

GNS-WSI2

Grid Network Service - Web Services Interface, version 2

G-lambda Project

National Institute of Advanced Industrial Science and
Technology (AIST)

KDDI R&D Laboratories Inc.

NTT Network Innovation Labs.

Jun. 2006

GNS-WSI2

- GNS-WSI (Grid Network Services - Web Services Interface)
 - Web services based interface
 - 1-phase commit
- GNS-WSI2
 - **WSRF**(Web Services Resource Framework) based interface
 - GT4 (Globus Toolkit 4) Java WS Core
<http://www.globus.org/toolkit/>
 - **2-phase commit**

GNS-WSI2 Services

- **ReservationFactoryService**
 - Creates **ReservationResource** instance in **ReservationService**
 - Provides network resource information
 - **ReservationService**
 - Provides general reservation operations
 - reservation, modification, release
 - Creates **ReservationCommandResource** instance in **ReservationCommandService**
 - **ReservationCommandService**
 - Realizes 2 phase commit and non-blocking operation
- (**xxxResource** is a service instance for each user request)

GNS-WSI2 Operations

Service	Operations	Remarks
Reservation Service	netResourceReservation	advance reservation request
	netResourceModification	modification request
	netResourceRelease	release request
	getResourceProperty(QName)	returns specified ReservationResource property
Reservation Factory Service	createReservationResource	creates reservation resource
	netResourceQuery	discovery of resources between designated sites
	netAvailableResourceQuery	discovery of available resources
Reservation Command Service	getResourceProperty	returns CommandStatus
	commit	commit the request
	abort	abort the request

Service Parameters

Parameter	Usage	Value	Remarks
Site ID (APoint, ZPoint)	ID to specify A and Z points	String	Name or ID of sites
bandwidth	Bandwidth of the resource	Positive integer (kbit/s)	
latency	Latency between end points	Positive integer (msec)	
availability	Network protection of network resource	Integer ($-2^{32} \sim 2^{32}-1$)	0 = Un-protected 1 = Protected
Reservation time (startTime, endTime)	Start time and end time of the reservation	xsd:dateTime	YYYY-MM-DDTHH:MM:SSZ
localUsername	user name of certificate	String	GT4 GSI
reservationStatus	status of reservation	String	p. 15
commandStatus	status of each command	String	p. 16
resourceStatus	status of network resource	String	<i>Available / NotAvailable</i>

ReservationServicePortType

- **netResourceReservation**
in: APoint, ZPoint, startTime, endTime, bandwidth(, availability, latency)
out: reservationCommandEPR
- **netResourceModification**
in: (bandwidth, startTime, endTime)
out: reservationCommandEPR
- **netResourceRelease**
in: N/A
out: reservationCommandEPR
- **getResourceProperty**
in: QName of a ReservationResourceProperty
out: ReservationResourceProperty

Operation name

in: input parameters

out: output parameters

fault: fault type

(): Optional parameters

ReservationResourceProperties

- Indispensable properties
 - reservationStatus
 - Optional properties
 - APoint
 - ZPoint
 - availability
 - bandwidth
 - latency
 - startTime
 - endTime
 - reservationCommandHistory
 - localUsername
 - errorInfo
- reservationCommandHistory is array of:

 - timestamp
 - reservationCommandEPR
 - submitted operation name
 - operation arguments (optional)
 - final commandStatus (optional)

ReservationFactoryPortType

- **createReservationResource**

in: N/A

out: reservationEPR

- **netResourceQuery**

in: array of [pathId, APoint, ZPoint, startTime, endTime, bandwidth
(, availability, latency)]

out: array of [pathId, resourceStatus("Available"/"NotAvailable")]

- **netAvailableResourceQuery**

in: startTime, endTime(, APoint, ZPoint, availability, bandwidth,
latency)

out: array of [APoint, ZPoint, bandwidth(, availability, latency)]

Operation name

in: input parameters

out: output parameters

fault: fault type

(): Optional parameters

ReservationCommandPortType

- **commit**
in: N/A
out: N/A
- **abort**
in: N/A
out: N/A
- **getResourceProperty**
in: QName of
"commandStatus"
out: commandStatus

ReservationCommand ResourceProperties

- Indispensable properties
 - commandStatus
 - reservationResourceKey
- Optional properties
 - errorInfo

Operation name

in: input parameters

out: output parameters

fault: fault type

(): Optional parameters

Error Cases

- Applies WS-BaseFault
- Operation exception (OperationErrorMessages)
 - GNS-WSI2 operations throw org.oasis.wsrp.faults.BaseFaultType

```
FaultHelper helper = new FaultHelper(new BaseFaultType());  
String[] description = {"Parameter error / APoint.", "Parameter error /  
startTime."};  
helper.setDescription(description);  
throw helper.getFault();
```
- Internal error (InternalErrorMessages)
 - NRM updates ResourceProperties.{reservationStatus,errorInfo}

```
ReservationStatus reservationStatus = ReservationStatus.ERROR  
BaseFaultType errorInfo =  
    new BaseFaultType(Calendar.getInstance(), null, description, null);
```
- Some typical error messages are defined

OperationErrorMessages (1/2)

Messages	Remarks	FS	RS	CS
Parameter error / APoint	String of APoint is out of range	Y	Y	
Parameter error / ZPoint	String of ZPoint is out of range	Y	Y	
Parameter error / bandwidth	bandwidth is out of range	Y	Y	
Parameter error / latency	latency is out of range	Y	Y	
Parameter error / availability	availability is out of range	Y	Y	
Parameter error / startTime	startTime is out of range	Y	Y	
Parameter error / endTime	endTime is out of range	Y	Y	

FS: ReservationFactoryService fault message

RS: ReservationService fault message

CS: ReservationCommandService fault message ¹¹

OperationErrorMessages (2/2)

Messages	Remarks	FS	RS	CS
System error	Error of the NRM service system	Y	Y	Y
Internal error	Server system internal error	Y	Y	Y
Service busy	Congestion of the NRM services	Y	Y	Y
Command busy	The previous command has not completed		Y	Y
Resource finalized	The Resource has finalized		Y	Y
Not in service / APoint	APoint does not exist or is not in service	Y	Y	
Not in service / ZPoint	ZPoint does not exist or is not in service	Y	Y	

