# The Next Stage in High Performance Networking

STAC Summit - Chicago
November 4, 2013

Paul Grun
Cray, Inc.
Vice Chair - OpenFabrics Alliance

The summit topic(s) are

- “Big Workloads” (big compute/big data), and
- “Fast Workloads” (low latency, high throughput)

High performance networking is fundamental to both

So let’s talk about I/O

Or more properly, “Data Movement”

## Agenda for the next 14 minutes

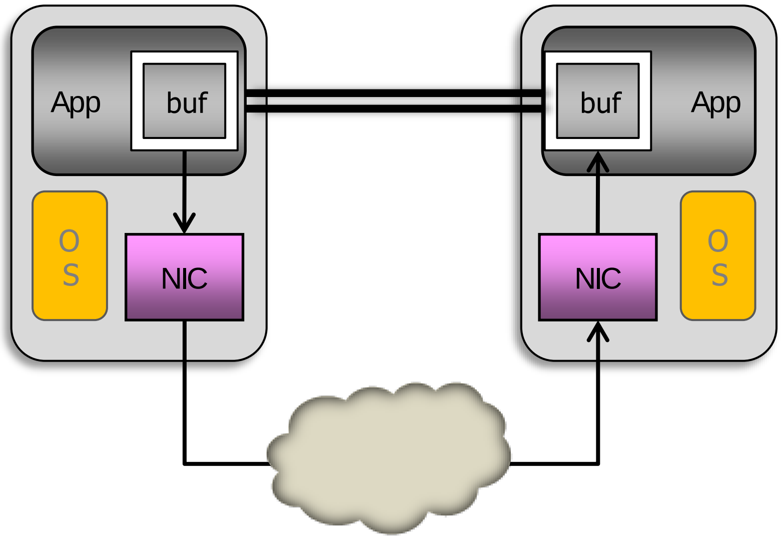
1. Here's what the OpenFramework working group is…
   …and why you probably care

2. How you can influence the outcome

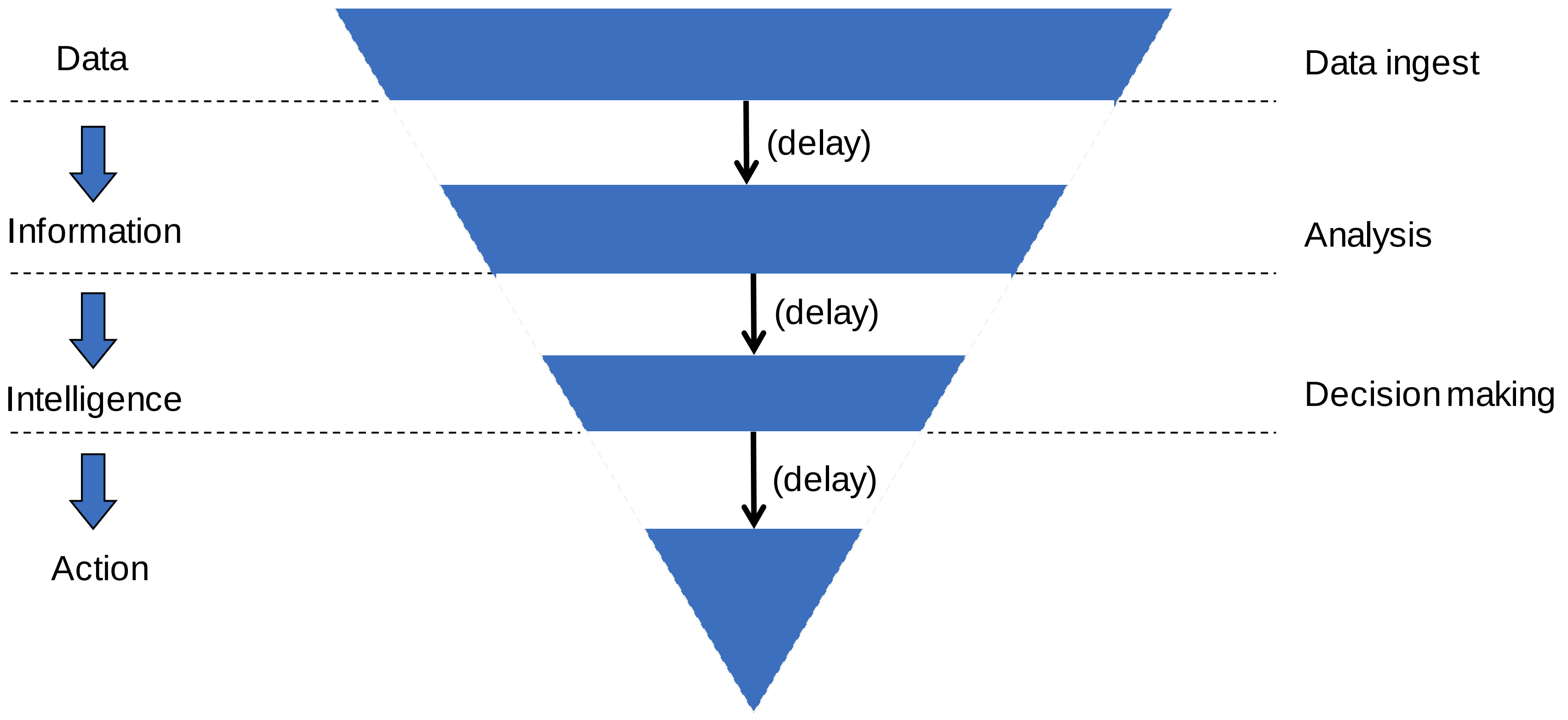
But the main agenda item is to enlist your involvement

# An brief introduction to the OpenFabrics Alliance

The OpenFabrics Alliance (OFA) develops, tests, licenses, supports and distributes OpenFabrics Enterprise Distribution (OFED™) open source software to deliver high-efficiency computing, wire-speed messaging, ultra-low microsecond latencies and fast I/O.



