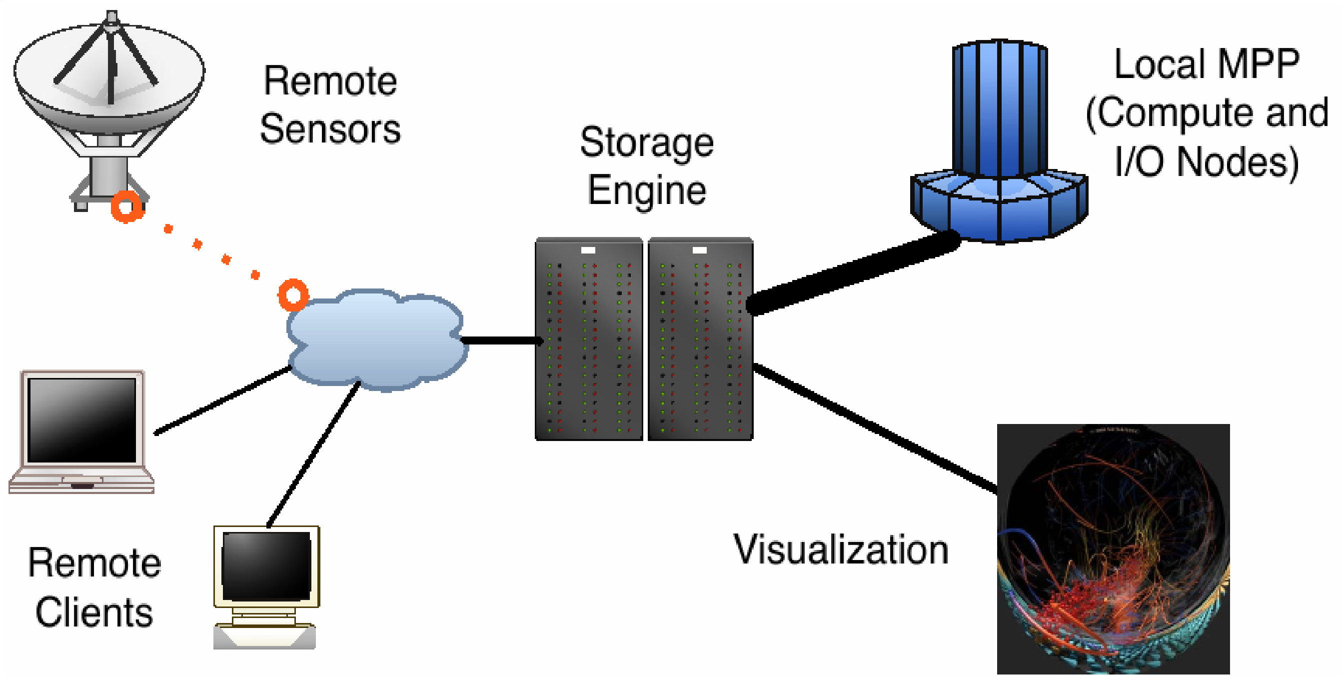
SDSS: Peta-scale Storage for Very High End Computing

Collaborative Research

College of Computing
Georgia Institute of Technology

Computer Science Department
University of New Mexico

Karsten Schwan
Arthur Maccabe
Patrick Bridges
Greg Eisenhauer
Patrick Widener
Matt Wolf
and
Ron A. Oldfield, Sandia National Laboratories



Research Challenges:

- \# clients vs. limited # nodes in I/O partition
- balance b/w utilization across system for scalability

=> address mismatch between current point-to-point, file-based I/O and the actual needs of data-intensive HPC applications

