# Aspect-Oriented Model Development at Different Levels of Abstraction

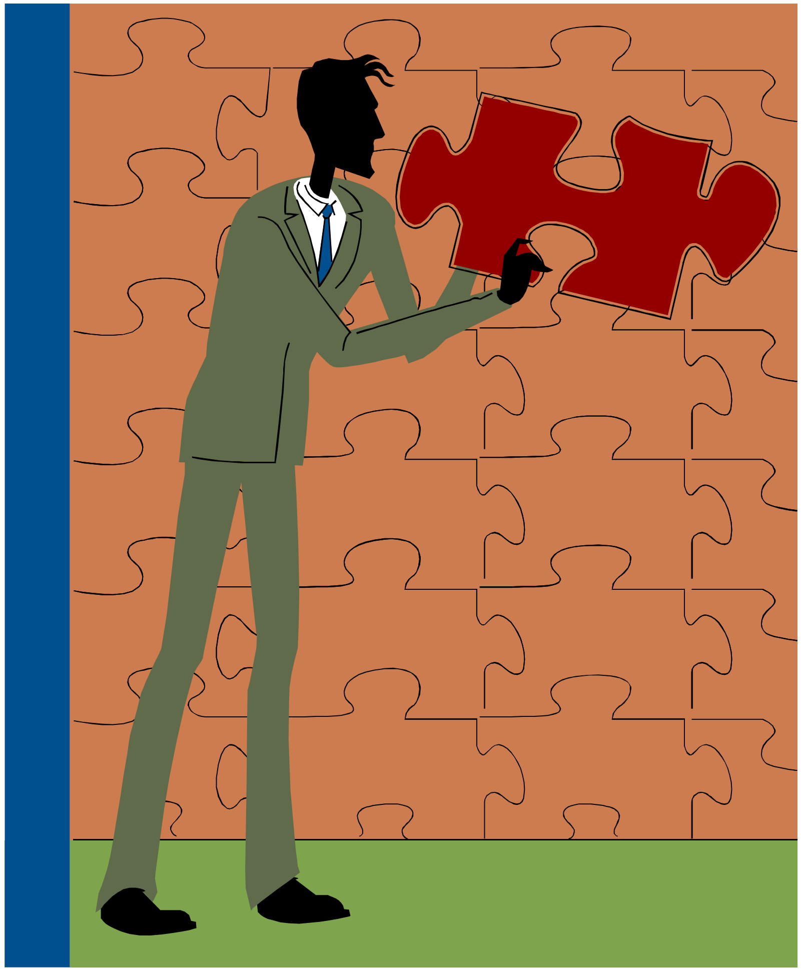
Mauricio Alfeez, Universidade Nova de Lisboa, Portugal
Nuno Amalio, University of Luxembourg
Selim Ciraci, University of Twente, the Netherlands
Franck Fleurey, SINTEF IKT, Norway
Jorg Kienzle, McGill University, Canada
Jacques Klein, University of Luxembourg
Max Kramer, Karlsruhe Institute of Technology, Germany
Sebastien Mosser, INRIA Lille - Nord Europe
Gunter Mussbacher, SCE, Carleton University, Canada
Ella Roubtsova, Open University of the Netherlands and Munich University of Applied Sciences, Germany
Gefei Zhang, Ludwig-Maximilians-Universitat Munchen
arvato systems, Germany

## Outline

- Motivation
- Aspects at the level of
  - Features
  - Use cases
  - Classes with sequence diagrams
  - Classes with state machines
  - Services
  - Mixins
  - Contacts
- Localization of concerns and localization of reasoning in aspectual models at different level of abstraction

# Motivation

- Understand possibilities and limitations in keeping intellectual control over evolving models using different type of modeling semantics via
  - localization of concerns,
  - verification and
  - localization of reasoning on parts about behaviour of the whole



# Authentication Concern to compare approaches

1. System prompts CMSEmployee for login id and password
2. CMSEmployee enters login id and password into System.
3. System validates the login information. Use case ends in success.

Extensions:

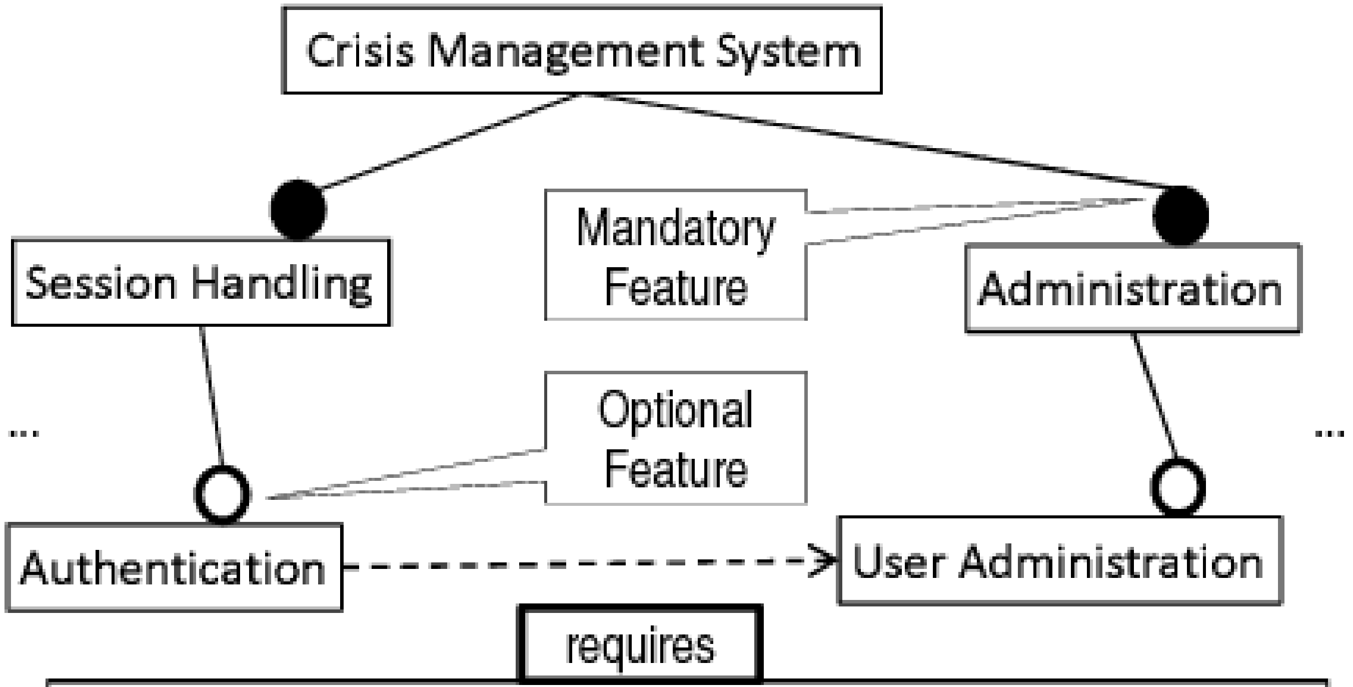
2a. CMSEmployee cancels the authentication process.
    Use case ends in failure.

3a. System fails to authenticate the CMSEmployee.

3a.1 Use case continues at step 1.

