OPENFABRICS ALLIANCE

An Introduction to the OpenFabrics Interface

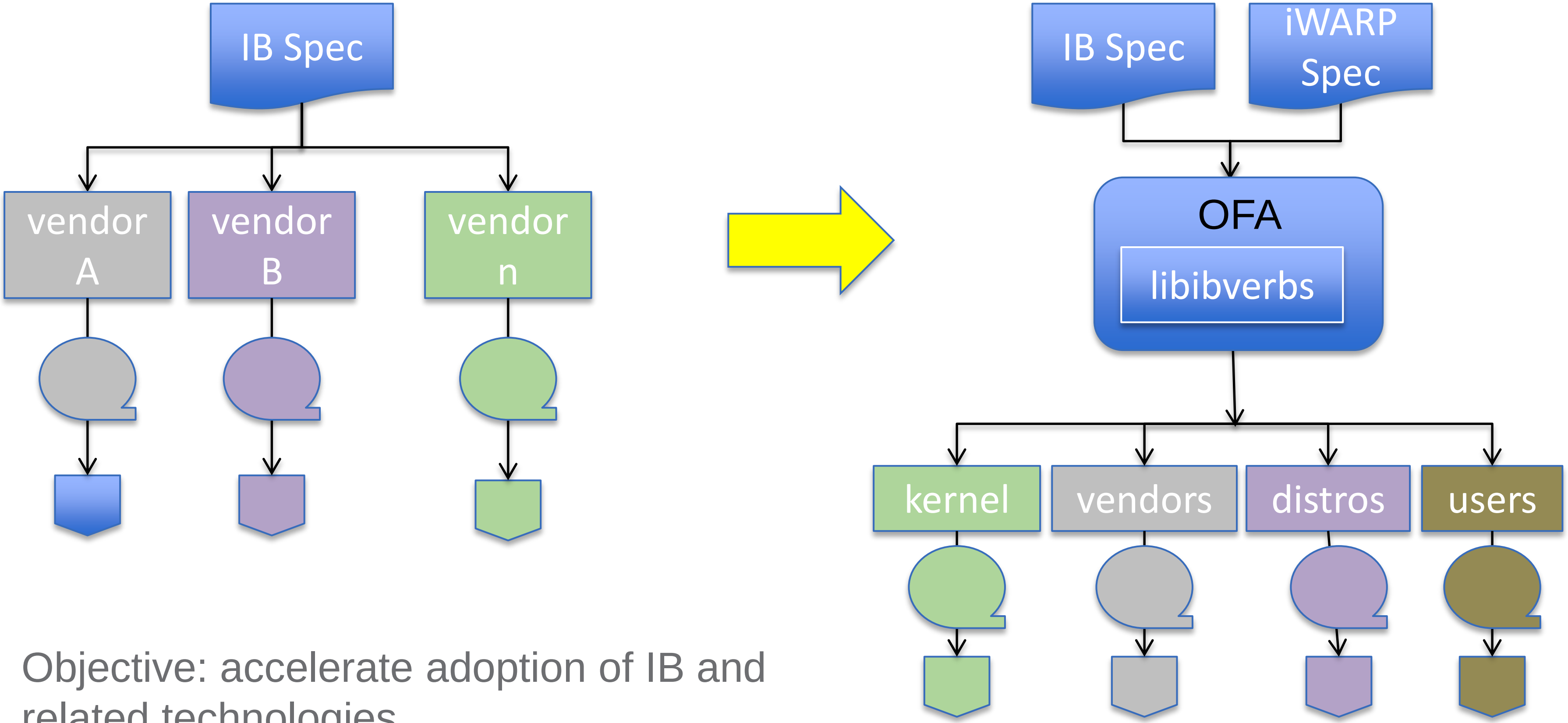
11TH ANNUAL INTERNATIONAL OPENFABRICS SOFTWARE DEVELOPERS' WORKSHOP

#OFAUserGroup
Paul Grun – Cray
w/ slides stolen (with pride) from Sean Hefty

Agenda

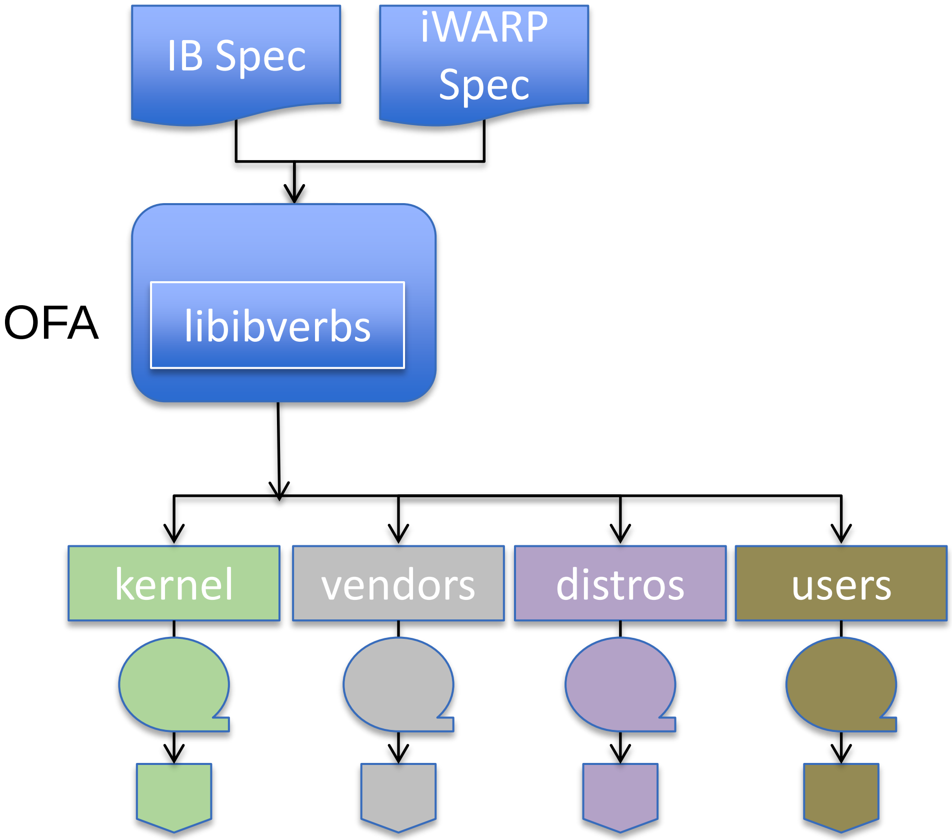
- Where the OFA is going
- Forming the OFI WG
- First Principles
  - Application-centric I/O
  - Transport Agnostic
- Architecture overview
- Release process, timeline
- When should you get involved?

