Open Resource Control Architecture
Orca-BEN Cluster D

Jeff Chase

GEC-3 Control Plane Status 10/28/08

Duke Systems

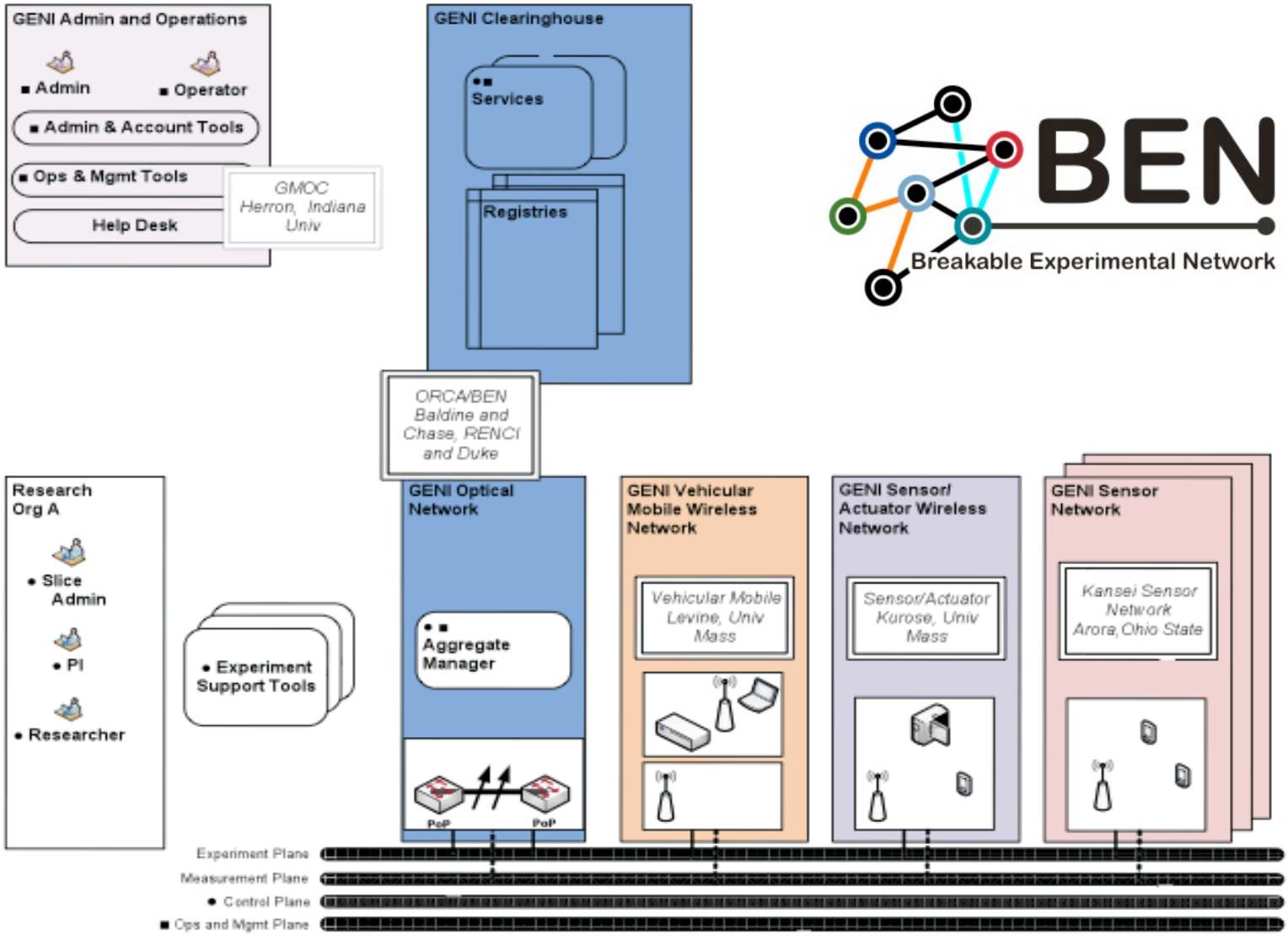


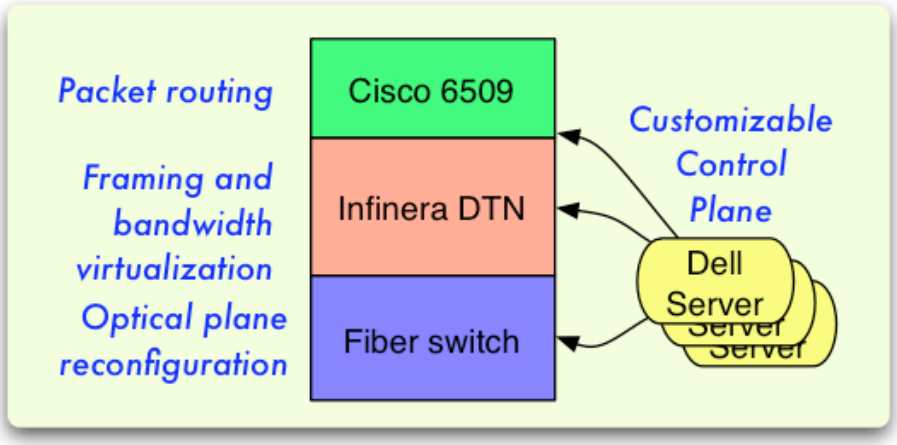
Figure 5-4. Cluster D utilizing ORCA control framework

