Resource description in GENI: Rspec model

Ted Faber (USC/ISI) & Rob Ricci (University of Utah)

Second GENI Engineering Meeting
3 March, 2008

Outline

- The Problem: Describing GENI Resources
- Current design: Rspec
  - Design features
  - Driving principles
- Open issues
  - Interactions with other working groups

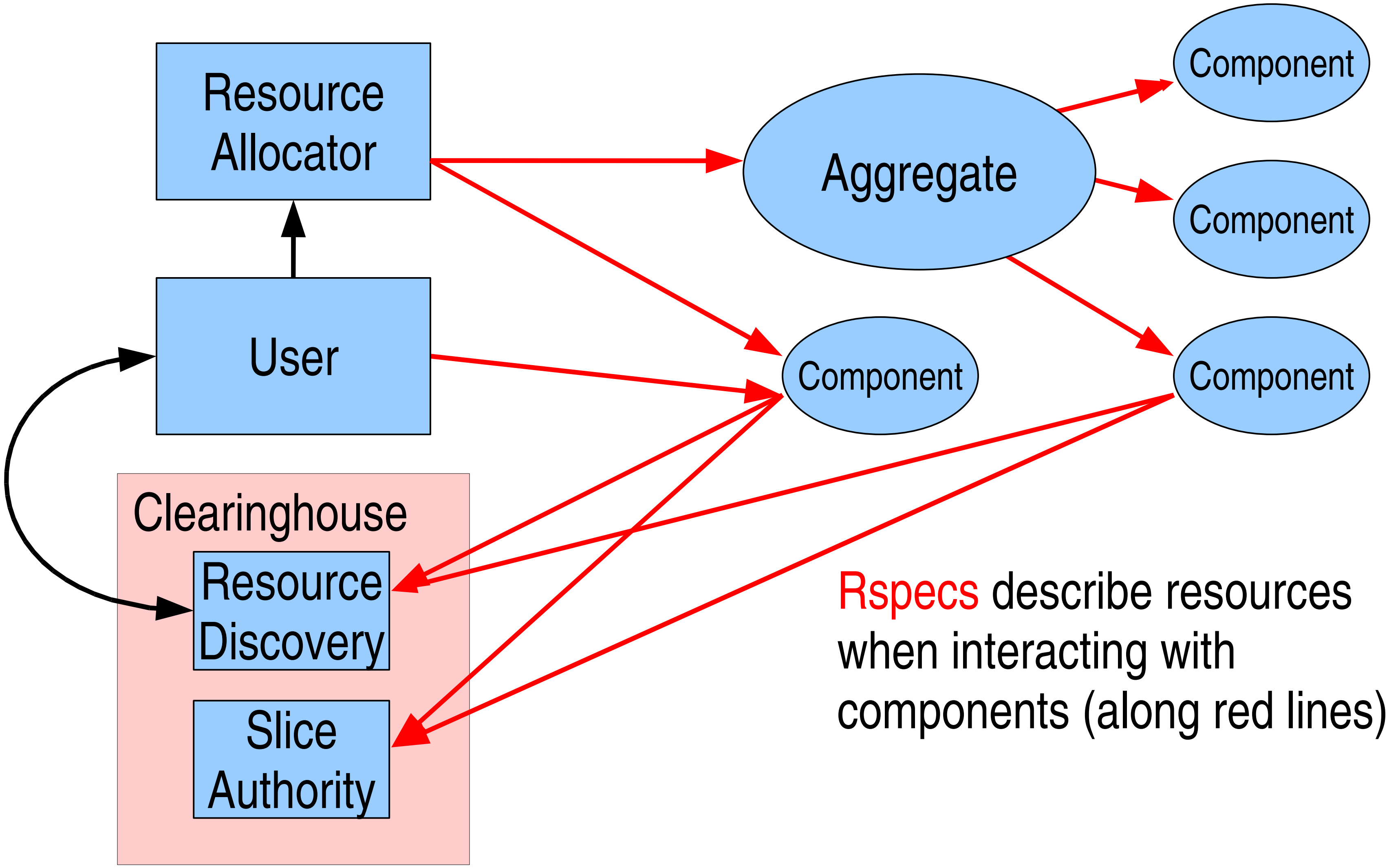
Resource Description in GENI

- Experiment: set of configured resources configured by GMC
- To assemble an experiment, entities must:
  - Describe available resources
    - Components, brokers, portals, discovery services
  - Describe resources requested
    - Components, users, brokers, portals
  - Describe resources allocated or promised
    - Components, aggregates, brokers

Rspec: GENI Resource Description

- Rspec: General, Extensible, Resource Description
- Low Level language: assembly code
- Rspec provides interoperability between
  - Components & resource discovery services
  - Components & resource allocation services
  - Components & users (who want to talk directly)
  - Components & domain specific tools
  - Components & clearinghouses/slice authorities
  - Tools & services that adopt Rspecs

Where Are Rspecs Used?



Rspecs describe resources when interacting with components (along red lines)

Principles of Rspec Design

- Standardized core schema
  - Universal attributes
  - Constraints on attributes
- Extensibility
  - New Components & Technologies
  - New Abstractions
- Composition
  - Incremental descriptions
  - Distributed descriptions

Core Schema: Wide base

- Elemental properties
  - Computation: how much can I process?
  - Communication: how much can I interact with others?
  - Storage: how much information can I hold?
  - Measurement: what can I see?
- Constraints
  - Relations between properties: e.g., CPU v. disk tradeoff

Extensibility: Planning for the Future

- Abstractions
  - Multiple operational modes (virtual router, switch, OS)
- Unique capabilities
  - Lambda switching, frequency allocation
  - Future capabilities
- Configuration mechanisms
- Framework allows interpretation or omission

