A Scalable Tools Communication Infrastructure

LEADERSHIP
COMPUTING FACILITY
NATIONAL CENTER FOR COMPUTATIONAL SCIENCES

presented by

Richard L. Graham

Oak Ridge National Laboratory
U.S. Department of Energy

Motivation

- Not many tools exist for HPC application developers
  - Standalone
  - Domain-, application-, problem- and/or site-specific
  - Not scalable
  - Not interoperable with other tools
- Tool infrastructure is reinvented each time
  - Process launch
  - Process management
  - Communication
- Upcoming ultrascale systems have greater demands
  - Scalability
  - Robustness
- Common, portable infrastructure services will be essential to enable
  - More extensive tool capabilities
  - New types of analysis tools

Scalable Tool Communications Infrastructure (STCI)

- STCI collaboration was formed to address tool *infrastructure* needs at the ultrascale
  - System architecture independent API
  - Implementation design guided by ultrascale and multi-tool requirements

- Current Active Collaborators
  - George Bosilca (MPI)
  - Darius Buntinas (MPI)
  - Rich Graham (MPI)
  - Geoffroy Vallee (Sysem R&D)
  - Greg Watson (IDE, Debugging)

Scalable Tool Communications Infrastructure (STCI)

- STCI capabilities
  - Multicast/reduction-style network
    - Scalable communication between tool UI and data sources/sinks
  - Aggregate and point-to-point communication
  - Scalable system resource management
  - Tool lifecycle management

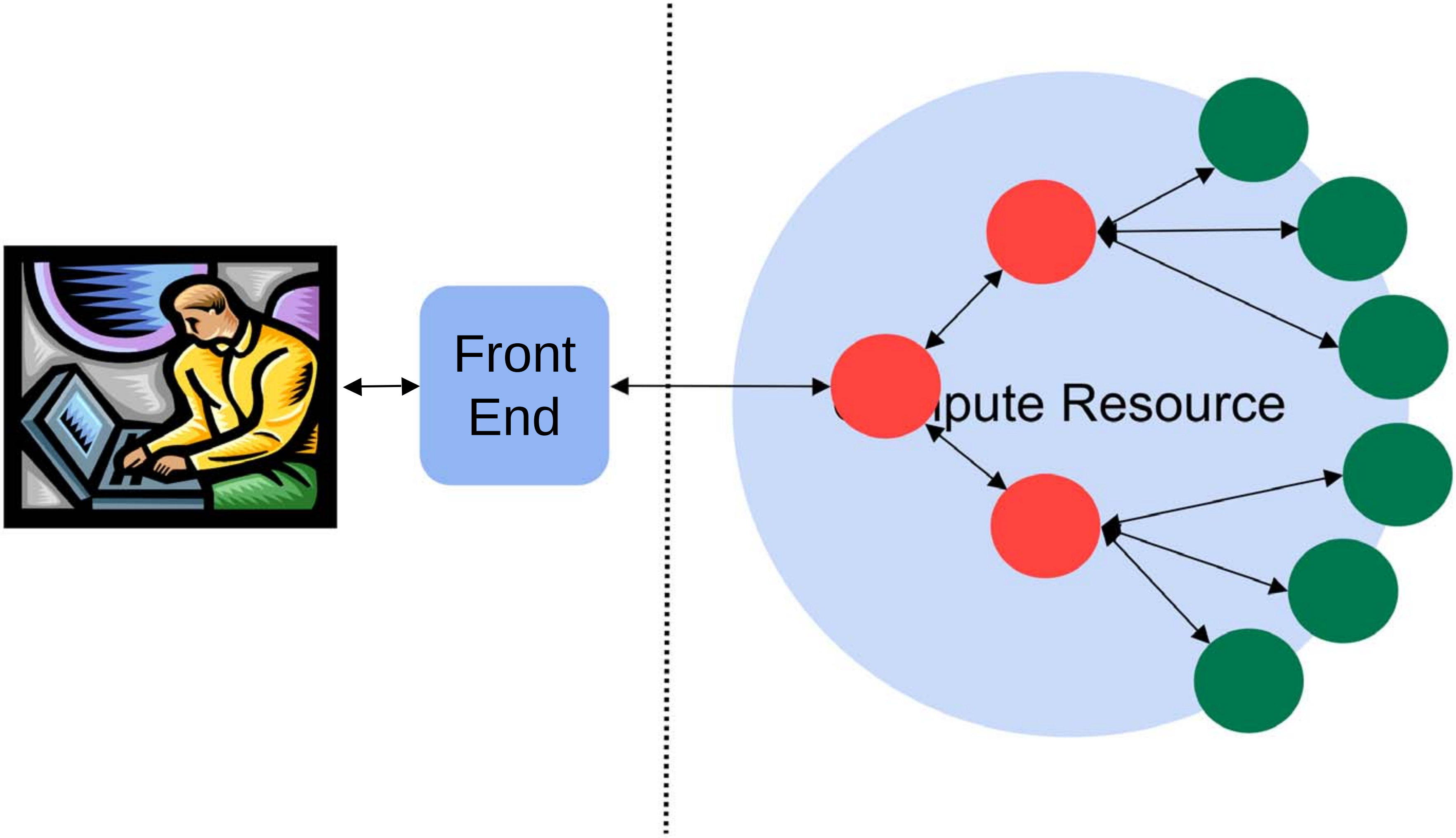
- Tool use cases
  - Interactive tool
  - Instrumented code