FS: ReservationFactoryService fault message

RS: ReservationService fault message

CS: ReservationCommandService fault message ¹²

InternalErrorMessage

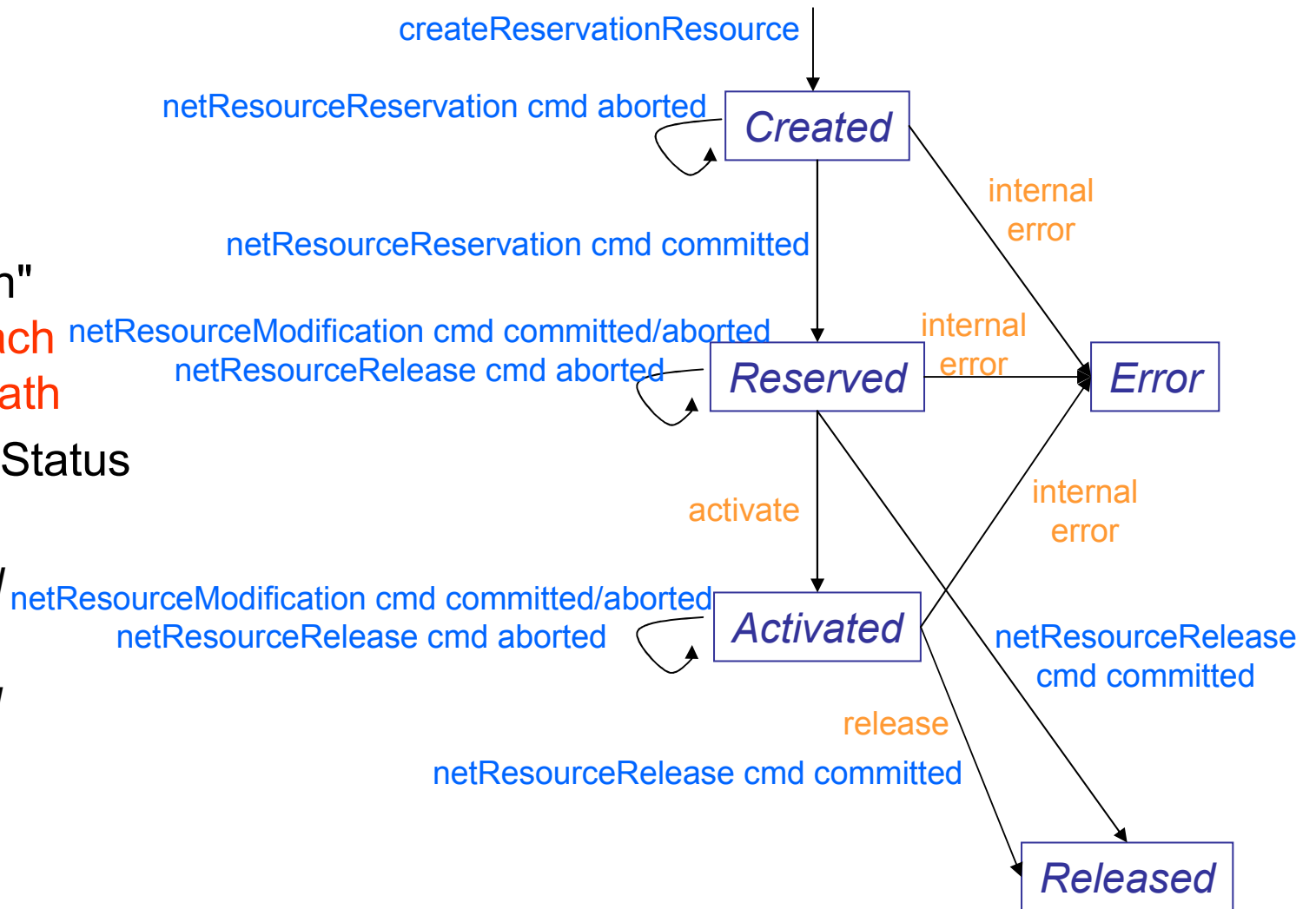
Messages	Remarks	Rsv	Rsv Cmd
Reservation commit failure	Reservation commit has failed	Y	
Activation failure	Activation of the reserved path has failed	Y	
Modification commit failure	Modification commit has failed	Y	
Release commit failure	Release commit has failed	Y	
Reservation not acceptable	Pre-reservation is not acceptable		Y
Modification not acceptable	Pre-modification is not acceptable		Y
Release not Acceptable	Pre-release is not acceptable		Y

Rsv: errorInfo of ReservationResource

RsvCmd: errorInfo of ReservationCommandResource 13

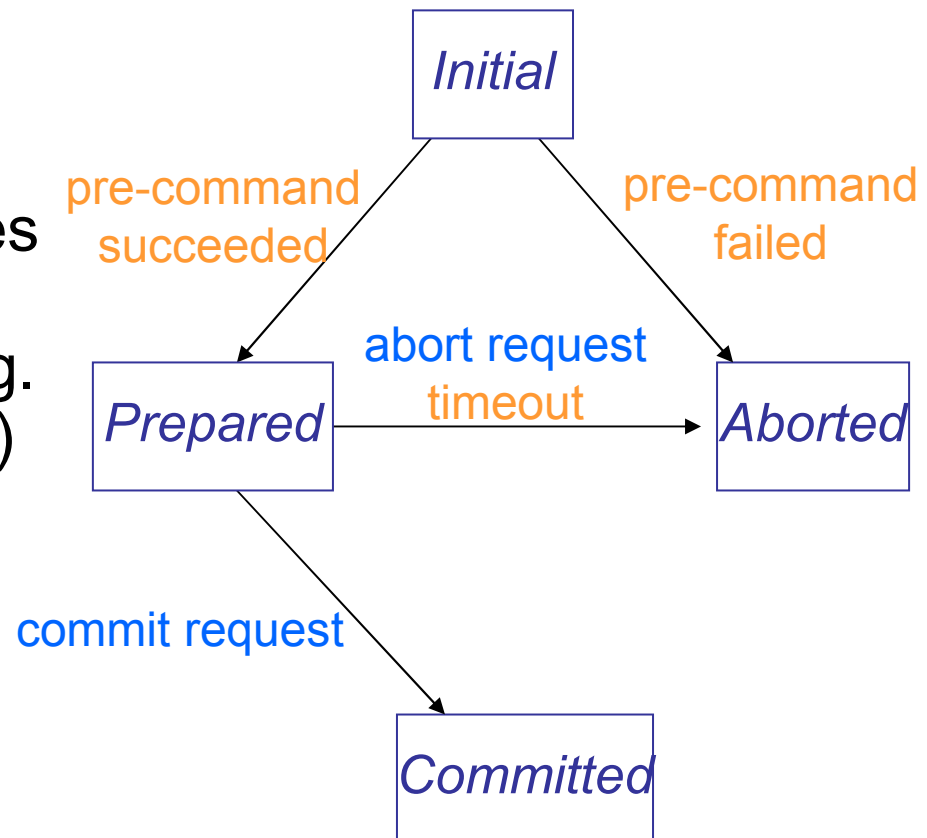
ReservationStatus (Reservation)