# Classical data transform model



A data movement problem

Each layer comprises compute (servers, clusters or mainframes) and storage

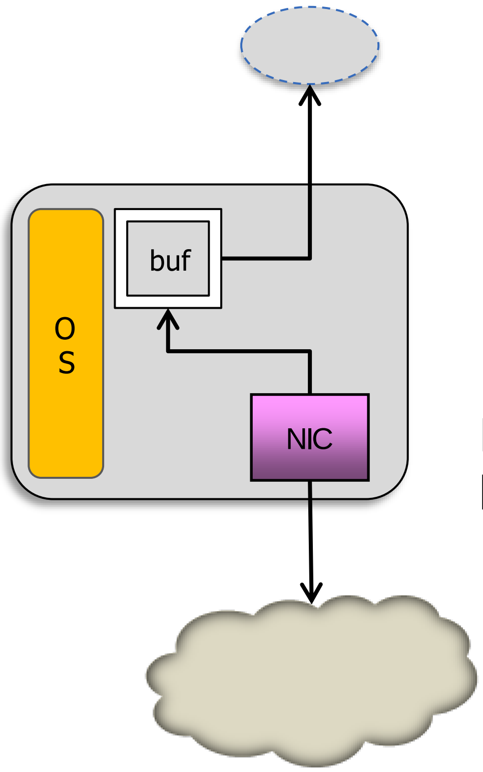
Up until now, this transactional model has been well-served by the ubiquitous server platform

But notice the importance of I/O in the model

So let’s take a quick look at the classic network platform

# A network platform architecture

The basic I/O model hasn't really changed at least since the emergence of Ethernet



An application still uses sockets or some similar API to request I/O service

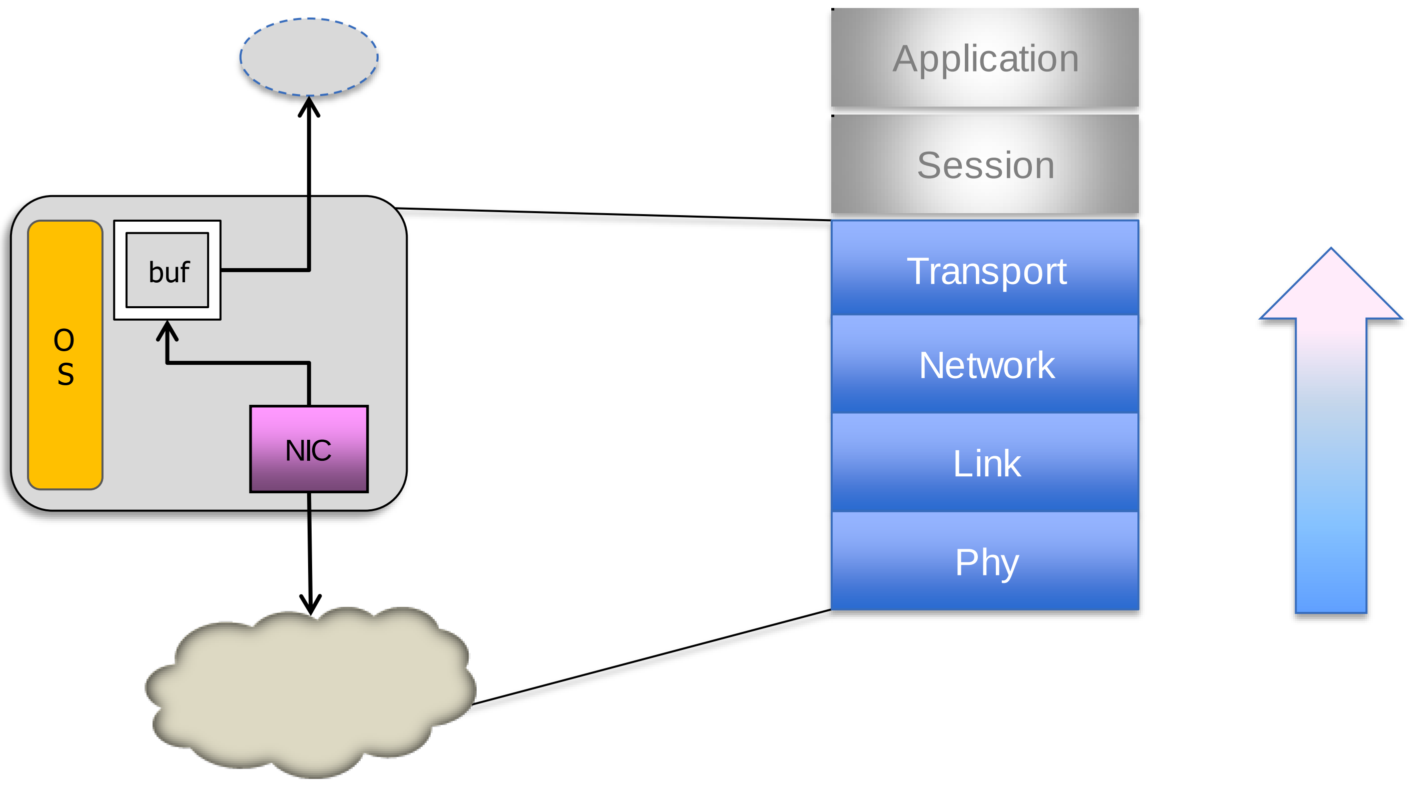
Data is copied through kernel memory

OS is still a central part of the I/O subsystem

"platform-centric I/O"