2000 – OpenIB → OpenFabrics



Objective: accelerate adoption of IB and related technologies.
The OFA was mainly an enabling body.

An interesting development



IB Spec

iWARP Spec

OFA

libibverbs

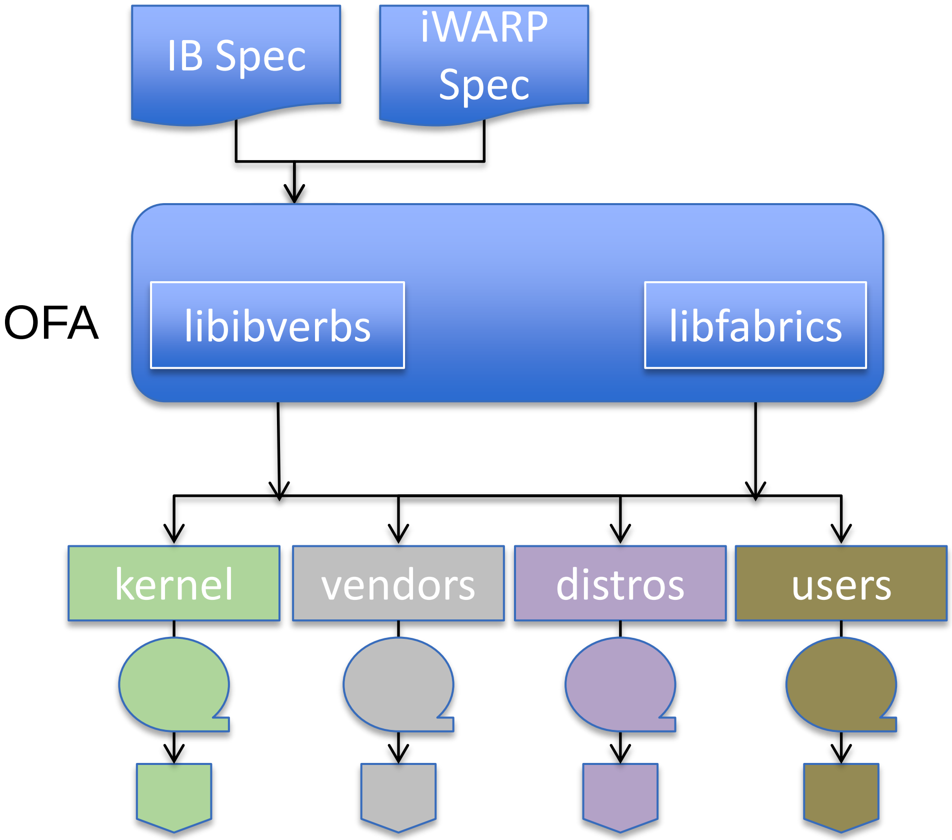
kernel

vendors

distros

users

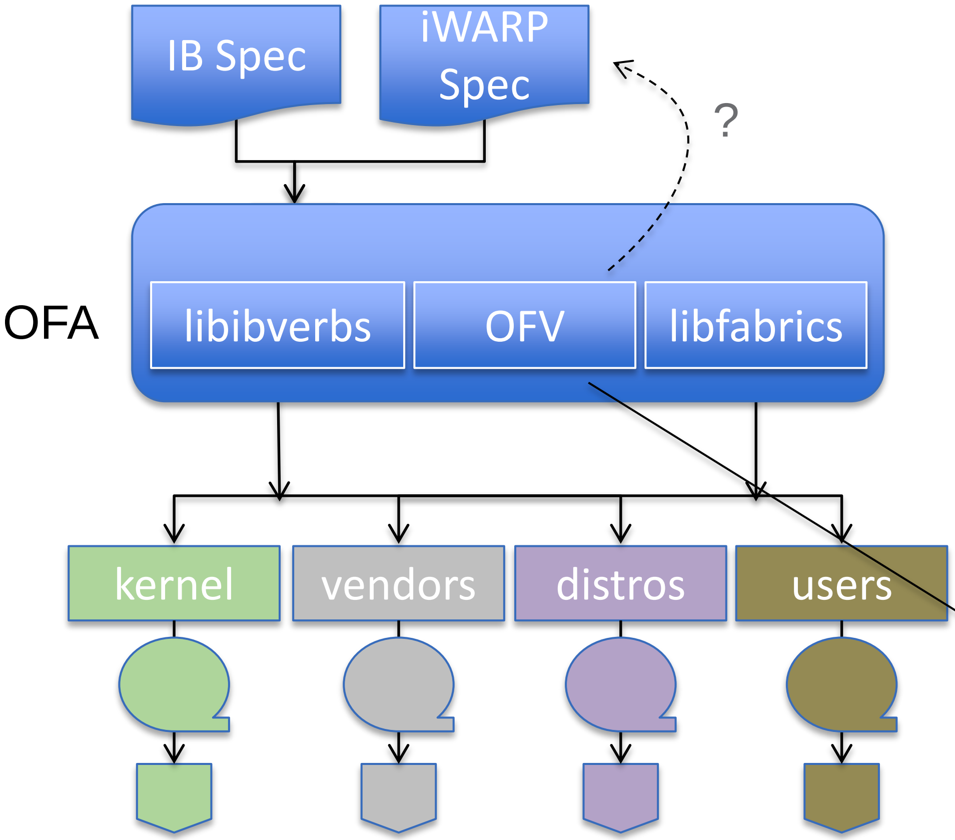
An interesting development



For the 1st time, the OFA is developing new technology, *without* an underlying industry standard specification

