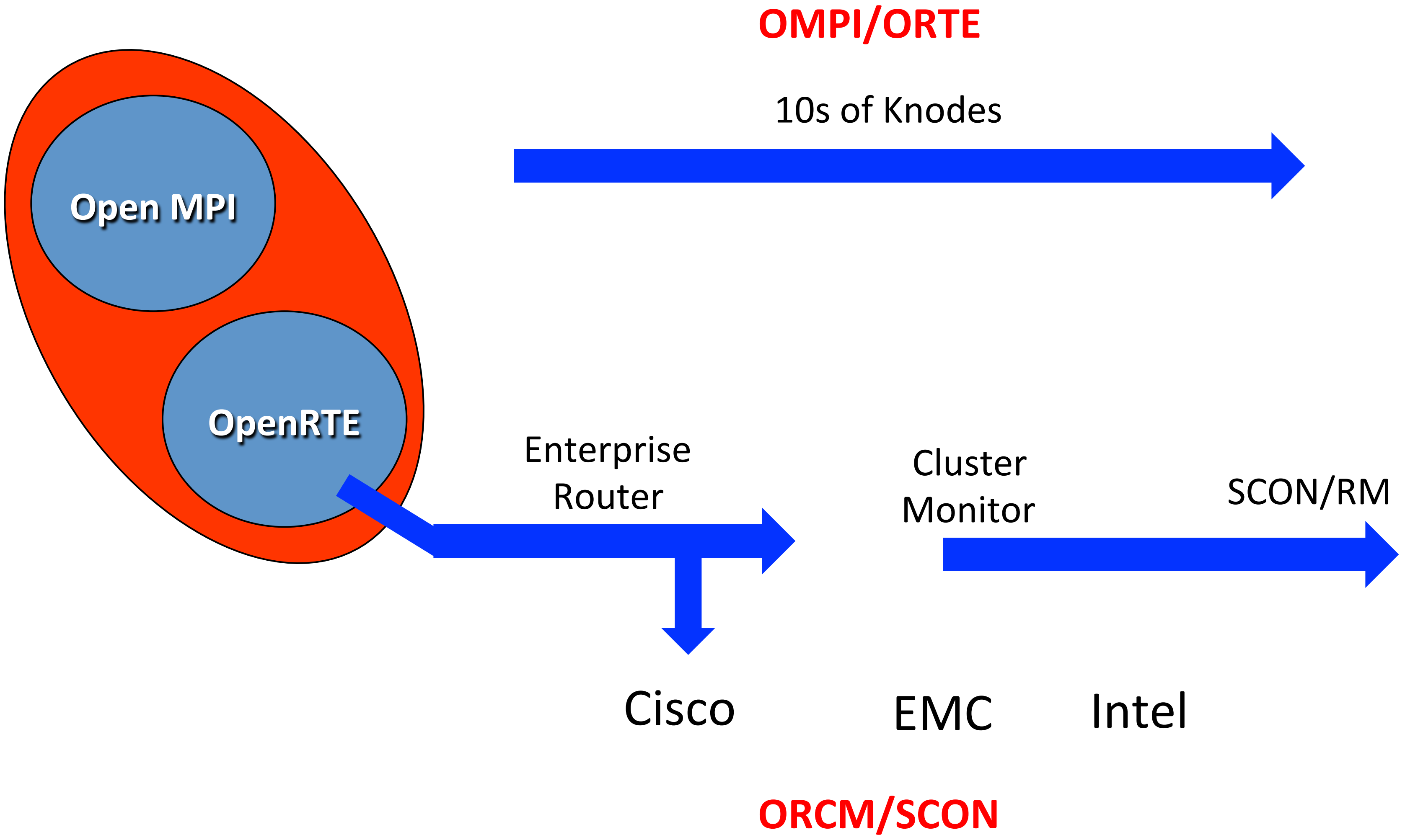
Open Resilient Cluster Manager (ORCM)

Ralph H. Castain, Ph.D.

Objective

- Create a software platform
  - Easily customized, extended
  - Replace/override any behavior
  - Support proprietary as well as open extensions
  - Fully utilize existing installed infrastructure
- Establish ecosystem
  - Academic, industry collaborators, OMPI-like community
- Provide a reference solution
  - Publicly available, performant, flexible, scalable
  - Easily replace any pieces