The network platform is designed to host user applications

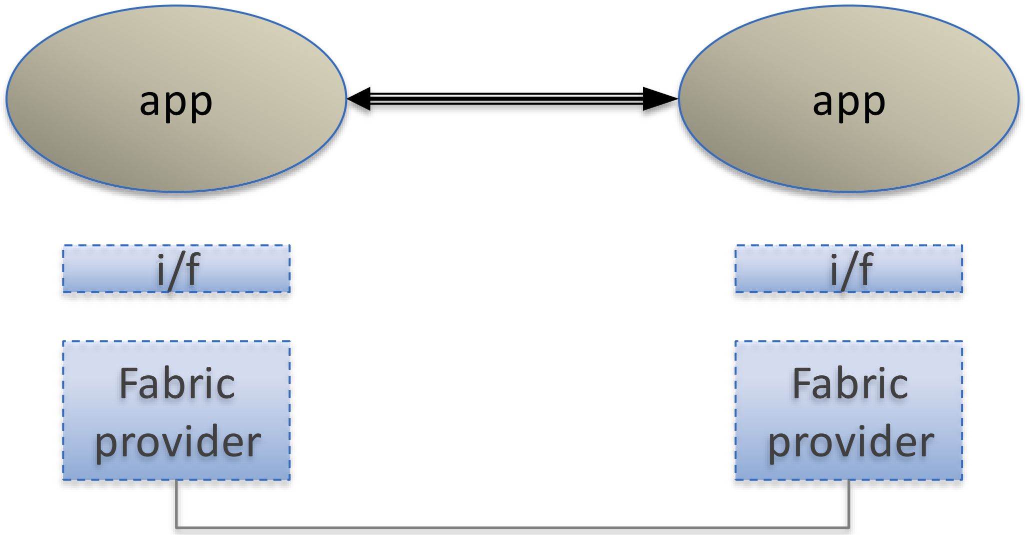
# A network platform architecture



The platform has a ‘bottoms up’ feel to it

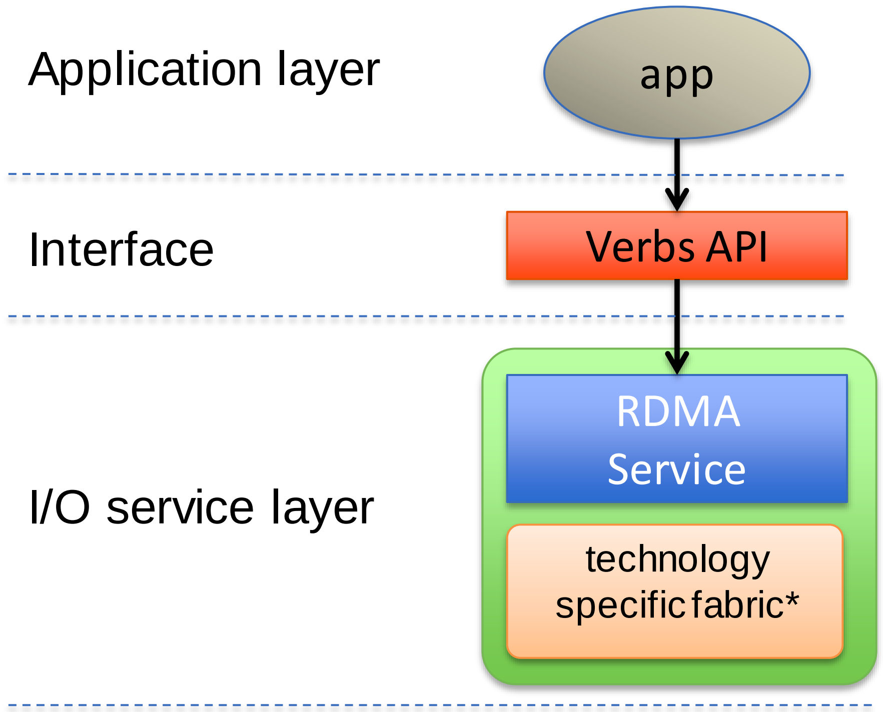
# Application-centric I/O

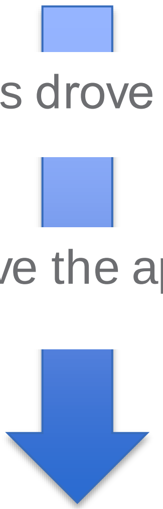
But…
If application communication is so important, wouldn't it make sense to start instead at the application interface and work down?



*"Application-centric I/O" is the art and science of defining an I/O system that maximizes application effectiveness."*

# An example - RDMA





1. app reqmts drove the process…

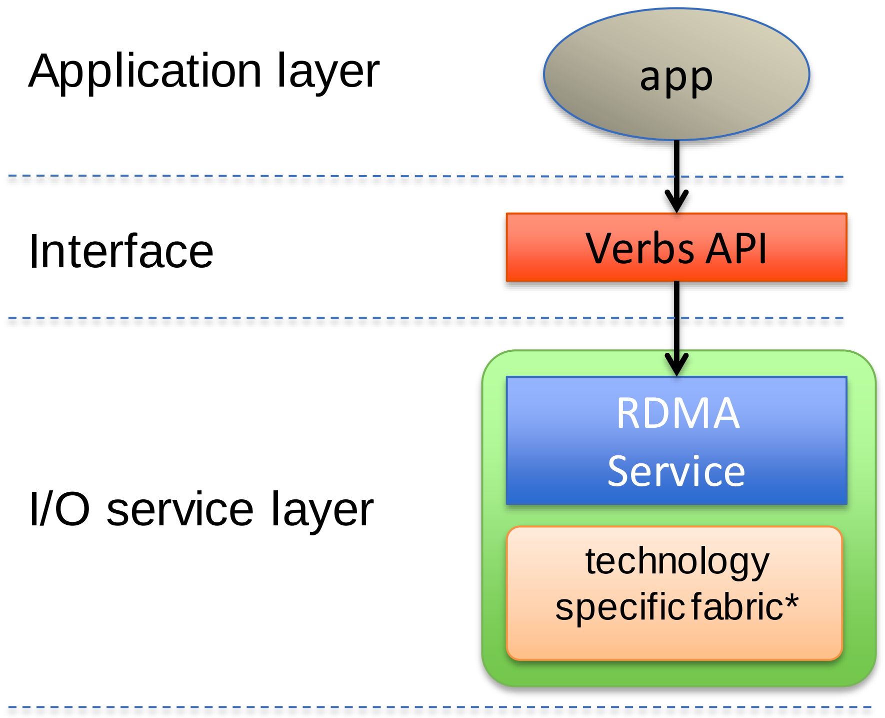
2. which drove the app interface…

3. which drove the fabric design

Objective: scale-out clustering

* Important point! RDMA is independent of the fabric! Could be InfiniBand or an Ethernet fabric.