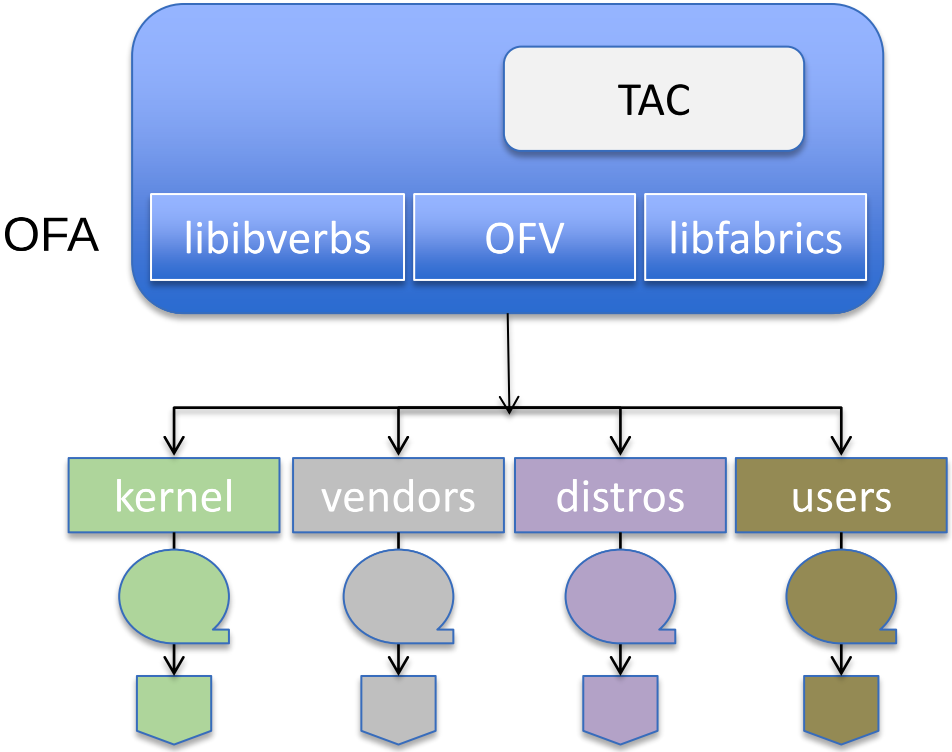
Pure, requirements-driven, technology development

An interesting development



We recently added the OFW WG as a new technical working group

An interesting development

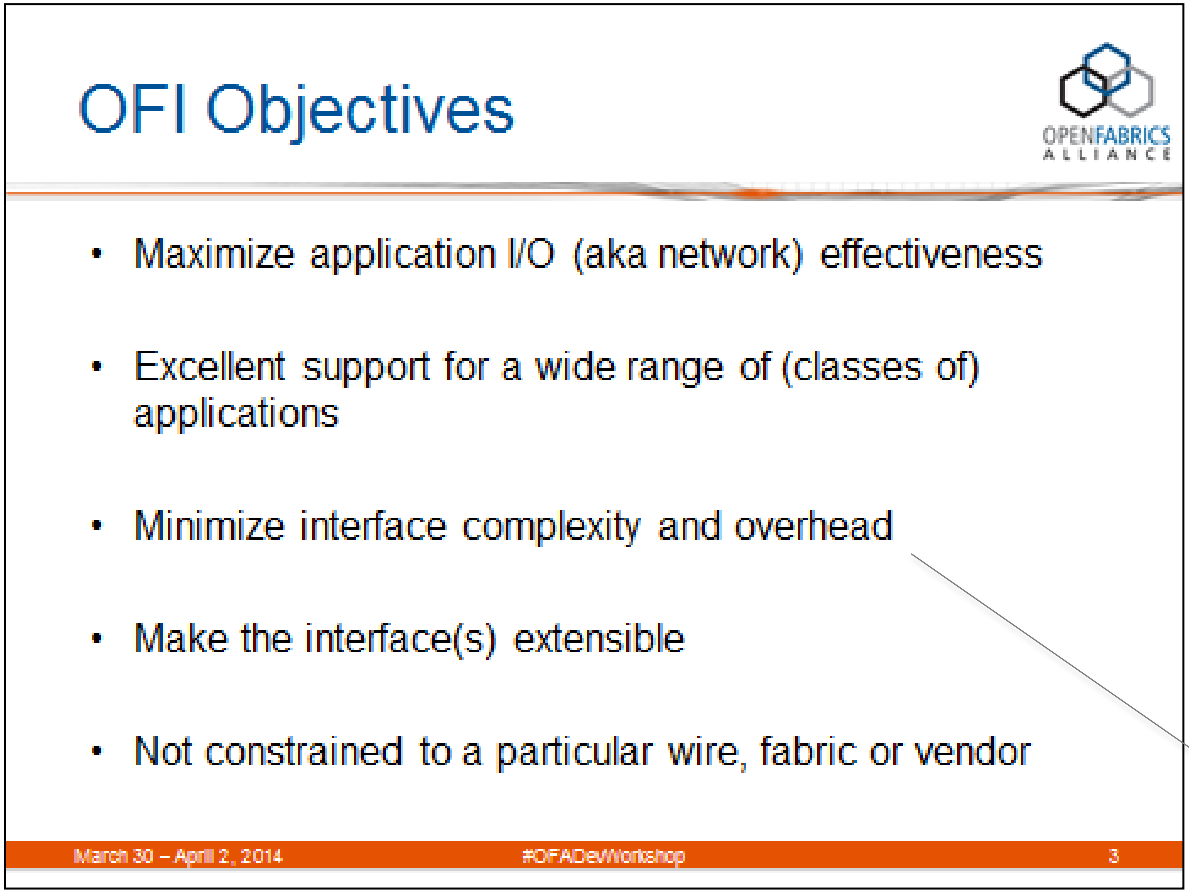


Technical Advisory Council

- OFA’s technology incubator
- observe trends in technology
- clearing house for new ideas
- spawn new development efforts

