Sequence Diagrams – language and method

Version 090904

Sequence Diagrams

- Sequence Diagrams are
  - simple
  - powerful
  - readable
  - used to describe interaction sequences
- History
  - Has been used for a number of years informally
  - Standardized in 1992 in Z.120 (Message Sequence Charts - MSC)
  - Last major revision of MSC is from 1999 (called MSC-2000)
  - Formal semantics of MSC-96 is given in Z.120 Annex B
  - Included in UML from 1999, but in a rather simple variant
  - UML 2.2 http://www.uml.org/

Purpose

- Emphasizes the interaction between objects indicating that the interplay is the most important aspect
  - Often only a small portion of the total variety of behavior is described improve the individual understanding of an interaction problem
- Sequence Diagrams are used to ...
  - document protocol situations,
  - illustrate behavior situations,
  - verify interaction properties relative to a specification,
  - describe test cases,
  - document simulation traces.

The example context: Dolly Goes To Town

- Dolly is going to town and
  - wants to subscribe for bus schedules back home
  - given her current position
  - and the time of day.
  - The service should not come in effect until a given time in the evening

The informal architecture

positioning

position

Service Base

web terminal

local ads

ads

local cache

subscription

info

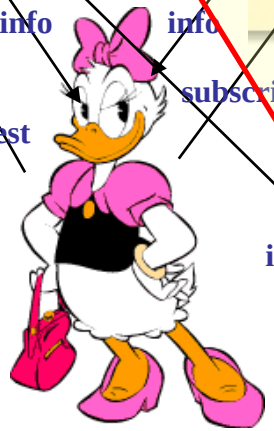
mobile terminal

request

info

info

subscription



user

info

information pusher (SMS sender, WAP portal)

info

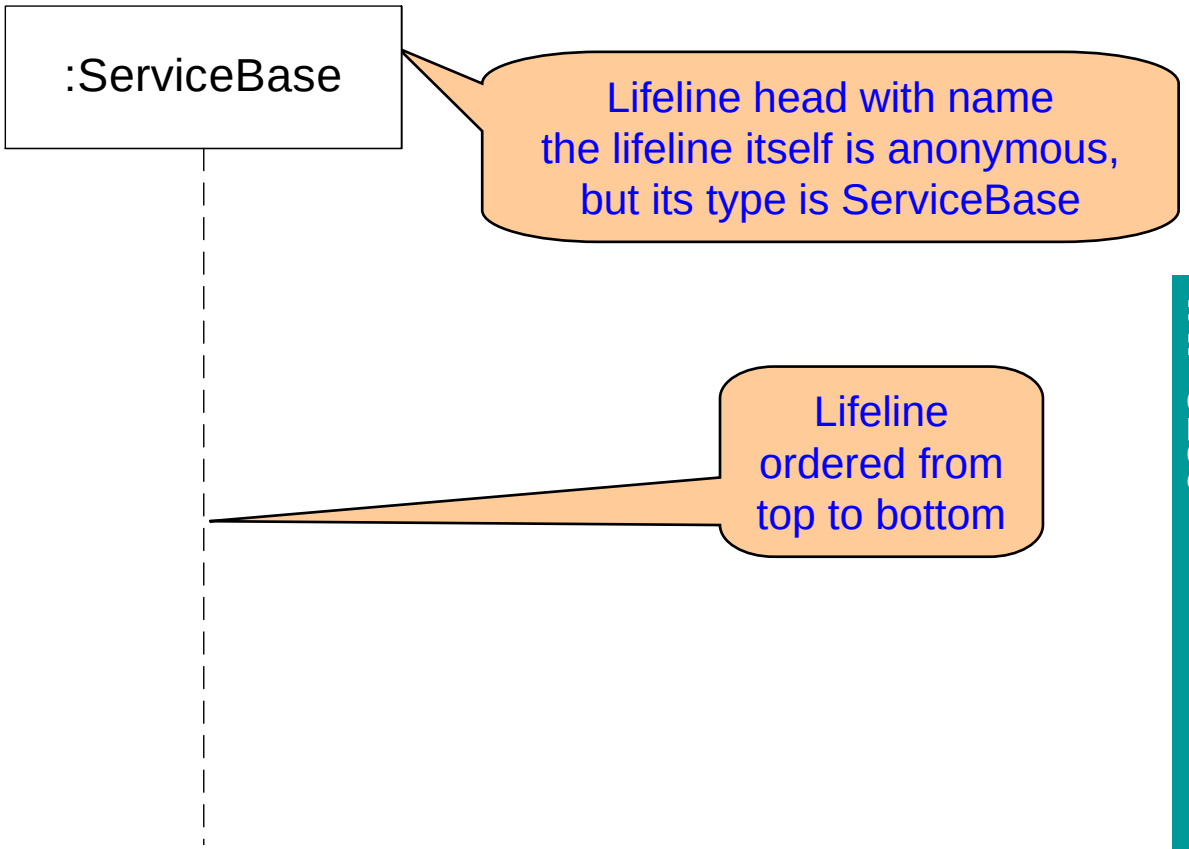
information provider

Service Terminal

Service User

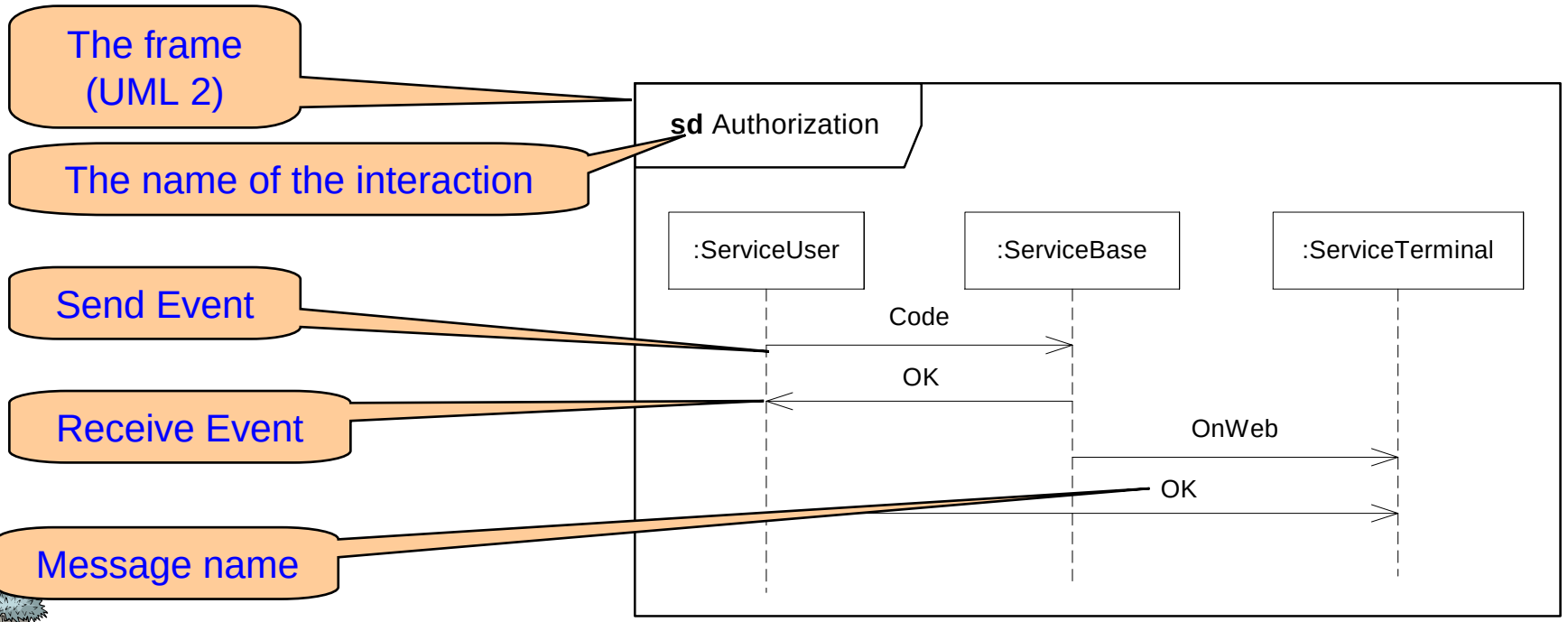
INF 5150

Lifeline – the “doers”



(Simple) Sequence Diagram

- Messages have one send event, and one receive event.
  - The send event must occur before the receive event.
  - The send event is the result of an Action
- Events are strictly ordered along a lifeline from top to bottom

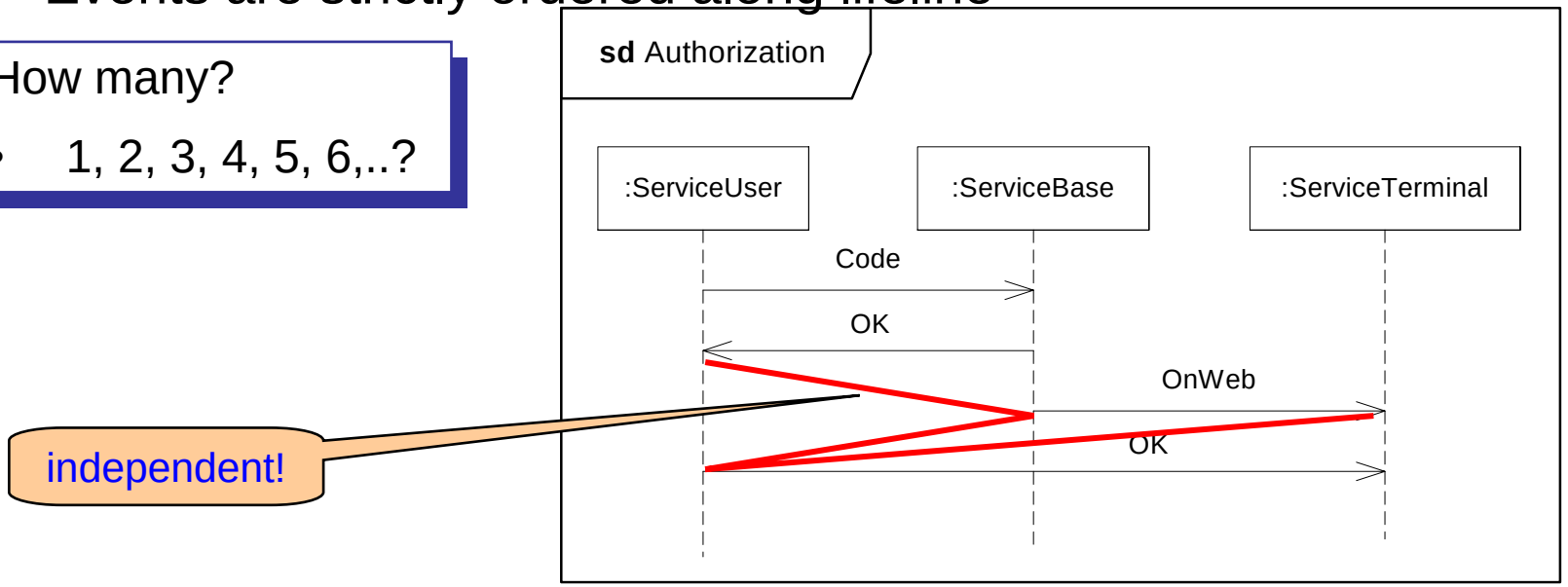


How many global traces are there in this diagram?