Breakable Experimental Network (BEN)

- Research/Triangle optical network
- RENCI PoPs, e.g., campus RECs
  - Duke, NCSU, UNC, MCNC
  - Cisco/MPLS bridge to campus/NLR
- Sliverable at multiple levels
  - From the fiber up (BYOT)
  - Lambda WDM + TDM
  - Infinera Bandwidth Virtualization
  - VLANs and flows

Renci
Renaissance Computing Institute

infinera



Cluster-D Substrate

- "Cloud computing" ensembles at the edge
  - Cluster-on-Demand (COD)
  - Eucalyptus (UCSB)
  - NCSU VCL

- Cross-layer network slivering
  - BEN: nested allocation at optical transport and below
  - Configurable IP overlays (e.g., VRF)
- Sensor fields/testbeds
  - Vise (U. Mass Amherst)
  - Kansei (Ohio State)
- Mobility and intermittent connectivity
  - DieselNet (U. Mass Amherst)

Orca Control Framework

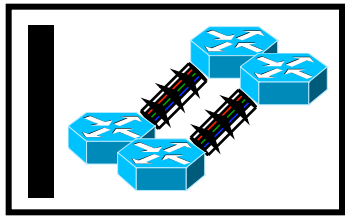
- Open
  - Protocol-centric: actors representing providers (*hosts*) and consumers (*guests*) negotiate lease contracts.
  - Open to innovation: plugin structure for contract exchange, representation, allocation policy, and configuration.
- Resource
  - Allocate/sliver diverse substrate elements
  - “Virtual network resources”
- Control
  - Resource contracts confer specific rights and assurance of isolation; holder controls/programs leased resources.
- Architecture
  - Actor roles, protocols, and interfaces
  - Java-based toolkit implementation: Shirako leasing core

Control Framework: Actors

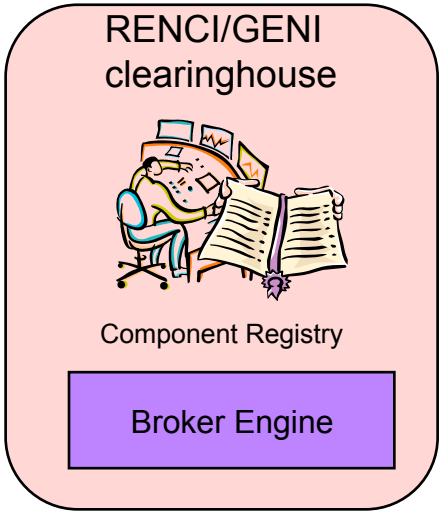
- Principle #1: actors represent primary stakeholders.
  - Provider (host): aggregate manager for an ensemble
  - Consumer (guest): controller acting for researcher
  - Broker represents loose federation of hosts and guests

Edge cloud (site authority)

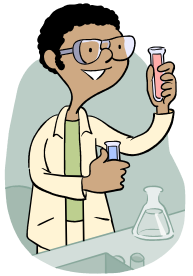
transit provider (domain authority)

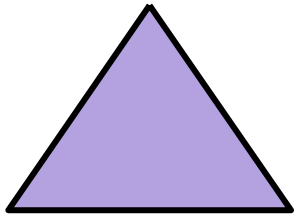


Provider aggregates



Facility clearinghouse

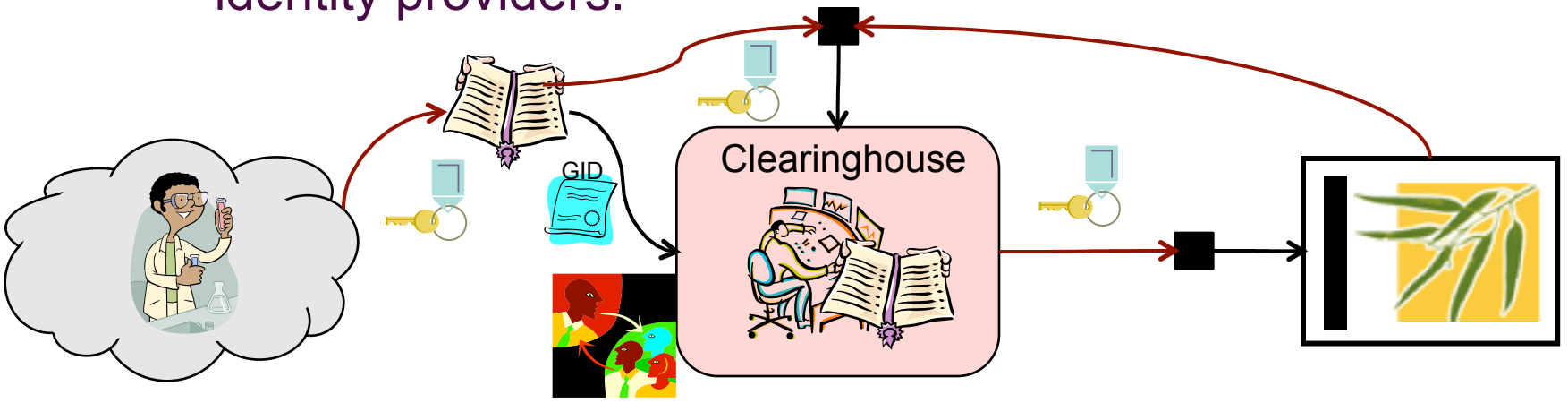




Guest slice controller

Identity and Trust

- Other entities (e.g., institutions) endorse identities and their security attributes to establish trust chains.
  - No direct role in the control plane protocols, although they may wield powers of the entities they endorse.
- Actor operators register public keys of partners.
  - User and aggregate authority (MA) pick clearinghouse.
  - Clearinghouse registers qualified aggregates and user identity providers.