The OFA has really become a technology development organization

Original Objectives for OFI



OFI Objectives

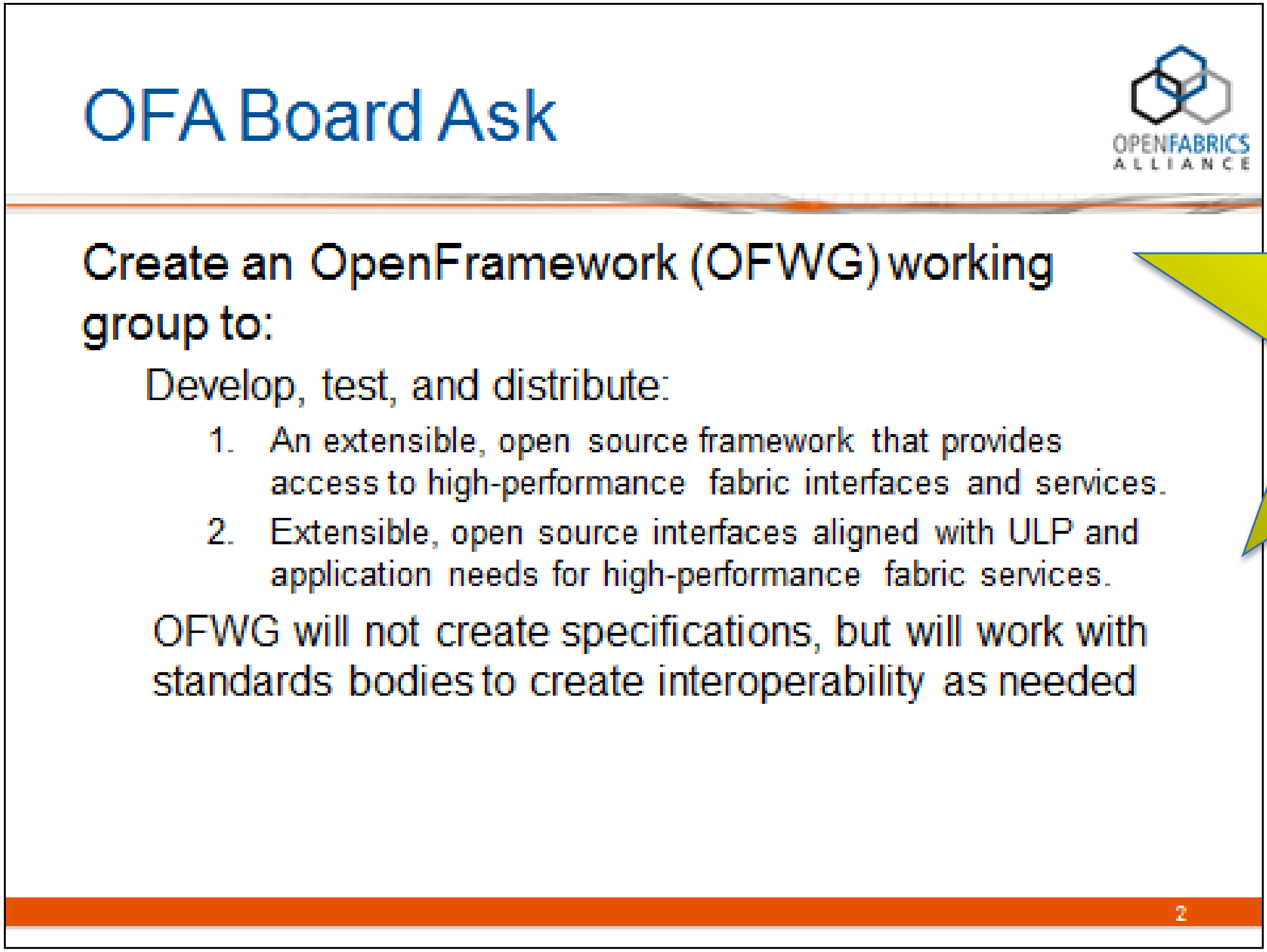
- Maximize application I/O (aka network) effectiveness
- Excellent support for a wide range of (classes of) applications
- Minimize interface complexity and overhead
- Make the interface(s) extensible
- Not constrained to a particular wire, fabric or vendor

March 30 – April 2, 2014 #OFADevWorkshop 3

Pretty much on track!

