



the globus alliance
www.globus.org

How to Build a Service Using GT4

The short version!

Globus Alliance Staff

Rachana Ananthakrishnan, Charles Bacon,
Lisa Childers, Jarek Gawor, Joe Insley,
Argonne National Laboratory

Ben Clifford, formerly of the USC/Information Sciences Institute



How to Build a Service Using GT4

- **Overview of Services and GT4**
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

Very Brief Overview of GT

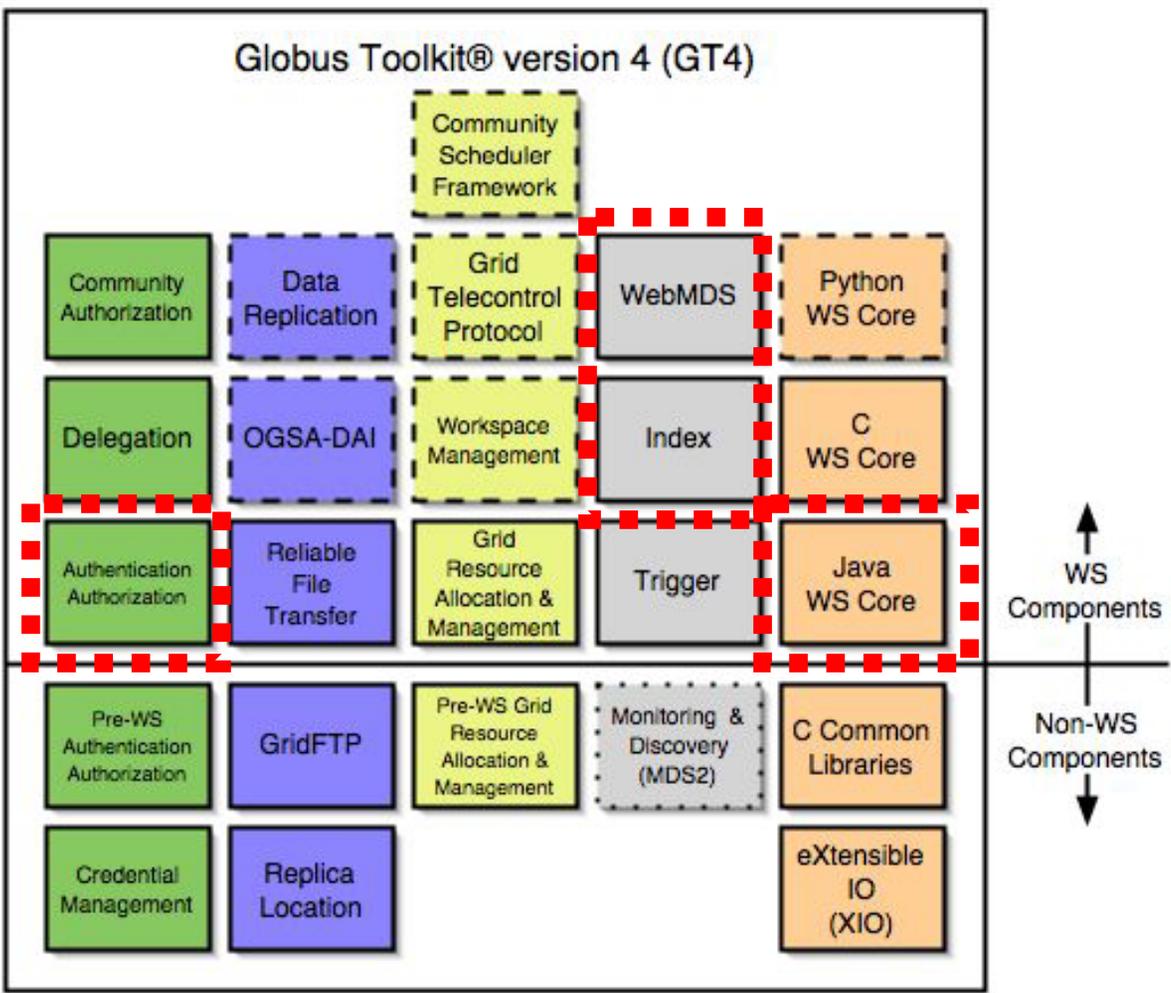
- The Globus Toolkit (GT) is organized as a collection of loosely-coupled components
- GT components contain:
 - ◆ services and clients
 - ◆ libraries
 - ◆ development tools
- GT components are used to build Grid-based applications and services
 - ◆ GT can be viewed as a Grid SDK
- GT components can be categorized across a couple of different dimensions
 - ◆ By broad domain area
 - ◆ By protocol support

GT Domain Areas

- Security
 - ◆ Apply uniform policy across distinct systems
- Data management
 - ◆ Discover, transfer, & access large data
- Execution management
 - ◆ Provision, deploy, & manage resources
- Information services
 - ◆ Discover & monitor resources
- Common runtime
 - ◆ Host existing services and build new services

GT Protocol Support

- Web service protocols
 - ◆ WS-I, WSRF, WSN, WS Addressing, WS Security
- Non Web service protocols
 - ◆ Standards-based, such as GridFTP
 - ◆ Proprietary



- Core GT Component: public interfaces frozen between incremental releases; best effort support
- Contribution/Tech Preview: public interfaces may change between incremental releases
- Deprecated Component: not supported; will be dropped in a future release

How to Build a Service Using GT4

- Overview of Services and GT4
- **Build a Service**
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

Tutorial Structure

- The tutorial is organized as a series of exercises in which increasing functionality is added to a skeletal service implementation
- The exercises demonstrate fundamental Web service interactions using the Globus Toolkit® 4.0
- Each exercise includes:
 - ◆ A discussion of the concepts behind the exercise
 - ◆ A discussion of implementation details
 - ◆ A demonstration of the code
 - ◆ A summary

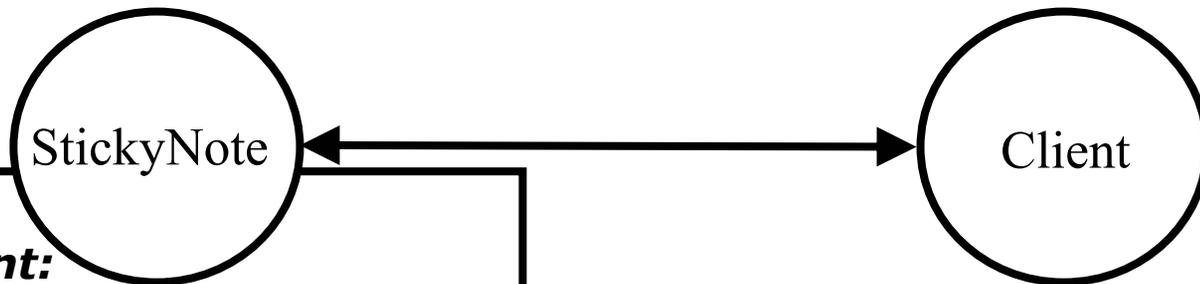
Tutorial Materials

- This slideset
- A GT distribution
- Demonstration code
- Exercise notes
- A community index service
- Two visualizers for the community index's data
- X.509 certificates

How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ **1. Getting Started: Deploy a Service**
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

Exercise 1: Deploy a Service



**1. Deployment:
Stand up a StickyNote service**

Concepts

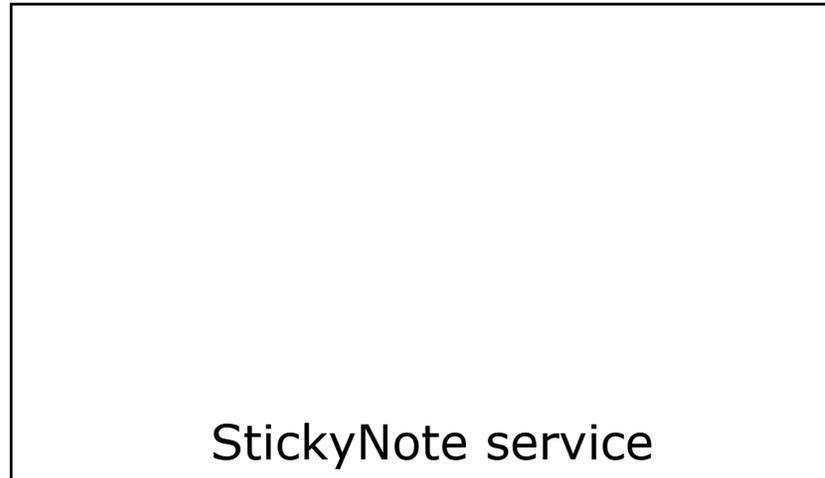
The StickyNote Service

- Today we will work with demonstration code that writes and shows a text message
- The demonstration code is a web service called StickyNote
- StickyNote, like all web services, executes inside a hosting environment (also called a container)
- Today we will use the Java container that is included in GT4

The StickyNote Service

**write-note
client**

**show-note
client**



Implementation details...

Software Requirements

- Jdk
 - ◆ The Java software development kit
- jakarta ant
 - ◆ Open source tool for building Java code (similar to “make”)
- a GT4 binary installation
 - ◆ A subset of the Globus Toolkit, the flagship product of the Globus Alliance. The distribution includes the Java WS Core and WS MDS components of GT.
- StickyNote code
 - ◆ Demonstration code written specifically for this tutorial

Demonstration

Student Notes, Chapter 1

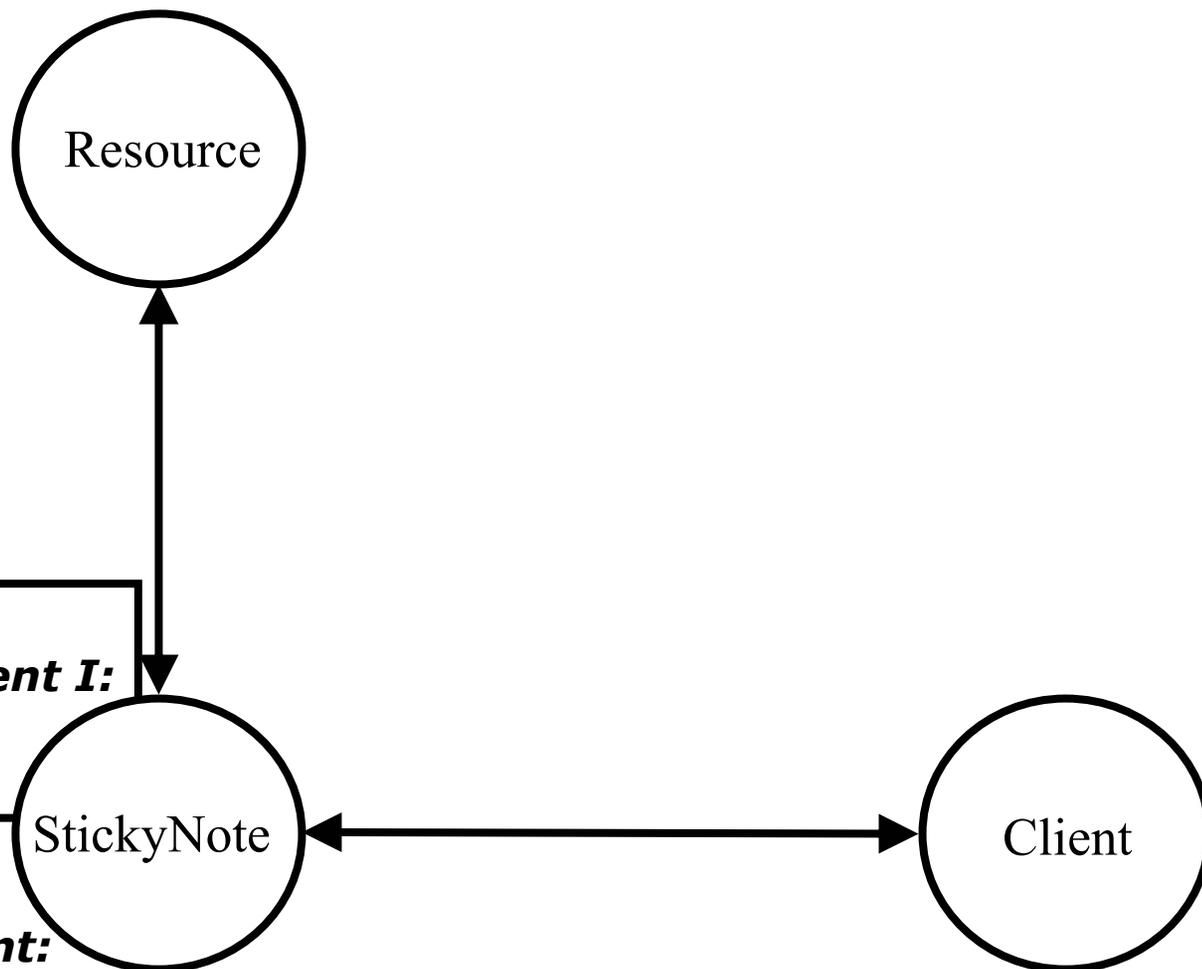
Exercise 1 Review

- Grid services execute inside a hosting environment
 - ◆ Services are deployed into a container
 - ◆ Deployed services start-up when the hosting environment is started
- GT4 includes a hosting environment that supports Java service
- Service capabilities are invoked by clients
 - ◆ show-note, write-note
- Clients do not need to be written in the same language, nor run on the same platform as the services

How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ **2. State Management Part I: Create Resources**
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

Exercise 2: State Management I



**2. State Management I:
Create Resources**

**1. Deployment:
Stand up a StickyNote service**

Service Basics

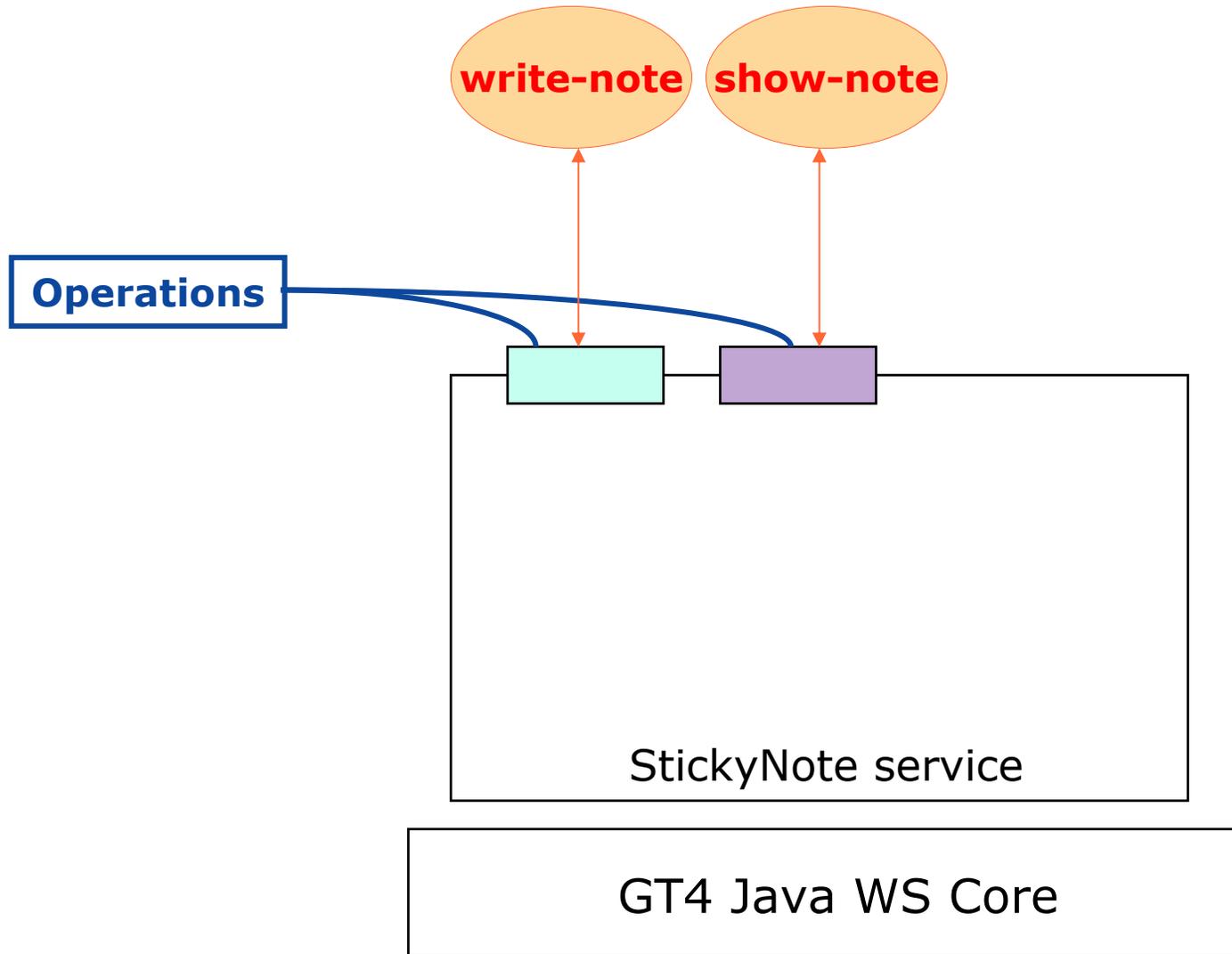
Now that we've deployed and executed the barebones demonstration service we'll begin to modify it in ways that highlight some key Grid computing concepts

We begin with explanations of some GT4 service basics

Service Basics: Operations

- Services have operations
- Service developers can define as many operations as they wish for their services
- Our StickyNote currently has two operations defined
 - ◆ write a note
 - ◆ show a note
- In this chapter we will learn how to add a new operation
 - ◆ create a note

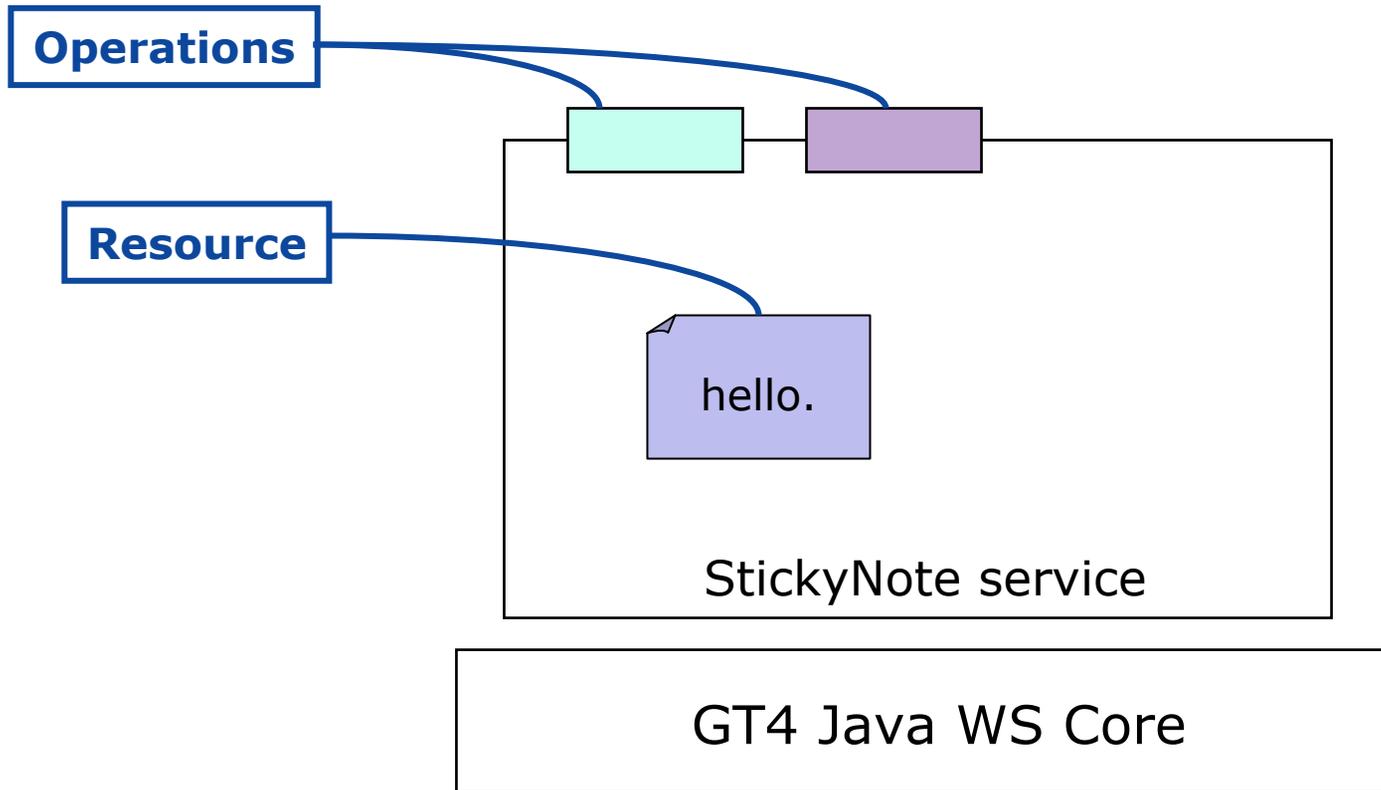
Service Basics: Operations



Service Basics: Resources

- Web services can contain state
 - ◆ state is service-related data that persists across client invocations
- The state of the StickyNote is its message
- The state is stored in a Resource
- The StickyNote service currently contains only a single Resource
- In this chapter we will learn how to add support for multiple notes

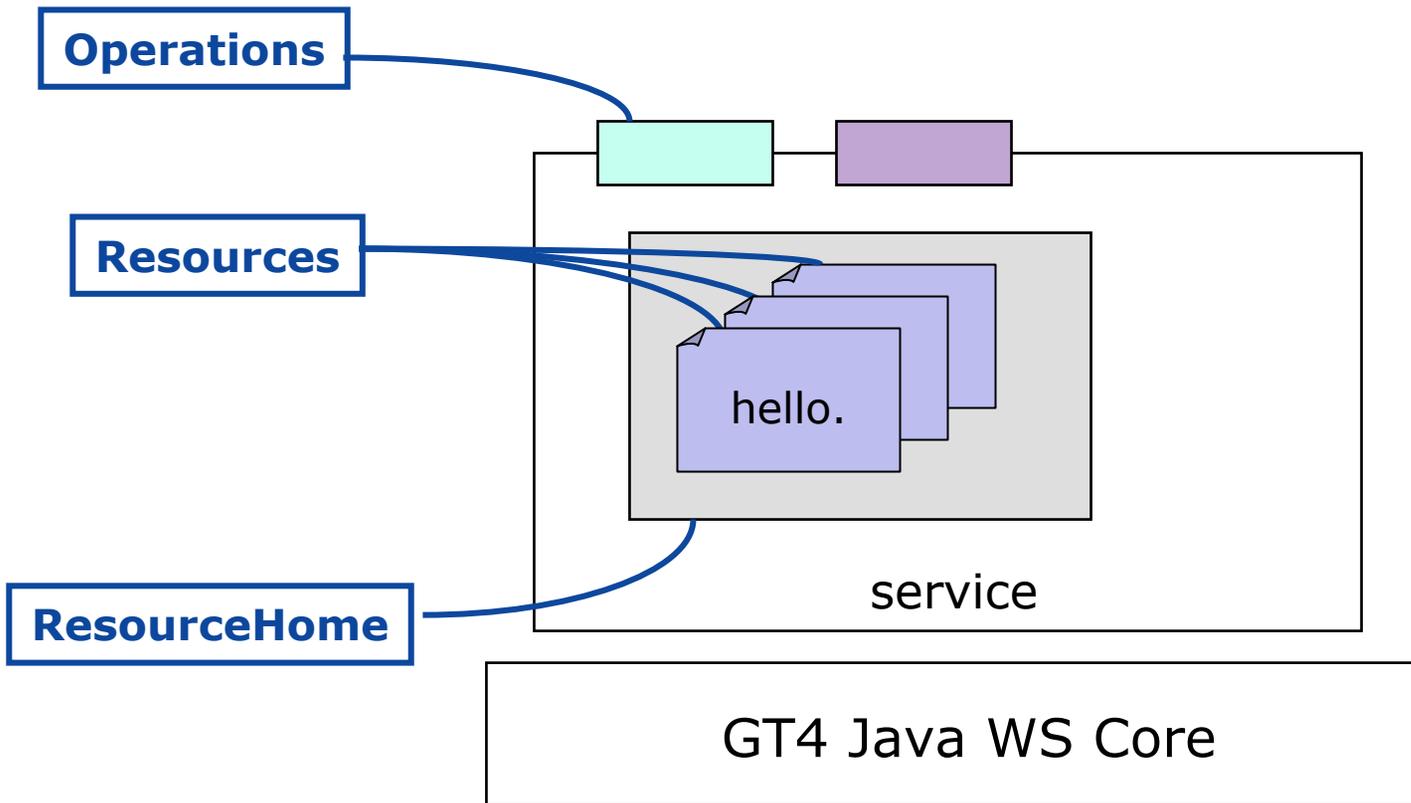
Service Basics: Resources



Service Basics: Resource Homes

- Resources are managed by Resource Homes
 - ◆ Resource Homes provide an interface for adding and removing Resources
- In order to add support for multiple notes we must change from using a Resource Home that is limited to one Resource to a Resource Home that can hold multiple Resources

Service Basics: Resource Homes



Implementation details...