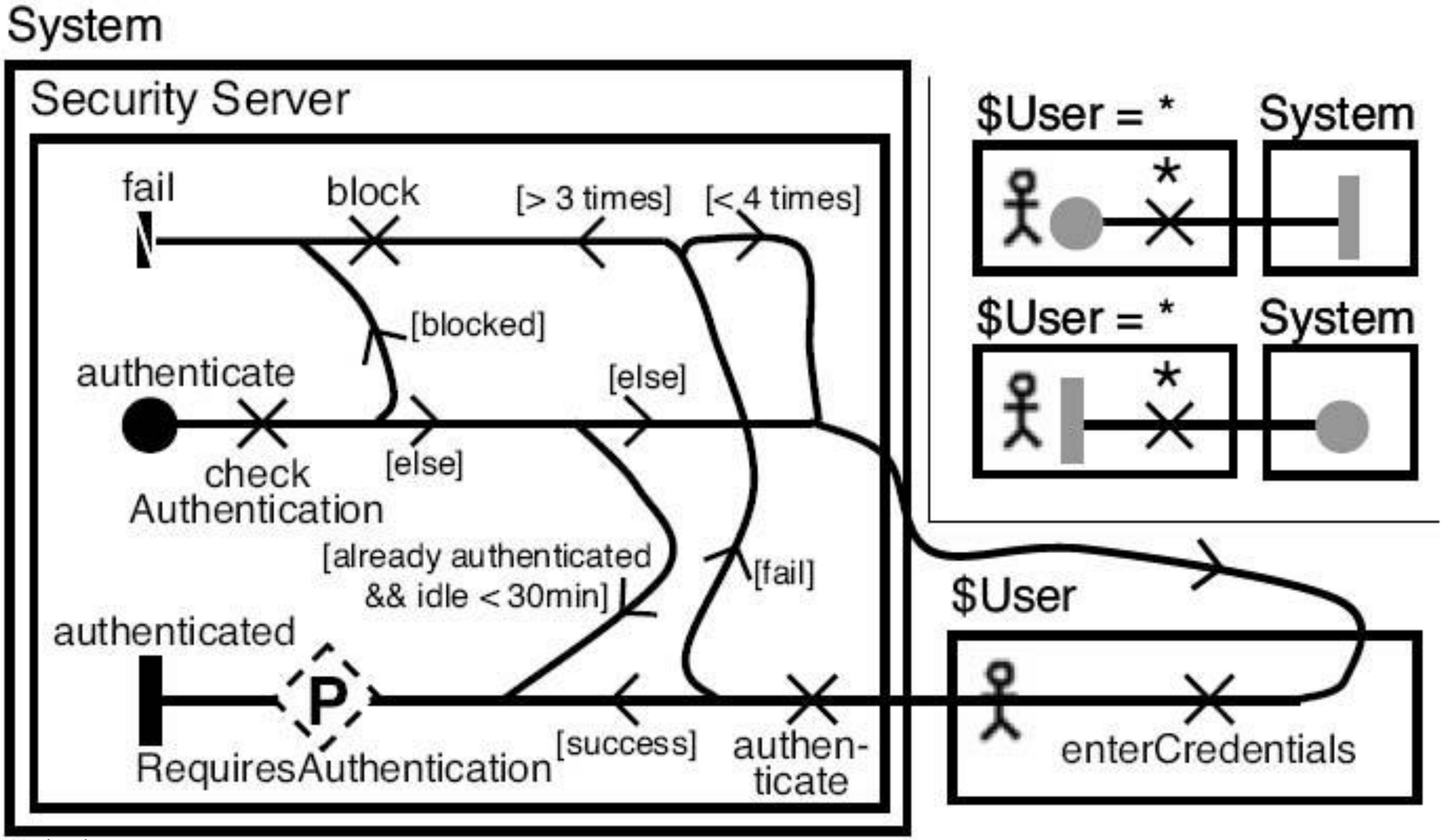
3a.1a CMSEmployee performed three consecutive failed attempts. Use case ends in failure.

# AOM revision of Features: Variability Modelling Language for Requirements (VML4RE)



**VML4RE specification**
**variant for** "Authentication" {
**IncludeConcern** "Authentication"
} ...

# AOM revision of Use Cases: Aspect-oriented User Requirements Notation AOURN



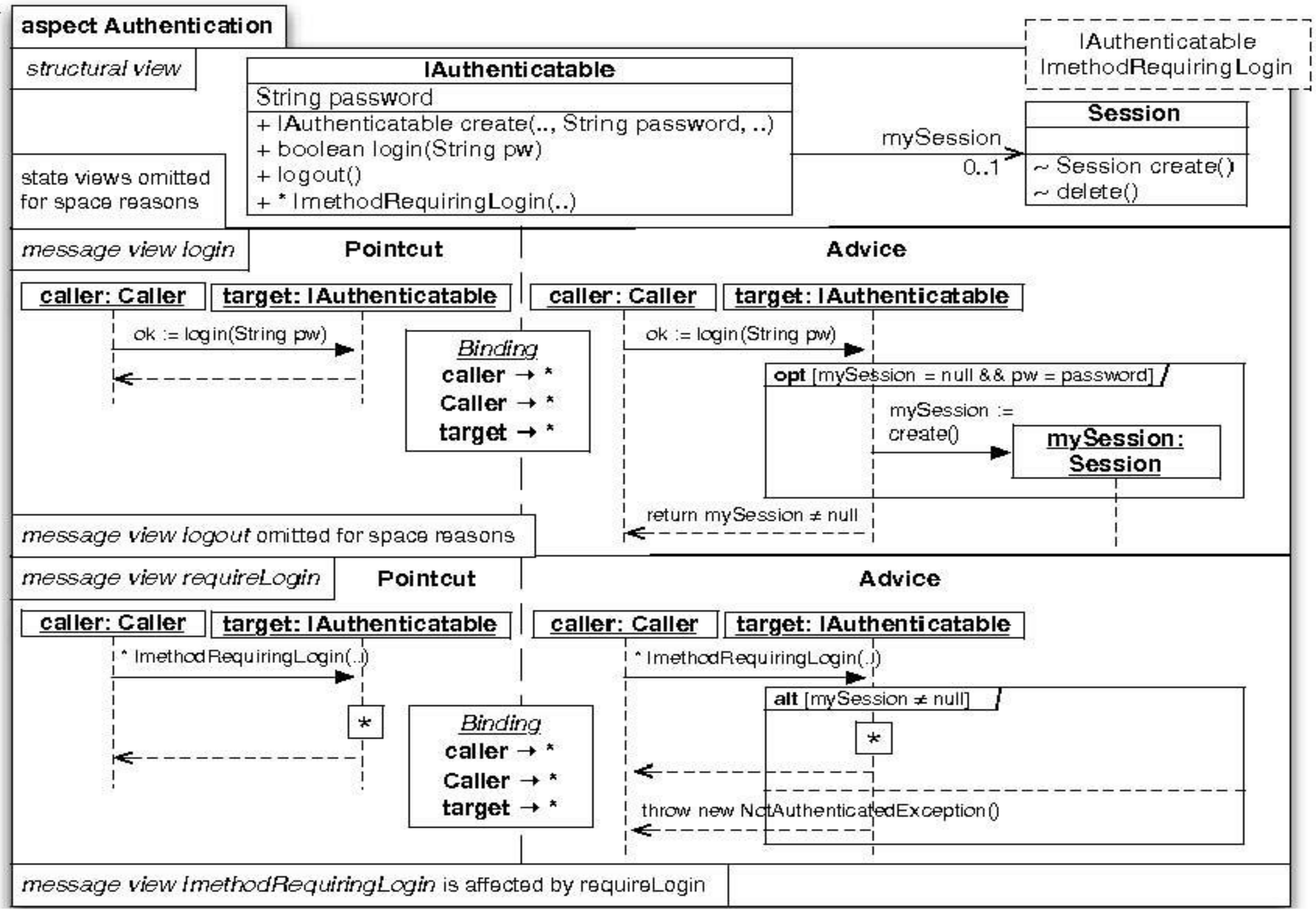
## AoURN and composition

- Decomposition of use cases
- Composition is restricted to concatenation of sequences and hierarchy
- Interactive execution is impossible. The specification allows generating of test suits for models and implementations