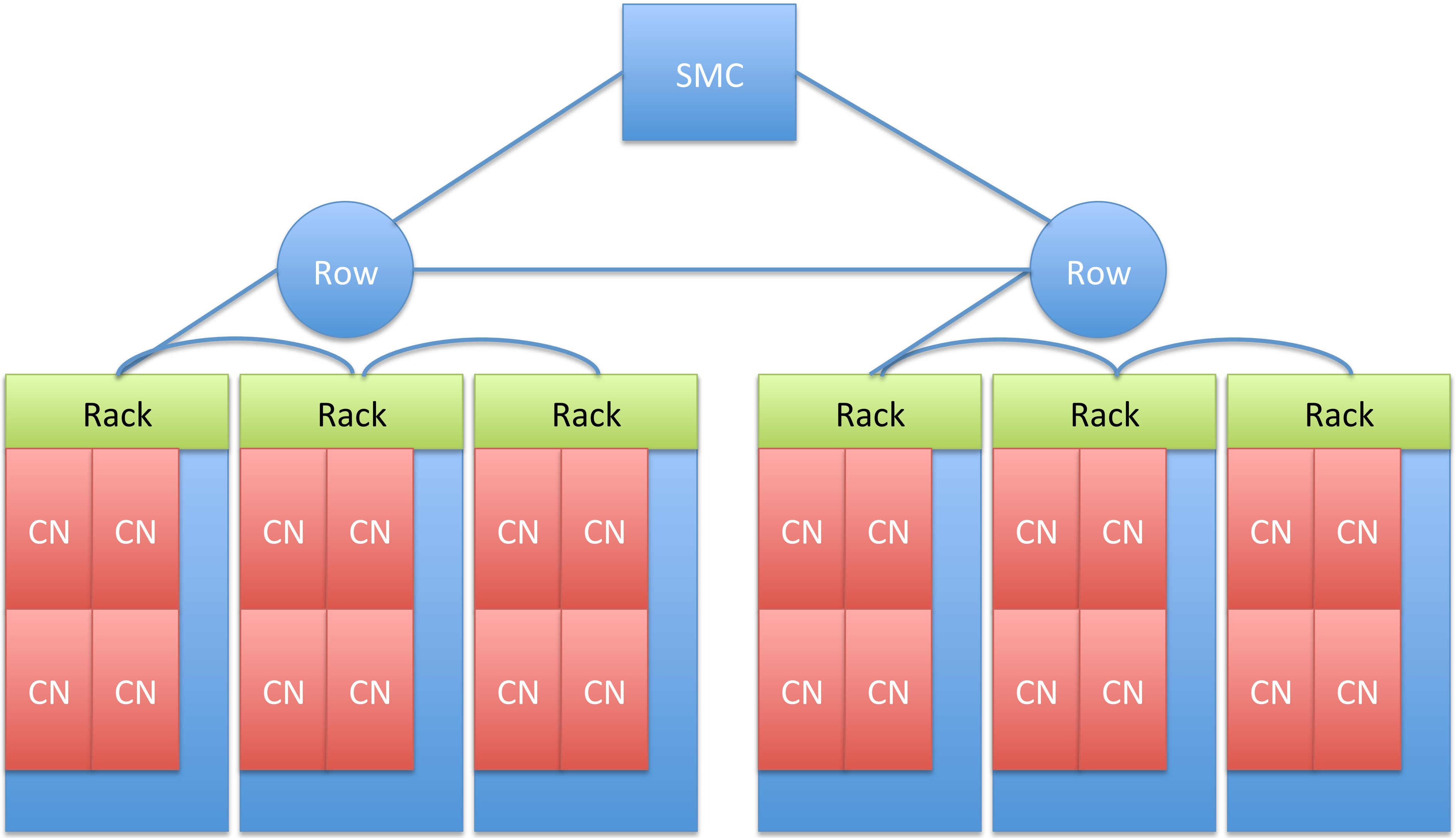
2003 - present



ORCM Roadmap

- Monitoring system (1Q2015)
  - System environment, power, process usage, etc.
- Overlay network/pub-sub (3Q2015)
  - More in a minute
- Job launch (3Q2015)
  - Can do it now, but need support for all MPIs
- Workload manager (2016)
  - Lightweight

Hierarchical Arch



Integration (examples)

- IO subsystem
  - Pre-stage executables and data
  - On-the-fly application-directed data positioning
  - Manage inter-node memory allocations, persistent NVM
- Network
  - QoS controls
  - Static endpoint support
- Power
  - Various modes, dynamic controls, site-level control

Instant On Steps

- Prestage executables to IO nodes
- Allocate and launch
  - Launch message => orte_job_t, included in allocation cmd
  - User-level step daemons spawned and initiate local launch
- Distributed mapping/rolling start (branch)
  - Each daemon computes map, stores all data (map, endpoints, network topo) in shared memory region for job
  - Connect/accept => pass SM connection
- Eliminate modex
  - Static endpoints provided by RM
  - (non-blocking) Lookup on first message
- Eliminate fence at end of mpi_init
  - Modex-recv becomes flag that proc is ready
  - RM flags all procs on node upon first request so subsequent checks are local