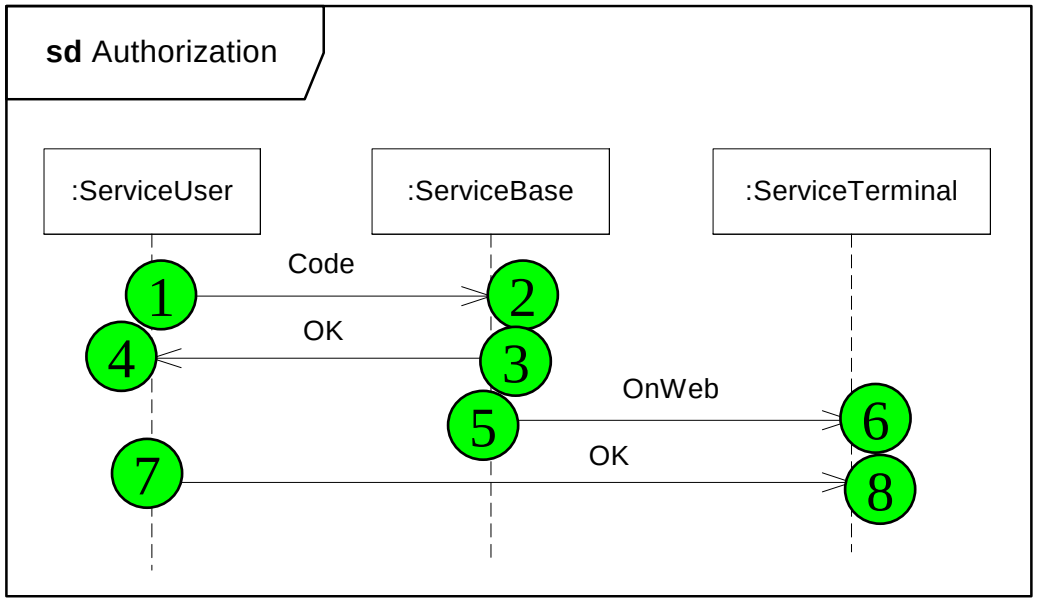
- The only invariants:
  - Messages have one send event, and one receive event. The send event must occur before the receive event.
  - Events are strictly ordered along lifeline

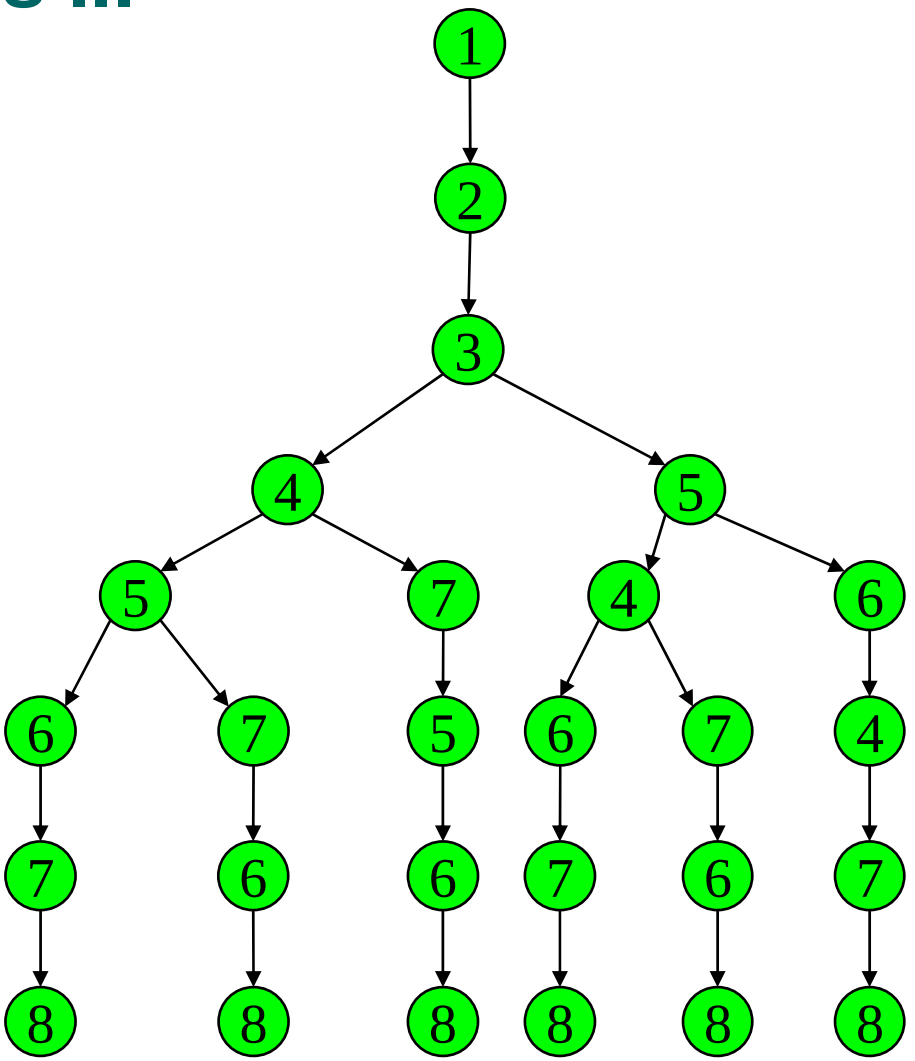
How many?

- 1, 2, 3, 4, 5, 6,..?



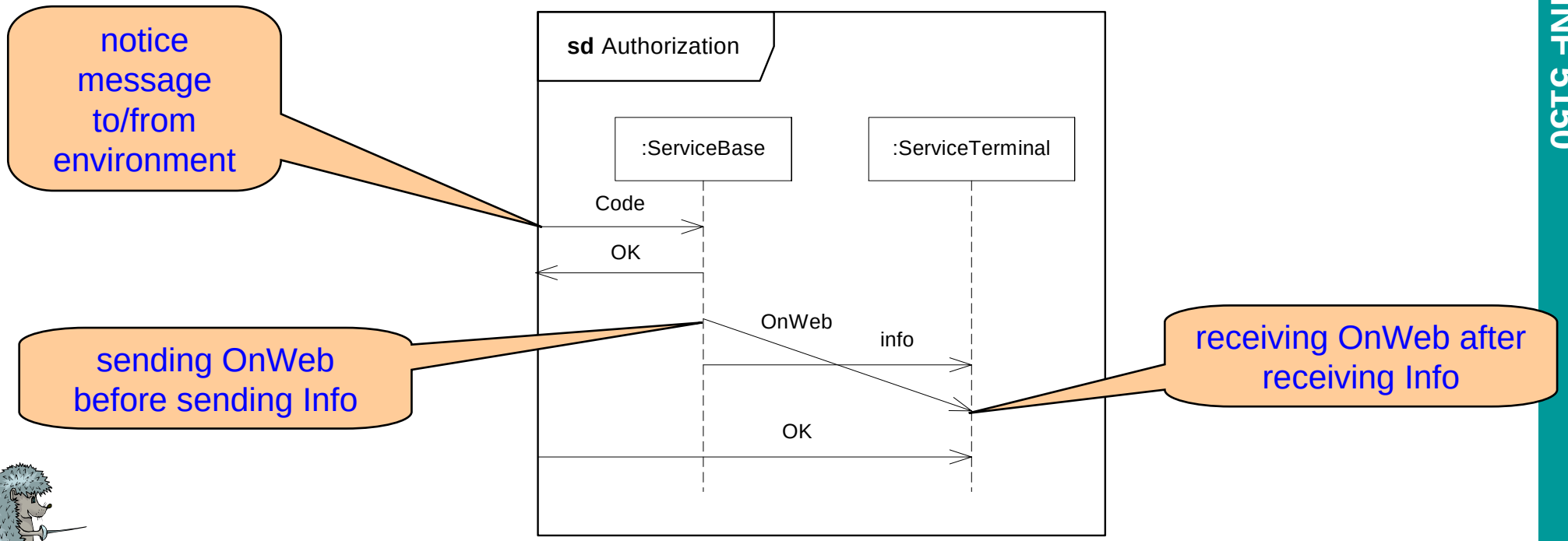
Really counting the traces ...





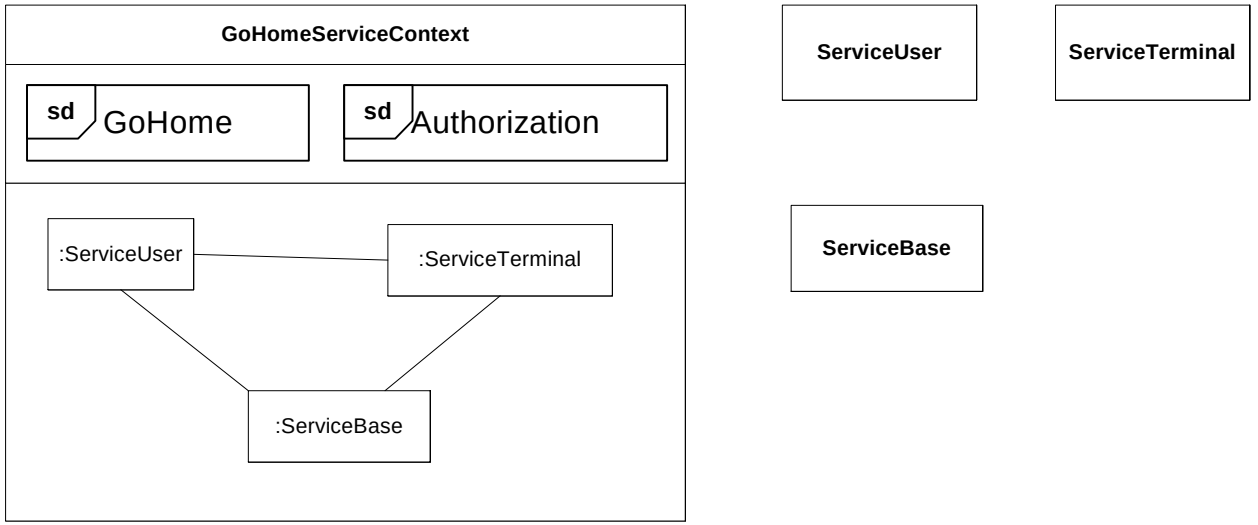
Asynchronous messages: Message Overtaking

- asynchronous communication = when the sender does not wait for the reply of the message sent
- Reception is normally interpreted as consumption of the message.
- When messages are asynchronous, it is important to be able to describe message overtaking.