# An example - RDMA

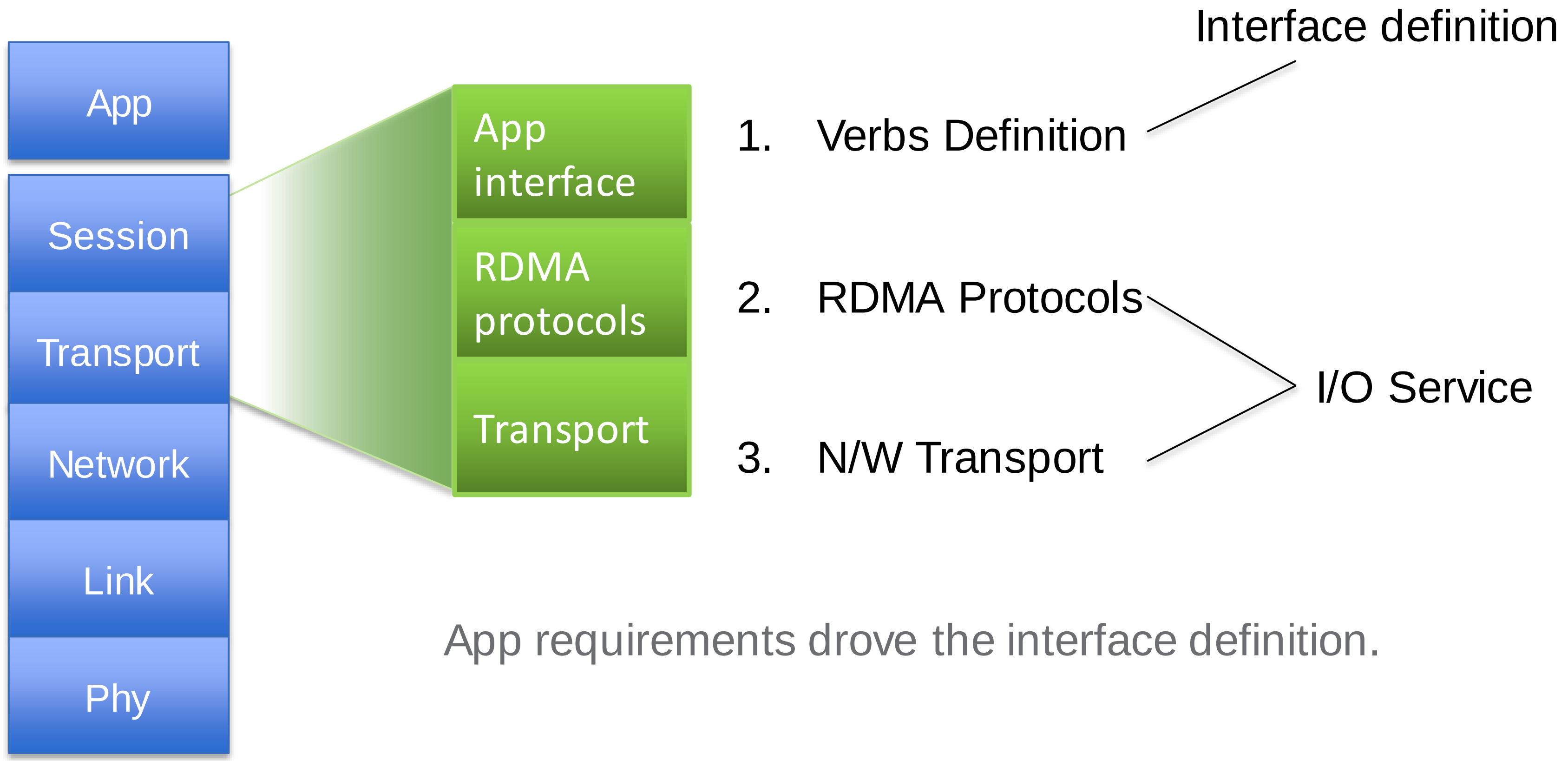


This worked well for the applications for which it was designed; scale-out applications, clustering, storage and other performance or cost sensitive environments

* RDMA Service runs over either an InfiniBand or an Ethernet fabric.