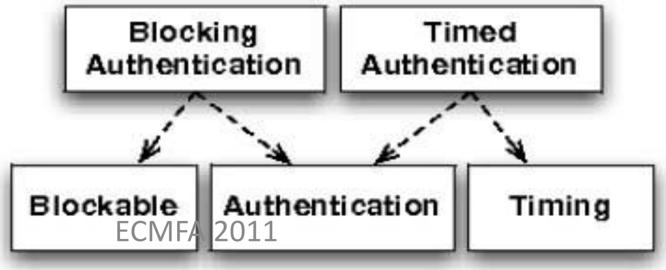
## AOM revision of Classes with Sequence Diagrams

- Theme
- Reusable Aspect Models (RAM)
- Graph-based Adaptation, Configuration and Evolution (GrACE)
- Point cuts and Advices  are models as separate sequence diagrams. Advise diagrams are woven into base sequence diagrams. State views are used to show how the state is changed  to prevent specification of

# RAM

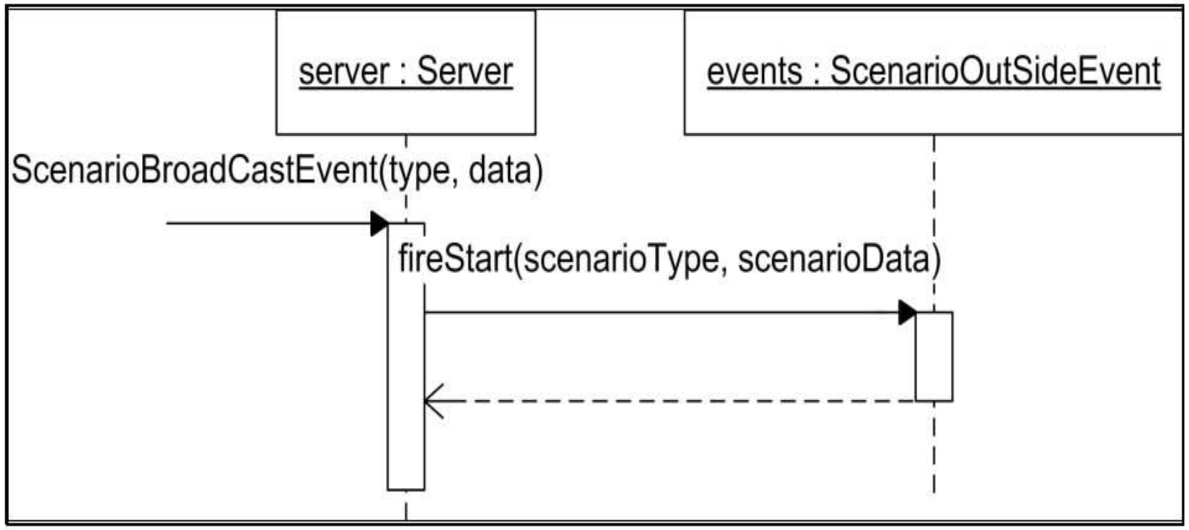


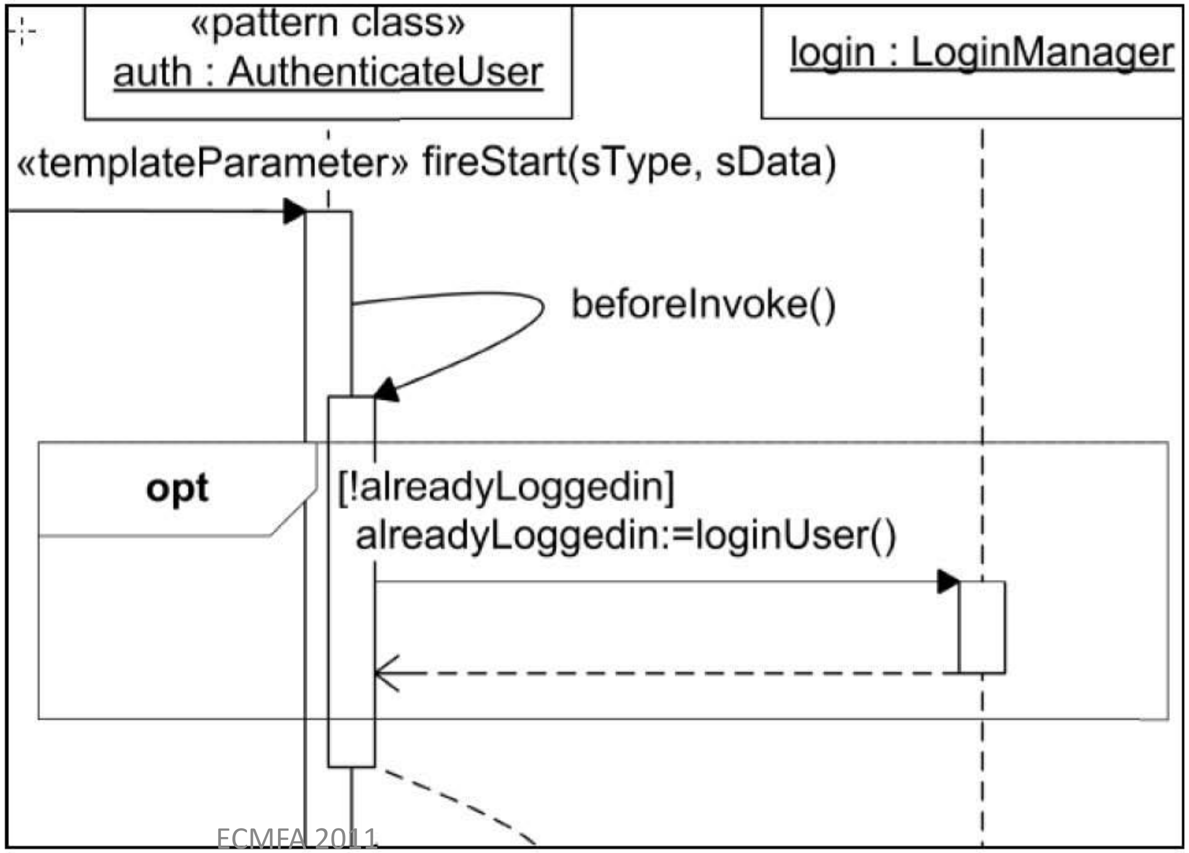
aspect Authentication
structural view
IAuthenticatable
String password
+ IAuthenticatable create(.., String password, ..)
+ boolean login(String pw)
+ logout()
+ * ImethodRequiringLogin(..)
mySession
0..1
Session
~ Session create()
~ delete()
IAuthenticatable
ImethodRequiringLogin
state views omitted for space reasons
message view login
Pointcut
Advice
caller: Caller
target: IAuthenticatable
ok := login(String pw)
Binding
caller → *
Caller → *
target → *
opt [mySession = null && pw = password]
mySession := create()
mySession: Session
return mySession ≠ null
message view logout omitted for space reasons
message view requireLogin
* ImethodRequiringLogin(..)
alt [mySession ≠ null]
*
throw new NotAuthenticatedException()
message view ImethodRequiringLogin is affected by requireLogin



