Service sessions as concurrent parts

Version 081024

ICU 5

INF 5150

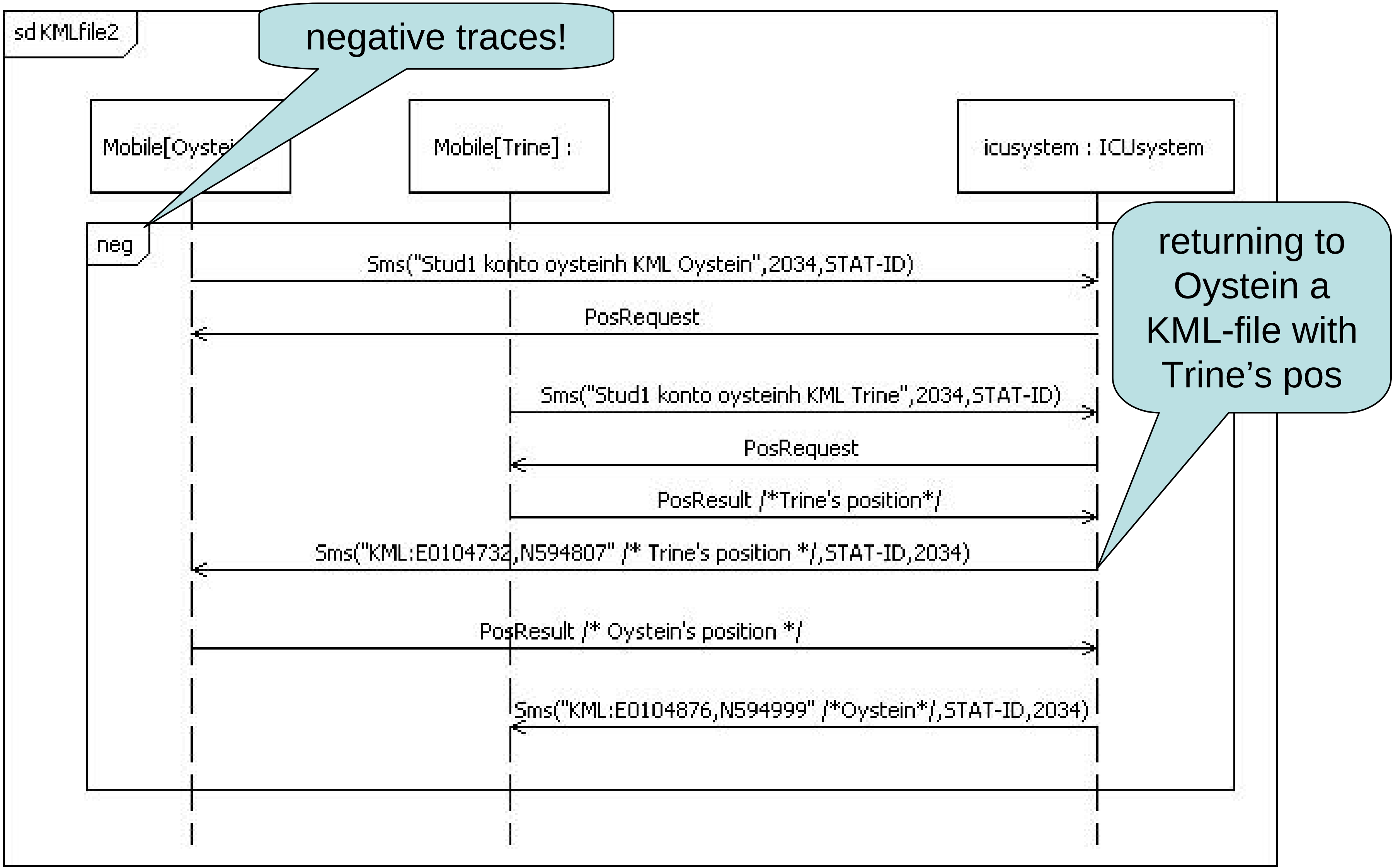
Motivation

- Assume having several users using ICU concurrently
  - The system could try and handle one user at the time
  - The system could try and handle everybody at the same time, but keep their data apart
- Some things take real time outside the ICU system
  - Users thinking
  - Positioning
  - SMS forwarding
- Potentially
  - Handling all users ”at the same time” may gain overall throughput

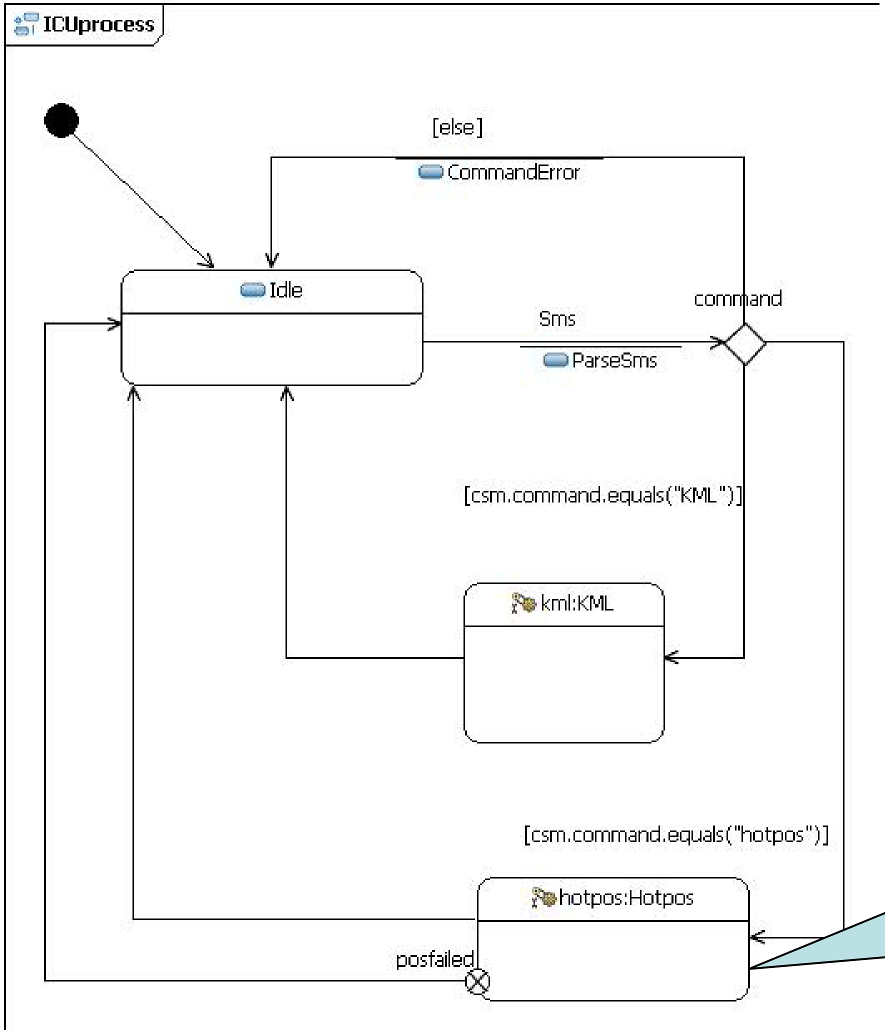
Risks

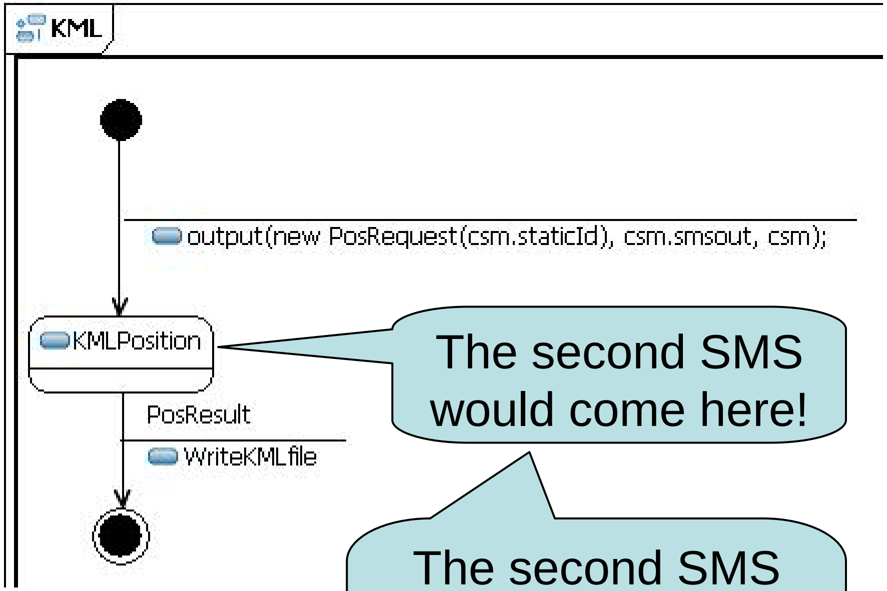
- The ICU system confuses which user has which position
- The ICU system returns SMS'es to the wrong user
- Coordinates are garbled
  - x-coordinate from one user and y-coordinate from another

This should not happen ....


sd KMLfile2
negative traces!
Mobile[Oystei
Mobile[Trine] :
icusystem : ICUsystem
neg
returning to Oystein a KML-file with Trine's pos
Sms("Stud1 konto oysteinh KML Oystein",2034,STAT-ID)
PosRequest
Sms("Stud1 konto oysteinh KML Trine",2034,STAT-ID)
PosRequest
PosResult /*Trine's position*/
Sms("KML:E0104732,N594807" /* Trine's position */,STAT-ID,2034)
PosResult /* Oystein's position */
Sms("KML:E0104876,N594999" /*Oystein*/,STAT-ID,2034)

What would our current design do?



