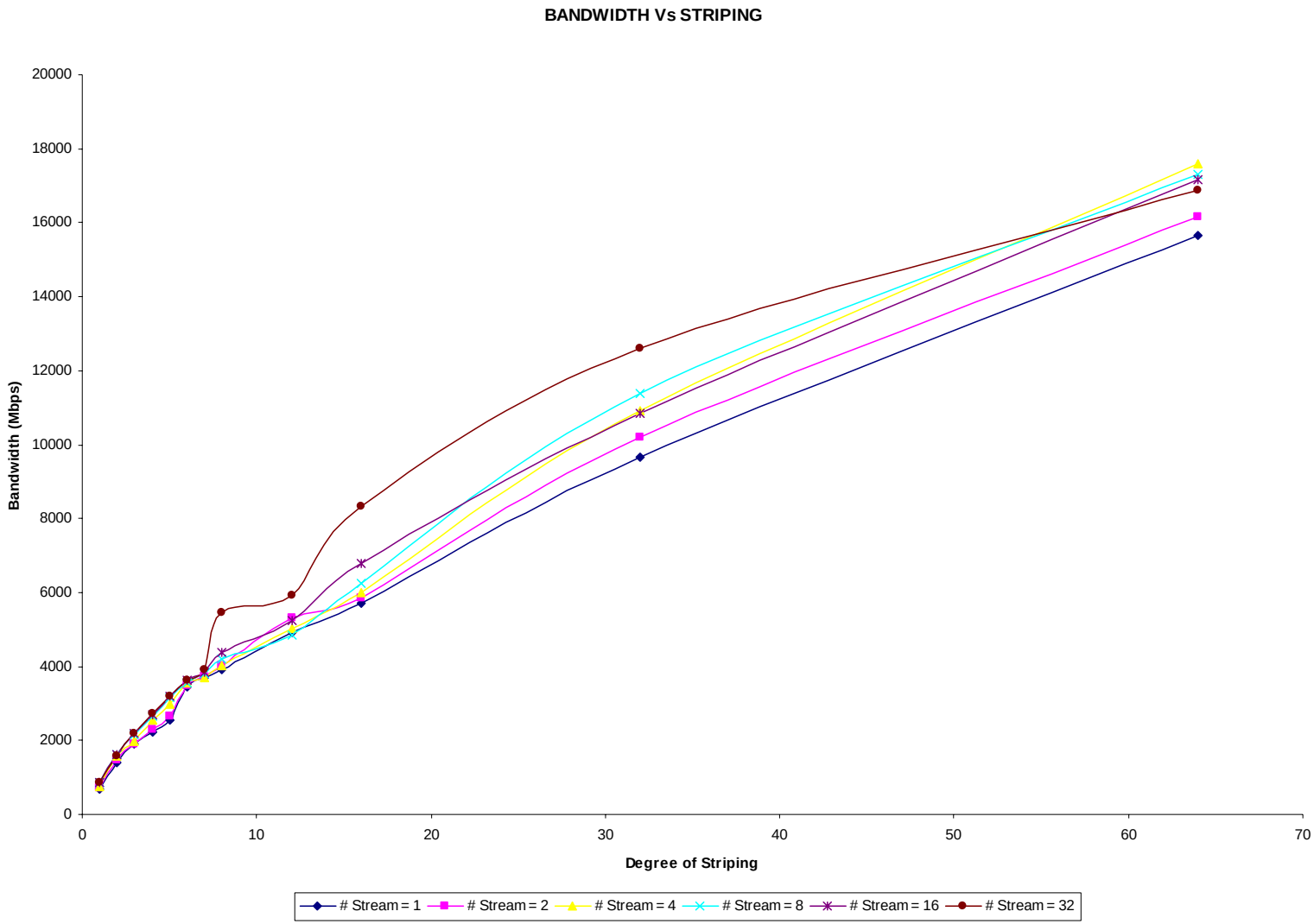
TeraGrid Striping results

- Ran varying number of stripes
- Ran both memory to memory and disk to disk.
- Memory to Memory gave extremely good (nearly 1:1) linear scalability.
- We achieved 27 Gbs on a 30 Gbs link (90% utilization) with 32 nodes.
- Disk to disk we were limited by the storage system, but still achieved 17.5 Gbs

Memory to Memory Striping Performance



Disk to Disk Striping Performance



New Server Architecture

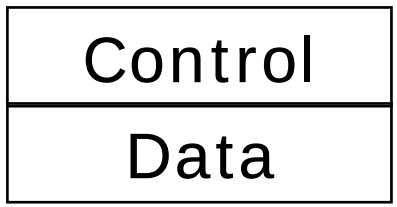
- GridFTP (and normal FTP) use (at least) two separate socket connections:
  - A control channel for carrying the commands and responses
  - A Data Channel for actually moving the data
- Control Channel and Data Channel can be (optionally) completely separate processes.
- A single Control Channel can have multiple data channels behind it.
  - This is how a striped server works.
  - In the future we would like to have a load balancing proxy server work with this.

New Server Architecture

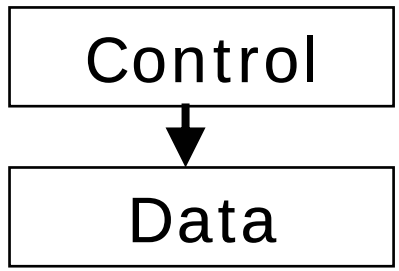
- Data Transport Process (Data Channel) is architecturally, 3 distinct pieces:
  - The protocol handler. This part talks to the network and understands the data channel protocol
  - The Data Storage Interface (DSI). A well defined API that may be replaced to access things other than POSIX filesystems
  - ERET/ESTO processing. Ability to manipulate the data prior to transmission.
    - Not implemented as a separate module for 4.0, but planned for 4.2
- Working with several groups to on custom DSIs
  - LANL / IBM for HPSS
  - UWis / Condor for NeST
  - SDSC for SRB

Possible Configurations

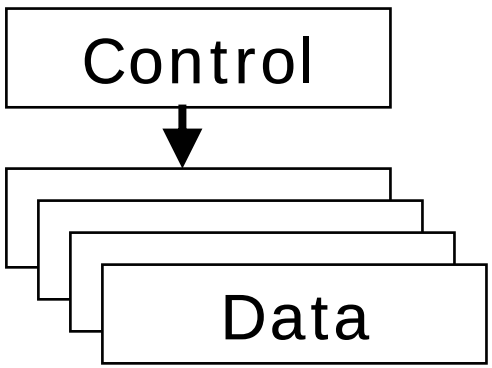
Typical Installation



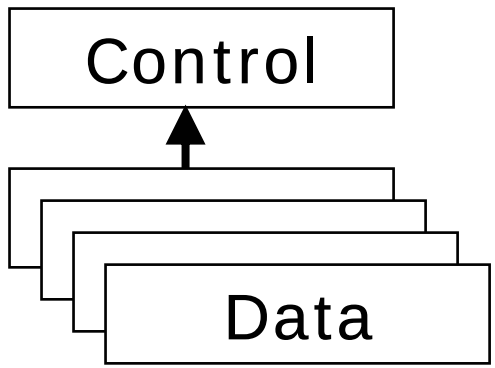
Separate Processes



Striped Server



Striped Server (future)



GridFTP: Caveats

- Protocol requires that the sending side do the TCP connect (possible Firewall issues)
  - Working on V2 of the protocol
    - Add explicit negotiation of streams to relax the directionality requirement above
    - Optionally adds block checksums and resends
    - Add a unique command ID to allow pipelining of commands
- Client / Server
  - Currently, no server library, therefore Peer to Peer type apps VERY difficult
  - Generally needs a pre-installed server
    - Looking at a “dynamically installable” server

Extensible IO (XIO) system

- Provides a framework that implements a Read/Write/Open/Close Abstraction
- Drivers are written that implement the functionality (file, TCP, UDP, GSI, etc.)
- Different functionality is achieved by building protocol stacks
- GridFTP drivers will allow 3rd party applications to easily access files stored under a GridFTP server
- Other drivers could be written to allow access to other data stores.
- Changing drivers requires minimal change to the application code.