- Represents current "Reservation" status for **each requested path**
- ReservationStatus
 - *Created*
 - *Reserved*
 - *Activated*
 - *Released*
 - *Error*



CommandStatus (ReservationCommand)

- Represents current "ReservationCommand" status of each ReservationCommandResource created by a reservation operation (e.g. netResourceReservation)
- CommandStatus:
 - *Initial*
 - *Prepared*
 - *Committed*
 - *Aborted*



Status Transition & Action

top: reservation
middle: modification
bottom: release

CmdStatus	Condition	Action	RS
Initial ↓	netResourceReservation req.		Cr
	netResourceModification req.		Rs / Ac
	netResourceRelease req.		Rs / Ac
Initial ↓ Prepared	MR available	pre-reserve MR	Cr
	AMR available	pre-reserve AMR, DMR release ready	Rs / Ac
	MR release available	MR release ready	Rs / Ac
Prepared ↓ Committed	commit req.	reserve MR	Cr → Rs
		reserve AMR, release DMR	Rs / Ac
		release MR	Rs/Ac → RI
Initial ↓ Aborted	MR not available		Cr
	AMR not available		Rs / Ac
	MR release ready not available		Rs / Ac
Prepared ↓ Aborted	abort req. / timeout	cancel MR pre-reservation	Cr
		cancel AMR pre-reservation, cancel DMR release ready	Rs / Ac
		cancel MR release ready	Rs / Ac

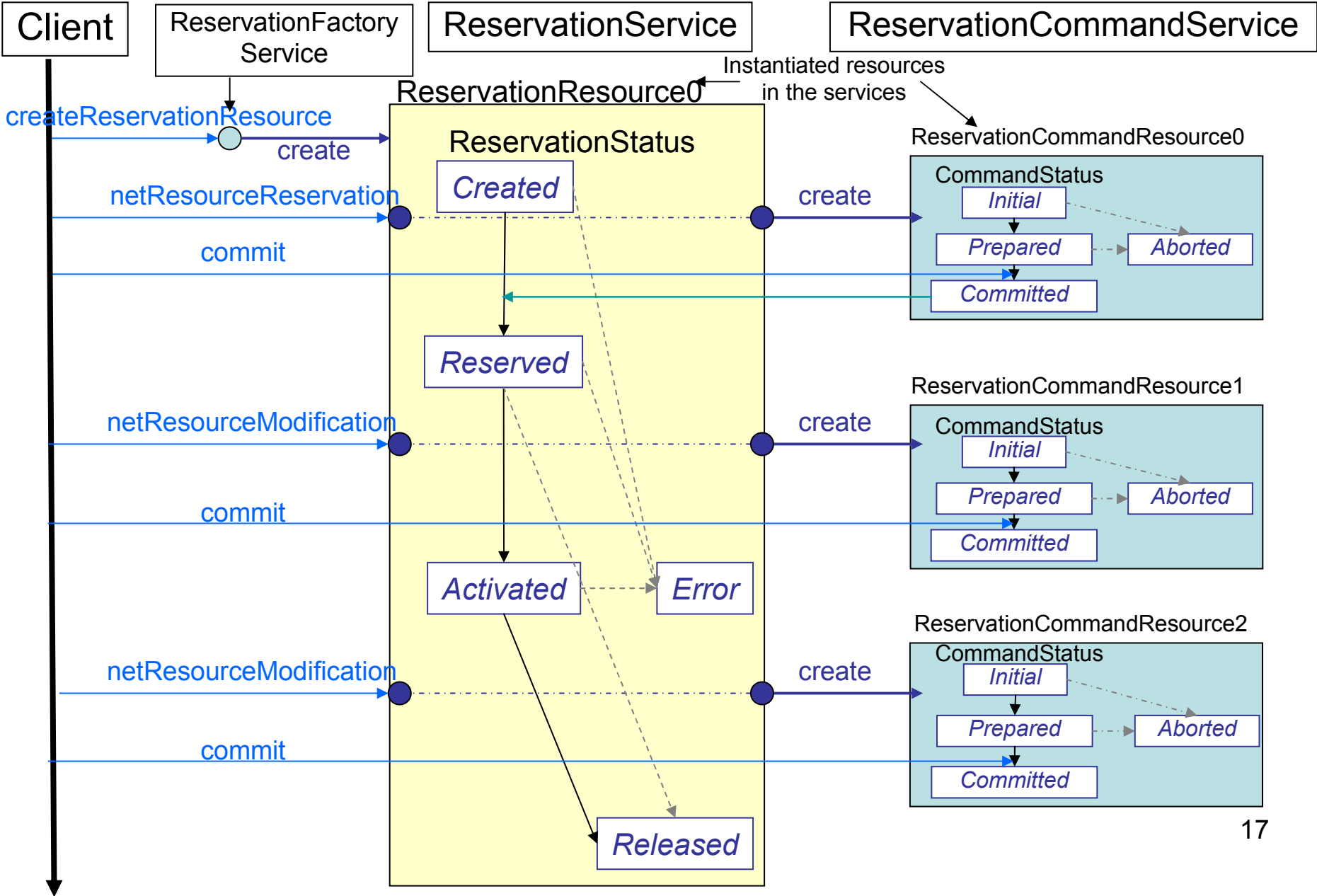
MR: Managed Resources, i.e. the resources under management of the resource manager