Types of Files

- Interface description files: **.wsdl*
 - ◆ Defines the public interface for the service
- Service and client source files: **.java*
 - ◆ Contain the implementation of the interface
- Build instructions for ant: *build.xml*
 - ◆ Like a makefile
- Service deployment instructions: **.wsdd & jndi*
 - ◆ Specifies container-level service configuration information

Demonstration

Student Notes, Chapter 2

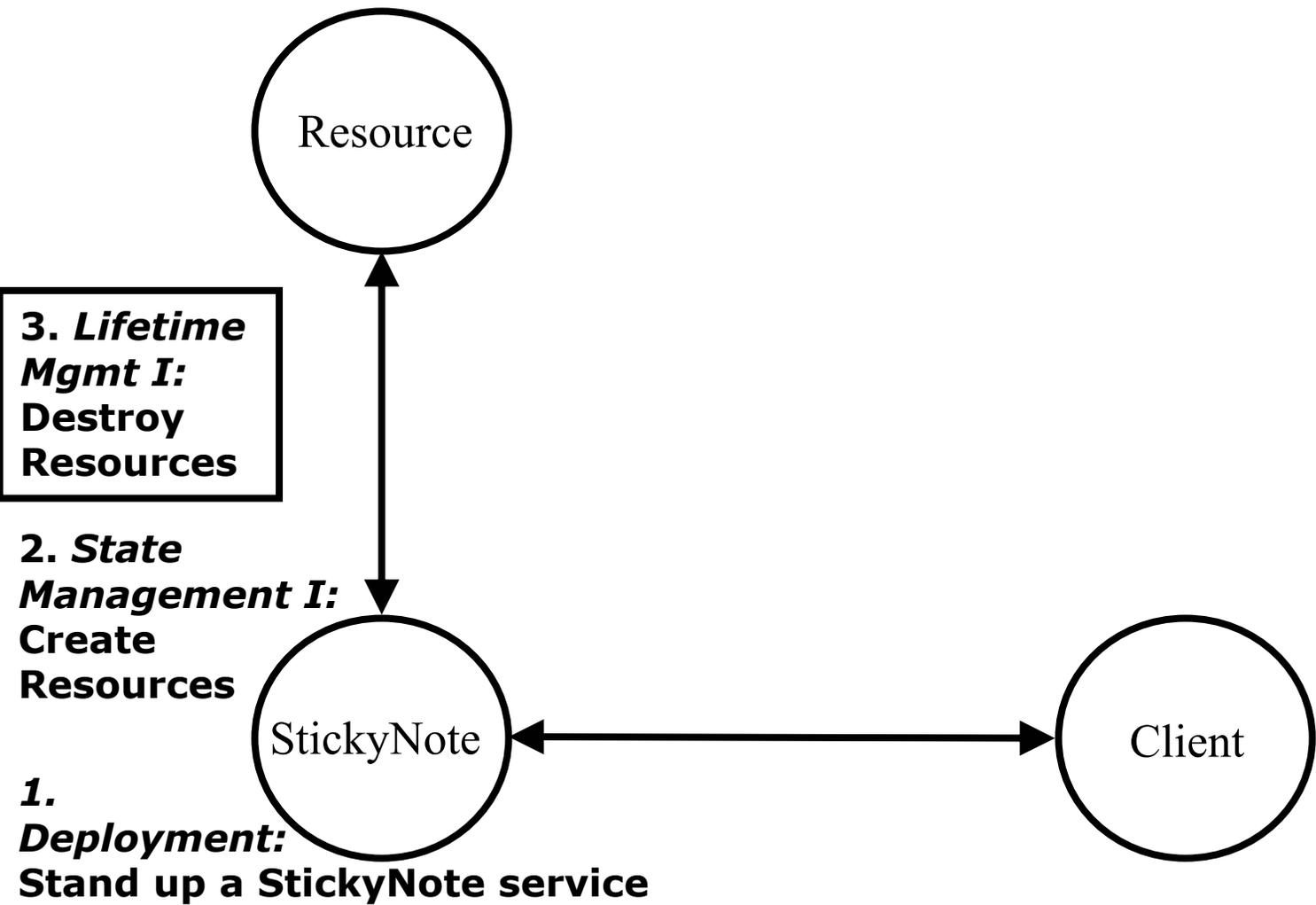
Exercise 2 Review

- Services have operations
- Services can have stateful components, called Resources
- The multiple resources contain different state, but are accessed via the same service
- The WSDL contains definitions of types, messages, and operations
- The Java contains implementations of the operations defined in WSDL

How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ **3. Lifetime Management Part I: Destroy Resources**
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

Exercise 3: Lifetime Management

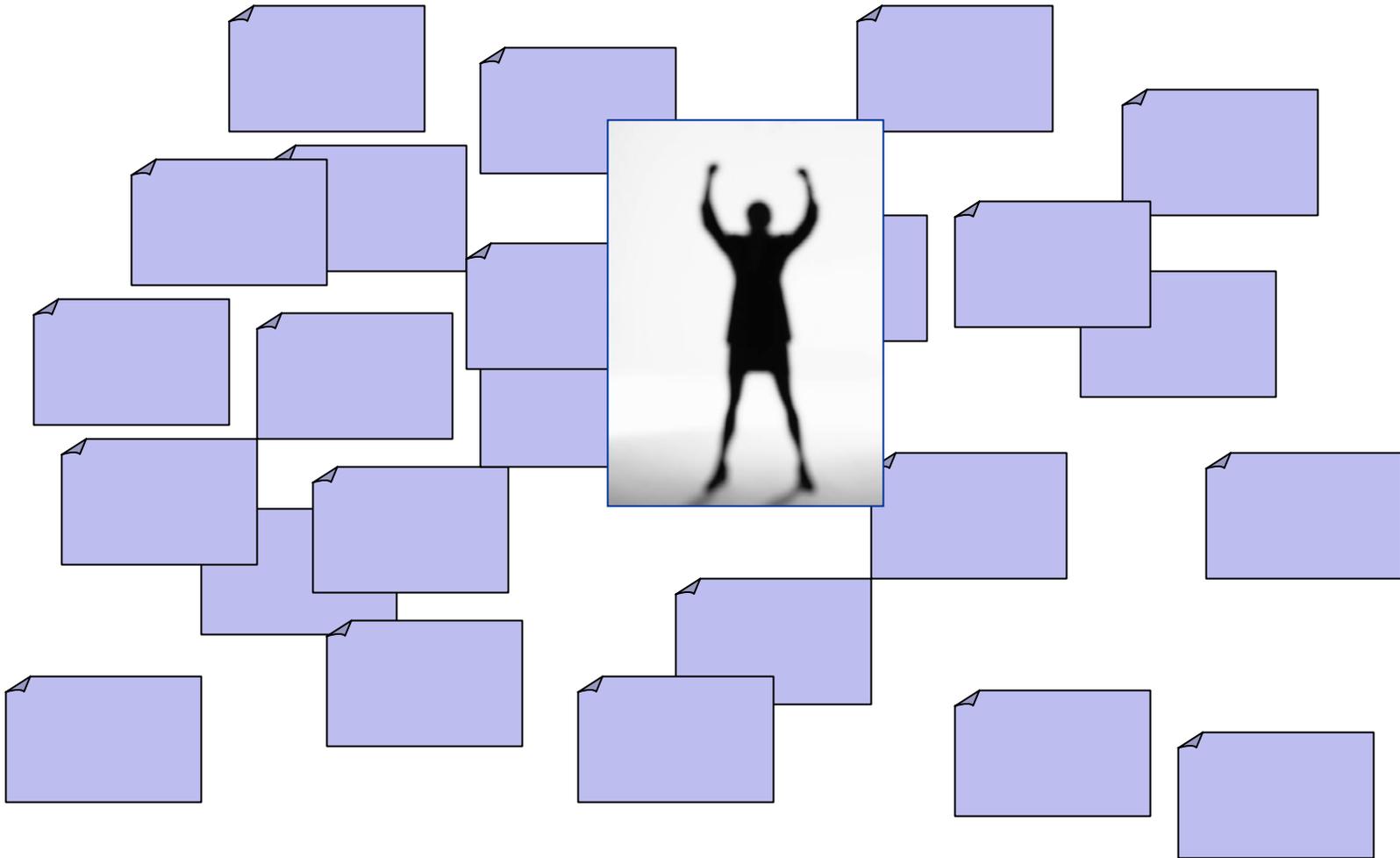


Concepts

Chapter Overview

- If we keep creating resources, we'll eventually overwhelm our system
- In this chapter we will learn how to destroy resources
- To do so, we must be able to identify an individual resource

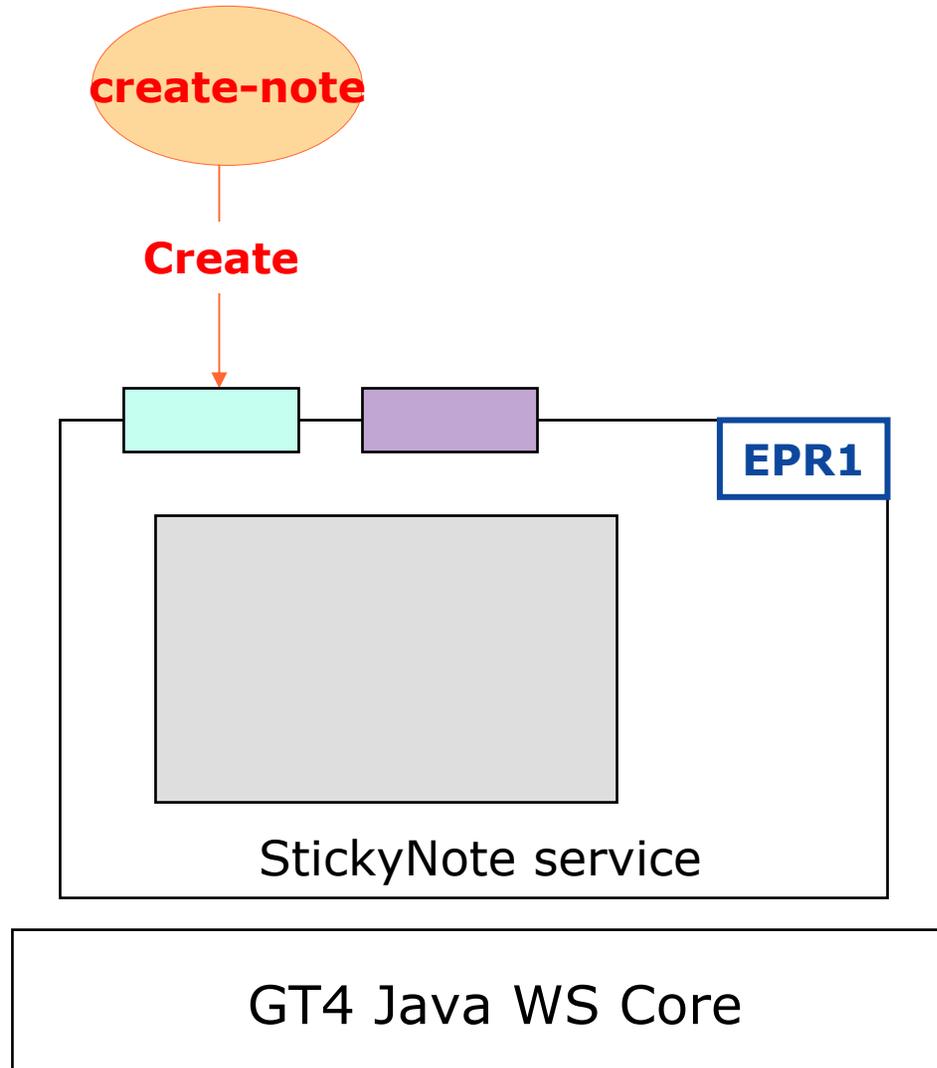
On Lifetime Management



EPRs as Identifiers

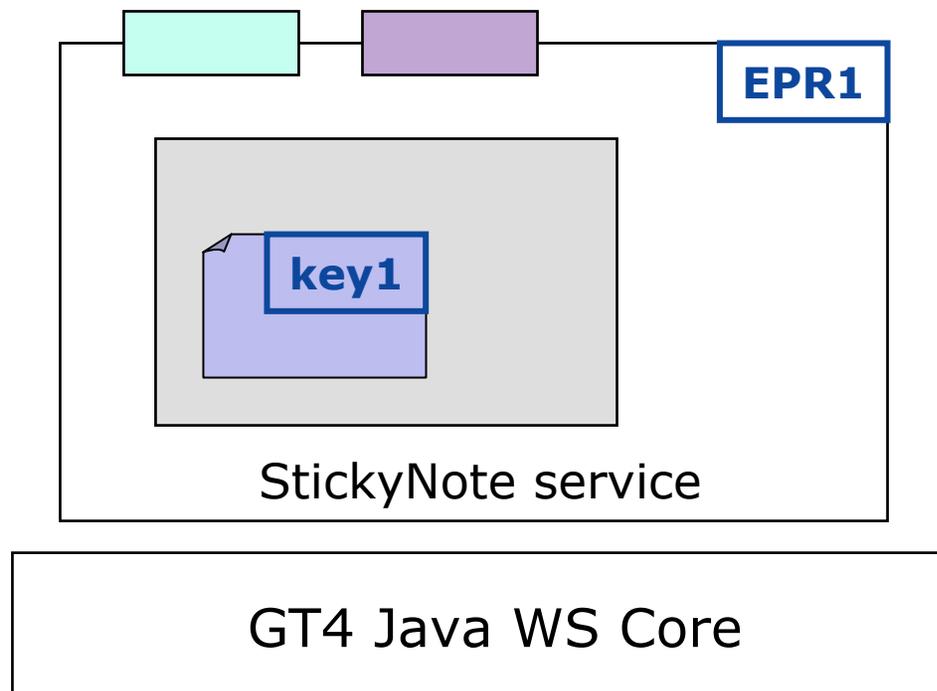
- In order to destroy a particular Resource, we must be able to identify it
- In Web services, End Point Identifiers (EPRs) are used to identify a particular endpoint on the network
- The WSRF specs define a way that EPRs can be used to identify a single Resource
- Each Resource in our StickyNote service has an EPR

A Runtime Example of EPR Creation

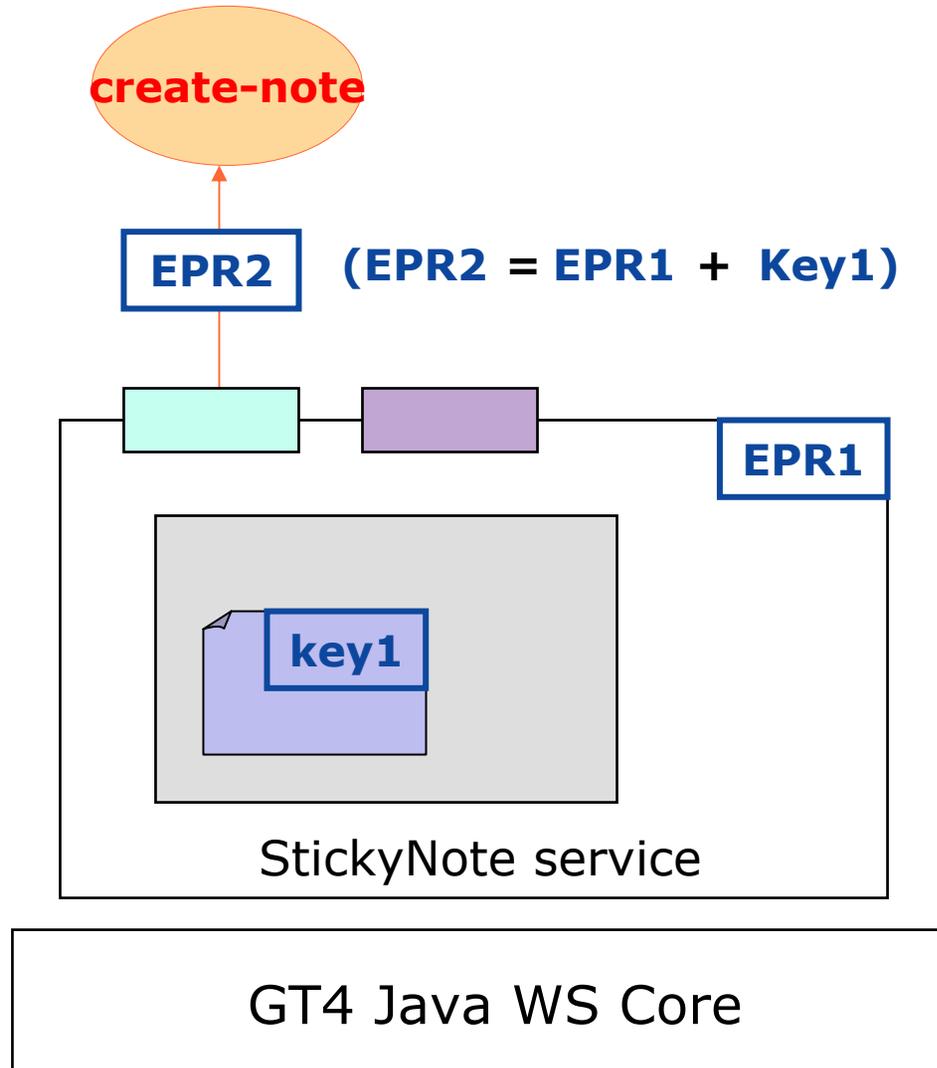


A Runtime Example of EPR Creation

create-note

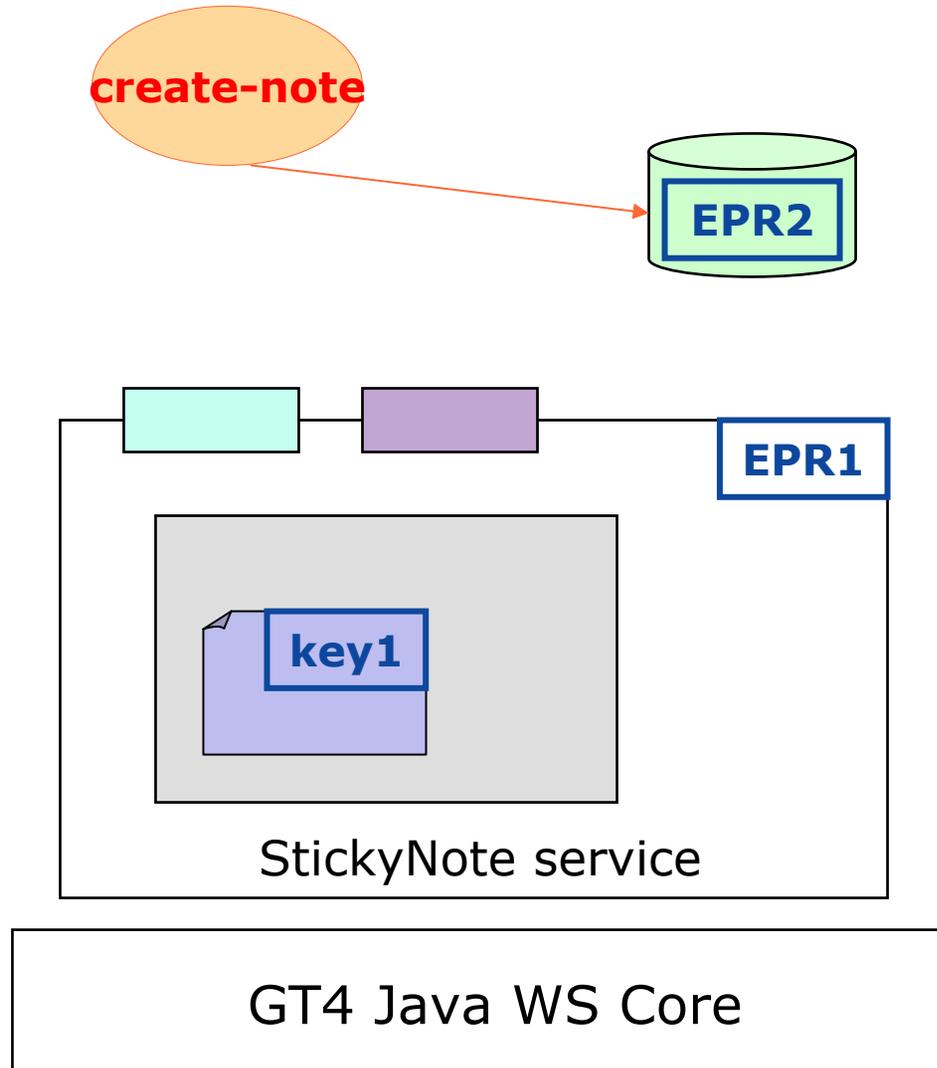


A Runtime Example of EPR Creation



A Runtime Example of EPR Creation

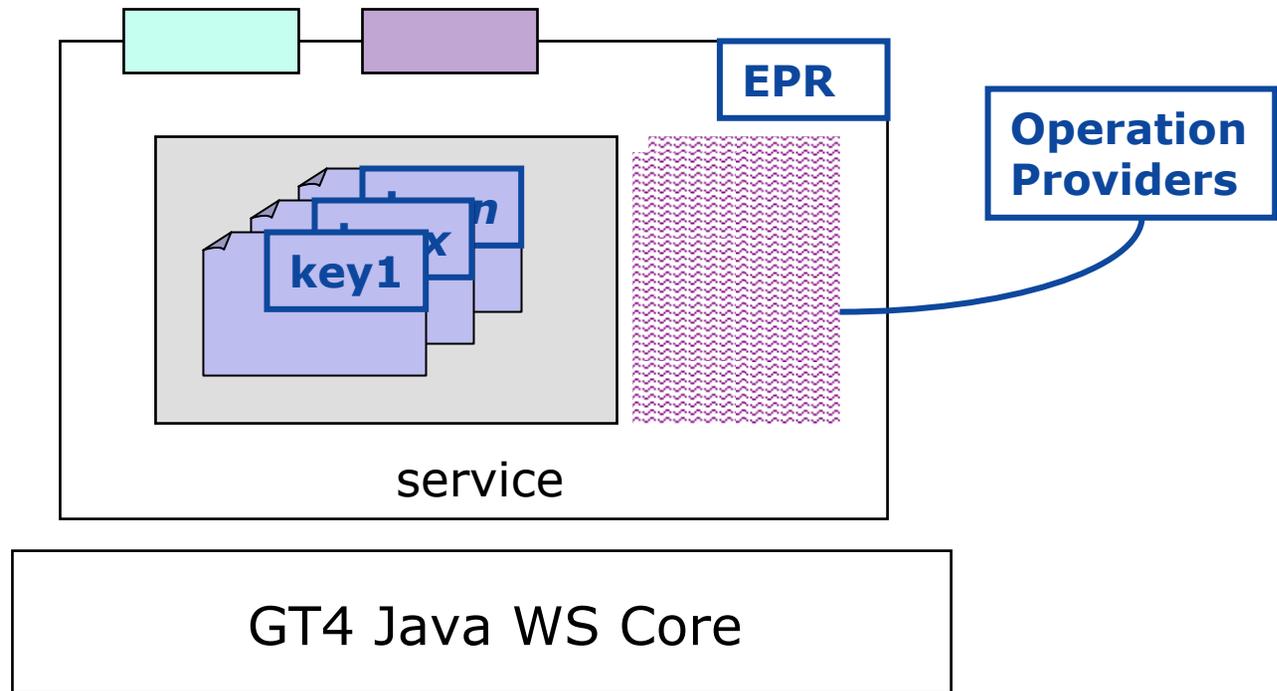
create-note



Resource Destruction

- The ability to destroy Resources is built-in to the GT4 hosting environment
- Code to perform destruction is included in the Toolkit as an operation provider
- An operation provider is simply a Java class that contains a discrete set of functionality
- GT4 includes support for adding operation providers to existing services

On Operation Providers

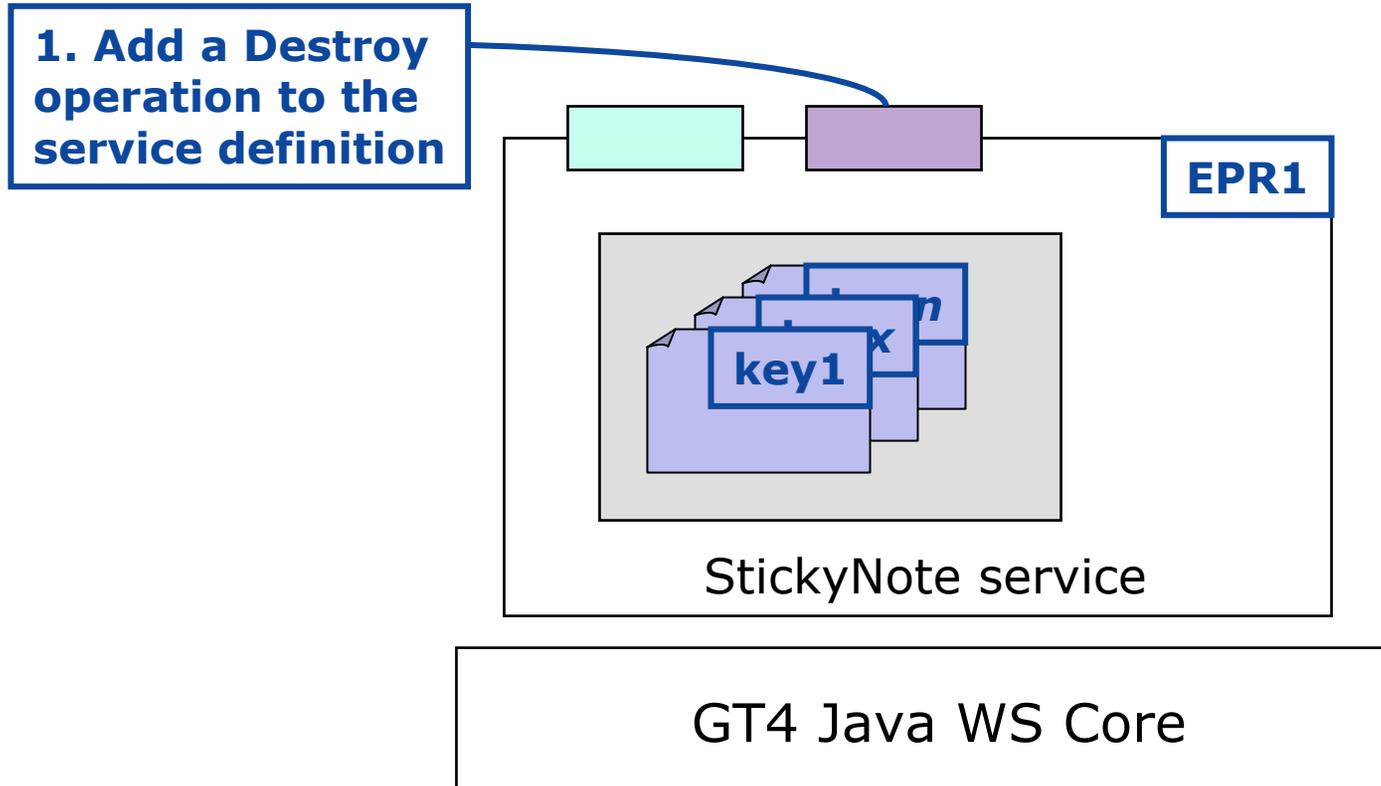


Implementation details...