Use Cases: Interactive Tool

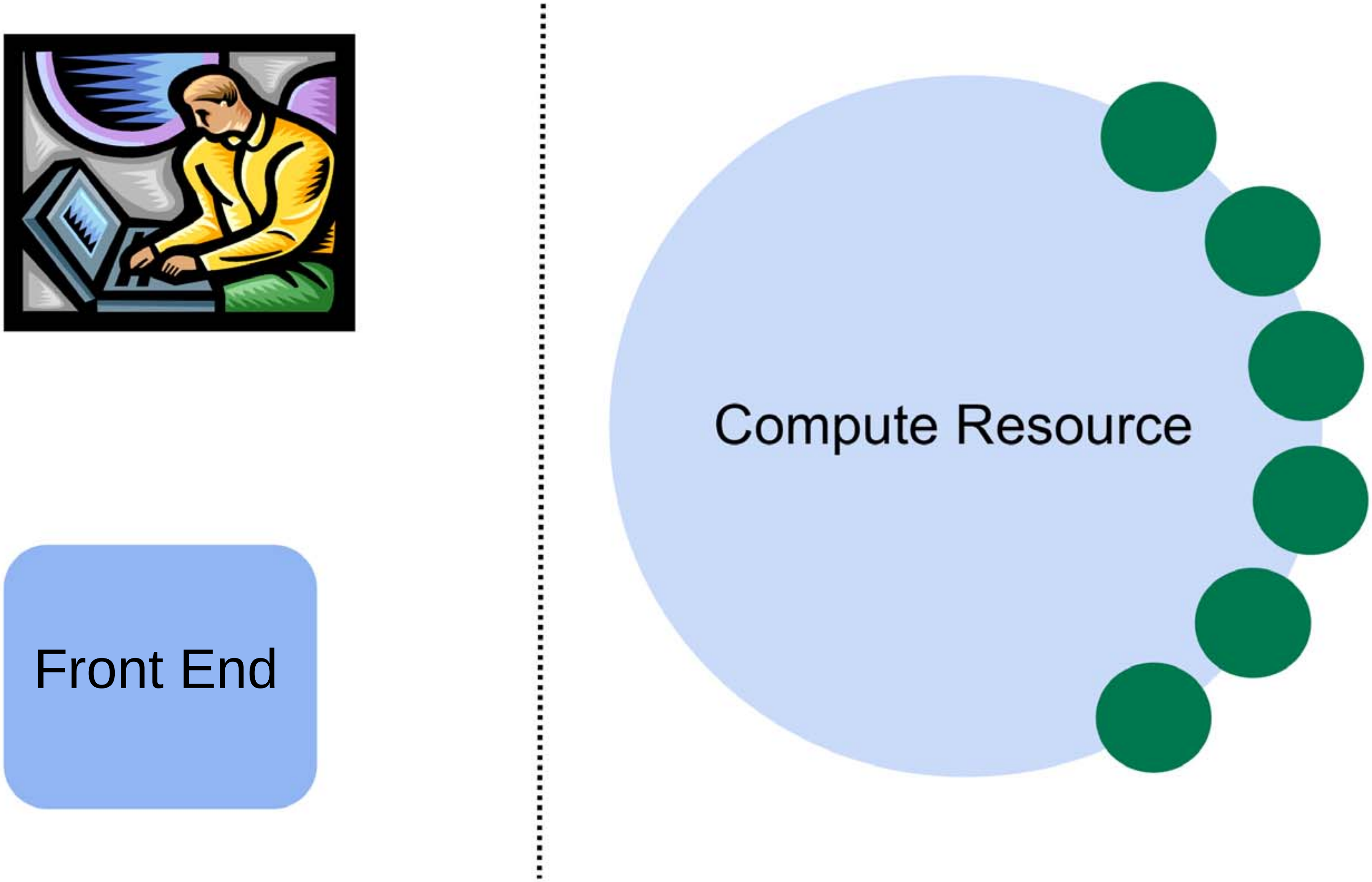
Front End

Compute Resource

Use Cases: Interactive Tool



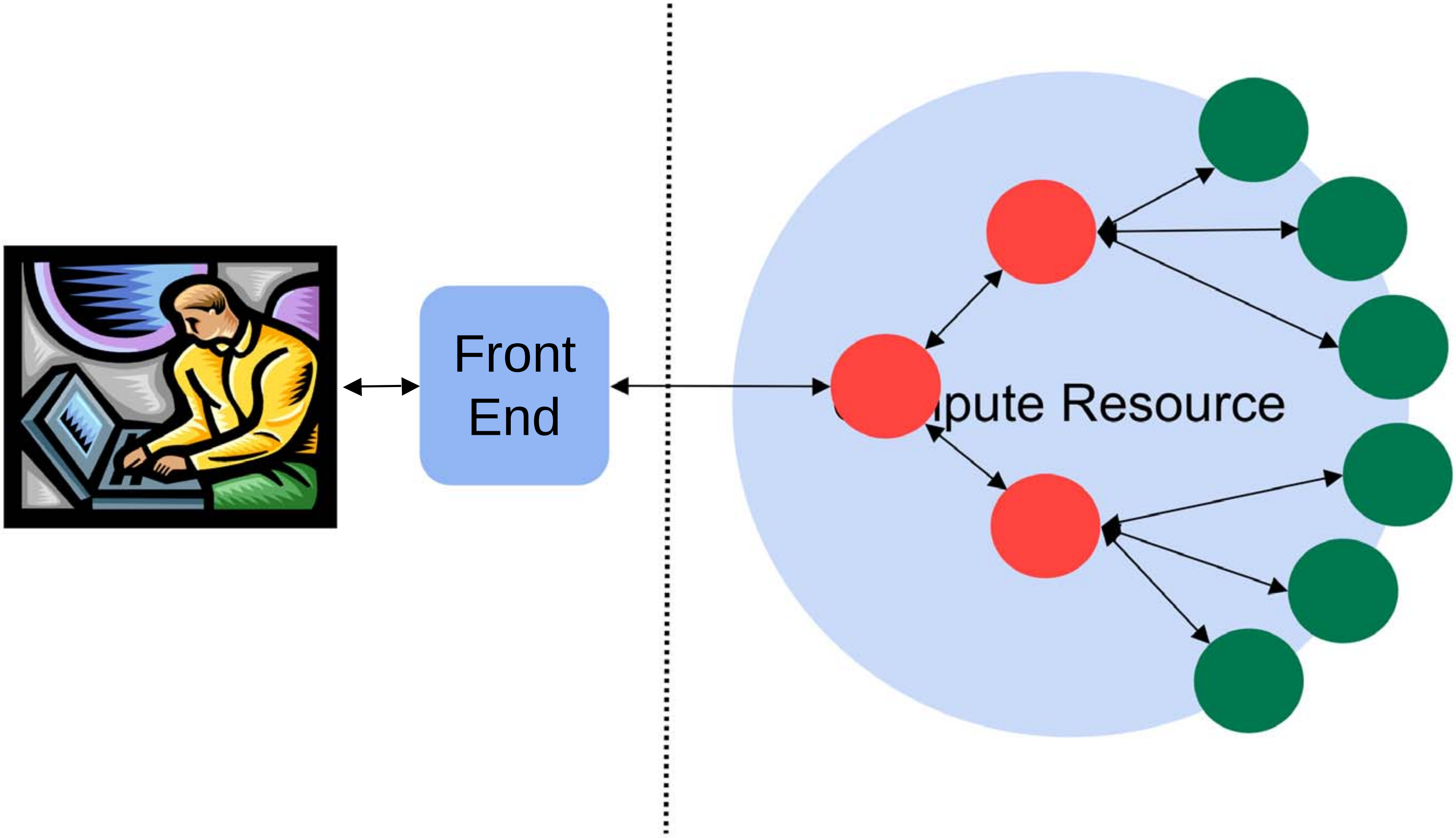
Use Cases: Instrumented Code