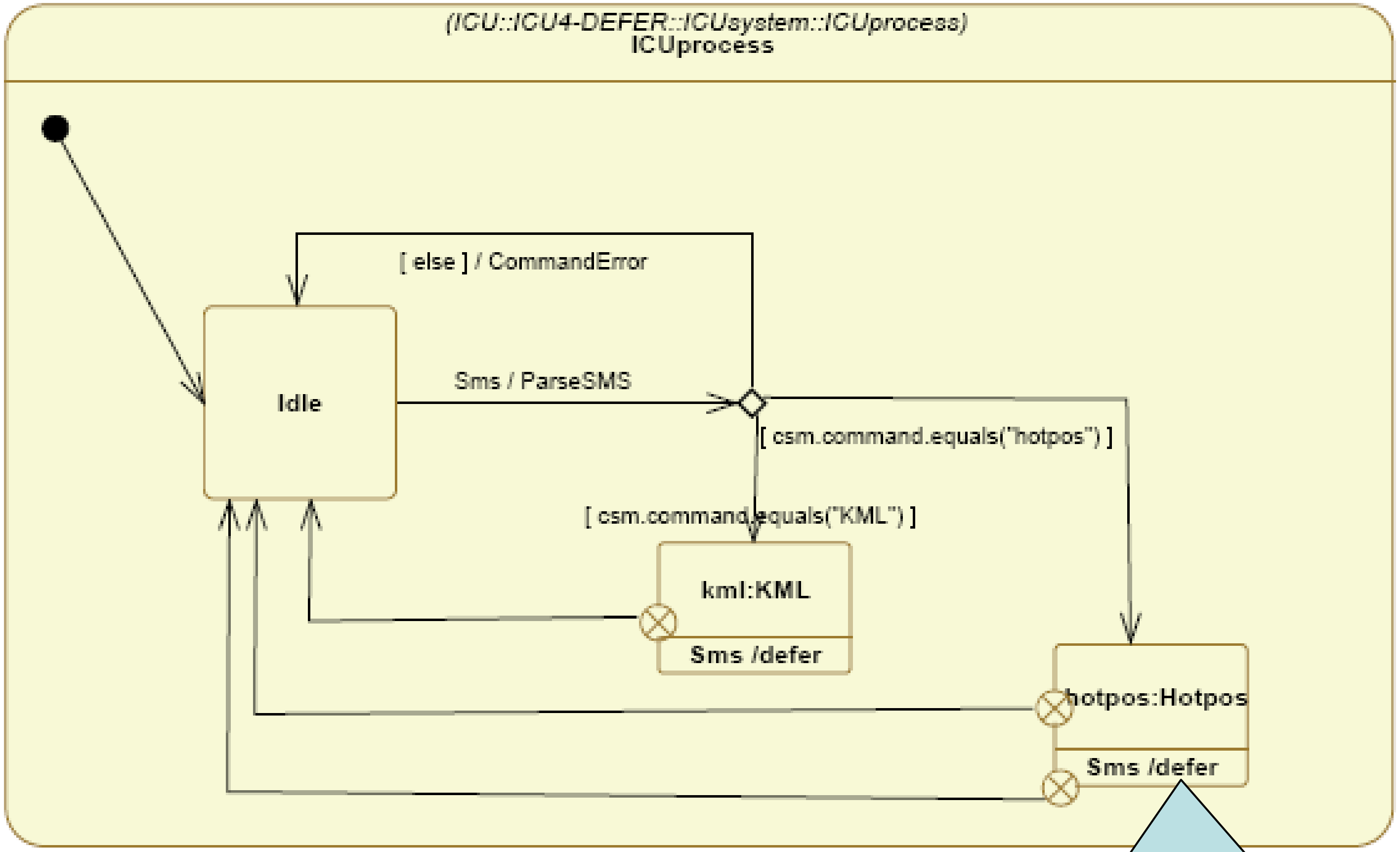


The second SMS would come here!

The second SMS would simply be discarded as a ”default transition”

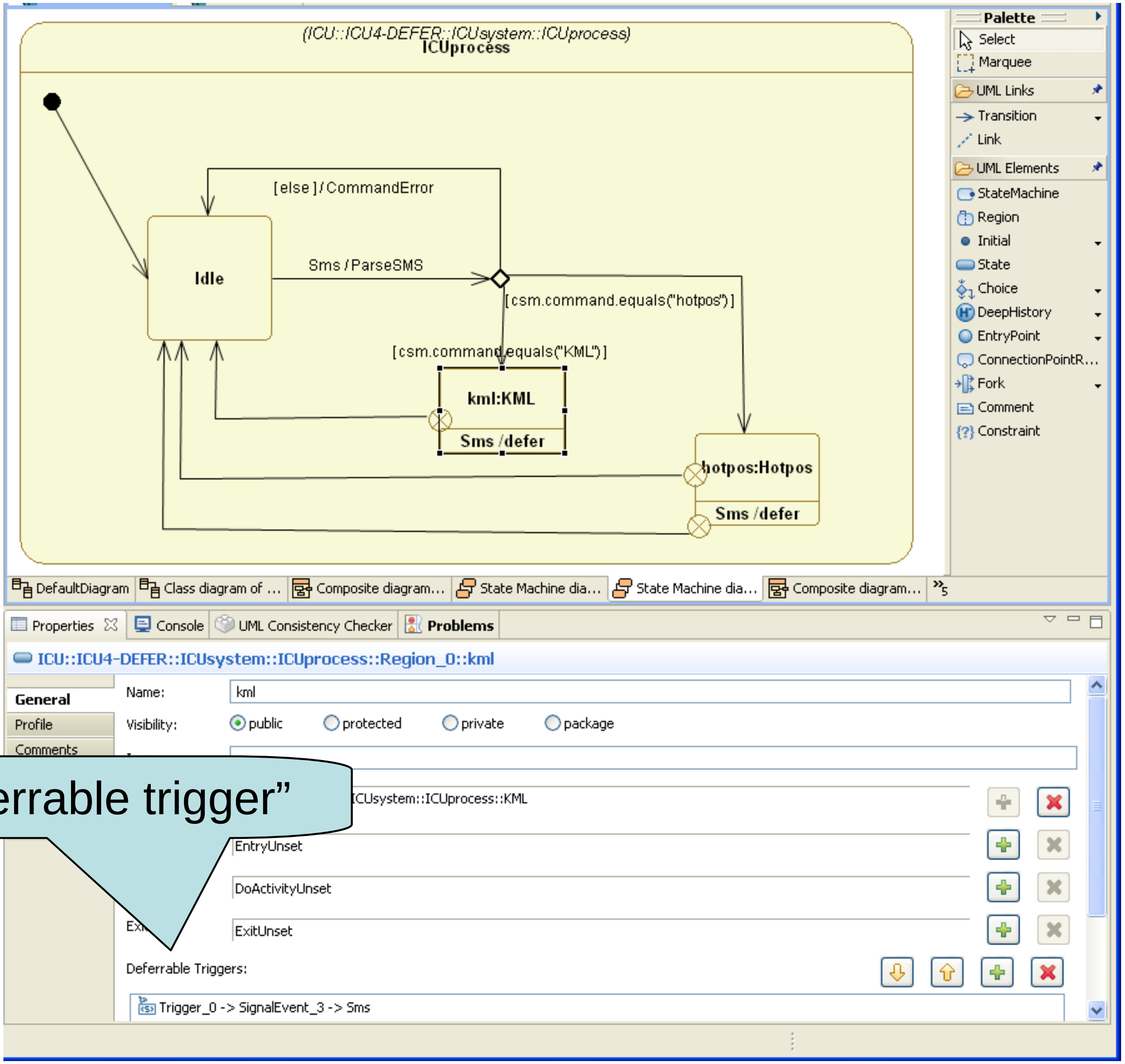
Introducing *defer* for *Sms* on every state but Idle, will cause services to be performed in sequence

Defer in Papyrus can be shown in the state


(ICU::ICU4-DEFER::ICUsystem::ICUprocess)
ICUprocess
[else] / CommandError
Idle
Sms / ParseSMS
[csm.command.equals("hotpos")]
[csm.command.equals("KML")]
kml:KML
Sms /defer
hotpos:Hotpos
Sms /defer

Making SMS a "signal trigger" on a transition

In the Papyrus tool



ICU::ICU4-DEFER::ICUsystem::ICUprocess::Region_0::kml

Name: kml

Visibility: ● public ○ protected ○ private ○ package

ICUsystem::ICUprocess::KML

EntryUnset

DoActivityUnset

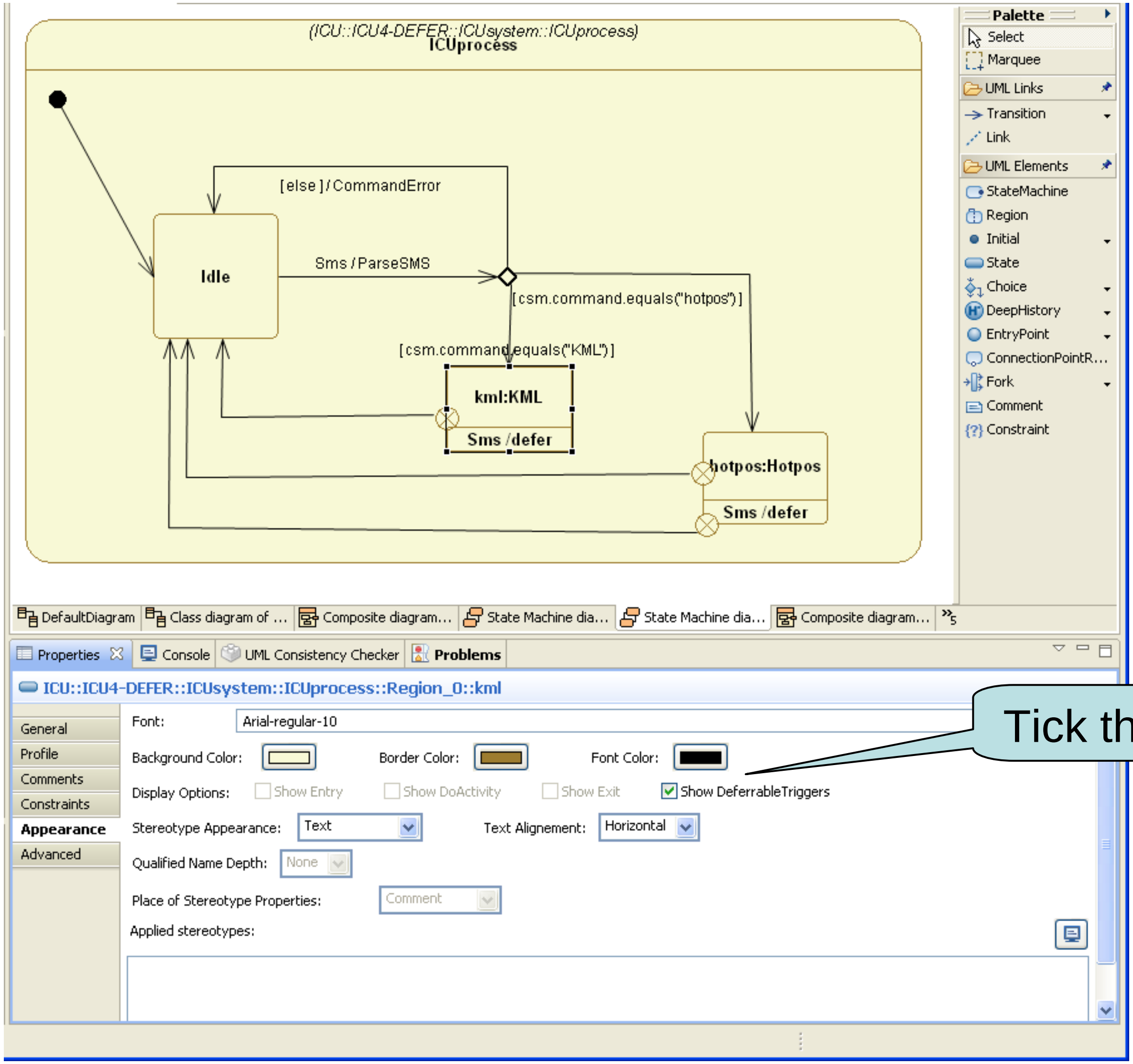
Ex ExitUnset

Deferrable Triggers:

Trigger_0 -> SignalEvent_3 -> Sms

Defining a ”deferrable trigger”

... and how to make it appear



ICU::ICU4-DEFER::ICUsystem::ICUprocess::Region_0::kml

Font: Arial-regular-10

Background Color: ▭ Border Color: ▬ Font Color: ■

Display Options: □ Show Entry □ Show DoActivity □ Show Exit ☑ Show DeferrableTriggers

Stereotype Appearance: Text Text Alignement: Horizontal

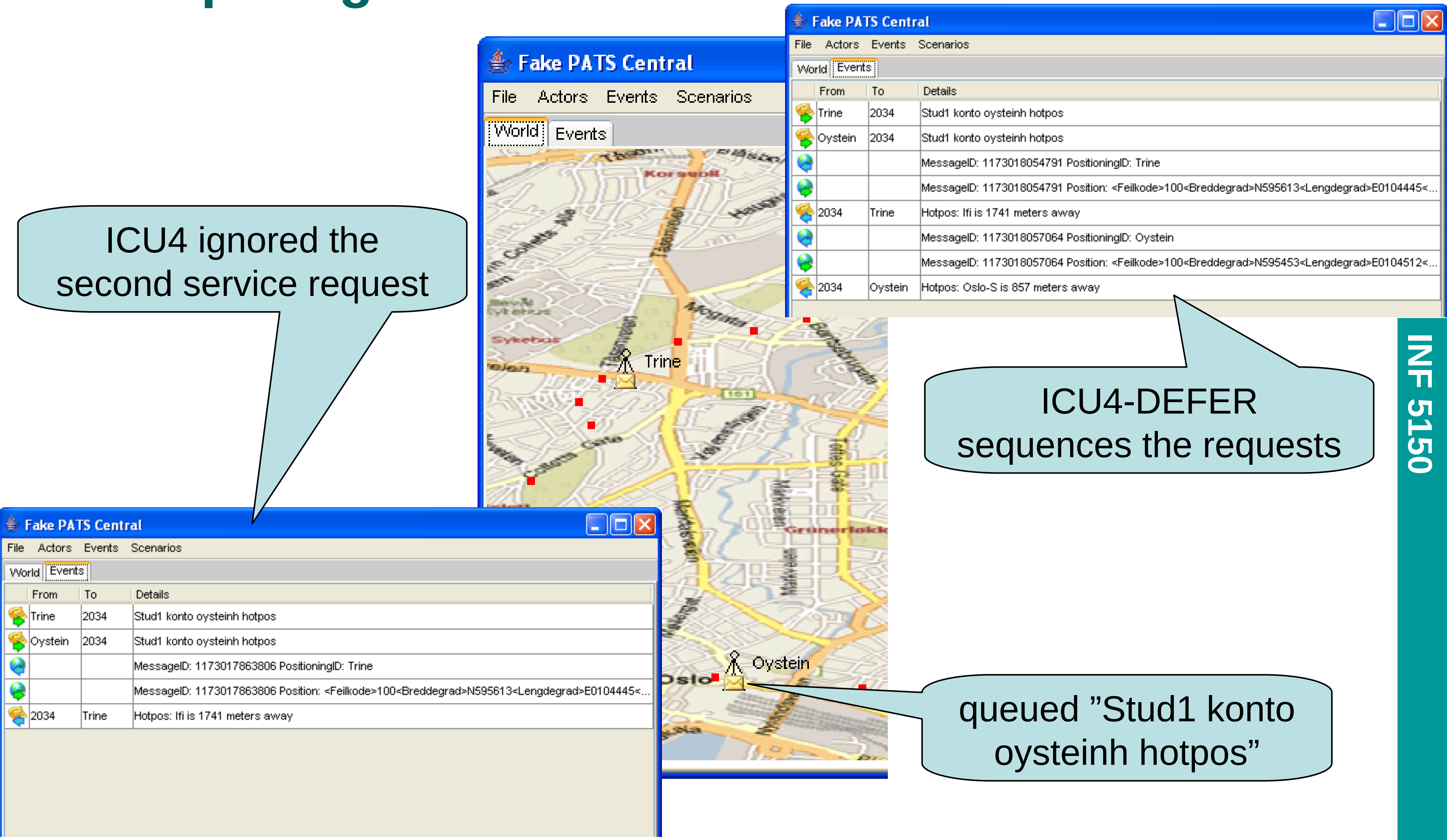
Qualified Name Depth: None

Place of Stereotype Properties: Comment

Applied stereotypes:

Tick this!

Comparing ICU4 and ICU4-DEFER



ICU4 ignored the second service request

	From	To	Details
	Trine	2034	Stud1 konto oysteinh hotpos
	Oystein	2034	Stud1 konto oysteinh hotpos
			MessageID: 1173017863806 PositioningID: Trine
			MessageID: 1173017863806 Position: <Feilkode>100<Breddegrad>N595613<Lengdegrad>E0104445<...
	2034	Trine	Hotpos: Ifi is 1741 meters away

ICU4-DEFER sequences the requests

	From	To	Details
	Trine	2034	Stud1 konto oysteinh hotpos
	Oystein	2034	Stud1 konto oysteinh hotpos
			MessageID: 1173018054791 PositioningID: Trine
			MessageID: 1173018054791 Position: <Feilkode>100<Breddegrad>N595613<Lengdegrad>E0104445<...
	2034	Trine	Hotpos: Ifi is 1741 meters away
			MessageID: 1173018057064 PositioningID: Oystein
			MessageID: 1173018057064 Position: <Feilkode>100<Breddegrad>N595453<Lengdegrad>E0104512<...
	2034	Oystein	Hotpos: Oslo-S is 857 meters away

queued ”Stud1 konto oysteinh hotpos”

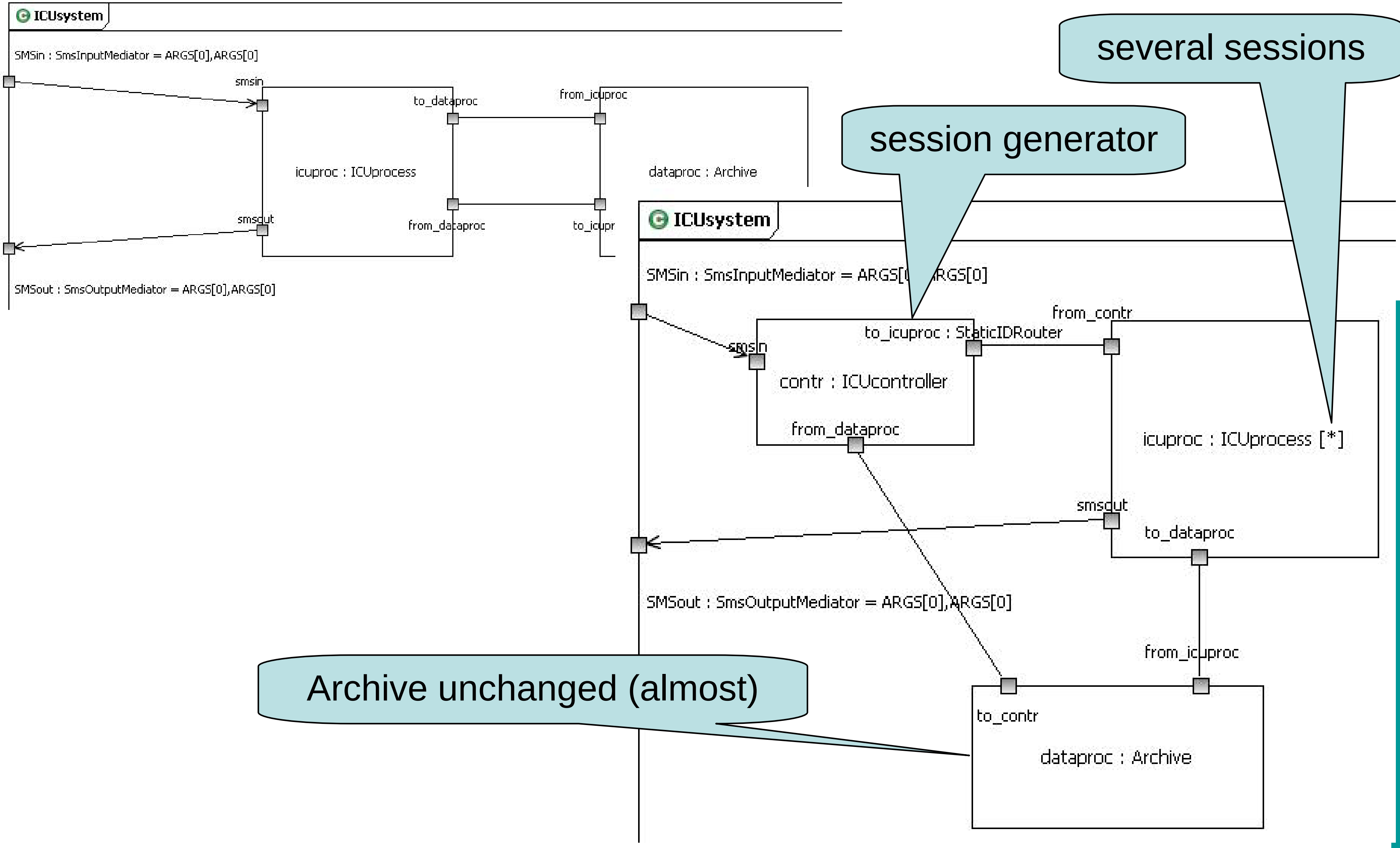
The ”session” solution

- Each initiative by a user is represented by an instantiation of a state machine (a session)
  - with all the temporary data associated with that user
  - taking care of all the communication related to that user
- The session is generated when the user initiates a service
- The session is terminated when the service is finished

Buzzzzz Groups (5 minutes)

- Discuss what represents sessions in the ICU systems
- Discuss what could represent sessions in ”TaskSolvers”
- Determine what should identify a session of the ICU system
- Determine what could identify a session in ”TaskSolvers”
- What would we need to make sessions come alive starting from ICU4?