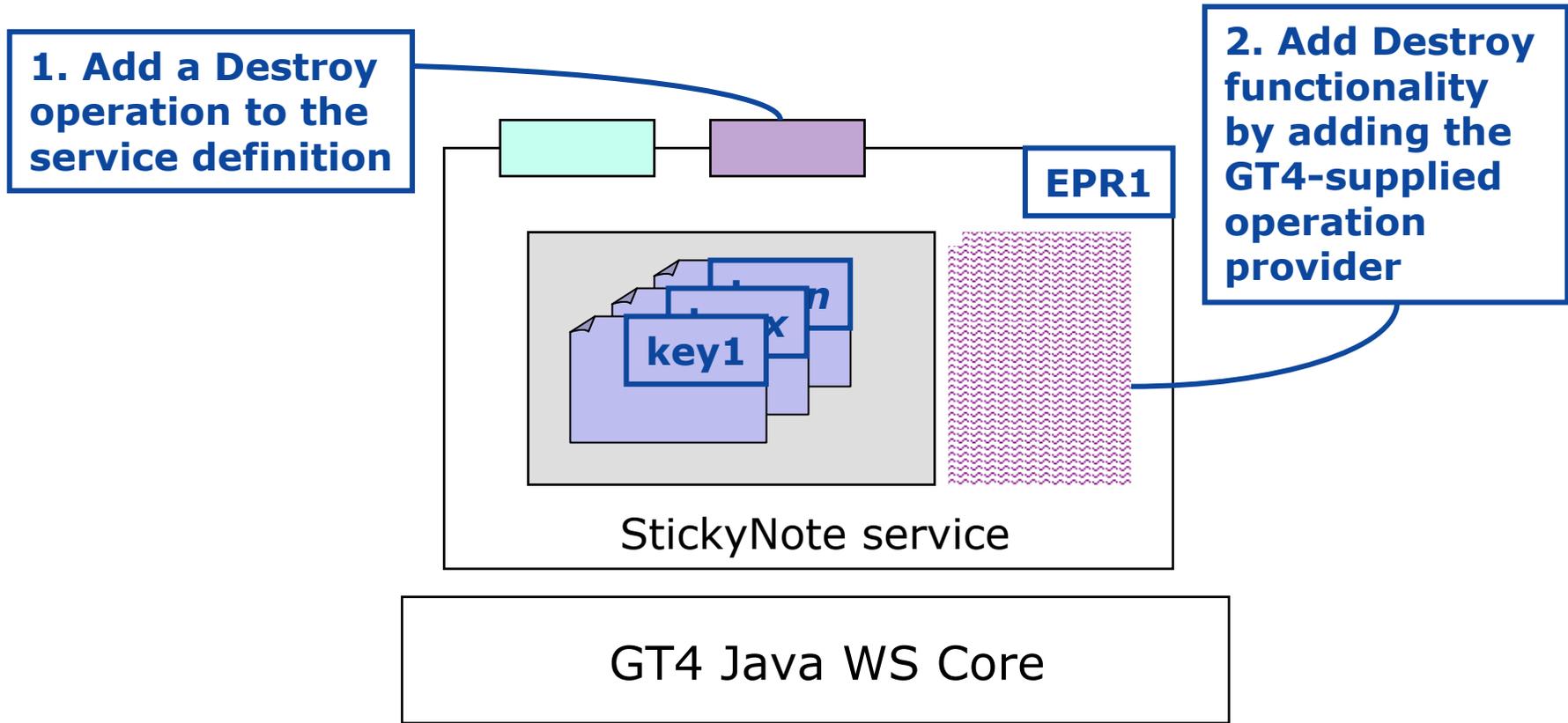
Destroying StickyNote Resources

- We will use the EPR generated by the create operation to identify the resource we want to destroy
- To do this, we need to add a destroy operation defined in WSDL, and provide the implementation of the operation in our service via an operation provider

Adding Destroy to StickyNote



Adding Destroy to StickyNote



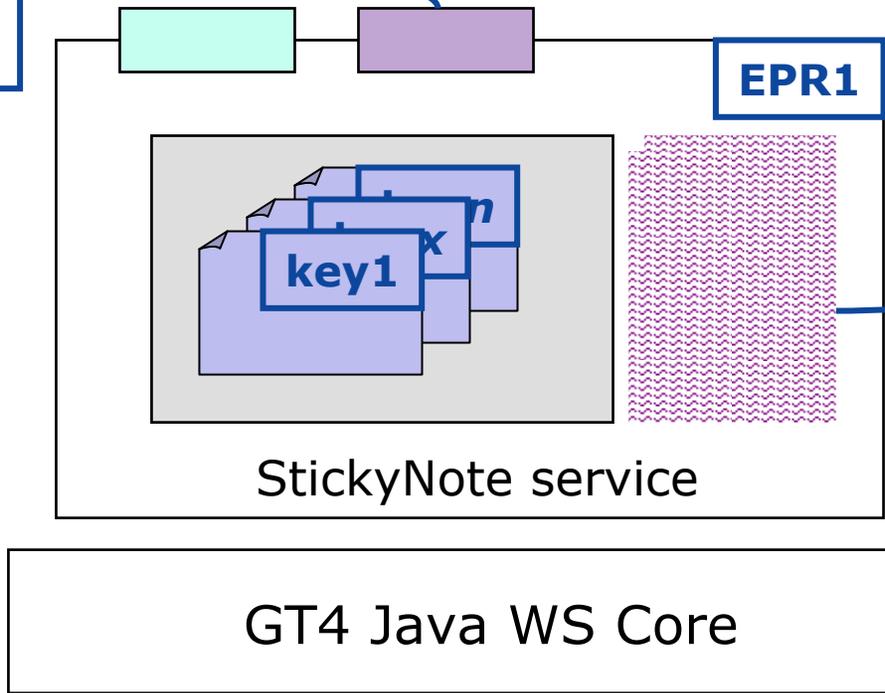
Adding Destroy to StickyNote

3. Use the GT4-supplied Destroy client to invoke the operation

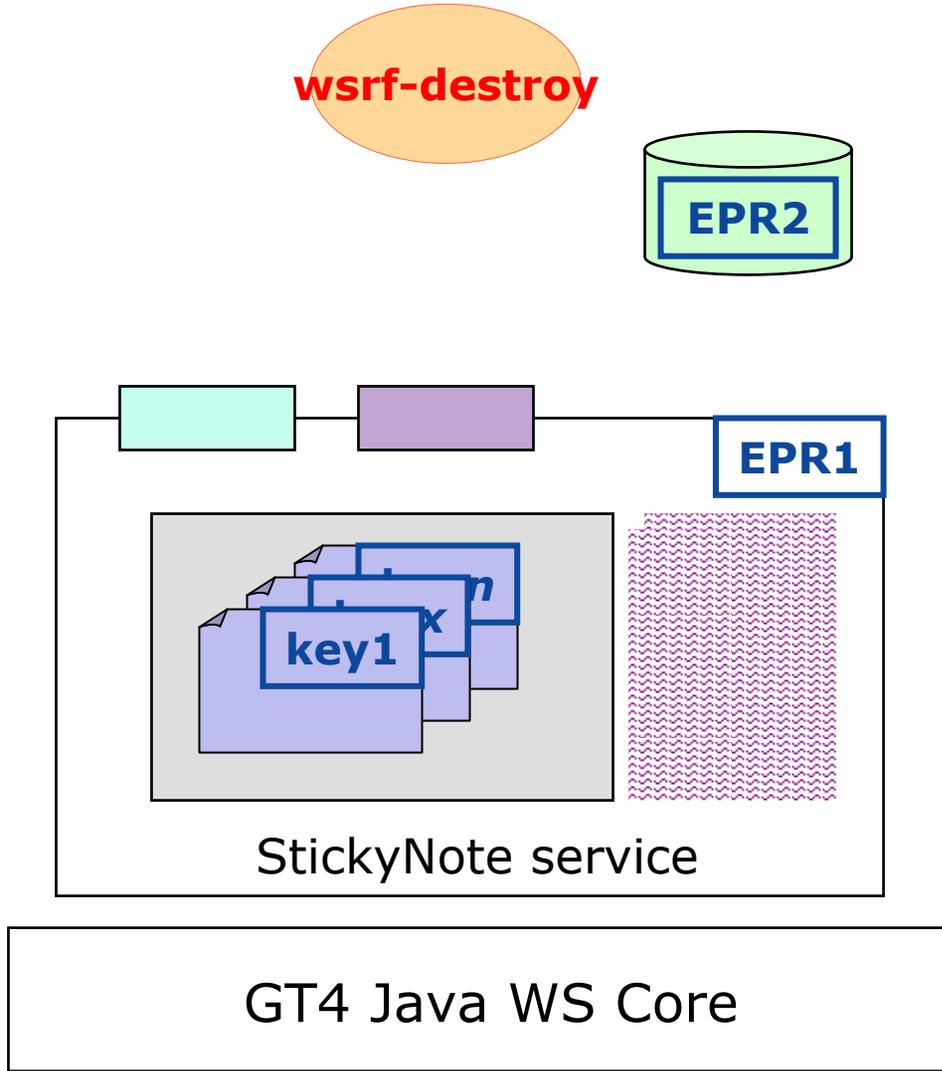
wsrfl-destroy

1. Add a Destroy operation to the service definition

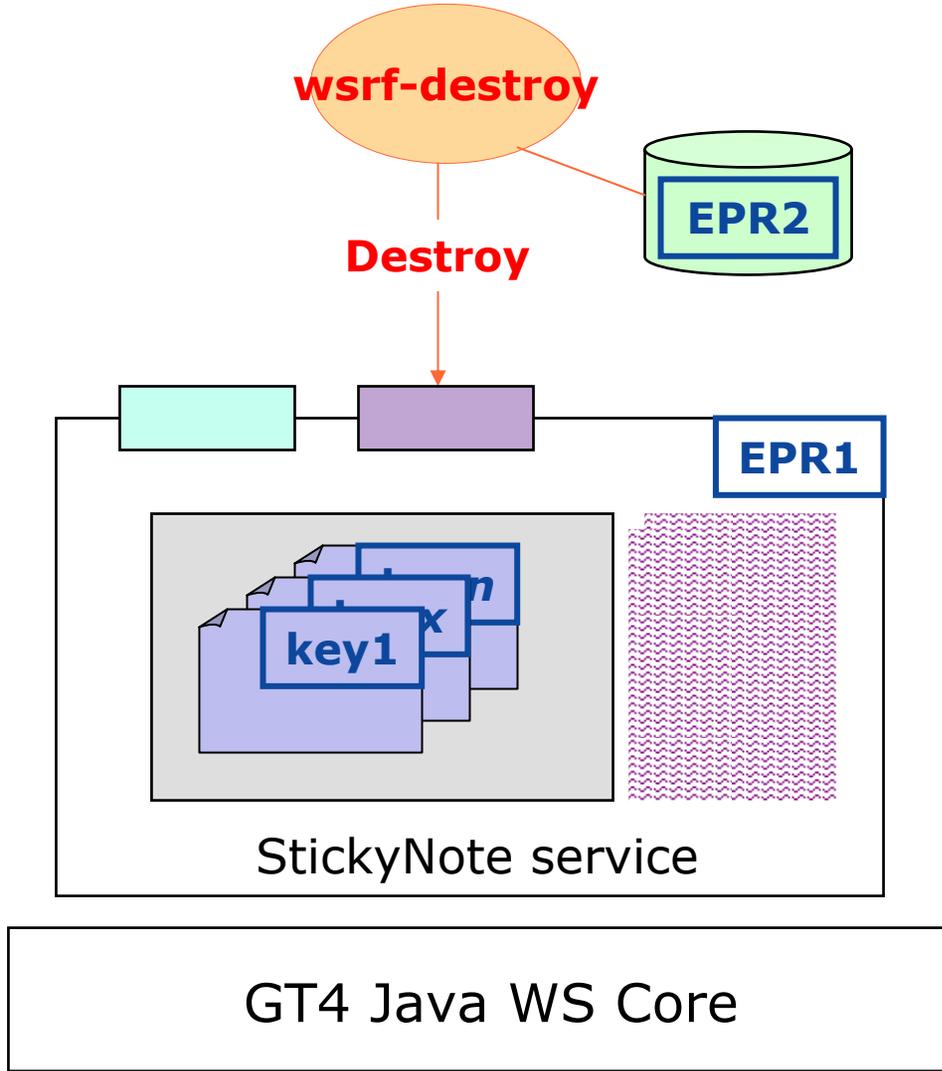
2. Add Destroy functionality by adding the GT4-supplied operation provider



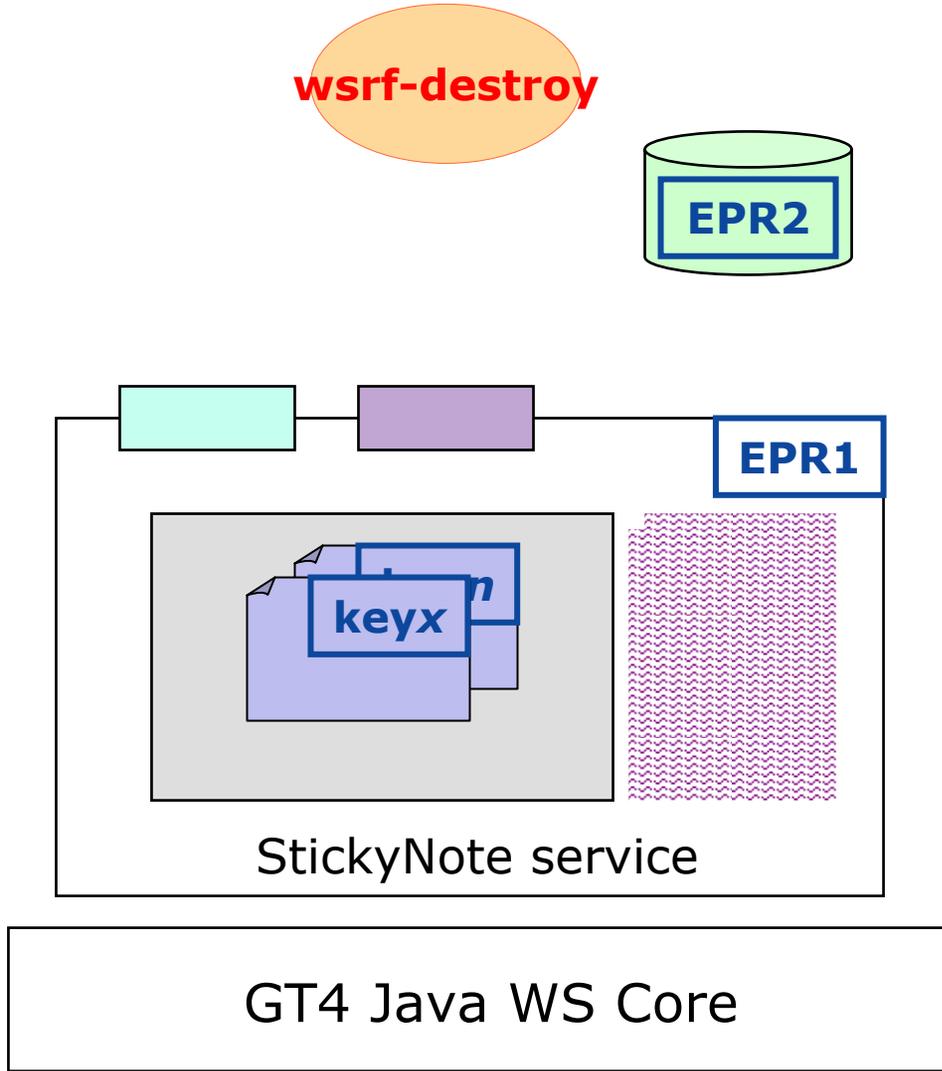
Resource Destruction Runtime



Resource Destruction Runtime



Resource Destruction Runtime



Demonstration

Student Notes, Chapter 3

Exercise 3 Review

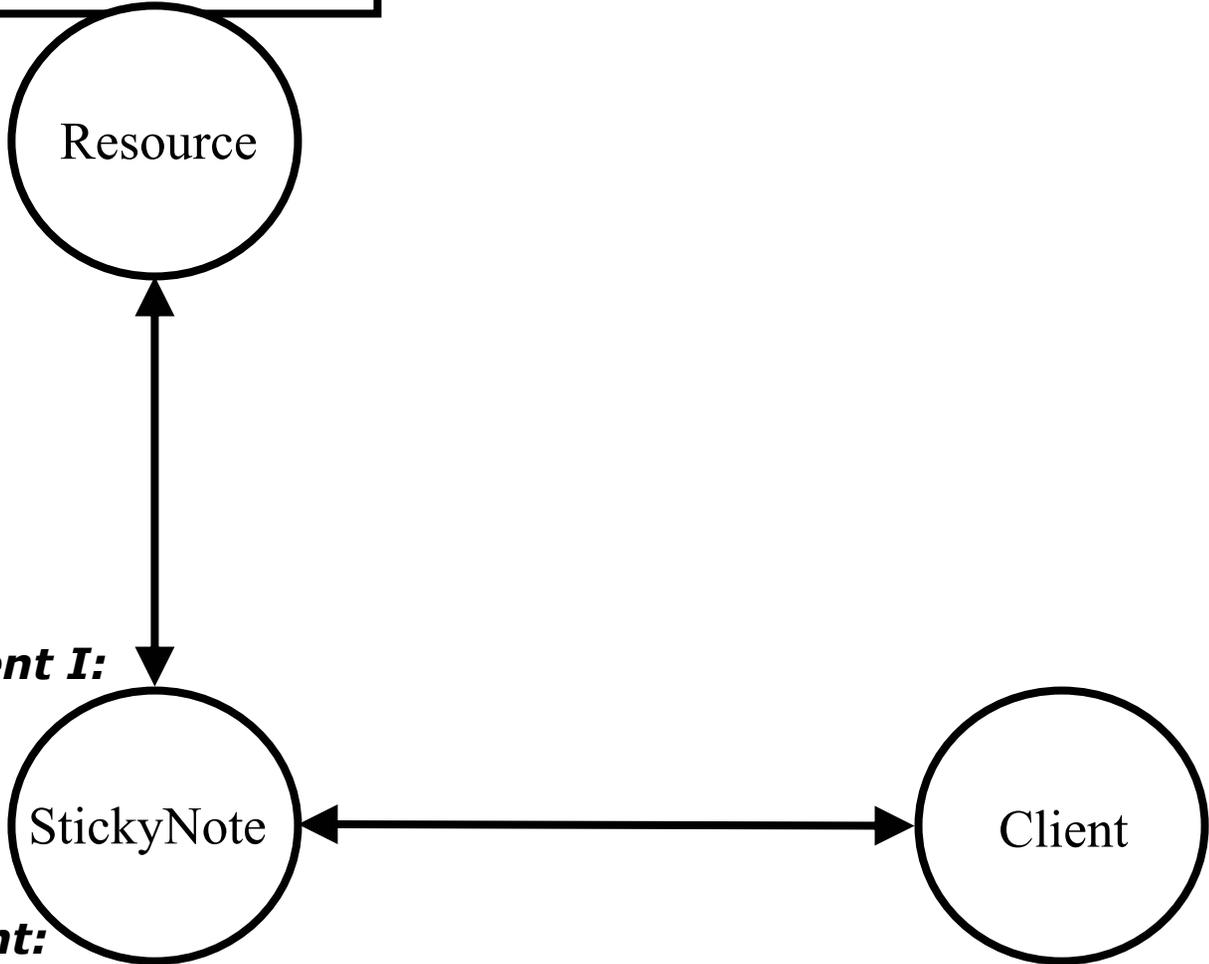
- EPRs identify targets (endpoints)
 - ◆ In our case, identifying a service+resource pair
- EPRs can be used to identify Resources to destroy
- Service developers use operation providers to add service functionality
- GT4 provides several pre-written operation providers and clients for managing Resources, including the destroy capability
- GT4 also provides pre-written clients for managing Resources, such as wsrf-destroy

How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ **4. State Management Part II: Add a Resource Property**
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

Exercise 4: State Management II

**4. State Management II:
Add a Resource Property**



**3. Lifetime
Mgmt I:
Destroy
Resources**

**2. State
Management I:
Create
Resources**

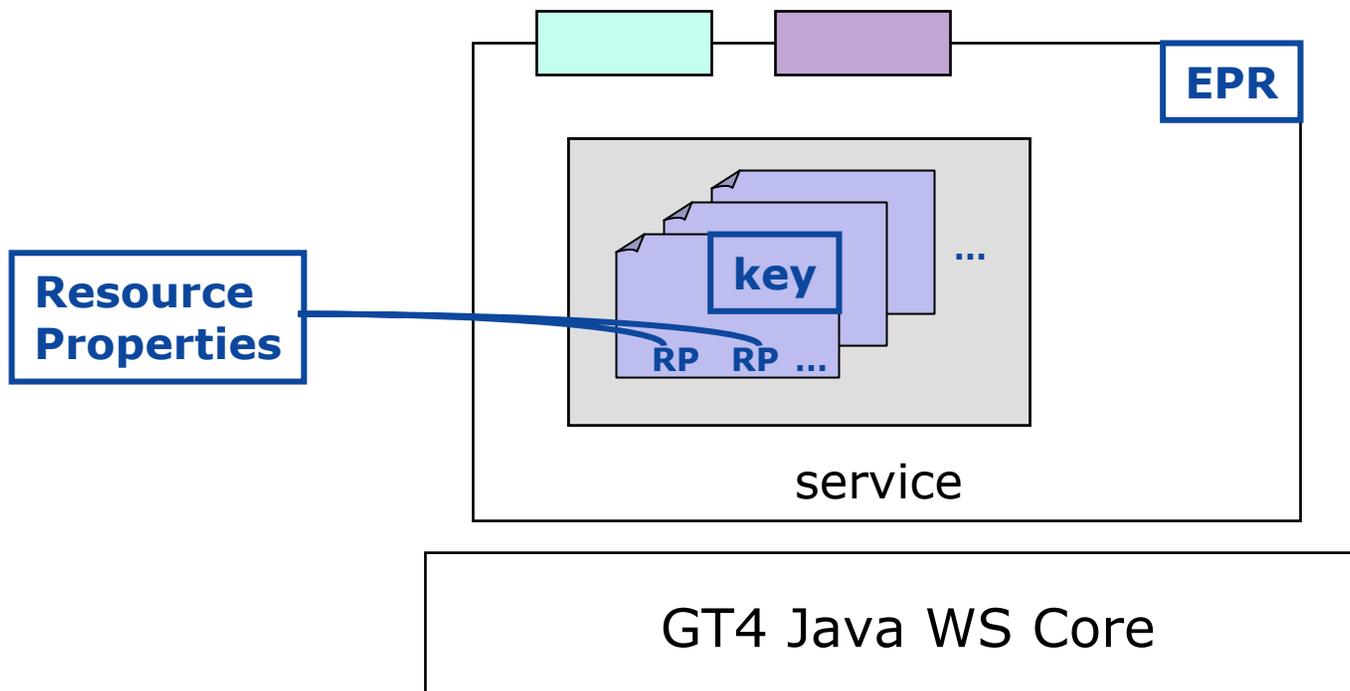
**1. Deployment:
Stand up a StickyNote service**

Concepts

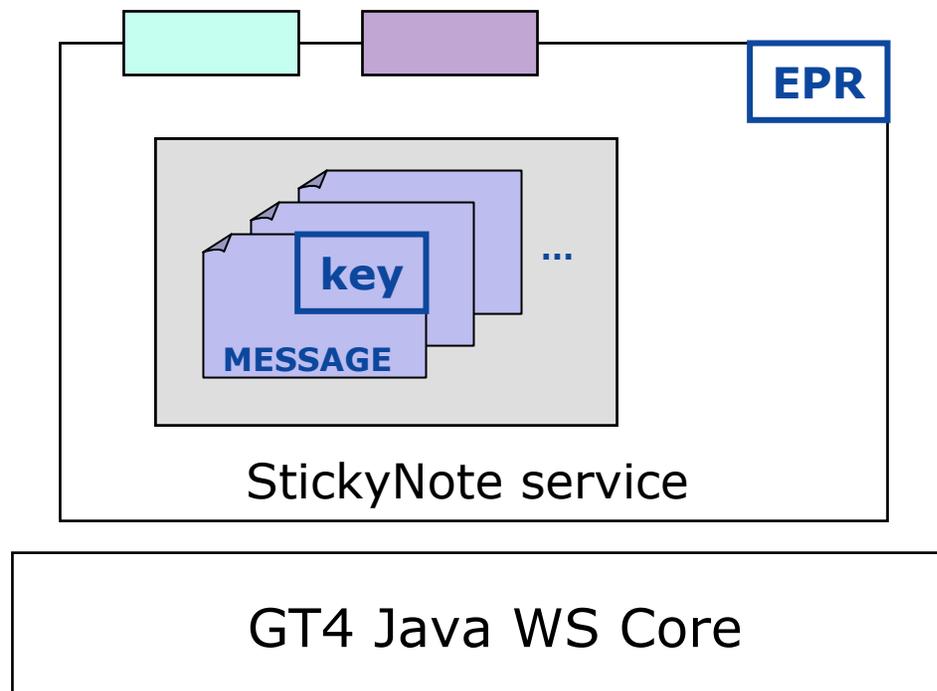
State Management Part II: Resource Properties

- A Resource Property (RP) is an XML element contained within a Resource
- RPs provide a view on the current state of the Resource
 - ◆ For us, note contents
 - ◆ For data transfer, perhaps # of files successfully transferred, bandwidth available, etc.
- Any Resource can expose internal state as Resource Properties

State Management Part II: Resource Properties



StickyNote Resource Properties



Implementation details...