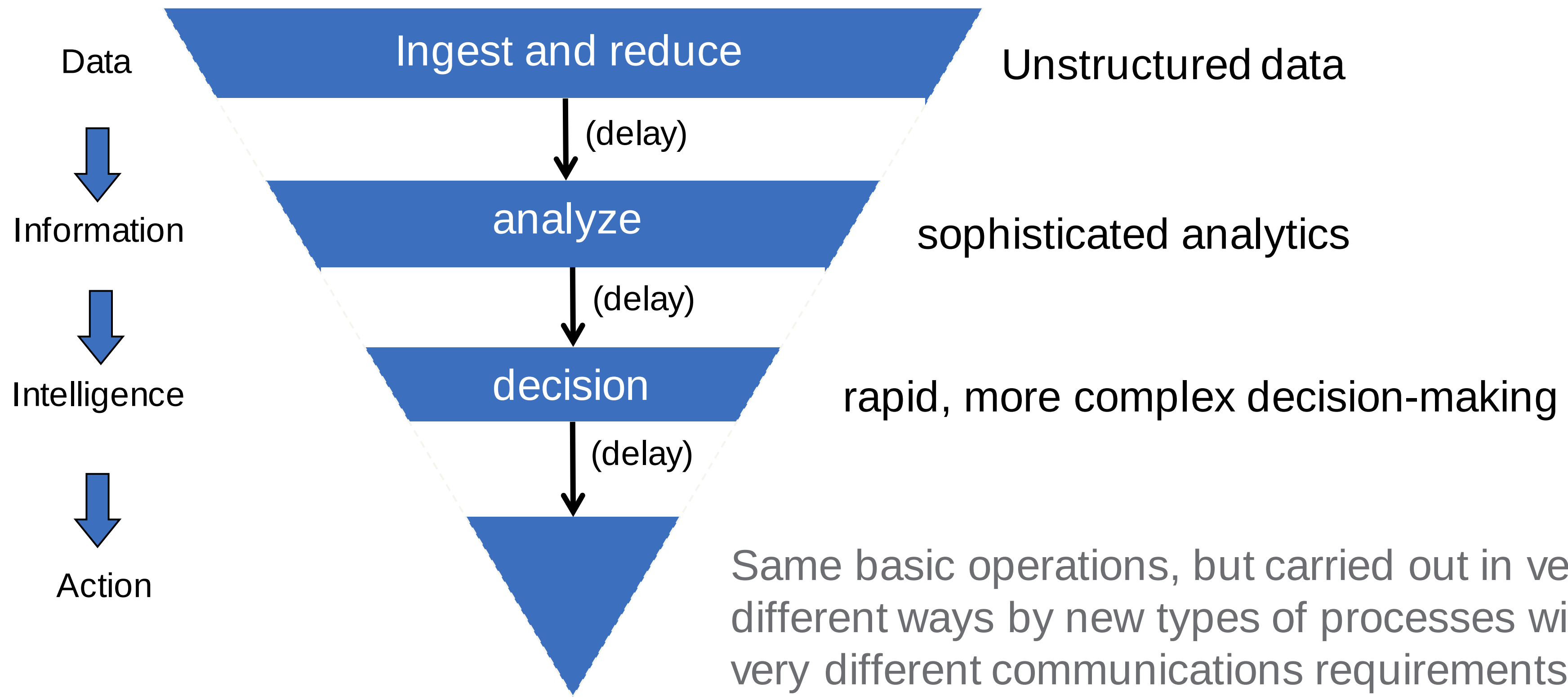
# RDMA architecture

Roughly speaking…



App requirements drove the interface definition.

# Neo-classical data transformation



Same basic operations, but carried out in very different ways by new types of processes with very different communications requirements.

An imperative to re-think the I/O model?

# Evolving uses (short list)

- Demand for methods to ingest, sort and process avalanches of unstructured data
  → Big Data

- Access and store data in new ways
  → the Cloud

- Larger, complex problems requiring new collaboration methods
  → Data access over long distances

In short, "application requirements" continue to shift over time

## A “brief” look at API requirements

- datagram – streaming
- connected, unconnected
- client-server, point to point
- multicast
- tag matching
- active messages
- reliable datagram
- strided transfers
- one-sided reads/writes
- send-receive transfers
- triggered transfers
- atomic operations
- collective operations
- synchronous - asynchronous transfers
- QoS
- ordering, flow control

Courtesy of Sean Hefty

## Observations

- A single API cannot meet all requirements
  - and still be usable
- Any particular app is likely to need only a small subset of such a large API
- Extensions will still be required
  - *There is no correct API!*
- We need more than an updated API – we need an updated *infrastructure*

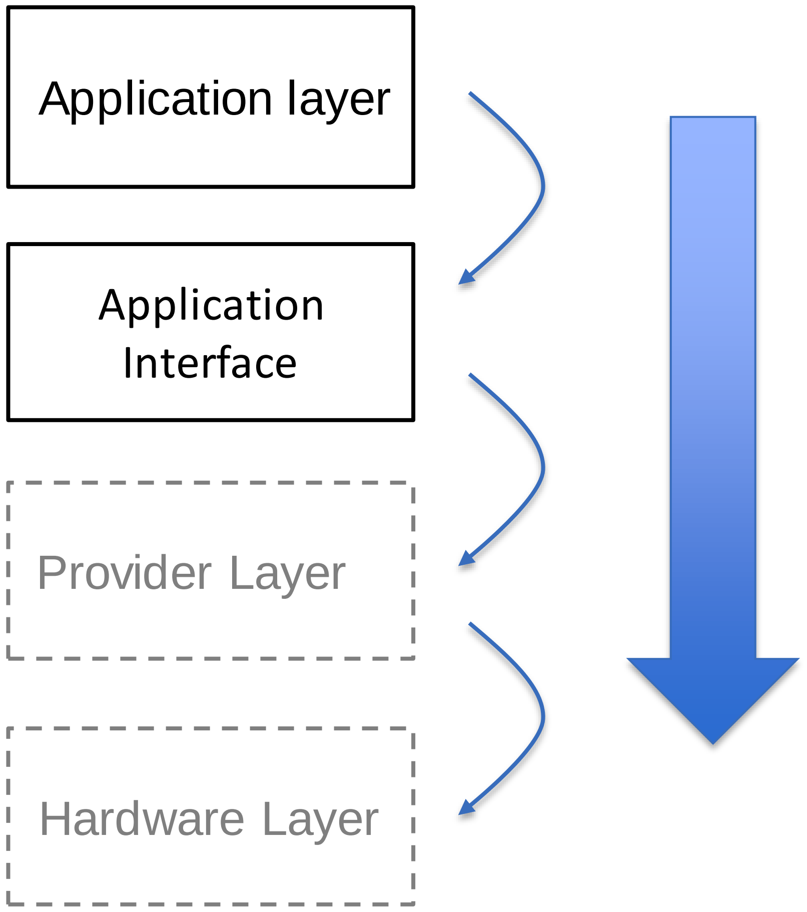
Courtesy of Sean Hefty

## Two things are needed

1. We need *a set of application interfaces* (and hence fabric services) that are responsive to application requirements,