A new composite structure

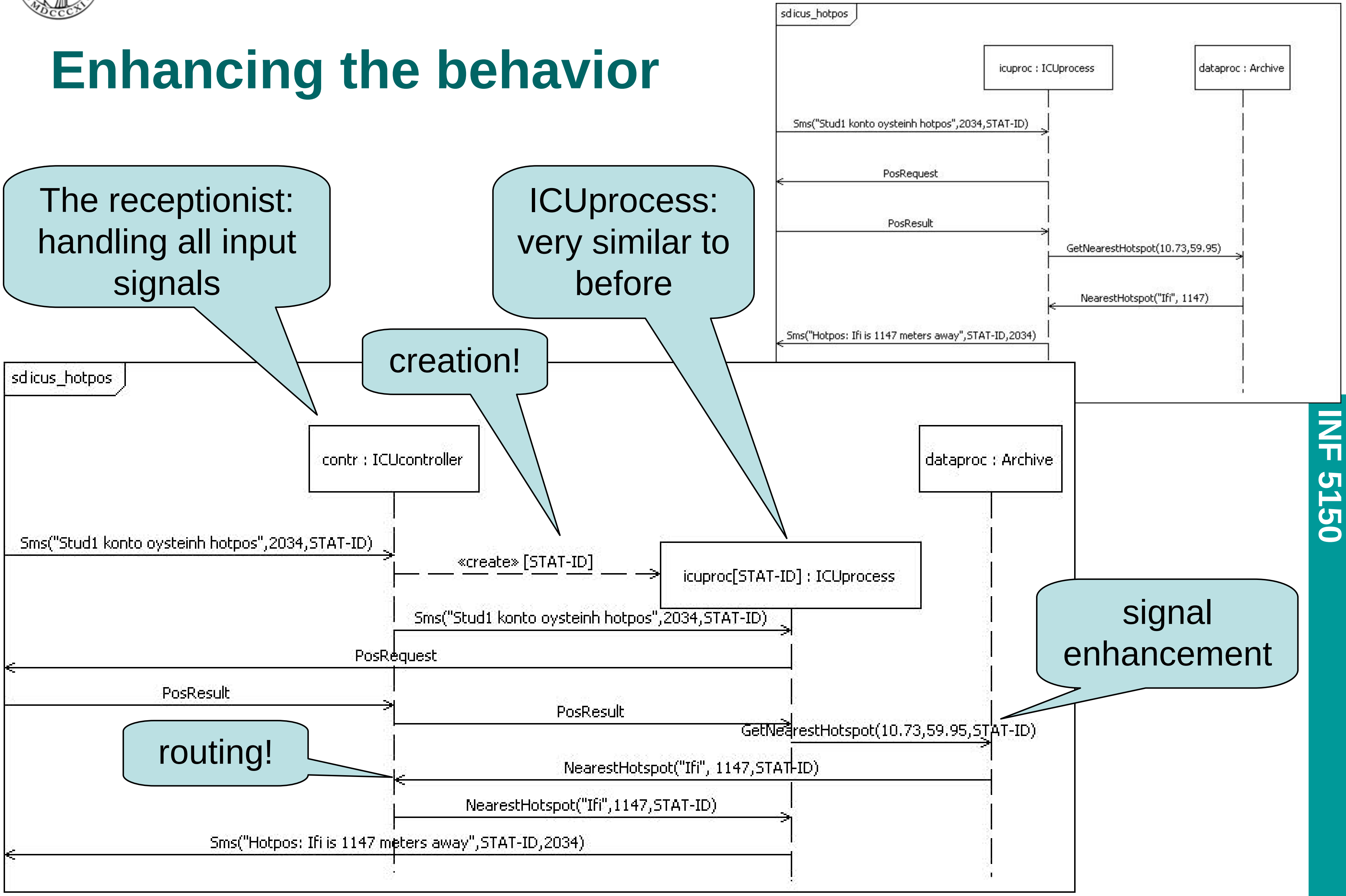


several sessions

session generator

Archive unchanged (almost)

Enhancing the behavior


sd icus_hotpos
icuproc : ICUprocess
dataproc : Archive
Sms("Stud1 konto oysteinh hotpos",2034,STAT-ID)
PosRequest
PosResult
GetNearestHotspot(10.73,59.95)
NearestHotspot("Ifi", 1147)
Sms("Hotpos: Ifi is 1147 meters away",STAT-ID,2034)
sd icus_hotpos
contr : ICUcontroller
dataproc : Archive
Sms("Stud1 konto oysteinh hotpos",2034,STAT-ID)
«create» [STAT-ID]
icuproc[STAT-ID] : ICUprocess
Sms("Stud1 konto oysteinh hotpos",2034,STAT-ID)
PosRequest
PosResult
PosResult
GetNearestHotspot(10.73,59.95,STAT-ID)
NearestHotspot("Ifi", 1147,STAT-ID)
NearestHotspot("Ifi",1147,STAT-ID)
Sms("Hotpos: Ifi is 1147 meters away",STAT-ID,2034)

The receptionist: handling all input signals

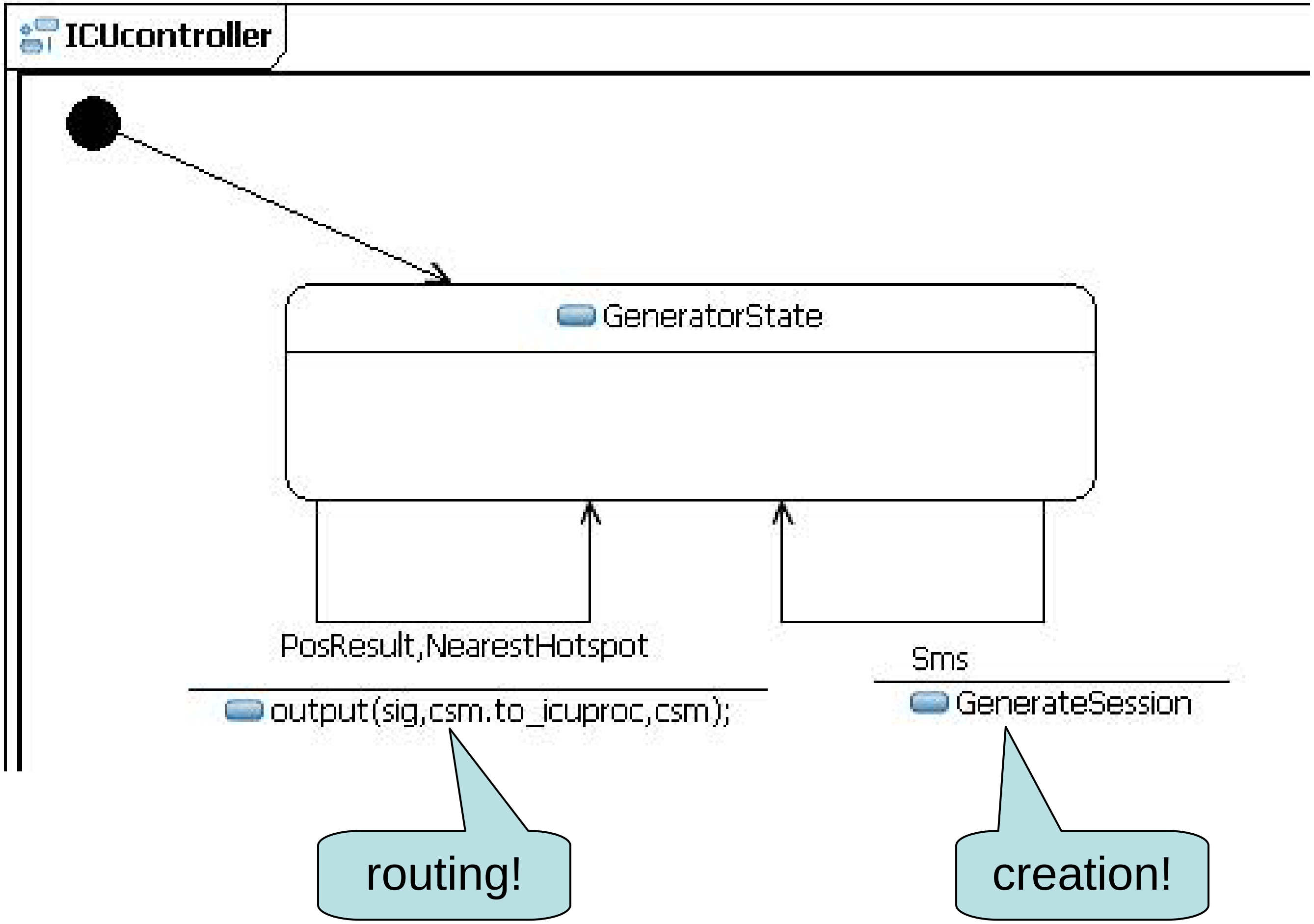
ICUprocess: very similar to before

creation!

signal enhancement

routing!

