OPENFABRICS ALLIANCE

A Taste of OFI

Sean Hefty

11TH ANNUAL INTERNATIONAL OPENFABRICS SOFTWARE DEVELOPERS' WORKSHOP

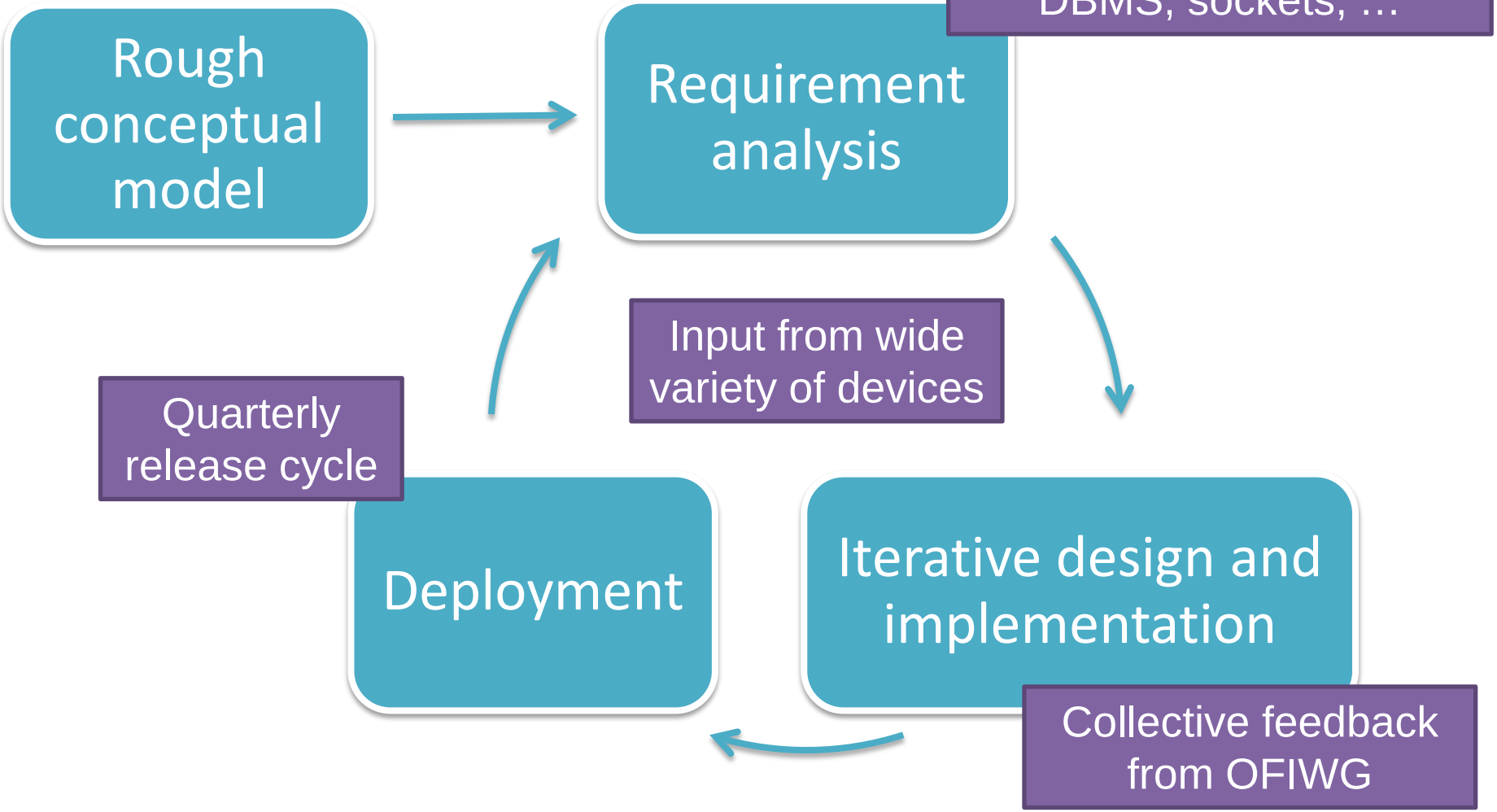
#OFADevWorkshop

Taste of OFI

Elegant and intellectually satisfying with subtle undertones?

Selected analysis of the Open Fabrics Interfaces architecture and implementation

Development



Application Requirements

*Give us a **high**-level interface!*

*Give us a **low**-level interface!*

And this was just the MPI developers!