AMR: Additional Managed Resources DMR: Disused Managed Resources

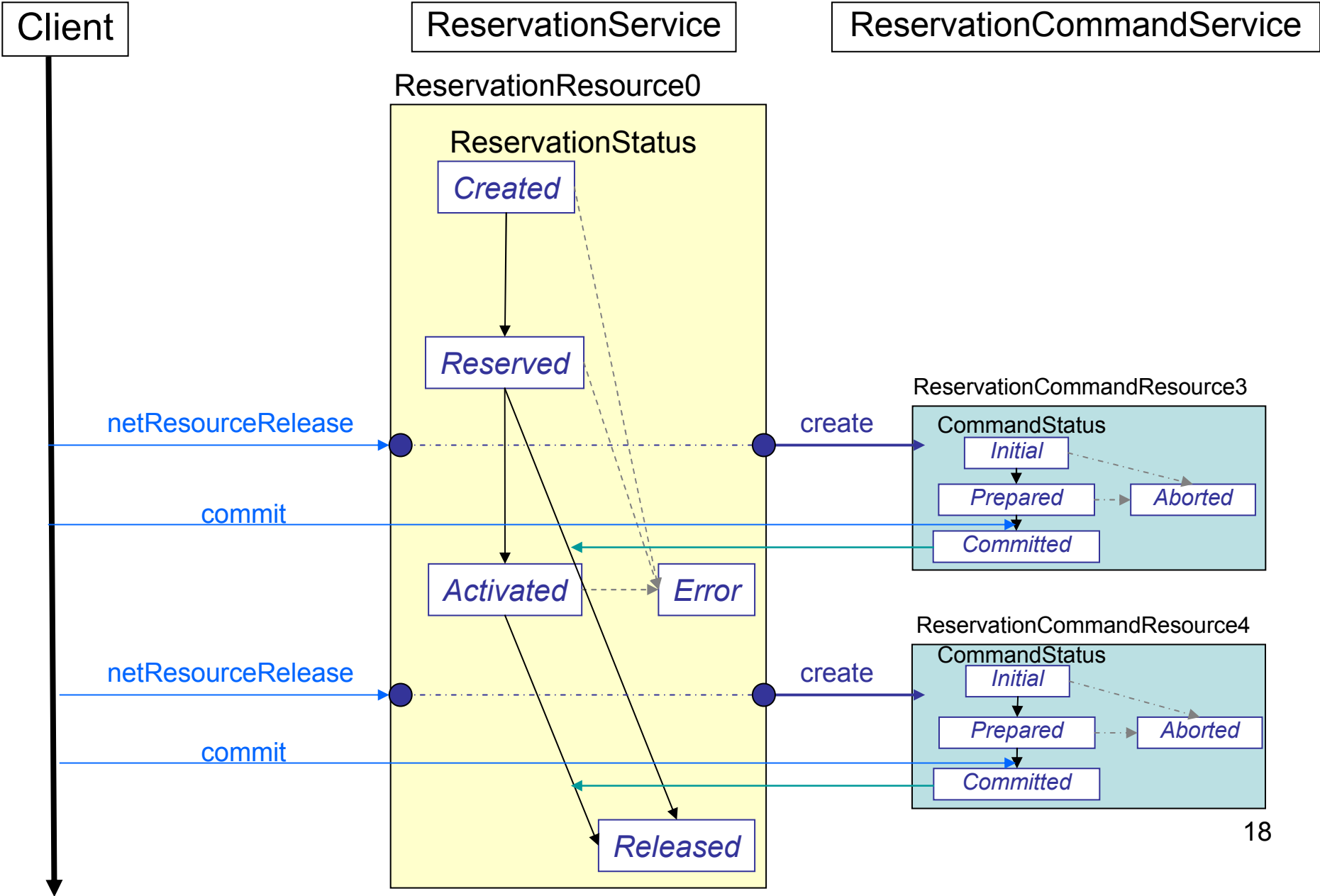
RS: ReservationStatus Cr: Created
Rs: Reserved Ac: Activated

RI: Released E: Error

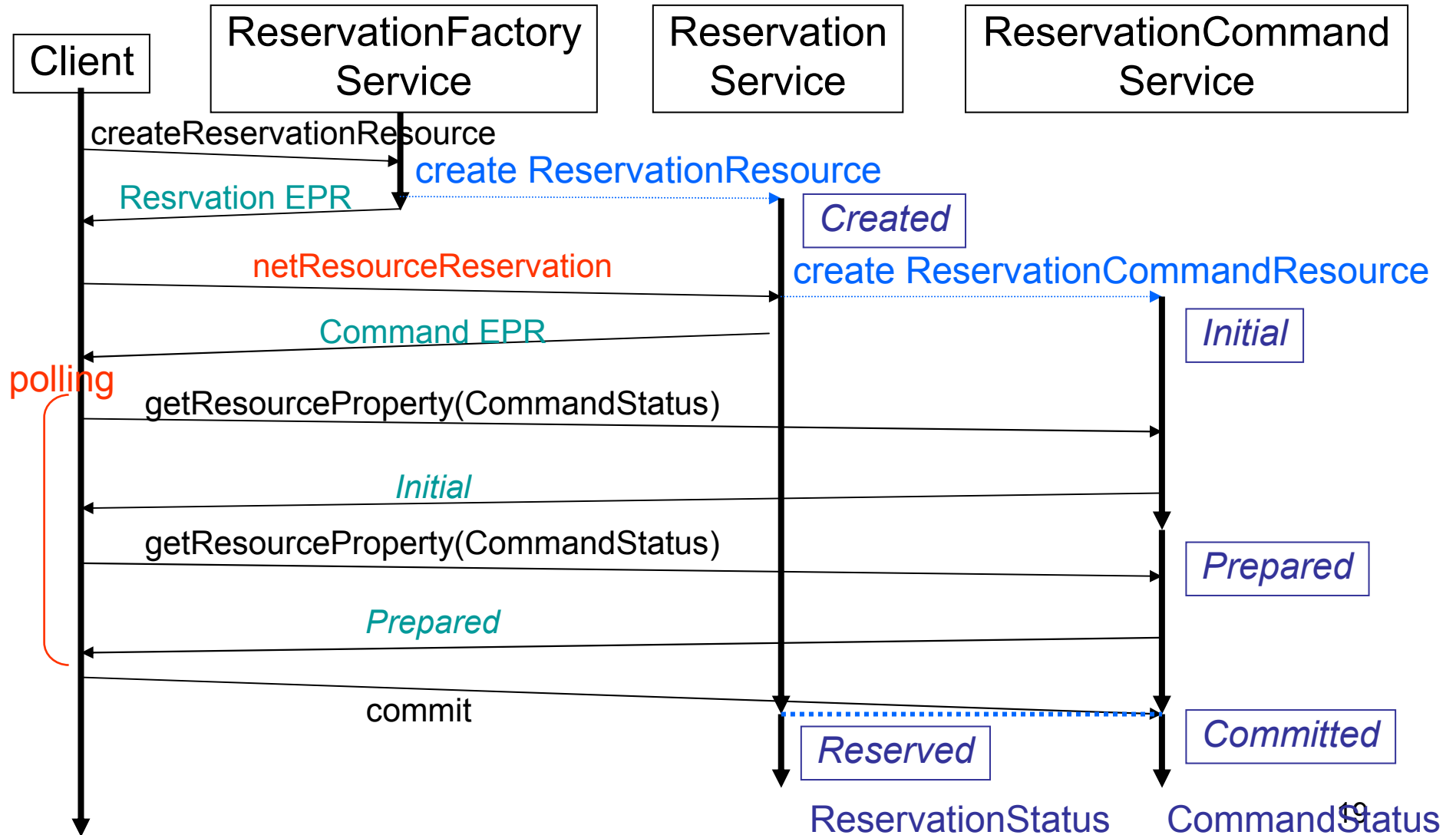
2 Phase Commit & Status Transition(1)



