The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets

University of Maryland
CMSC818S

Presented by: Burcu Karagol-Ayan
burcu@cs.umd.edu

Outline

- Overview
- Design
- Core Data Grid Services
  - Data Access Service (Storage)
  - Metadata Management
- Other Basic Services
- Higher-Level Data Grid Components
  - Replica management
  - Replica selection and data filtering
- Implementation Experiences

Overview

- Motivation behind Data Grids
  - Combination of large dataset size
  - Geographic distribution of users and resources
  - Computationally intensive analysis
- Goal of this study
  - Define requirements a data grid must satisfy
  - Define components and APIs that are required in implementation

Design

- **Principles** that drive the design
  - Mechanism neutrality
  - Policy neutrality
  - Compatibility with Grid infrastructure
  - Uniformity of information infrastructure
- **Layered architecture**
  - lowest layers
    - provide high-performance access to an orthogonal set of basic mechanisms
    - not enforce specific usage policies
  - application specific behavior is limited to upper layers
    - promote reuse of basic mechanisms
    - deliver high performance and specialized capabilities to end user and application

Core Data Grid Services

- Data access
  - provides mechanisms for accessing, managing, and initiating third-party transfers of data stored in storage systems
- Metadata access
  - provides mechanisms for accessing and managing information about data stored in storage systems

Storage Systems and Data Access

- Data may be stores in different locations, different devices with different characteristics
- Because of mechanism neutrality, applications
  - Not aware of specific low-level mechanisms to access data at a particular location
  - But presented a uniform view of data and uniform mechanisms for accessing that data
- These are met by
  - Storage system abstraction
  - Prototype API for data access

Data Abstraction: Storage Systems

- Storage system
  - Provides functions for creating, destroying, reading, writing, and manipulating file instances
  - Can be implemented by any storage technology that can support required access functions
  - Need not map directly to a single low-level storage device
  - Will associate a set of properties (name, attributes like size and access restrictions) with each of the file instances that it contains
    - The name assigned to a file instance by a particular storage system has meaning only to that storage system
- File instance
  - Basic unit of information in a storage system
  - Consists of a named, uninterrupted sequence of bytes
  - May actually reside in a file sys., database or other storage sys.

Data Access

- An API
- Defines the behavior of a storage system as seen by a data grid user
- Describes the possible operations on storage systems and file instances
- Functionalities include:
  - Support for remote requests
    - to read/write named file instances
    - to determine file instance attributes
  - Third party transfer operation to transfer entire contents of a file instance from one storage system to another

Data Access - 2

- Basic storage system functions are relatively simple
- But various data grid considerations can increase the complexity of an implementation
  - Integration of storage system functions with security environment of each remote site
  - Reservation capabilities within storage systems and network interfaces
- Applications should be able to
  - Provide storage systems with hints concerning access patterns, network performance, etc, for storage systems can use these to optimize
- Storage systems should be capable of
  - Characterizing and monitoring their performance
- Data movement functions must be able to
  - Detect and report errors

Metadata

- Information about the data grid itself
  - file instances
  - contents of file instances
  - various storage systems
- Different types
  - Application metadata: information content represented by file, the circumstances under which data was obtained, and/or other information. Can be viewed as defining the logical structure or semantics that should apply to the uninterrupted bytes that make up a file instance or a set of file instances
  - Replica metadata: used to manage replication of data objects; includes information for mapping file instances to particular storage system locations
  - System configuration metadata: describes the fabric of the data grid itself (network connectivity, details about storage systems, like capacity and usage policy)

Metadata Service

- Concerned with the management of metadata
- Provides a uniform means for naming, publishing, and accessing different types of metadata
- Each type has its own characteristics in
  - frequency
  - mechanism of update
  - its logical relationship to other grid components and data items
- A single interface for accessing all types