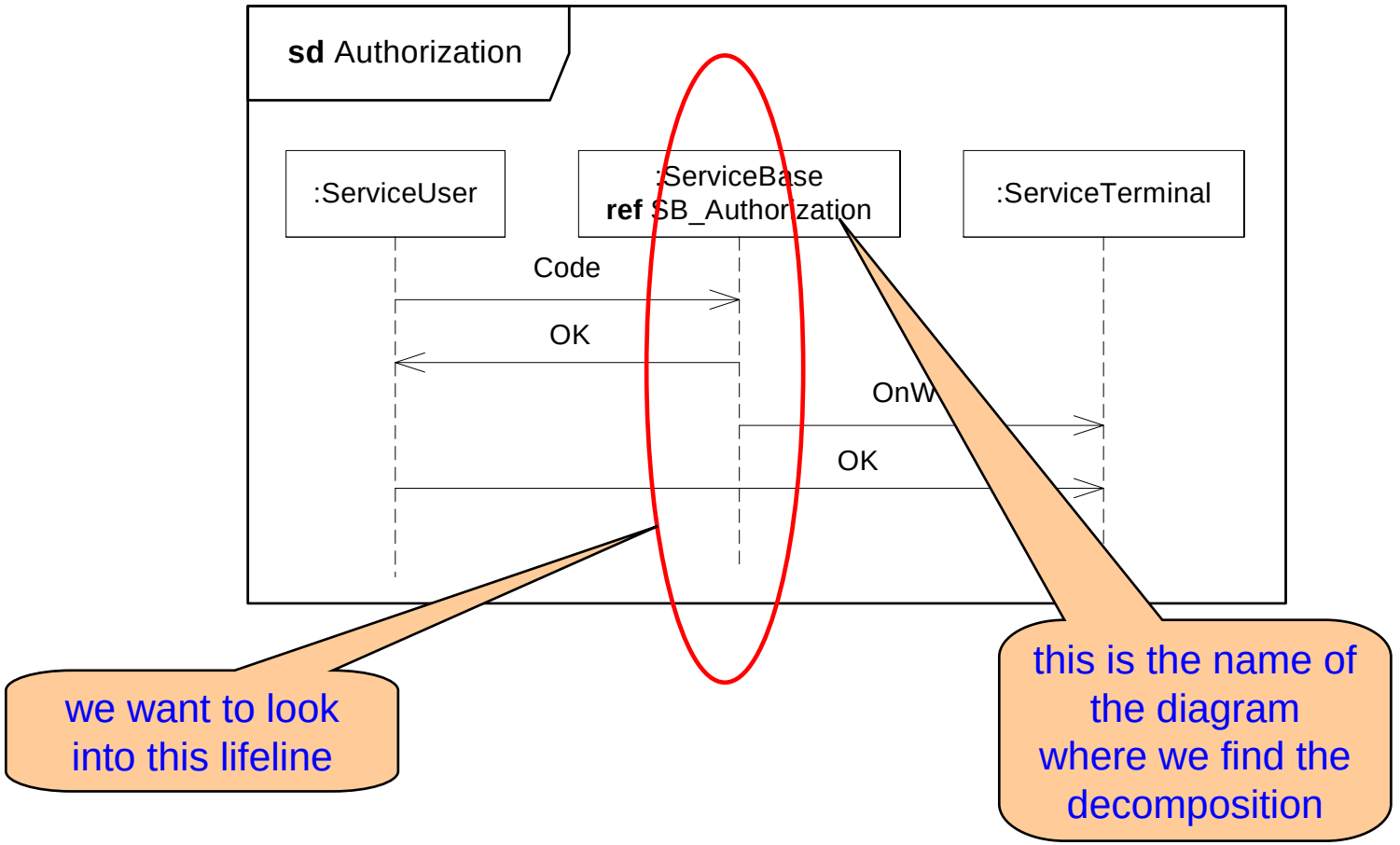


The context of a Sequence Diagram

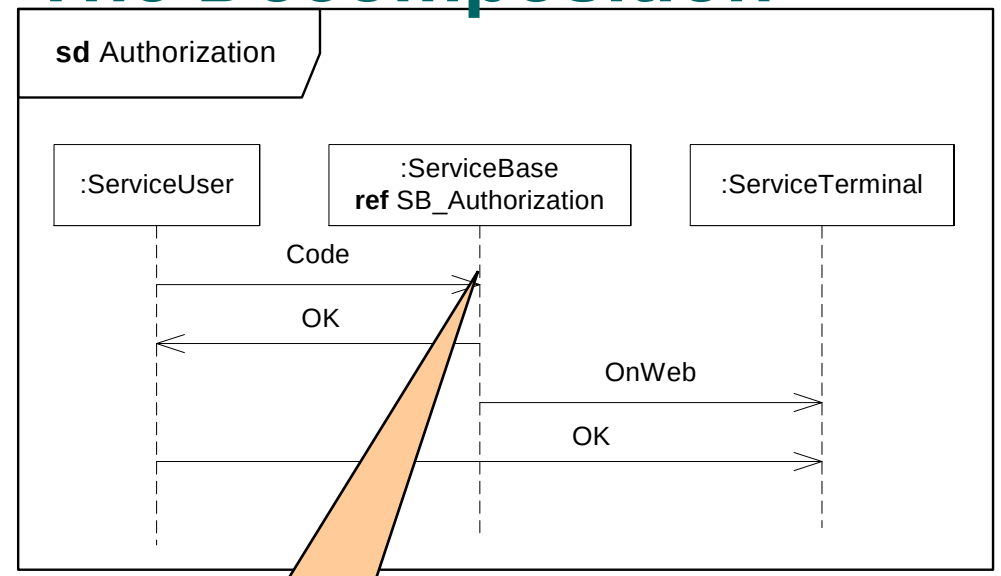
- The context is a Classifier with Composite Structure (of properties)
  - Properties (parts) are represented by Lifelines
  - Generic Parts of Collaborations must be bound to concrete Parts
  - Concrete Parts of Classes can be Lifelines directly
- The concept of a context with internal structure leads to an aggregate hierarchy of entities (parts)
  - We exploit this through the concept of Decomposition



Decomposing a Lifeline relative to an Interaction

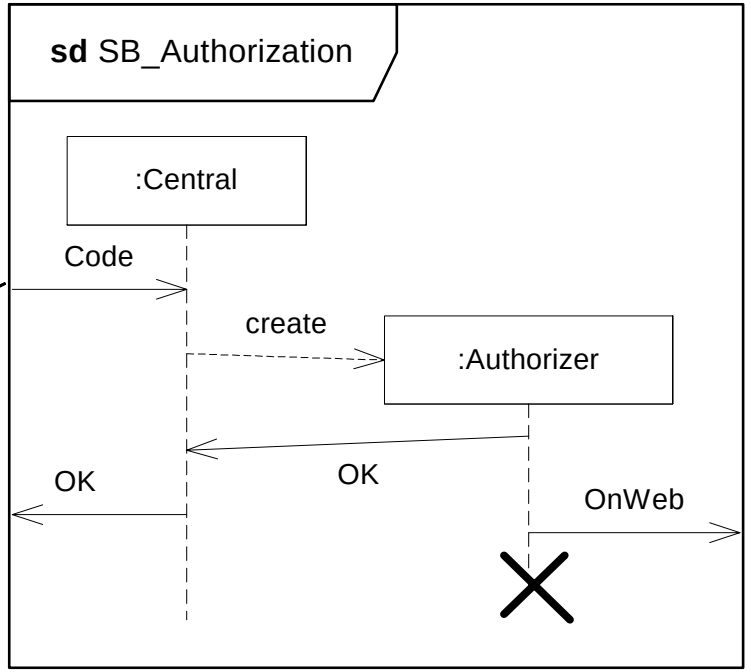


The Decomposition



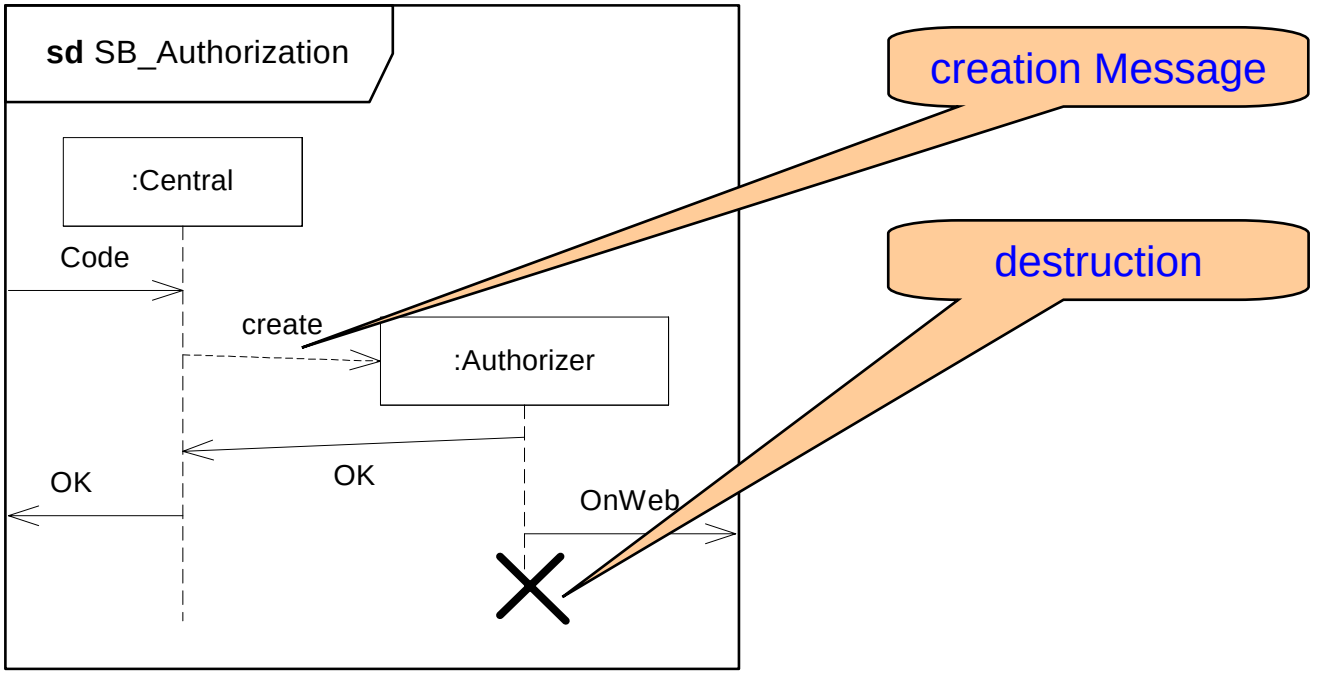
notice the correspondance!

notice the correspondance!