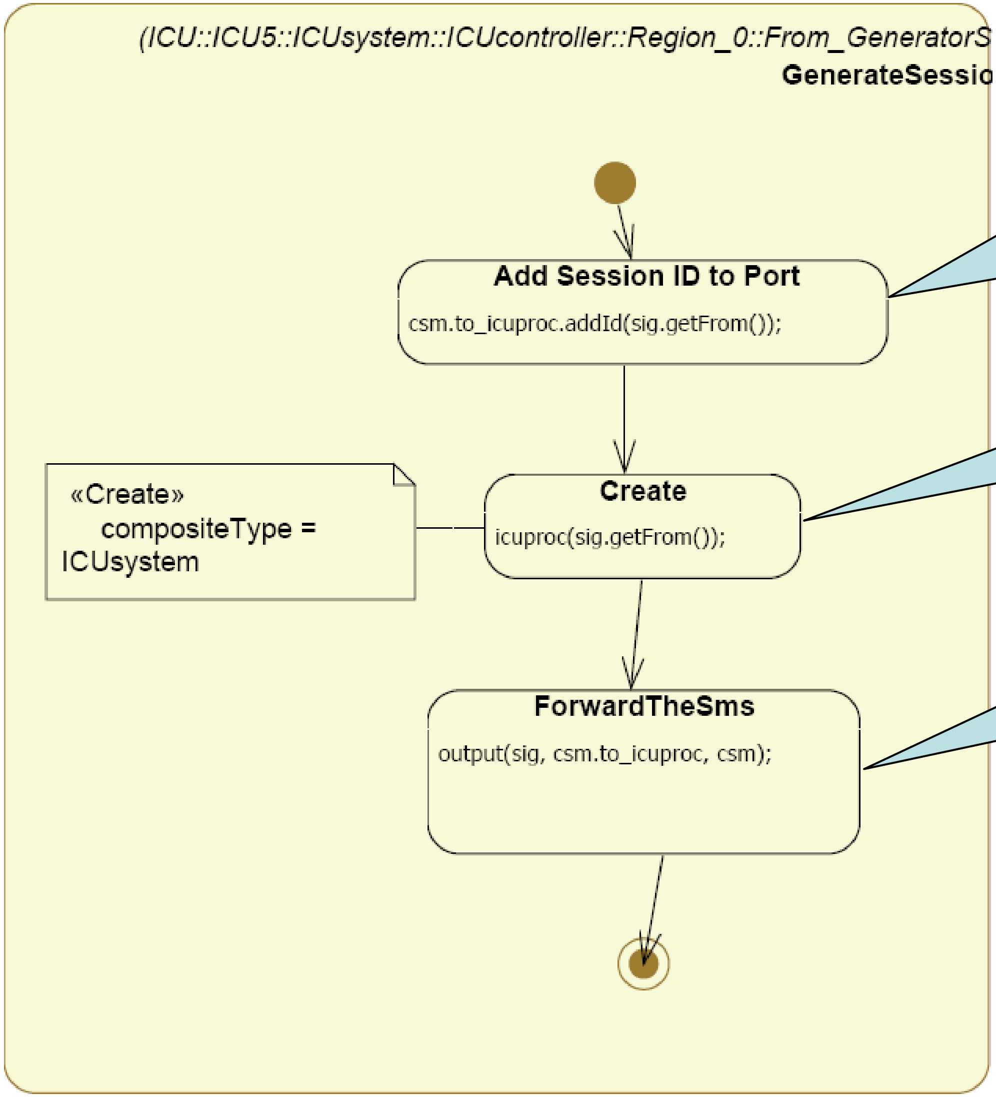
ICUcontroller



routing!

creation!

Creating a session



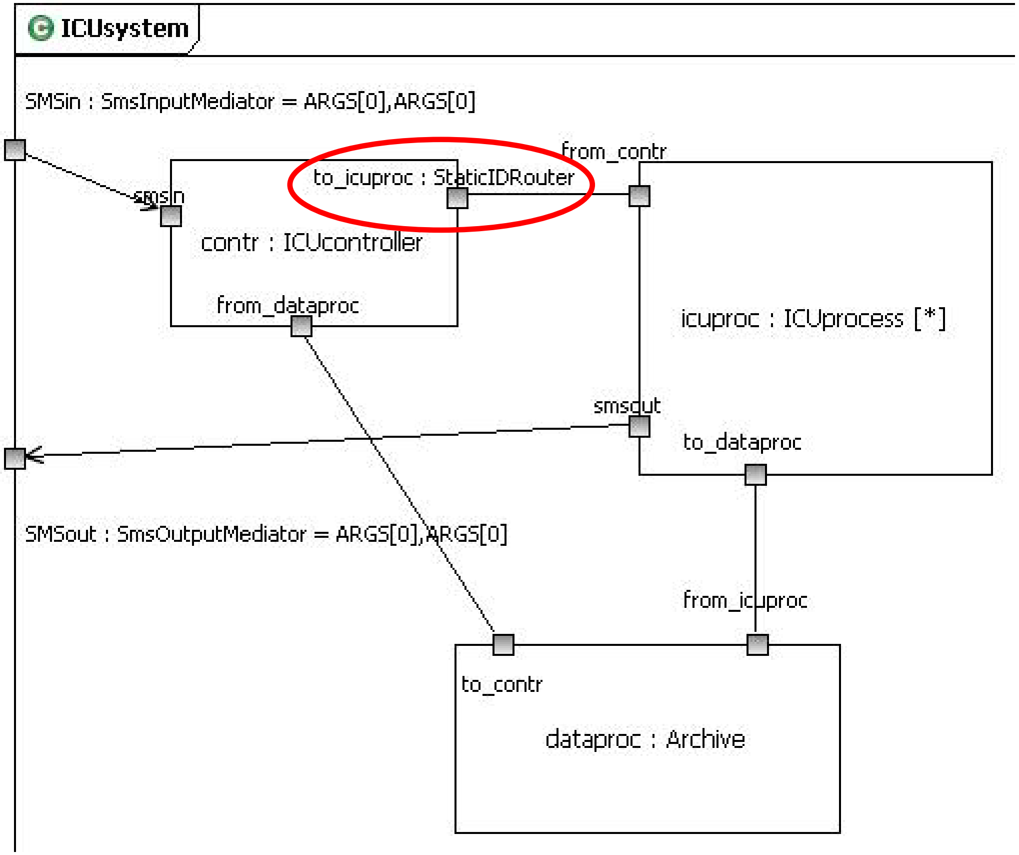
```
csm.to_icuproc.addId(sig.getFrom());
icuproc(sig.getFrom());
output(sig, csm.to_icuproc, csm);
```

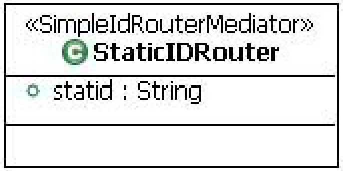
routing information so that the new member to come will ever be found

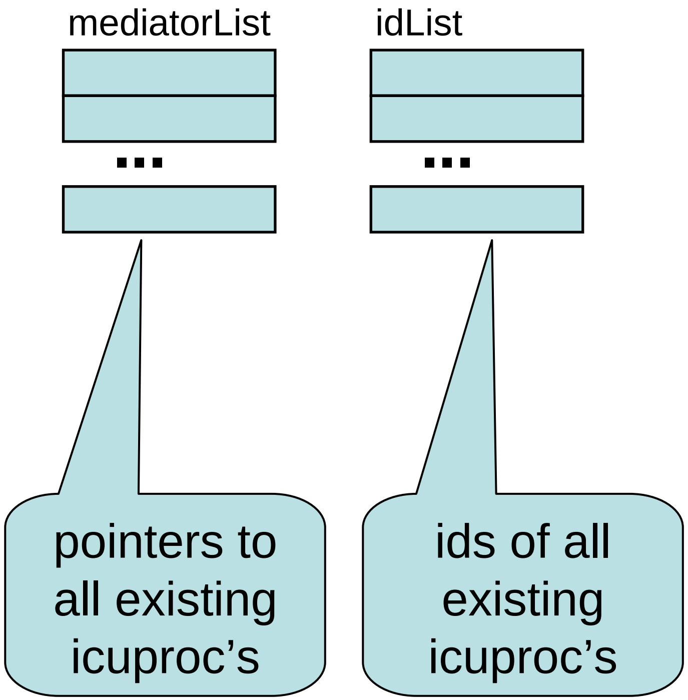
create a new member of the *icuproc* set of parts

