14th ANNUAL WORKSHOP 2018

MOVING FORWARD WITH FABRIC INTERFACES

Sean Hefty, OFIWG co-chair

Intel Corporation

April, 2018

USING THE PAST TO PREDICT THE FUTURE

- ✓ *OFI Provider Infrastructure*
- ✓ *OFI API Exploration*
- ✓ *Companion APIs (Bonus!)*

1.5 API Updates

- RxM provider
- SOCK endpoint types
- Memory registration
- API optimizations

2017	v1.4.0..	..1.4.2	v1.5.0..	..1.5.3
2018	v1.6.0..	*v1.6.1*	*v1.6.2*	*v1.7.0*

1.6 Provider Enhancements

- PSM2 – native
- RxM performance
- SHM – shared memory support
- Persistent memory

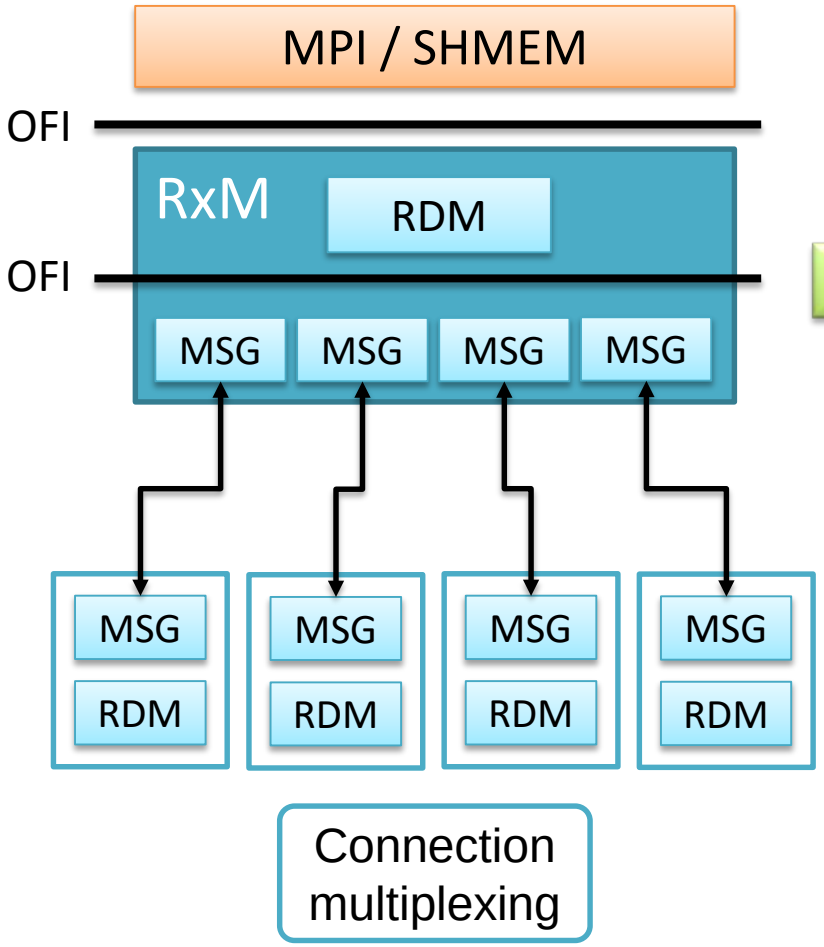
1.7 Predictions

- New providers
  - RxD, multi-rail, new vendors
- SHM – xpmem support
- API enhancements