Blocking Authentication
Timed Authentication
Blockable
Authentication
Timing

## GrACE


server : Server
events : ScenarioOutSideEvent
ScenarioBroadCastEvent(type, data)
fireStart(scenarioType, scenarioData)

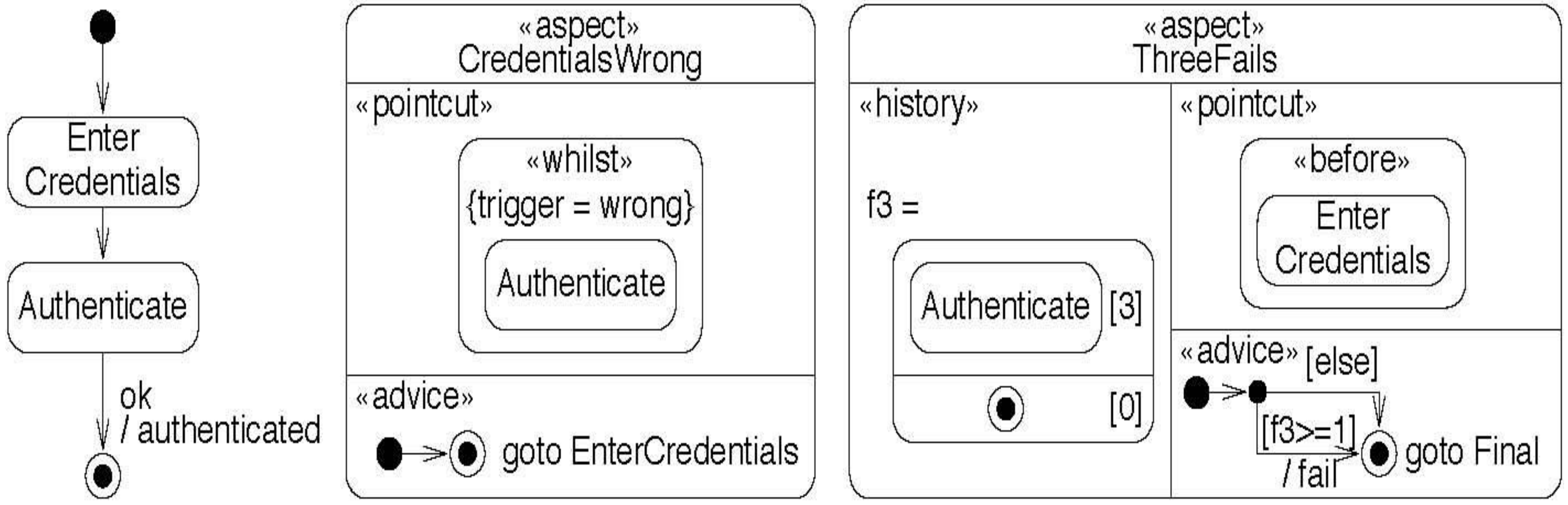

«pattern class»
auth : AuthenticateUser
login : LoginManager
«templateParameter» fireStart(sType, sData)
beforeInvoke()
opt
[!alreadyLoggedin]
alreadyLoggedin:=loginUser()

## Properties of the approaches working with sequence diagrams

- Localization of concern specification
- Specification of behavior for complex systems is often incomplete.
- If the specification of state is involved then the specification of
  - Spectative aspects
  - Regulative aspects
  - Invasive aspects

  is possible
- Local reasoning is impossible and after any evolution step of the model the whole model should be re-generated and re-analysing or verified.

# AOM revision of Classes with Behavioural State Machines: High-Level Aspects (HiLA)



# HiLA: properties of models

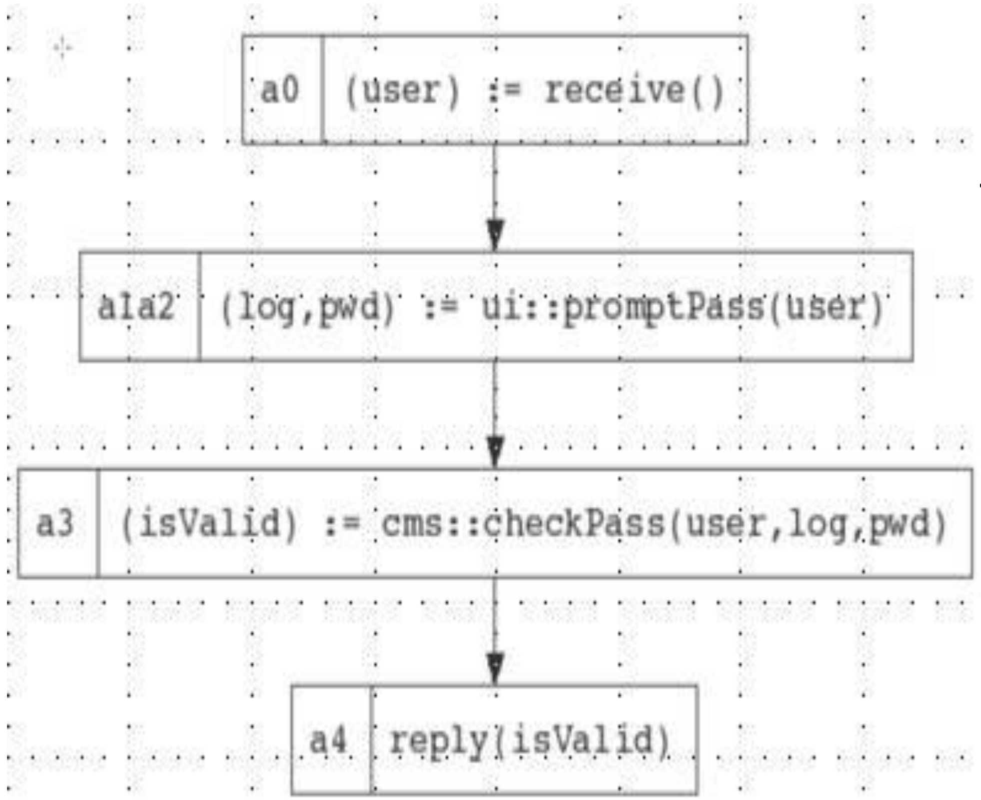
**Semantics of BSM:**

- If an event recognized by a machine does not enabled in the current state of machine, but is included in the list deferred events for this state, it is kept in a queue for later processing, otherwise the event is discarded.
- Processing of an event by different machines is asynchronous and the result of processing is non-deterministic.

**Verification:**

- HiLA uses formal methods of model validation. The application of aspects to BSM results in another UML state machine which is analyzed using the model checking component that translates the state machine and the assertions into the input language of a back-end model checker SPIN. SPIN then is used to verify the given properties presented in Linear Temporal Logic.