Front End

Compute Resource

Use Cases: Instrumented Code

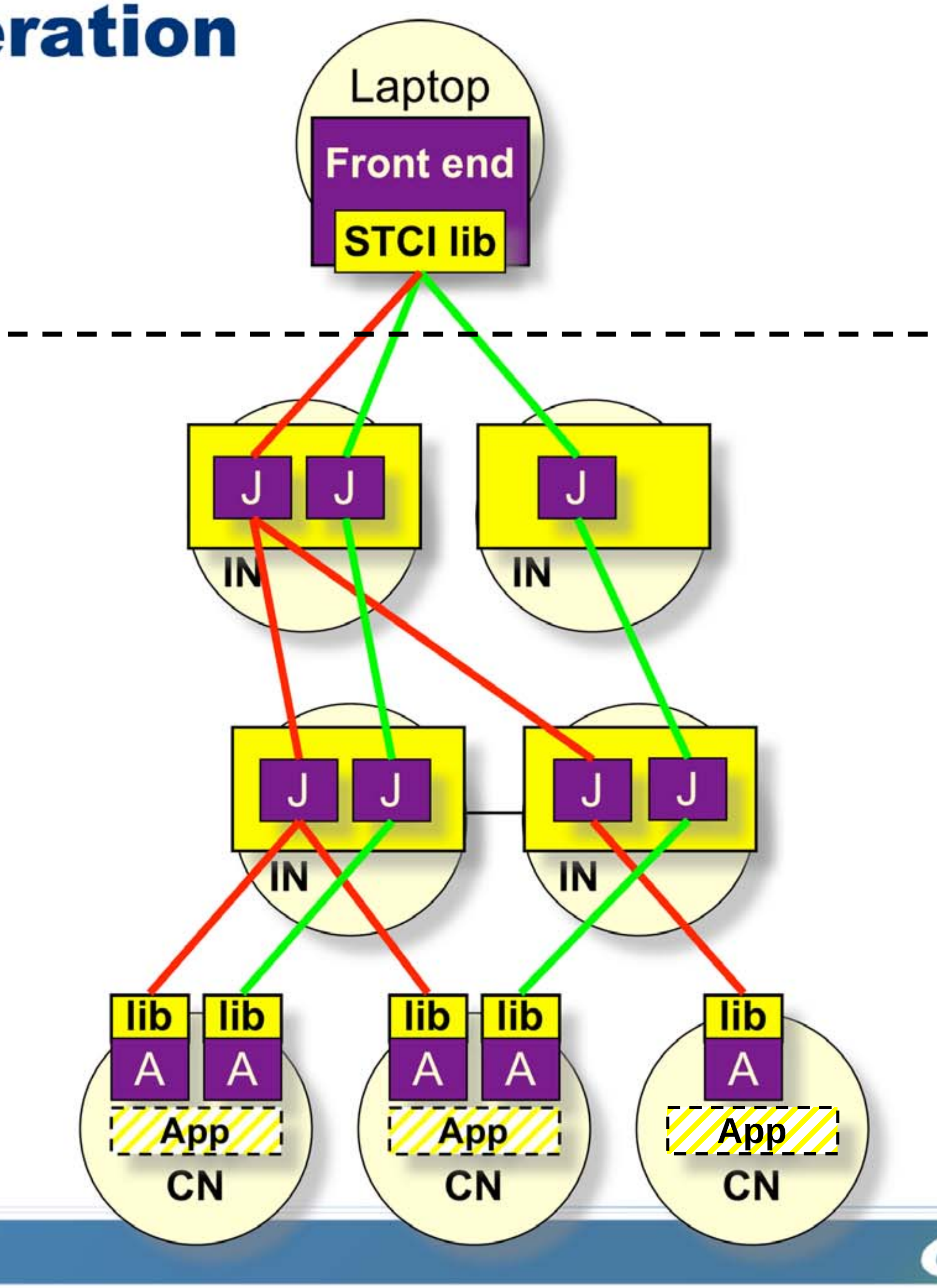

Front End

pute Resource

STCI Tool Model

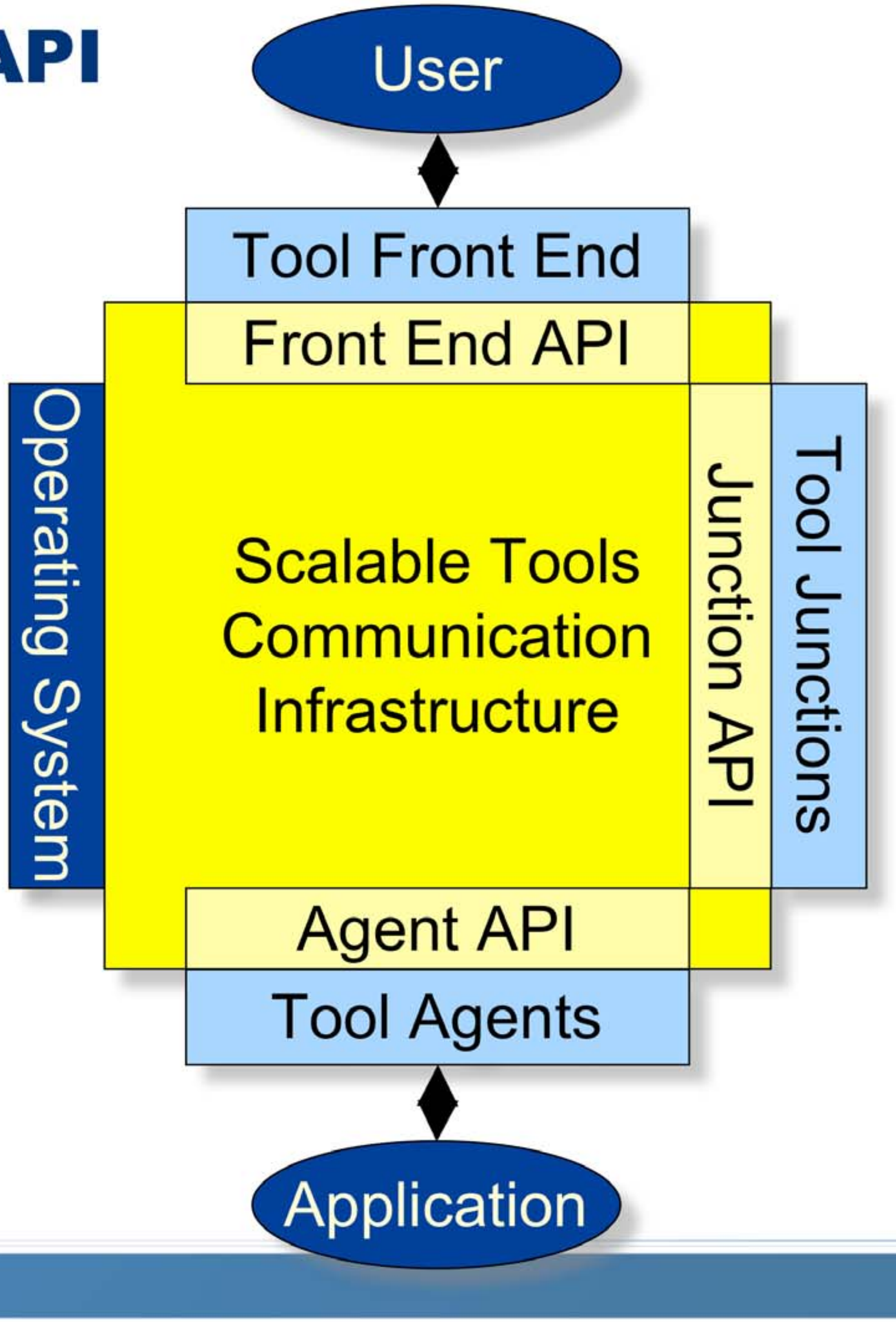
- Monolithic tools are no longer feasible
  - Scalable tools comprise cooperating parts
- Tool model
  - Tool front-end
    - Typically interacts with the user, e.g., GUI
  - Tool agent(s)
    - Interact with application processes, e.g., debugger, profiler
  - Tool junction(s)
    - Aggregate, filter, modify, transform data sent between FE and agents
- Tool developer will implement these parts
- STCI will manage interaction between them