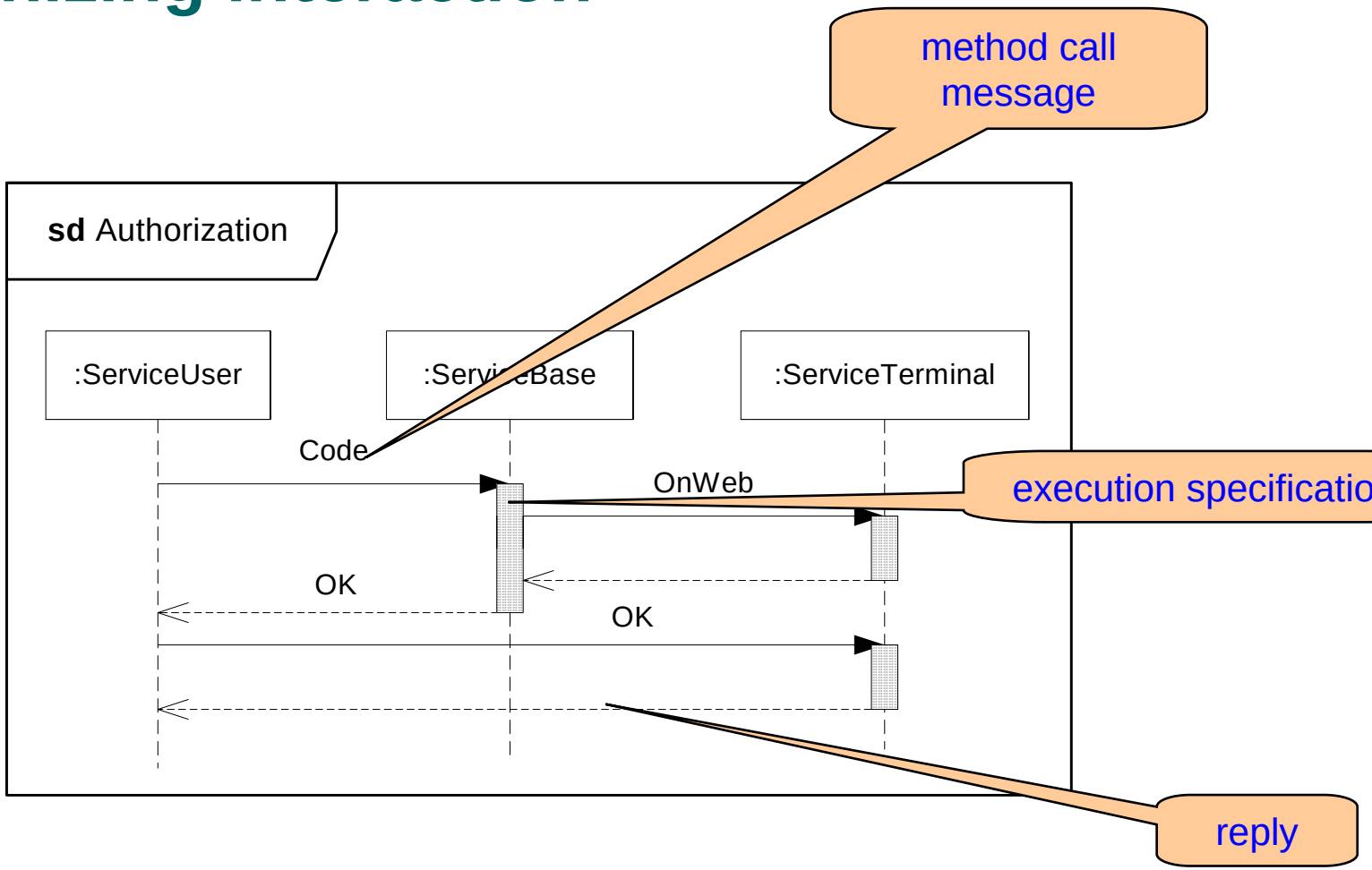


Lifeline creation and destruction

- We would like to describe Lifeline creation and destruction
- The idea here (though rather far fetched) is that the *ServiceBase* needs to create a new process in the big mainframe computer to perform the task of authorizing the received *Code*. We see a situation where several *Authorizers* work in parallel



Synchronizing interaction



Basic Sequence Diagrams Summary

- We consider mostly messages that are asynchronous, the sending of one message must come before the corresponding reception
- UML has traditionally described synchronizing method calls rather than asynchronous communication
- The events on a lifeline are strictly ordered
- The distance between events is not significant.
- The context of Interactions are classifiers
- A lifeline (within an interaction) may be detailed in a decomposition
- Dynamic creation and destruction of lifelines

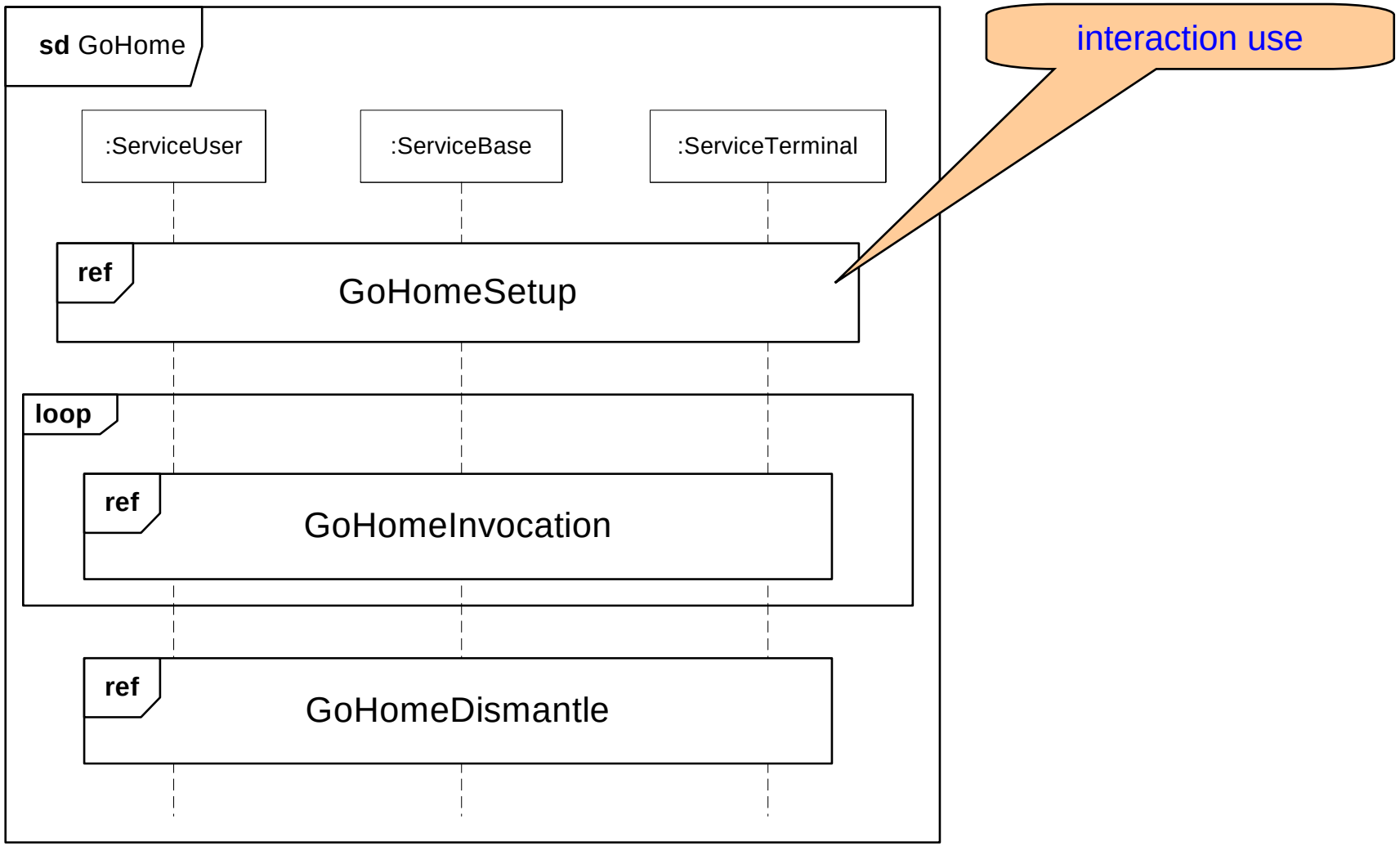
BZZZ: What would you include in advanced SD?

- Pretend that you are the language designer
- What needs do you think the early SD users had?
- What constructs would you include?

More structure (UML 2.0 from MSC-96)

- interaction uses – such that Interactions may be referenced within other Interactions
- combined fragments – combining Interaction fragments to express alternatives, parallel merge and loops
- better overview of combinations – High level Interactions where Lifelines and individual Messages are hidden
- gates – flexible connection points between references/expressions and their surroundings

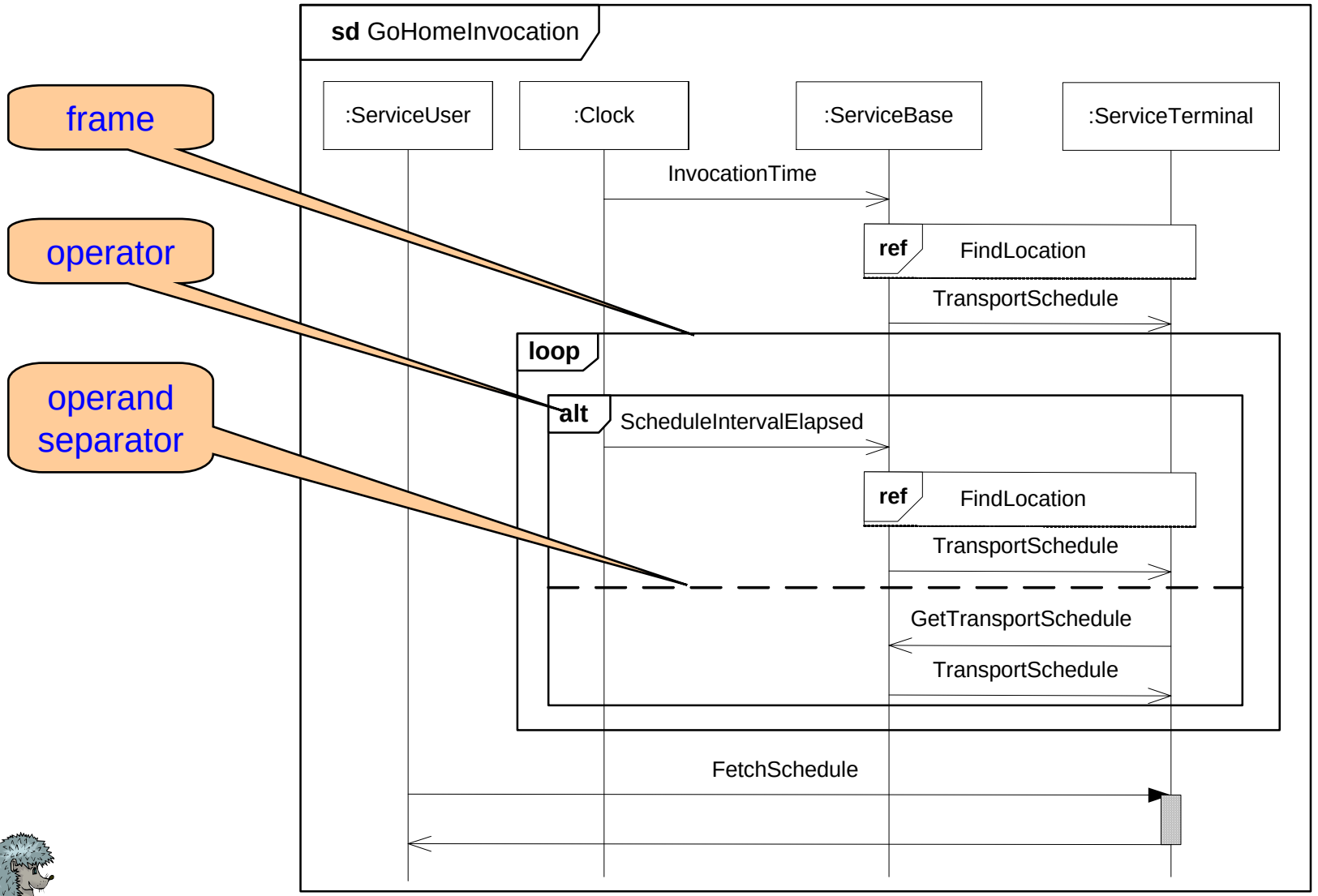
References


sd GoHome
:ServiceUser
:ServiceBase
:ServiceTerminal
interaction use
ref
GoHomeSetup
loop
ref
GoHomeInvocation
ref
GoHomeDismantle

Combined fragments of Interaction

- UML 2.0: “combined fragments”
- We want to express
  - choices: alternative, option, break
  - parallel merge
  - loops
- We also want to add other operators
  - negation
  - critical region
  - assertion
- Other suggested operators that are not in UML 2.2
  - interrupt
  - disrupt