Maybe not so much

OFA WG



OFA Board Ask

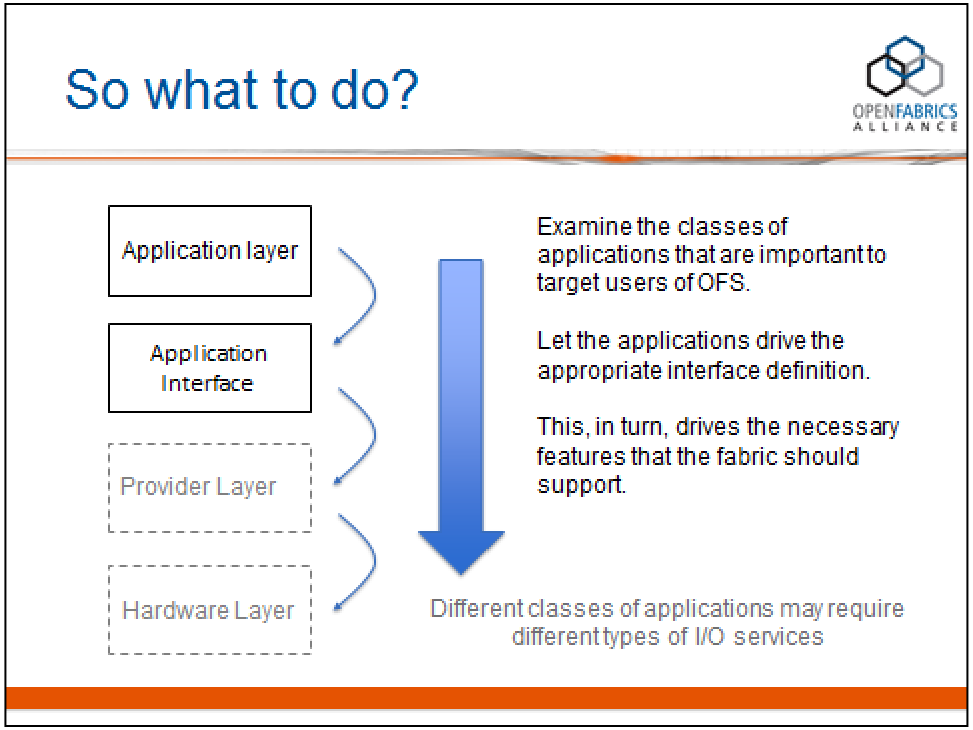
Create an OpenFramework (OFWG) working group to:

Develop, test, and distribute:

1. An extensible, open source framework that provides access to high-performance fabric interfaces and services.
2. Extensible, open source interfaces aligned with ULP and application needs for high-performance fabric services.

OFWG will not create specifications, but will work with standards bodies to create interoperability as needed

1st Principle



So what to do?

Application layer

Application Interface

Provider Layer

Hardware Layer

Examine the classes of applications that are important to target users of OFS.

Let the applications drive the appropriate interface definition.

This, in turn, drives the necessary features that the fabric should support.

Different classes of applications may require different types of I/O services

What are the central requirements of consumers of a network API?

“Application-centric I/O” is the art and science of defining an I/O architecture to maximize application effectiveness”

Consumer orientation has emerged as a key watchword for the OFA

Classes of Applications

OFI APIs are being driven by requirements from specific classes of applications

Legacy apps (skts, IP)

- Skts apps
- IP apps

Data Analysis

- Structured data
- Unstructured data

Data Storage, Data Access

- Filesystems
- Object storage
- Block storage
- Distributed storage
- Storage at a distance

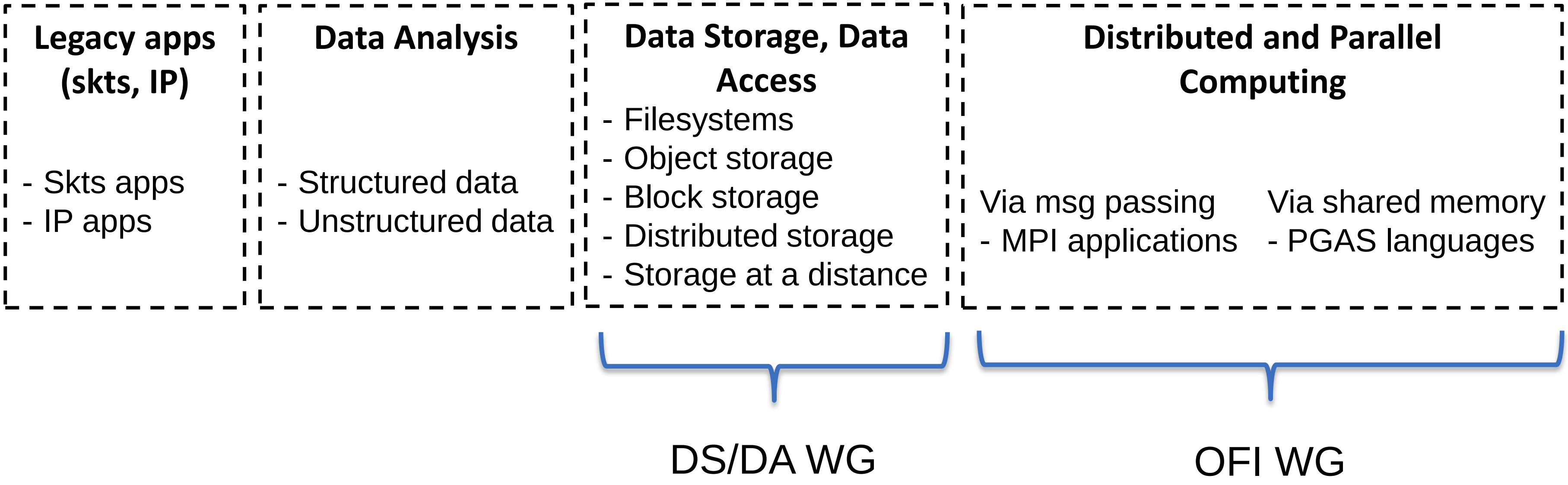
Distributed and Parallel Computing

Via msg passing
- MPI applications

Via shared memory
- PGAS languages

Classes of Applications

OFI APIs are being driven by requirements from specific classes of applications



Legacy apps (skts, IP)

- Skts apps
- IP apps

Data Analysis

- Structured data
- Unstructured data

Data Storage, Data Access

- Filesystems
- Object storage
- Block storage
- Distributed storage
- Storage at a distance