Control Framework Principles

(2) Actors enter into contracts to lease resources.

- Specific assurances for performance and isolation
- Reservations, best effort, etc.
- Specific time intervals
- Flexible representation
- Accountability

(3) Contract model supports delegation (subcontracting).

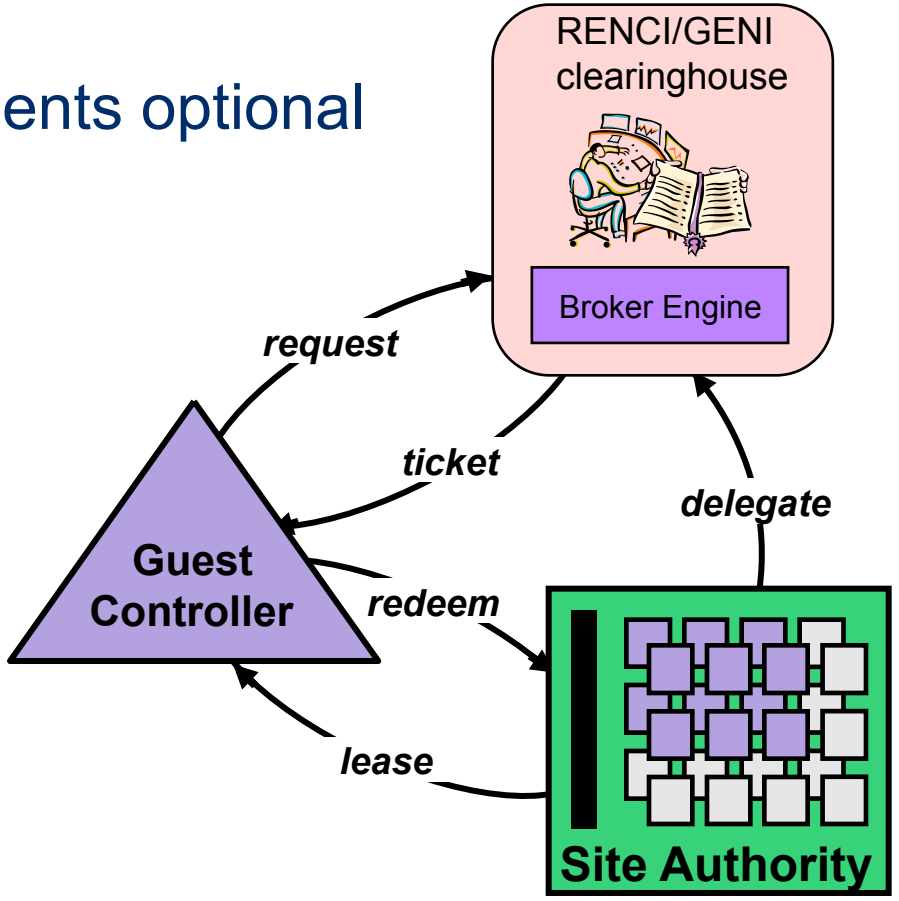
- Providers delegate allocation to clearinghouse
- Ticket broker service with “ticket splitting”

(4) Use aggregation.

- AM is a “wrapper” around existing resource manager technologies for substrate components.

Ticket Broker Service

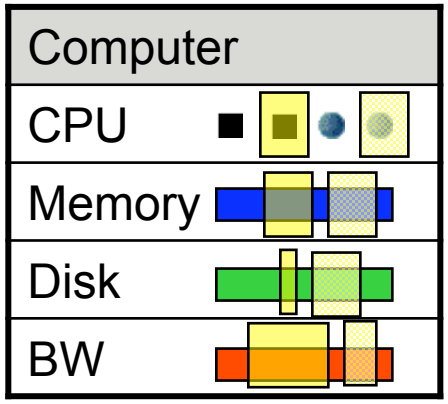
- Orca clearinghouse implements optional GENI *ticket broker service*.
- AMs delegate to broker.
- Broker issues resource tickets to user slices.
- Match request properties.
  - Coordinated allocation
  - Calendar scheduling by user identity or group
- Foundation for discovery
  - resources, paths, topology

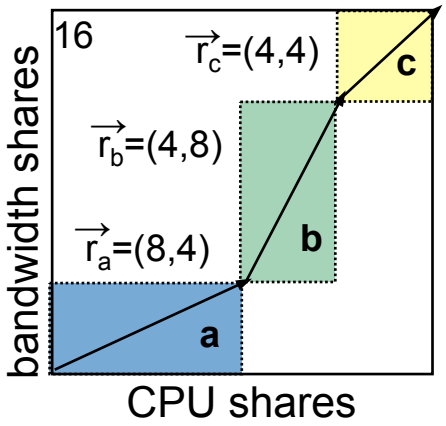


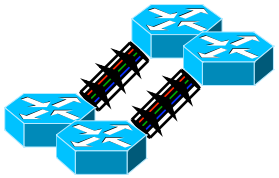
SHARP [SOSP 2003]
w/ Vahdat, Schwab

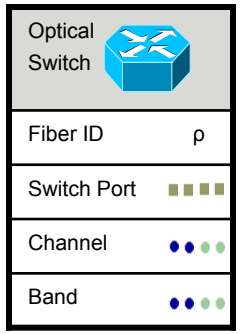
Issue: Ticket Splitting

- Clearinghouse/broker maintains a registry of components, attributes, and relationships.
- Some aggregates are pools of interchangeable instances of a given type.
  - E.g., edge cluster cells, storage
- Broker plugins process representations and issue tickets for subsets that match guest requests.