Combined fragment example



Interaction Overview

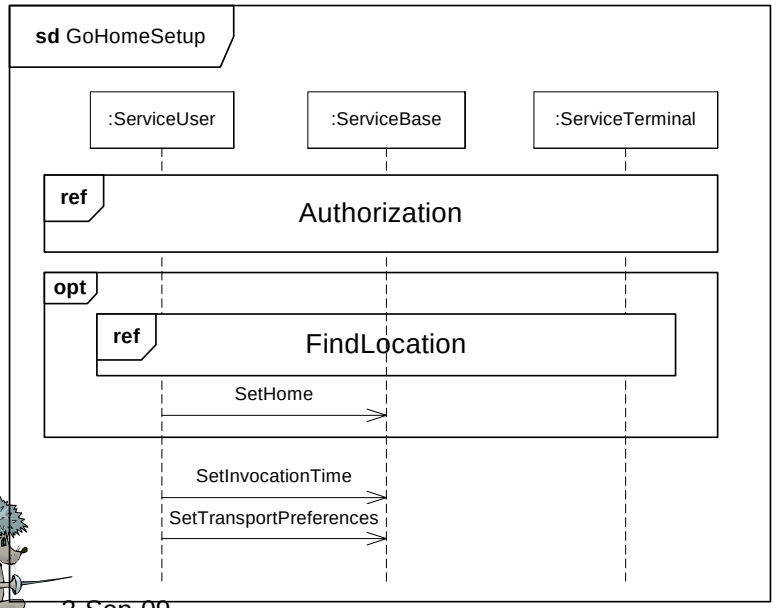
similar to activity diagram

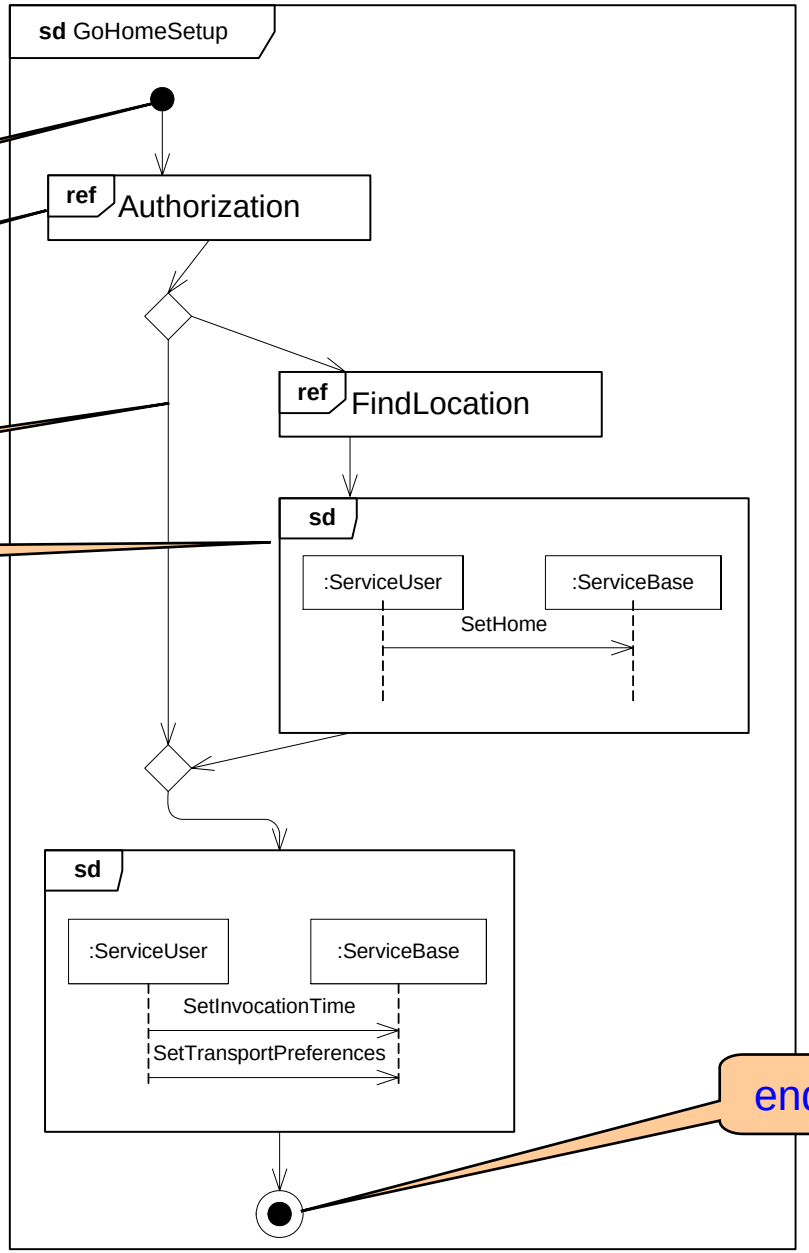
start

references

flow line

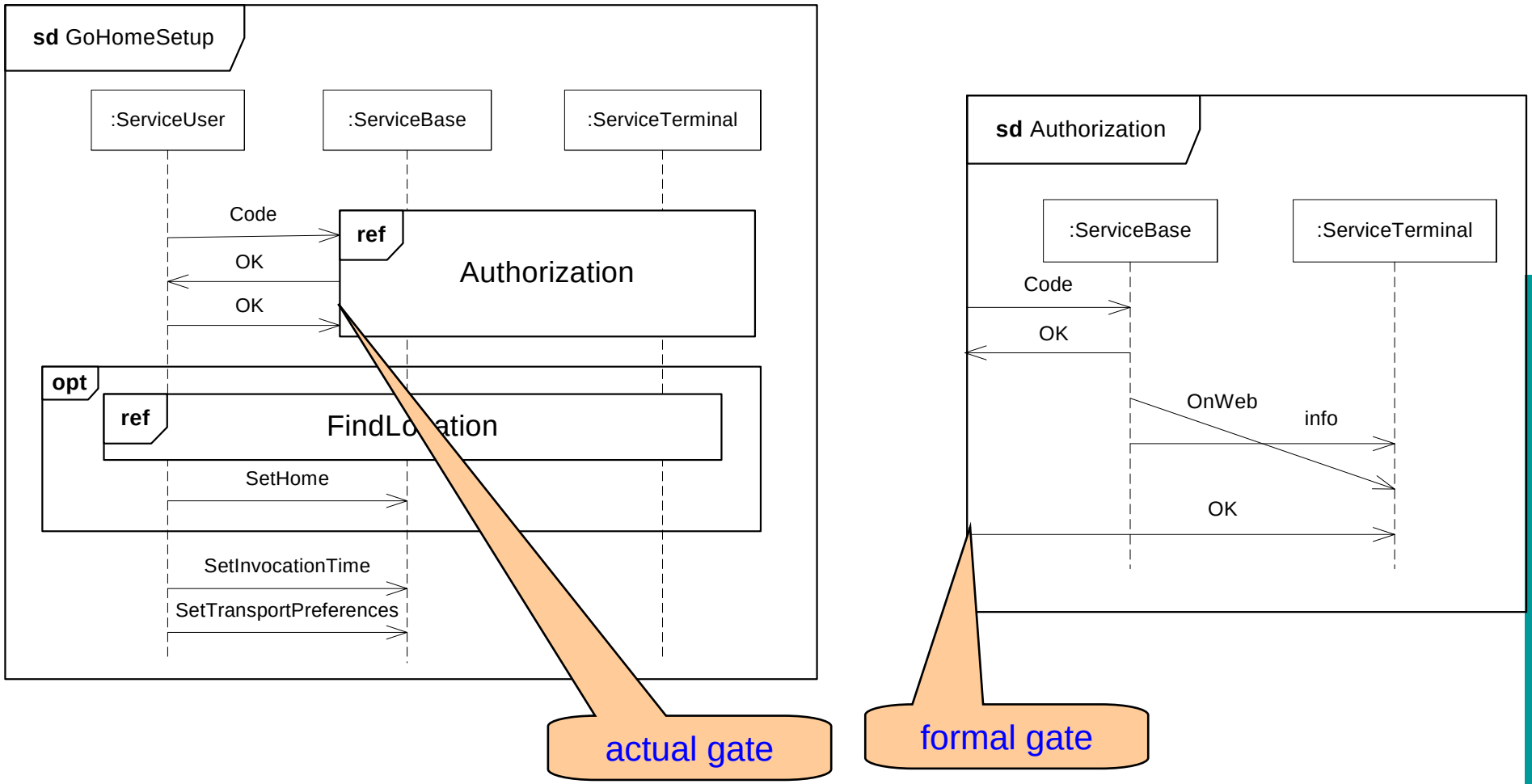
inline diagram



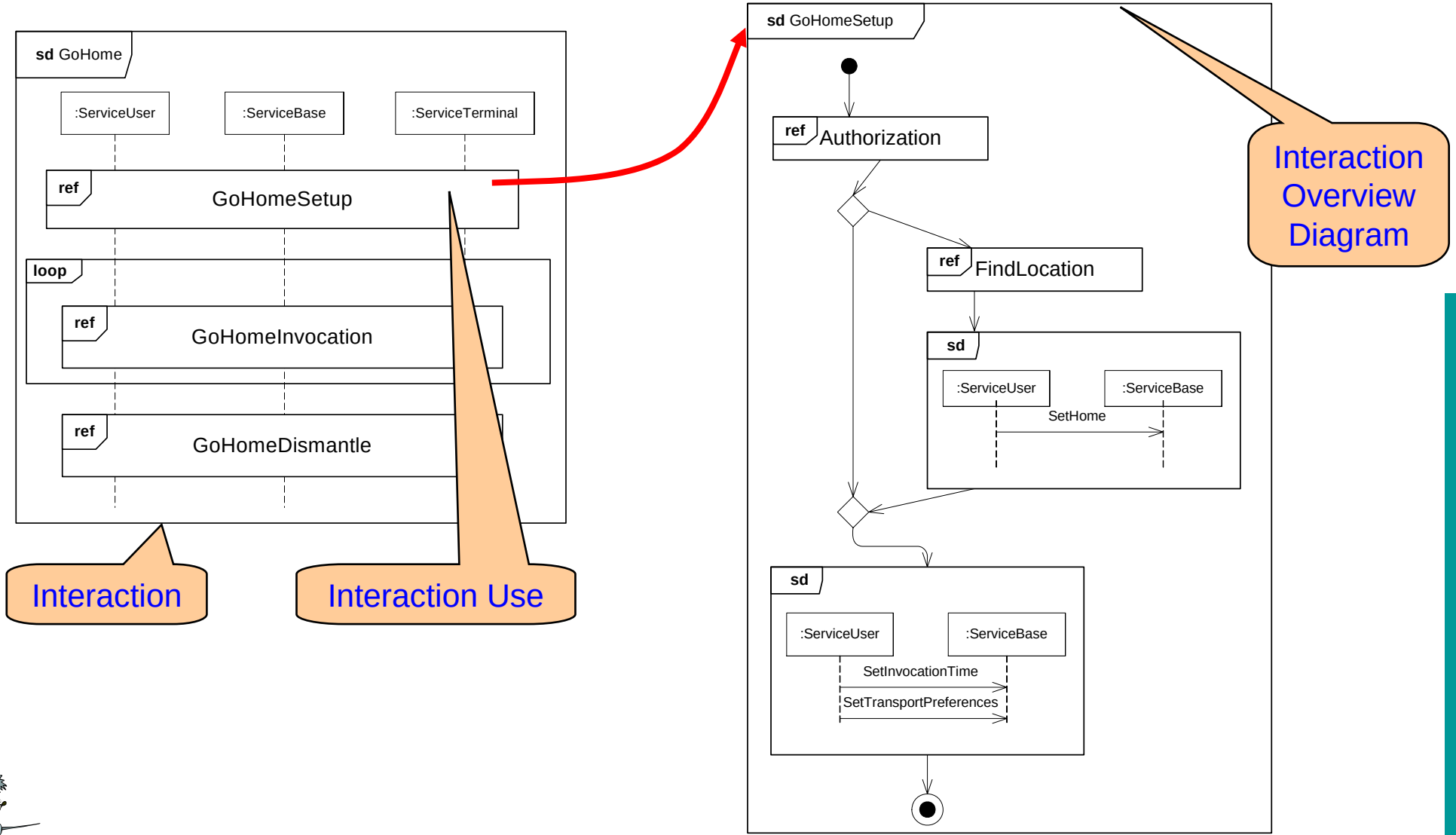


end

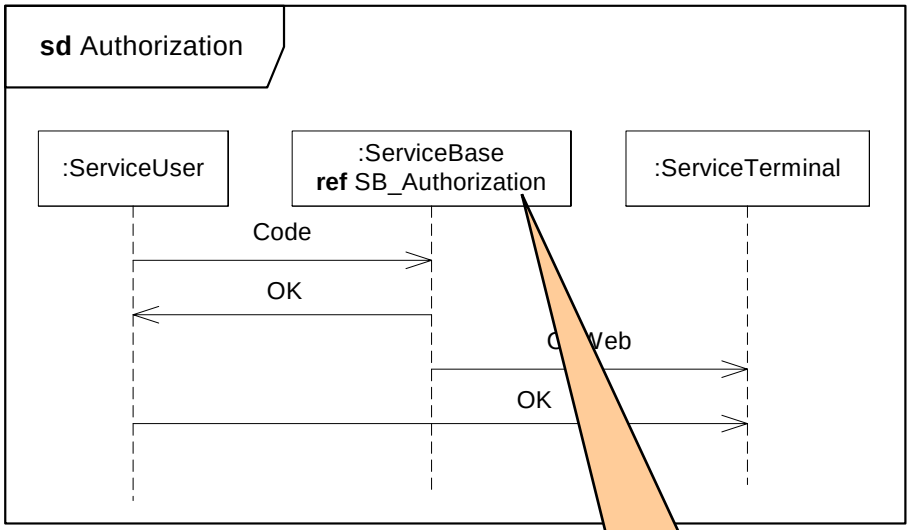
Gates

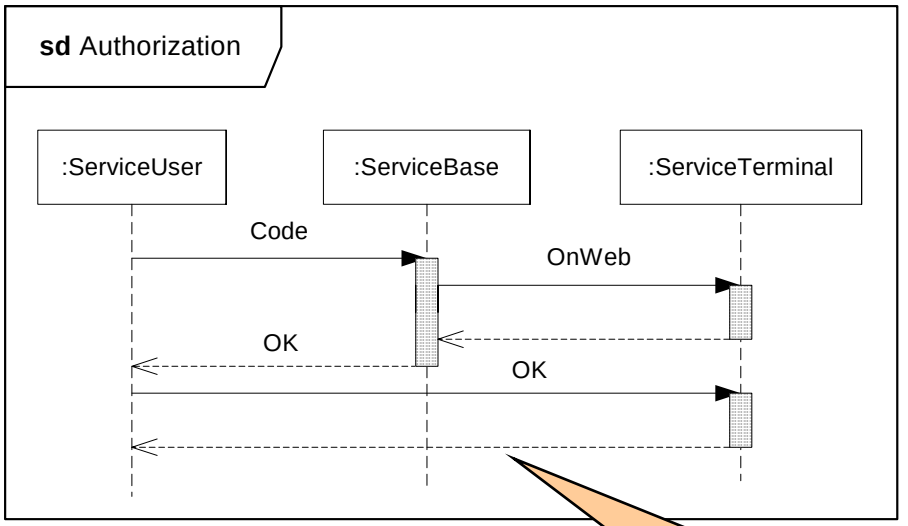