PROVIDER INFRASTRUCTURE

ARUN AND DMITRY'S AMAZING RXM PROVIDER

High-priority



Connection multiplexing

Primary path for HPC apps accessing verbs hardware

Verbs

NetworkDirect

TCP

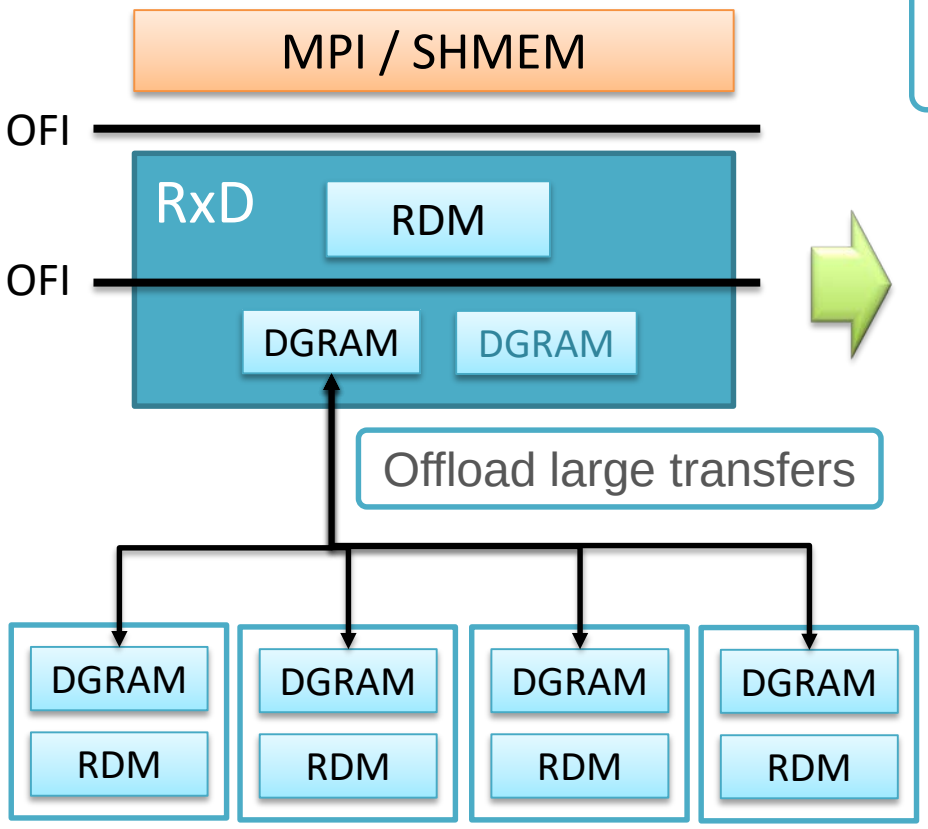
TCP to replace sockets

Optimizes for hardware features

- **Strong MPI performance**
- **Evaluating tighter provider coupling**

RXD

Focus for v1.7



Future path for HPC scalability

- Verbs UD
- usNIC
- UDP
- Raw Ethernet
- Other..?

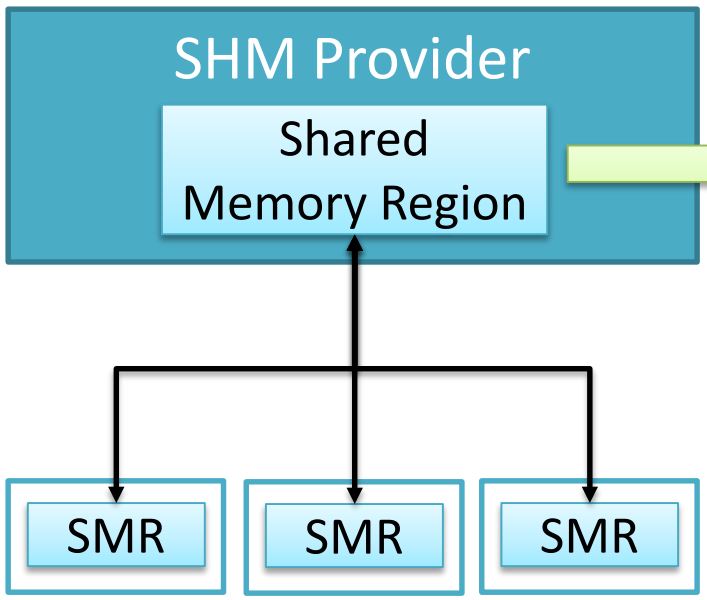
Fast development path for hardware support

Extend features of simple RDM provider

- **Re-designing for performance and scalability**
- **Analyzing provider specific optimizations**

ALEXIA'S FANTASTIC SHARED MEMORY PROVIDER

Now available in stores near you!



One-sided and two-sided transfers

Shared memory primitives

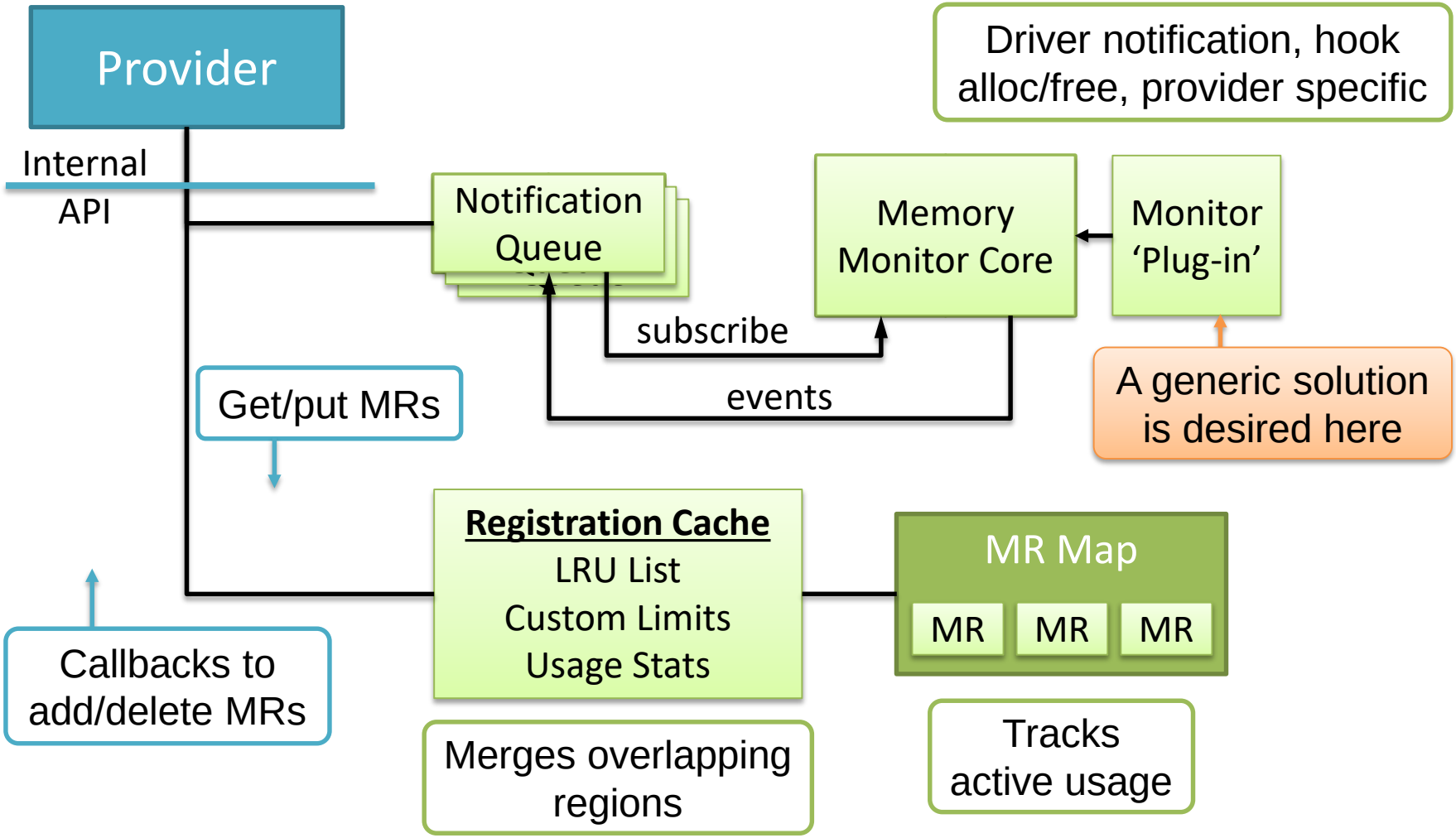
Version
Flags
PID
Region Size
Lock
Command Queue
Response Queue
Inject Buffers
Peer Address Map