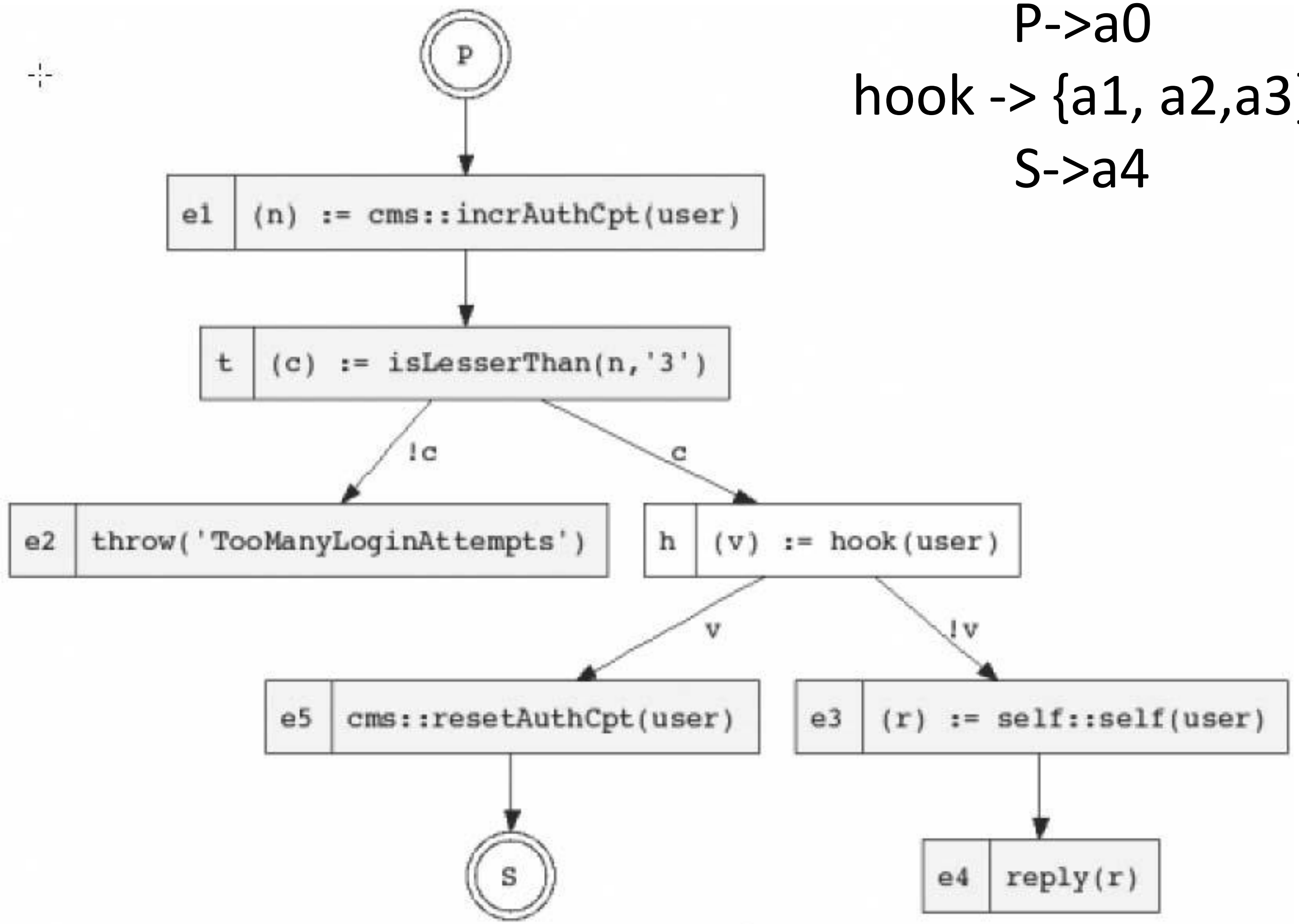
# AOM revision of Activity diagrams: Activity moDel supOrting oRchestration Evolution ADORE


a0
(user) := receive()
a1a2
(log,pwd) := ui::promptPass(user)
a3
(isValid) := cms::checkPass(user,log,pwd)
a4
reply(isValid)

In Adore, an orchestration of services defined as a partially ordered set of activities.

The different types of activities that can be defined in Adore include

- (i) service invocation (denoted by **inv**oke), **inv**
- (ii) variable assignment (**a**ssign), **a**
- (iii) fault reporting (**t**hrow), **t**
- (iv) message reception (**rec**ei**v**e), **rcv**
- (v) response sending (reply), **rp**
- (vi) the null activity, which is used for synchronization purpose (**nop**).