Distributed and Parallel Computing

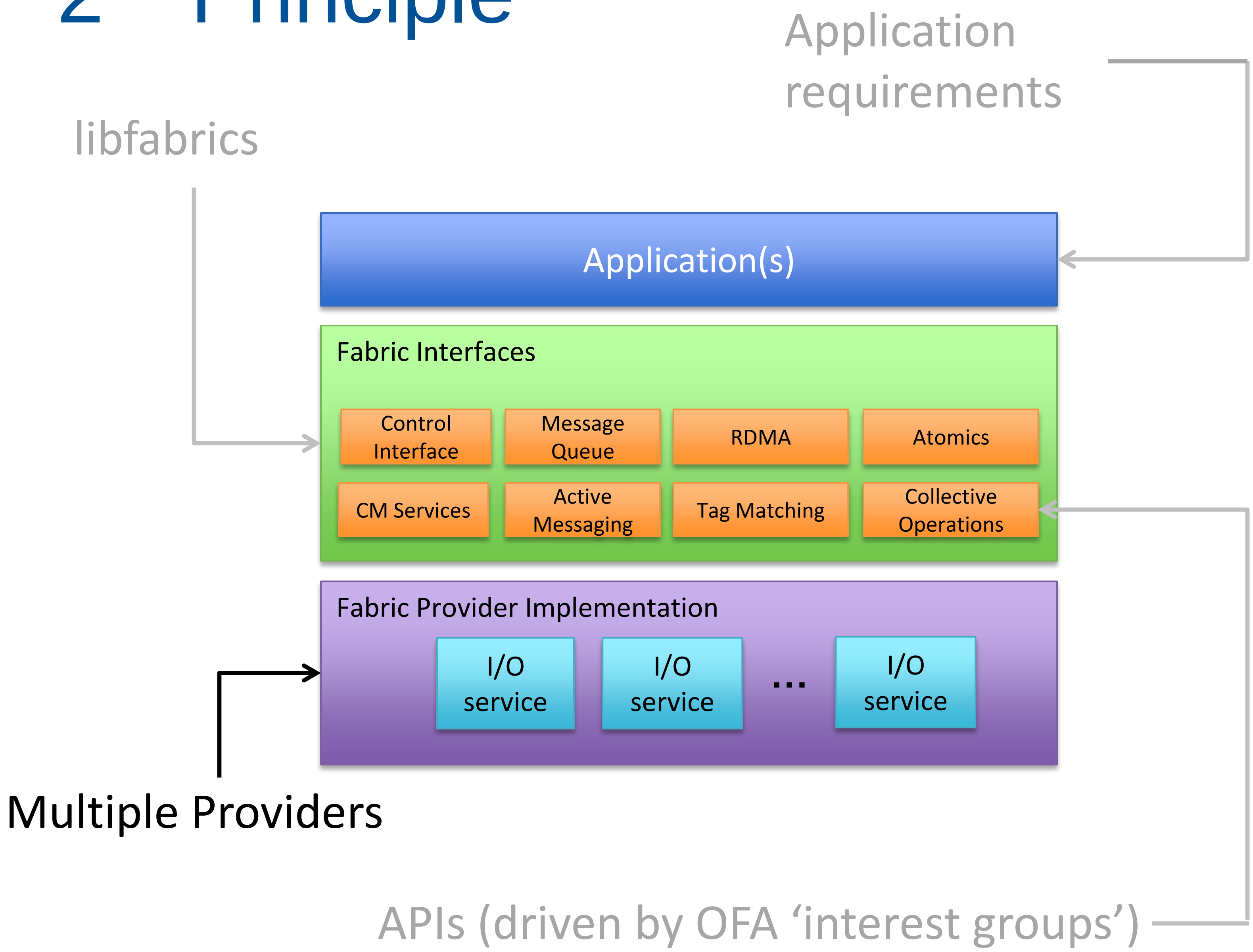
Via msg passing
- MPI applications

Via shared memory
- PGAS languages

DS/DA WG

OFI WG

2nd Principle



Application requirements

libfabrics

Application(s)