sd GoHomeSetup
:ServiceUser
:ServiceBase
:ServiceTerminal
Code
ref
Authorization
OK
OK
opt
ref
FindLo ation
SetHome
SetInvocationTime
SetTransportPreferences
actual gate
sd Authorization
:ServiceBase
:ServiceTerminal
Code
OK
OnWeb
info
OK
formal gate

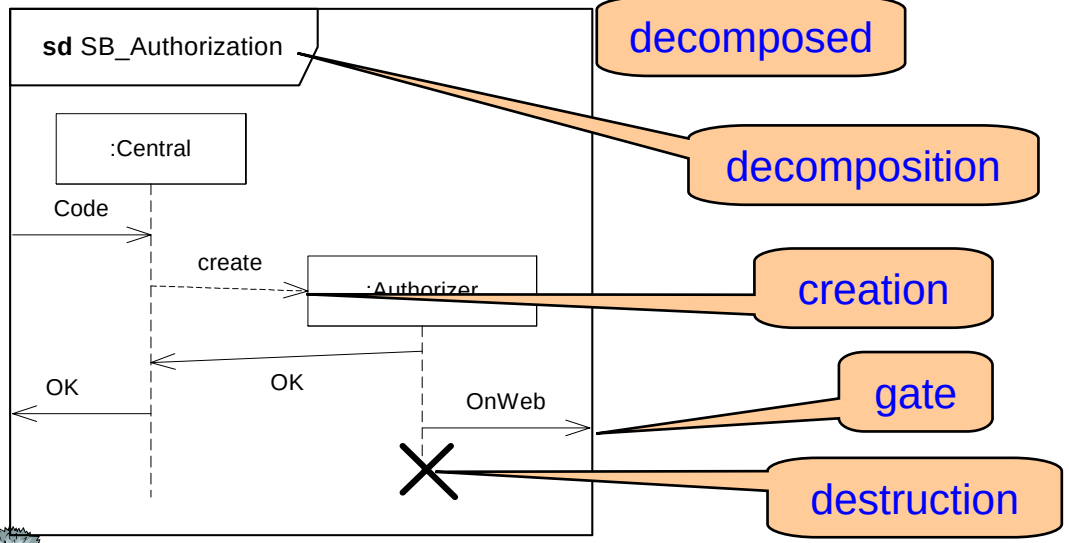
Summary: Dolly Goes To Town (1)



Dolly Goes To Town (2)