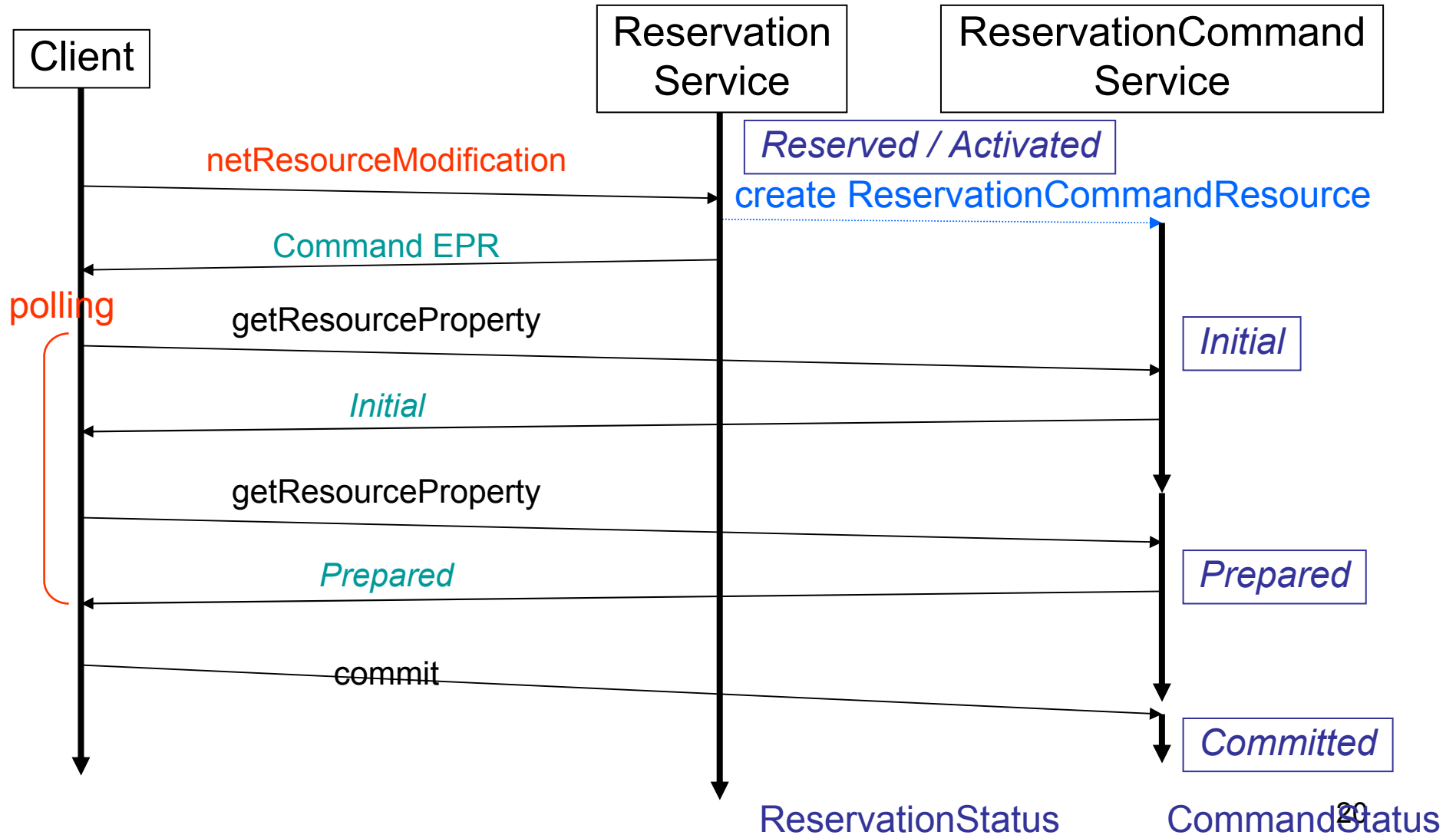
2 Phase Commit & Status Transition(2)