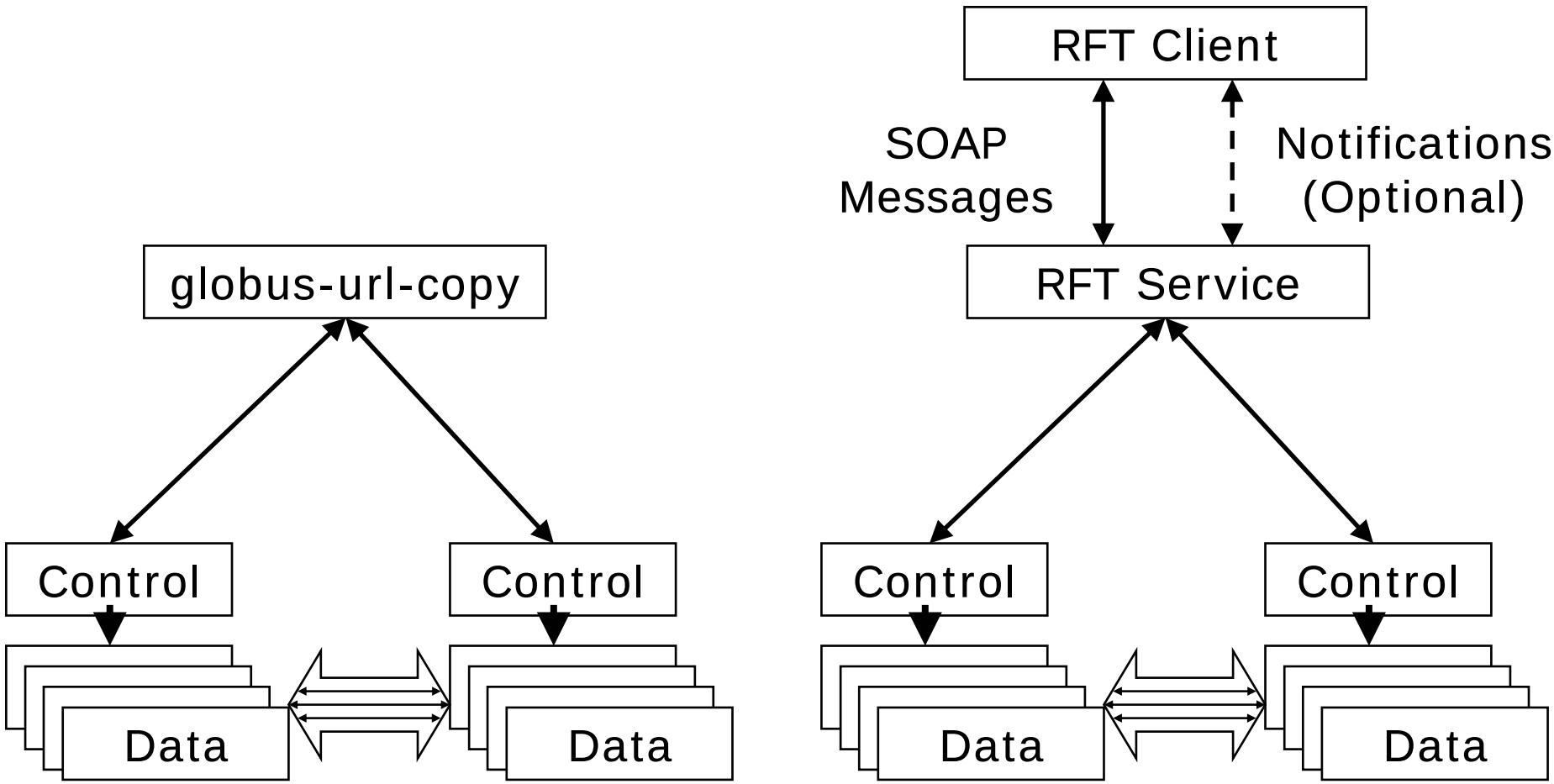
Reliable File Transfer

- Comparison with globus-url-copy
  - Supports all the same options (buffer size, etc)
  - Increased reliability because state is stored in a database.
  - Service interface
    - The client can submit the transfer request and then disconnect and go away
    - Think of this as a job scheduler for transfer job
- Two ways to check status
  - Subscribe for notifications
  - Poll for status (can check for missed notifications)

Reliable File Transfer

- RFT accepts a SOAP description of the desired transfer
- It writes this to a database
- It then uses the Java GridFTP client library to initiate 3rd part transfers on behalf of the requestor.
- Restart Markers are stored in the database to allow for restart in the event of an RFT failure.
- Supports concurrency, i.e., multiple files in transit at the same time. This gives good performance on many small files.

Data Transfer Comparison



GridFTP and RFT Plans for 2005

GridFTP

- Performance, robustness, ease of use
- Work on allowing variable stripe width
- Work on improving performance on many small files
- Access to non-standard backends (SRB, HPSS, NeST)

RFT

- Performance, robustness, ease of use
- Support for priorities
- Support for http, https, file (ala globus-url-copy)
- Add support for GridFTP changes resulting from the above

the globus alliance
www.globus.org

Design, Performance and Scalability of a Replica Location Service

Ann L. Chervenak

Robert Schuler, Shishir Bharathi

USC Information Sciences Institute

Replica Management in Grids

Data intensive applications produce terabytes or petabytes of data

- Hundreds of millions of data objects

Replicate data at multiple locations for reasons of:

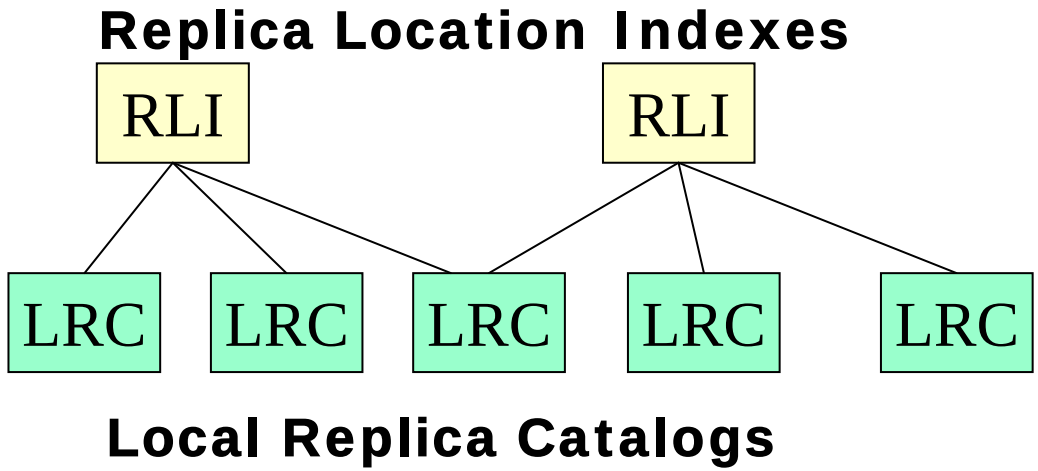
- Fault tolerance
  - Avoid single points of failure
- Performance
  - Avoid wide area data transfer latencies
  - Achieve load balancing

A Replica Location Service

- **A Replica Location Service (RLS)** is a distributed registry that records the locations of data copies and allows replica discovery
  - RLS maintains mappings between *logical* identifiers and *target names*
  - Must perform and scale well: support hundreds of millions of objects, hundreds of clients
- E.g., LIGO (Laser Interferometer Gravitational Wave Observatory) Project
  - RLS servers at 8 sites
  - Maintain associations between 3 million logical file names & 30 million physical file locations
- RLS is one component of a Replica Management system
  - Other components include consistency services, replica selection services, reliable data transfer, etc.

RLS Framework

- Local Replica Catalogs (LRCs) contain consistent information about logical-to-target mappings



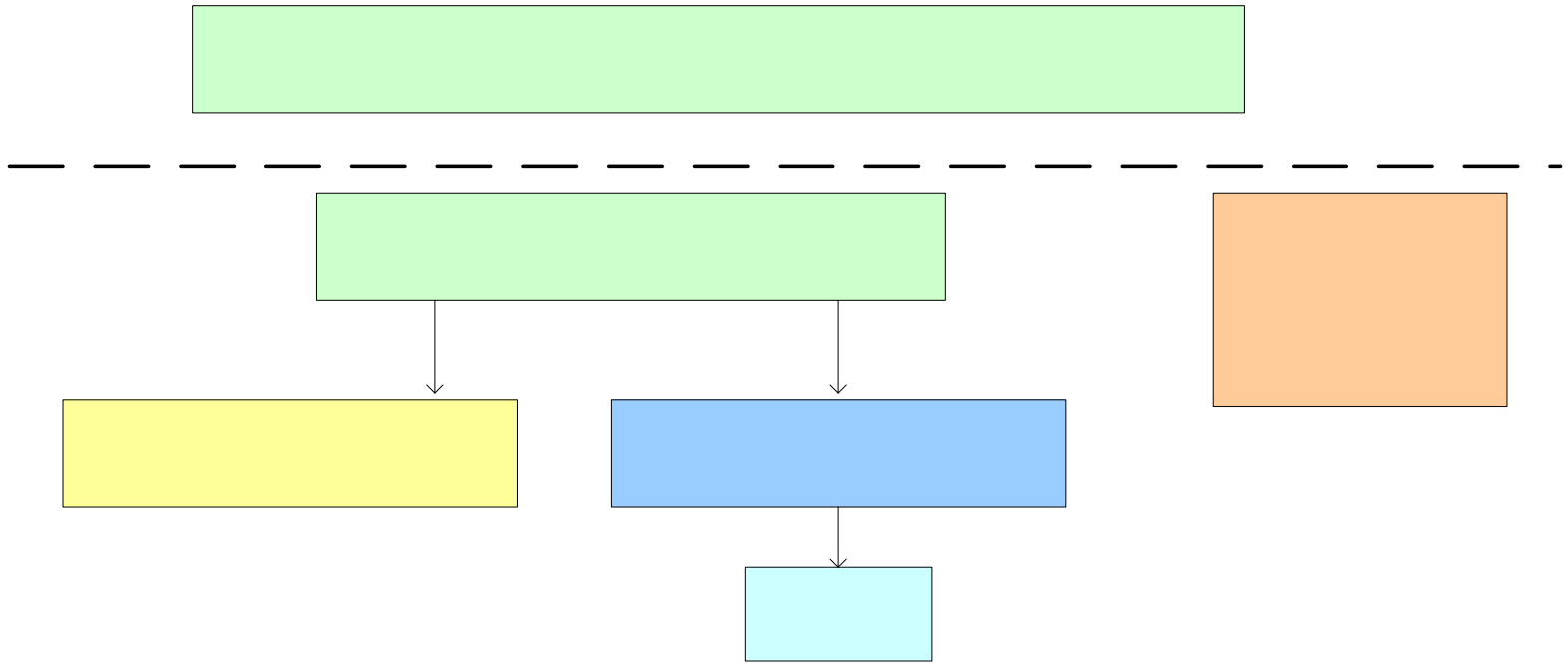
- Replica Location Index (RLI) nodes aggregate information about one or more LRCs

- LRCs use soft state update mechanisms to inform RLIs about their state: relaxed consistency of index

- Optional compression of state updates reduces communication, CPU and storage overheads

- Membership service registers participating LRCs and RLIs and deals with changes in membership

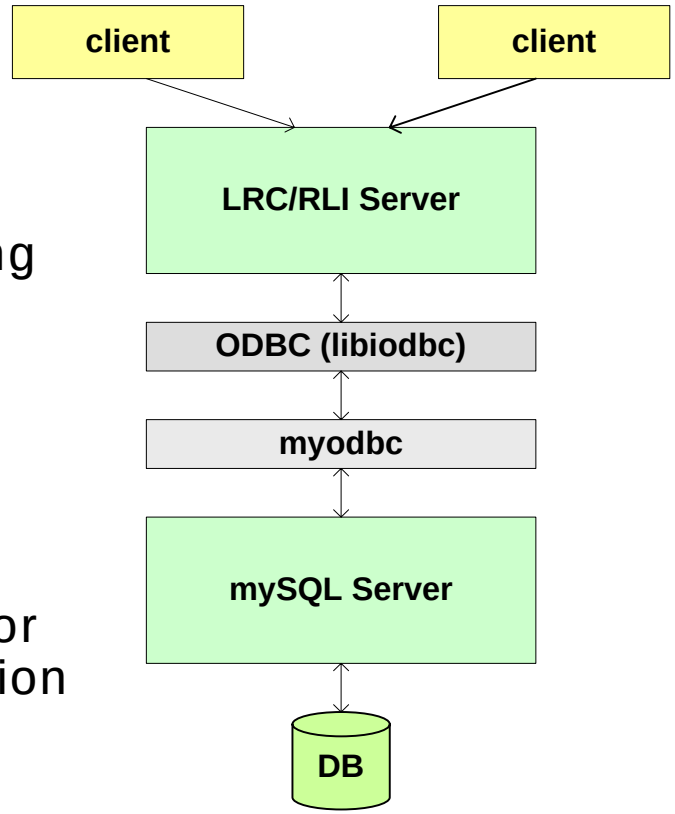
Replica Location Service In Context



- The Replica Location Service is one component in a layered data management architecture
- Provides a simple, distributed registry of mappings
- Consistency management provided by higher-level services