2. We need *a framework* to allow applications to access the I/O interfaces and underlying fabric services.

# Application interfaces



Understand I/O characteristics of the applications of interest

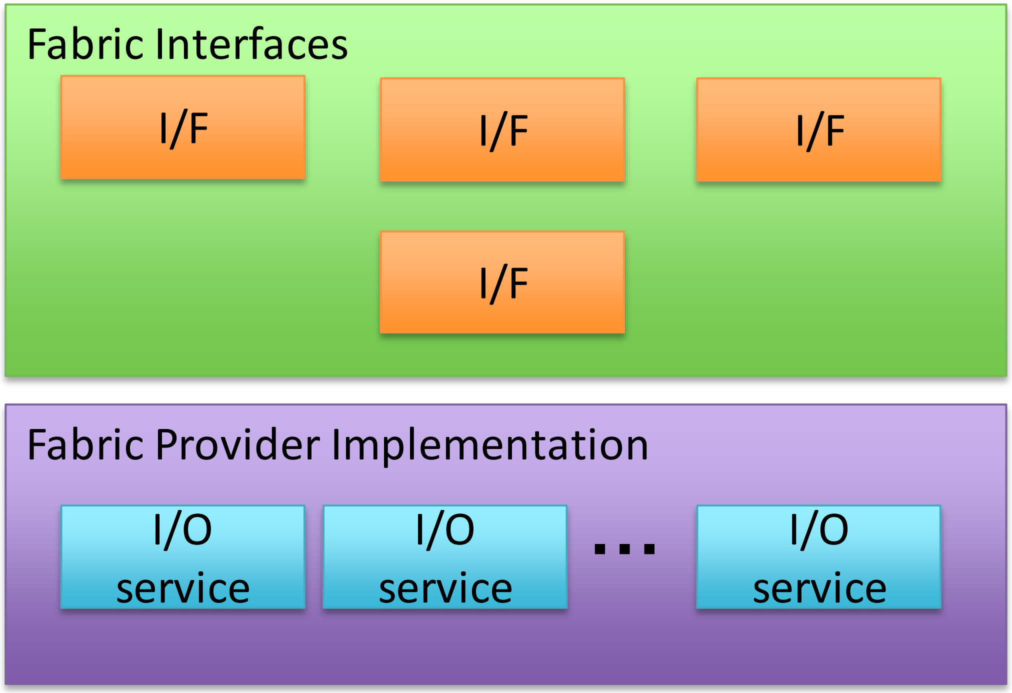
Let those characteristics drive the interface definition(s)

Which ultimately drives the fabric feature set(s)

# A framework

Framework defines multiple interfaces

The framework exports a number of I/O services (e.g. message passing service, large block transfer service, collectives offload service, atomics service…) via a series of defined interfaces.



Vendors provide optimized implementations

* Important point! The framework does not define the fabric.

## OFWG

**<u>O</u>**pen **<u>F</u>**ramework **<u>W</u>**orking **<u>G</u>**roup

Created by the OpenFabrics Alliance to:

Develop, test, and distribute:

1. Extensible, open source interfaces aligned with application demands for high-performance fabric services.

2. An extensible, open source framework that provides access to high-performance fabric interfaces and services.

## Process-wise

OFWG meets weekly, participation is welcomed

Rough priorities:

- Agree on the structure of the required framework
- Understand application requirements
- Design APIs as needed
- Map these onto specific I/O services
- Work with standards groups (e.g. IBTA) to specify new I/O services

Timeline guesstimate – initial release targeted for 12 months

## A request

Your assistance is requested to:

1. Describe the set of applications (or middleware) of interest to the financial services industry

2. Define the set of services demanded by those applications

3. Prioritize as needed

## Useful contacts

OpenFabrics Alliance – www.openfabrics.org

OpenFramework Working Group - http://lists.openfabrics.org/cgi-bin/mailman/listinfo

OpenFramework Working Group co-chairs –

| | |
|---|---|
| Paul Grun (Cray, Inc.) | grun@cray.com |
| Sean Hefty (Intel) | sean.hefty@intel.com |