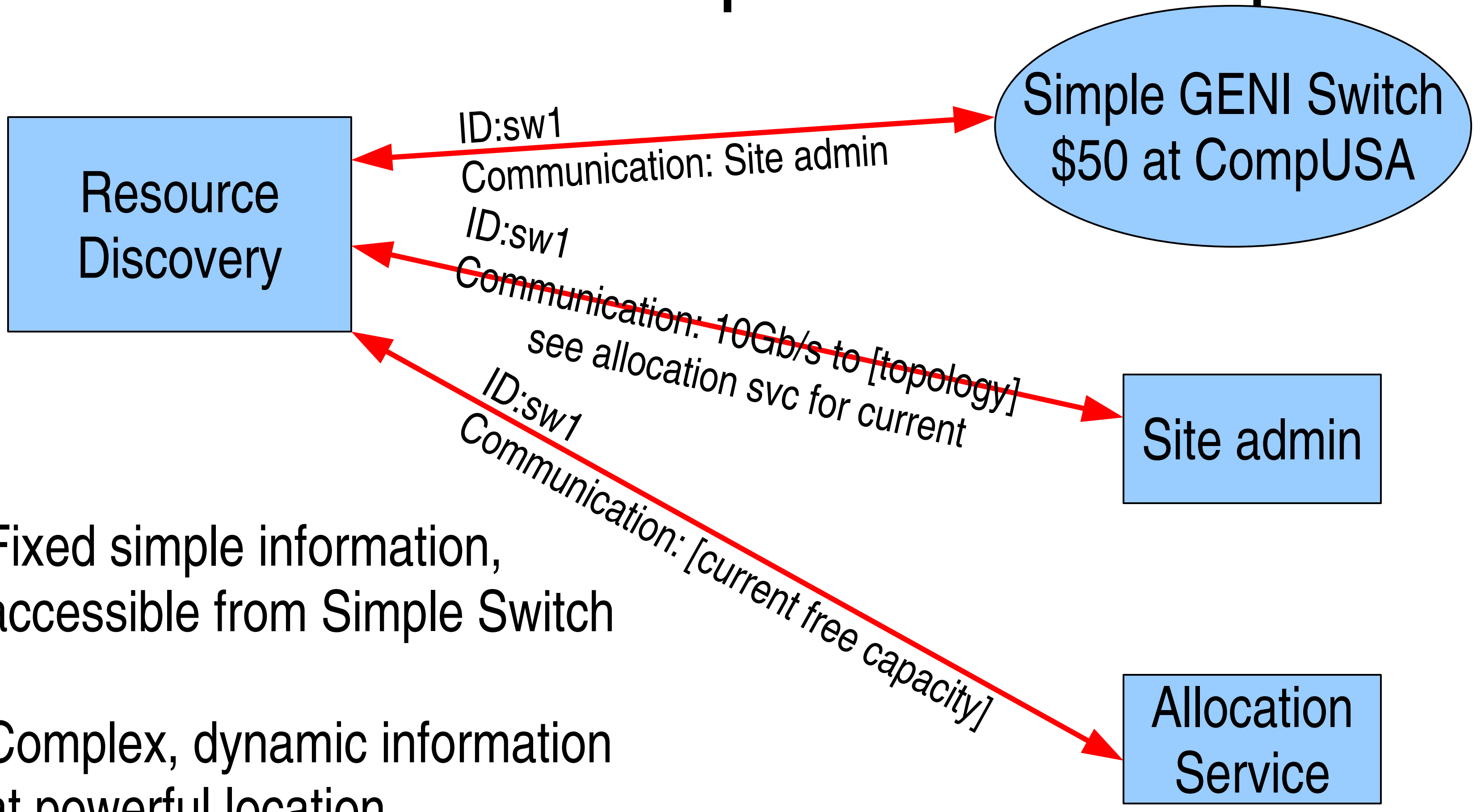
Composition: Scalability & Flexibility

- Basic idea: Identify every resource unambiguously
- Combine data from multiple sources, knowing we are talking about the same resource
- Don't define boundaries in RSpec itself:
  - Who supplies which data about resources
  - Static/dynamic data
  - Appropriate level of detail for everybody
- Biggest challenge: defining identity for “logical” resources

Composition: Iterative and Distributed

- Iterative composition: Tell me more...
  - Allows components to refine description
  - Detail known to users/services that need it
- Distributed composition: Gathering data
  - Different data kept different places
    - Capacity from topology
    - Capacity from availability
  - Allows administrators to partition data appropriately
  - Composition extends to services that use Rspecs

Distributed Composition Example


Resource Discovery
Simple GENI Switch
$50 at CompUSA
ID:sw1
Communication: Site admin
ID:sw1
Communication: 10Gb/s to [topology]
see allocation svc for current
Site admin
ID:sw1
Communication: [current free capacity]
Allocation Service

Fixed simple information,
accessible from Simple Switch

Complex, dynamic information
at powerful location

More Information

- RSpec paper
  - http://groups.geni.net/geni/attachment/wiki/GeniControl/rspec-draft-v0.4.pdf
- XSD schemas
  - XSD: http://www.isi.edu/~faber/GMC/schemas/resources.xsd
  - Documentation: http://www.isi.edu/~faber/GMC/html/resources.html

Open Issues

- Practical extensions: Describing real components
  - Proper framework
  - Coordination
- Connectivity and Topology: Connecting components
- Constraint Language: How expressive?
  - “I never want to execute an Rspec” - Rob Ricci
- Naming Resources and Views: bigger than Rspecs
  - Logical or Virtual resources
  - Lifetimes

Interactions with Other WGs

- Substrate
  - Examples of representing specialized components
    - Assumption testers: Wireless & Optical & ???
- OMISS
  - Rspecs useful for internal tools?
  - Staff may construct Rspecs for legacy equipment
- Services
  - Services must “compile to” Rspecs
  - Use Rspecs between tools?
- Opt-in?