Components of RLS Implementation

- **Common server implementation for LRC and RLI**
- **Front-End Server**
  - Multi-threaded
  - Written in C
  - Supports GSI Authentication using X.509 certificates
- **Back-end Server**
  - MySQL or PostgreSQL Relational Database (later versions support Oracle)
  - No database back end required for RLIs using Bloom filter compression
- **Client APIs: C and Java**
- **Client Command line tool**



RLS Implementation Features

- Two types of soft state updates from LRCs to RLIs
  - Complete list of logical names registered in LRC
  - Compressed updates: Bloom filter summaries of LRC
- Immediate mode
  - Incremental updates
- User-defined attributes
  - May be associated with logical or target names
- Partitioning (without bloom filters)
  - Divide LRC soft state updates among RLI index nodes using pattern matching of logical names
- Currently, static membership configuration only
  - No membership service

Alternatives for Soft State Update Configuration

- LFN List
  - Send list of Logical Names stored on LRC
  - Can do exact and wildcard searches on RLI
  - Soft state updates get increasingly expensive as number of LRC entries increases
    - space, network transfer time, CPU time on RLI
  - E.g., with 1 million entries, takes 20 minutes to update mySQL on dual-processor 2 GHz machine (CPU-limited)
- Bloom filters
  - Construct a summary of LRC state by hashing logical names, creating a bitmap
  - Compression
  - Updates much smaller, faster
  - Supports higher query rate
  - Small probability of false positives (lossy compression)
  - Lose ability to do wildcard queries

Immediate Mode for Soft State Updates

- Immediate Mode
  - Send updates after 30 seconds (configurable) or after fixed number (100 default) of updates
  - Full updates are sent at a reduced rate
  - Tradeoff depends on volatility of data/frequency of updates
  - Immediate mode updates RLI quickly, reduces period of inconsistency between LRC and RLI content
- Immediate mode usually sends less data
  - Because of less frequent full updates
- Usually advantageous
  - An exception would be initially loading of large database

Performance Testing

- Extensive performance testing reported in HPDC 2004 paper

- Performance of individual LRC (catalog) or RLI (index) servers
  - Client program submits operation requests to server
- Performance of soft state updates
  - Client LRC catalogs sends updates to index servers

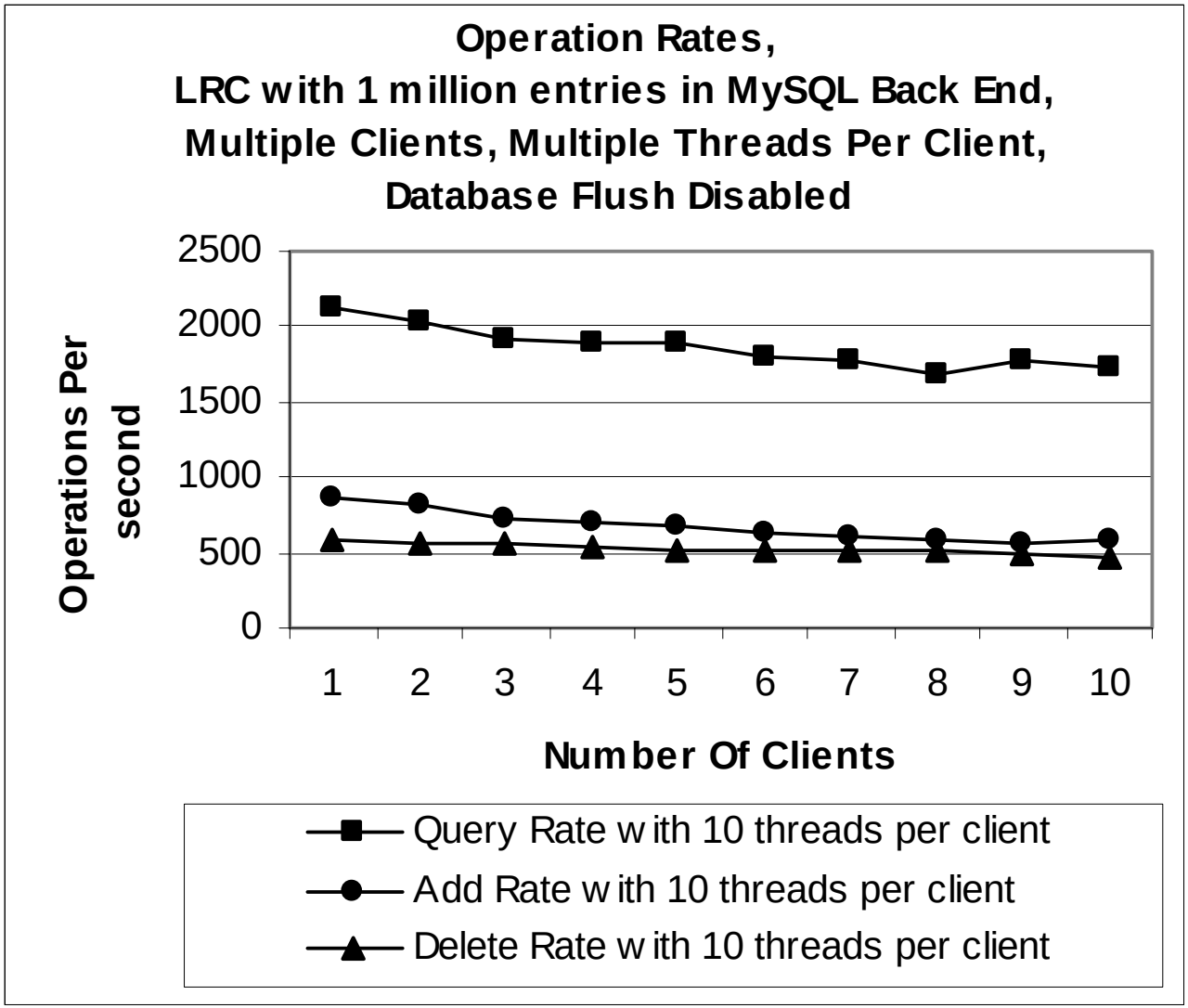
Software Versions:

- Replica Location Service Version 2.0.9
- Globus Packaging Toolkit Version 2.2.5
- libiODBC library Version 3.0.5
- MySQL database Version 4.0.14
- MyODBC library (with MySQL) Version 3.51.06

Testing Environment

- Local Area Network Tests
  - 100 Megabit Ethernet
  - Clients (either client program or LRCs) on cluster: dual Pentium-III 547 MHz workstations with 1.5 Gigabytes of memory running Red Hat Linux 9
  - Server: dual Intel Xeon 2.2 GHz processor with 1 Gigabyte of memory running Red Hat Linux 7.3
- Wide Area Network Tests (Soft state updates)
  - LRC clients (Los Angeles): cluster nodes
  - RLI server (Chicago): dual Intel Xeon 2.2 GHz machine with 2 gigabytes of memory running Red Hat Linux 7.3

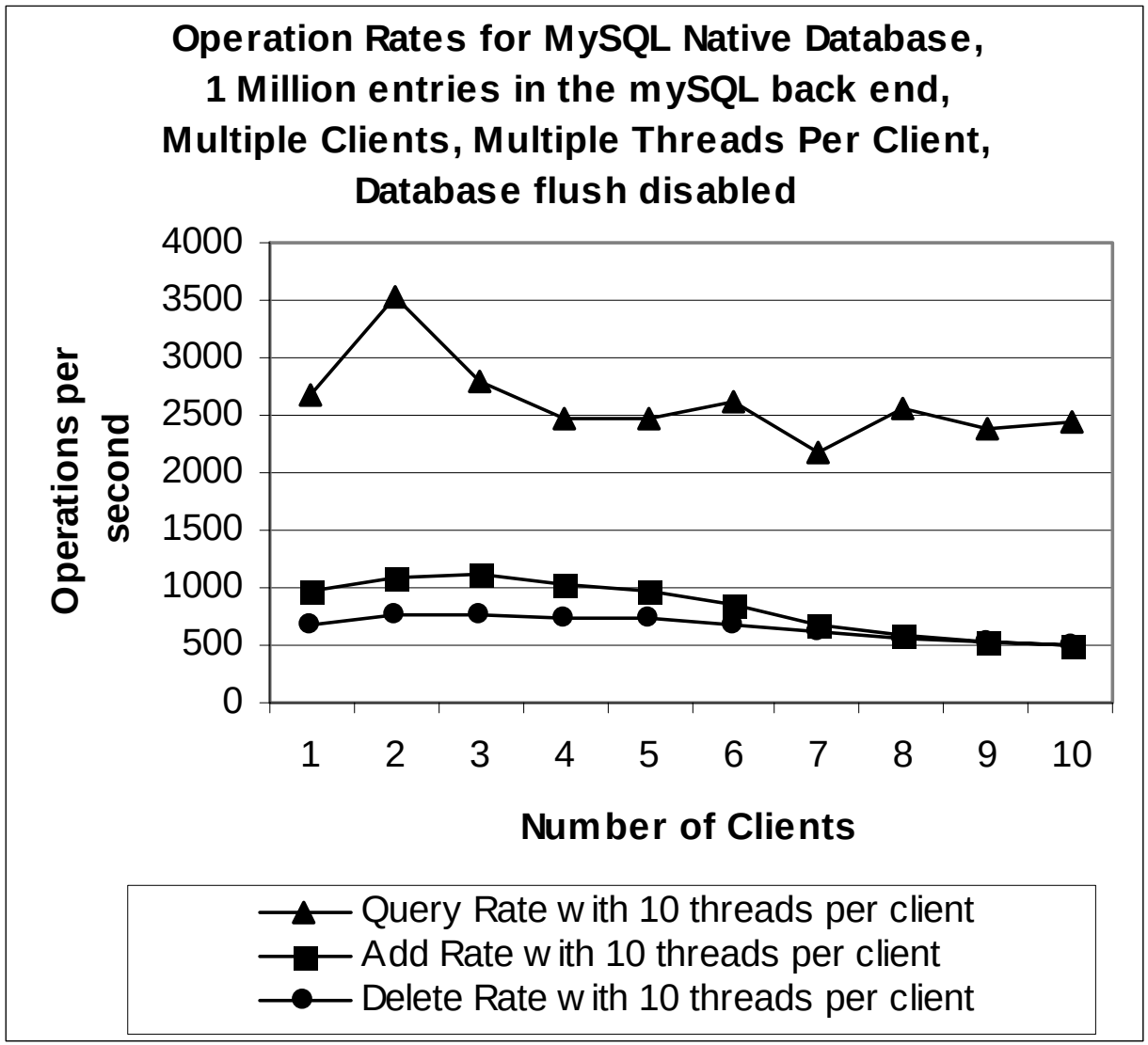
LRC Operation Rates (MySQL Backend)



