CLARISSE: Cross-Layer Abstractions and Run-time for I/O Software Stack of Extreme-Scale Systems

Florin Isaila

ANL & University Carlos III

Collaborators: Prasanna Balaprakash (ANL), Phil Carns (ANL), Jesus Carretero (UC3M), Javier Garcia (UC3M), Kevin Harms (ANL), Dries Kimpe (ANL), Rob Latham(ANL), Rob Ross (ANL), Stefan Wild (ANL)

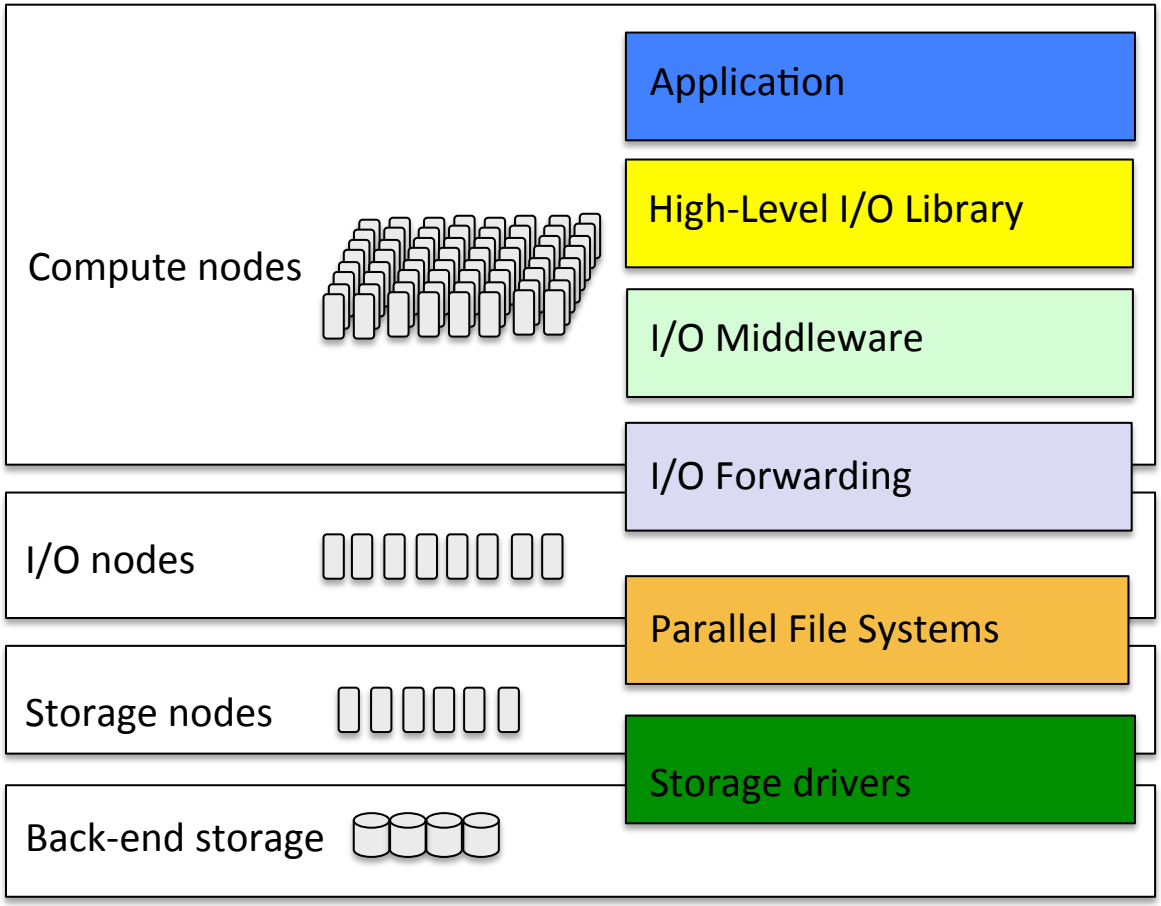
Argonne NATIONAL LABORATORY

ARCOS

Outline

- Motivation
- Goals
- CLARISSE Approach
- CLARISSE Architecture
- Challenges
- Conclusions