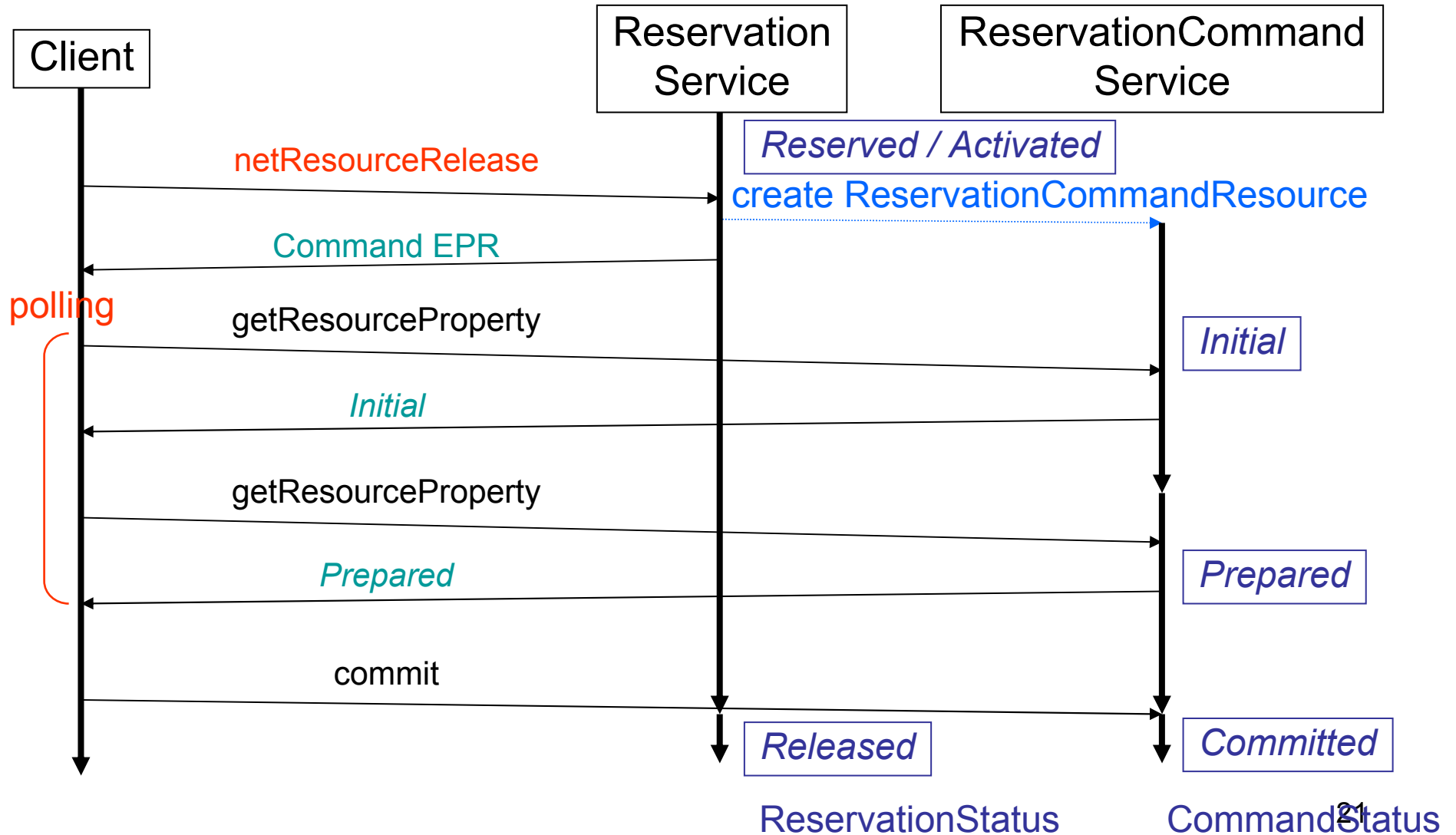
Reservation Diagram



Modification Diagram

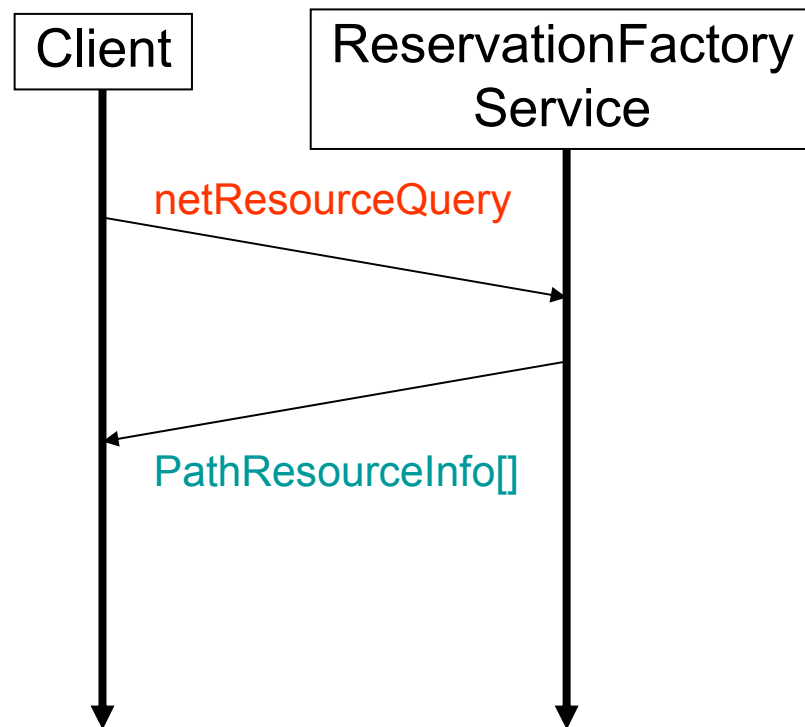


Release Diagram

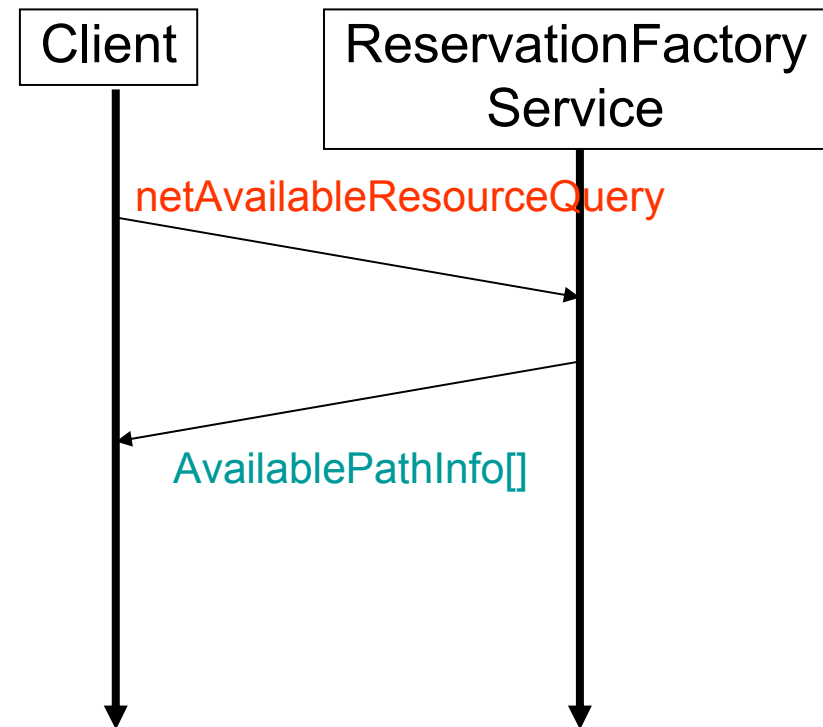


Information Service Diagrams

- netResourceQuery (blocking)



- netAvailableResourceQuery (blocking)



Lifetime of ReservationResource, CommandResource

- Applied WS-ResourceLifetime
- ReservationResource terminationTime
 - Set terminationTime at resource creation
$$\text{terminationTime} = \text{currentTime} + \alpha$$
 - Update terminationTime at update of reservation endTime (after netResourceReservation or netResourceModification)
$$\text{terminationTime} = \text{endTime} + \beta$$
- CommandResource terminationTime
 - Set terminationTime at resource creation
$$\text{terminationTime} = \text{currentTime} + \chi$$
 - Update terminationTime at termination status: *Committed* or *Aborted*
$$\text{terminationTime} = \text{currentTime} + \delta$$

Time Out (Modified 2 phase commit)

- 2 phase commit is a blocking protocol
 - If Coordinator fails after netResourceXXX req., commandStatus may be left in *Prepared* state until the coordinator is repaired
 - In our case, coordinator and cohorts are loosely coupled, and the coordinator (= user) may not issue commit or abort req.
- Provided a timeout to transit from *Prepared* to *Aborted* automatically
 - *Prepared* waiting for commit or abort req. timeouts at timeOutTime
 - timeOutTime = transit time from *Initial* to *Prepared* + ϵ