Try talking to the government!

Design

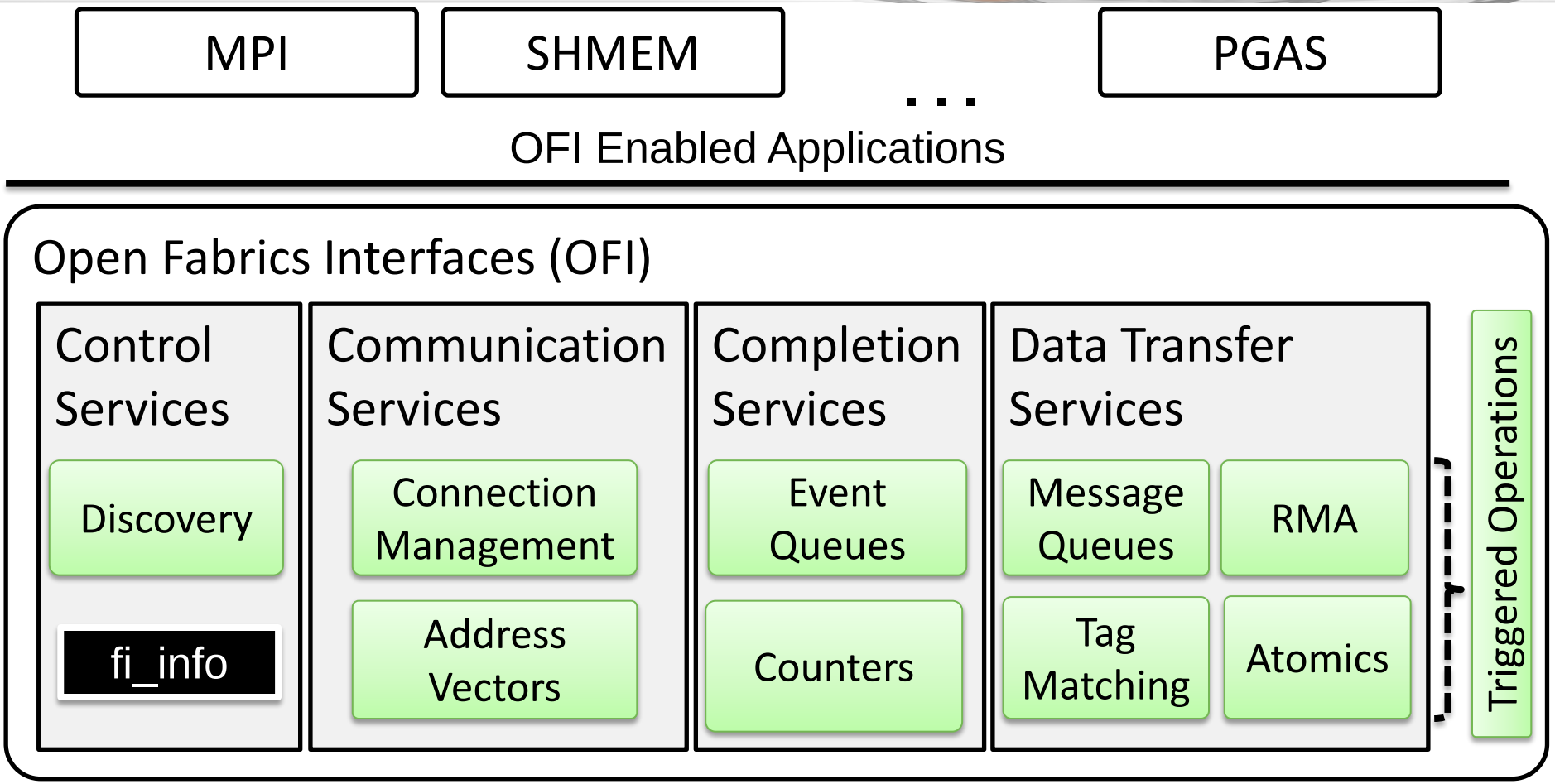
Implementation Agnostic

- Enable simple, basic usage
- Move functionality under OFI

- Advanced application constructs
- Expose abstract HW *capabilities*

Range of usage models

Architecture



Fabric Information

OPENFABRICS ALLIANCE

Endpoint Types

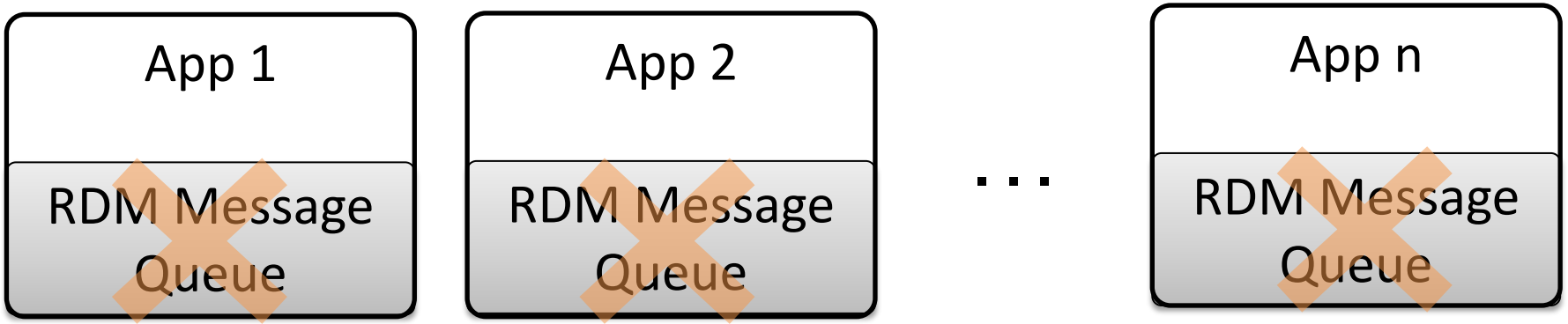
- MSG
  - Reliable connected
- DGRAM
  - Datagram
- RDM
  - Reliable datagram messages
  - Reliable unconnected

Capabilities

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

Select desired endpoint type and capabilities

Fabric Information



RDM Message Queue Common Implementation

DGRAM Message Queue

Fabric Information

OPENFABRICS ALLIANCE

- 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

Threading Options

Is synchronization needed?

Fully thread safe

Identify resource usage constraints needed for lockless access

Example:

thread 1: {endpoint 1, CQ 1}

thread 2: {endpoint 2, CQs 2-3}

Progress

What thread context does a request use to complete?

Automatic – submit and forget

Manual – application thread used to complete request

- Timeouts and retries, ack processing, atomic operations, data placement, etc.

Resource Management

Flow control and queuing

Enabled – prevent overrunning local and remote queues, including completion queues

Disabled – application responsible for preventing overruns

App: *behavior* that can be relied on.
Provider: *requirements* and *potential optimizations* not *restrictions*.

Ordering

Sequence of request and completion processing

Strict – requests are processed and completed in order

Relaxed – enable out of order processing

- Dynamic routing, optimized completion processing, parallel data transfers, optimized retry algorithms, etc.