Current problems of storage I/O stack



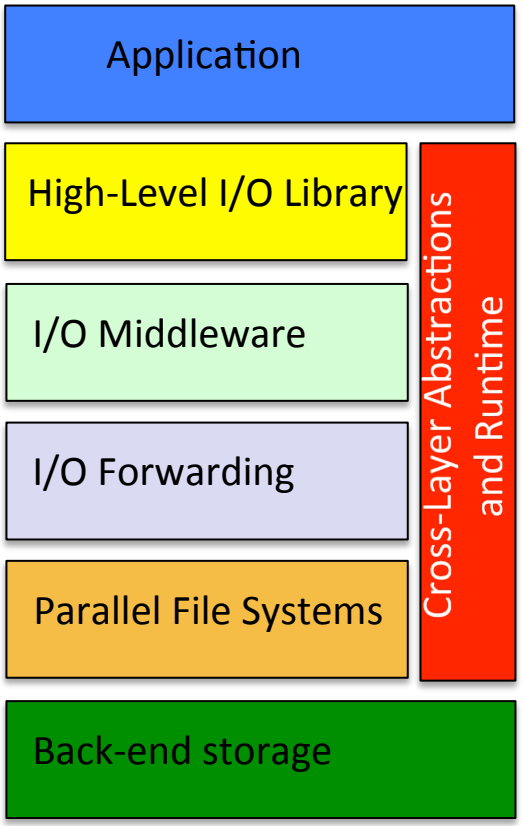
- Long path from compute nodes to final storage impacts performance (latency, throughput)
- Storage I/O optimizations are local: Difficult to perform global optimizations
- Cross-layer adaptive control mechanisms are not available (e.g for data staging, dynamic load balancing, resilience)

CLARISSE goals

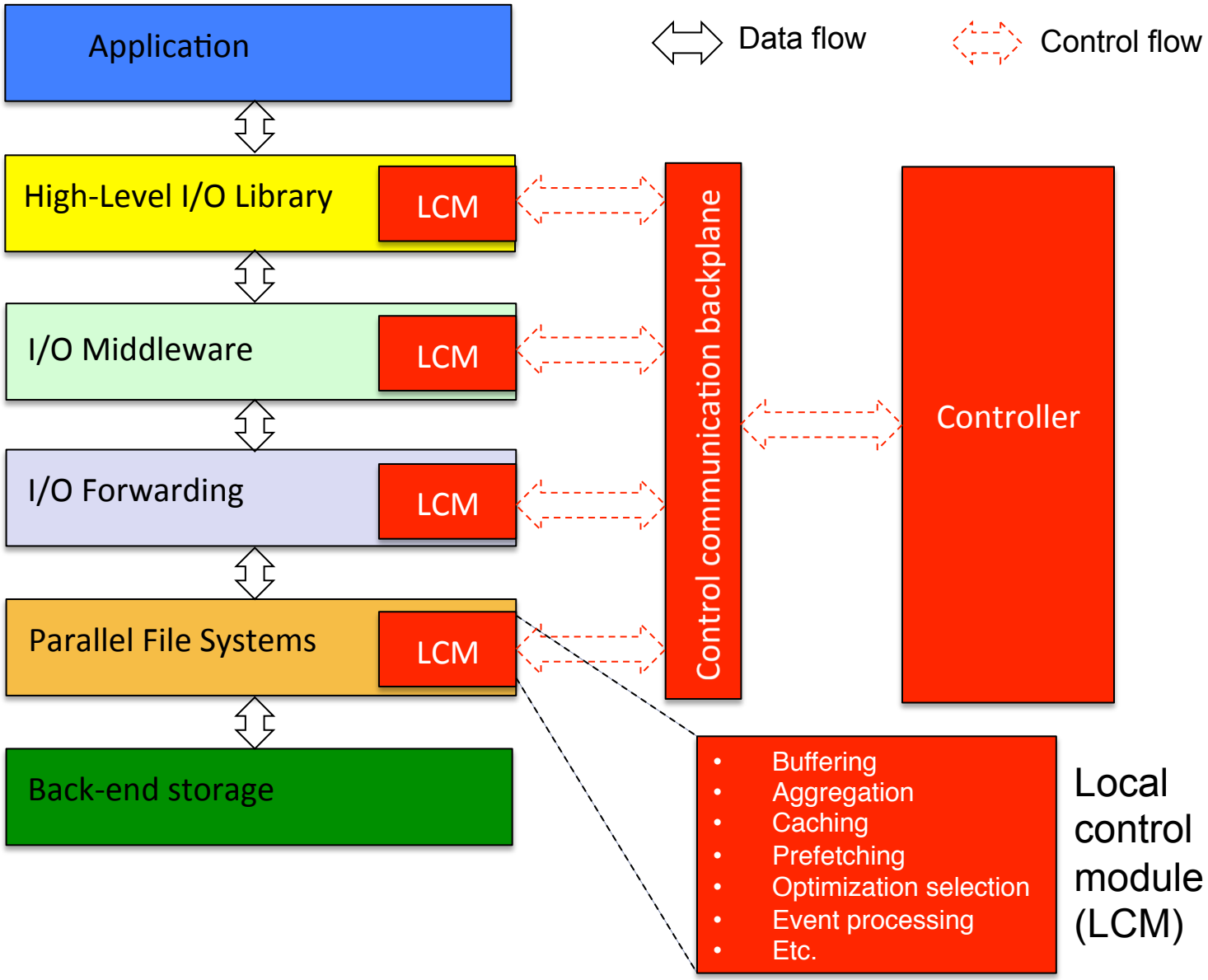
- Enable global optimizations of the software I/O stack
- Separation of data flow and control flow through the storage hierarchy
- Develop a run-time for cross-layer control of the data flow
  - Data staging
  - Data locality