Single command queue

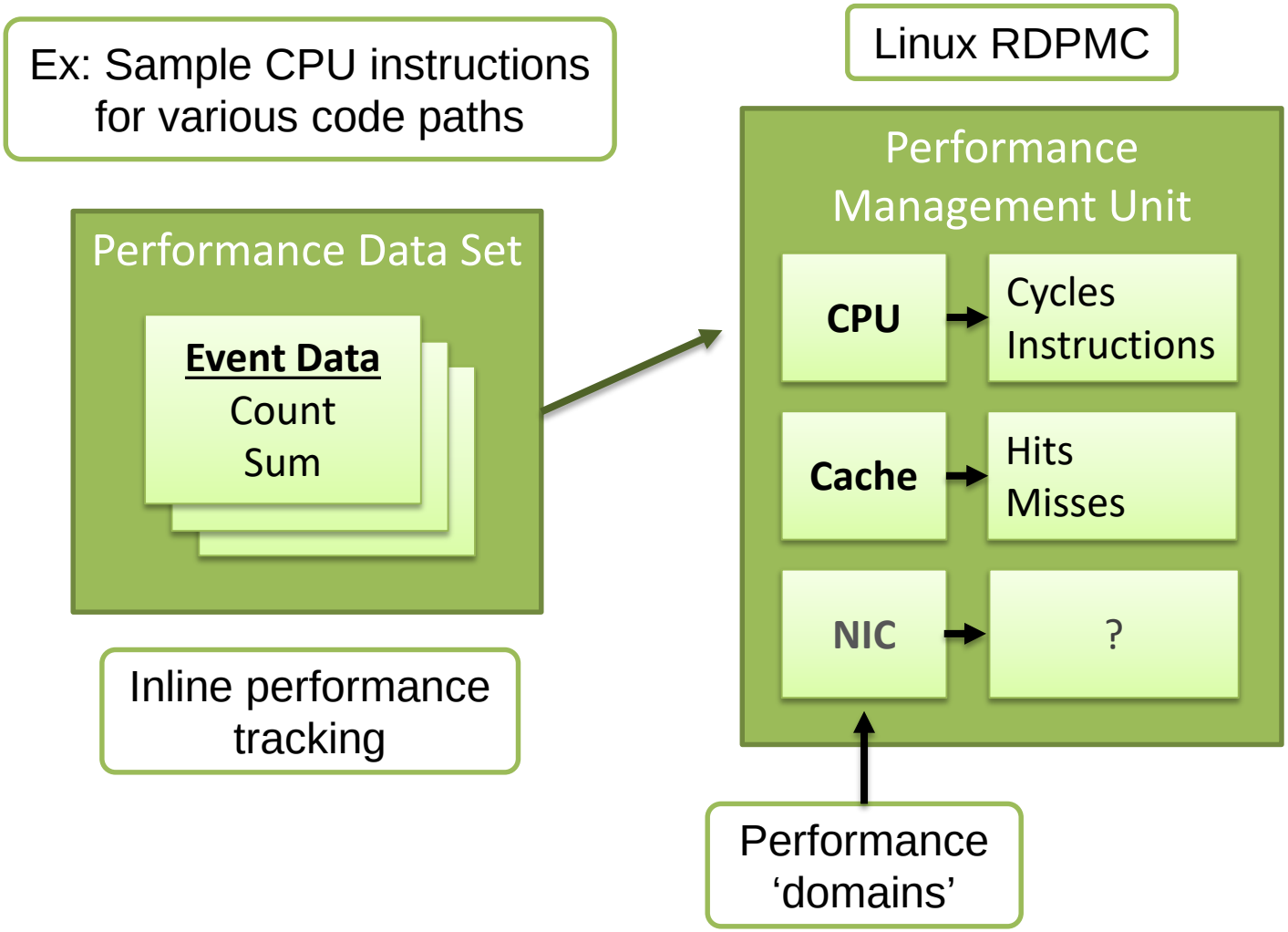
CMA (cross-memory attach) for large transfers