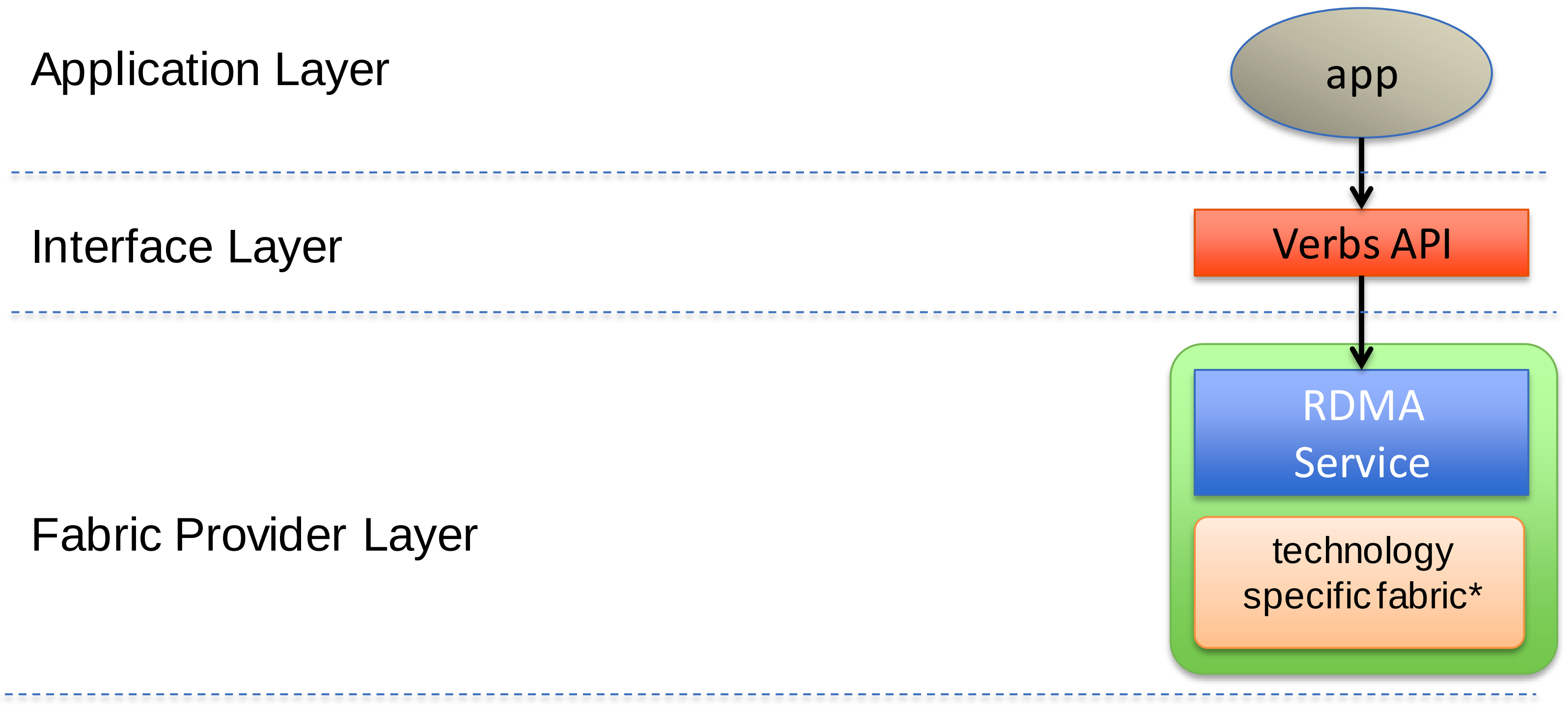
Thank You

OPENFABRICS ALLIANCE

# Abstract

**The next stage in high-performance networking**
The world is adopting new ways to analyze avalanches of data. The data may be stored and accessed in new ways, such as from public and private clouds. And larger, complex problems are demanding new ways to collaborate. The OpenFabrics Alliance (OFA) believes that these trends demand new approaches to high-performance networking. For the past several years, the OFA has led the development of open source software for high-performance, low-latency communication in trading and scientific applications, via technologies such as RDMA. Now the OFA has begun an effort to drive high-performance networks to the next level in scope and performance. By focusing first on application requirements, the OFA intends to bring HPC-style benefits to new use cases. Paul will describe this approach and suggest ways that application developers from the financial services sector can get involved in defining this next push forward in network technology.