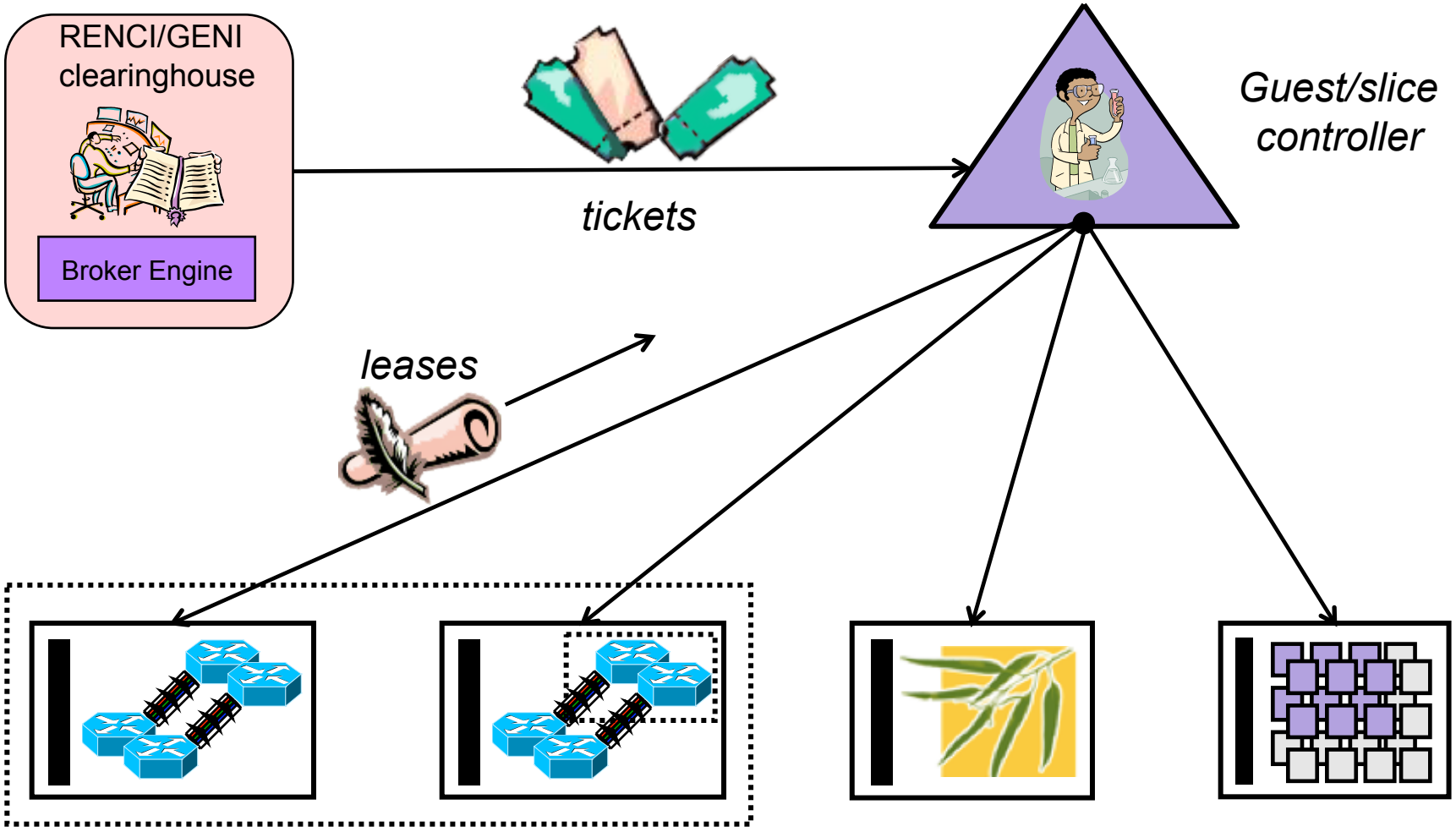






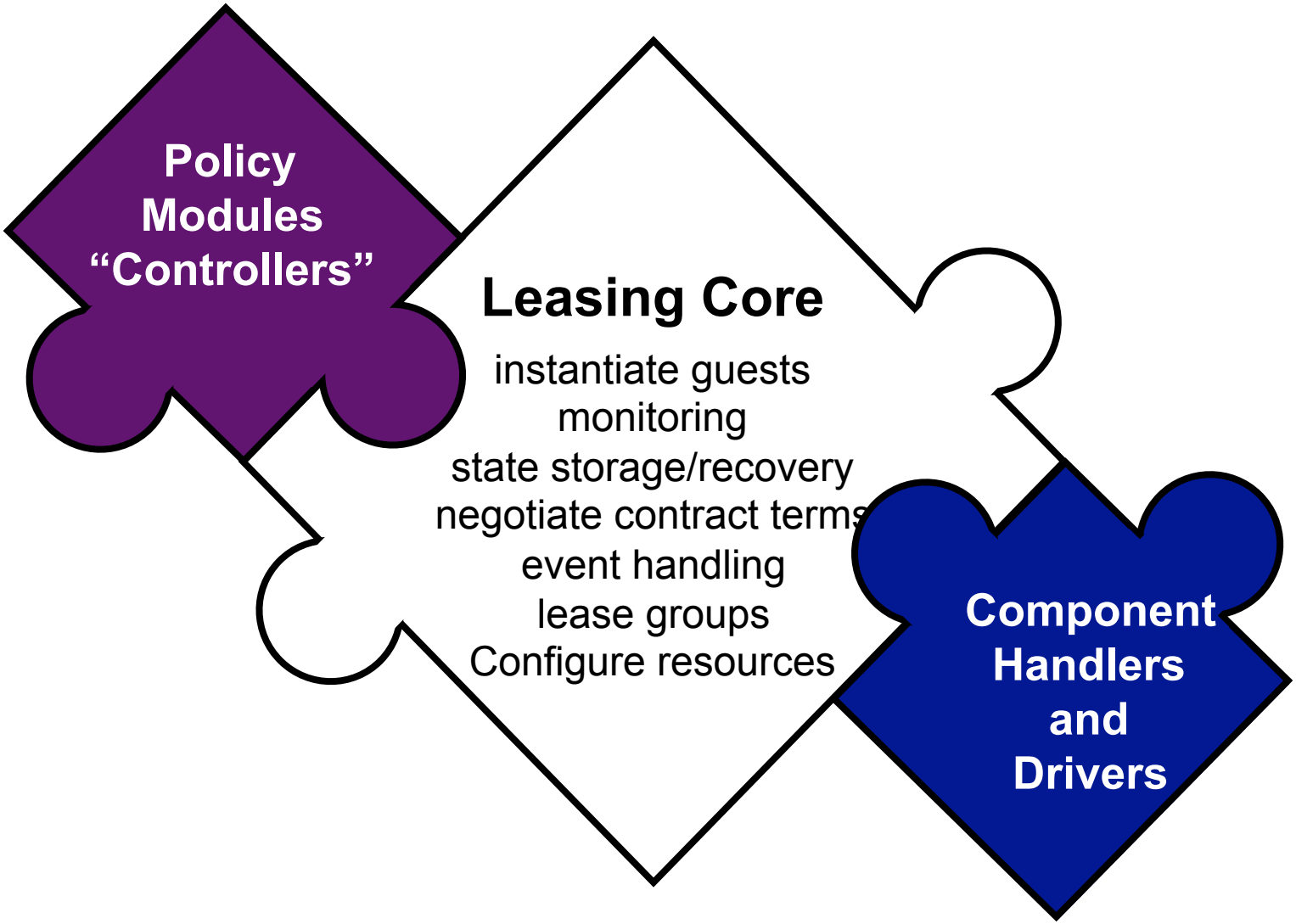
NDL + GMPLS?

Slice Setup



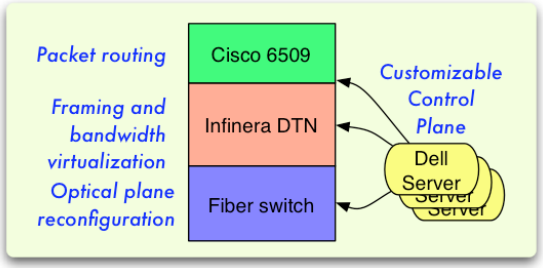
Exchange of labels, tokens, configuration attributes, etc.