xpmem support under development

MEMORY MONITOR AND REGISTRATION CACHE

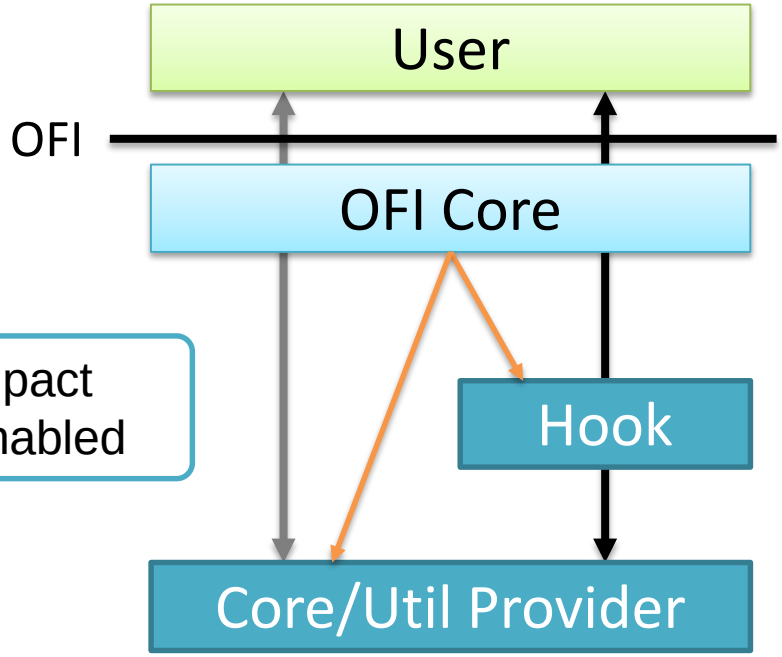


PERFORMANCE MONITORING



HOOKING PROVIDER

Framework done, needs core integration



Always available – release and debug builds

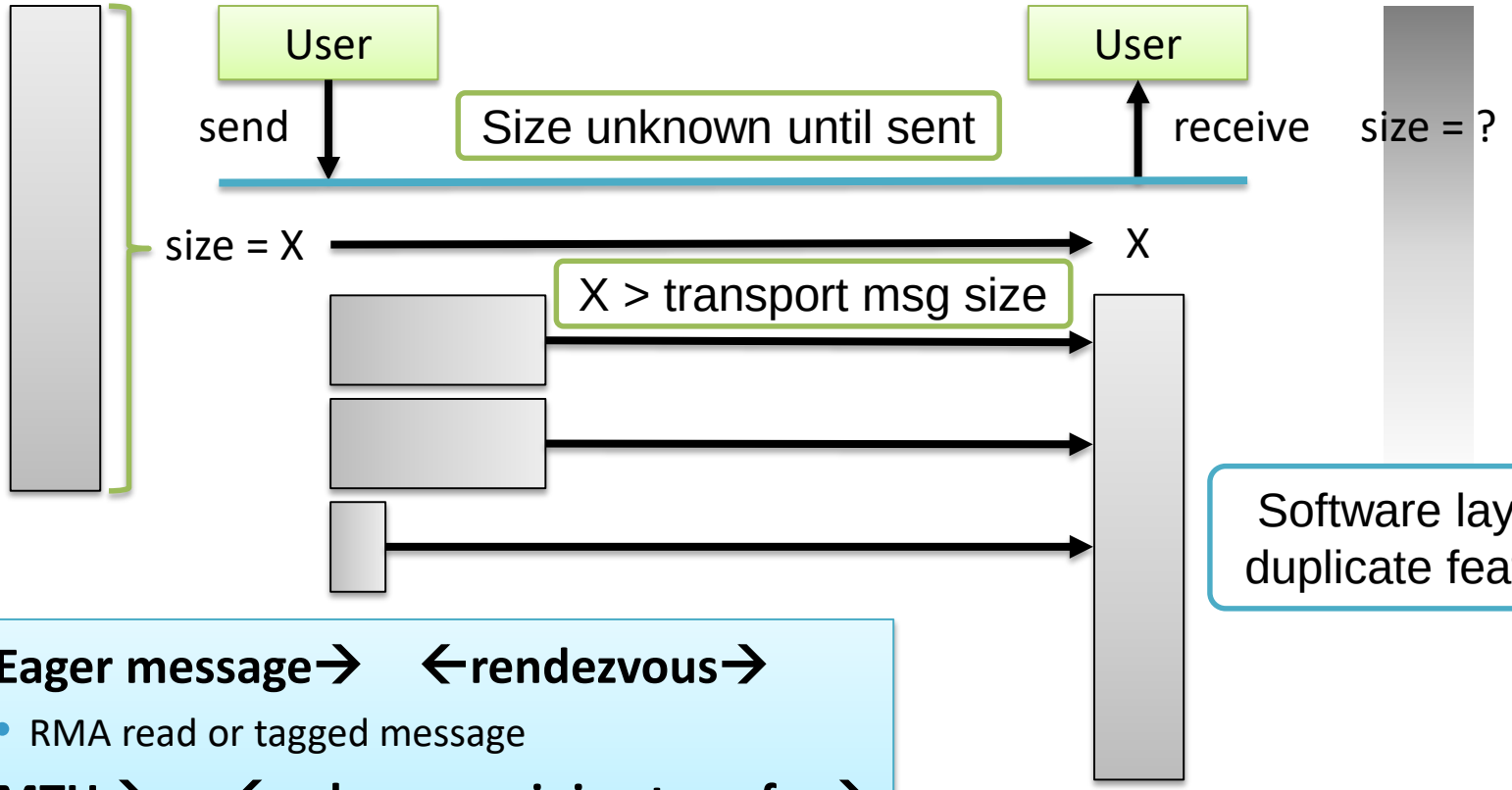
Zero-impact unless enabled

Intercept calls to any provider

Debugging, performance analysis, feature enhancements, testing

API EXPLORATION

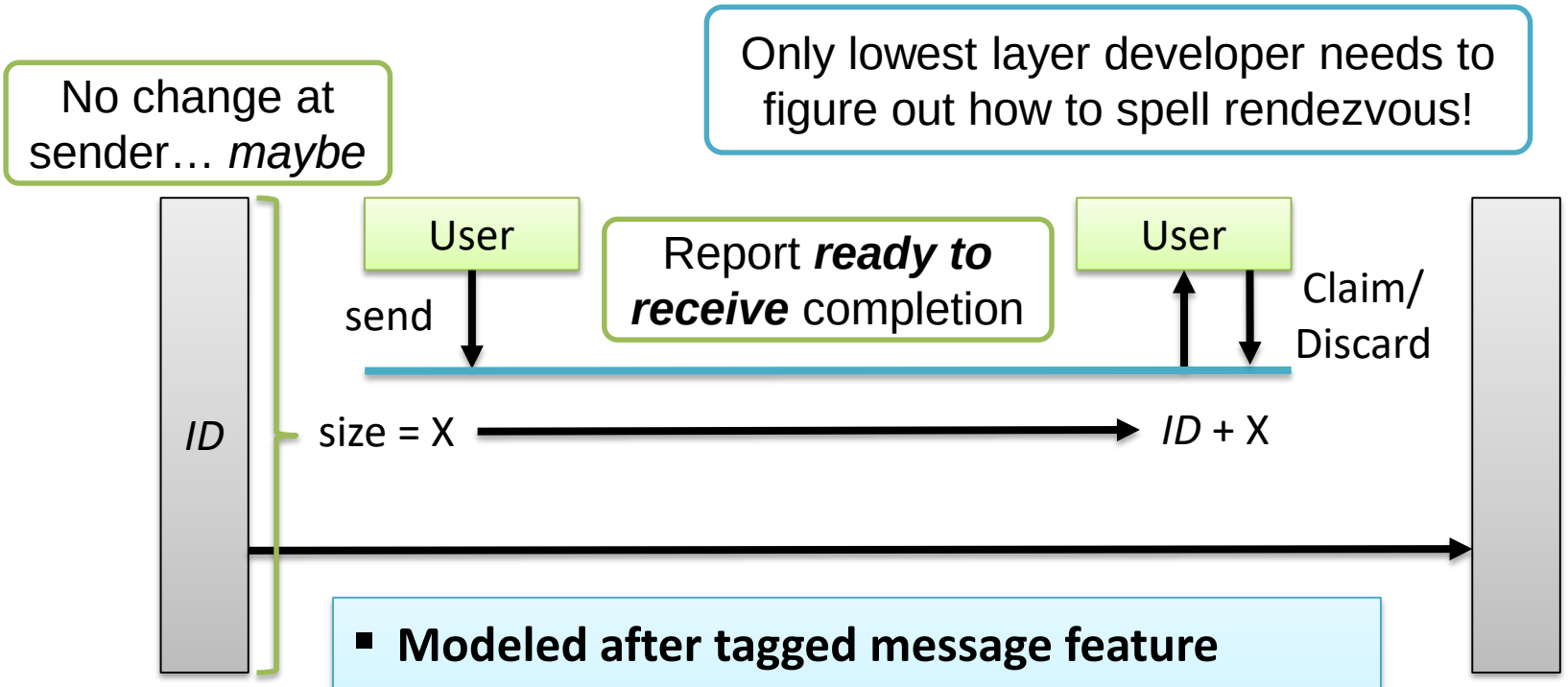
VARIABLE LENGTH MESSAGES



- **Eager message→ ←rendezvous→**