Solution Approach: LWFS and I/O Graphs

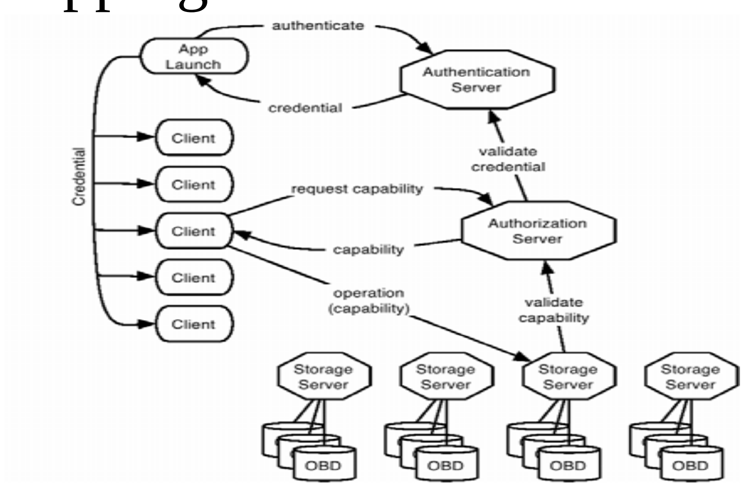
Rich application(-specific), customizable I/O structure:

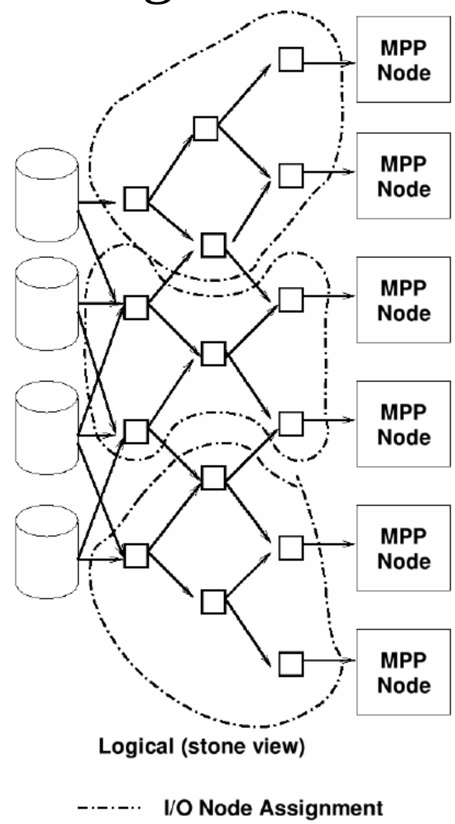
LWFS: Light Weight File System:

- asynchronous or off-line processing of file/storage actions

I/O Graphs:

- explicitly represent I/O tasks and their mapping to MPP and I/O nodes





Key Elements of Solution Approach

Addressing scalability challenges:

- prohibit system-imposed functions requiring O(n) functions (n = #client nodes)
- prohibit data structures of size O(n) (e.g., no connection-based mechs.)

New tools to `upgrade’ from lightweight nature of I/O system:

- deal with differing storage organizations
- enhance metadata associated with I/O data and actions
- ensure performance and QoS requirements of I/O

Proposed SDSS System

- I/O described as ‘structured events’ manipulated by dynamic flow graphs (i.e., I/O Graphs)

- I/O Graphs ‘connect’ simulation components with I/O subsystem; created, configured, removed dynamically; mapped automatically

- I/O Graph nodes implement application-specific I/O services (e.g., data staging); managed automatically

- I/O Graphs are integrated with LWFS, in fact, they perform some of the actions required by LWFS (e.g., consistency management done by transaction managers)

- I/O Graphs can operate asynchronously and concurrently (e.g., use of MetaBots)

Implementation

- SDSS extends, not replaces, existing back-end storage/file systems (e.g., LWFS) and HPC transport protocols (e.g., portals)
- SDSS uses efficient binary representation of metadata (e.g., PBIO)
- SDSS interoperates with other systems (e.g., Lustre)
- SDSS will run on realistic machines (e.g., ORNL/Sandia machines) with realistic applications (e.g., fusion modeling, ORNL)

SDSS Team Capabilities

- UNM: Maccabe, Bridges, Widener
- Sandia: Oldfield (LWFS)

- Georgia Tech: Schwan, Eisenhauer, Wolf
- (ORNL: Klasky (Fusion Modeling))