Metadata Service - 2

- Applications identify files of interest
  - by posing queries that specifies the characteristics of the desired data
  - to a metadata service that includes a metadata repository or catalog
- Metadata repository associates such characteristics with logical files
  - entities with globally unique names that may have one or more physical instances
- When metadata service has identified logical files with the desired attributes, replica manager uses replica metadata to locate the physical file instance to be accessed

Metadata Service - 3

- Specifying a general structure is difficult because
  - variety of approaches used to describe application metadata
  - additional requirements imposed by large-scale data grid environments
  - Metadata service must
    - provide a means of integrating different approaches to metadata storage and representation
    - operate efficiently in a distributed environment
    - be scalable
    - be robust in the face of failure
    - let organizations assert local control over their information

Metadata Service - 4

- Therefore:
  - Metadata service must be structured as a hierarchical and distributed system
  - allows to
    - achieve scalability
    - avoid any single point of failure
    - facilitate local control over data
  - However distribution complicates efficient retrieval
  - But can be solved if data organization exploit the hierarchical nature of the metadata service

Other Basic Services

- Authorization and authentication infrastructure: supports multi-institutional operation. The public key-based Grid Security Infrastructure meets our requirements.
- Resource reservation and co-allocation mechanisms: for storage systems and other resources such as networks, to support the end-to-end performance guarantees required for predictable transfers
- Performance measurements and estimation techniques: for key resources involved in data grid operation, including storage systems, networks, and computers
- Instrumentation services: enable the end-to-end instrumentation of storage transfers and other operations

Higher-Level Data Grid Components

- Basic services can be used to develop higher-level services
- Two representative components
  - Replica management
  - Replica selection and data filtering

Replica Manager

- Role: To create/delete copies of file instances, or replicas, within specified storage systems
- A replica
  - A user-asserted correspondence between two physical files
  - Created because new storage location offers better performance or availability
  - Might be deleted because of storage need
  - Assumed to be read-only (in the paper)
- Maintains a repository or catalog
  - Entries in catalog correspond to logical files and possibly collections of logical files
  - Each logical file or collection is associated with one or more replicas
  - Replica catalog contains mapping information from a logical file or collection to one or more physical instances of the object

Replica Manager - 2

- A data grid may contain multiple replica catalogs
- It is possible to create hierarchies of replica catalogs to impose a directory-like structure on related logical collections
- Existence of a replica manager does not determine
  - when or where replicas are created
  - which replicas are to be used by an application
  - require that every file instance be entered into a replica catalog
- By keeping policy out of definition
  - Types of situations that replica manager will be useful is maximized

Replica Selection (and Data Filtering)

- Process of choosing a replica that will provide an application with data access characteristics that optimize a desired performance criterion (absolute performance, cost, or security)
- The selected file instance may be local or accessed remotely, or
- A new replica whose performance will be superior to the existing ones can be created by selection process
- If replicas are to be selected based on access time
  - Grid information services can provide information about network performance, and perhaps the ability to reserve network bandwidth
  - Metadata repository can provide information about the size of the file
- Using this information selector can determine:
  - Which replica will yield fastest data access time?
  - *Or* is there a storage system that would result in better performance if a replica was created on it?

Replica Selection and Data Filtering - 2

- When an application requires only subset of data
  - selection function may provide a file instance that contains only the needed subset of the data found in the original file instance.
  - can reduce the amount of data that must be accessed or moved
- Requires ability to invoke filtering
- The subset becomes a file instance with its own metadata and physical characteristics (provided to replica manager)
- Replication policies determine
  - Recognized as a new logical file *or*
  - Should be known only locally

Implementation Experiences

- initial design of catalogs for metadata and replica management are used to support two applications
- Lightweight Distributed Access Protocol (LDAP) is used to construct prototype catalogs in
  - Climate Modeling Application
  - Data Visualization Application
- Several changes based on these
- Separation of Functionality: *Lesson:* importance of distinguishing the functions of metadata and replica management