  - RMA read or tagged message
- **MTU→ ←ack remaining transfer→**
  - RMA write, tagged send, send
- **RTS→ ←CLS transfer→**

Similar wire protocols – different implementations

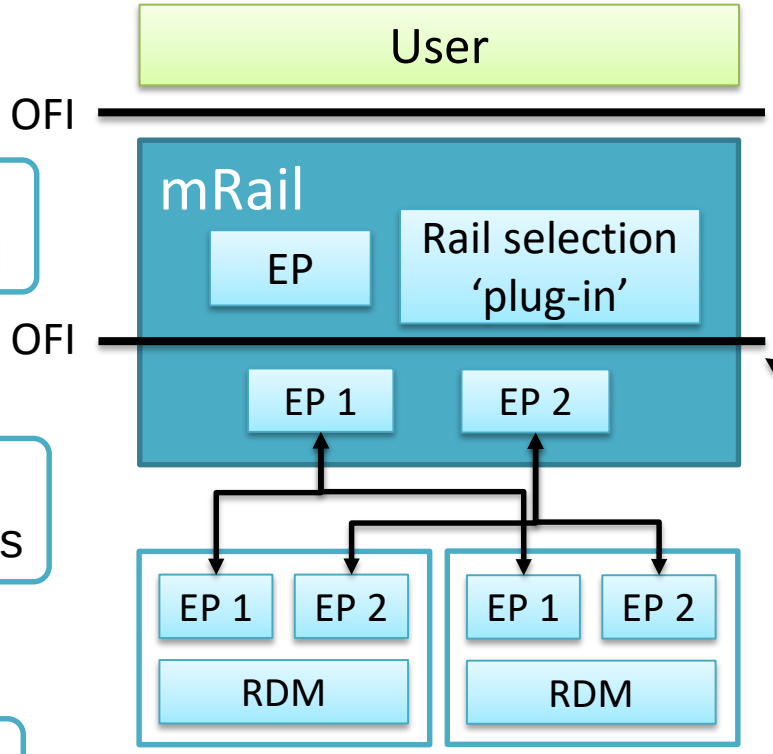
VARIABLE LENGTH MESSAGES



- **Modeled after tagged message feature**
- **Opt-in –impacts protocol**
- **Provider optimizes around hardware abilities**
- **Opportunity: report *discard* to sender**
  - Application flow control and load balancing
  - Dynamically disable receive processing (e.g. EBUSY)