Temporary Operation for Demo

- ReservationFactoryService.**getTimetable**
 - Supports visualization of status of requested path resources
 - Interface
 - in: Calendar startTime, Calendar endTime
 - out: String timetable (XML document)
 - Timetable includes reservation information:
 - Reservation ID, Site IDs, Bandwidth, start/end time

An Example of Timetable

```
<timetable nrm="g-lambda1"> <!-- nrm = "NRM FactoryService URL" -->
  <reservation id="12345"> <!-- id : key of EPR -->
    <aPoint value="AKB"/>
    <zPoint value="TKB"/>
    <bandwidth value="1000000"/>
    <startTime value="YYY0-M0-D0TH0:M0:S0Z"/>
    <endTime value="YYY1-M1-D1TH1:M1:S1Z"/>
  </reservation>
  <reservation id="67890">
    <aPoint name="AKB"/>
    <zPoint name="TKB"/>
    <bandwidth value="1000000"/>
    <startTime value="YYY2-M2-D2TH2:M2:S2Z"/>
    <endTime value="YYY3-M3-D3TH3:M3:S3Z"/>
  </reservation>
</timetable>
```

GNS-WSI2 Reference Implementation

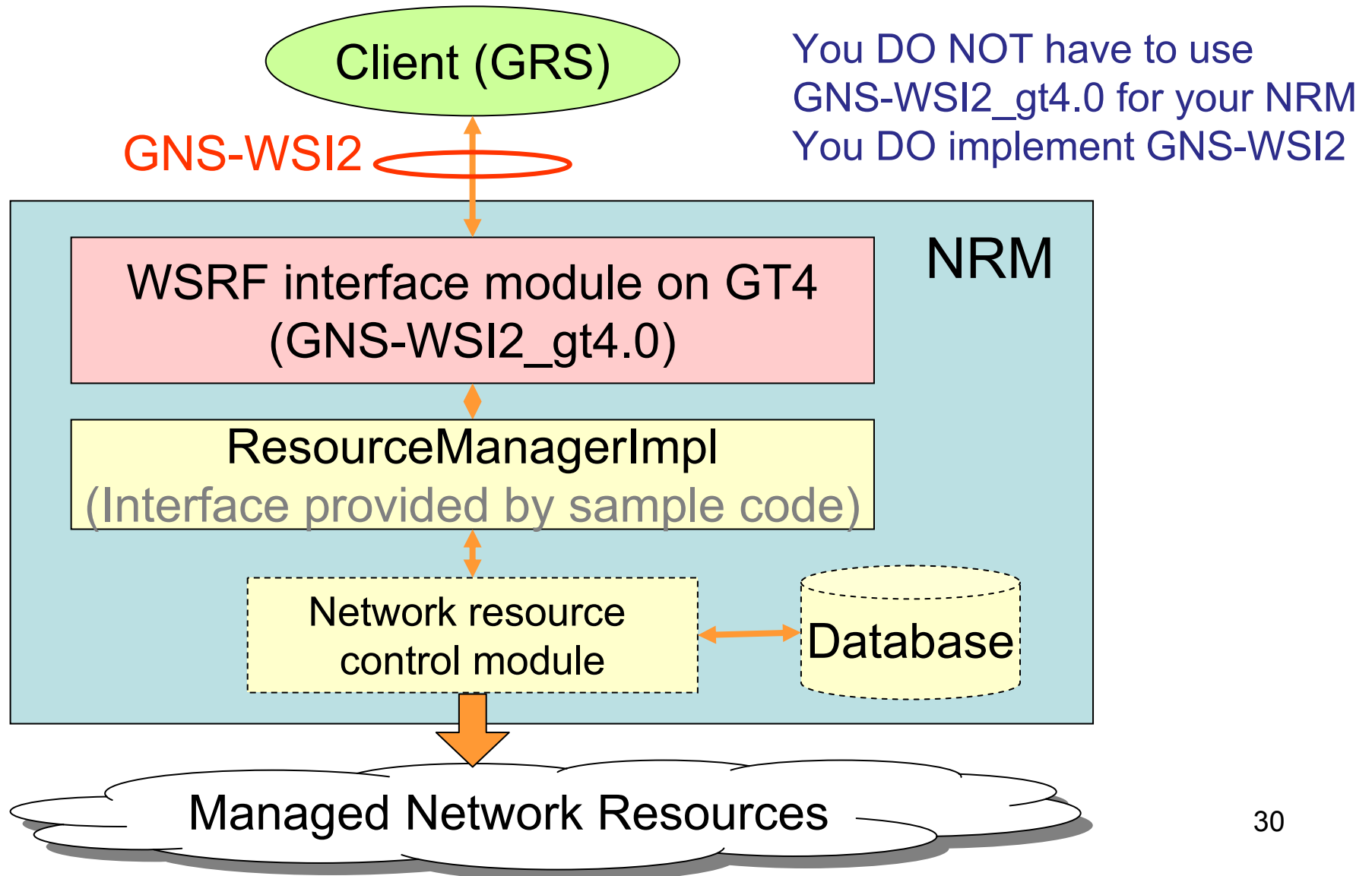
GNS-WSI2_gt4.0

- Developed by AIST
- A reference implementation of the GNS-WSI2 NRM WSRF modules
- Developed using GT4.0 Java WS Core
- ReservationResource and ReservationCommandResource have implemented using PersistentReflectionResource
 - Store the both Resource Properties persistently
 - ~globus/.globus/persisted/{IP address}-8080/{ReservationResourceProperties,ReservationCommandResourceProperties}