Fabric Interfaces

Control Interface

Message Queue

RDMA

Atomics

CM Services

Active Messaging

Tag Matching

Collective Operations

Fabric Provider Implementation

I/O service

I/O service

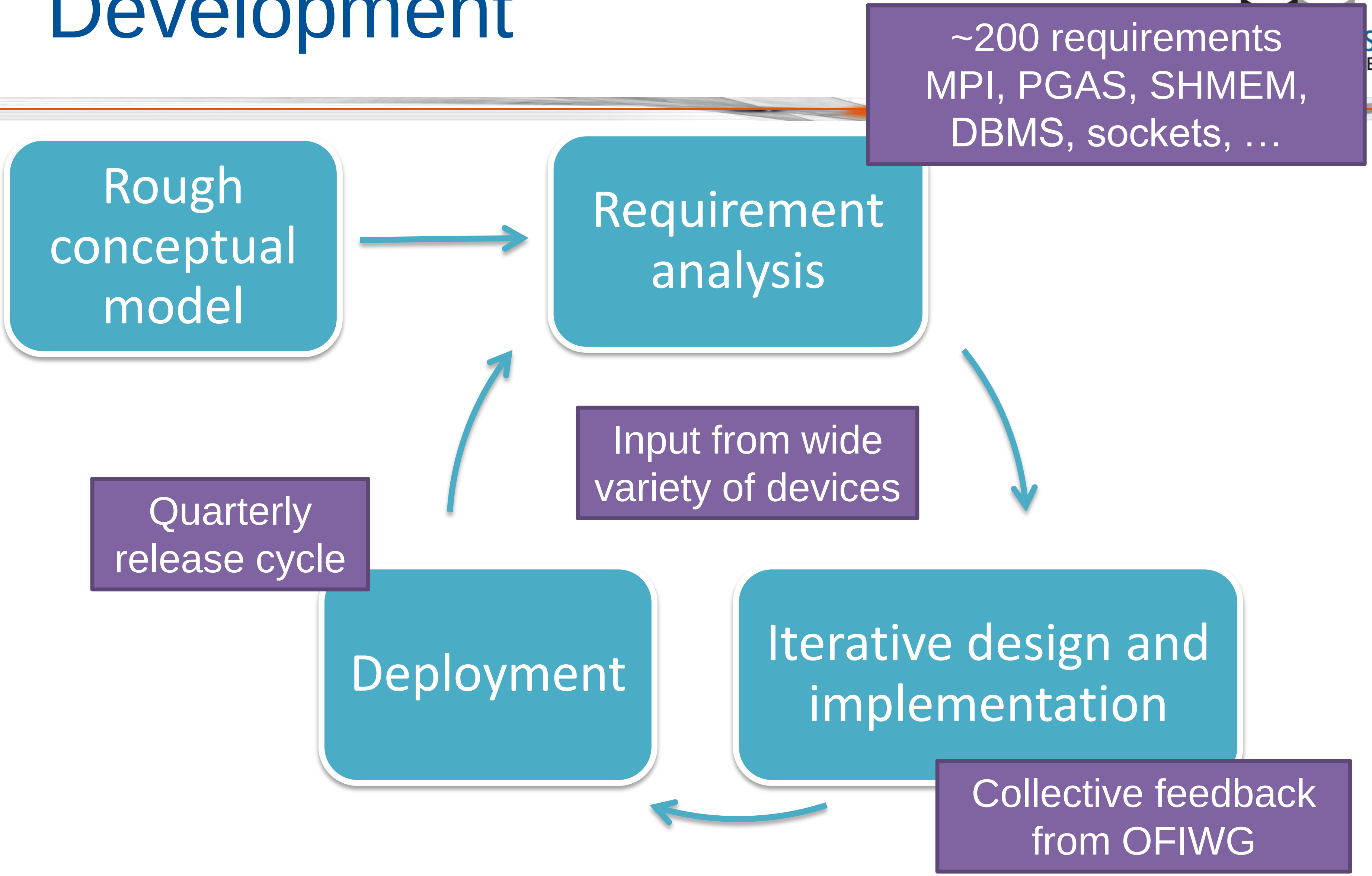
…

I/O service

Multiple Providers

APIs (driven by OFA ‘interest groups’)

Development



~200 requirements
MPI, PGAS, SHMEM, DBMS, sockets, …

Rough conceptual model

Requirement analysis

Input from wide variety of devices

Iterative design and implementation

Collective feedback from OFIWG

Deployment

Quarterly release cycle

Design

Implementation Agnostic

EASY

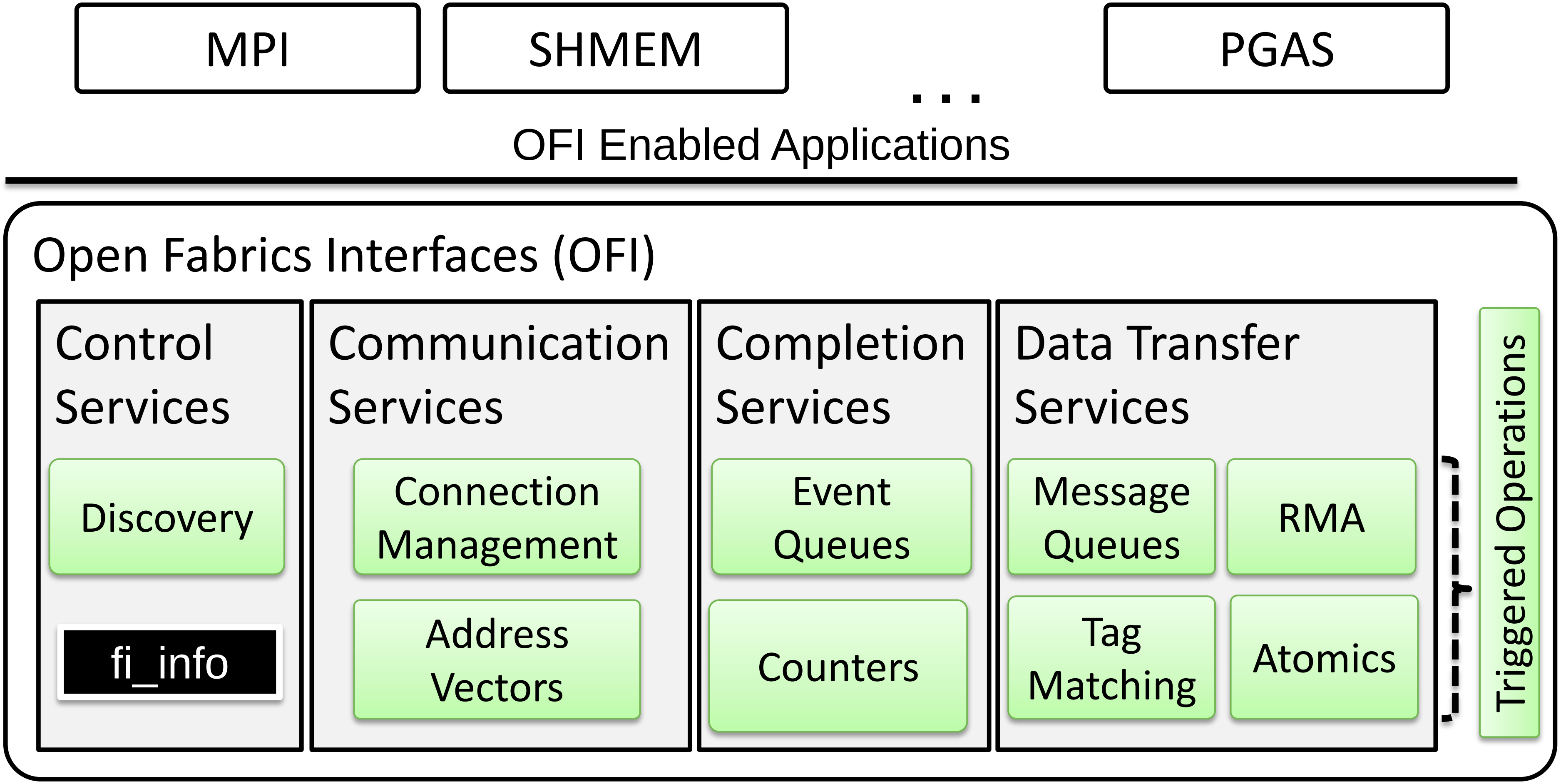
- Enable simple, basic usage
- Move functionality under OFI