- Up to 100 total requesting threads
- Clients and server on LAN
- Query: request the target of a logical name
- Add: register a new <logical name, target> mapping
- Delete a mapping

the globus alliance
www.globus.org

Comparison of LRC to Native MySQL Performance



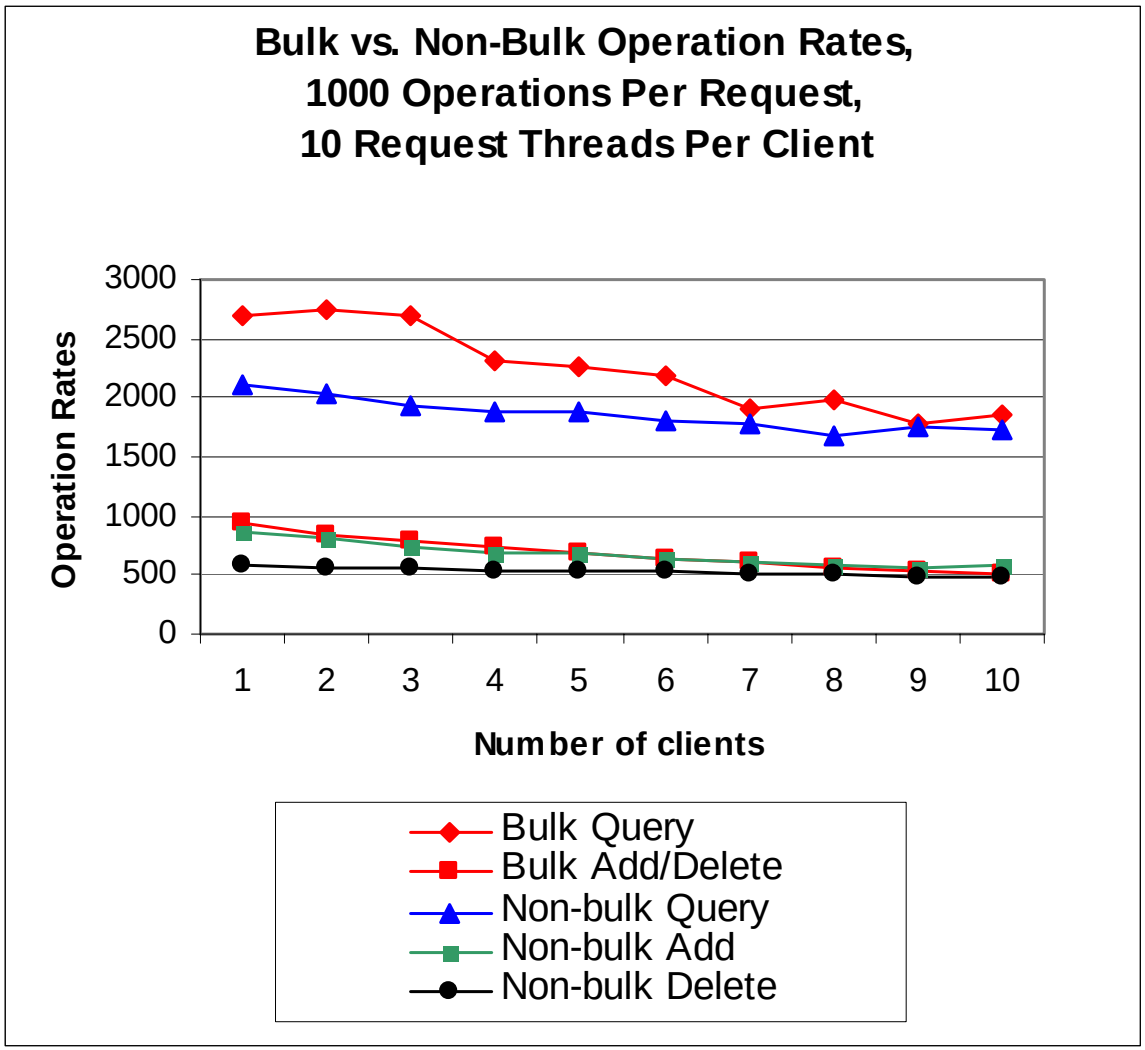
LRC Overheads

Highest for queries: LRC achieve 70-80% of native rates

Adds and deletes: ~90% of native performance for 1 client (10 threads)

Similar or better add and delete performance with 10 clients (100 threads)

Bulk Operation Performance



- For user convenience, server supports bulk operations
- E.g., 1000 operations per request
- Combine adds/deletes to maintain approx. constant DB size
- For small number of clients, bulk operations increase rates
- E.g., 1 client (10 threads) performs 27% more queries, 7% more adds/deletes

Bloom Filter Compression

- Construct a summary of each LRC's state by hashing logical names, creating a bitmap
- RLI stores in memory one bitmap per LRC

Advantages:

- Updates much smaller, faster
- Supports higher query rate
  - Satisfied from memory rather than database

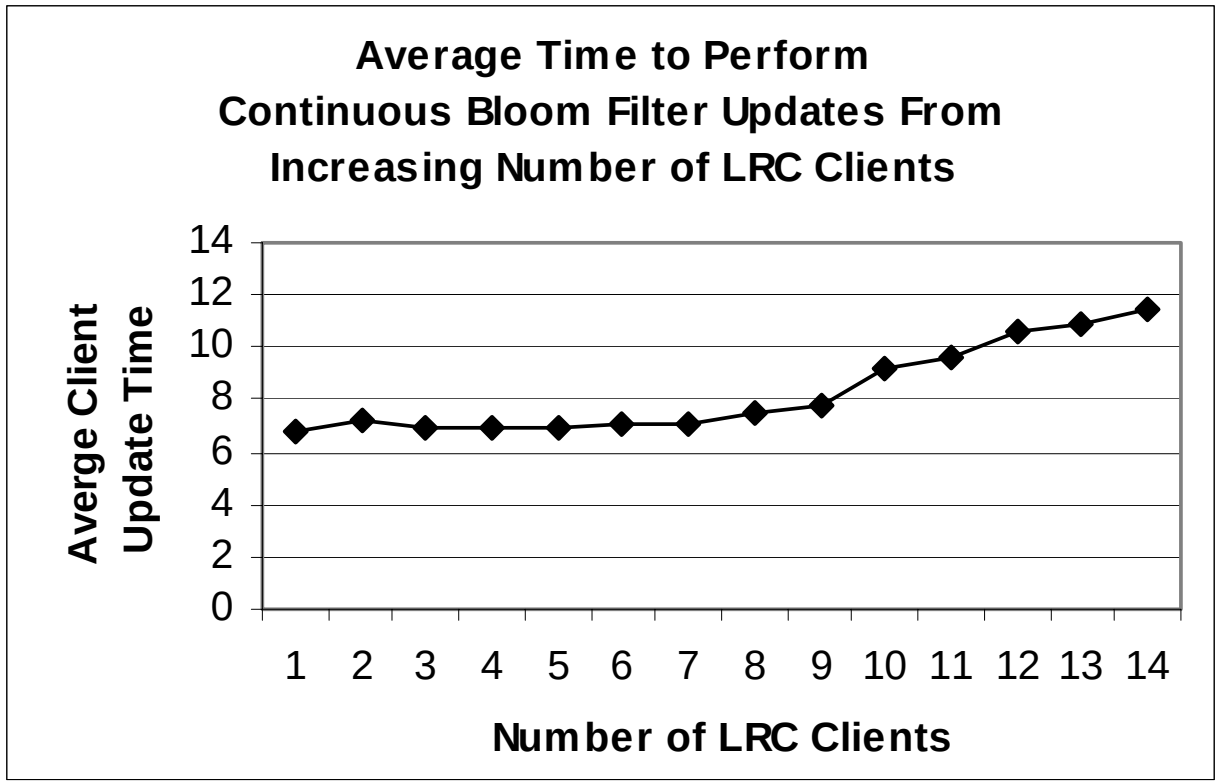
Disadvantages:

- Lose ability to do wildcard queries, since not sending logical names to RLI
- Small probability of false positives (configurable)
  - Relaxed consistency model

Bloom Filter Performance: Single Wide Area Soft State Update (Los Angeles to Chicago)

LRC Database Size	Avg. time to send soft state update (seconds)	Avg. time for initial bloom filter computation (seconds)	Size of bloom filter (bits)
100,000 entries	Less than 1	2	1 million
1 million entries	1.67	18.4	10 million
5 million entries	6.8	91.6	50 million