decomposed

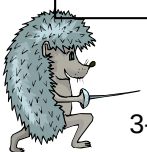
synchronizing

decomposition

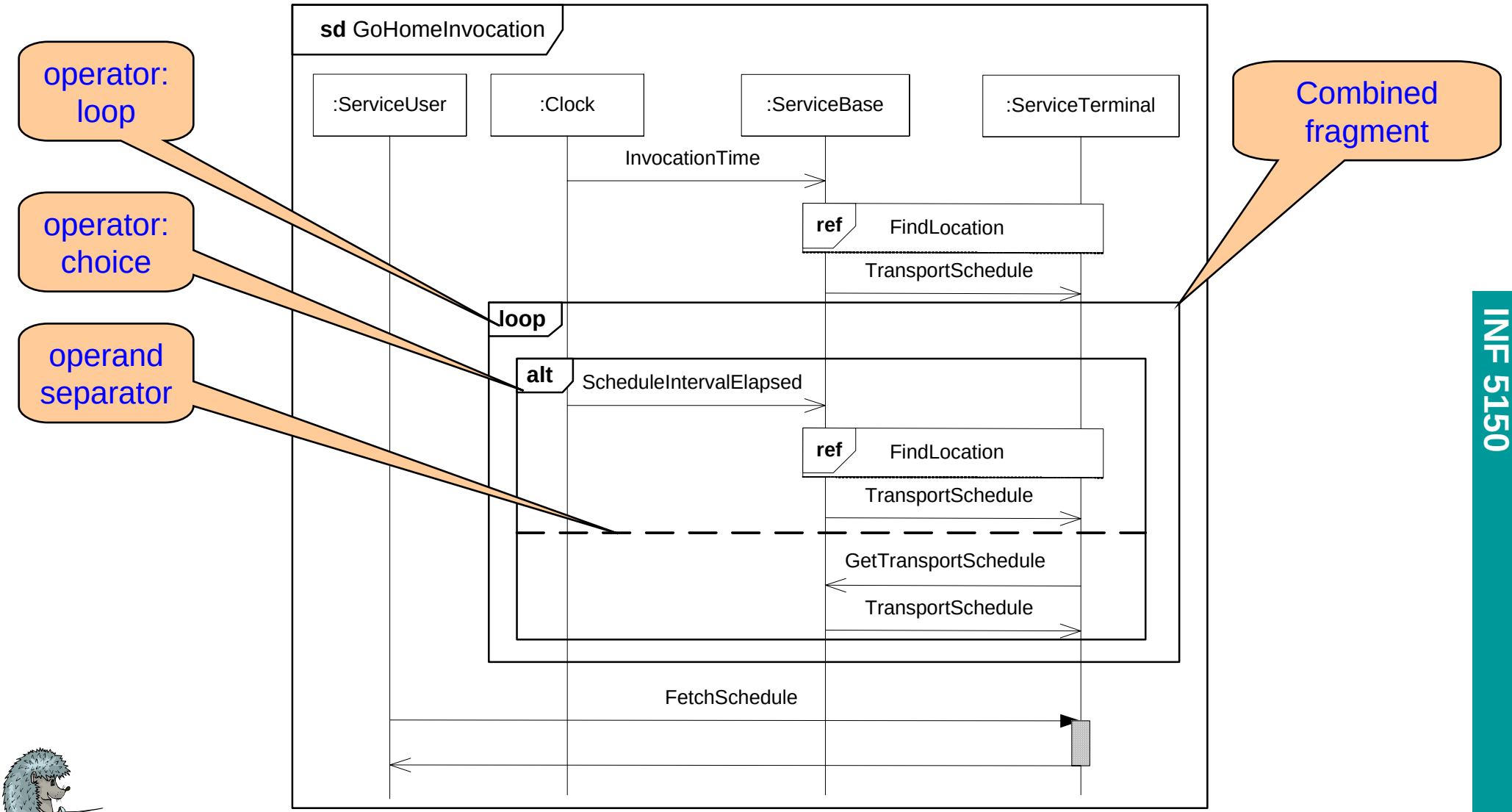
creation

gate

destruction



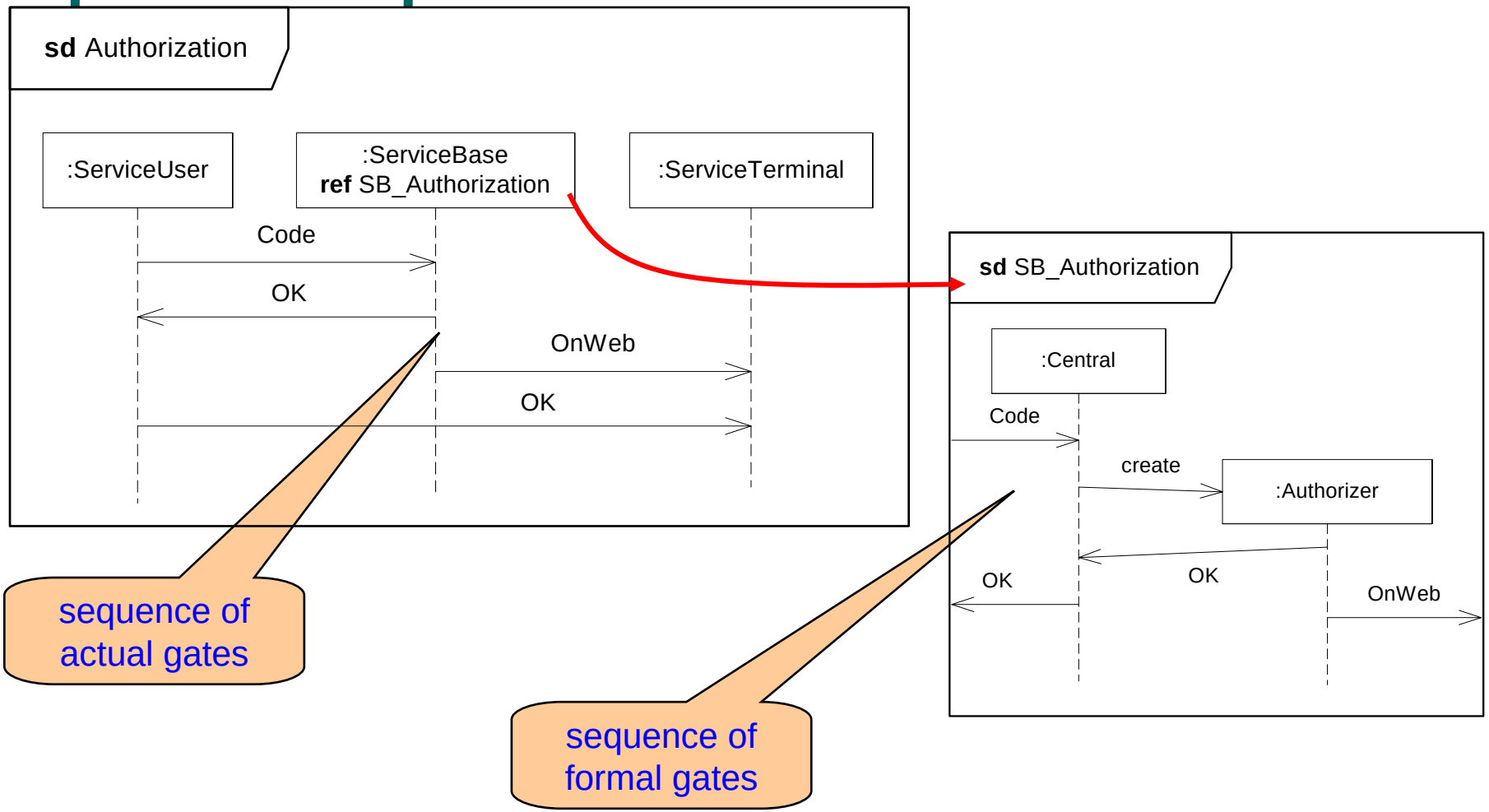
Dolly Goes To Town (3)



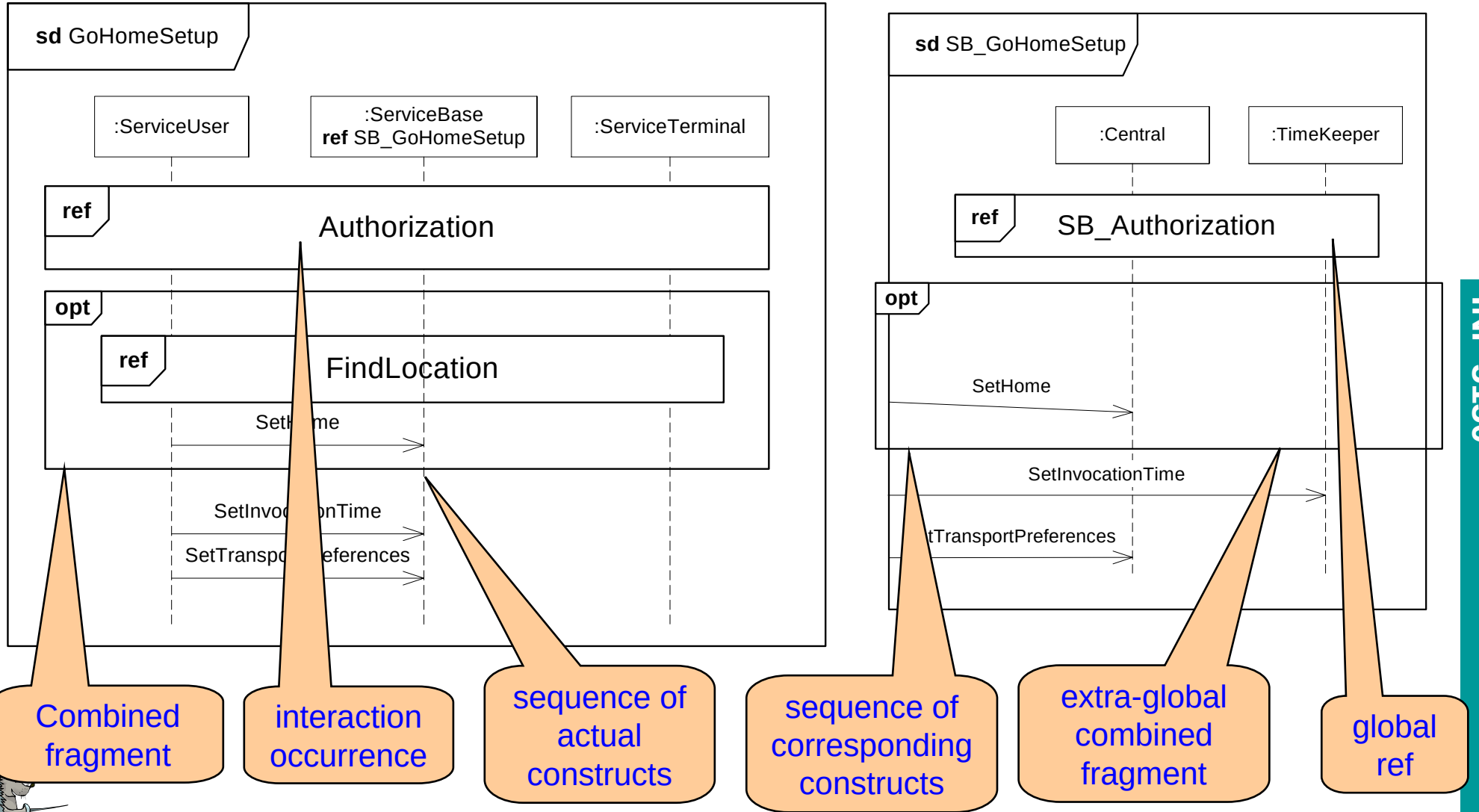
Problem areas

- Decomposition and References
  - how can we precisely define the combination of decomposition and references?
  - what about decomposition and combined interactions?
- Data
  - where is data in interactions?
  - what data can be involved in guards?