Adding a Resource Property

- Resource Properties are part of the exposed interface of the service/resource pair, so they appear in WSDL
- To add a new RP, we must define it in WSDL, then write code to maintain the value in our implementation
- We already have “message” as an RP
- In this exercise, we’ll add a “last modified time” resource property to the note

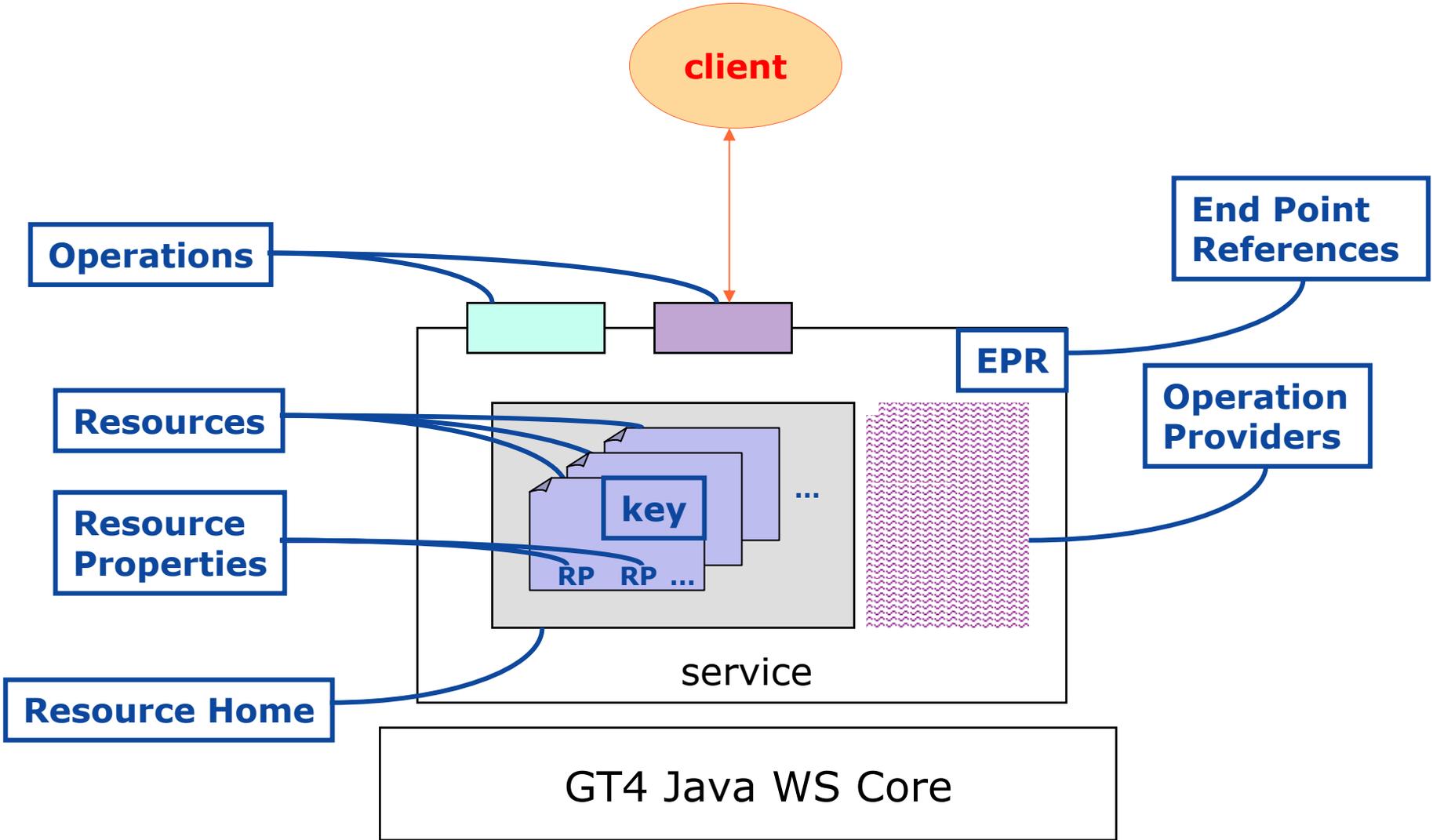
Demonstration

Student Notes, Chapter 4

Exercise 4 Review

- Resource Properties are defined in WSDL
- Resource Properties are XML
- Resources can have as many RPs as they wish
- RPs provide a view on the current state of the Resource
- Any Resource can expose internal state as Resource Properties

What We've Covered So Far



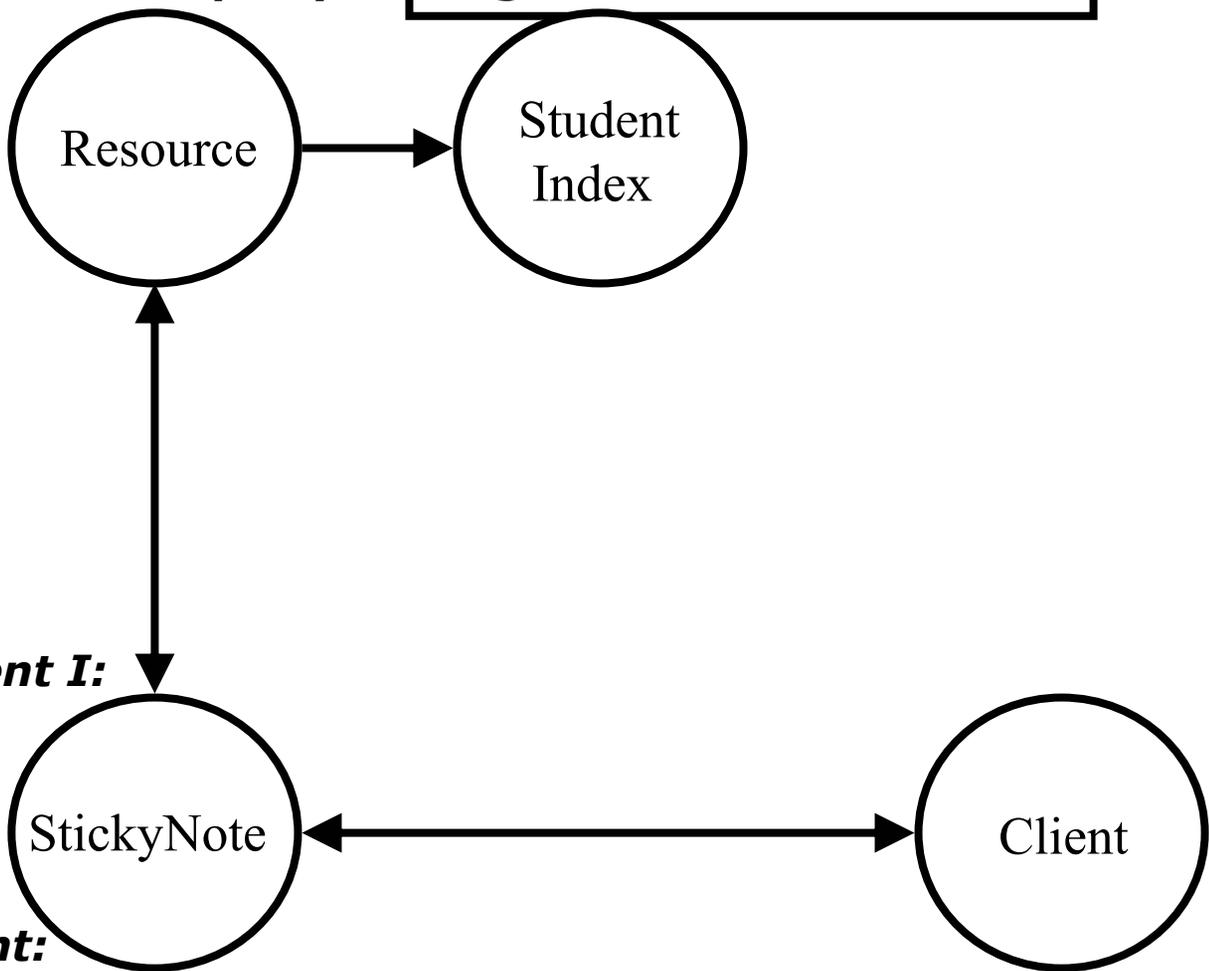
How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ **5. Aggregating Resources: Register with a Local Index**
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

Exercise 5: Aggregating Resources

**4. State Management II:
Add a Resource Property**

**5. Aggregating Resources:
Register with a local index**



**3. Lifetime
Mgmt I:
Destroy
Resources**

**2. State
Management I:
Create
Resources**

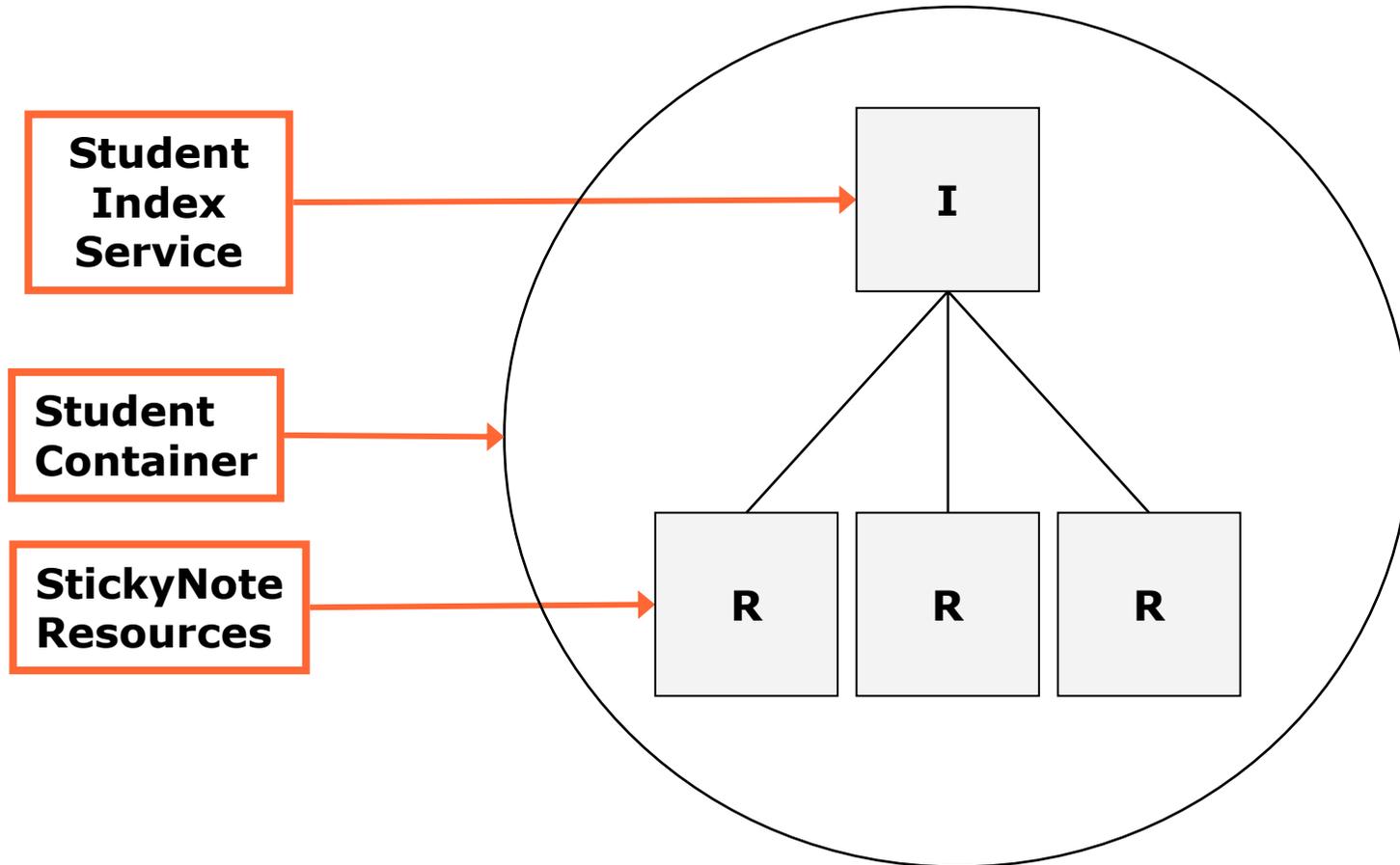
**1. Deployment:
Stand up a StickyNote service**

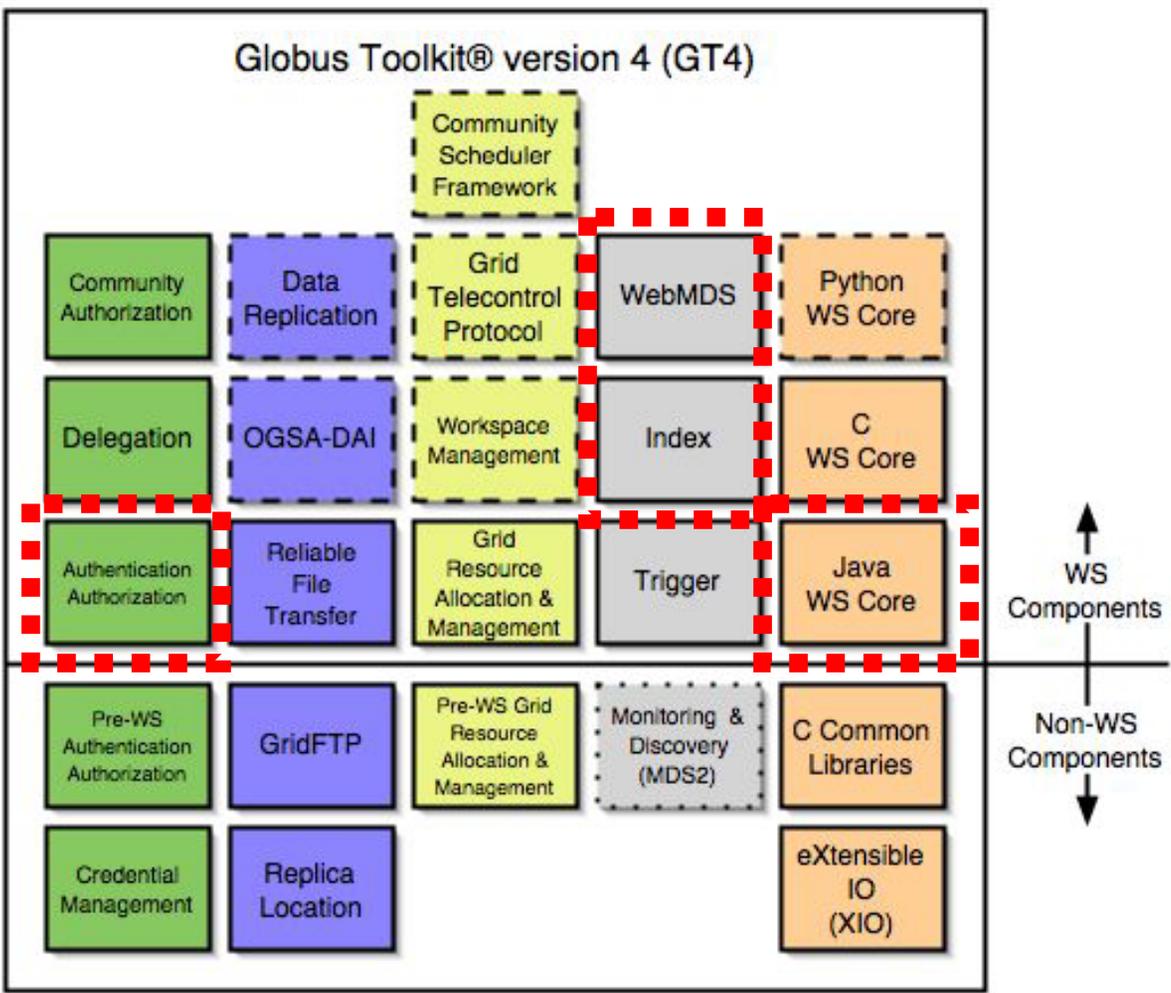
Concepts

Aggregating Resources

- Thus far we have manually kept track of the existence of our individual sticky notes
 - ◆ In real grid deployments this is infeasible
- GT4 includes an Index Service that can be used to maintain a list of available resources
- In this chapter we will register our sticky note resources with a local index service

Registering with a Local Index





- Core GT Component: public interfaces frozen between incremental releases; best effort support
- Contribution/Tech Preview: public interfaces may change between incremental releases
- Deprecated Component: not supported; will be dropped in a future release

Implementation details...

Override Default Behavior of the Resource Home

- When a Resource is created, the “add” method of the ResourceHome will be called
- Override this method to include logic to register the Resource in the local index service

Demonstration

Student Notes, Chapter 5

Exercise 5 Review

- GT4 includes a default Index service
- Local index services are often used to aggregate local Resources of interest
- You may browse the Index with wsrp-query
- Destroying a resource causes it to be removed from the Index

How to Build a Service Using GT4

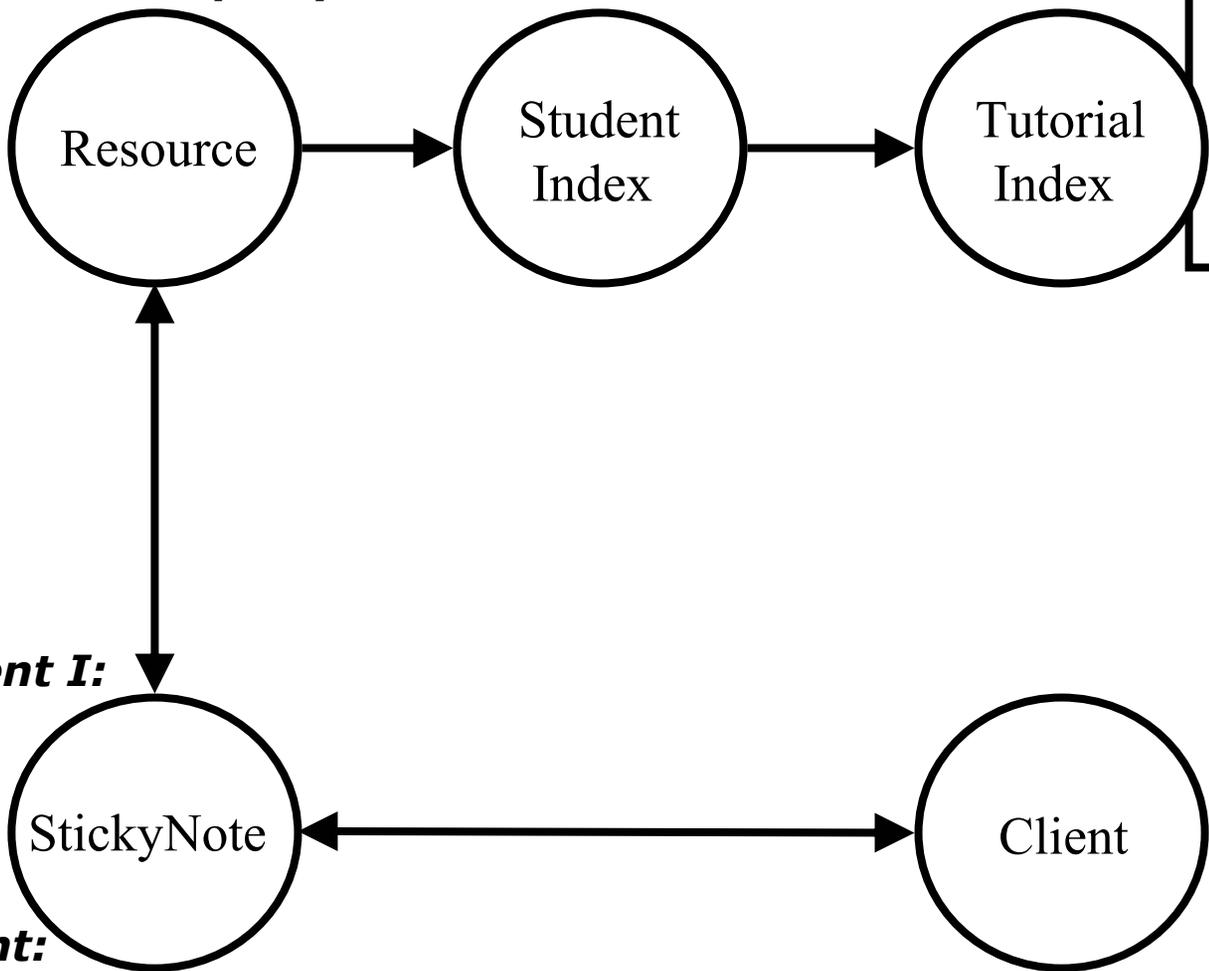
- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ **6. Building a VO: Register with a Community Index**
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

Exercise 6: Virtual Organizations

**4. State Management II:
Add a Resource Property**

**5. Aggregating Resources:
Register with a local index**

**6. Virtual
Organization:
Register with
a community
index**



**3. Lifetime
Mgmt I:
Destroy
Resources**

**2. State
Management I:
Create
Resources**

**1. Deployment:
Stand up a StickyNote service**

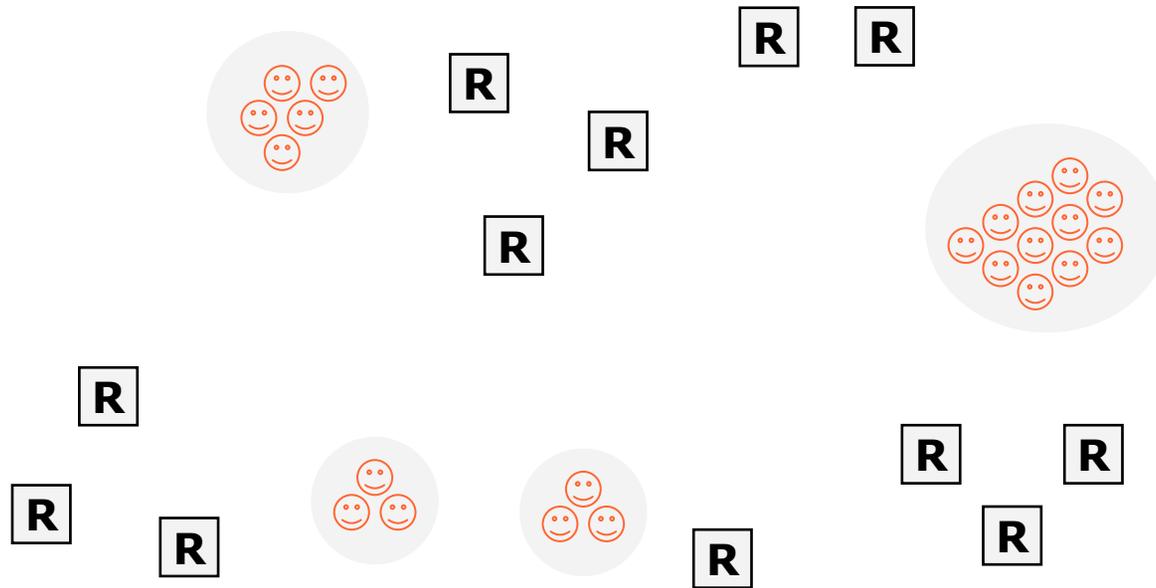
Concepts

Virtual Organizations

- There is a structure that underlies many Grid problem spaces
- The structure is referred to as a Virtual Organization
- Virtual Organizations drive many of the requirements for Grid technology

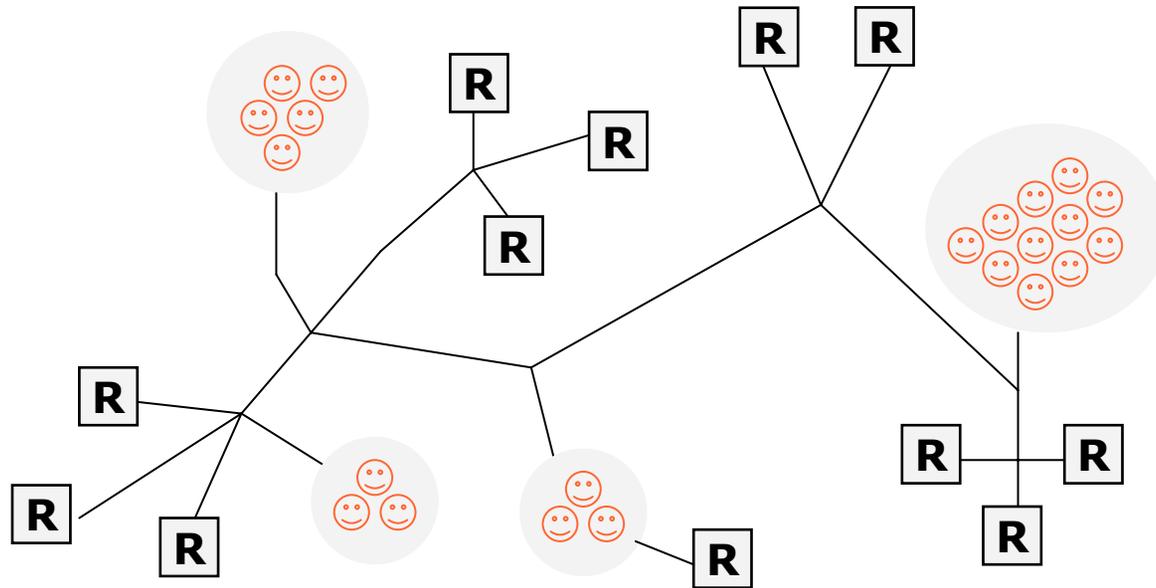
Virtual Organizations

- Distributed resources and people



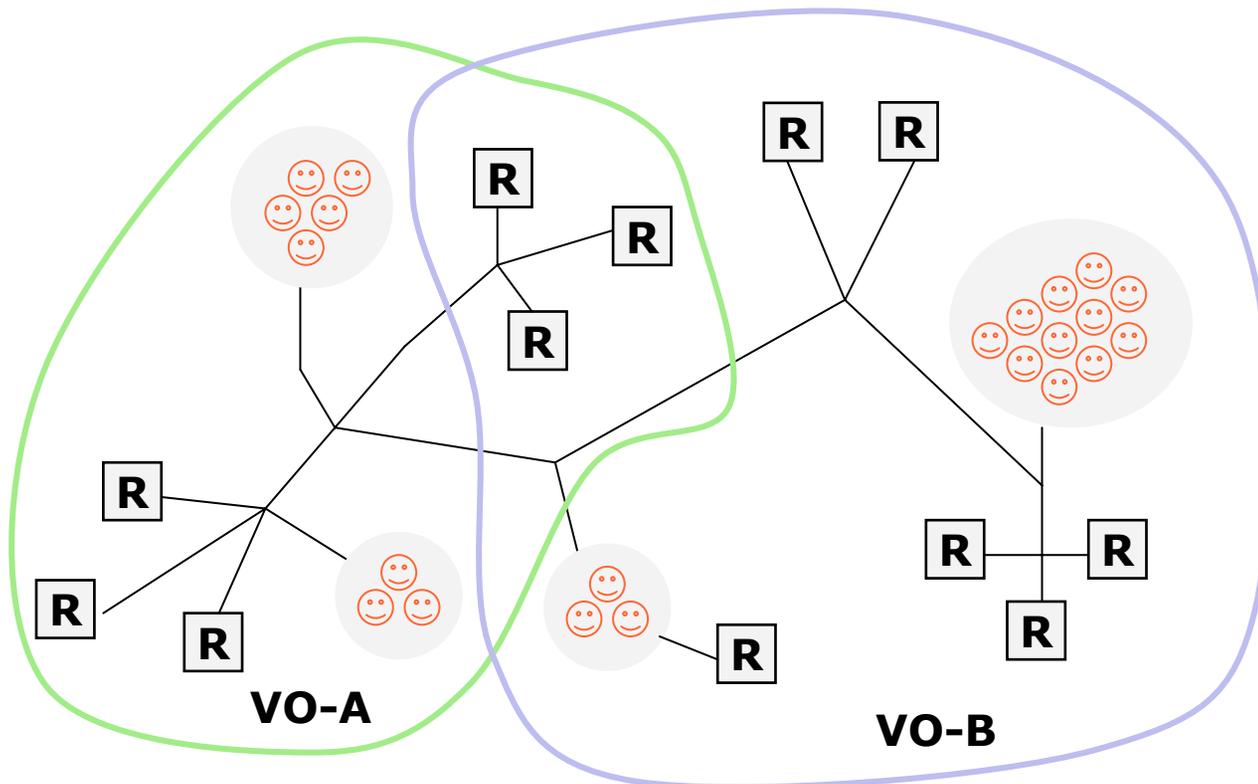
Virtual Organizations

- Distributed resources and people
- Linked by networks, crossing administrative domains