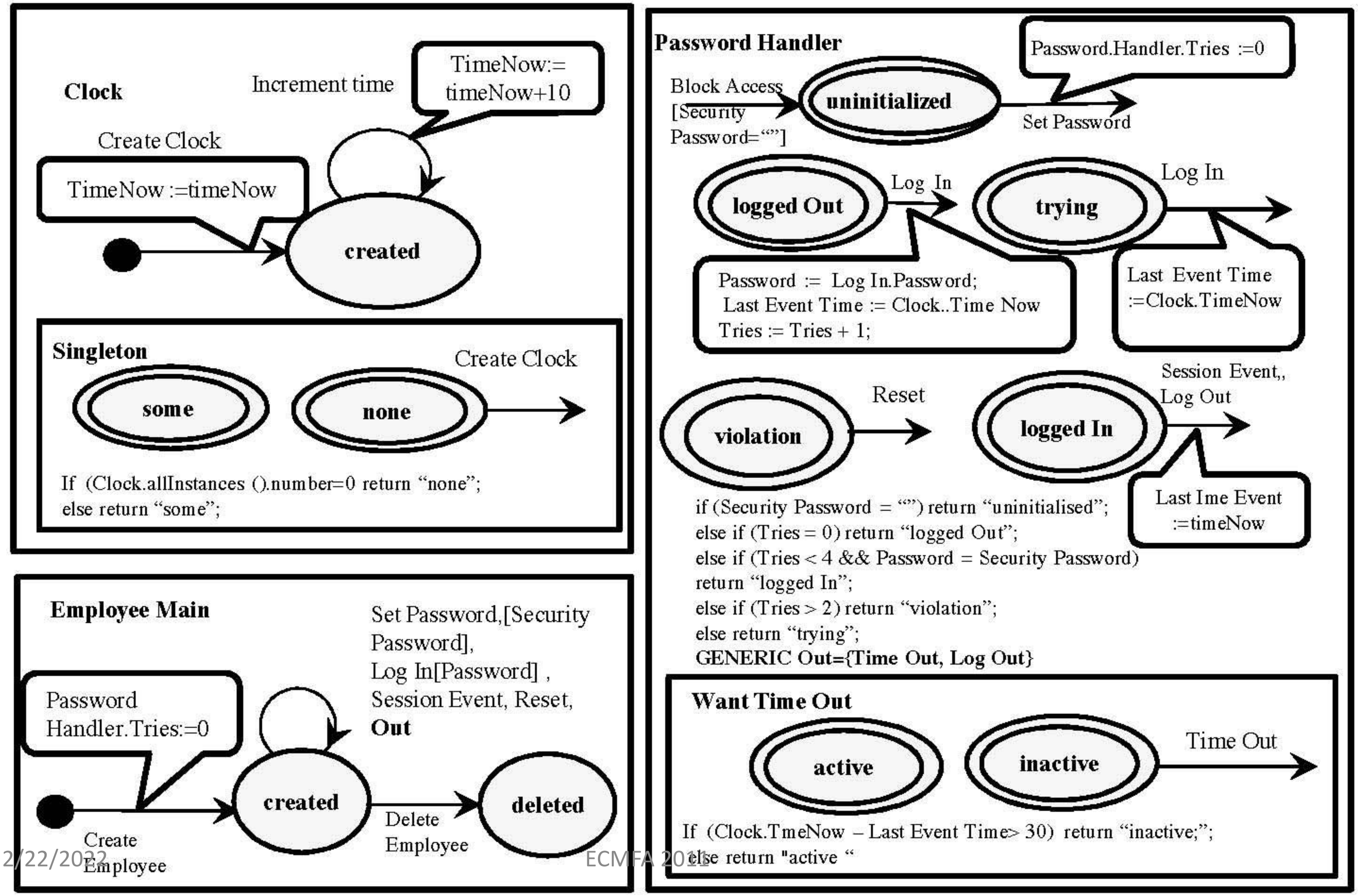
Fragment only3times

P->a0

hook -> {a1, a2,a3}

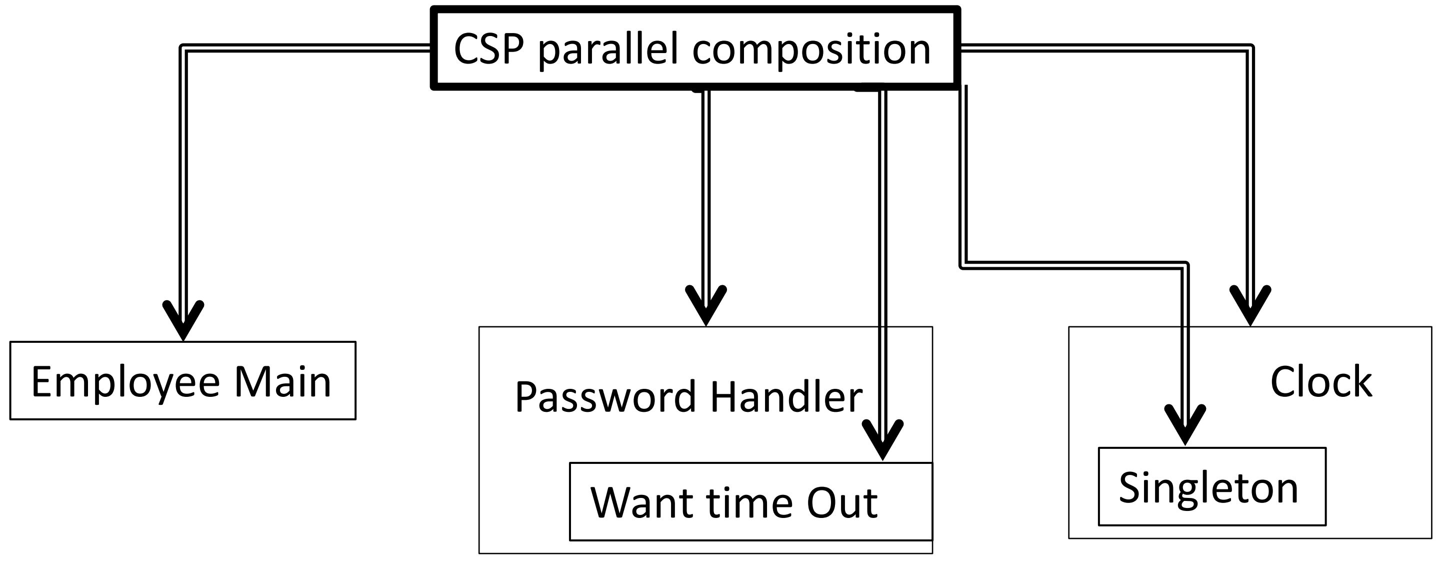
S->a4

# AOM revision of Mixins: Protocol Modelling



Clock
Increment time
TimeNow:= timeNow+10
Create Clock
TimeNow :=timeNow
created
Singleton
Create Clock
some
none
If (Clock.allInstances ().number=0 return "none"; else return "some";
Employee Main
Set Password,[Security Password], Log In[Password] , Session Event, Reset, **Out**
Password Handler.Tries:=0
created
deleted
Create Employee
Delete Employee
Password Handler
Password.Handler.Tries :=0
Block Access [Security Password=""]
uninitialized
Set Password
logged Out
Log In
trying
Log In
Password := Log In.Password; Last Event Time := Clock..Time Now Tries := Tries + 1;
Last Event Time :=Clock.TimeNow
violation
Reset
logged In
Session Event,, Log Out
Last Ime Event :=timeNow
if (Security Password = "") return "uninitialised"; else if (Tries = 0) return "logged Out"; else if (Tries < 4 && Password = Security Password) return "logged In"; else if (Tries > 2) return "violation"; else return "trying"; GENERIC Out={Time Out, Log Out}
Want Time Out
active
inactive
Time Out
If (Clock.TmeNow – Last Event Time> 30) return "inactive;"; else return "active "

# CSP parallel Composition of Mixins



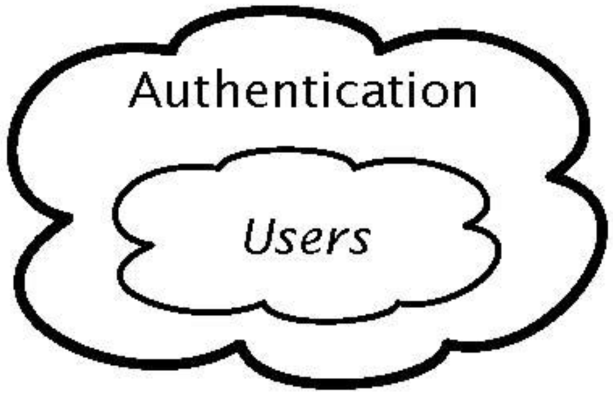
# Properties of Protocol Modeles

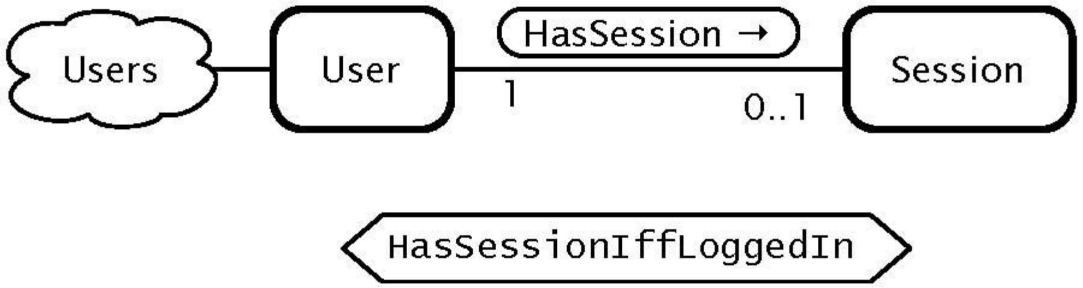
- Localization of concerns
- Synchronous handling of an event by different protocol machines
- Deterministic  behaviour
- Invasive aspects are not  possible
- CSP composition preserves the order of sequences  of events and makes possible localization of reasoning

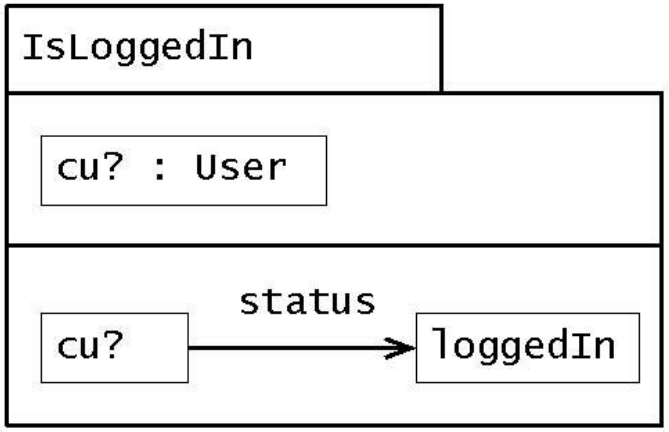
# AOM revision of design by contract: Visual Contract Language (VCL)

- DbC extends the ordinary definition of abstract data types with preconditions, postconditions and invariants.
- If a precondition is violated, the effect of the section of code becomes **undefined** and thus may or may not carry out its intended work.