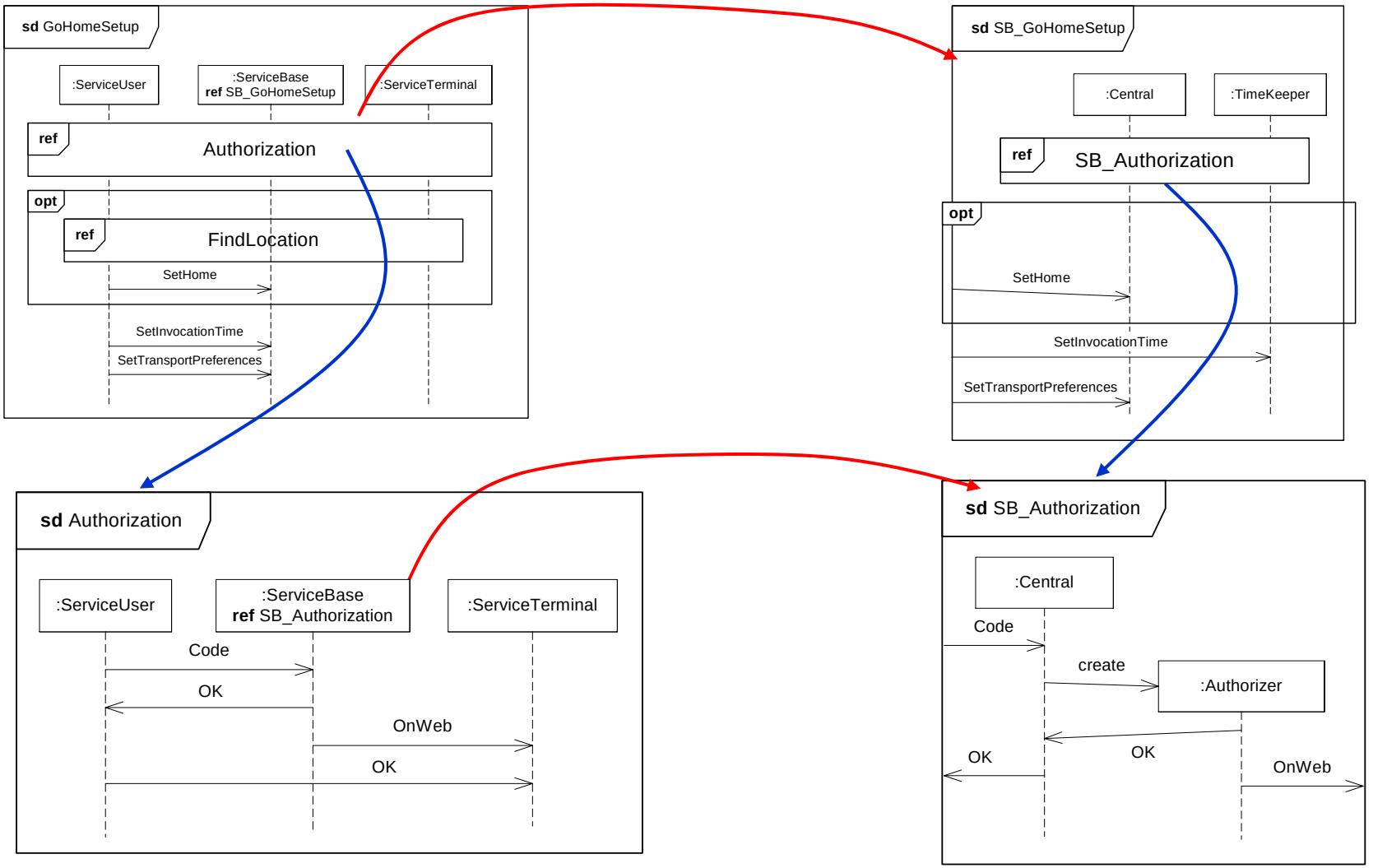
Simple Decomposition Revisited



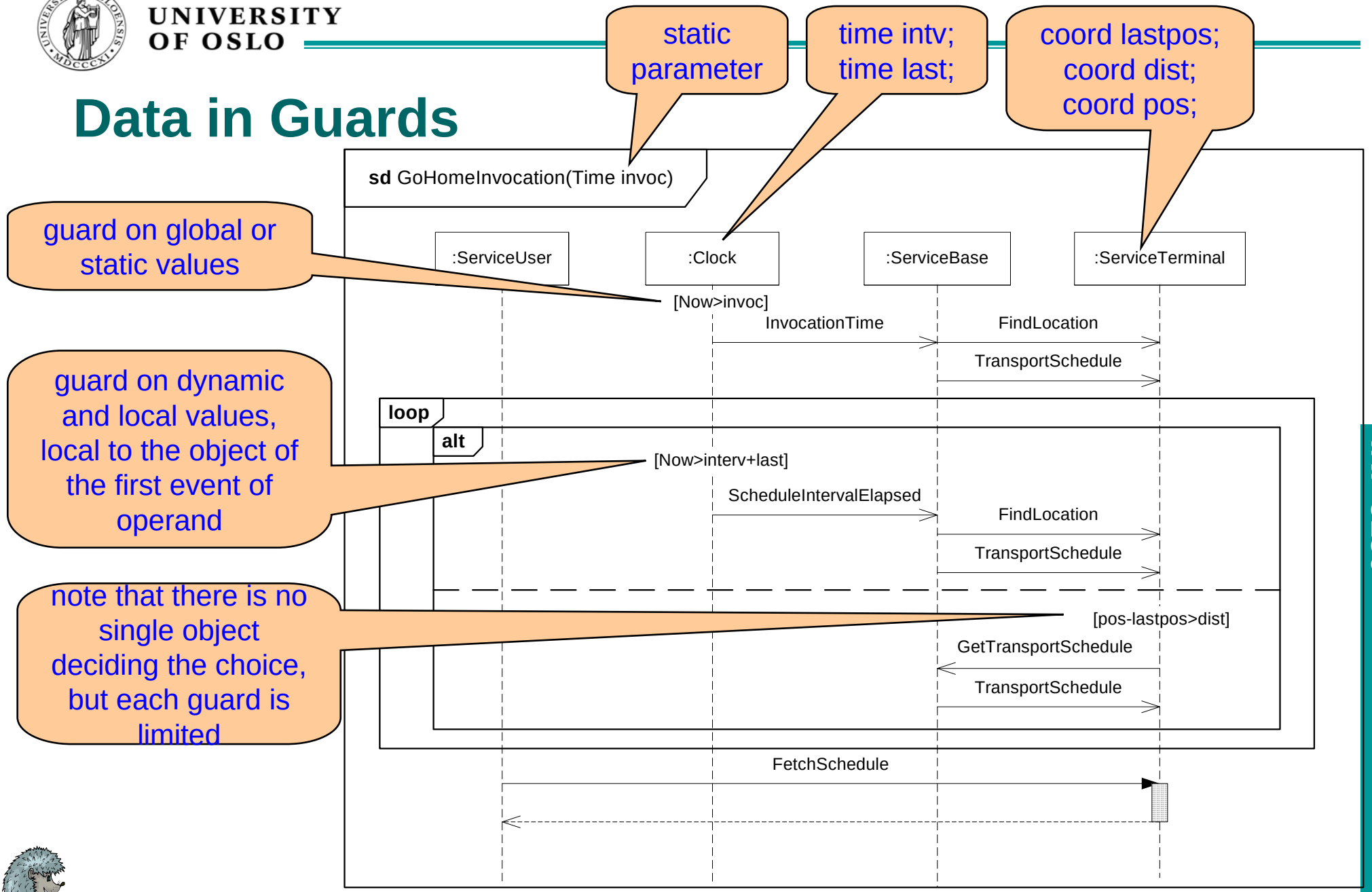
Decomposing covering ref's and combined fragments



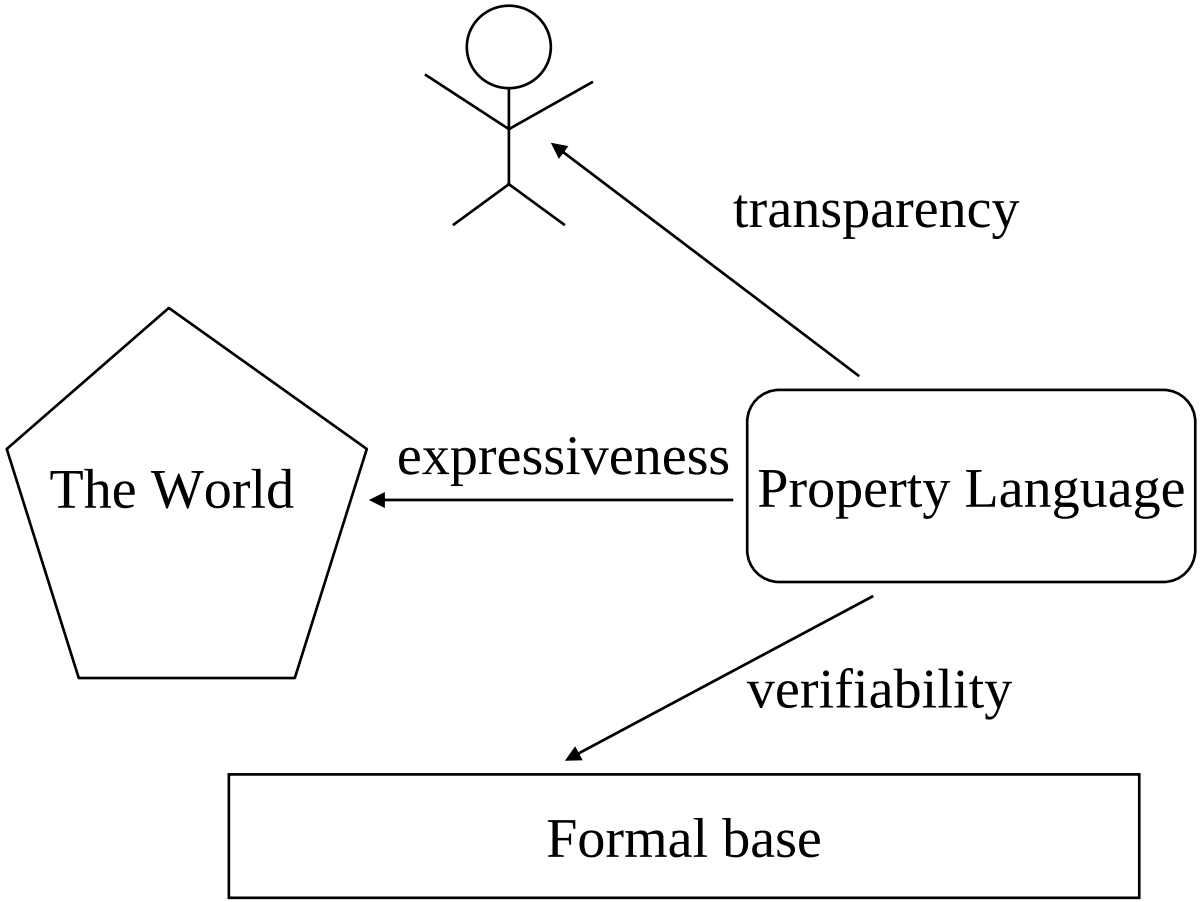
Commutative Decomposition



Data in Guards



Evaluating Modeling Languages



Comparison between Property Languages (BZZ)

Property	**Prose**	**Seq. diag.**	**Math.**	**State Machines**
transparency				
expressiveness				
formalization				
liveness, safety				
overview				
interaction				
time req.				
capacity				

Basic Sequence Diagram Methodology

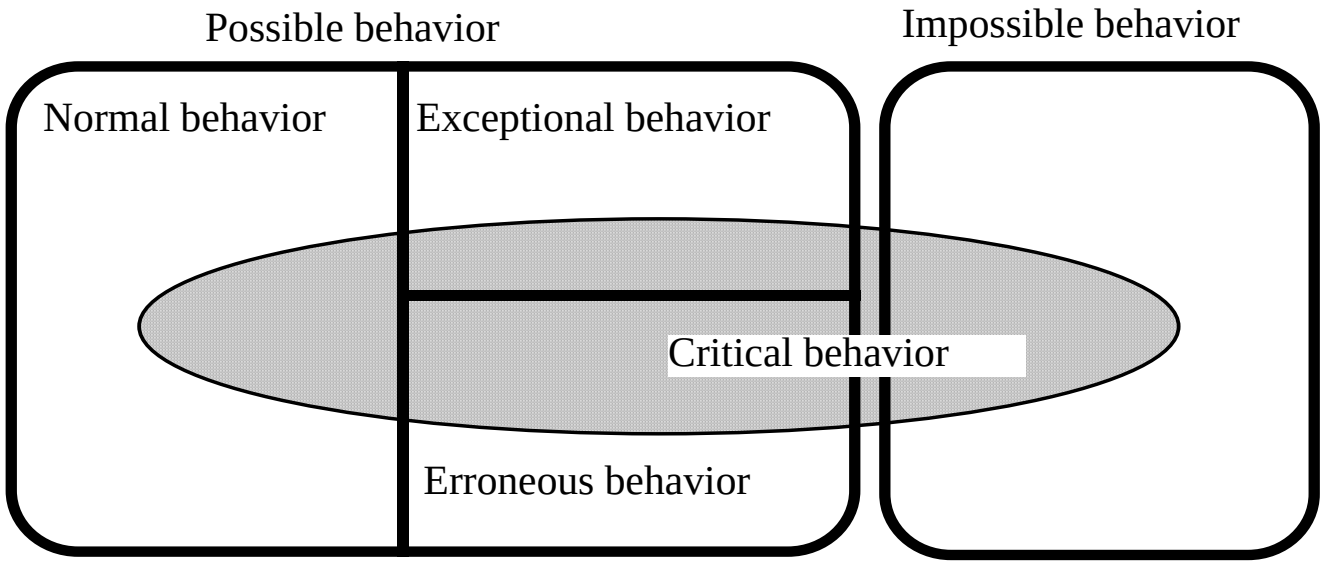
Even though Sequence Diagrams are simple and may be read and produced by engineers without much formal training, it is possible to:

- make beautiful diagrams that say nothing,
- make messy diagrams that are meant to convey critical information,
- make terrible diagrams in an early phase that make it impossible to design a sensible system in a later phase.
- use extensions to UML/MSC that are not standard and that may prevent you from using (more than one) tools.

The methodology aims at bridging the gap between the notation and the development process using it.

Seq. diag. classification: case evaluation

- *Normal* behavior is the behavior that we expect
- *Exceptional* cases are those that may happen, and that we should prepare for, but which we do not consider normal.
- The *erroneous* behavior is behavior that we try to avoid, but which should not destroy our system.
- *Impossible* behavior is behavior that cannot happen



Seq. diag. classification: descriptive goal

Descriptive goal	Target audience	Life span
historical	project members, managers, potential customers	temporary
documentary	managers, customers	negotiations or product span
requirements	customers, project team	product span
design	project team	project
test	testers, customers	product span

Step 0: Make explicit the company SD strategy

- *Tools*: What tools will be used to produce and maintain the mscs?
- *Coverage Profile*: How do the diagrams cover the universe of scenarios?
- *Document Profile*: What diagrams are to be produced?
- *The Inexpressible*: How is information not expressible in UML/MSC attached?

Step 1a: the first sequence diagrams

- Our metaphor for building our MSC document is a news photographer covering a major event.
- Firstly he will make sure to take pictures of the main characters – the *normal* cases.
- Then he will look for some *exceptional* situation which might sell better to the public and which may capture unexpected problems like the police horse galloping.
- Then he digs for *errors* like the possible assassin in the bushes.
- Finally he could illustrate the *impossible* by manipulating a picture like placing Forrest Gump with President Nixon.

Step 1b: Establish the interplay with non-developers

- Require responsibility and approval from the non-developers;
- Involve the non-developers in making additional diagrams making sure that they understand UML/MSC and that they understand that they understand UML/MSC;
- Associate concrete input/output with the user interface.
- Encourage the non-developers to use their UML/MSC knowledge during the design and model checking phases

Summary Basic Sequence Diagram strategy

Step 0 : company strategy
what tools
what coverage
what MSC documents
How to attach informal text

Step 1a : the first mscs
normal
exceptional
erroneous
impossible
critical

Step 1b : interplay with non-developers
require responsibility
active involvement
be concrete
encourage further use of MSC

Step 2a : Variants and similarity
global conditions
road map
MSC document table

Step 2b : Refinement
message hierarchy
instance hierarchy

Step 2c : Inexpressibles
dependency
capacity and duration

Step 3 : Support the design
alignment table
checking existence
checking full coverage

Step 4 : Test mscs
isolate IUT
project existing mscs

How is this related to unassailability?

- UML 2 sequence diagrams are
  - intuitive
    - but only partial
  - precise
  - supported by tools
- Proper methodology is needed
  - recognizing that sequence diagrams do not tell the whole story
  - increasing the consciousness of
    - which diagrams to make
    - their purpose
- Achieving
  - early awareness of problems