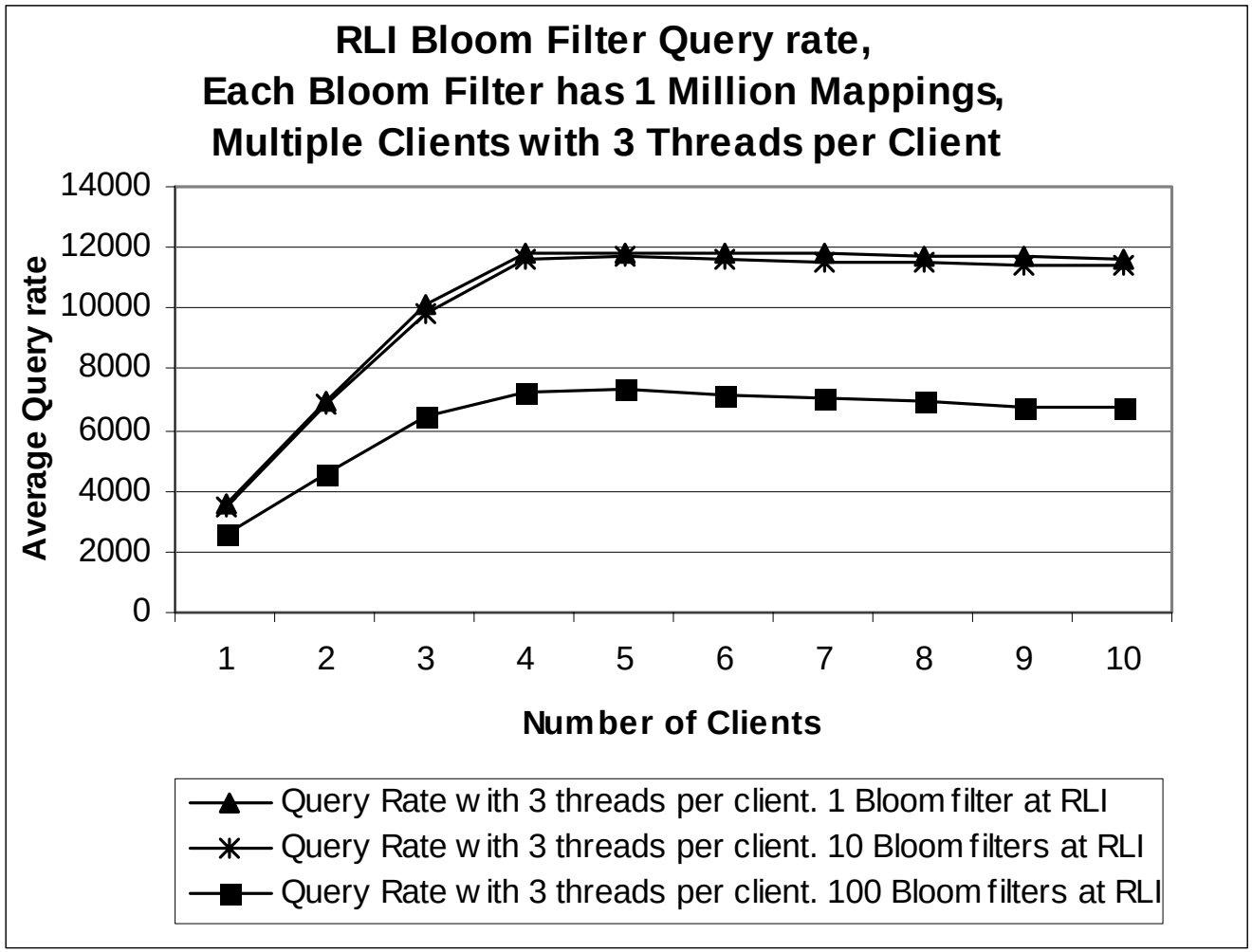
Scalability of Bloom Filter Updates



- 14 LRCs with 5 million mappings send Bloom filter updates continuously in Wide Area (unlikely, represents worst case)
- Update times increase when 8 or more clients send updates
- 2 to 3 orders of magnitude better performance than uncompressed (e.g., 5102 seconds with 6 LRCs)

the globus alliance
www.globus.org

Bloom Filter Compression Supports Higher RLI Query Rates



- Uncompressed updates: about 3000 queries per second
- Higher rates with Bloom filter compression
- Scalability limit: significant overhead to check 100 bit maps
- Practical deployments: <10 LRCs updating an RLI

WS-RF Data Publishing and Replication Service

- Being developed for the Tech Preview of GT4.0 release
- Based in part on Lightweight Data Replicator system (LDR) developed by Scott Koranda from U. Wisconsin at Milwaukee
- Ensures that a specified set of files exist on a storage site
  - Compares contents of a local file catalog with a list of desired files
  - Transfers copies of missing files other locations
  - Registers them in the local file catalog
- Uses a pull-based model
  - Localizes decision making
  - Minimizes dependency on outside services

Publishing and Replication Service (Cont.)

- WS-RF interface allows a client to explicitly specify the list of files that should exist at the local site
  - associates priorities with files should they need to be replicated from another site
  - allows clients to remove files from this list
- Each storage site uses the Replica Location Service (RLS) to determine
  - what files from the desired set are missing from the local storage system
  - where missing files exist elsewhere in the Grid
- Missing files are replicated locally
  - Issue requests to pull data to the local site from remote copies using the Reliable File Transfer Service (RFT)
- After files are transferred, they are registered in the Local Replica Catalog

RLS Plans for 2005

- Ongoing RLS scalability testing
- Incorporating RLS into production tools, such as POOL from the physics community
- Developing publishing tool that uses RLS that is loosely based on the LDR system from the LIGO project
  - Will be included in GT4.0 release as a technical preview
- Investigating peer-to-peer techniques
- OREP Working Group of the Global Grid Forum working to standardize a web services (WS-RF) interface for replica location services
  - WS-RF implementation planned for 2005

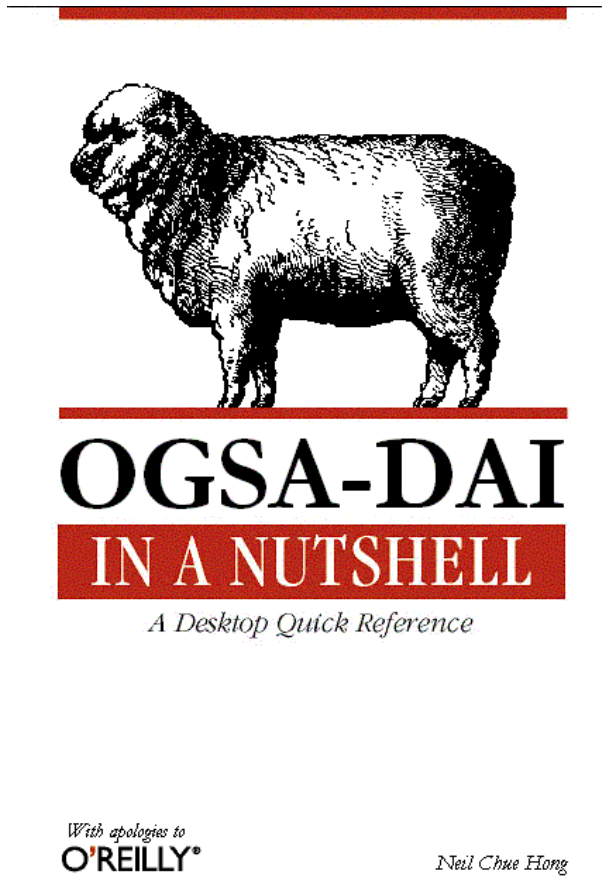
the globus alliance
www.globus.org

OGSA-DAI
Data Access and Integration for the Grid

Neil Chue Hong
EPCC, The University of Edinburgh
N.ChueHong@epcc.ed.ac.uk
http://www.ogsadai.org.uk

OGSA-DAI in a Nutshell

- All you need to know about OGSA-DAI in a handy pocket sized presentation!



OGSA-DAI Motivation

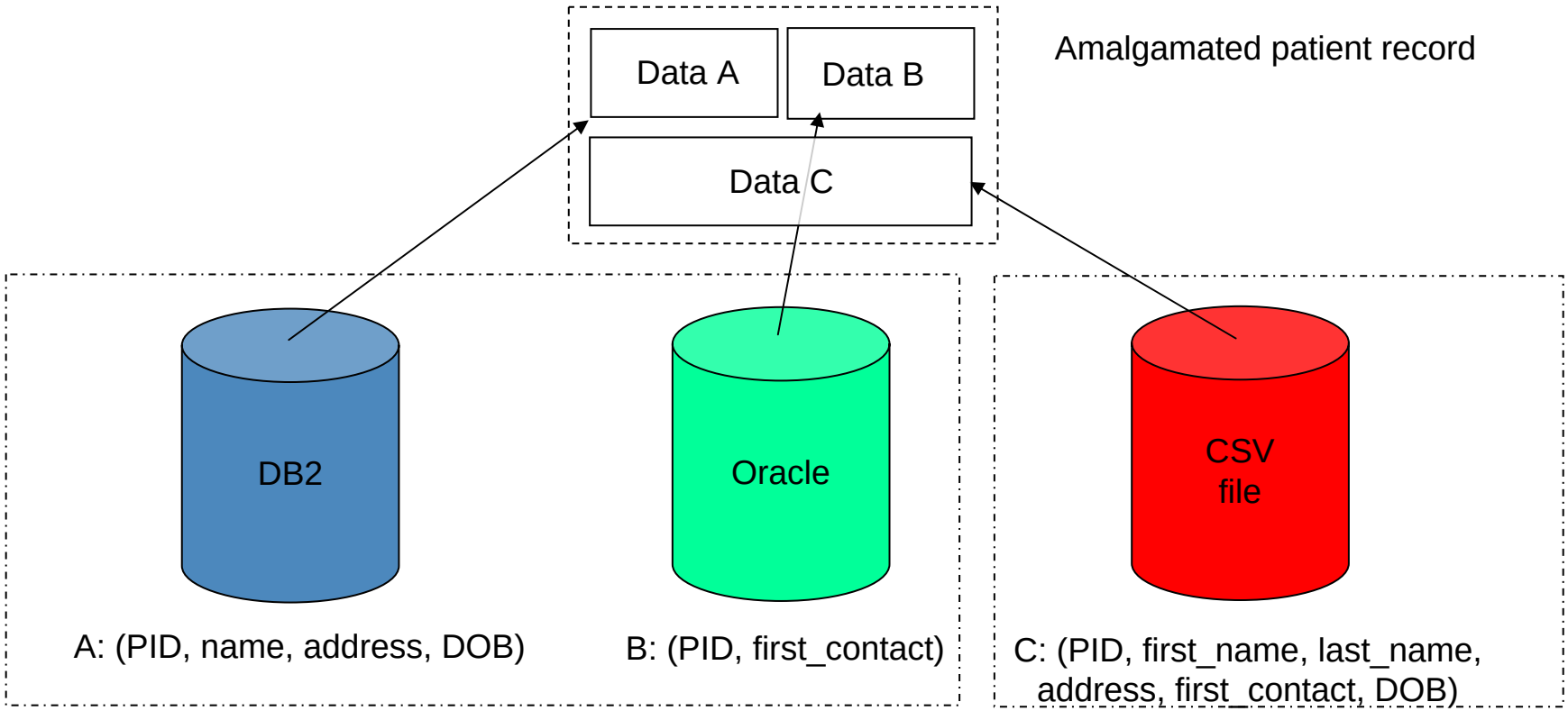
- Entering an age of data
  - Data Explosion
    - CERN: LHC will generate 1GB/s = 10PB/y
    - VLBA (NRAO) generates 1GB/s today
    - Pixar generate 100 TB/Movie
  - Storage getting cheaper
- Data stored in many different ways
  - Data resources
    - Relational databases
    - XML databases
    - Flat files
- Need ways to facilitate
  - Data discovery
  - Data access
  - Data integration
- Empower e-Business and e-Science
  - The Grid is a vehicle for achieving this