Pluggable Resources and Policies



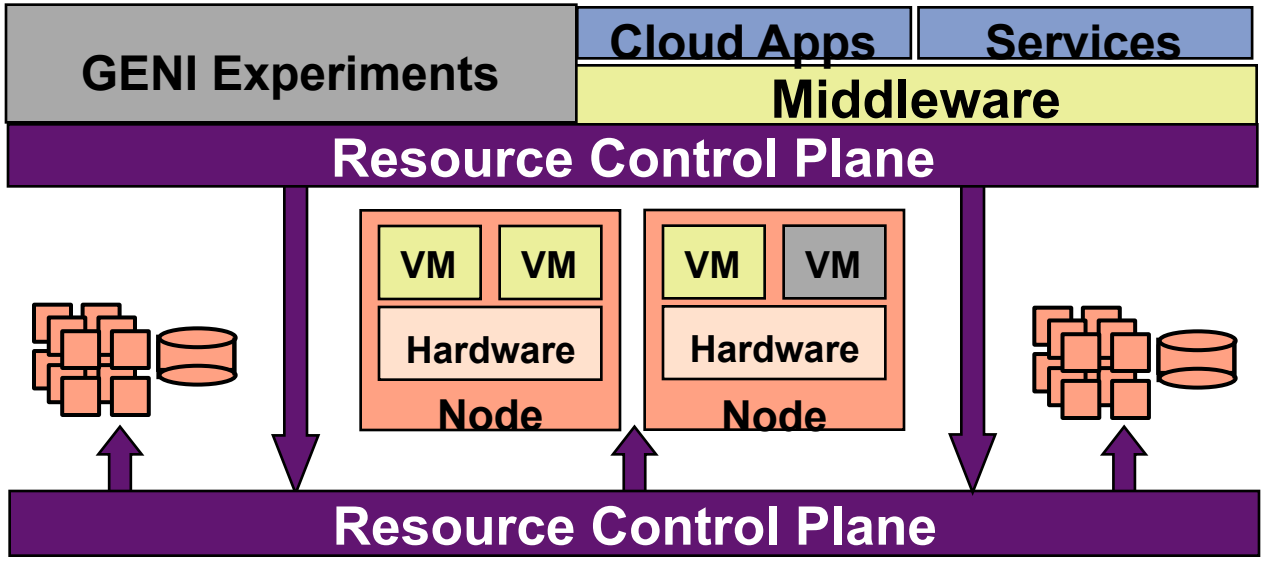
Experimentation w/ Orca/BEN

- Extend Orca to enable slivering of
  - Network elements:
    - Fiber switches
    - DWDM equipment