CLARISSE approach

- Global logical view of optimizations space
  - Distributed application of global optimization
  - Facilitate the combination of local optimizations based on a global view
- Cross-layer abstractions and run-time
  - Facilitate the flow of control and data across the I/O stack
  - Decouple the data and control planes
    - Control backplane
    - Abstractions for controlling the data plane
- Control adaptation to application context
  - Applications run unmodified
  - Flexibly define control policies adapted to the requirements of each application
- Data flow optimizations
  - Coordinated cross-layer buffering, caching, prefetching
  - Run-time optimizations (e.g. load-aware data staging)

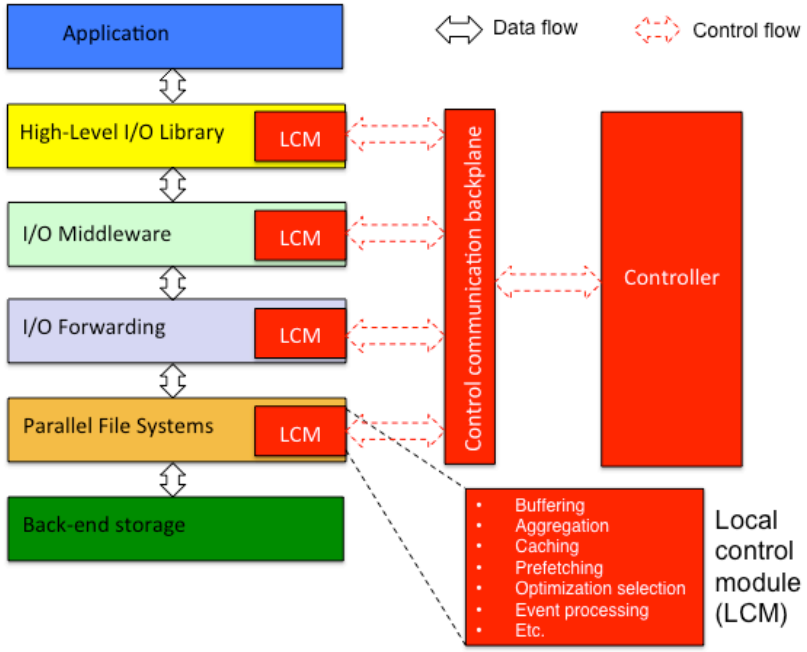


CLARISSE Architecture



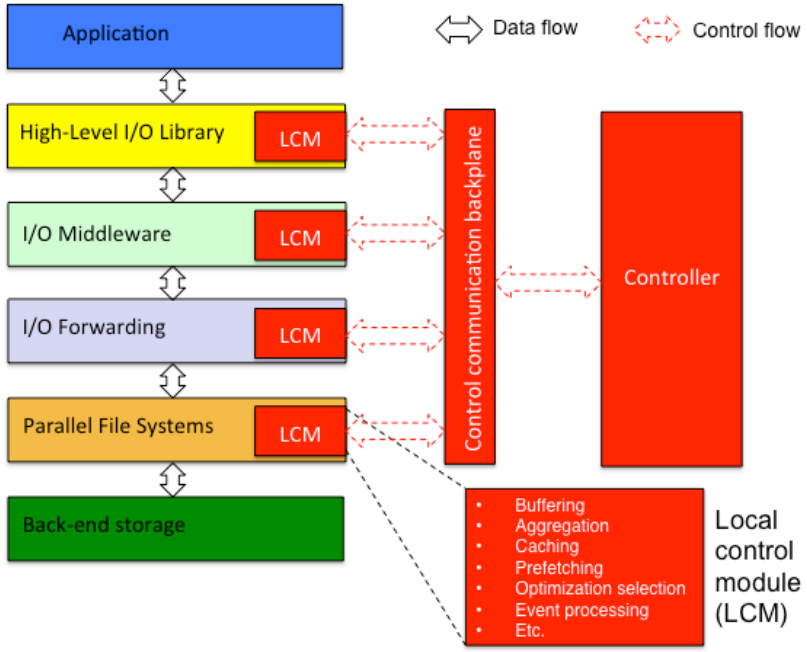
Controller

- Cross-layer logic
- Gathers information
  - Application patterns
  - Data related attributes
  - System characteristics
  - Available optimizations
  - Run-time events and statistics
- Global optimization inference
- Distributes the control decisions to LCMs
- API for defining control policies
  - e.g. prediction, load-aware, data locality-aware



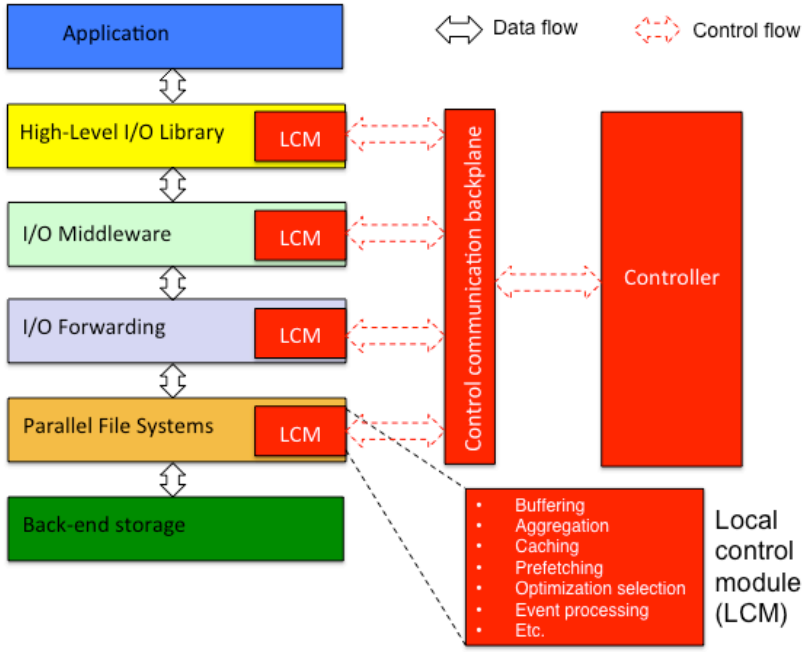
Local control module (LCM)

- Enforce control at I/O software stack layers
- Select local optimizations
- Event processing
- Collects run-time information and forwards them to controller
- Optional
  - Not all layers have to provide it
  - Each layer can use a default policy



Communication backplane

- Coordination between controller and I/O software stack layers