MULTI-RAIL PROVIDER

Focus for v1.7



Application or admin configured

Isolate rail selection algorithm

Require variable message support

Multiple EPs, ports, NICs, fabrics

TBD: recovery fallback

One fi_info structure per rail

Increase bandwidth and message rate

Active

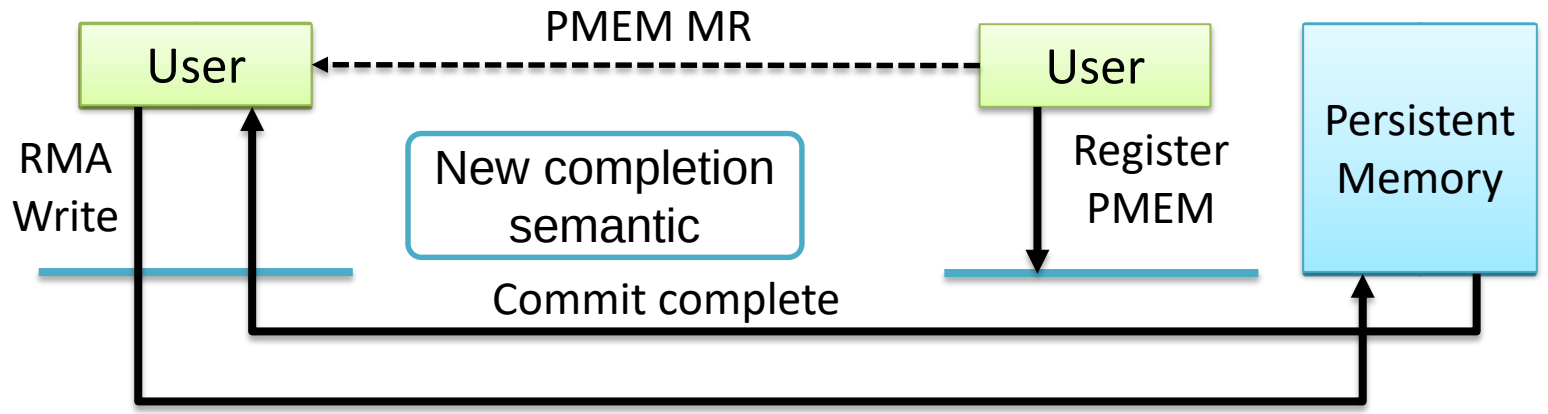
Failover

Standby

PERSISTENT MEMORY

High-availability model (v1.6)

Documentation limits use case



Work with SNIA (Storage Networking Industry Association)

Evolve APIs to support other usage models

- **Exploration**
  - Byte addressable or object aware
  - Single or multi-transfer commit
  - Advanced operations (e.g. atomics)
- **Keep implementation agnostic**
  - Handle offload and on-load models
  - Support multi-rail
  - Minimize state footprint