- Adapt mechanisms to enable flexible description of network slices
  - NDL
- Demonstrate end-to-end slicing on BEN
  - Create realistic but “prefab” slices containing compute, storage and network resources
  - Launch sample applications that expose topology properties
- Link to self-contained edge resources, e.g., over NLR

Open Resource Control



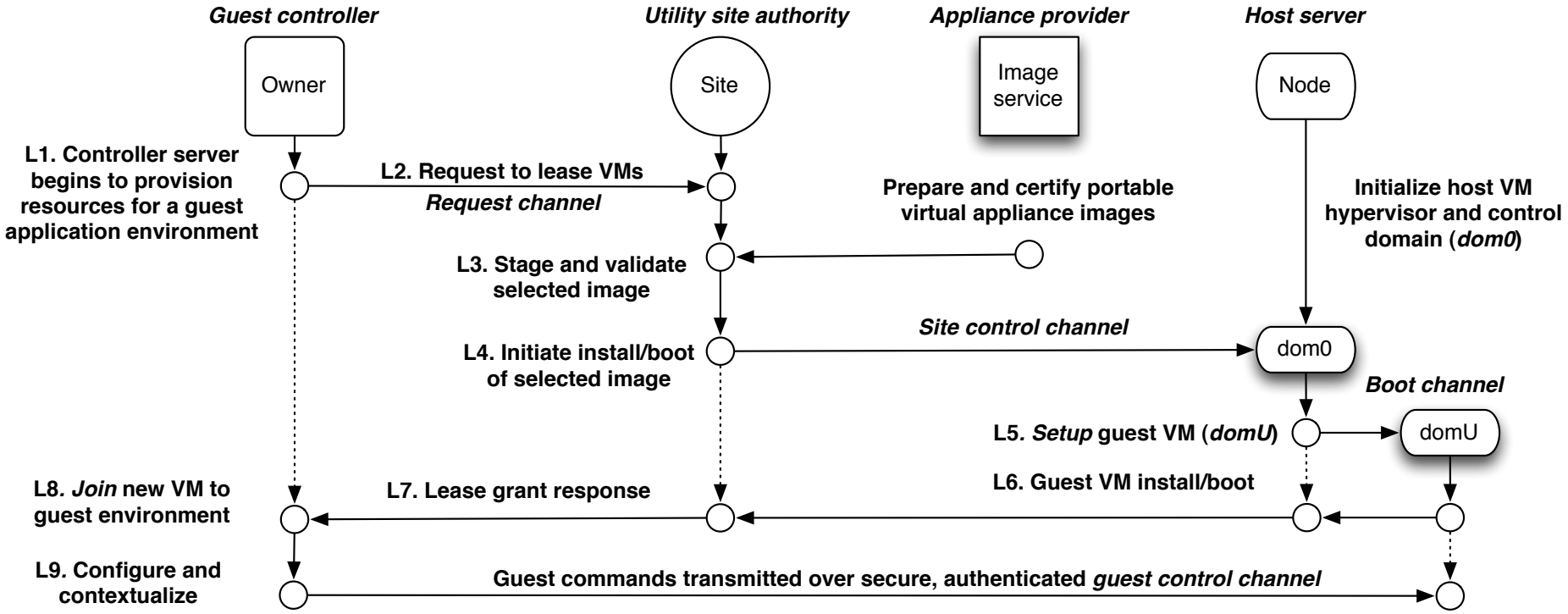
- Contract model for resource peering/sharing/management
- Automated lease-based allocation and assignment
- Flexible resource/contract representations
- Aggregation and accountable delegation
- *http://www.cs.duke.edu/nicl/*