sending a message to the generated process

Simple Routing (1) One-to-many Port



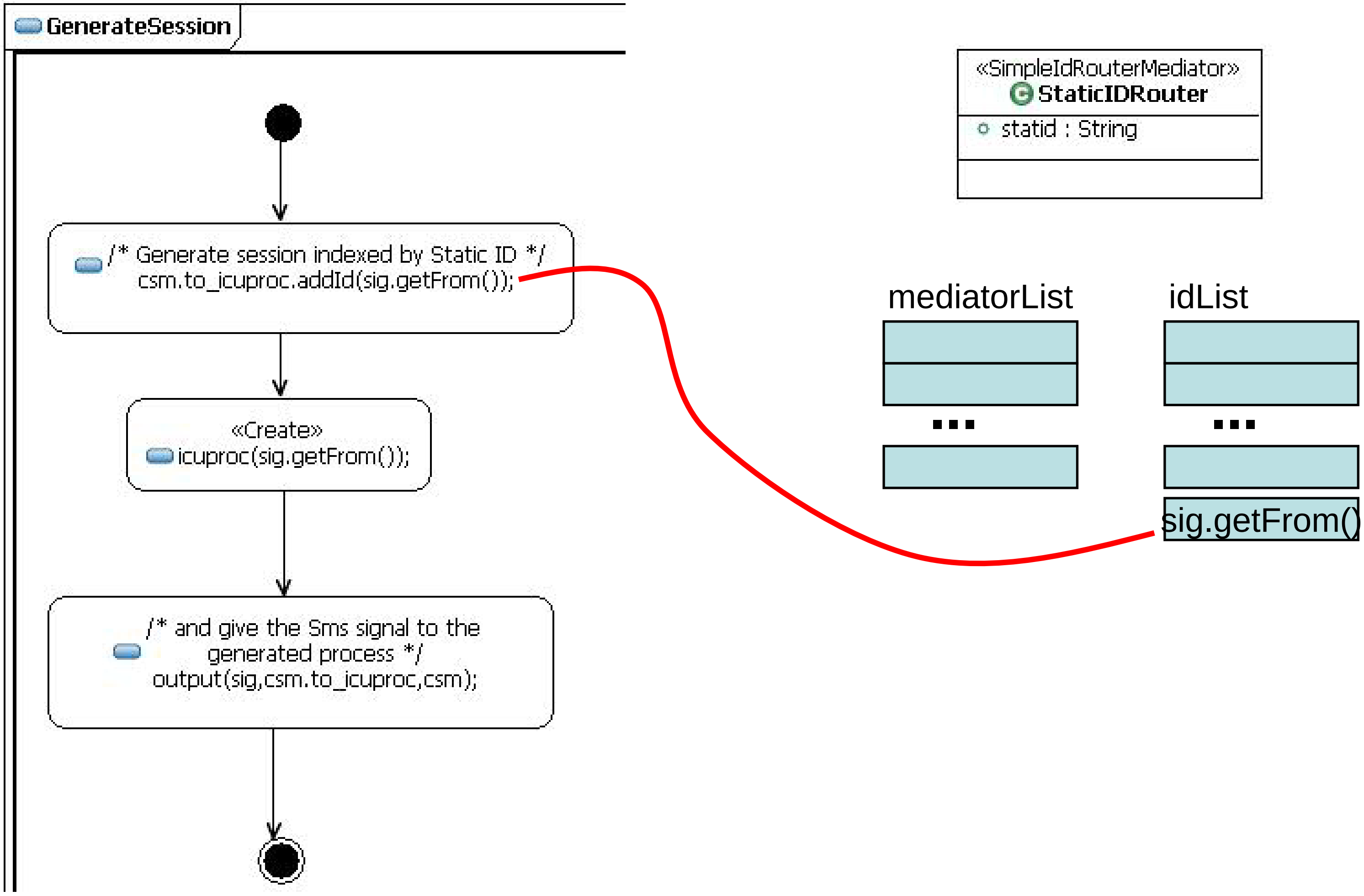




pointers to all existing icuproc’s

ids of all existing icuproc’s

Simple Routing (2) Adding the ID



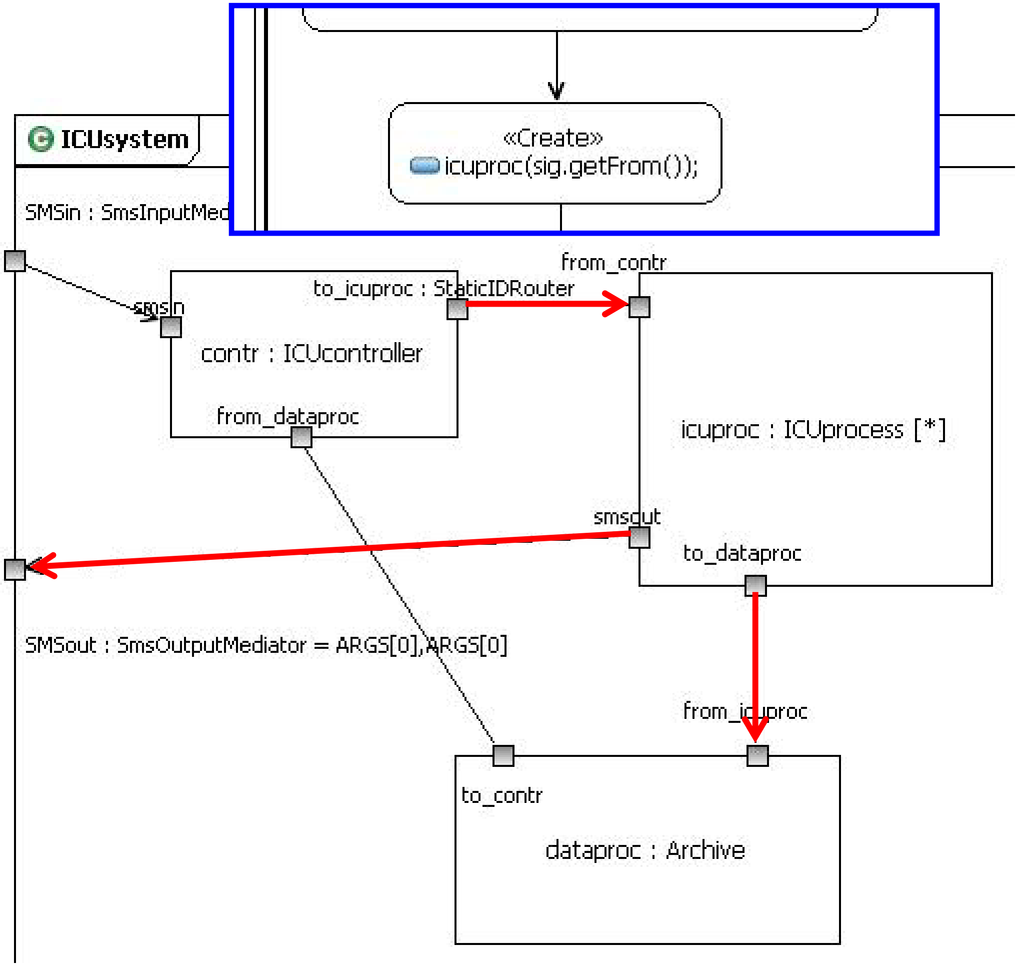
GenerateSession

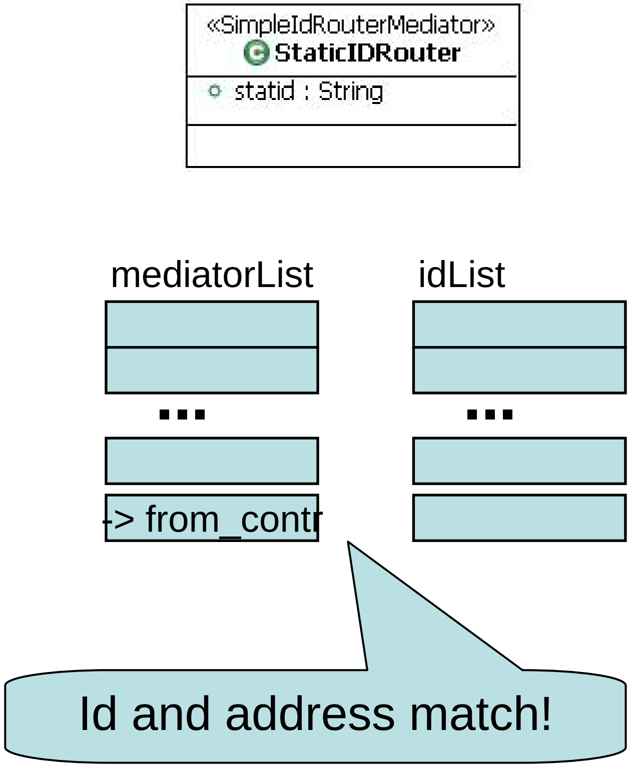
```
/* Generate session indexed by Static ID */
csm.to_icuproc.addId(sig.getFrom());
```

```
«Create»
icuproc(sig.getFrom());
```

```
/* and give the Sms signal to the
generated process */
output(sig,csm.to_icuproc,csm);
```

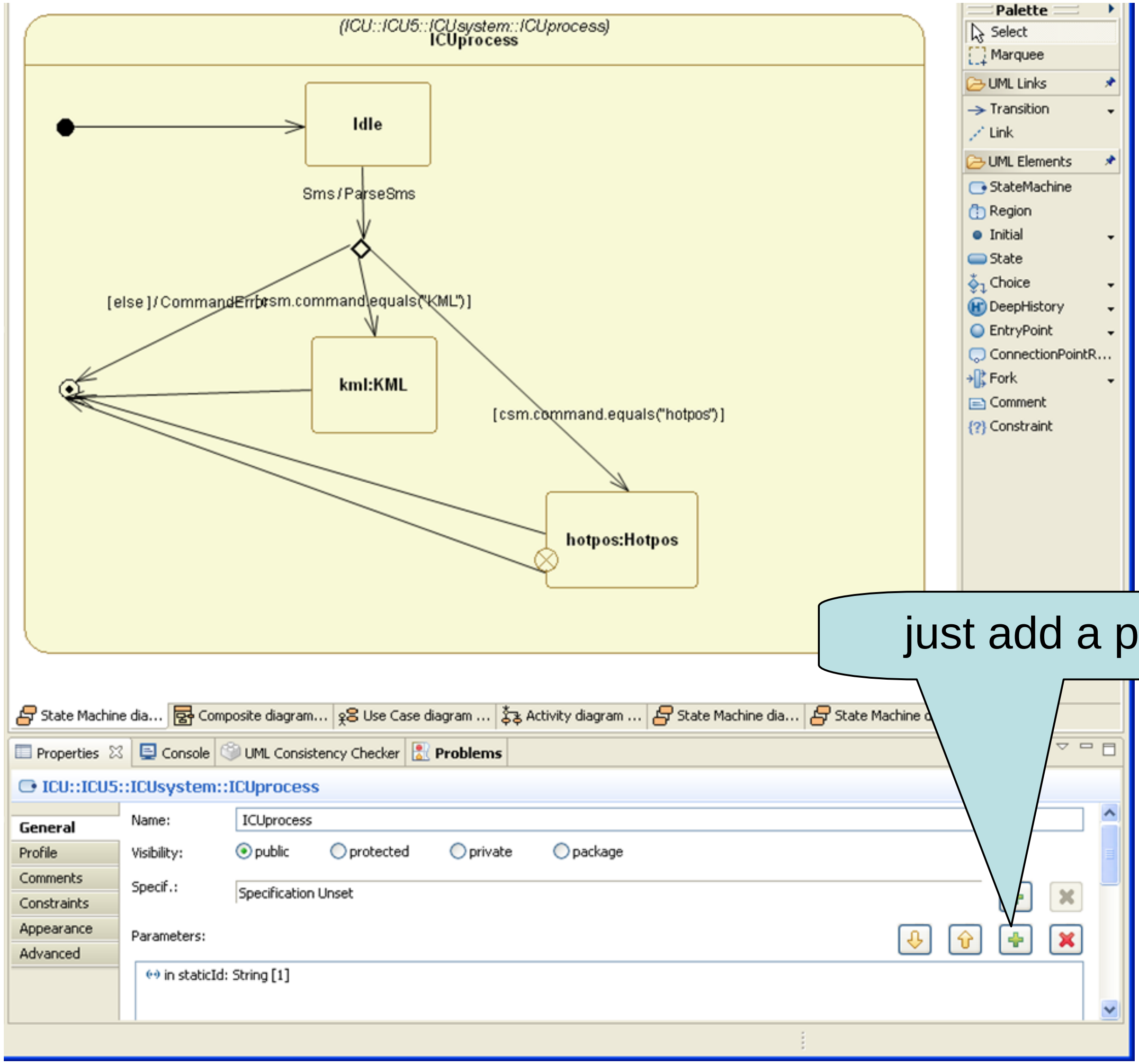
Simple Routing (3) Connecting connectors





Id and address match!