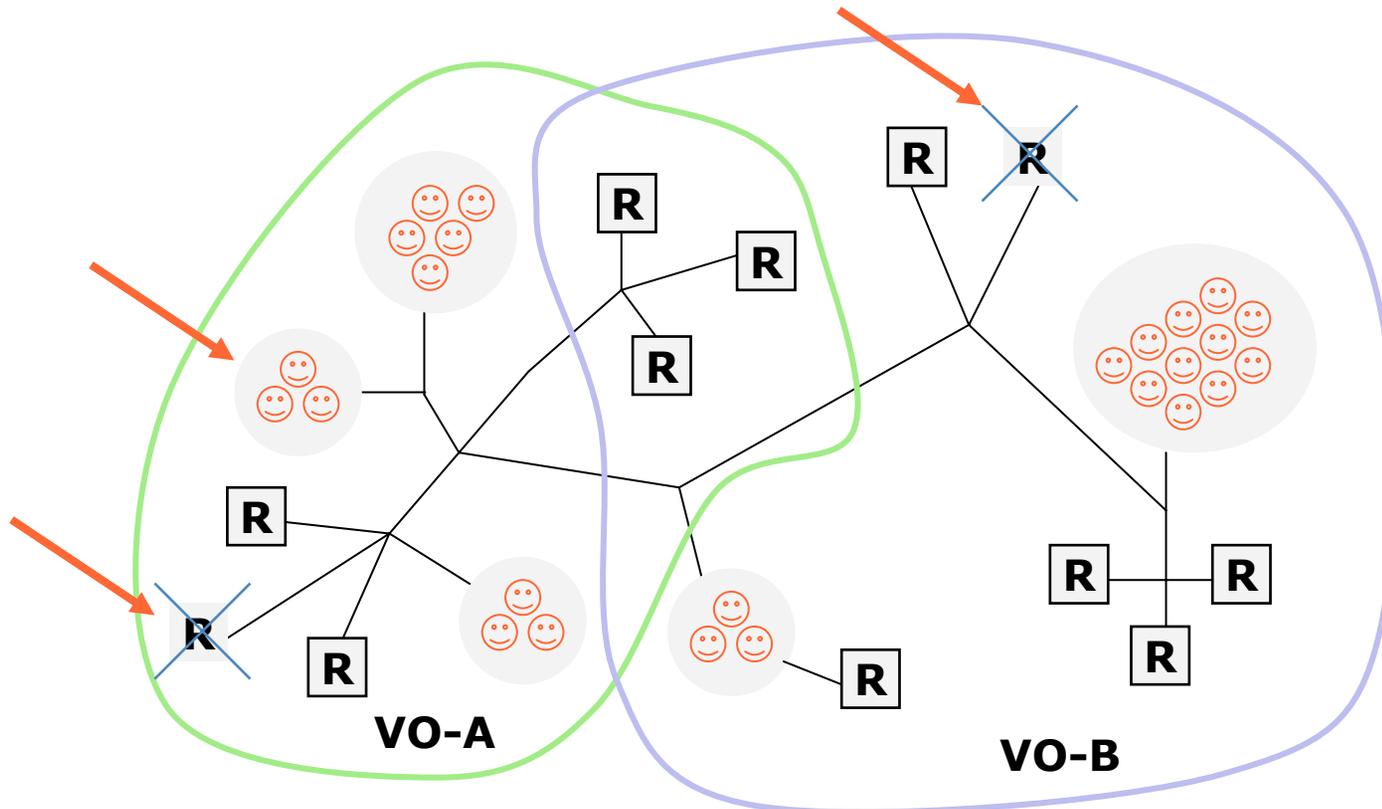
Virtual Organizations

- Distributed resources and people
- Linked by networks, crossing administrative domains
- Sharing resources, common goals



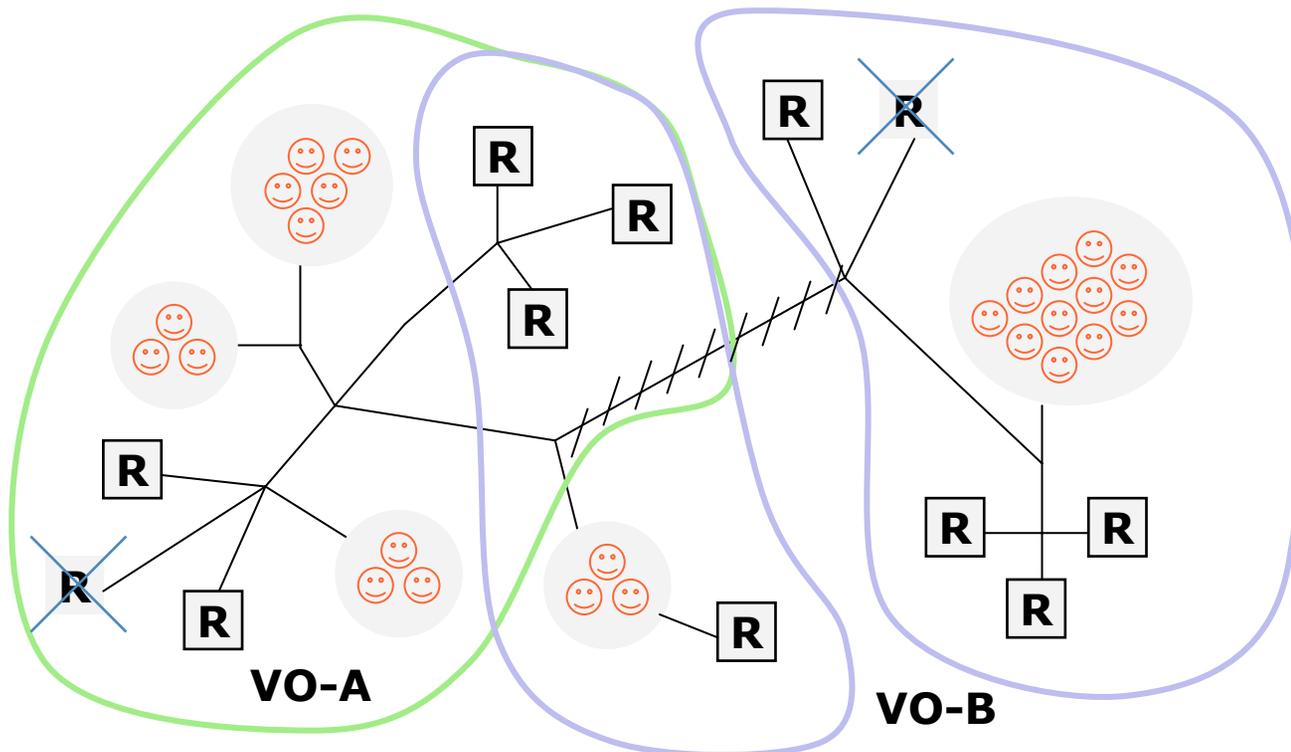
Virtual Organizations

- Distributed resources and people
- Linked by networks, crossing administrative domains
- Sharing resources, common goals
- Dynamic

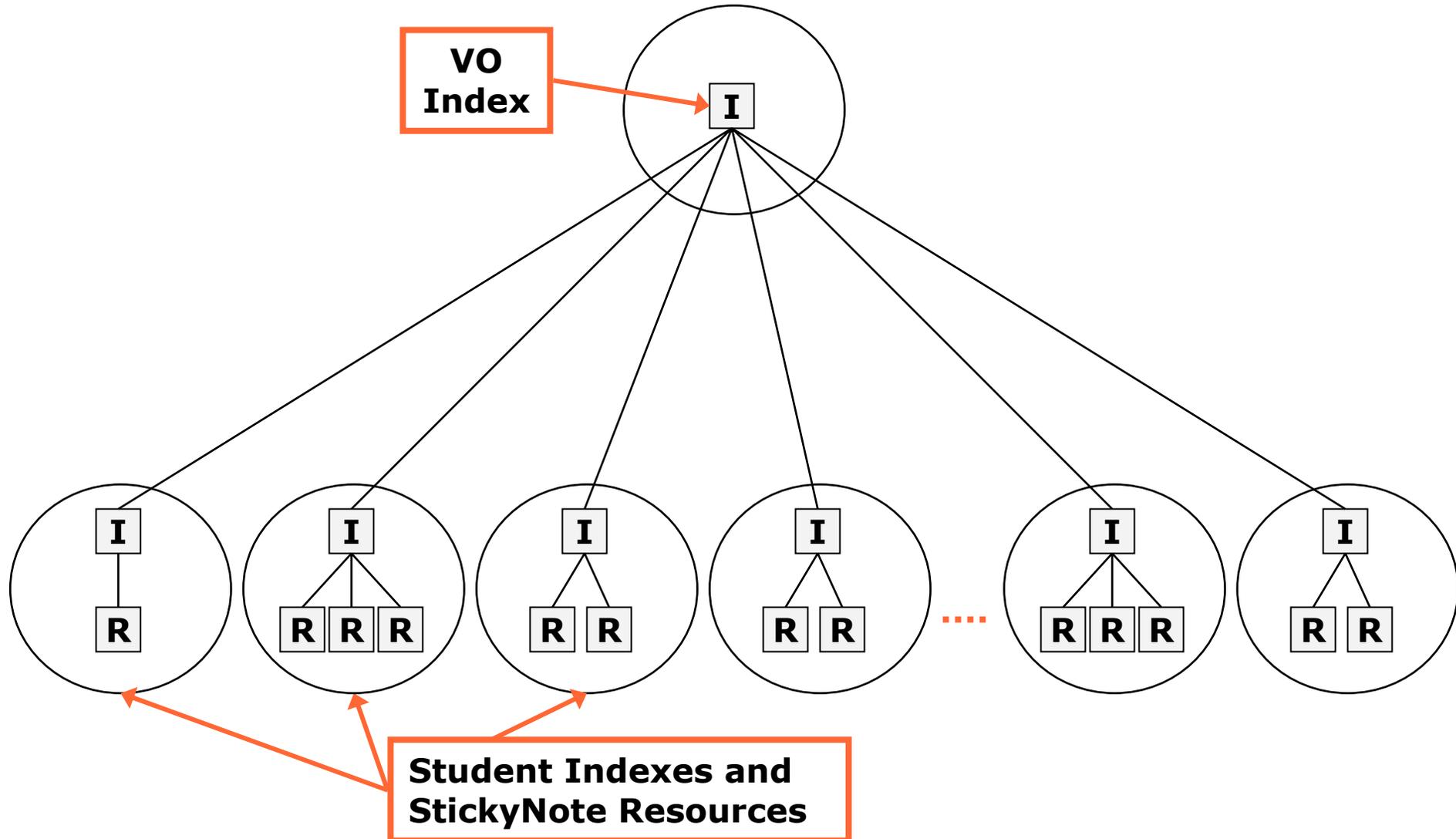


Virtual Organizations

- Distributed resources and people
- Linked by networks, crossing administrative domains
- Sharing resources, common goals
- Dynamic
- Fault tolerant



Tutorial VO



The VO Index

- The community Index will be running on one of the instructor's machines
- The Index represents a collection of Resources
 - ◆ A way for each StickyNote Resource to advertise its existence to the VO
 - ◆ A copy of each note's Resource Properties will be cached in a single place

Implementation details...

Registering to the Index

- We configure the local Index to register with the VO Index
- Because our local index is registered to the VO Index, the local information will propagate

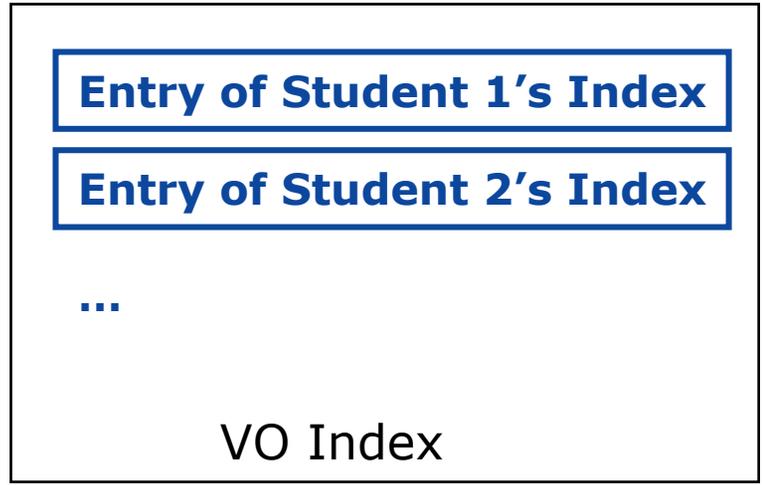
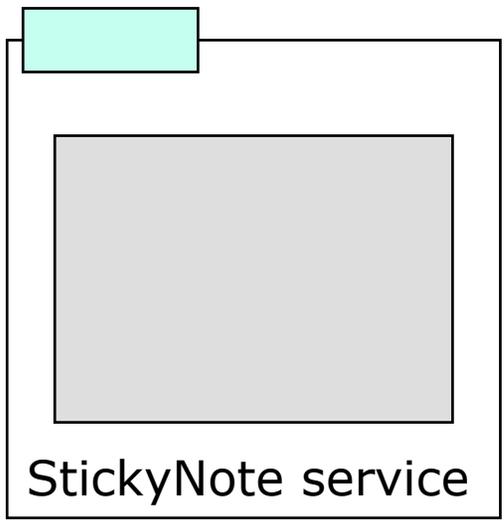
Resource Creation Runtime Revisited

VO Index

Instructors' GT4
Java WS Core

Resource Creation Runtime Revisited

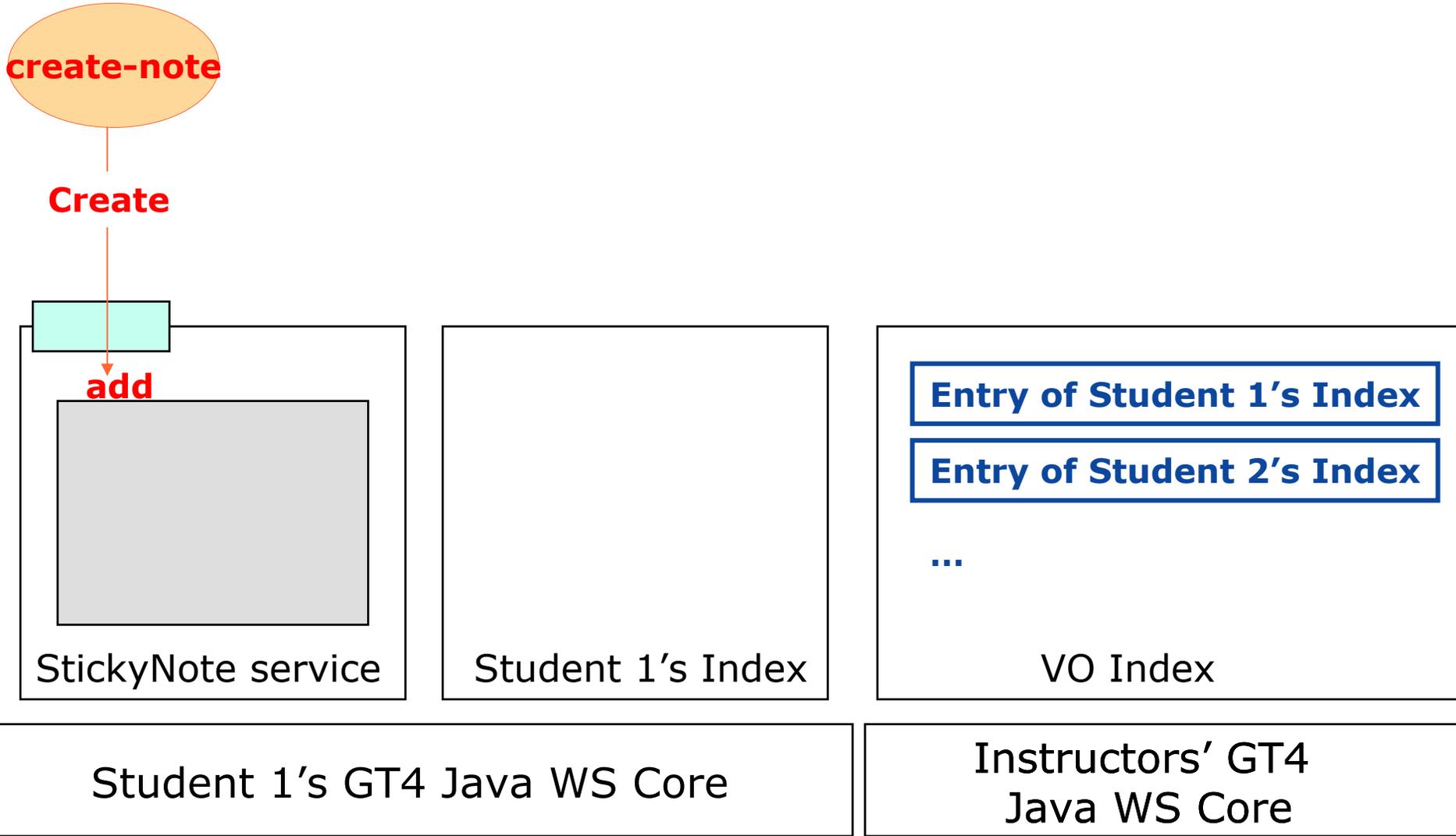
create-note



Student 1's GT4 Java WS Core

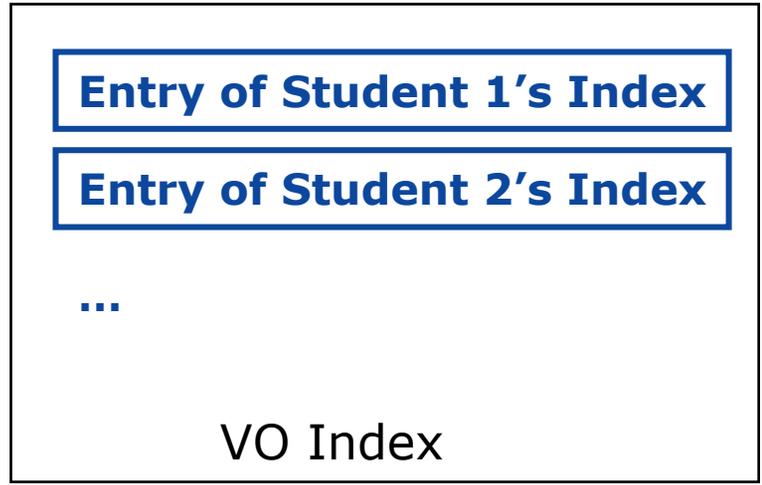
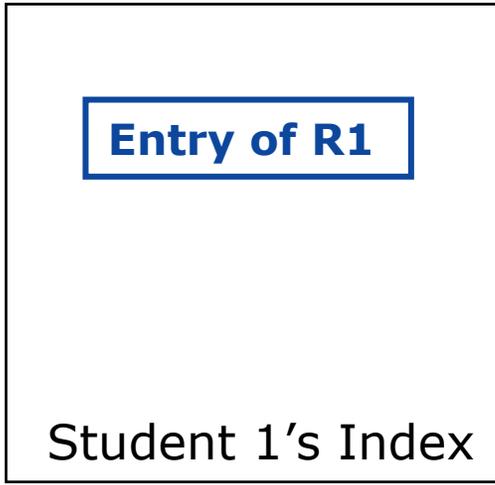
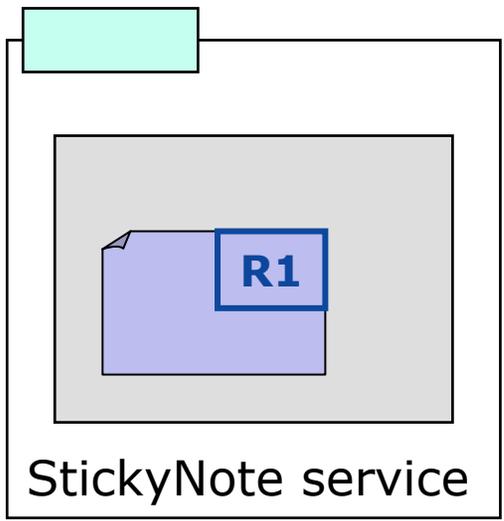
Instructors' GT4 Java WS Core

Resource Creation Runtime Revisited



Resource Creation Runtime Revisited

create-note

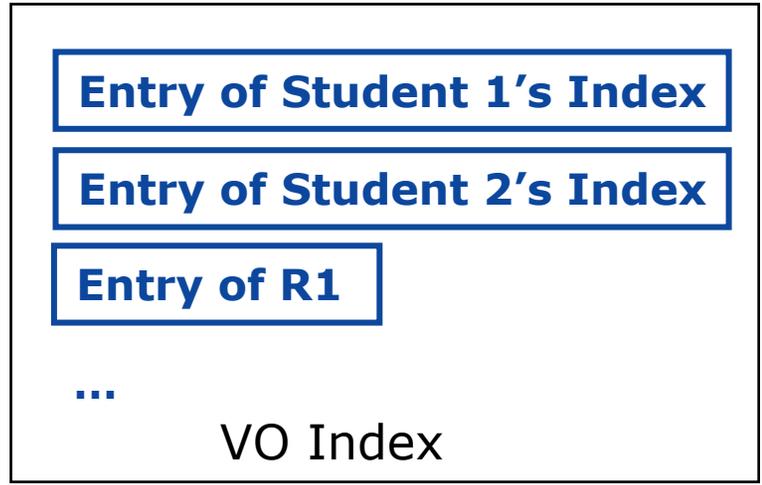
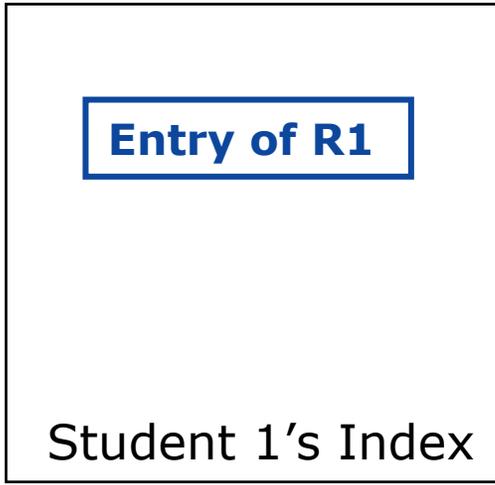
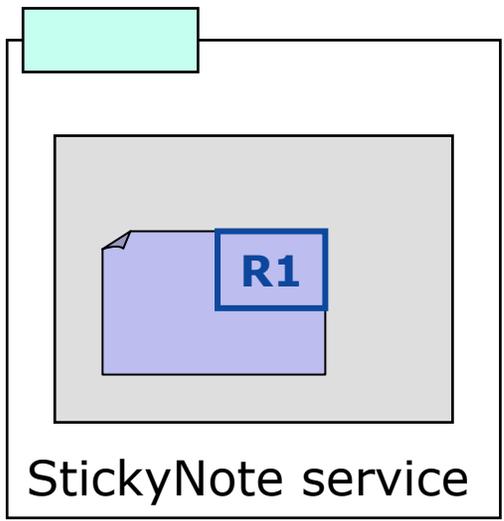


Student 1's GT4 Java WS Core

Instructors' GT4
Java WS Core

Resource Creation Runtime Revisited

create-note



Student 1's GT4 Java WS Core

Instructors' GT4 Java WS Core

Demonstration

Student Notes, Chapter 6

Exercise 6 Review

- The VO Index contains the aggregate contents of student Index services
- You may browse the Index with wsrp-query or WebMDS or the fishtank
- Destroying a resource causes it to be removed from the Index
- Virtual Organizations make the world go round

How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ **7. Lifetime Management Part II: Lease-based Model**
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

Exercise 7: Leased-Based Lifetime Mgmt

**4. State Management II:
Add a Resource Property**

**5. Aggregating Resources:
Register with a local index**

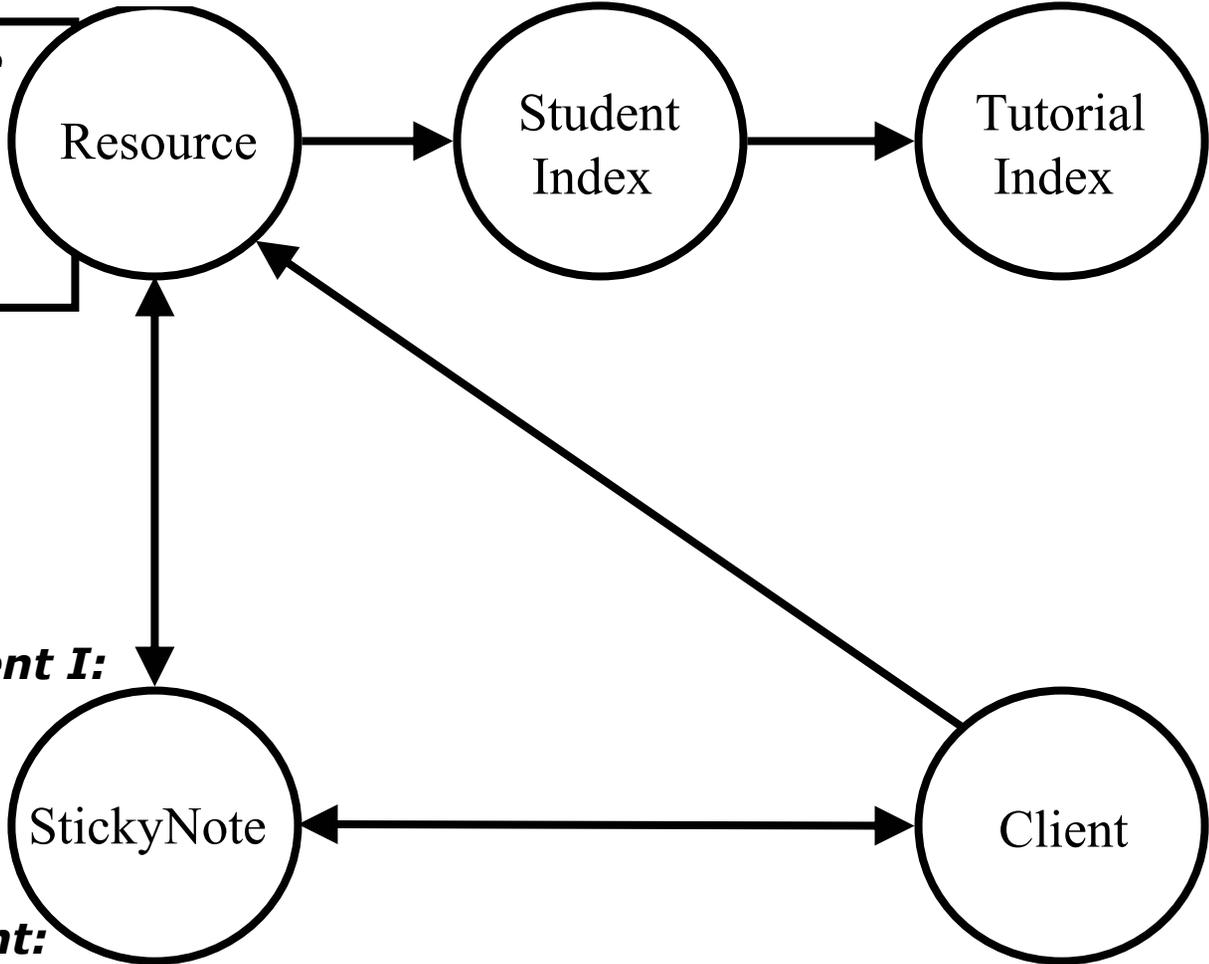
**6. Virtual
Organization:
Register with
a community
index**