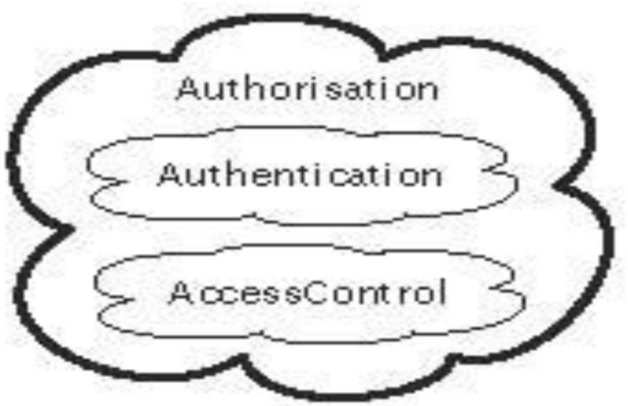
# AOM revision of design by contract: Visual Contract Language (VCL)

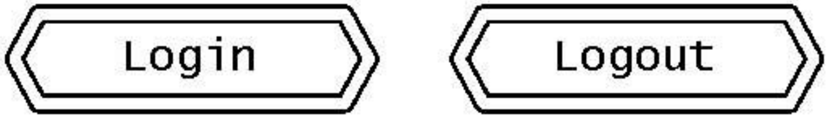


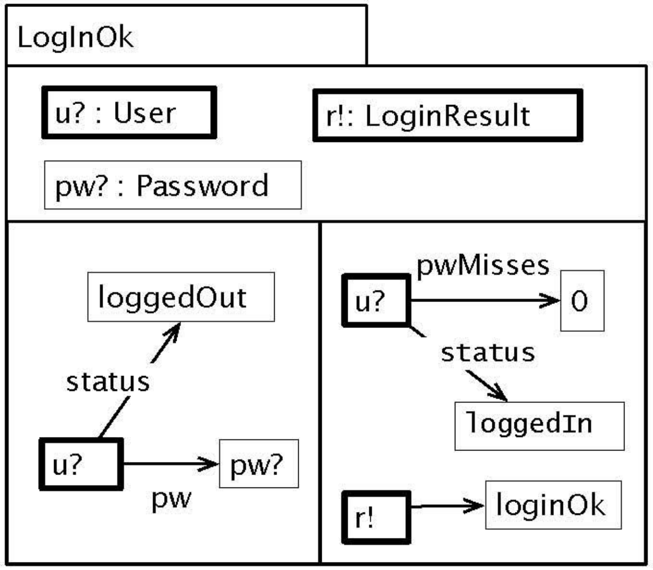


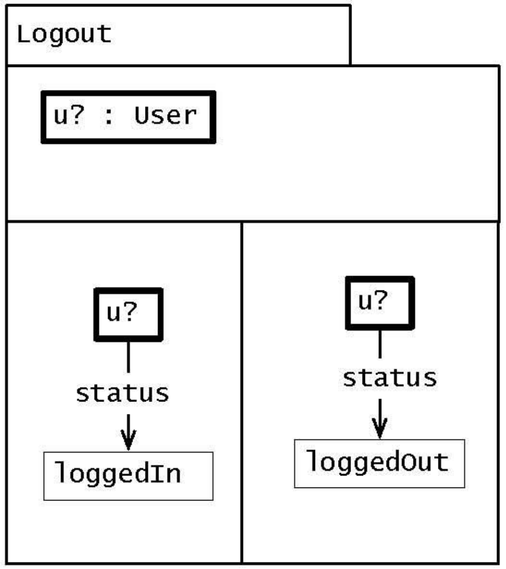


# VCL: Package Authentication

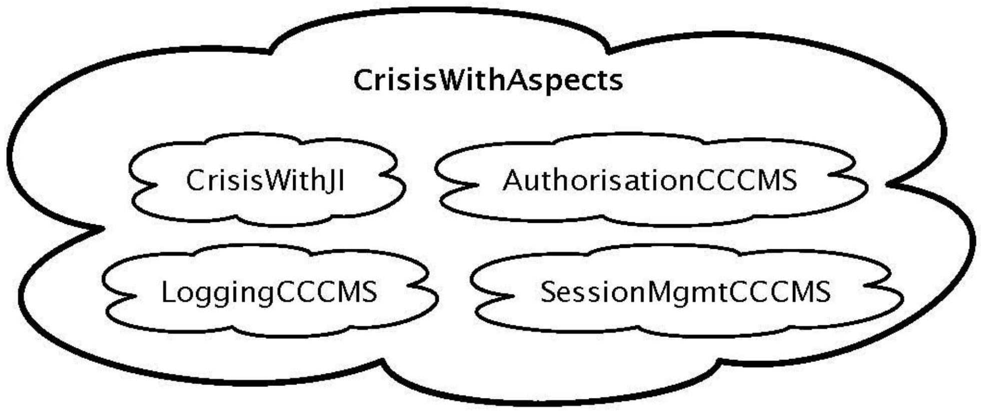


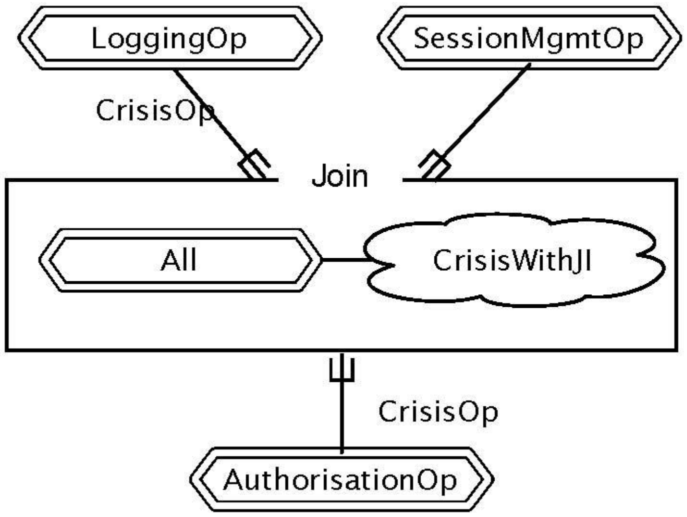


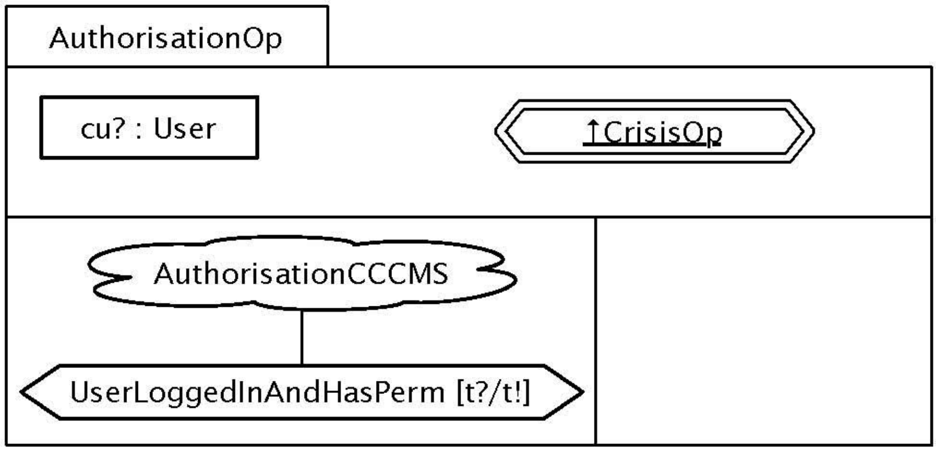




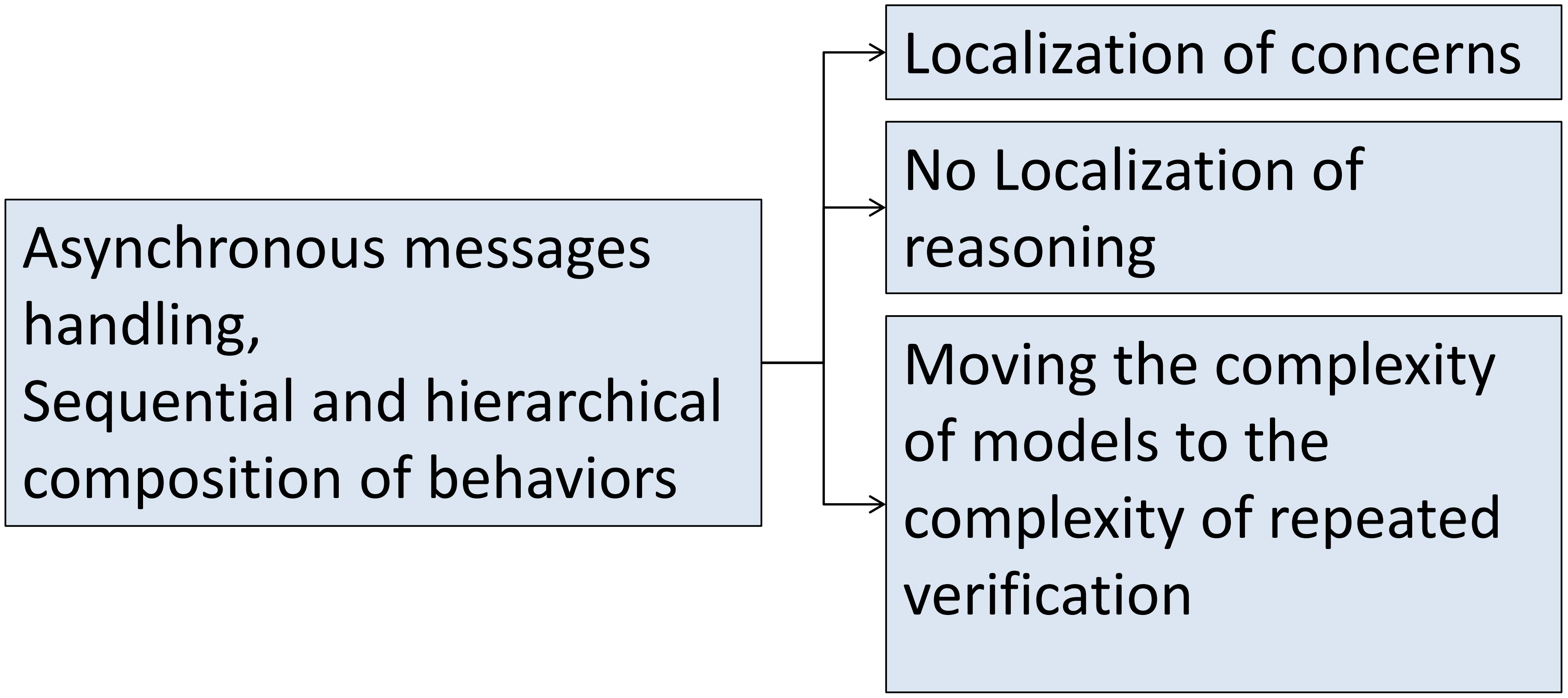
# VCL: Crisis with Aspects



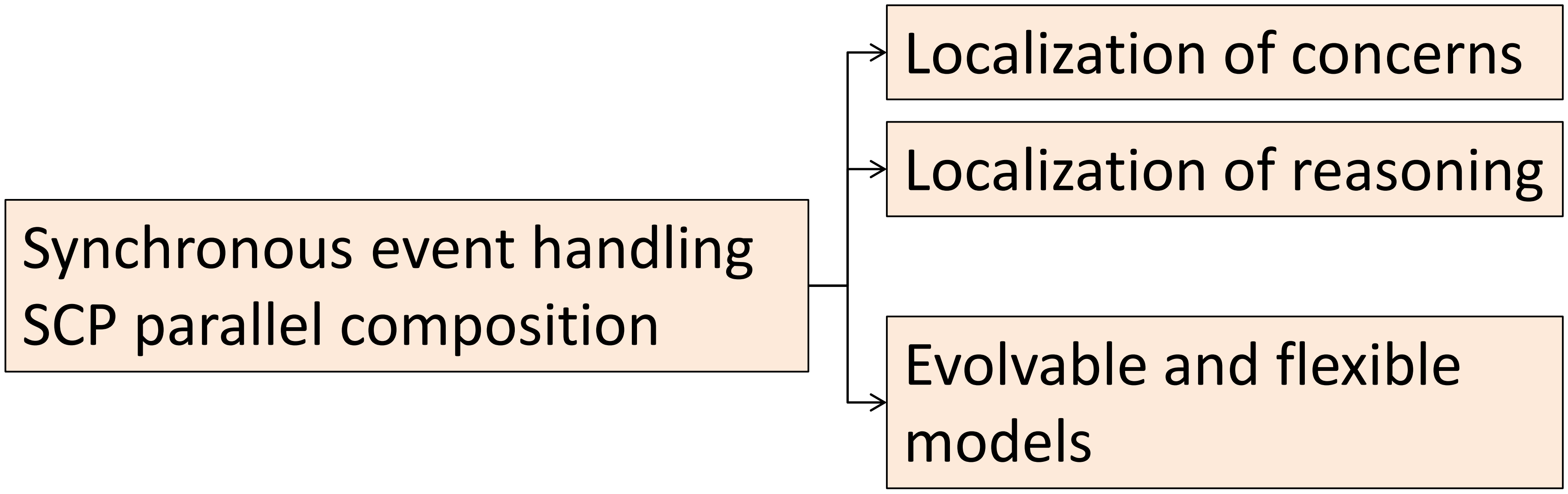




# Conclusions


Asynchronous messages handling,
Sequential and hierarchical composition of behaviors
Localization of concerns
No Localization of reasoning
Moving the complexity of models to the complexity of repeated verification

# Conclusions


Synchronous event handling
SCP parallel composition
Localization of concerns
Localization of reasoning
Evolvable and flexible models

## Conclusions

- Having the experience of the last ten years with aspects at different level of abstraction, it is possible to predict the effectiveness of aspect-orientation in any new language .