GNI Specifications intro

GLIF Specifications WG
Oct 1, 2008

Evangelos Chaniotakis (haniotak@es.net)
ESnet Network Engineer

Lawrence Berkeley National Laboratory

➢Contents

- Overview
- Rationale & design parameters
- API overview
- Common messages
- Example message exchange
- Base types
- Technology-specific extensions
- Future work
- Discussion

➢Overview

- **Objectives:**
  - Short-term interoperability between our systems
  - Provide lessons learned to OGF NSI WG
- **Activity so far:**
  - Task Force formed after Joint Techs Feb 2008
  - TF character is primarily technical
  - Monthly phone conferences
  - Collaborators from lots of different entities
    - G-lambda, IDC, Inocybe have put in effort
  - Decided to come up with an API and a framework
  - First version of framework just committed to SVN

(See at: http://gusi.inocybe.ca/)

➢Rationale & design parameters

- **Why do we do this?**
  - Our users will be needing VCs
  - Choosing one of the existing schemas is hard
  - An exercise in technical collaboration
- **Design parameters**
  - Initial documents are strawmen
  - Keep it simple for rapid prototyping
  - Minimal number of classes
  - (almost) no assumptions about control plane architecture
  - No assumptions about data plane architecture
  - No provisions for AA (NOT a simple problem!)
  - No provision for exchange of routing information

➢API overview

- **A set of messages that contain:**
  - Common, required, technology-agnostic parameters
  - Optional technology-specific parameters

- **The messages and parameters:**
  - Do not specify a control or data plane architecture
  - Do not specify the message transport mechanism
  - Do not require complicated business logic
  - Were kept intentionally minimal and flexible to ease coding, and accommodate different approaches
  - This is probably not sufficient in the long term…
  - But this TF is NOT about the long term.

➢Discussion

- **Questions so far?**

- **Moving on to the technical side of things..**

➢Messages

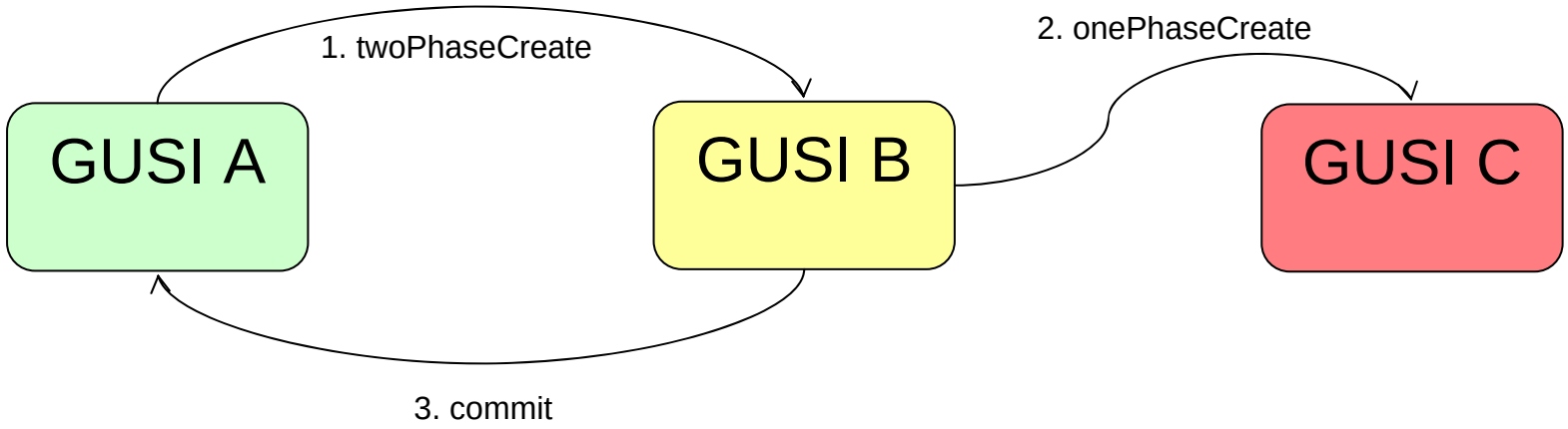
- **listCapabilities()**
  - A “meta” message to allow for capability discovery & business logic adaptation.
  - Is there a simple standard for this?
- **onePhaseCreate()**
  - Basic VC reservation message; does not need a commit().
- **twoPhaseCreate()**
  - Two-phase commit VC reservation message; a commit() message must follow. (Merge with onePhaseCreate?)
- **commit()**
- **rollback()**
  - Two-phase transaction messages