**7. Lifetime
Mgmt II:
Lease-
Based
Model**

**3. Lifetime
Mgmt I:
Destroy
Resources**

**2. State
Management I:
Create
Resources**

**1. Deployment:
Stand up a StickyNote service**



Concepts

Lease-Based Lifetime Management

- In Grid computing, we need a mechanism to clean up old/unwanted state
- So far we've cleaned up after ourselves explicitly using the Destroy operation
- Manually destroying our own notes is fine in some contexts, but is not always possible or appropriate
 - ◆ If a client dies or the network goes down, you would like a mechanism to clean up unused state automatically
- Under the lease-based model resources must be kept alive by interested parties

Lease-Based Lifetime Management

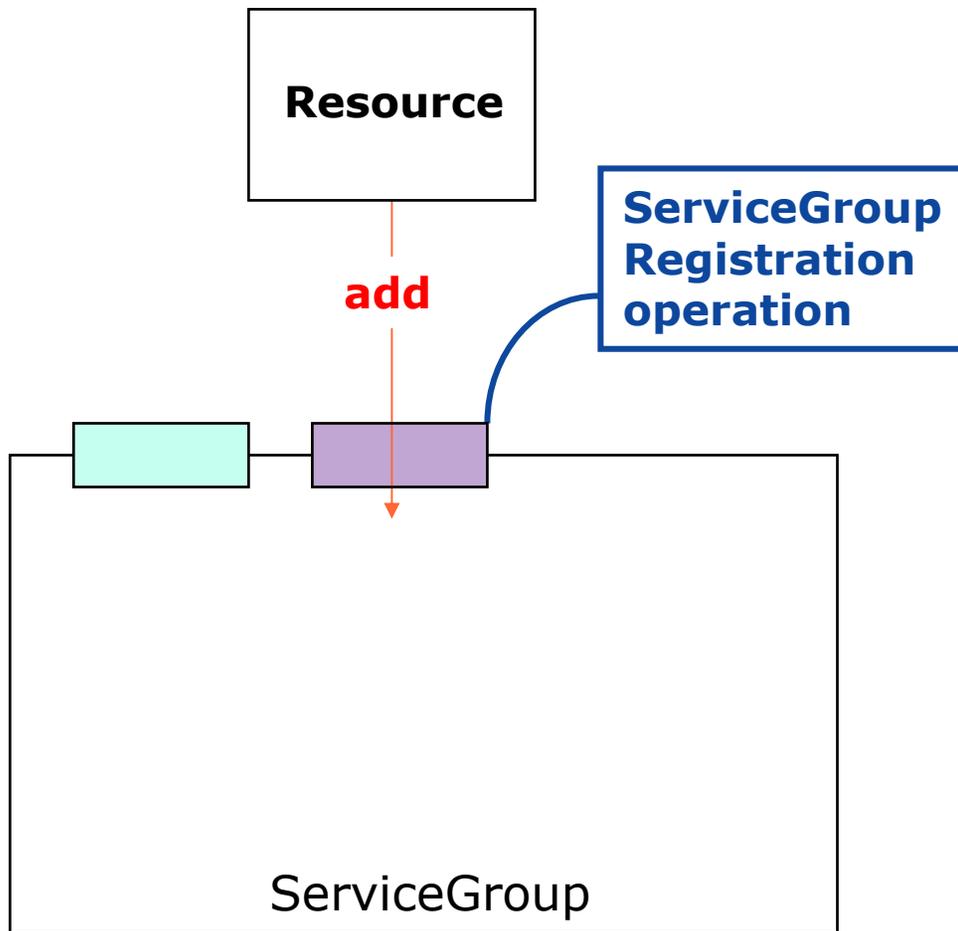
- To illustrate this, we will support scheduled destruction of notes
- If the lease expires, the resource will be destroyed
- Interested parties can renew the lease
 - ◆ `wsrp-set-termination-time`
- The termination time is exposed as a Resource Property

Implementation details...

Service Group

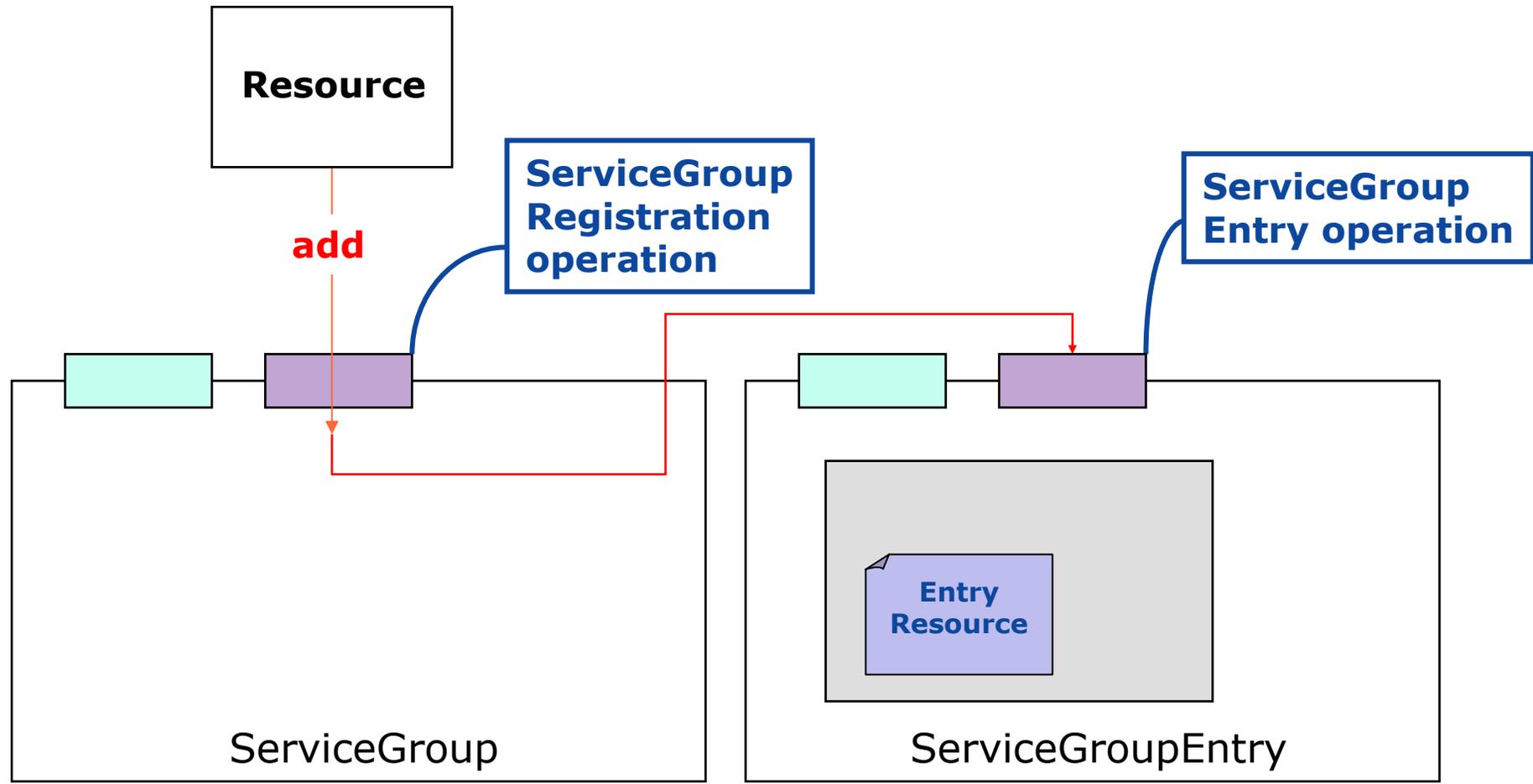
- Indexes are a kind of **service group**
- Service groups represent collections of services and resources
- Each registration is represented as a ServiceGroupEntry resource
- Each entry resource has a lifetime, which will be renewed automatically by the StickyNote service

Service Group Structure



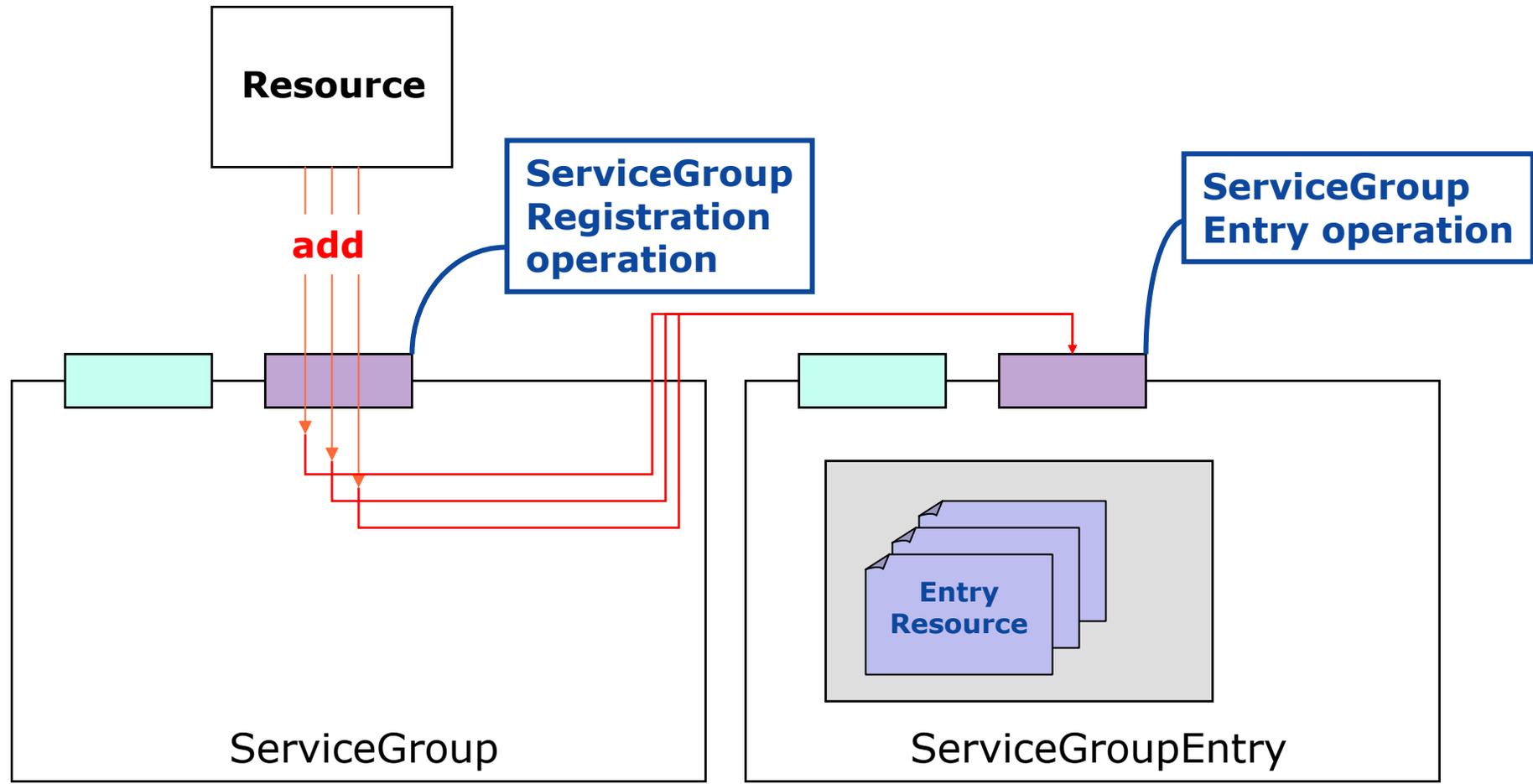
GT4 Java WS Core

Service Group Structure



GT4 Java WS Core

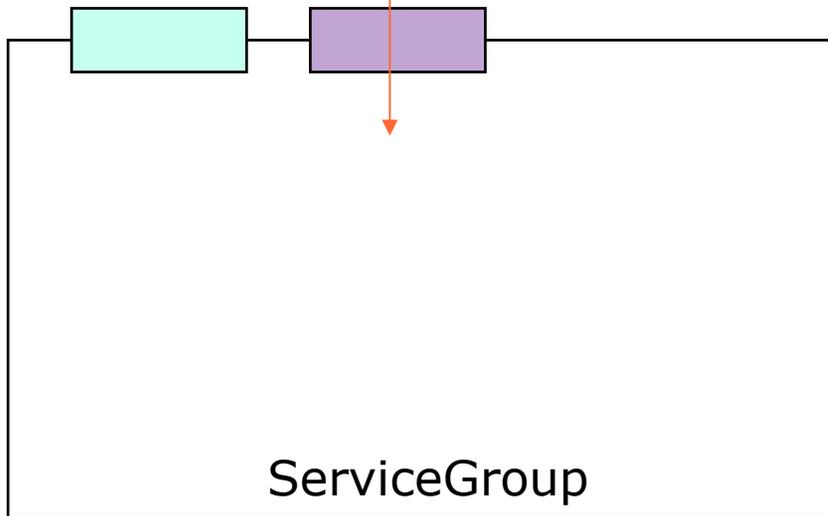
Service Group Structure



Service Group Lifetime Management Runtime



add

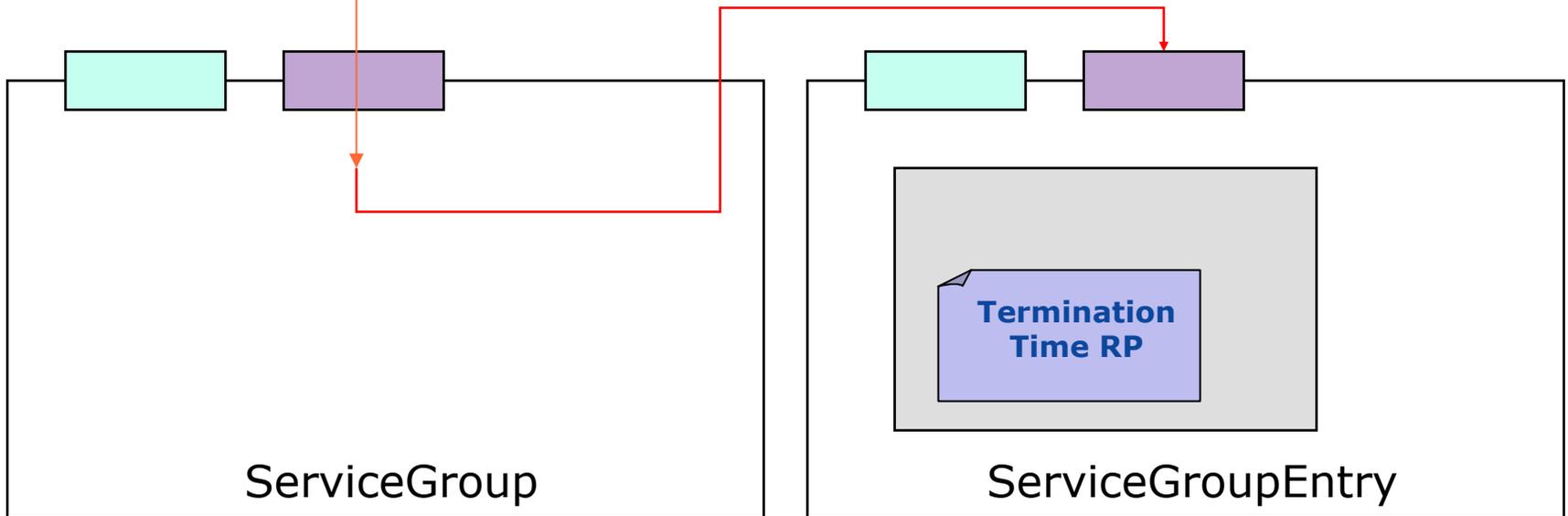


GT4 Java WS Core

Service Group Lifetime Management Runtime

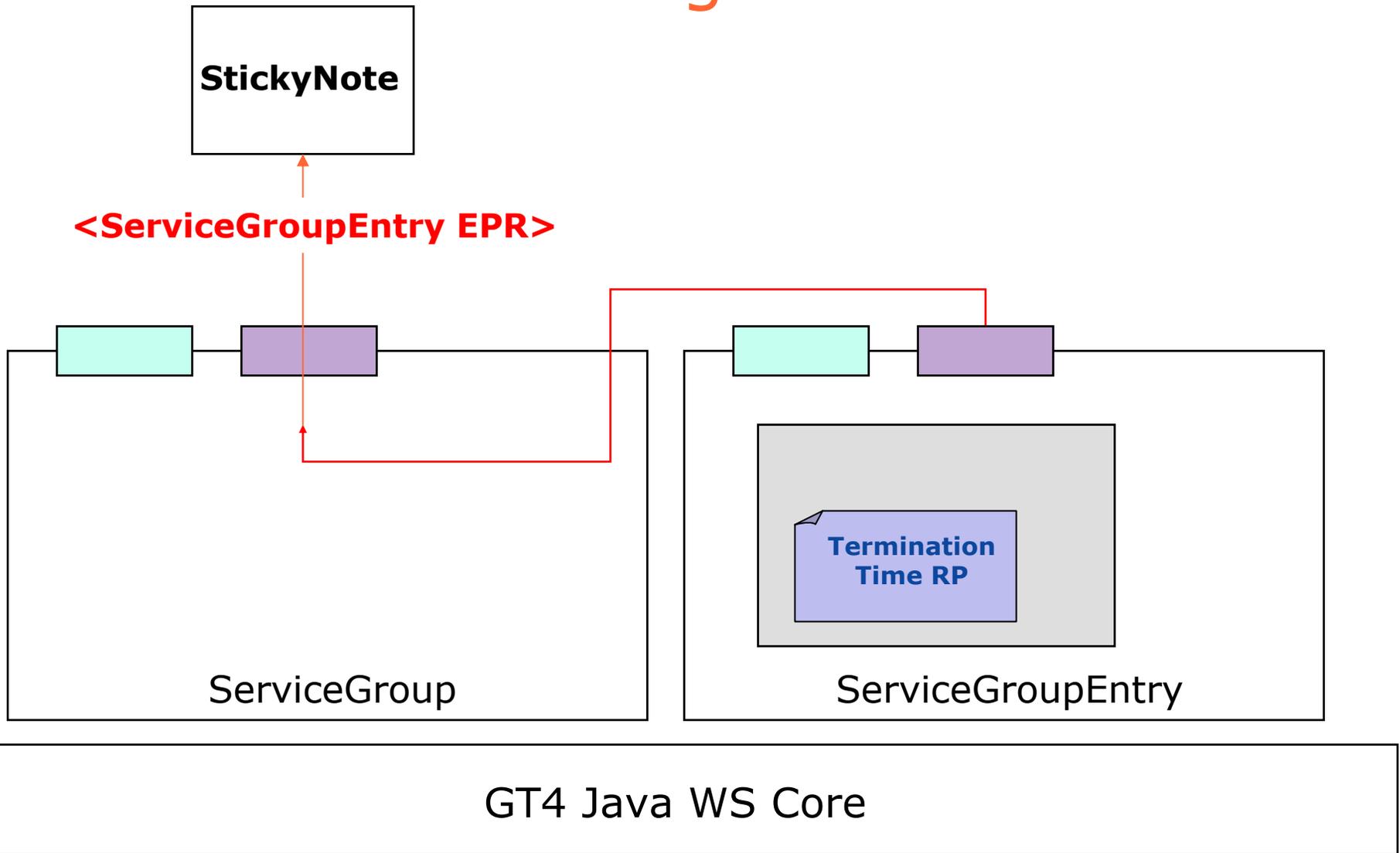


add

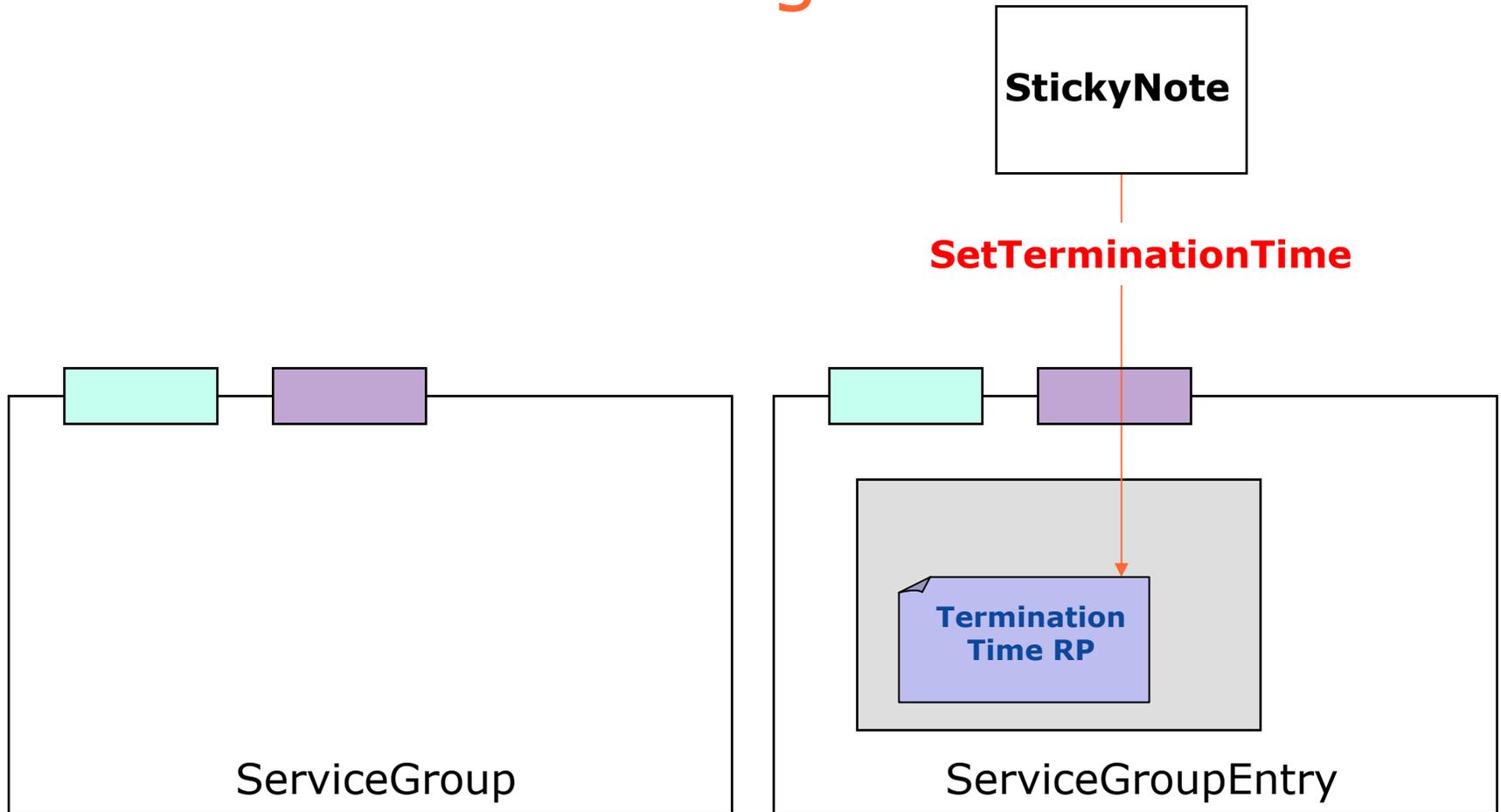


GT4 Java WS Core

Service Group Lifetime Management Runtime



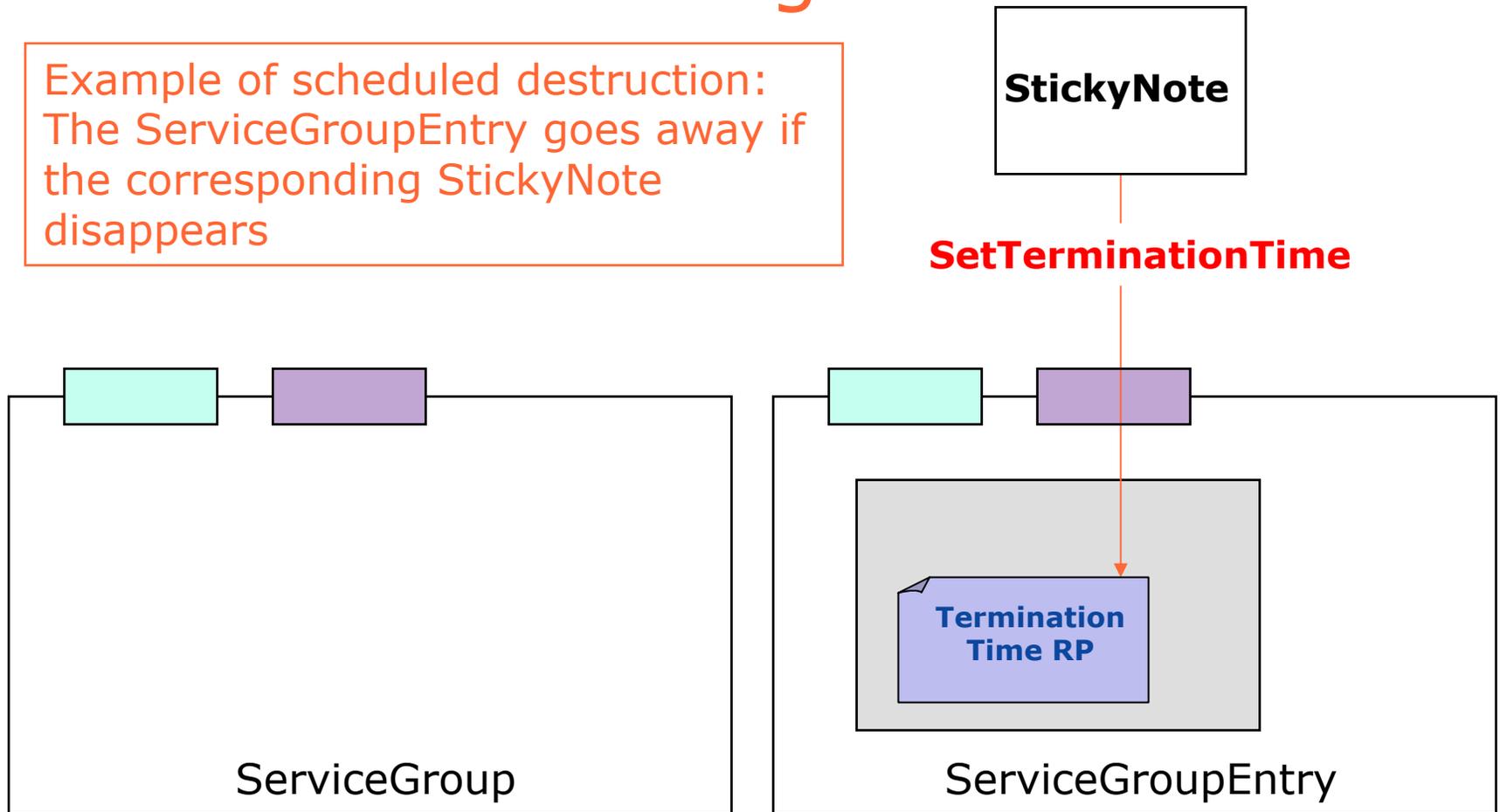
Service Group Lifetime Management Runtime