Mode Bits

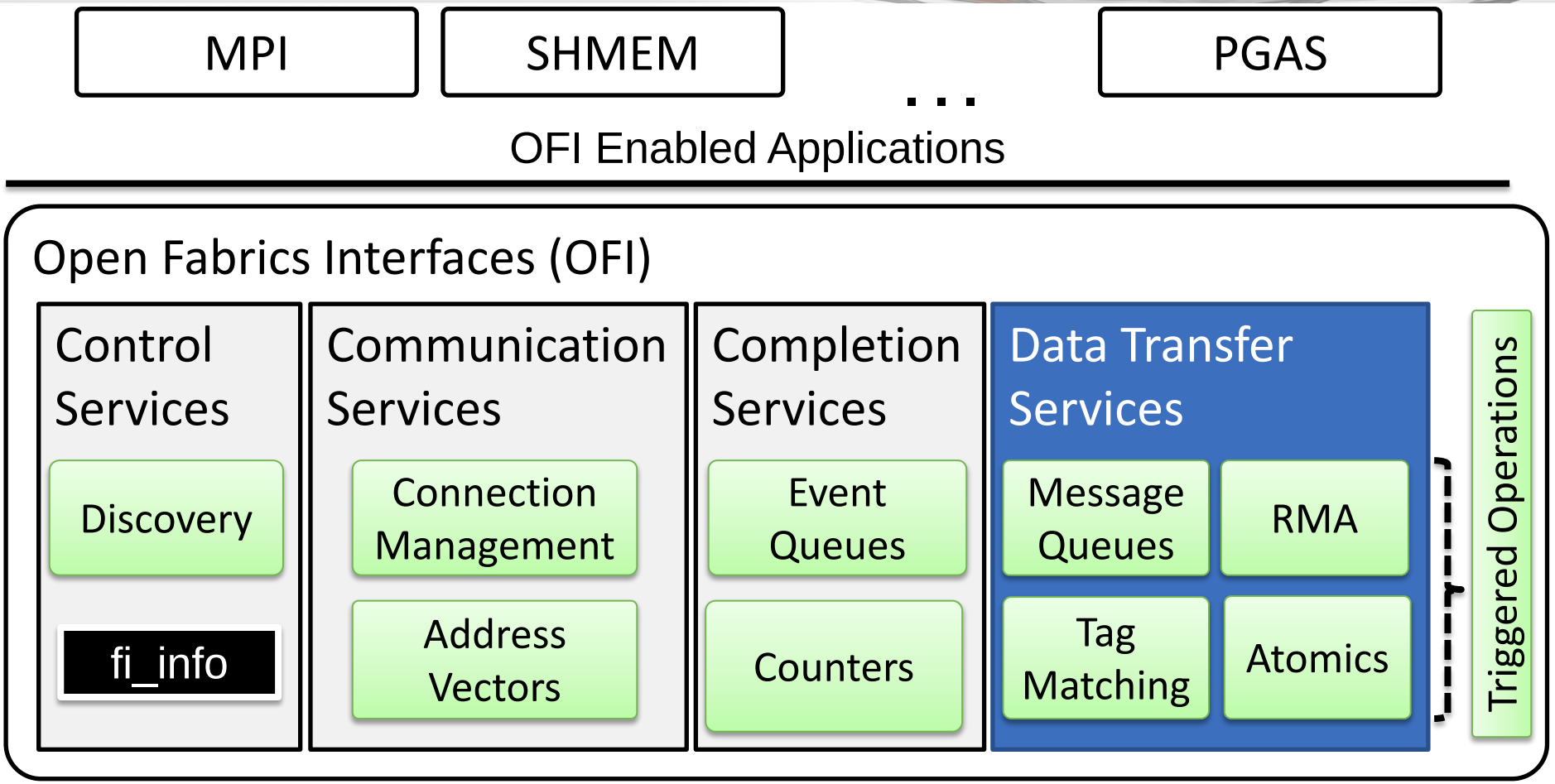
Provider hints on how it should best be used

None! – provider does all the work!

Application support may improve performance

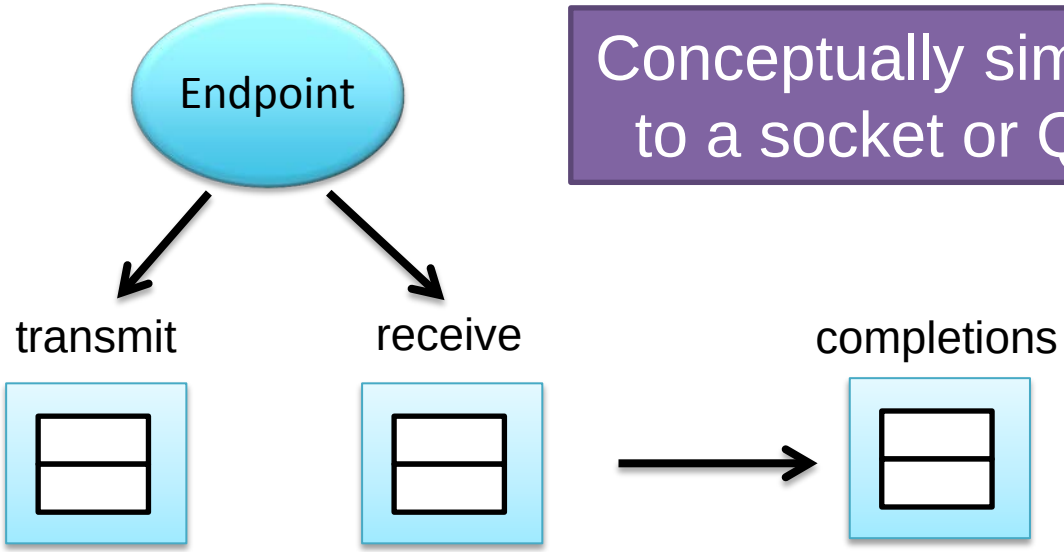
- Local MR – must register buffers for local operations (send/receives)
- Context – app provides ‘scratch space’ for provider to track request (full or partial onload models)
- Buffer prefix – app provides space for network headers (usnic, IB GRH)