DATA DOMAINS

APIs assume memory mapped regions

CPU load/stores

Same coherency domain

CPU

Memory

PMEM

May need to sync results with CPU

May not want to write data through CPU caches

Peer Device

Device Memory

(Smart) NIC

FPGA

Device Memory

Memory regions may not be mapped

Programmable offload capabilities and flow processing

Results may be cached by NIC for long transactions

COMPANION APIS

C++ STANDARDIZATION

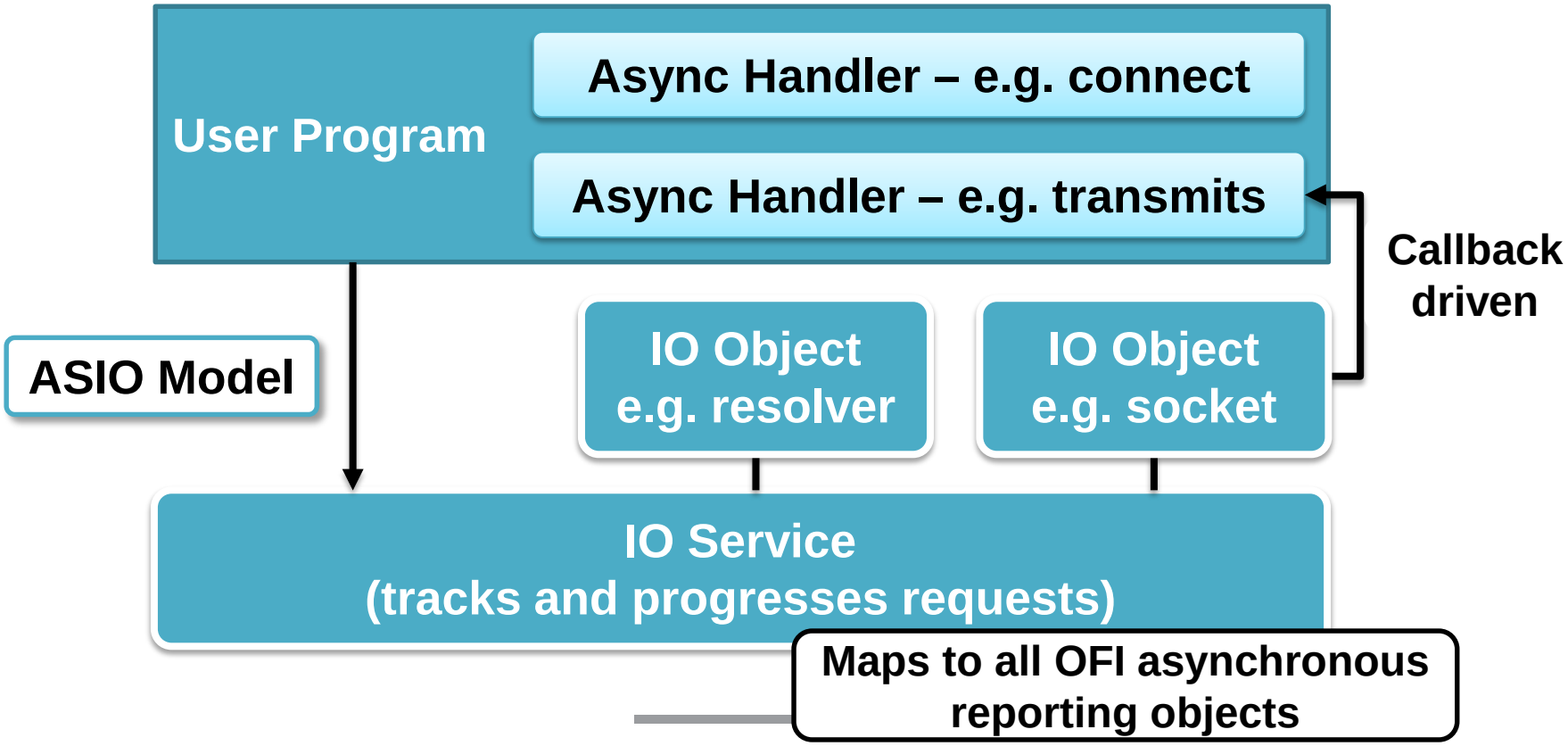
Add support for fabrics directly to the C++ language

Feedback from C++ community

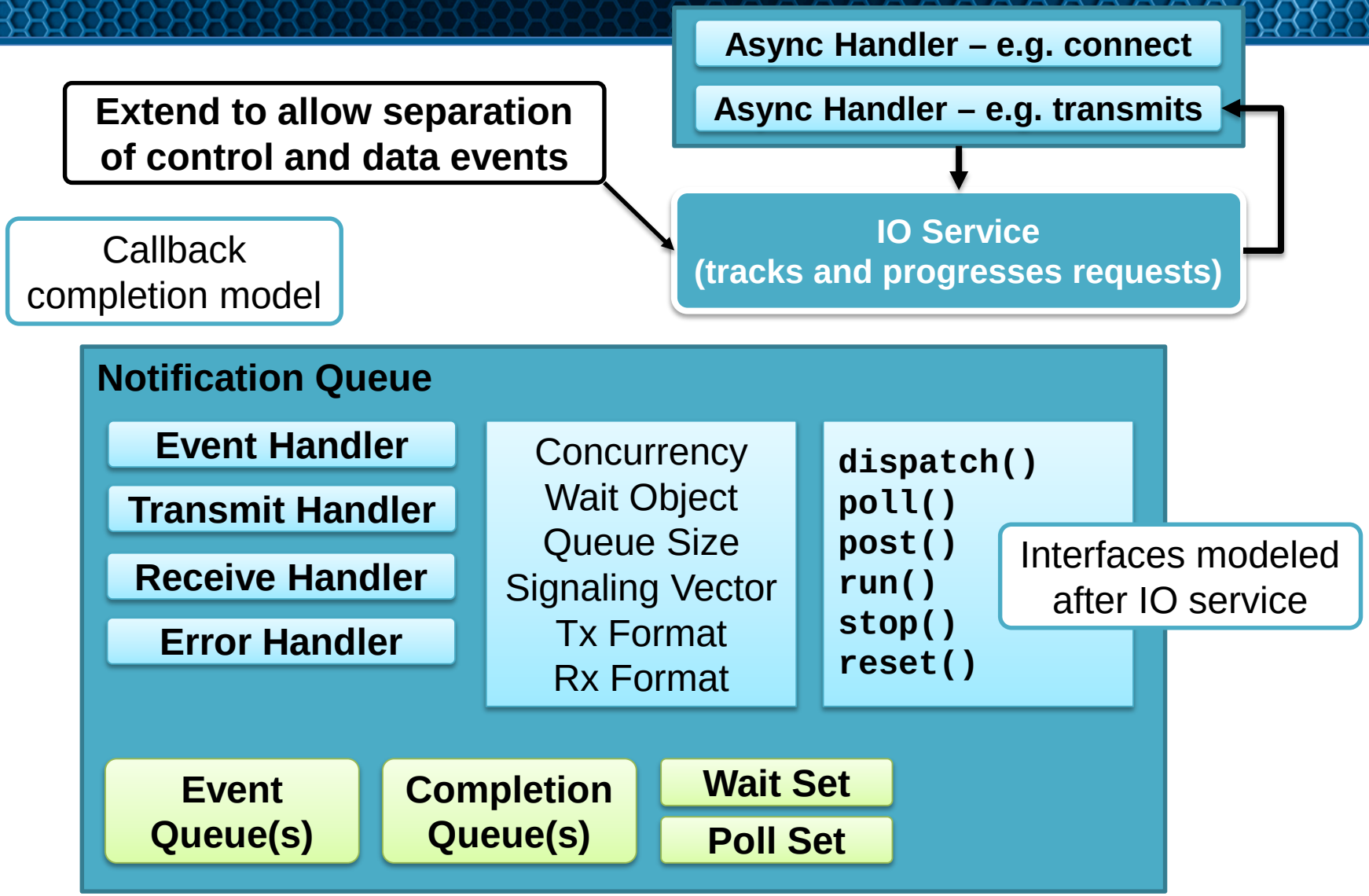
- Implement proposal
- Detail alternatives
- Justify extensions

