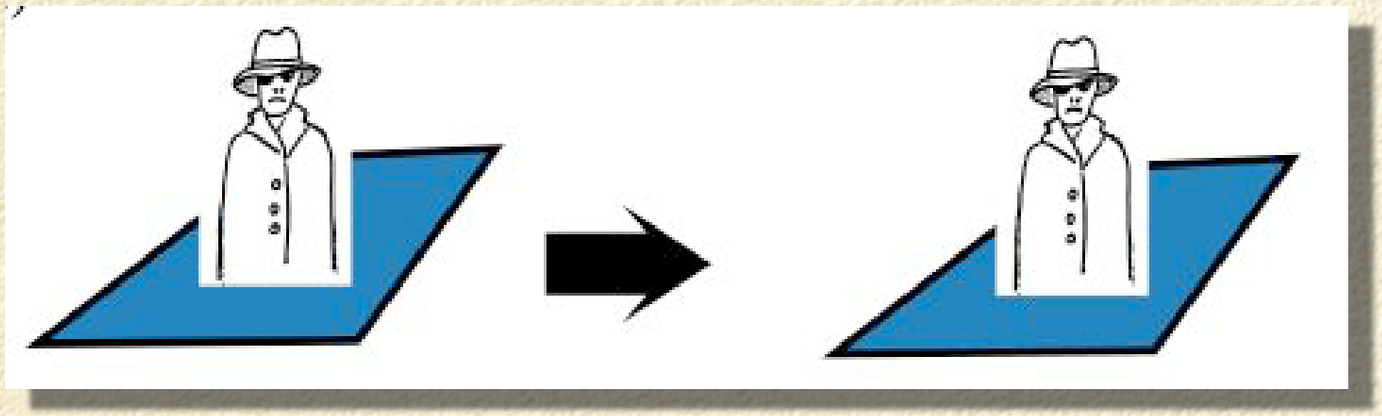
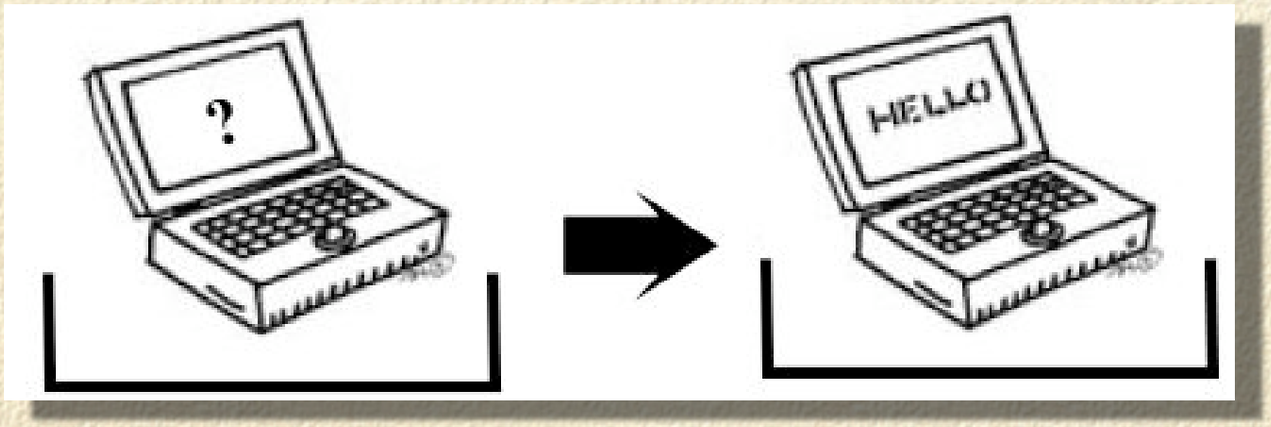
Nets within Nets