GURU

- Advanced application constructs
- Expose abstract HW *capabilities*

Range of usage models

Architecture



EASY

Fabric Information

Endpoint Types

- MSG
  - Reliable connected
- DGRAM
  - Datagram
- RDM
  - Reliable datagram messages
  - Reliable unconnected

Capabilities

- Message queue
  - FIFO
- RMA
- Tagged messages
  - Sends match with specific receive buffers
- Atomics

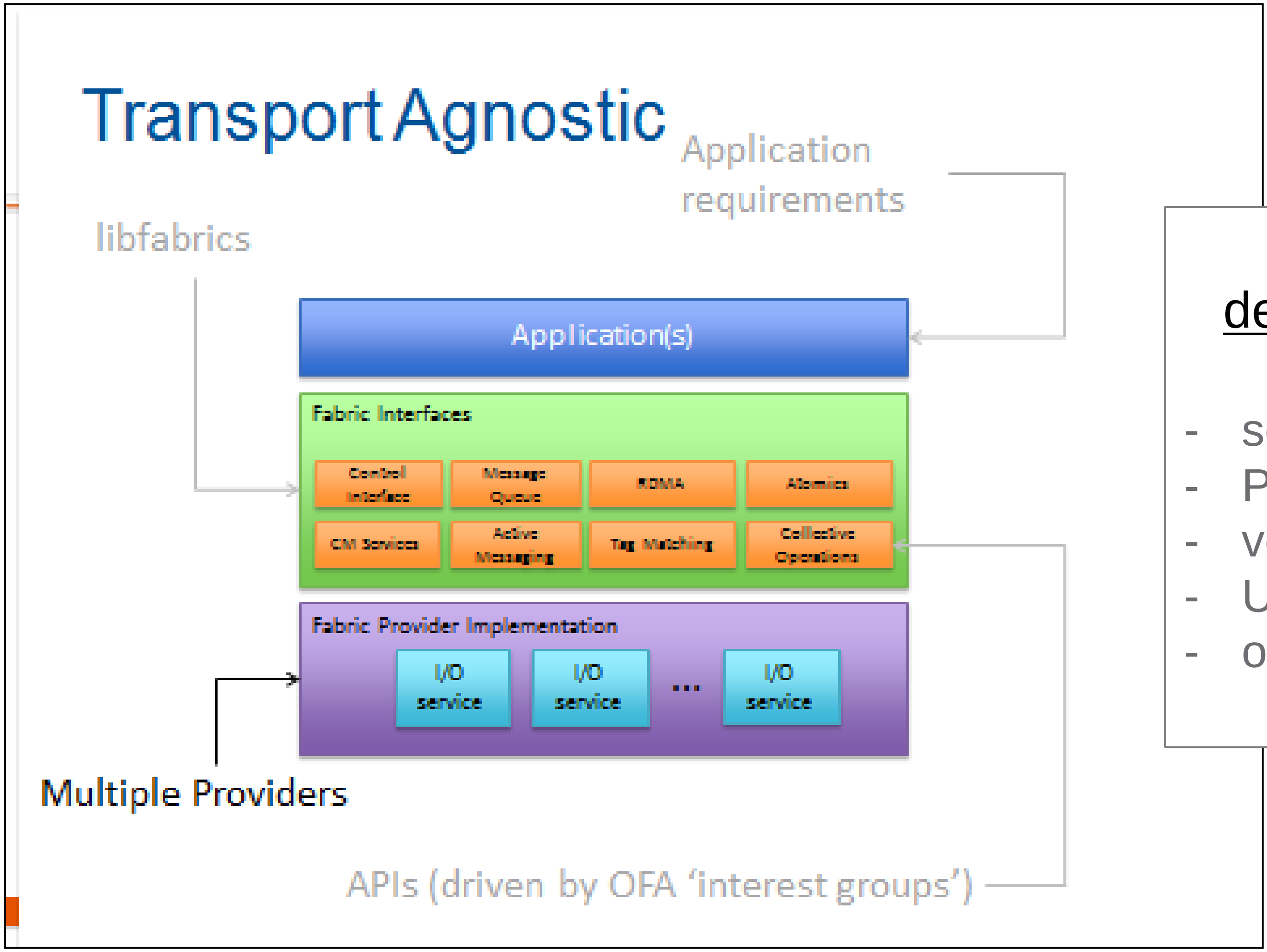
Select desired endpoint type and capabilities

GURU

Fabric Information

- Capabilities
  - Application desired *features* and *permissions*
  - Primary capabilities
    - Must be requested by application
  - Secondary capabilities
    - May be requested by application
    - May be offered by provider
- Attributes
  - Defines the *limits* and *behavior* of selected interfaces
  - Negotiated
- Mode
  - Provider request on application

Providers



<u>Under development for Release 1.0</u>

- sockets
- PSM
- verbs
- Usnic
- others expected

OFI 1.0 Providers

- Sockets
  - Implement all interfaces and functionality
  - App. development & debug
- Verbs
  - Targets any verbs HW
    - Not optimized for a specific device
  - Only common verbs functionality supported
- PSM
  - Targets non-verbs HW
  - Expands capabilities beyond lower software driver
- USNIC
  - Targets non-verbs HW
  - Cisco will address

Input from verbs derivative and non-verbs providers also fed into OFI design

Release timeline

- time-based release process
- quarterly releases planned for the early stages
- Release 1.0 rc2 available now
- Release 1.0 rc3 planned for end of March
- Release 1.0 soon thereafter

When to get involved

- Walking a fine line:
  - Don’t want to release too early, but…
  - Need to get the broader community involved

The time to get advanced developers involved is now!

“If Verbs Programming is like the assembly language version of network programming,
OFI is the C language version.”

Doug Ledford

Thank You

OPENFABRICS
ALLIANCE

11TH
ANNUAL
INTERNATIONAL
OPENFABRICS SOFTWARE
DEVELOPERS' WORKSHOP