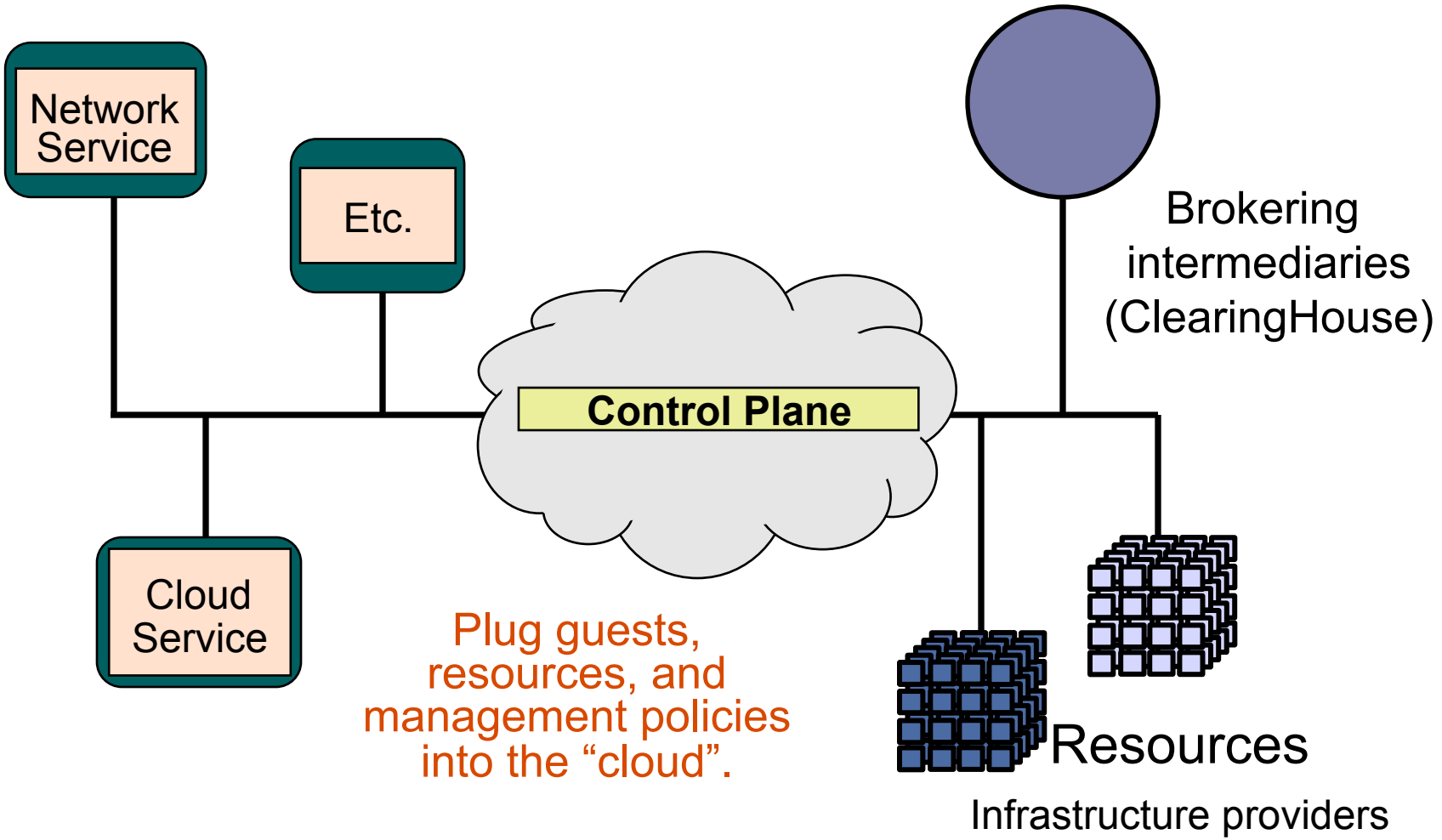
Thanks!

Some Observations

- The Classic Internet is “just an overlay”.
  - GENI is underlay architecture (“underware”)…an exokernel for the Internet.
- Incorporate edge resources: GENI is “cloud computing” + sliverable network
- Multiple domains (MAD): not a “Grid”, but something like dynamic peering contracts
  - Decouple services from substrate; manage the substrate; let the services manage themselves.
- Requires predictable (or at least “discoverable”) allocations for reproducibility
  - QoS at the bottom or not at all?

Example: Guest VM Setup




Network Service
Etc.
Cloud Service
Control Plane
Brokering intermediaries (ClearingHouse)
Plug guests, resources, and management policies into the “cloud”.
Resources
Infrastructure providers