Architecture: Operation



- 🟨 SCTI component
- 🟪 User supplied component
  - A Agent
  - PI Plug-in
- 🟩🟥 Streams
- ○ Physical node
  - IN Infrastructure node
  - CNCompute node

Architecture: API



🟨 STCI Components

🟦 Tool Components

■ External Components

Services Provided by STCI

- STCI provides services related to
  - Execution contexts
  - Sessions
  - Communication
  - Persistence
  - Security

Execution Contexts

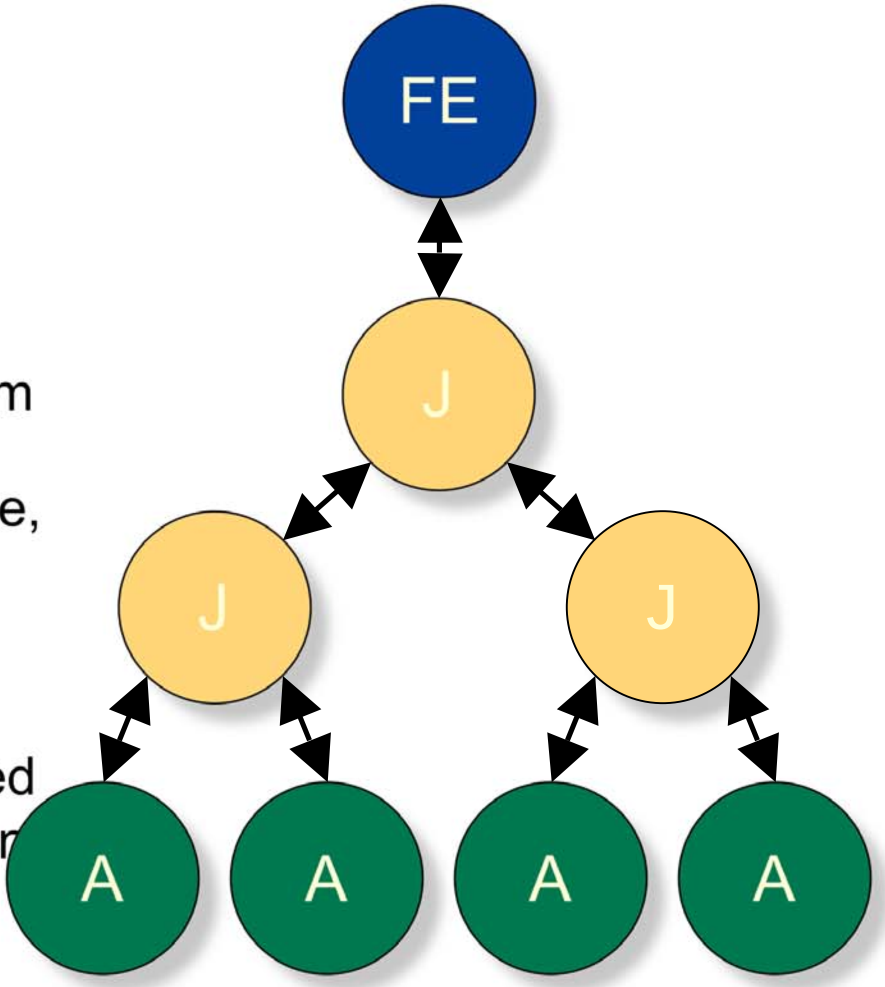
- Bootstrapping
  - Managing infrastructure lifecycle
    - Installation and deployment of STCI
  - Managing tool lifecycle
- Execution context management
  - Starting/killing processes
  - Monitoring
  - Reacting to changes (e.g., process dies)
- Resource management
  - E.g., allocate *locations* (aka nodes)