Architecture



Endpoints

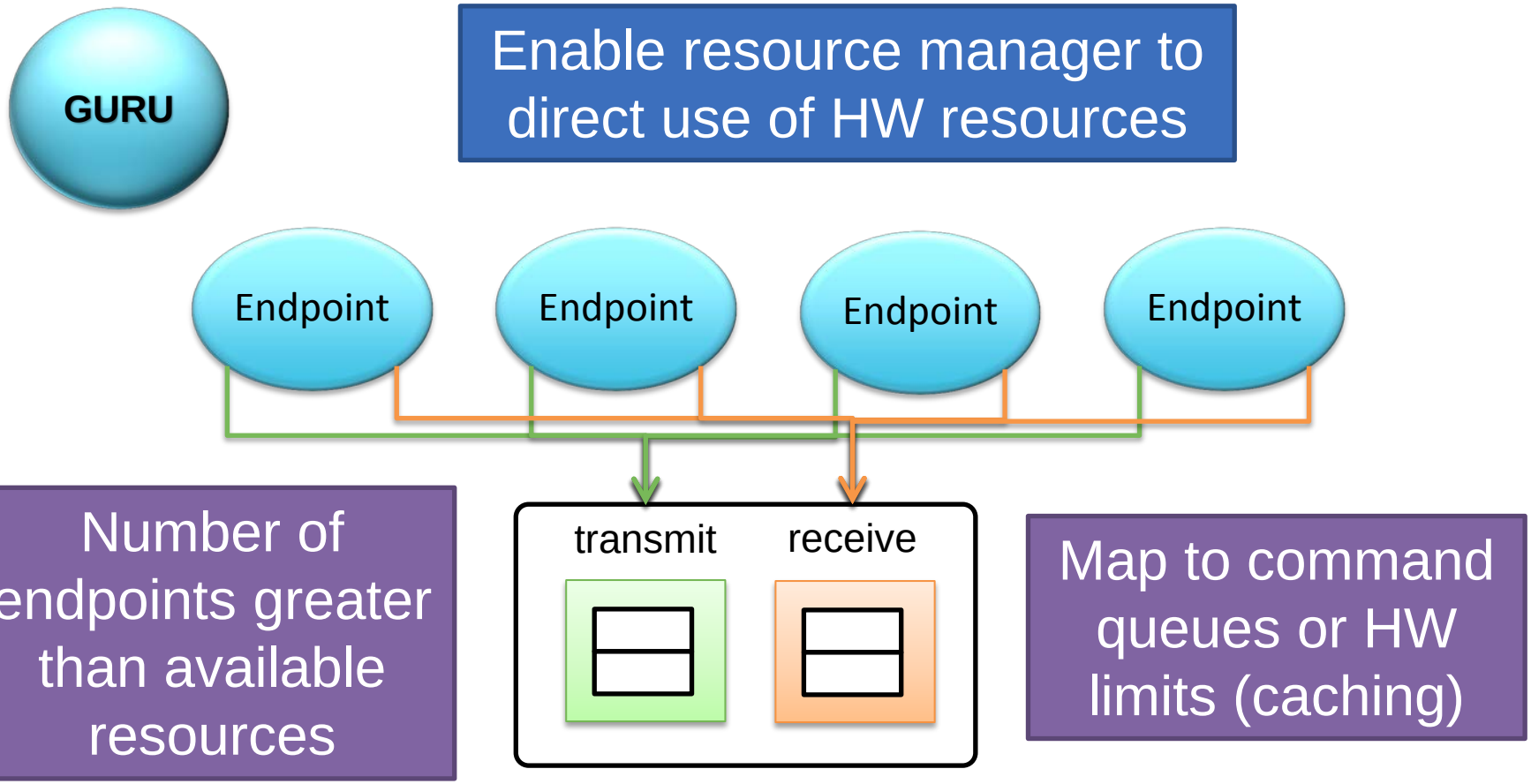
Addressable communication portal



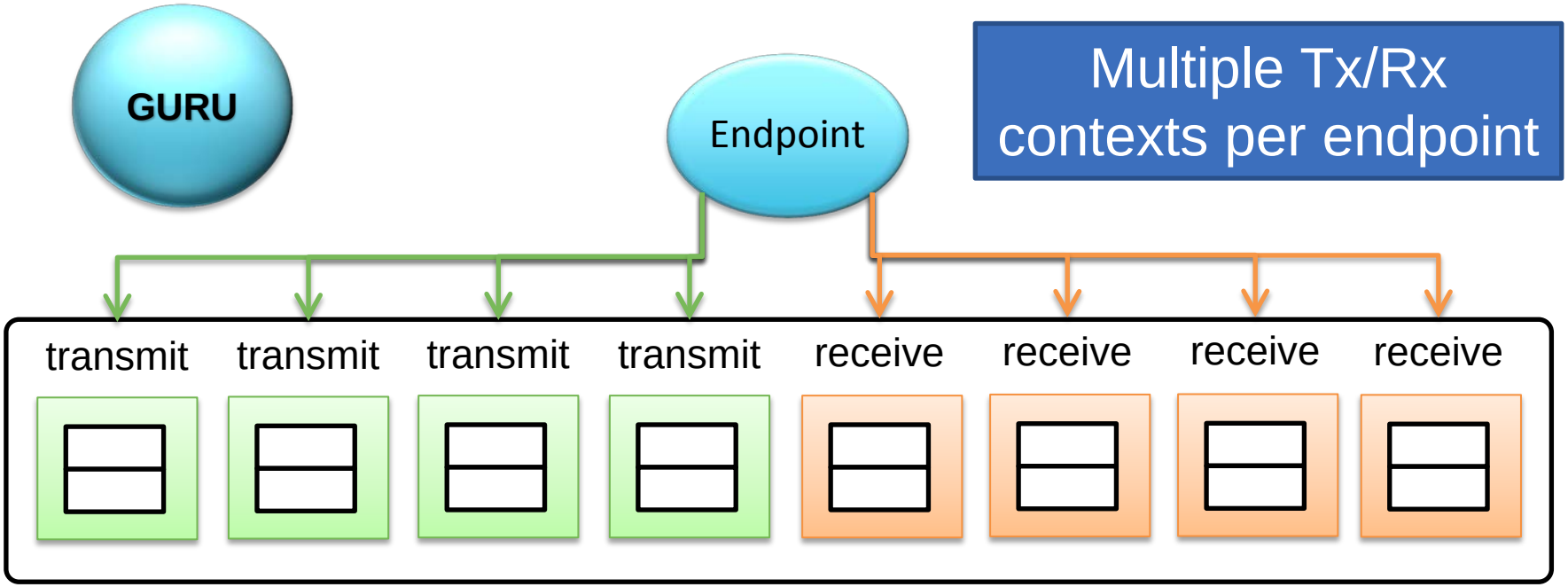
Conceptually similar to a socket or QP

Sequence of request and completion processing

Shared Tx/Rx Contexts



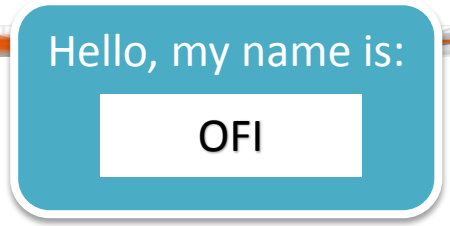
Scalable Endpoints



- Multi-threading
- Ordering
- Progress
- Completions

Incoming requests may be able to target a specific receive context

OFI Is a Full Meal Deal



- Select a main dish and a side

- OFI tells you today's specials
- You select the ingredients and tell OFI how to assemble them

flair

We treat you right!

OFI 1.0

- Framework – libfabric
  - Interfaces & data structures definitions
  - 'Spec' = man pages
- Functional implementation
  - Quickly enable hardware and fabrics
    - Portions layer over vendor interfaces
  - Allow for application development
  - Amount and quality of support is provider specific

Important to distinguish between architecture and direction versus current implementations

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

OFI 1.x

- Address other requirements
  - Multicast
  - Virtualization
  - Features cut from 1.0 release
- Expand and optimize providers
  - Native providers
  - Additional hardware

Thank You

OPENFABRICS ALLIANCE

11th ANNUAL INTERNATIONAL OPENFABRICS SOFTWARE DEVELOPERS' WORKSHOP