Proposal

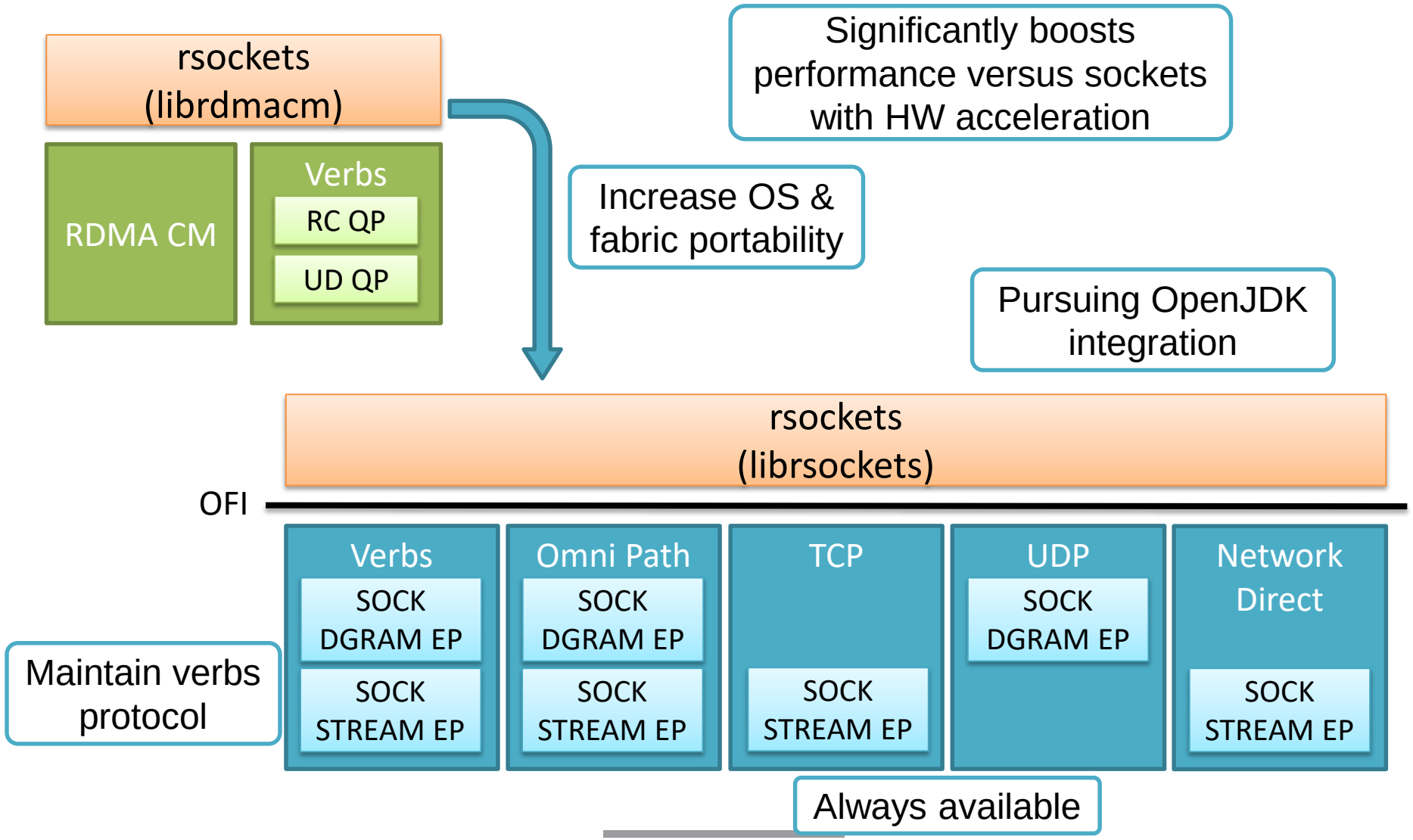
- Extend ASIO
- Implement over libfabric



NOTIFICATION QUEUE



RSOCKETS


rsockets
(librdmacm)
RDMA CM
Verbs
RC QP
UD QP
Significantly boosts performance versus sockets with HW acceleration
Increase OS & fabric portability
Pursuing OpenJDK integration
rsockets
(librsockets)
OFI
Verbs
SOCK DGRAM EP
SOCK STREAM EP
Omni Path
SOCK DGRAM EP
SOCK STREAM EP
TCP
SOCK STREAM EP
UDP
SOCK DGRAM EP
Network Direct
SOCK STREAM EP
Maintain verbs protocol
Always available

14th ANNUAL WORKSHOP 2018

THANK YOU

Sean Hefty, President and CEO

My Own Little World