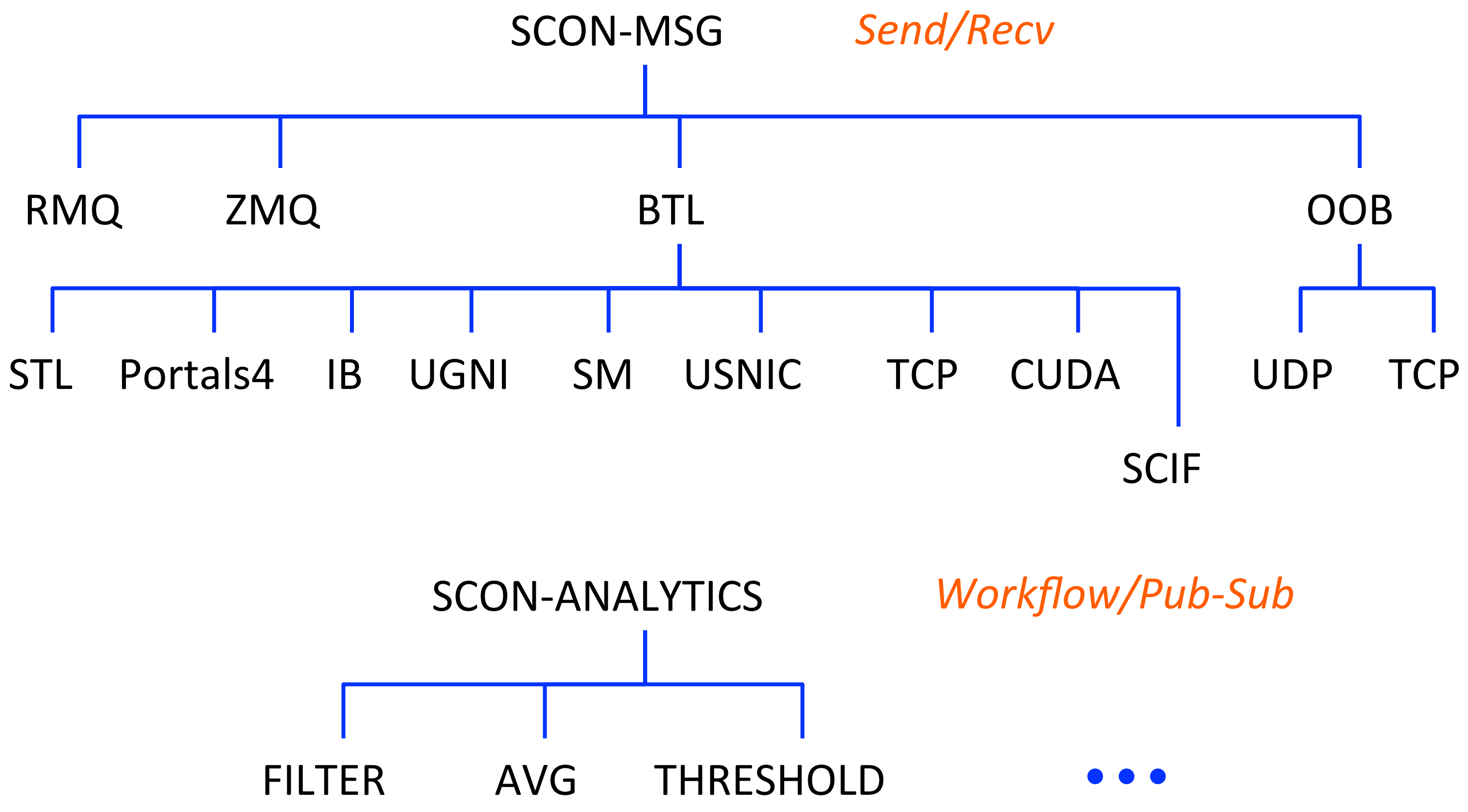
Definition: Overlay Network

- Messaging system
  - Scalable/resilient communications
  - Integration-friendly with user applications, system management software
- In-flight analytics
  - Insert analytic workflows anywhere in the data stream
  - Tap data stream at any point

Requirements

- Scalable to exascale levels
  - Better-than-linear scaling of broadcast
- Resilient
  - Self-heal around failures
  - Reintegrate recovered resources
- Dynamically configurable
  - Sense and adapt, user-directable
  - On-the-fly updates
- Open source (non-GPL)

High-Level Architecture



Messaging APIs

- Typical send/recv
  - Non-blocking, iovec or buffered (built-in heterogeneous support)
- Open channel
  - Specify remote peer and endpoint tag
  - Provide hints on type of data messaging to be used
    - Stream, command/control, etc.
  - Specify desired quality of service
    - Guaranteed delivery of every message, high priority, etc
- Subscribe to data stream
  - Specify source and data

Message Routing

- Detect/select
  - Various algorithms (trees, meshes)
  - Topological
- Defined per transport (branch)
- Heals routes
  - Provides alternate route upon failure
  - Up-level error if no alternate available on this transport
  - Allows re-routing over alternate transports (define prioritized policy)

Message Reliability

- Plugin architecture
  - Selected per transport, requested quality of service
- ACK-based (cmd/ctrl)
  - Ack each message, or window of messages, based on QoS
  - Resend or return error – QoS specified policy and number of retries before giving up
- NACK-based (streaming)
  - Nack if message sequence number is out of order indicating lost message(s)
  - Request resend or return error, based on QoS
- Multipath
  - Send each message across multiple paths
  - Recipient takes first received, discards rest

Analytics

- Event-based state machine
  - Each workflow in own thread, own instance of each plugin
  - Branch and merge of workflow
  - Tap stream between workflow steps
  - Tap data streams (sensors, others)
- Event generation
  - Generate events/alarms
  - Specify data to be included (window)