Building Fault-Tolerant Consistency Protocols for an Adaptive Grid Data-Sharing Service

Gabriel Antoniu, Jean-François Deverge, Sébastien Monnet

PARIS Research Group
IRISA/INRIA
Rennes, France

Workshop on Adaptive Grid Middleware - AGridM 2004
September 2004, Juan-Les-Pins
Context: Grid Computing

- Target architecture: cluster federations (e.g. GRID 5000)
- Target applications: distributed numerical simulations (e.g. code coupling)
- Problem: the right approach for data sharing?

Solid mechanics

Optics

Dynamics

Thermodynamics

Satellite design
Current Approaches: Explicit Data Management

- Explicit data localization and transfer
  - GridFTP [ANL], MPICH-G2 [ANL]
    - Security, parallel transfer
  - Internet Backplane Protocol [UTK]

- Limitations
  - Application complexity at large-scale
  - No consistency guarantees for replicated data
Handling Consistency: Distributed Shared Memory Systems

- Features:
  - Uniform access to data via a global identifier
  - Transparent data localization and transfer
  - Consistency models and protocols

- But:
  - Small-scale, static architectures

- Challenge on a grid architecture:
  - Integrate new hypotheses!
    - Scalability
    - Dynamic nature
    - Fault tolerance

Node 0

Migration?
Replication?

Node 1
Case Study:
Building a Fault-Tolerant Consistency Protocol

- Starting point: a home-based protocol for entry consistency
  - Relaxed consistency model
    - Explicit association of data to locks
    - MRSW: Multiple Reader Single Writer
      - acquire(L)
      - acquireRead(L)
  - Implemented by a home-based protocol

![Diagram of Home node and Client nodes]
Home Based Protocol

Cluster A

Client A

acquire

lock

read x

data x

w(x)

Home

acquire

lock

read x

data x

w(x)

release

Cluster B

Client B

lock

read x

data x

w(x)

release
Inspired by CLRC[LIP6, Paris] and H2BRC[IRISA, Rennes]
Problem: Critical Entities May Crash

How to support home crashes on a grid infrastructure?
Idea: Use Fault-Tolerant Components

- Replicate critical entities on a group of nodes
- Group of nodes managed using the *group membership* abstraction
- Rely on *atomic multicast*
- Example architecture: A. Schiper[EPFL]
Approach: Decoupled Design

- Consistency protocol layer and fault-tolerance layer are separated

- Interaction defined by a junction layer
Consistency/Fault-Tolerance Interaction

- Critical consistency protocol entities implemented as fault-tolerant node groups
- Group management using traditional group membership and group communication protocols
- Junction layer handles
  - Group self-organization
  - Configuration of new group members
Replicate Critical Entities Using Fault-Tolerant Components

- Rely on replication techniques and group communication protocols used in fault-tolerant distributed systems
Replicate Critical Entities Using Fault-Tolerant Components

- Rely on replication techniques and group communication protocols used in fault-tolerant distributed systems
The JuxMem Framework

- DSM systems: consistency and transparent access
- P2P systems: scalability and high dynamicity
- Based on JXTA, P2P framework [Sun Microsystems]
Implementation in JuxMem

- Data group ≈ GDG + LDG

Juxmem group

Cluster group A

Cluster group B

Cluster group C

Virtual architecture

Physical architecture
Preliminary Evaluation

- Experiments
  - Allocation cost depending on replication degree
  - Cost of the basic data access operations
    - read/update
- Testbed: *paraci* cluster (IRISA)
  - Bi Pentium IV 2.4 Ghz, 1 Go de RAM, Ethernet 100
  - Emulation of 6 clusters of 8 nodes
Allocation Process

1. Discover \( n \) providers according to the specified replication degree

2. Send an allocation request to the \( n \) discovered providers

3. On each provider receiving an allocation request:
   - Instantiate the protocol layer and the fault-tolerant building blocs
Preliminary Evaluation: Allocation Cost

![Graph showing latency (msecs) vs. GDG and LDG group sizes (GDGxLDG)]
Cost of Basic Primitives: read/update

![Graph showing the cost of basic primitives for different LDG sizes. The graph compares read and update operations for 16K and 4M data sizes.]
Conclusion

- Handling consistency of mutable, replicated data in a volatile environment
- Experimental platform for studying the interaction fault-tolerance <-> consistency protocols
Future Work (AGRIDM 2003)

- Consistency protocols in a dynamic environment
- Replication strategies for fault tolerance
- Co-scheduling computation and data distribution
- Integrate high-speed networks: Myrinet, SCI.
Future Work (AGridM 2003)

- Consistency protocols in a dynamic environment
- Replication strategies for fault tolerance
- Co-scheduling computation and data distribution
- Integrate high-speed networks: Myrinet, SCI.
Future Work (AGridM 2004)

- **Goal:** build a Grid Data Service
  - Experiment various implementations of fault-tolerant building blocks (atomic multicast, failure detectors, ...)
  - Parametrizable replication techniques
  - Experiment various consistency protocols with various replication techniques
  - Experiment with realistic grid applications at large scales

- **GDS (Grid Data Service) project of ACI MD:**
  
  http://www.irisa.fr/GDS