GT4 Java WS Core

Service Group Lifetime Management Runtime

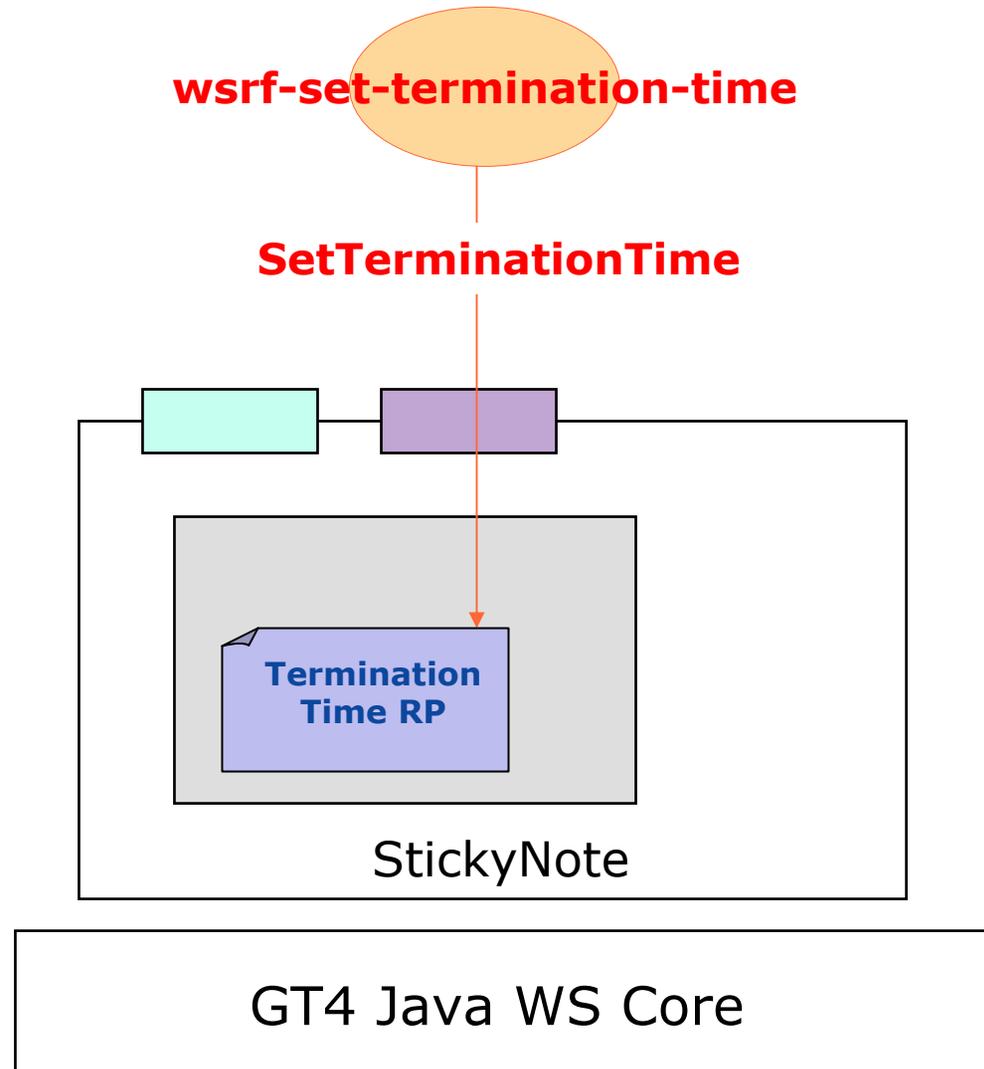
Example of scheduled destruction:
The ServiceGroupEntry goes away if
the corresponding StickyNote
disappears



GT4 Java WS Core

StickyNote

Lifetime Management Runtime



Demonstration

Student Notes, Chapter 7

Exercise 7 Review

- Resources can have a termination time
- Lifetime can be changed

- Service Groups contain resources
- Registrations are managed using standard lifetime mechanisms

How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ **8. Notification: Resource as Notification Producer**
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

Exercise 8: Notification

4. State Management II:
Add a Resource Property

5. Aggregating Resources:
Register with a local index

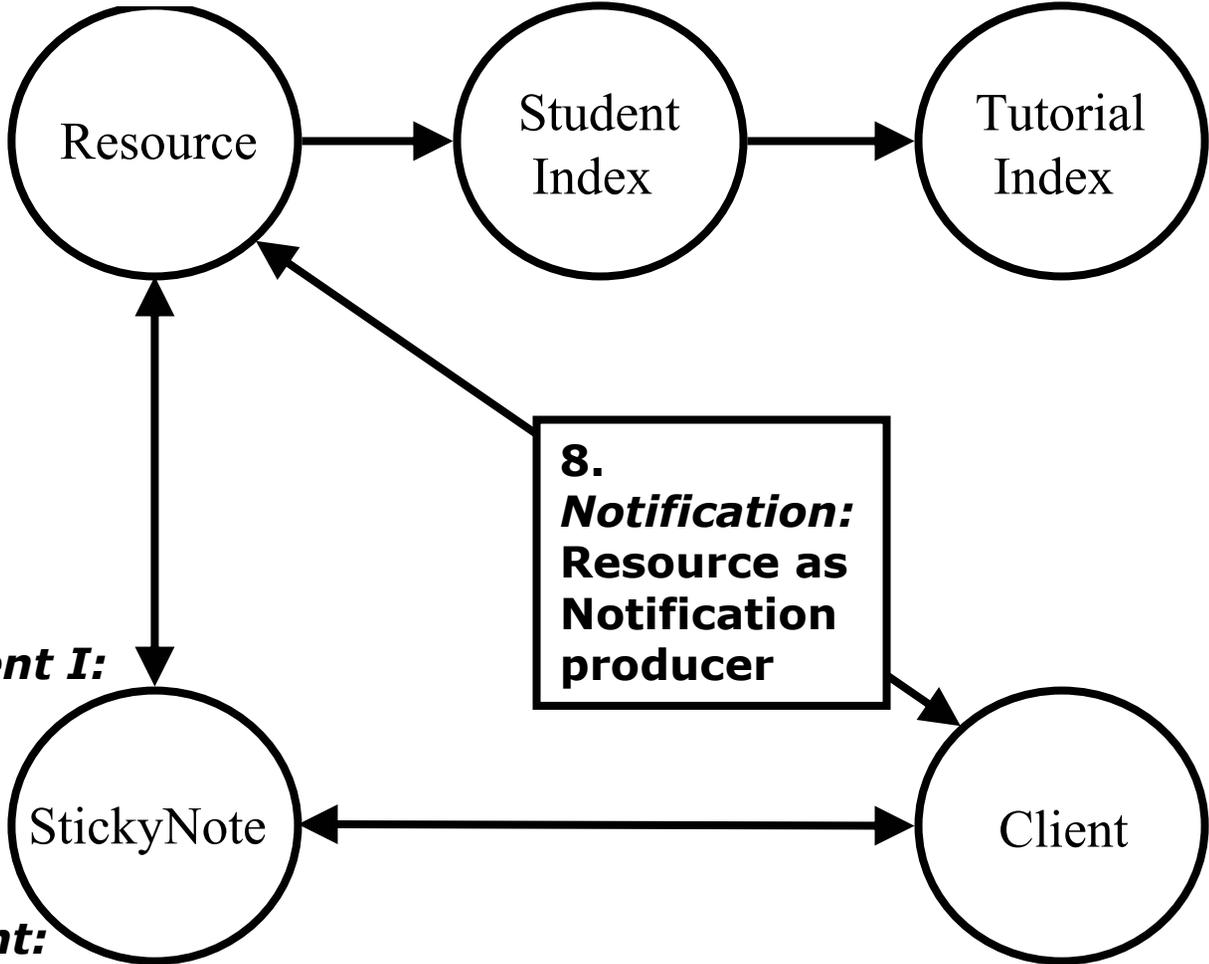
6. Virtual Organization:
Register with a community index

7. Lifetime Mgmt II:
Lease-Based Model

3. Lifetime Mgmt I:
Destroy Resources

2. State Management I:
Create Resources

1. Deployment:
Stand up a StickyNote service



Concepts

Notifications

- Right now make a one-off query for RP values
 - ◆ Show-note command
- We can use notifications to receive new RP values when the RP changes
- Notifications are from one hosting environment to another

Notification vocabulary

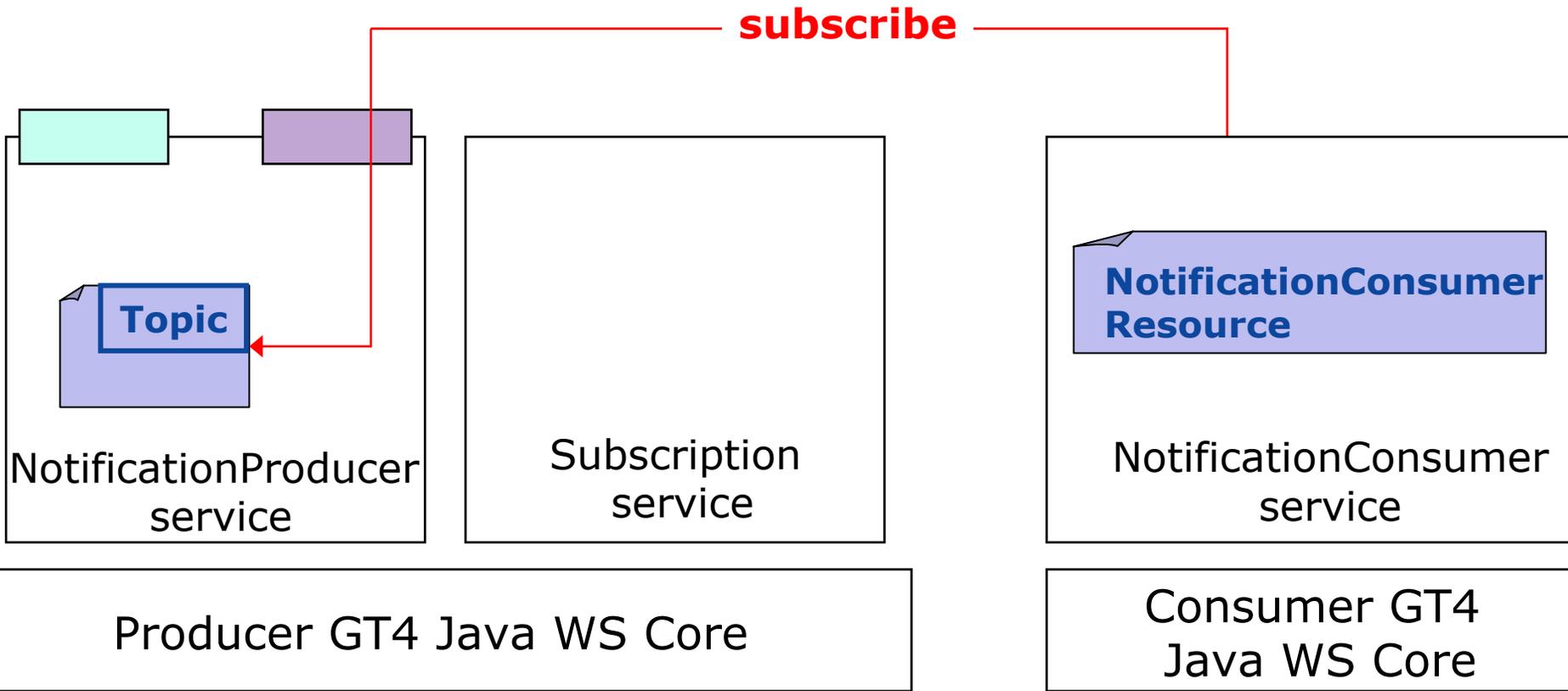
- We can **subscribe** to receive **notifications**
- Subscriptions are for a particular **topic**
- Notifications go from a **Producer** to a **Consumer**
- Each topic has a name
- Topics can be:
 - ◆ Changes in RP values
 - ◆ Resource destruction
 - ◆ Service group changes
 - ◆ Other interesting things...

Subscriptions

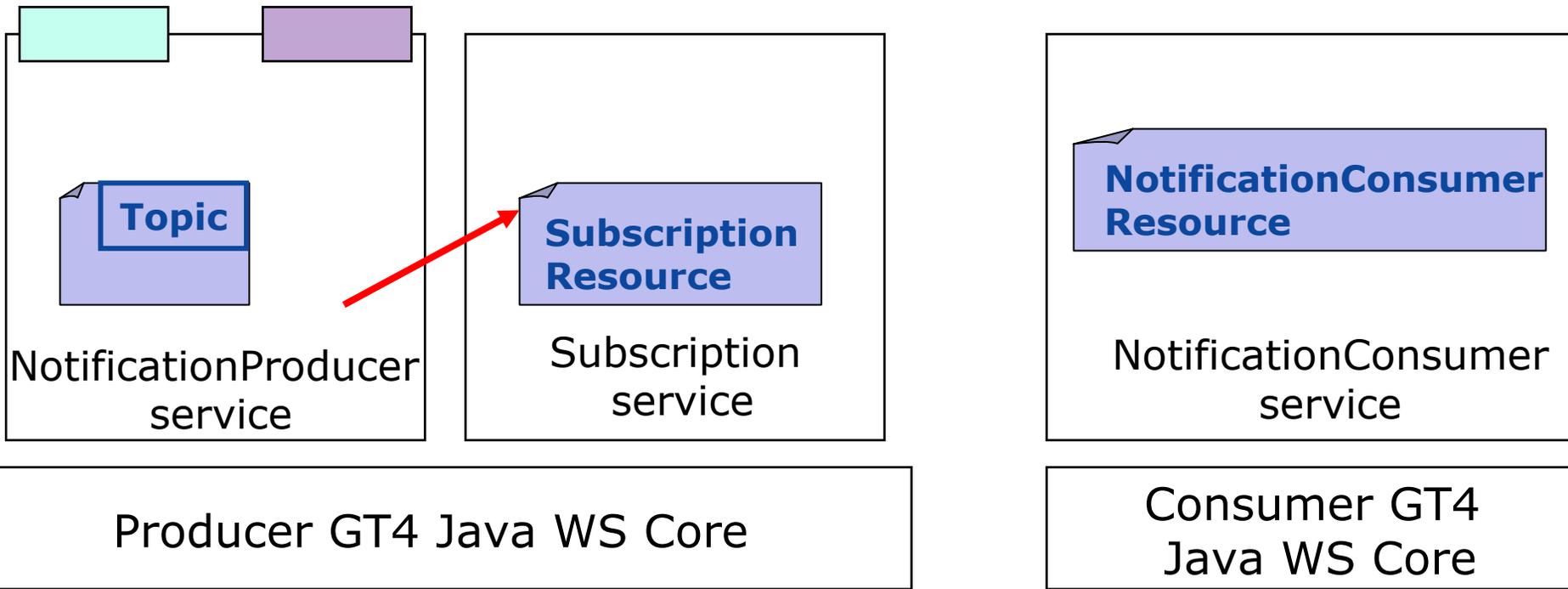
- Each subscription is represented as a new kind of resource
- We can manage the subscription through this resource
 - ◆ Eg. cancel subscription by destroying resource

Implementation details...

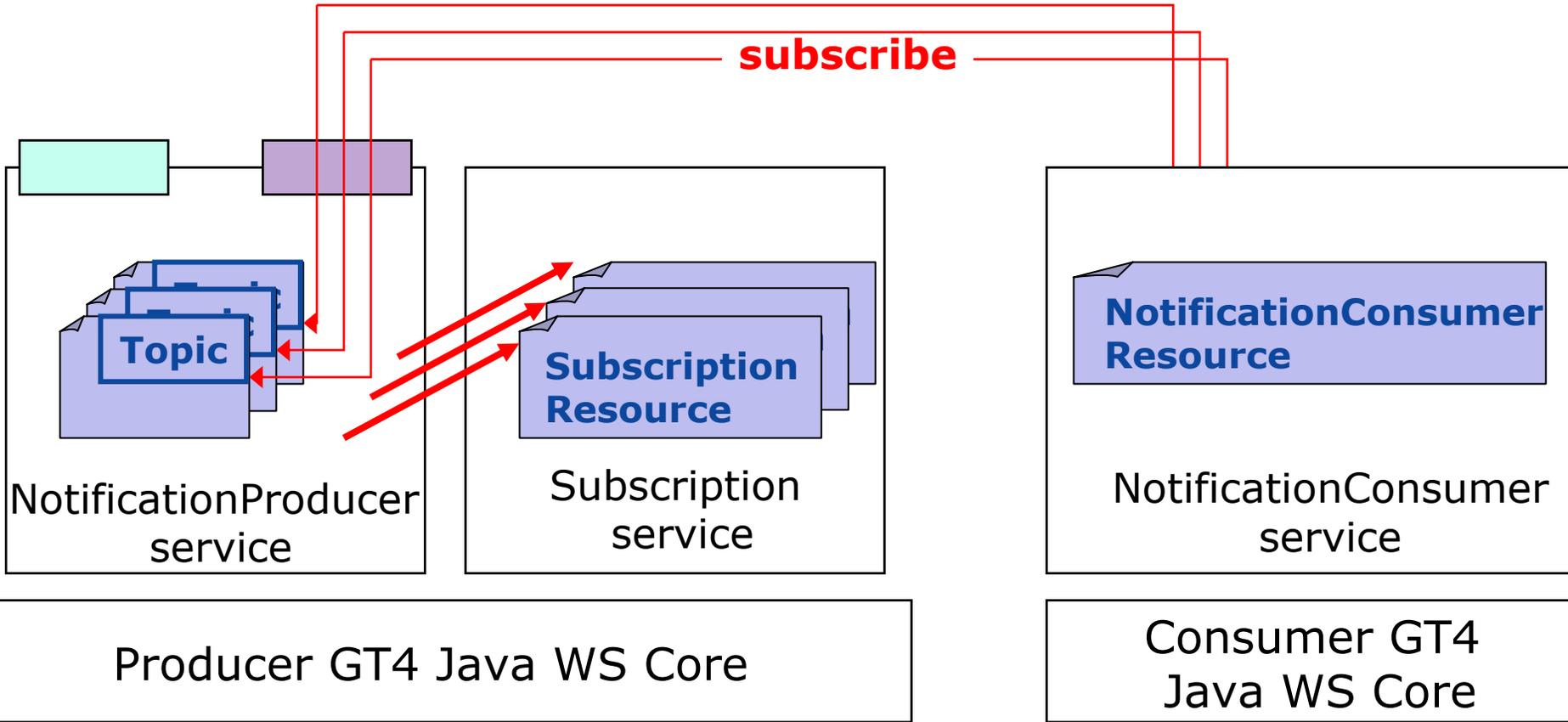
Subscription and Notification



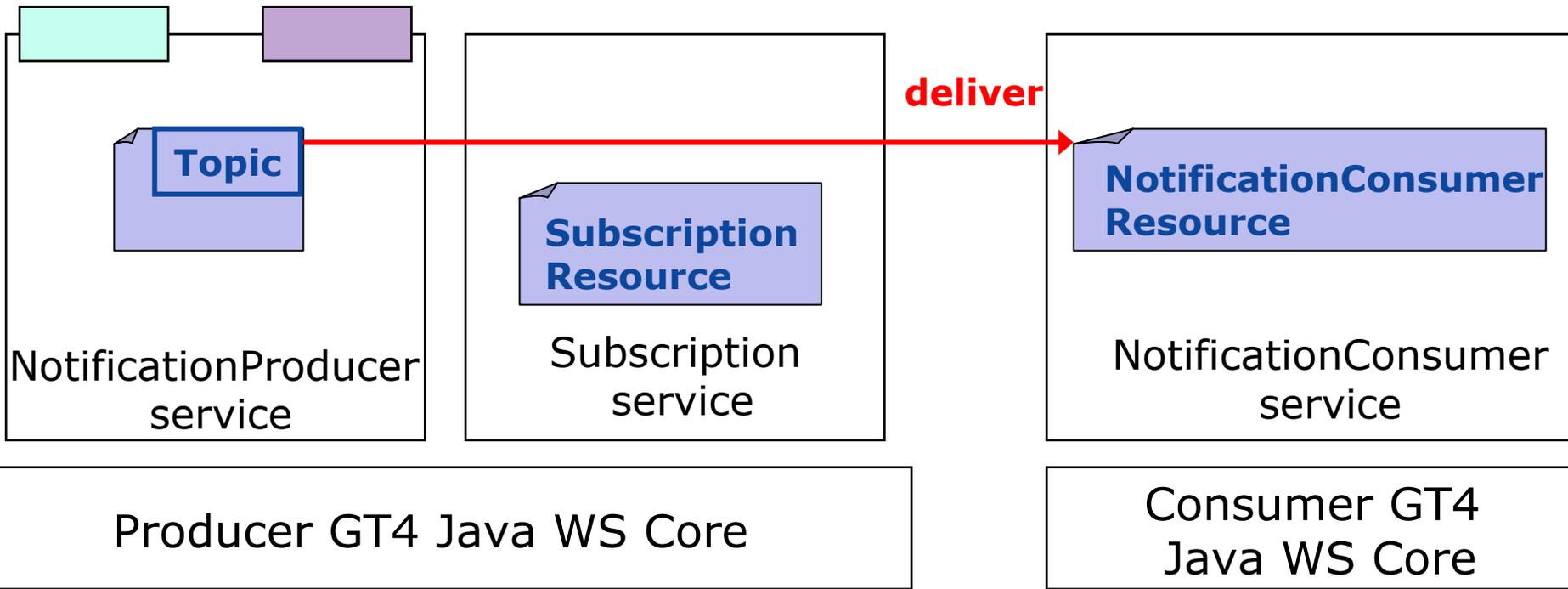
Subscription and Notification



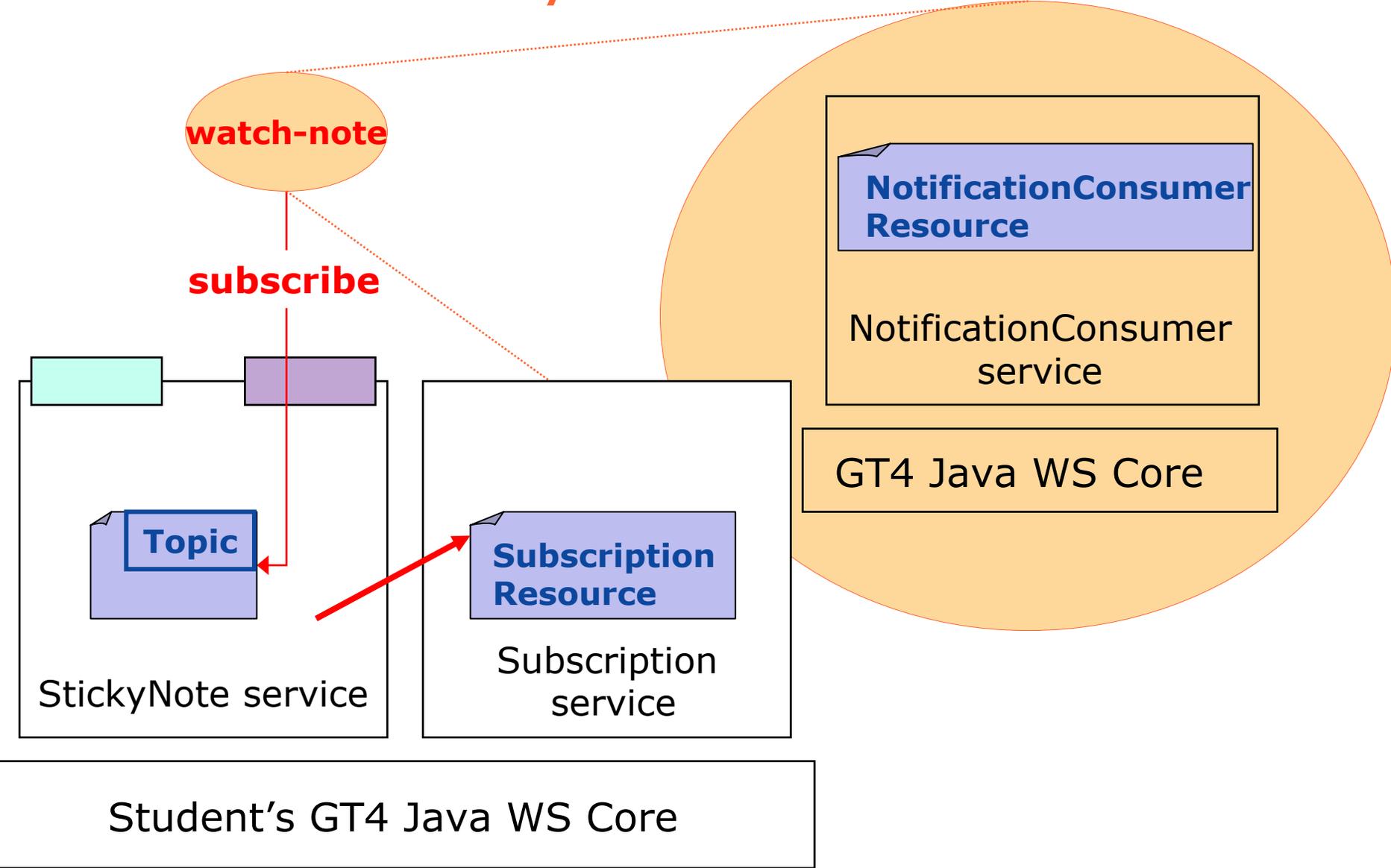
Subscription and Notification



Subscription and Notification



Sticky note notification



Demonstration

Student Notes, Chapter 8

Exercise 8 Review

- Notifications can deliver updates of RP changes
- The client must run another container to receive the notifications
- Notification delivery and order is not guaranteed