Goals for OGSA-DAI

- Aim to deliver application mechanisms that:
  - Meet the data requirements of Grid applications
    - Functionality, performance and reliability
    - Reduce development cost of data centric Grid applications
    - Provide consistent interfaces to data resources
  - Acceptable and supportable by database providers
    - Trustable, imposed demand is acceptable, etc.
    - Provide a standard framework that satisfies standard requirements
- A base for developing higher-level services
  - Data federation
  - Distributed query processing
  - Data mining
  - Data visualisation

Integration Scenario

- A patient moves hospital



Why OGSA-DAI?

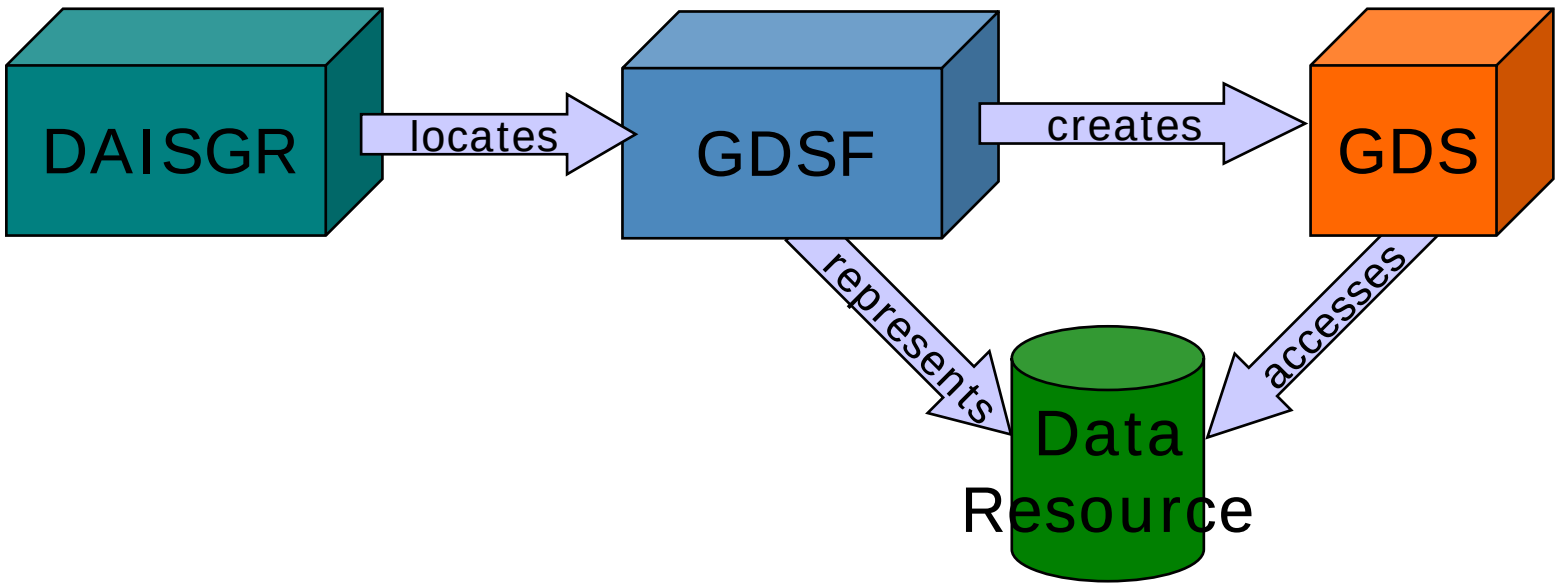
- Why use OGSA-DAI over JDBC?
  - Language independence at the client end
    - Do not need to use Java
  - Platform independence
    - Do not have to worry about connection technology and drivers
  - Can handle XML and file resources
  - Can embed additional functionality at the service end
    - Transformations, Compression, Third party delivery
    - Avoiding unnecessary data movement
  - Provision of Metadata is powerful
  - Usefulness of the Registry for service discovery
    - Dynamic service binding process
  - The quickest way to make data accessible on the Grid
    - Installation and configuration of OGSA-DAI is fast and straightforward

Core features of OGSA-DAI

- An extensible framework for building applications
  - Supports relational, xml and some files
    - MySQL, Oracle, DB2, SQL Server, Postgres, XIndice, CSV, EMBL
  - Supports various delivery options
    - SOAP, FTP, GridFTP, HTTP, files, email, inter-service
  - Supports various transforms
    - XSLT, ZIP, GZip
  - Supports message level security using X509 certificates
  - Client Toolkit library for application developers
  - Comprehensive documentation and tutorials
- Third production release on 3 December 2004
  - OGSI/GT3 based
  - Also previews of WS-I and WS-RF/GT4 releases

OGSA-DAI Services

- OGSA-DAI uses three main service types
  - DAISGR (registry) for discovery
  - GDSF (factory) to represent a data resource
  - GDS (data service) to access a data resource

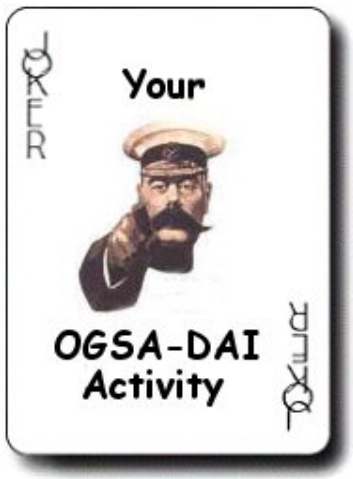


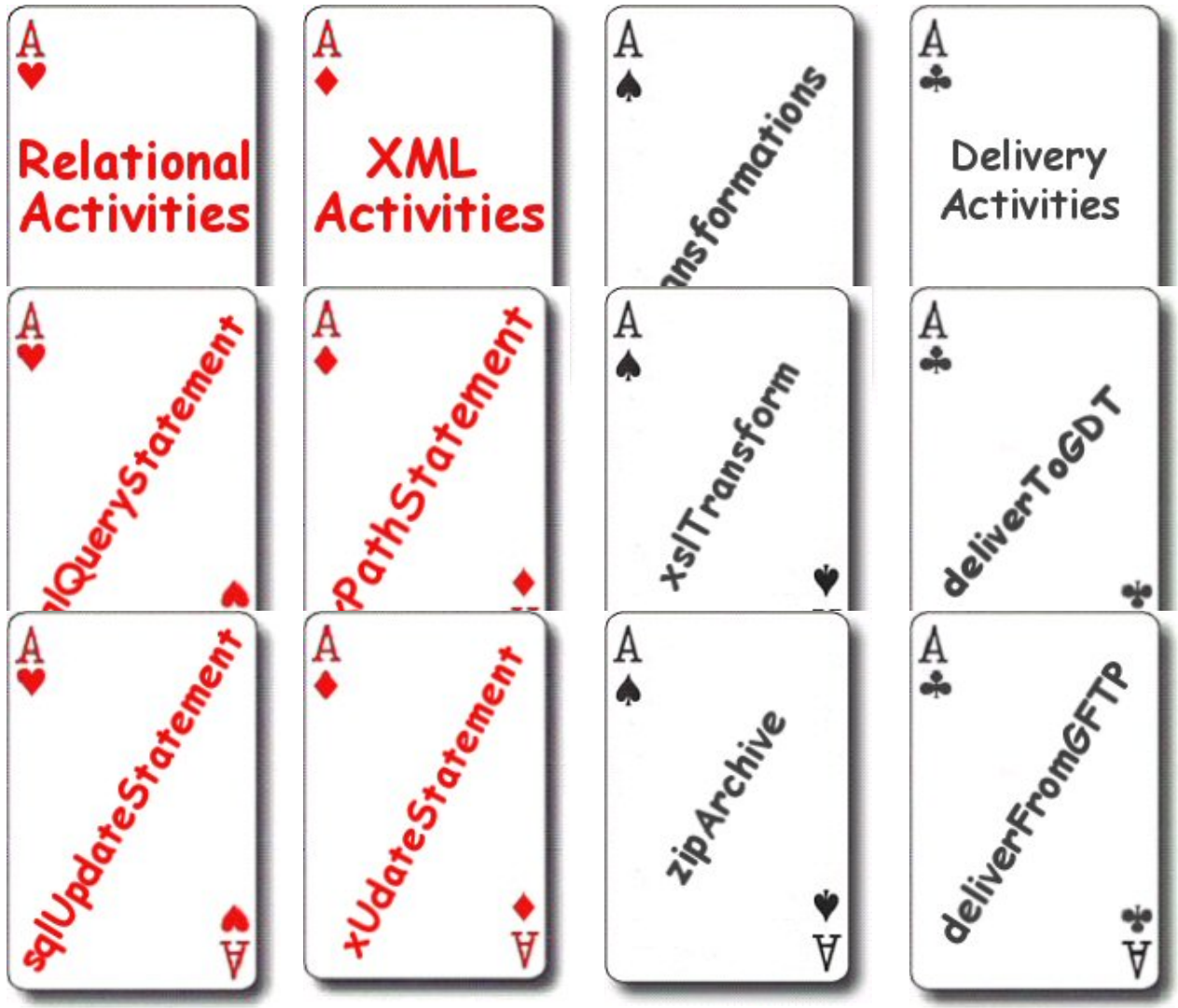
Activities are the drivers

- Express a task to be performed by a GDS
- Three broad classes of activities:
  - Statement
  - Transformations
  - Delivery
- Extensible:
  - Easy to add new functionality
  - Does not require modification to the service interface
  - Extension operate within the OGSA-DAI framework
- Functionality:
  - Implemented at the service
  - Work where the data is (do not require to move data back)