➢Messages

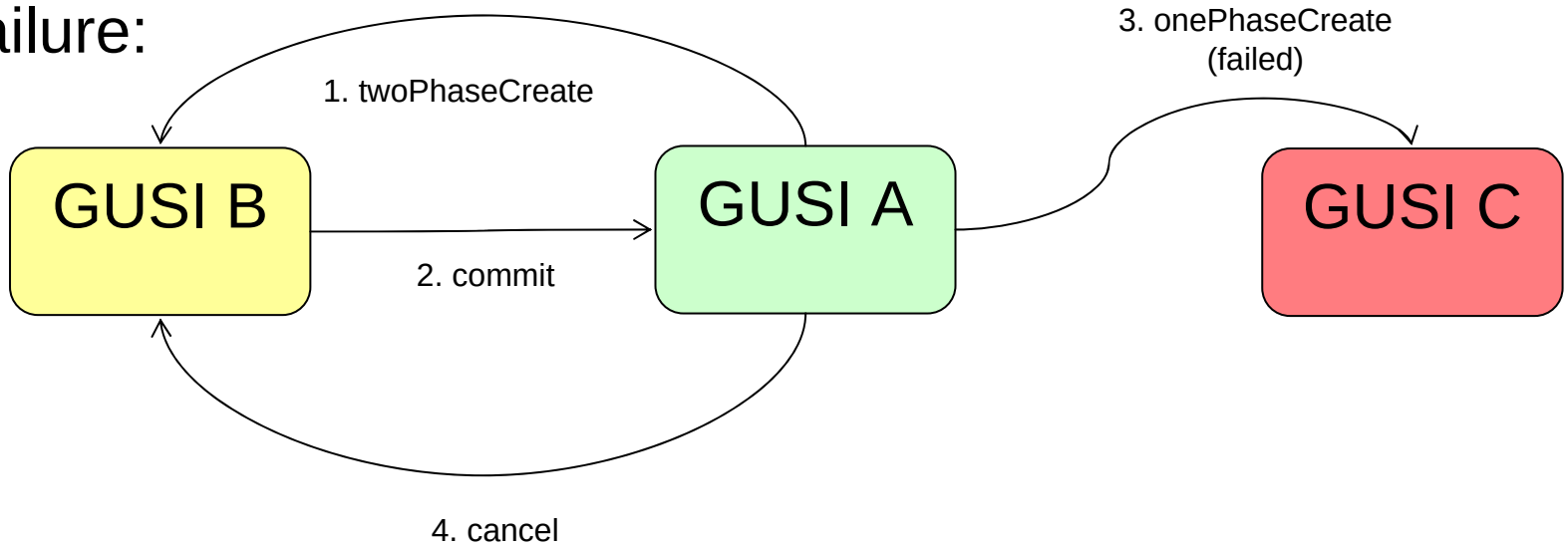
- **cancel()**
- **modify()**
- **list()**
  - Lists existing reservations according to user criteria
- **isPossible()**
  - Optional convenience function to determine whether a reservation is possible with these parameters.
  - Does not place a hold on the resources
- **getEndpoints**
  - Optional convenience function to pull out all the client-facing endpoints served by a GUSI instance
  - NOT a topology exchange!

➢Example message exchange

Success:



Failure:



➢Base types

- **circuitIdentifier:**
  - Globally unique string
- **endpoint:**
  - Globally unique string
    - Are client-facing endpoints different from peering points?
- **interdomainPath:**
  - Data structure that specifies a data plane path
    - Can hopefully be used for alternate / redundant paths, loose or strict
- **circuitDetails:**
  - Structure to contains the above plus trivial (?) admin stuff (reservation start/end times, bandwidth requested, etc)

➢Technology-specific types

- **Ethernet**
  - vlanParameters for each peering / client-facing endpoint
  - isTagged (just in case someone does untagged)
  - vlanRange
  - VLAN translation
- **TDM**
  - Timeslots
- **DWDM**
  - ?

➢Future work

- Shoot holes into strawman, fix omissions
- 10 develop the framework and our resource managers
- 20 test, break things
- 30 goto 10
- Interoperability demo for GLIF winter meeting
- Produce semi-formalized API spec
- Produce lessons learned document