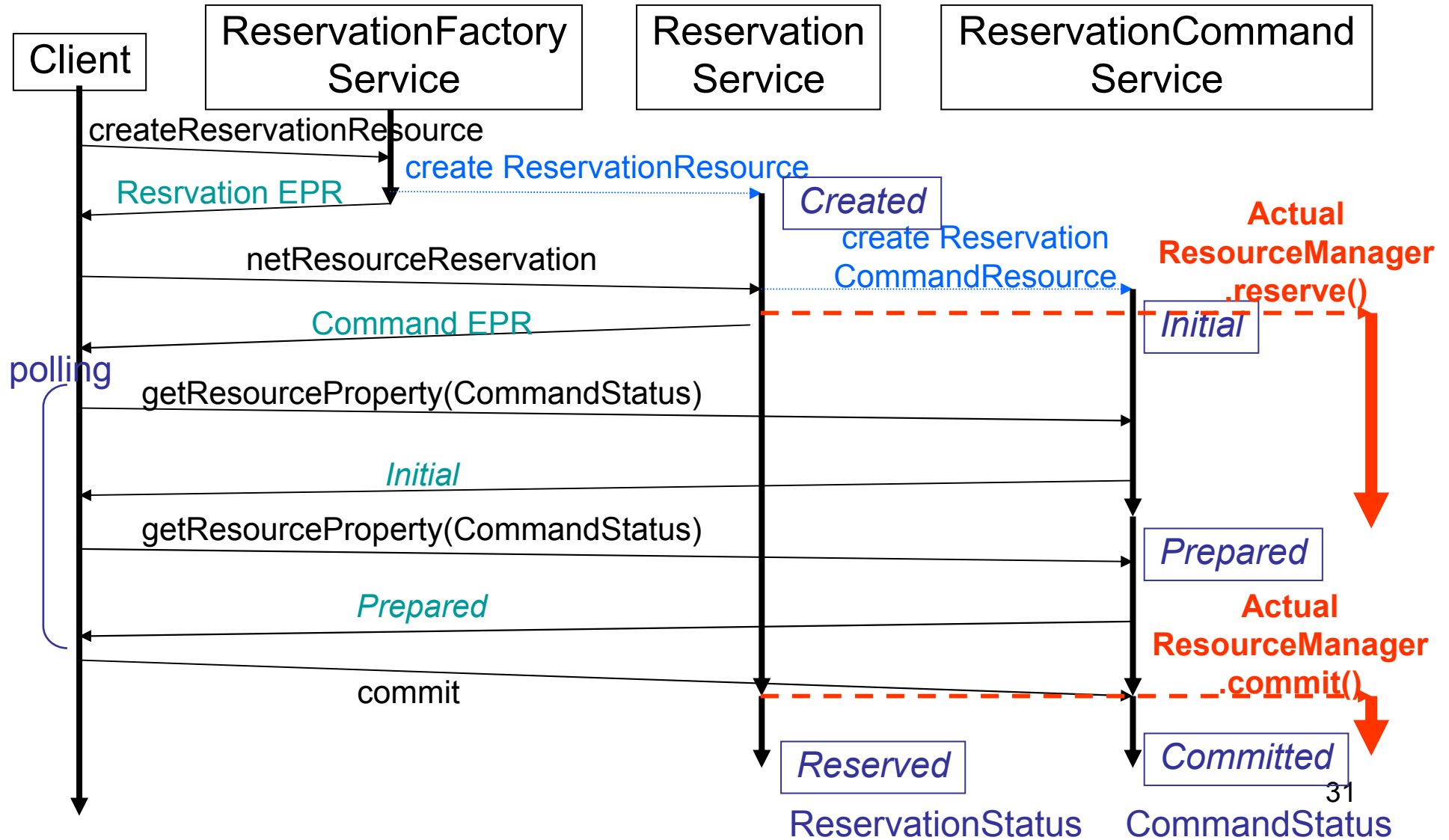
Install GNS-WSI2_gt4.0

- GNS-WSI2_gt4.0.tgz (current version)
 - build.xml
 - schema
- How to build the stub files
 - expand files in a directory
`$ tar zxvf GNS-WSI2_gt4.0.tgz`
 - edit "location" of "env.GLOBUS_LOCATION" property in build.xml
`<property name="env.GLOBUS_LOCATION" location="xxx"/>`
 - ant build
`$ ant`
 - build/lib/g-lambda_stubs.jar will be created

Assumed NRM Architecture



Reservation Diagram



Overview of Polling-based Detection of Actual Resource Status

```
ReservationResource

static class TimerTaskHolder {
    Timer timer;
}

final timerTaskHolder timerTaskHolder =
    new TimerTaskHolder();

void setTimerTask(TimerTask task, Calendar cal) {
    Timer timer = new Timer();
    long delay = cal - now - offset;
    timer.schedule(task, delay); // Starting task in "delay" duration
    this.timerTaskHolder = timer;
}

// ReservationCommandCallback calls:
void setReservedStatus() {
    // set reservationStatus = Reserved;
    // then, set polling task timer for detecting the Activated status
    setTimer(new PollingTask(), startTime);
}

// PollingTask instance for Activation calls:
void setActivatedStatus() {
    // set reservationStatus = Activated;
    // then, set polling task timer for detecting the Released status
    setTimer(new PollingTask(), endTime);
}
}
```

PollingTask instance for Activation

PollingTask extends TimerTask

```
// wait for "delay" duration
// and execute the run() method
public void run() {
    // start polling
    while (!timeout) {
        // get resource manager's
        // reservationStatus
        ReservationStatus status =
            ResourceManagerImpl.getStatus();
        if (status == Activated)
            change rsvResource.reservationStatus;
    }
    if (timeout or another error cases)
        // error handling
}
```

```
if (status == Released)
    change rsvResource.reservationStatus;
}
if (timeout or another error cases)
    // error handling
}
```

PollingTask instance for Release

How to Run GNS-WSI2_gt4.0 (1/2)

- Required environment
 - GT4 ver. 4.0.x (<http://www.globus.org/toolkit/downloads/4.0.x/>)
 - Java 5.0.x
 - Ant 1.6.x
- Build the sample code
 1. Make "globus" account and install GT4
 - <http://www.globus.org/toolkit/docs/4.0/admin/docbook/quickstart.html>
 - Even if NO GSI, the certificate settings are also required for -nosec container
 2. Expand sample code in your test directory

```
% cd ${TEST_DIR}
% tar -zxf sample060601.tgz
% mv sample060601/GNS-WSI2
```
 3. Build sample code

```
% export GLOBUS_LOCATION=??? // set ${GLOBUS_LOCATION}
% cd GNS-WSI2
% ant
.....
${TEST_DIR}/build/lib/glambda.gar created.
```

How to Run the Sample Code (2/2)

- Deploy the created glambda.gar file as "globus"
globus% `${GLOBUS_LOCATION}/bin/globus-deploy-gar`
`${TEST_DIR}/build/lib/glambda.gar`
- Start GT4 container with -nosec as "globus"
globus% `source ${GLOBUS_LOCATION}/etc/globus-user-env.sh`
globus% `${GLOBUS_LOCATION}/bin/globus-start-container -nosec`
If you prefer background job, run as follows:
globus% `${GLOBUS_LOCATION}/sbin/globus-start-container-detached -nosec`
(`${GLOBUS_LOCATION}/var/container.log` is the log file)
- Run "glambda_test", sample client code
% `${TEST_DIR}/GNS-WSI2/bin/glambda_test`
You can also invoke requests step-by-step using
`${TEST_DIR}/GNS-WSI2/bin/glambda_XXX`
- Stop the container as globus
Kill the container process by Ctrl-C, or
globus% `${GLOBUS_LOCATION}/sbin/globus-stop-container-detached`