the globus alliance
www.globus.org

OGSA-DAI Deck

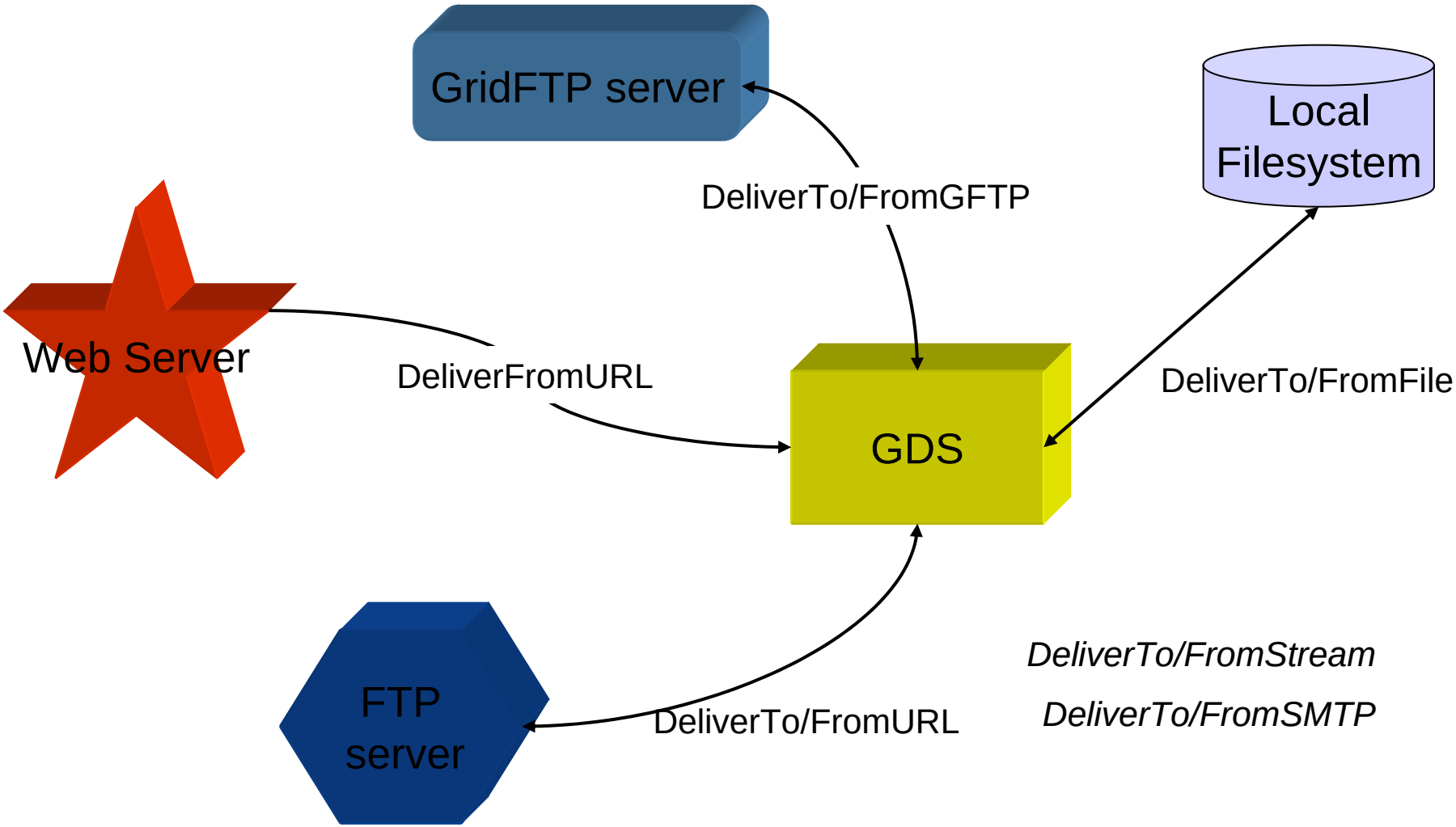




Activities and Requests

- A request contains a set of activities
- An activity dictates an action to be performed
  - Query a data resource
  - Transform data
  - Deliver results
- Data can flow between activities

Delivery Methods



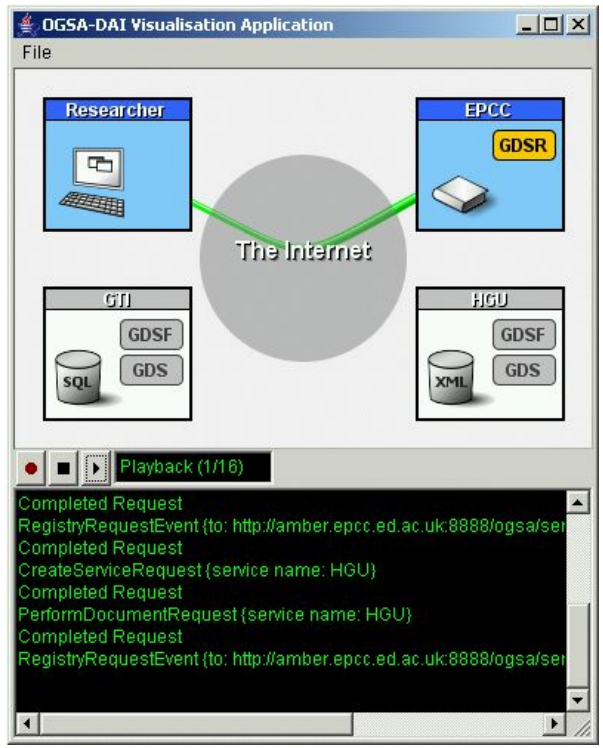
Client Toolkit

- Why? Nobody wants to write XML!
- A programming API which makes writing applications easier
  - Now: Java
  - Next: Perl, C, C#?, ML!?

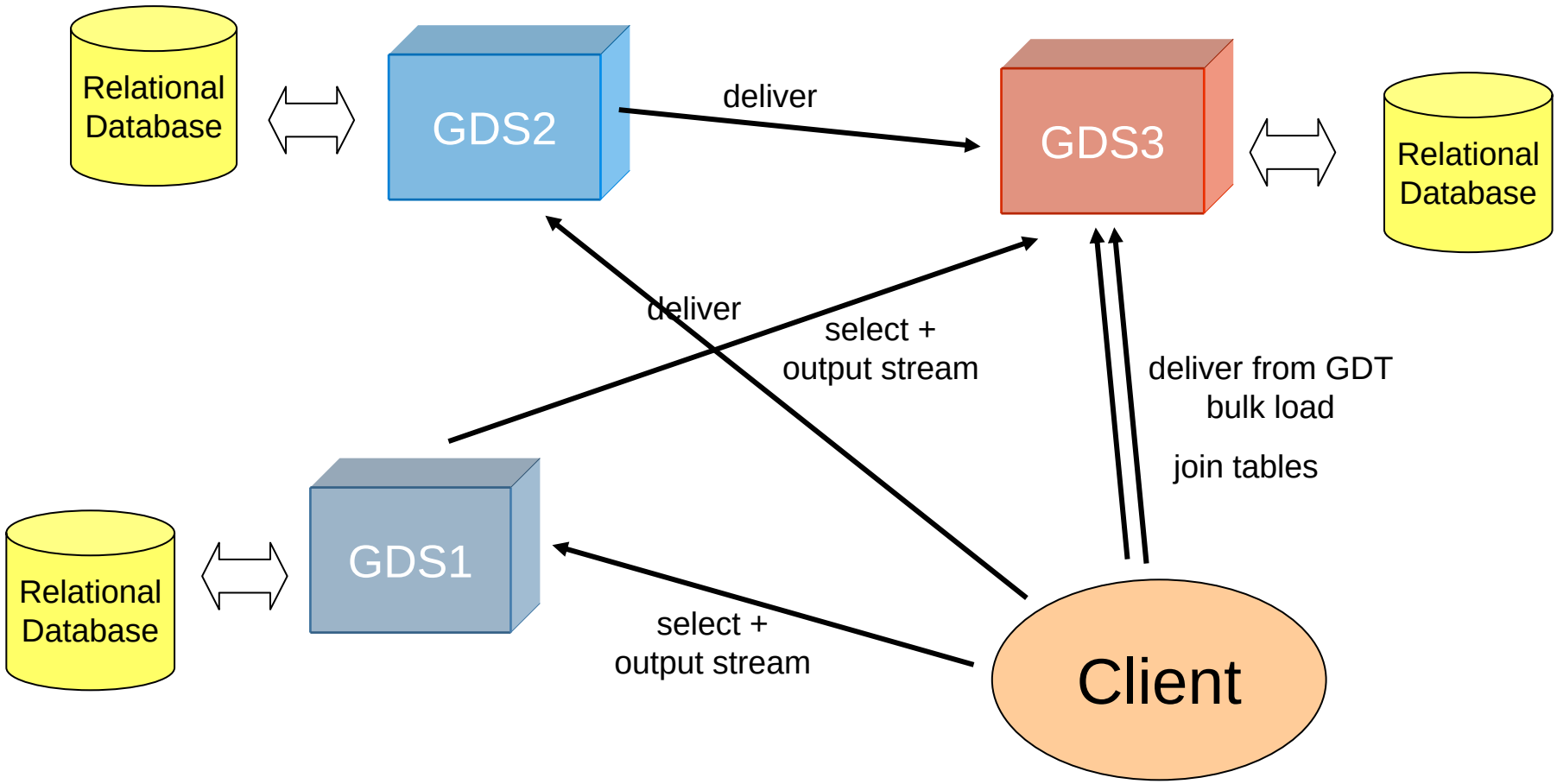
```
// Create a query
SQLQuery query = new SQLQuery(SQLQueryString);
ActivityRequest request = new ActivityRequest();
request.addActivity(query);

// Perform the query
Response response = gds.perform(request);

// Display the result
ResultSet rs = query.getResultSet();
displayResultSet(rs, 1);
```