Adding a parameter to the dynamic process



ICU::ICU5::ICUsystem::ICUprocess

Name: ICUprocess

Visibility: ● public ○ protected ○ private ○ package

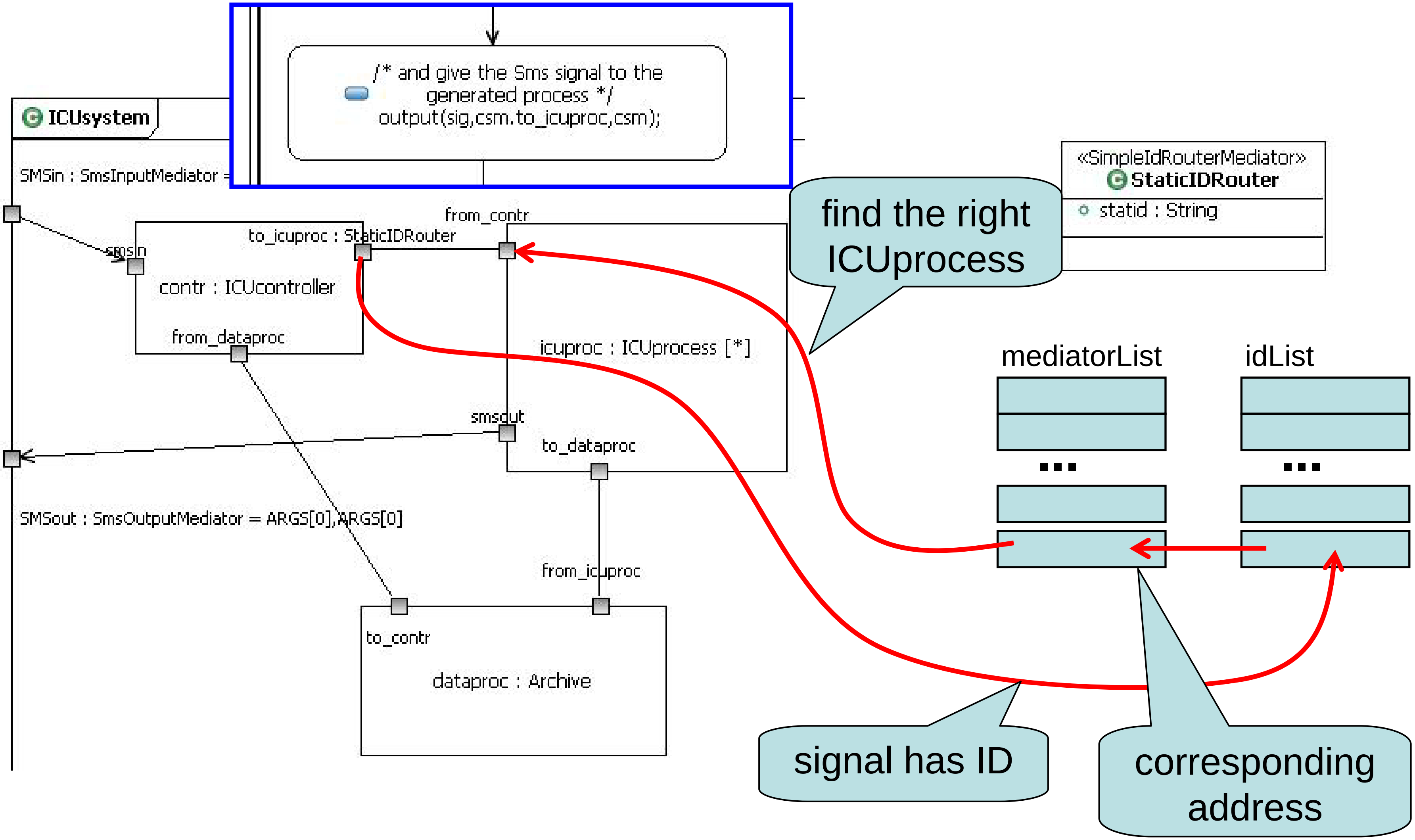
Specif.: Specification Unset

Parameters:

in staticId: String [1]

just add a parameter

Simple Routing (4) Forwarding from Port



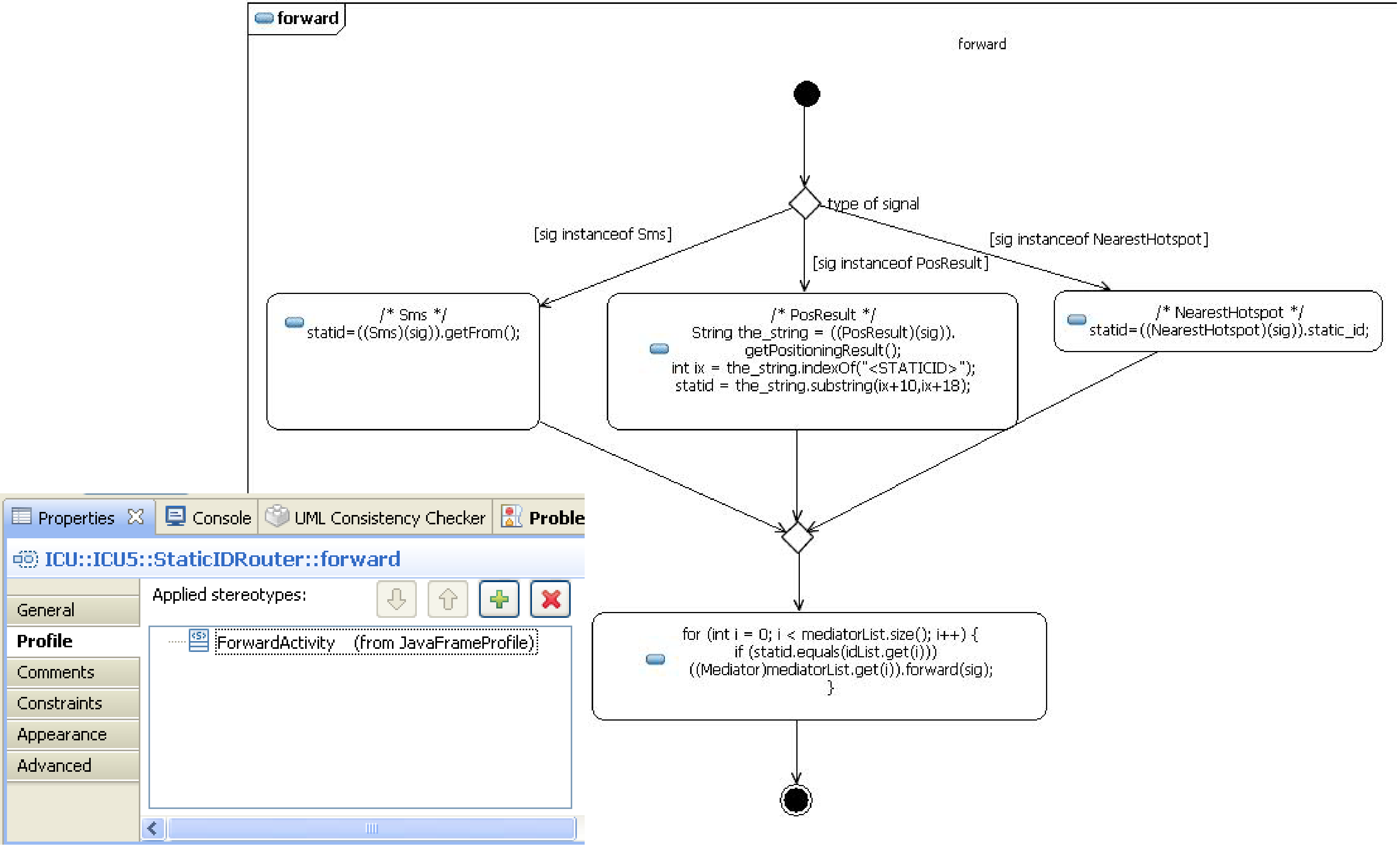
```
/* and give the Sms signal to the
generated process */
output(sig,csm.to_icuproc,csm);
```

find the right ICUprocess

signal has ID

corresponding address

Simple Routing (5) forward() is programmed!



```
/* Sms */
statid=((Sms)(sig)).getFrom();
```

```
/* PosResult */
String the_string = ((PosResult)(sig)).
getPositioningResult();
int ix = the_string.indexOf("<STATICID>");
statid = the_string.substring(ix+10,ix+18);
```

```
/* NearestHotspot */
statid=((NearestHotspot)(sig)).static_id;
```

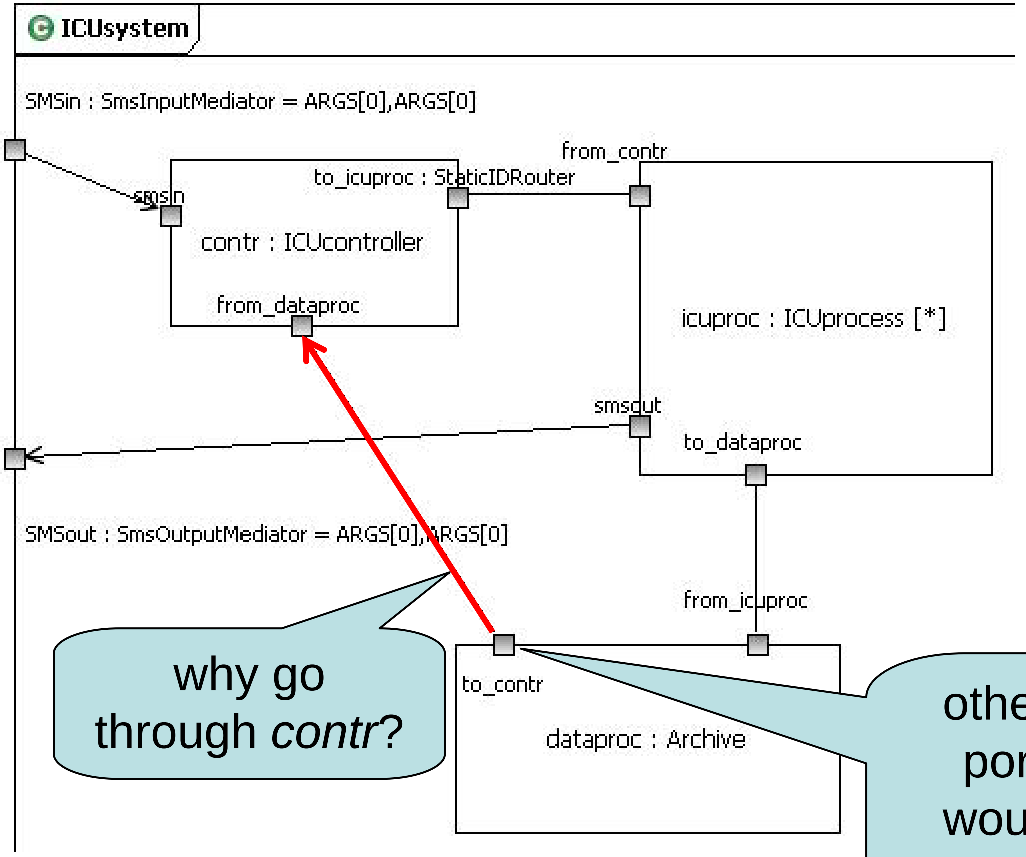
```
for (int i = 0; i < mediatorList.size(); i++) {
  if (statid.equals(idList.get(i)))
  ((Mediator)mediatorList.get(i)).forward(sig);
}
```

ICU::ICU5::StaticIDRouter::forward

Applied stereotypes:

ForwardActivity (from JavaFrameProfile)