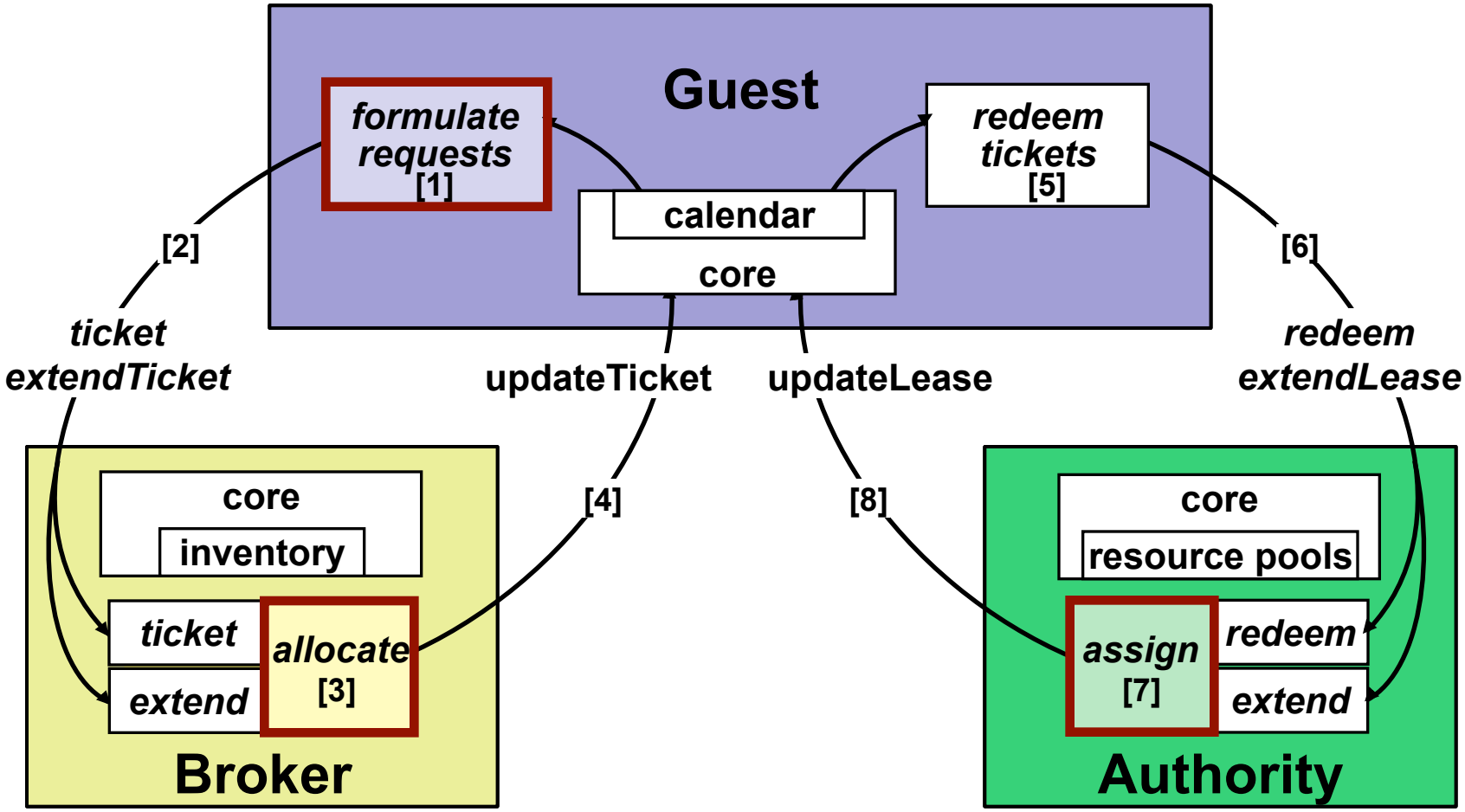
Delegation

- Principle #3: Contracts enable delegation of powers.
  - Delegation is voluntary and provisional.
- It is a building block for creating useful concentrations of power.
  - Creates a potential for governance
  - Calendar scheduling, reservation
  - Double-edged sword?
    - Facility can Just Say No

Aggregation

- Principle #4: aggregate the resources for a site or domain.
  - Primary interface is domain/site authority
- Abstraction/innovation boundary
  - Keep components simple
  - Placement/configuration flexibility for owner
  - Mask unscheduled outages by substitution
  - Leverage investment in technologies for site/domain management

Orca: Actors and Protocols



Network Description Language

```
<ndl:Interface rdf:about="#tdm3.amsterdam1.netherlight.net:501/3">
  <ndl:name>tdm3.amsterdam1.netherlight.net:501/3</ndl:name>
  <ndl:connectedTo
    rdf:resource="http://networks.internet2.edu/manlan/manlan.rdf#manlan:if1"/>
  <ndl:capacity
    rdf:datatype="http://www.w3.org/2001/XMLSchema#float">1.244E+9</ndl:capacity>
</ndl:Interface>
<ndl:Interface rdf:about="http://networks.internet2.edu/manlan/manlan.rdf#manlan:if1">
    <rdfs:seeAlso rdf:resource="http://networks.internet2.edu/manlan/manlan.rdf"/>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:ndl="http://www.science.uva.nl/research/sne/ndl#"
   xmlns:geo="http://www.w3.org/2003/01/geo/wgs84_pos#">
<!-- Description of Netherlight -->
<ndl:Location rdf:about="#Amsterdam1.netherlight.net">
   <ndl:name>Netherlight Optical Exchange</ndl:name>
   <geo:lat>52.3561</geo:lat>
   <geo:long>4.9527</geo:long>
</ndl:Location>
<!-- TDM3.amsterdam1.netherlight.net -->
<ndl:Device rdf:about="#tdm3.amsterdam1.netherlight.net">
      <ndl:name>tdm3.amsterdam1.netherlight.net</ndl:name>
      <ndl:locatedAt rdf:resource="#Amsterdam1.netherlight.net"/>
      <ndl:hasInterface rdf:resource="#tdm3.amsterdam1.netherlight.net:501/1"/>
      <ndl:hasInterface rdf:resource="#tdm3.amsterdam1.netherlight.net:501/2"/>
      <ndl:hasInterface rdf:resource="#tdm3.amsterdam1.netherlight.net:501/3"/>
      <ndl:hasInterface rdf:resource="#tdm3.amsterdam1.netherlight.net:501/4"/>
      <ndl:hasInterface rdf:resource="#tdm3.amsterdam1.netherlight.net:502/1"/>
      <ndl:hasInterface rdf:resource="#tdm3.amsterdam1.netherlight.net:502/2"/>
      <ndl:hasInterface rdf:resource="#tdm3.amsterdam1.netherlight.net:502/3"/>
>
```

UvA Universiteit van Amsterdam

Elements of Orca Research Agenda

- Automate management inside the cloud.
  - Programmable guest setup and provisioning
- Architect a guest-neutral platform.
  - Plug-in new guests through protocols; don't hard-wire them into the platform.
- Design flexible security into an open control plane.
- Enforce fair and efficient sharing for elastic guests.
- Incorporate diverse networked resources and virtual networks.
- Mine instrumentation data to pinpoint problems and select repair actions.
- Economic models and sustainability.