# RDMA as we know it today



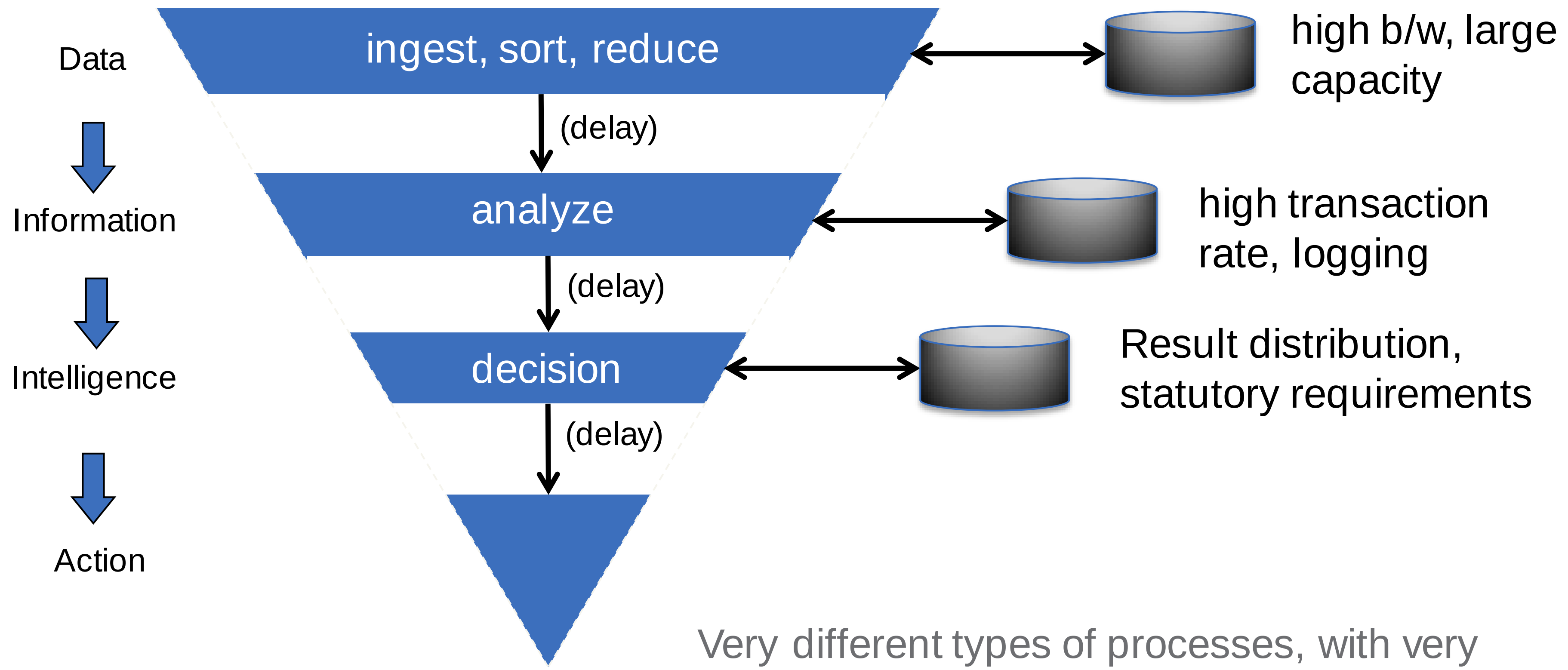
Objective: low latency app-to-app I/O

* The fabric provider could be IB, or iWARP, or RoCE. All are providers of RDMA services.

But we have faster Ethernet coming along, better PCIe and faster memory. Isn't that enough?

No, because faster wires are only part of the equation. Of critical importance are the interfaces used by applications.

# Neo-classical data transformation



Very different types of processes, with very different communications requirements

## Evolving applications

- Manage large volumes of raw data through highly parallel processes
  → e.g. map/reduce

- Tools allowing analysis of large volumes of unstructured data
  → e.g. data analytics

- Flexible storage and data access
  → e.g. cloud storage

- Larger, complex problems are requiring new collaboration methods
  → Data access over long distances

In short, “application requirements” continue to shift over time

# Evolving technology

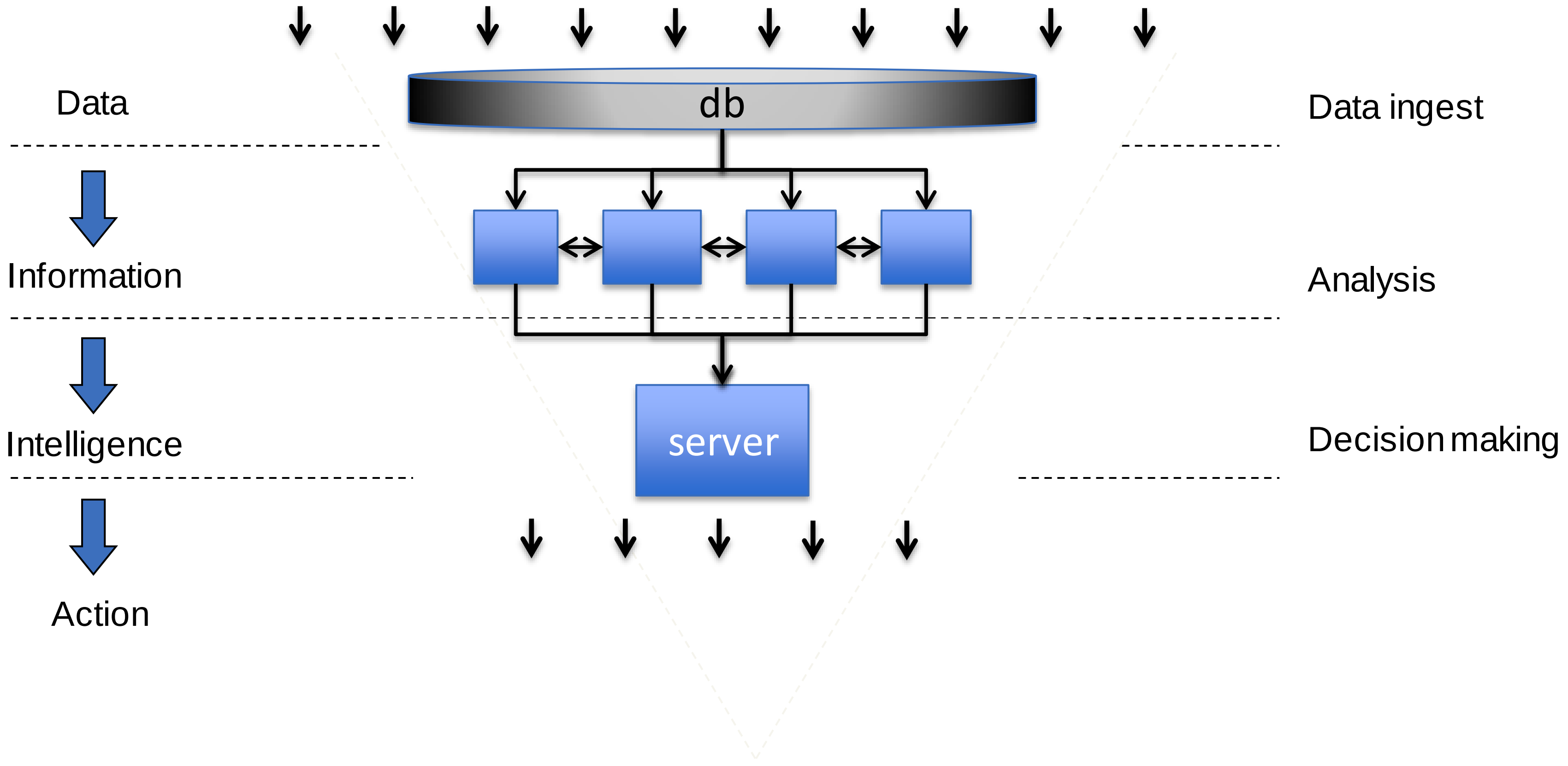
- There are now requirements to scale applications to run on increasingly large clusters

- New requirements to run applications on new compute models e.g. heterogeneous computing (x86 + GPU)

- There are new requirements to run Apps on many-, multi-core processors
- …

What should we be doing to ensure the I/O model remains "application-centric"?

# Classical data transform model



A data movement problem

Each layer comprises compute (servers, clusters or mainframes) and storage