- Generic
- Event-based: react to user-defined criteria
- Low overhead: low interference with the system workings
- Potential candidates
  - CIFTS: A Coordinated Infrastructure for Fault-Tolerant Systems
  - Beacon: The communication backplane of ARGO, an exascale operating system



Challenges

- Scalability
  - Minimize the controlling infrastructure interference
- Minimal modifications
  - Unmodified applications
  - Small number of changes to the current I/O stack
- Distributed protocols for control and data flow
- Resilience
  - Behavior under failure of controlling infrastructure
- Extensibility
  - Simplicity of adding new control policies

Related work

- Software Defined Networking (e.g. Open Flow): global control based on separation of control and data flow
- I/O Flow (Microsoft Research): A Software Defined Storage Architecture for virtualized data centers
- Fast Forward (Intel et al.): redesign of the storage I/O stack
- Argo (ANL et al.): whole-system view and optimization based on a OS/R environment
- Hobbes (Sandia et al.): a lightweight OS/R environment with flexibility to build custom runtimes
- SIOX (University of Hamburg et al.): collect system-wide information to learn storage I/O optimizations

Conclusions

- Global optimization of storage I/O stack
- Decoupling control and data flow
- Optimize storage access
- Benefits
  - Mechanisms for global view of the storage I/O activity on extreme scale machines
  - Evolution of the I/O software stack based on whole system view
  - Run-time for controlling data flow and data locality

Thank you