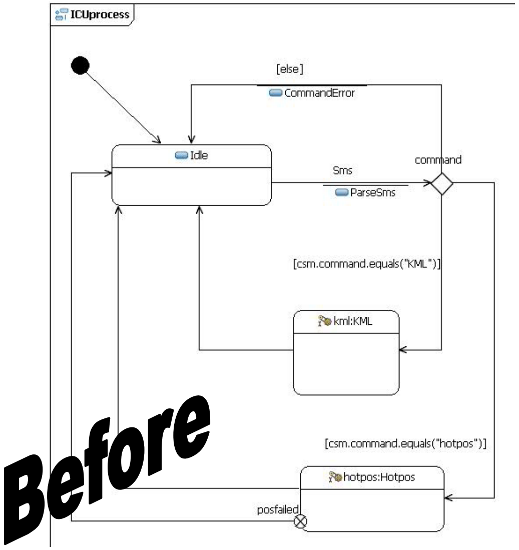
Simple Routing (6) The routing central

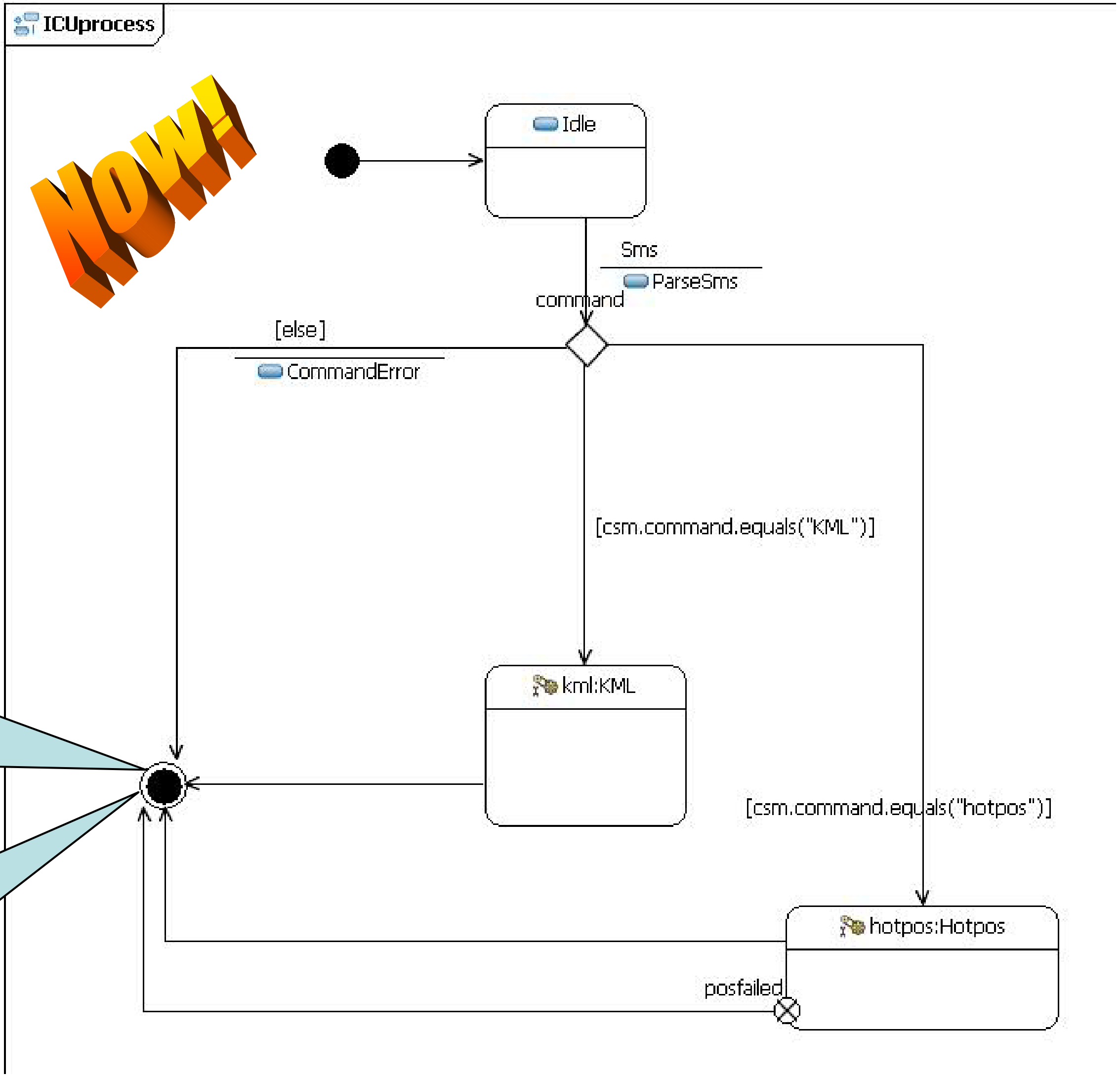


why go through *contr*?

otherwise the output port from *dataproc* would have to route; our approach is simpler

Terminating a session



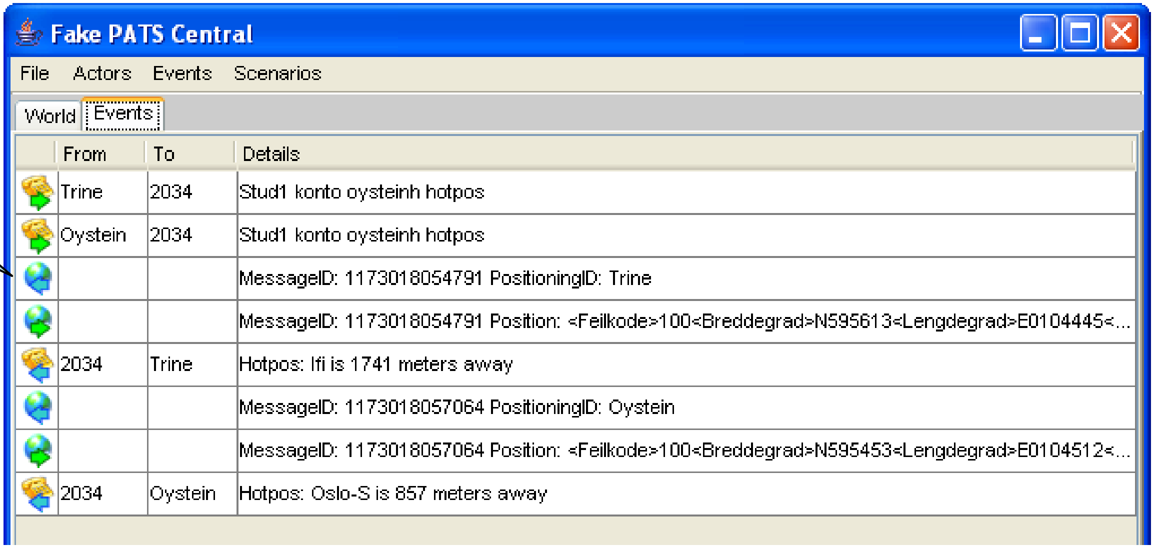


At the final state the *ICUprocess* representing the session will terminate

The compiler and JavaFrame make sure that the implementation gets rid of the session

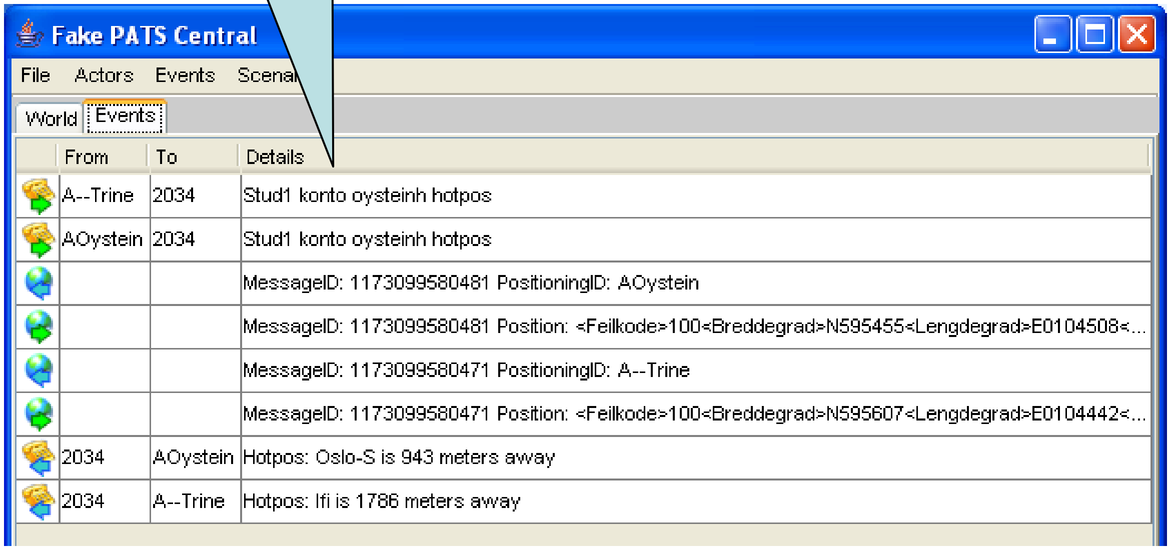
Executing ICU5 (with Sessions)

ICU4-DEFER: sequentialized



From	To	Details
Trine	2034	Stud1 konto oysteinh hotpos
Oystein	2034	Stud1 konto oysteinh hotpos
		MessageID: 1173018054791 PositioningID: Trine
		MessageID: 1173018054791 Position: <Feilkode>100<Breddegrad>N595613<Lengdegrad>E0104445<...
2034	Trine	Hotpos: Ifi is 1741 meters away
		MessageID: 1173018057064 PositioningID: Oystein
		MessageID: 1173018057064 Position: <Feilkode>100<Breddegrad>N595453<Lengdegrad>E0104512<...
2034	Oystein	Hotpos: Oslo-S is 857 meters away

ICU5: more concurrency



From	To	Details
A--Trine	2034	Stud1 konto oysteinh hotpos
AOystein	2034	Stud1 konto oysteinh hotpos
		MessageID: 1173099580481 PositioningID: AOystein
		MessageID: 1173099580481 Position: <Feilkode>100<Breddegrad>N595455<Lengdegrad>E0104508<...
		MessageID: 1173099580471 PositioningID: A--Trine
		MessageID: 1173099580471 Position: <Feilkode>100<Breddegrad>N595607<Lengdegrad>E0104442<...
2034	AOystein	Hotpos: Oslo-S is 943 meters away
2034	A--Trine	Hotpos: Ifi is 1786 meters away

Technicality: StaticID **must** be 8 chars

Summary of Sessions

- One session per concurrent user initiative
  - The state machine type *ICUprocess* describes the session
- One receptionist state machine creates the sessions
  - when the session initiation arrives
  - here: Sms-message
- Centralized routing through the receptionist *contr*
  - one routing port (SimpleIdRouterMediator)
  - all signals aiming for a session are sent through *contr*
- Terminating the session by reaching the final state
  - and the runtime system machinery takes care of the rest

New PapyrusIFIUML with improved JFDebug

- Help -> Manage Configuration ->
- Select: PapyrusIFIUML -> Scan for Updates ->
- Select last update and install

- Then to update the workspace with the proper JavaJars
  - Delete the JavaJars project
  - Create a dummy JavaFrame project
    - which will recreate the JavaJars project with the proper content