How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ **9. Discovery: Find a Resource**
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

Exercise 9: Discovery

4. State Management II:
Add a Resource Property

5. Aggregating Resources:
Register with a local index

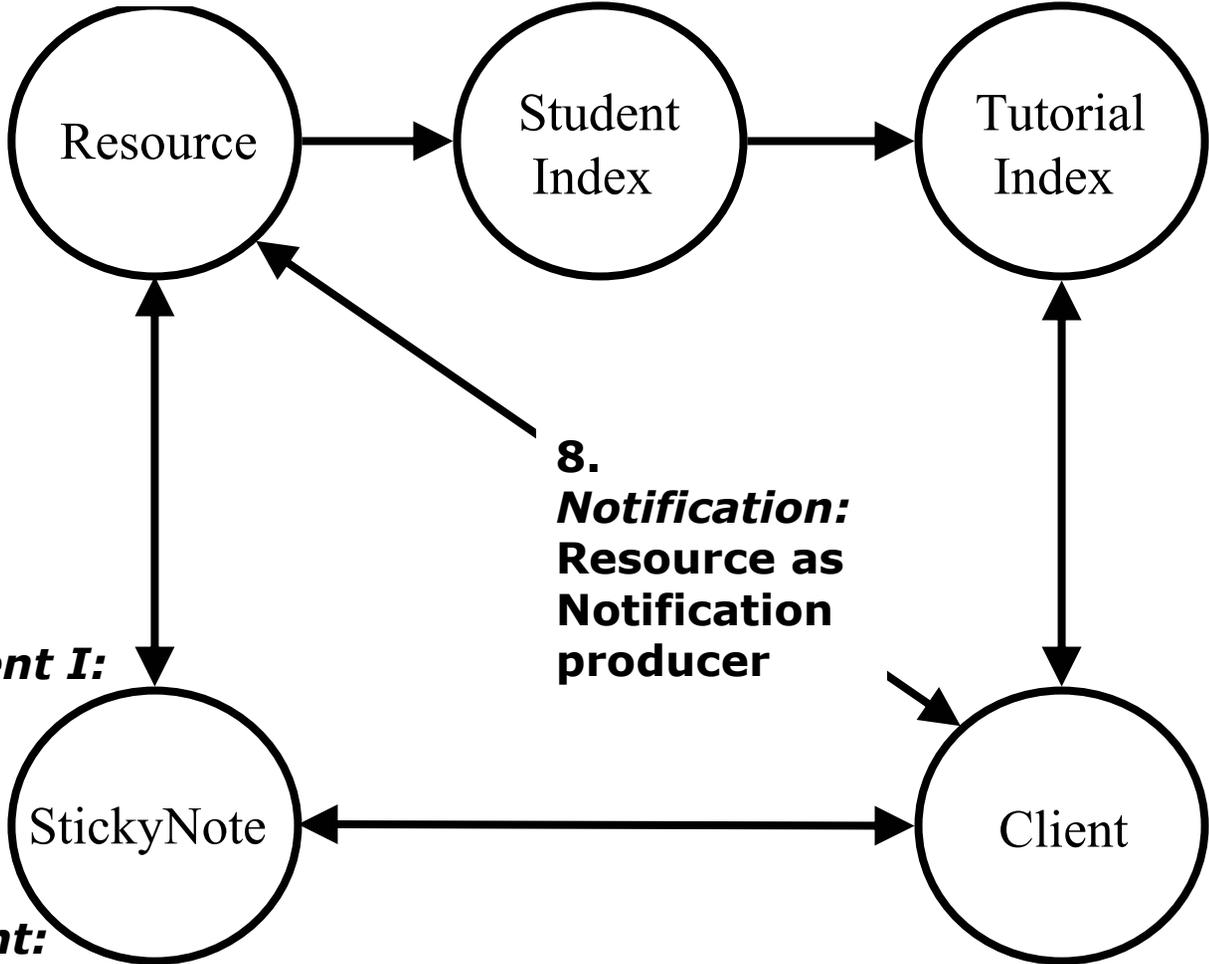
6. Virtual Organization:
Register with a community index

7. Lifetime Mgmt II:
Lease-Based Model

3. Lifetime Mgmt I:
Destroy Resources

2. State Management I:
Create Resources

1. Deployment:
Stand up a StickyNote service



8. Notification:
Resource as Notification producer

9. Discovery:
Find services that publish information you wish to retrieve

Concepts

What is Discovery?

- We want to find a resource that has some property
- In our case, a person wants to find a note that has particular content, like "hello."
- The VO Index knows about of all the notes and their messages
- We can search the VO Index for the notes that contain the word "hello."
 - ◆ As opposed to using EPRs
- The identification of the resource(s) meeting our criteria is called Discovery

Implementation details...

Implementation Details

- The VO Index publishes information on every StickyNote Resource that it knows about
- We can dump the entire content of that data using wsrp-query
- However, our Index will contain a large amount of data; it will be necessary to search inside of the output to find the data we need

Searching Inside Resource Properties

- To search the data we can use an XPath query against the Resource Property Set
- An XPath overview in two bullets:
 - ◆ XPath is a convenient query language for searching XML documents
 - ◆ XPath queries are formed by identifying a route to the desired data

We shall provide you with an XPath query to search the RP of the Tutorial Index...

Searching the Entries of the Tutorial Index

We can find each note that contains the word 'hello.' by sending the following XPath query to the Tutorial Index:

```
//*[local-name()='Entry'][.//*[contains(.,"hello.")]]
```

A human translation of this syntax:

“Select all the entries that contain the string 'hello.’”

Demonstration

Student Notes, Chapter 9

Exercise 9 Review

- The Index provides a way to discover services based on Resource Properties
- Because RPs are XML, GT4 allows XPath queries for searching and retrieving them

How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ **10. Security Part I: Service-Level**
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work



Exercise 10: Service-Level Security

4. State Management II:
Add a Resource Property

5. Aggregating Resources:
Register with a local index

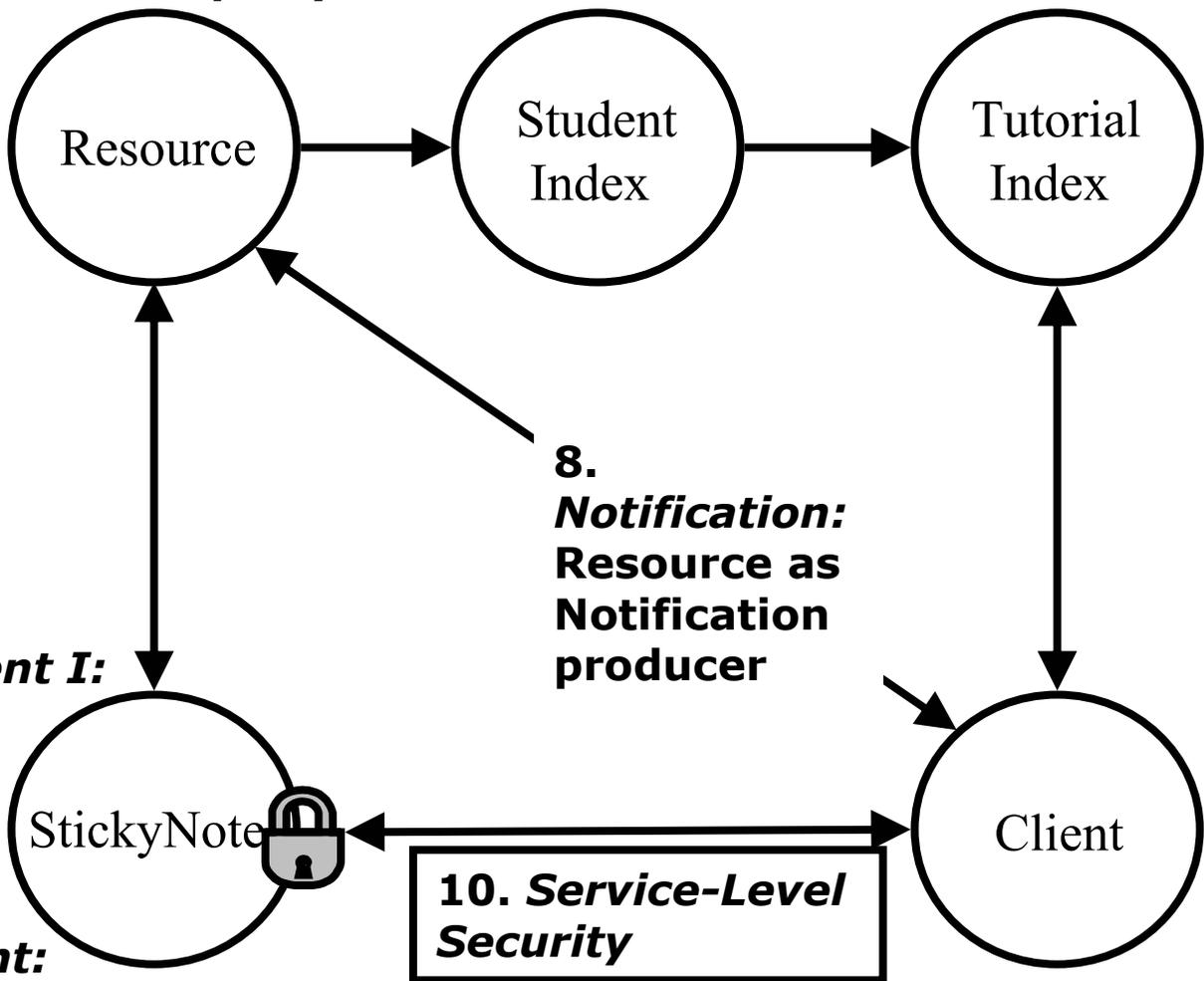
6. Virtual Organization:
Register with a community index

7. Lifetime Mgmt II:
Lease-Based Model

3. Lifetime Mgmt I:
Destroy Resources

2. State Management I:
Create Resources

1. Deployment:
Stand up a StickyNote service



8. Notification:
Resource as Notification producer

9. Discovery:
Find services that publish information you wish to retrieve

GT4 Security Details

- Built on top of PKI
 - ◆ Each entity has two keys: public and private
 - ◆ Data encrypted with one key can only be decrypted with other
 - ◆ The private key is known only to the entity
- The public key is given to the world encapsulated in a X.509 certificate

Certificates

- A X.509 certificate binds a public key to a name
- It includes a name and a public key bundled together and signed by a trusted party (Certificate Authority)
- An example of a Distinguished Name (DN):
“/O=Tutorial/OU=User/CN=Charles Bacon”

Certificate Authorities

- A Certificate Authority (CA) signs certificate requests
- To verify a certificate signature, you must have a copy of the CA certificate
- In GT land
 - ◆ By default, stored in /etc/grid-security/certificates
 - ◆ For our tutorial, stored in \$GLOBUS_LOCATION/share/certificates

Proxy Certificates

- Proxy certificates contain a new key pair, and are signed by the original certificate
 - ◆ Typically has shorter lifetime
 - ◆ By default stored in `/tmp/x509up_u$UID`
 - ◆ Protected by file system permissions
- Create a proxy using `grid-proxy-init`
 - ◆ Full GT4 install includes C command line client

Globus Certificate Service

- Globus Certificate Service
 - ◆ Web interface to get certificates
 - ◆ A certificate authority, but not quite
 - ◆ Low-quality certificates
- <http://gcs.globus.org:8080/gcs/index.html>
- Need CA certificate, already stored in `$GLOBUS_LOCATION/share/certificates` for the tutorial.

Certificates for tutorial

- Generate certificate to be signed:
 - ◆ `$GLOBUS_LOCATION/etc/grid-cert-request -caEmail false`
- Generates key (`userkey.pem`) and request (`usercert_request.pem`)
- Upload/Paste request in <http://gcs.globus.org:8080/gcs/usercert.html>
- Store returned certificate as `usercert.pem` in same directory as `userkey.pem`
- To generate proxy
 - ◆ `$GLOBUS_LOCATION/bin/visual-grid-proxy-init`

Authentication & Authorization

- Authentication
 - ◆ Establishing identity of an entity is what it is claimed to be
- Authorization
 - ◆ Ensuring an entity is allowed to access resource

GT4 Authentication Support

- Three types of authentication
 - ◆ Transport Level (HTTPS)
 - ◆ Message Level
 - Secure message
 - Secure conversation
- Signature and encryption
- Anonymous Clients

Server-side Security

- Security descriptors used to set security parameters
- Three levels of granularity:
 - ◆ Container level settings
 - Container security descriptor in WSDD file
 - ◆ Service level settings
 - Service security descriptor in service deployment in WSDD
 - ◆ Resource level settings
 - Programmatically set in resource object
- Resource-level setting has highest precedence

Server-side Security: Authentication

- Per-method authentication and run-as settings
- auth-method
 - ◆ none: no authentication
 - ◆ SecureMessage: GSI Secure Message
 - ◆ SecureConversation: GSI Secure Conversation
 - ◆ SecureTransport: GSI Transport
- run-as
 - ◆ caller: Execute method with caller's credential
 - ◆ system: Execute method with container credential
 - ◆ service: Execute method with service credential
 - ◆ resource: Execute method with resource credential

Server-side Security: Authorization

- Service-wide/resource-wide authorization settings
- A few built-in types:
 - ◆ None: no authorization of the client will be performed
 - ◆ Self: client will be authorized if it has the same identity as the service/resource/container
 - ◆ Identity: client will be authorized if it is the same identity as specified by the service/resource/container
 - ◆ GridMap: client will be authorized if its identity is listed in a gridmap file.
- Custom methods can be plugged in
 - ◆ For example, perform per-method checks
- Chainable

Client-side Security

- Security parameters set programmatically on the Stub (or Call object)
 - ◆ As individual properties
 - ◆ Using client security descriptor
- For example, the create-note client will set the protection mode of Secure Transport to encryption.

```
((Stub)portType)._setProperty(  
    GSIConstants.GSI_TRANSPORT,  
    Constants.ENCRYPTION);
```

Client-side Security: Authorization

- A few built-in types:
 - ◆ None, Self, Identity (same as on server)
 - ◆ Host: service/resource will be authorized if the host returns an identity containing the hostname

Client authorization is independent and separate from server authorization!

What Attendees Should Do

- Uncomment the “securityDescriptor” parameter in WSDD
- Edit grid-mapfile
 - ◆ Add identity of another person
 - ◆ Have them create/edit note on your service
- Create a proxy
- Create a note and write to it

What Attendees Should See

- Without proxy, operations should fail
- With valid credentials,
 - ◆ If you are not in gridmap, you cannot create or write notes
 - ◆ Read note (show-note) is not secure and should be allowed even if not in gridmap

Exercise 10 Review

- Container and Service security is configured declaratively using security descriptor file
- Resource and Client security is configured programmatically by setting properties in the code
- Service-side authentication may be specified on a per-operation basis
- Custom service-side authorization may be plugged in and chained with other schemes

How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ **11. Security Part II (optional): Resource-Level**
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- Ideas for Further Work

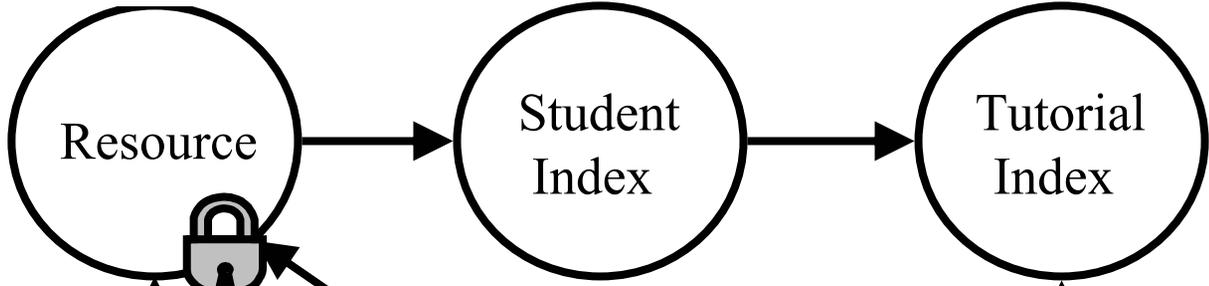
Exercise 11: Resource-Level Security

4. State Management II:
Add a Resource Property

5. Aggregating Resources:
Register with a local index

6. Virtual Organization:
Register with a community index

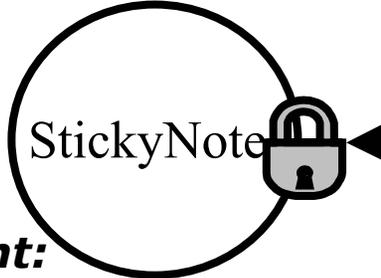
7. Lifetime Mgmt II:
Lease-Based Model



3. Lifetime Mgmt I:
Destroy Resources

2. State Management I:
Create Resources

1. Deployment:
Stand up a StickyNote service



8. Notification:
Resource as Notification producer

10. Service-Level Security

9. Discovery:
Find services that publish information you wish to retrieve

Resource-Level Security

- This is an optional exercise
- There is no lecture for this chapter
- The lesson is contained in the student notes

How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- **Overview of Tools for GT4 Service Developers**
- Tutorial Summary
- Ideas for Further Work

Tools

- WSDLPP
- Binding Generator
- Stub Generator (Axis)
- Launcher Generator

Tools: WSDLPP

- WSDL Preprocessor
- Custom attribute specifies PortType composition
- Avoids copy-and-paste errors
- Handles Resource Properties merging

```
<ant antfile="share/globus_wsrf_tools/build-stubs.xml" target="flatten">  
  <property name="source.flatten.dir"  
    location="core/samples/counter"/>  
  <property name="target.flatten.dir"  
    location="core/samples/counter"/>  
  <property name="wsdl.source"  
    value="counter_port_type.wsdl"/>  
  <property name="wsdl.target"  
    value="counter_flattened.wsdl"/>  
  <property name="wsdl.porttype"  
    value="CounterPortType"/>  
</ant>
```

Tools: WSDLPP

Before:

```
<portType name="CounterPortType"  
  wsdlpp:extends="wsrpw:GetResourceProperty ...">  
  <operation name="add">  
    ...  
  </operation>  
</portType>
```

After:

```
<portType name="CounterPortType">  
  <operation name="add">  
    ...  
  </operation>  
  <operation name="GetResourceProperty">  
    ...  
  </operation>  
</portType>
```

Tools: Binding Generator

- Generates WSDL Binding and Service parts from WSDL PortType
- SOAP binding with Document/literal
- Handles WS-Addressing action attributes

```
<ant antfile="share/globus_wsrf_tools/build-stubs.xml"  
  target="generateBinding">  
  <property name="source.binding.dir"  
    value="core/samples/counter"/>  
  <property name="target.binding.dir"  
    value="core/samples/counter"/>  
  <property name="porttype.wsdl"  
    value="counter_flattened.wsdl"/>  
  <property name="binding.root"  
    value="counter"/>  
</ant>
```

Tools: Launcher Generator

- Generates Windows batch files and Unix/Linux shell scripts for Java clients
- Classpath, standard environment variables are automatically handled

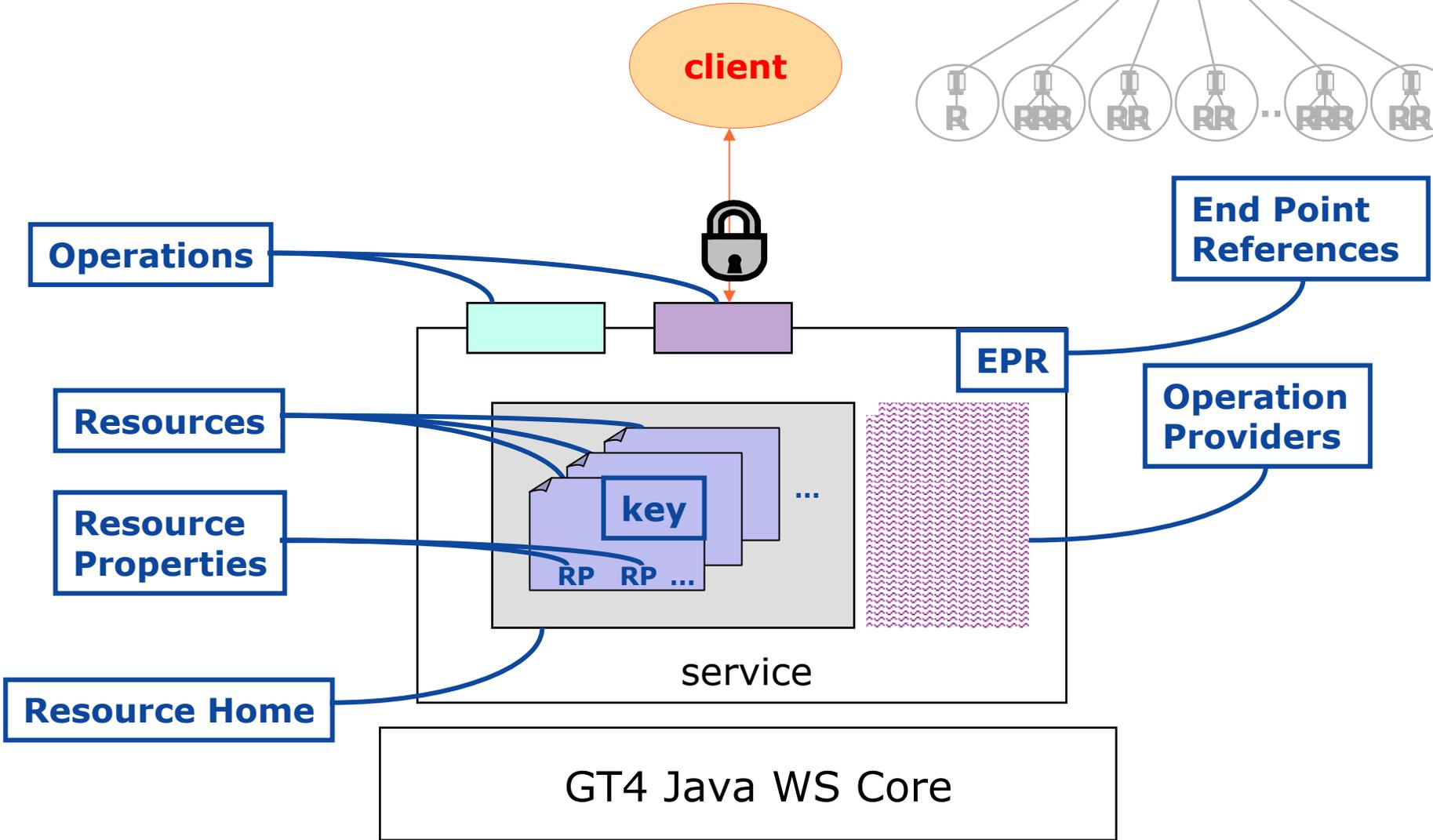
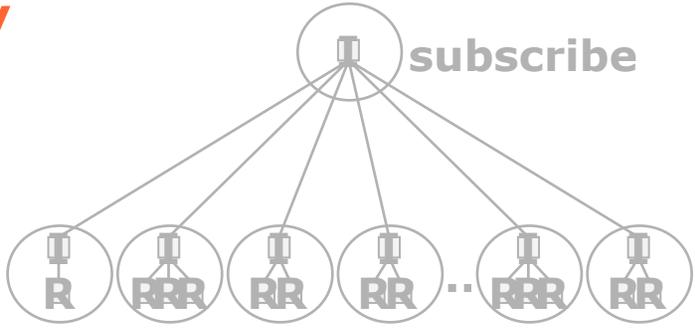
```
<ant antfile="share/globus_wsrf_common/build-launcher.xml"
    target="generateLauncher">
  <property name="launcher-name"
    value="grid-proxy-init"/>
  <property name="class.name"
    value="org.globus.tools.ProxyInit"/>
</ant>
```

How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- **Tutorial Summary**
- Ideas for Further Work

Summary

setTerminationTime



How to Build a Service Using GT4

- Overview of Services and GT4
- Build a Service
 - ◆ 1. Getting Started: Deploy a Service
 - ◆ 2. State Management Part I: Create Resources
 - ◆ 3. Lifetime Management Part I: Destroy Resources
 - ◆ 4. State Management Part II: Add a Resource Property
 - ◆ 5. Aggregating Resources: Register with a Local Index
 - ◆ 6. Building a VO: Register with a Community Index
 - ◆ 7. Lifetime Management Part II: Lease-based Model
 - ◆ 8. Notification: Resource as Notification Producer
 - ◆ 9. Discovery: Find a Resource
 - ◆ 10. Security Part I: Service-Level
 - ◆ 11. Security Part II (optional): Resource-Level
- Overview of Tools for GT4 Service Developers
- Tutorial Summary
- **Ideas for Further Work**

Join the GT4 Community

- Download and use the software, and provide feedback
 - ◆ Join the **gt4-friends@globus.org** mail list
- Review, critique, add to documentation
 - ◆ Globus Doc Project: **<http://gdp.globus.org>**
- Tell us about your GT4-related tool, service, or application
 - ◆ Send mail to **info@globus.org**

Further Reading

GT4 draft documentation page:

<http://www.globus.org/toolkit/docs/4.0/>

Globus Alliance publications page:

<http://www.globus.org/alliance/publications/papers.php>

WS-Resource Framework:

<http://www.globus.org/wsrp/>

On WSDL:

<http://www.w3.org/TR/wsdl>

The GT4 Fact Sheet:

<http://www.globus.org/toolkit/docs/4.0/GT4Facts/>

The Globus Documentation Project:

<http://gdp.globus.org>