Sessions

- All tool activities are performed within a *session*
- A session consists of
  - Resource allocation (e.g., CPUs, networks adapters)
  - Set of tool agents and junctions
  - Description of how agents and junctions are mapped onto resources
  - One or more *streams*

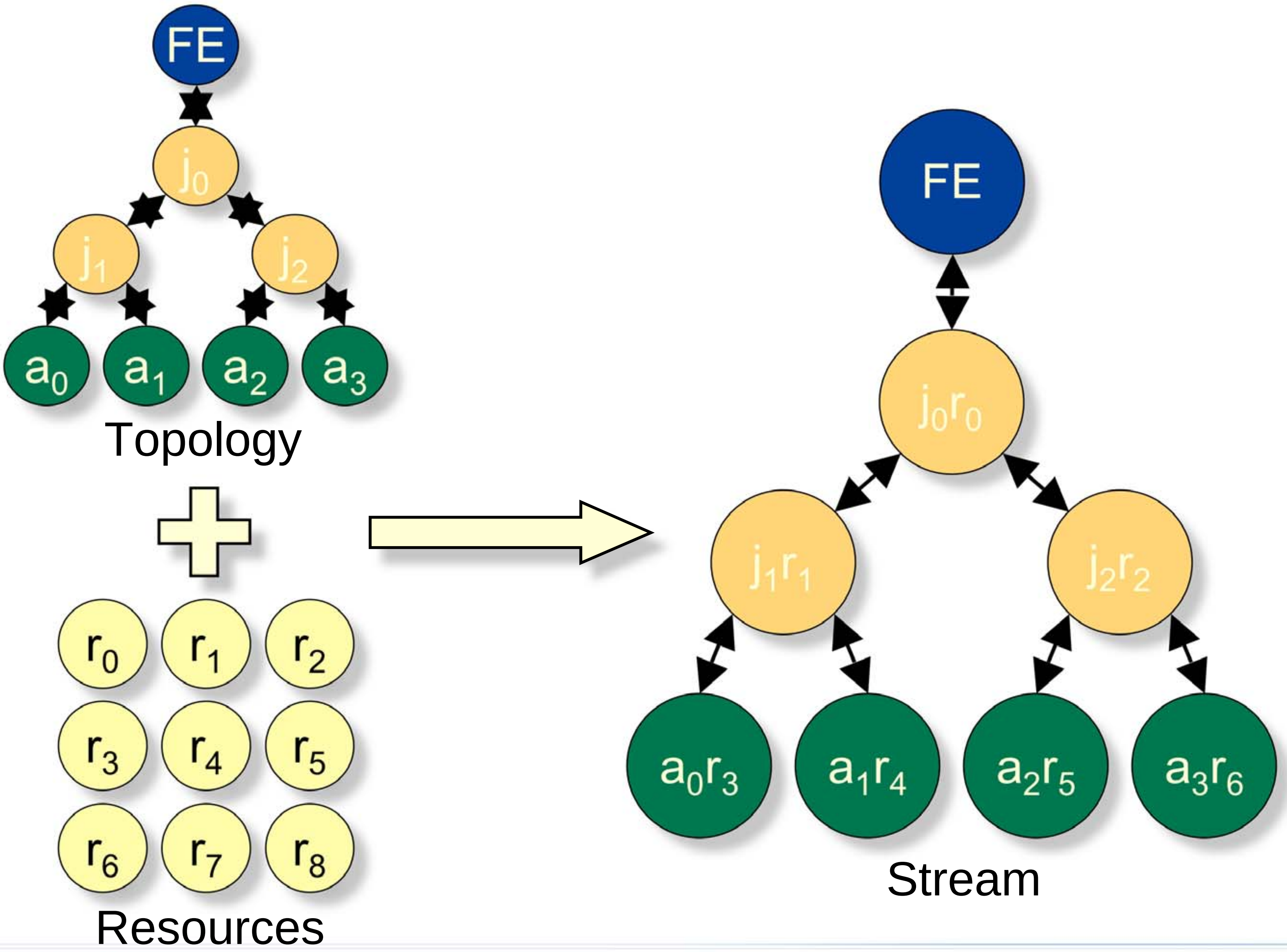
Streams

- A stream connects the FE to one or more Agents
  - Possibly through junctions
- Depending on the junctions, a stream can
  - Broadcast, gather, scatter, reduce, etc.
  - Modify, filter messages
  - Route messages
- Streams can be expanded/contracted
  - Minimize effect on communication
  - Don’t require stop and flush



Streams (cont'ed)

- Formed by mapping topology onto resources
- Topology
  - Predefined e.g., binary tree
  - Tool defined
- Mapping
  - Automatic
  - Tool defined
    - Specific resource
      - e.g., put junction "X" on node "c562"
    - Class
      - e.g., put junction "X" on any "I/O node" and an agent "Y" on any "compute node"


FE
j0
j1
j2
a0
a1
a2
a3
Topology
r0
r1
r2
r3
r4
r5
r6
r7
r8
Resources
FE
j0r0
j1r1
j2r2
a0r3
a1r4
a2r5
a3r6
Stream

Communications

- All communication is performed over a stream
- Active messages
- Stream parameters
  - Message ordering
  - Reliability
- Flow control
  - Pause and buffer
  - Pause and drop
  - Flush or quiesce a stream
- Group communication: Bcast, reduce, etc.
  - Can be implemented by tool using junctions
  - STCI provides built-in group communication streams
- Datatypes
  - Describe data layout and basic datatypes
  - Non-contiguous data
  - Heterogeneous system support

Persistence

- Persistent state is maintained by STCI
  - State of the infrastructure
    - Location of infrastructure components
  - Active sessions
    - Allocated resources
  - Policy & security
- Facilities for front-end disconnect and reconnect
  - Where to reconnect
- Cleanup when sessions exit or abort

Security

- Security services manage and control interaction between entities
  - Users, tools, applications, system resources
  - According to policies of a single security domain
- Services
  - Session authentication
    - Tool provides credentials to create or reconnect to a session
  - Service authorization
    - Tool will not have access to any greater privilege than the user would be allowed
- Keep as simple as possible
  - avoid conflicting with existing security mechanisms

Conclusion

- Developing efficient scalable tools has always been a challenge
  - Exascale systems make this even harder
- Existing tools are often
  - Architecture specific
  - Problem domain specific
  - Application specific
- Tools often have to re-invent the wheel
- STCI provides a standard HPC tool infrastructure
  - Scalability
  - Efficiency
  - Portability
  - Interoperability

For More Information

- STCI website
  - http://www.scalable-tools.org

- Email me
  - rlgraham@ornl.gov