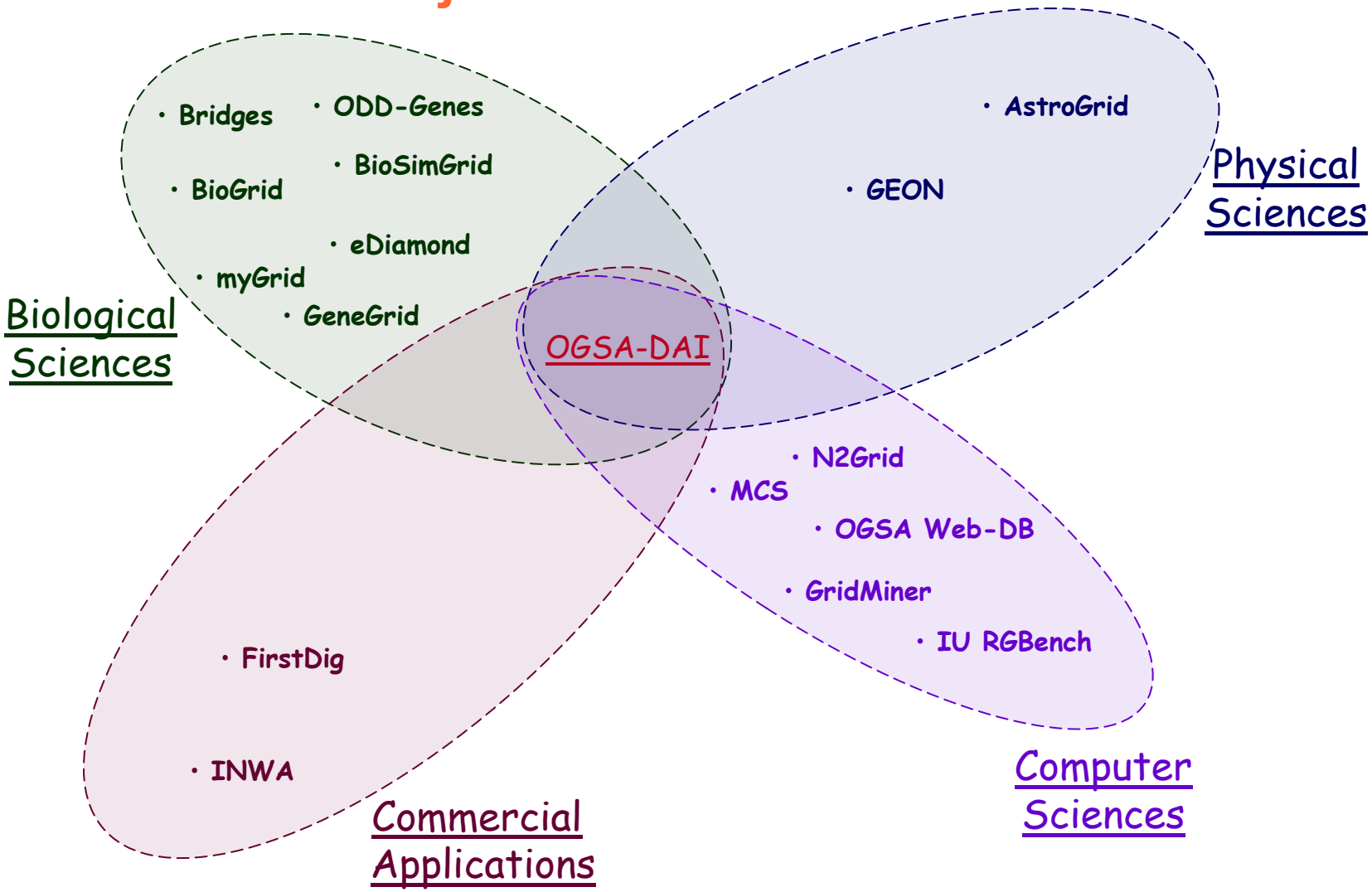
Data Integration Scenario



Release 5

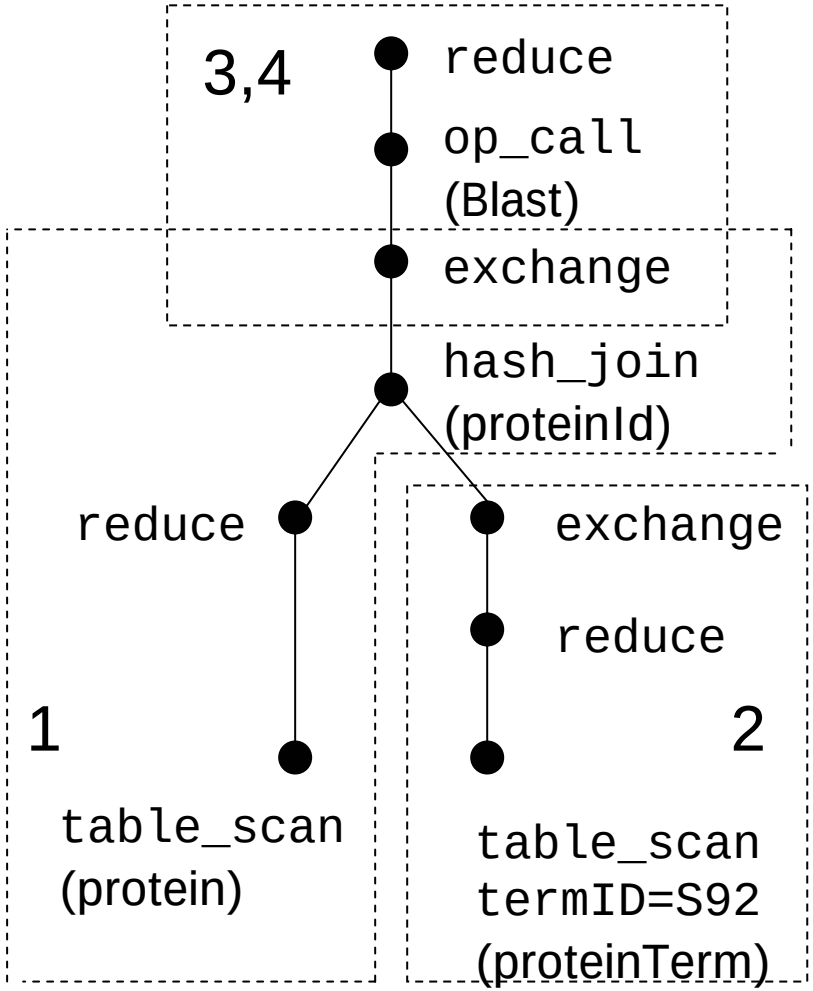
- Release 5.0 on 3 December 2004
- Builds on GT3.2.1
- Highlights include:
  - indexing, reading and full-text searching across files using the Apache Lucene text search engine library
    - e.g. SWISSPROT and OMIM
  - command line and graphical wizards to simplify installation, testing and configuration
  - per-activity configuration, defined in the activity configuration file
  - getNBlocks operation in GDT port type
  - Notification activity
  - bulk load for XML:DB databases

Project classification



Distributed Query Processing

- Higher level services building on OGSA-DAI
- Queries mapped to algebraic expressions for evaluation
- Parallelism represented by partitioning queries
  - Use exchange operators



Resources for OGSA-DAI Users

- Users Group
  - A separate independent body to engage with users and feedback to developers
  - Chair: Prof. Beth Plale of Indiana University
  - Twice-yearly meetings
- OGSA-DAI users mailing list
  - users@ogsadai.org.uk
  - See http://www.ogsadai.org.uk/support/list.php
- OGSA-DAI tutorials
  - Coming soon ... (Q1 at NeSC, elsewhere?)

Further information

- The OGSA-DAI Project Site:
  - http://www.ogsadai.org.uk
- The DAIS-WG site:
  - http://forge.gridforum.org/projects/dais-wg/
- OGSA-DAI Users Mailing list
  - users@ogsadai.org.uk
  - General discussion on grid DAI matters
- Formal support for OGSA-DAI releases
  - http://www.ogsadai.org.uk/support
  - support@ogsadai.org.uk
- OGSA-DAI training courses

OGSA DAI Plans for 2005

- Transition to new platforms and standards
  - WS-RF (GT4), WS-I+ (OMII)
  - Alignment with published DAIS specifications
- Data Integration
  - Implement simple patterns (e.g. AND, OR, PREFERRED, PARTIAL) within service code
  - Tighter integration of relational, XML and other resources
  - Better performance for inter-service data transfer
- Releases, support and community
  - Releases provisionally in April and September
  - Seek contributions in various areas of new architecture
  - Moving forward to new versions of OGSA-DAI

the globus alliance
www.globus.org

Summary of Globus Data Services and Plans for 2005

GridFTP

- A secure, robust, fast, efficient, standards based, widely accepted data transfer protocol
  - 3rd-party transfers
  - Striped/parallel data channels
  - Partial file transfers
  - Progress monitoring
  - Extended restart
- Plans for 2005
  - Performance, robustness, ease of use
  - Work on allowing variable stripe width
  - Work on improving performance on many small files
  - Access to non-standard backends (SRB, HPSS, NeST)

RFT

- Reliable File Transfer Service
  - WS-RF service
  - Accepts a SOAP description of the desired transfer
  - Writes this to a database (saves state, allows restart)
  - Uses Java GridFTP client library to initiate 3rd part transfers on behalf of the requestor
  - Supports concurrency, i.e., multiple files in transit at the same time
- Plans for 2005
  - Performance, robustness, ease of use
  - Support for priorities
  - Support for http, https, file (ala globus-url-copy)
  - Add support for GridFTP changes resulting from the above

RLS

- Replica Location Service
  - Distributed registry
  - Records the locations of data copies
  - Allows replica discovery
- Plans for 2005
  - Ongoing RLS scalability testing
  - Incorporating RLS into production tools, such as POOL from the physics community
  - Developing publishing tool for GT4.0 release as a technical preview
  - Investigating peer-to-peer techniques
  - WS-RF implementation planned for 2005

OGSA DAI

- Data Access and Integration Service
  - An extensible framework for building applications
  - Supports relational, xml and some files
  - Supports various delivery options and transforms
  - Supports message level security using X509 certificates
- Plans for 2005
  - Transition to new platforms and standards
  - Data Integration
    - Implement simple patterns within service code
    - Tighter integration of relational, XML and other resources
    - Better performance for inter-service data transfer