nets - within - nets



location A location B

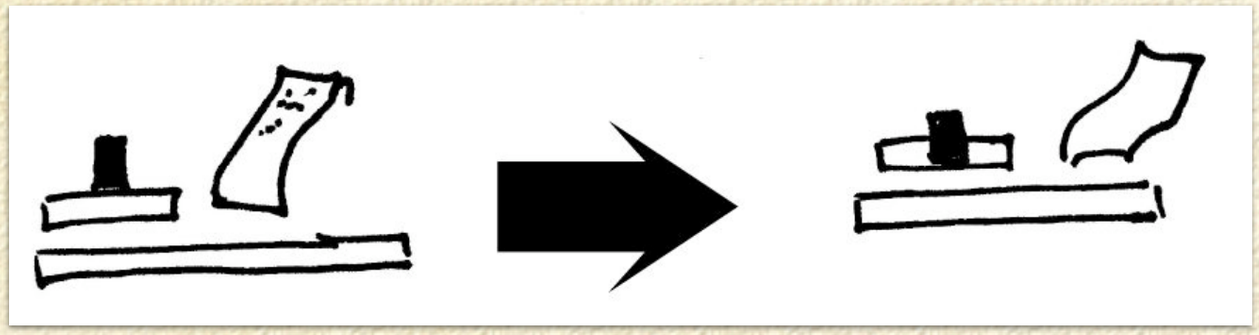
software agents



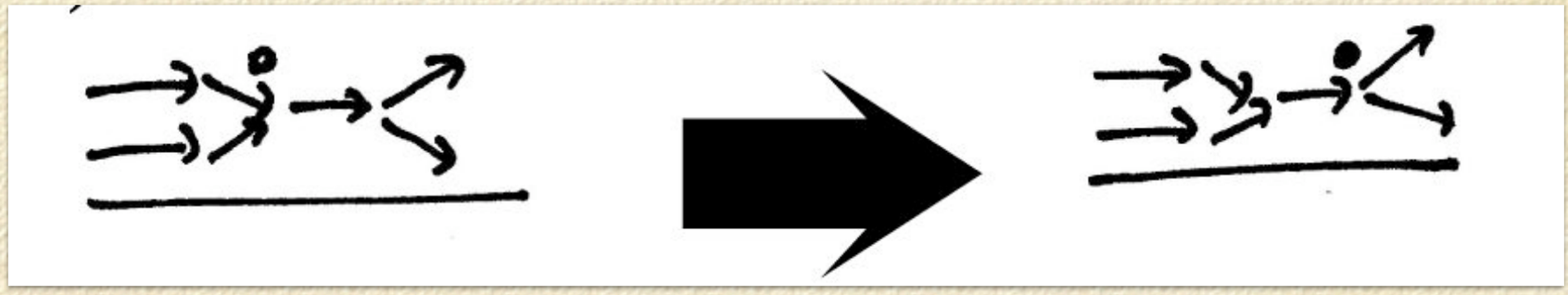
security environment A

security environment B

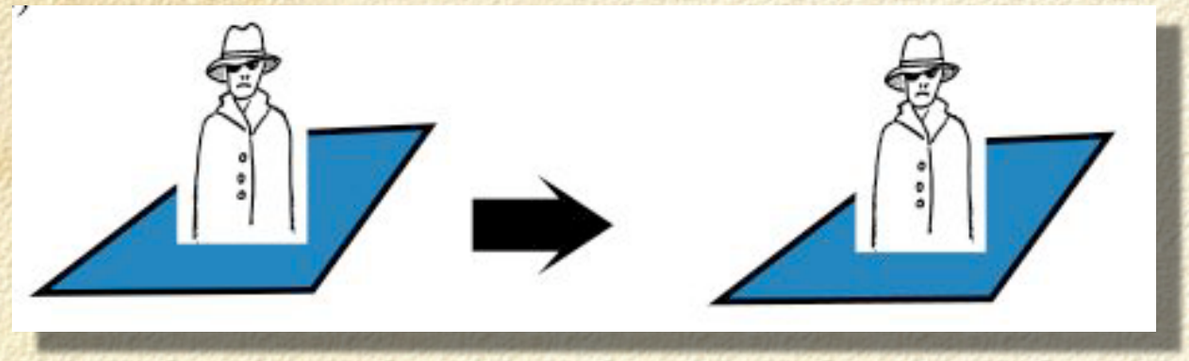
mobility



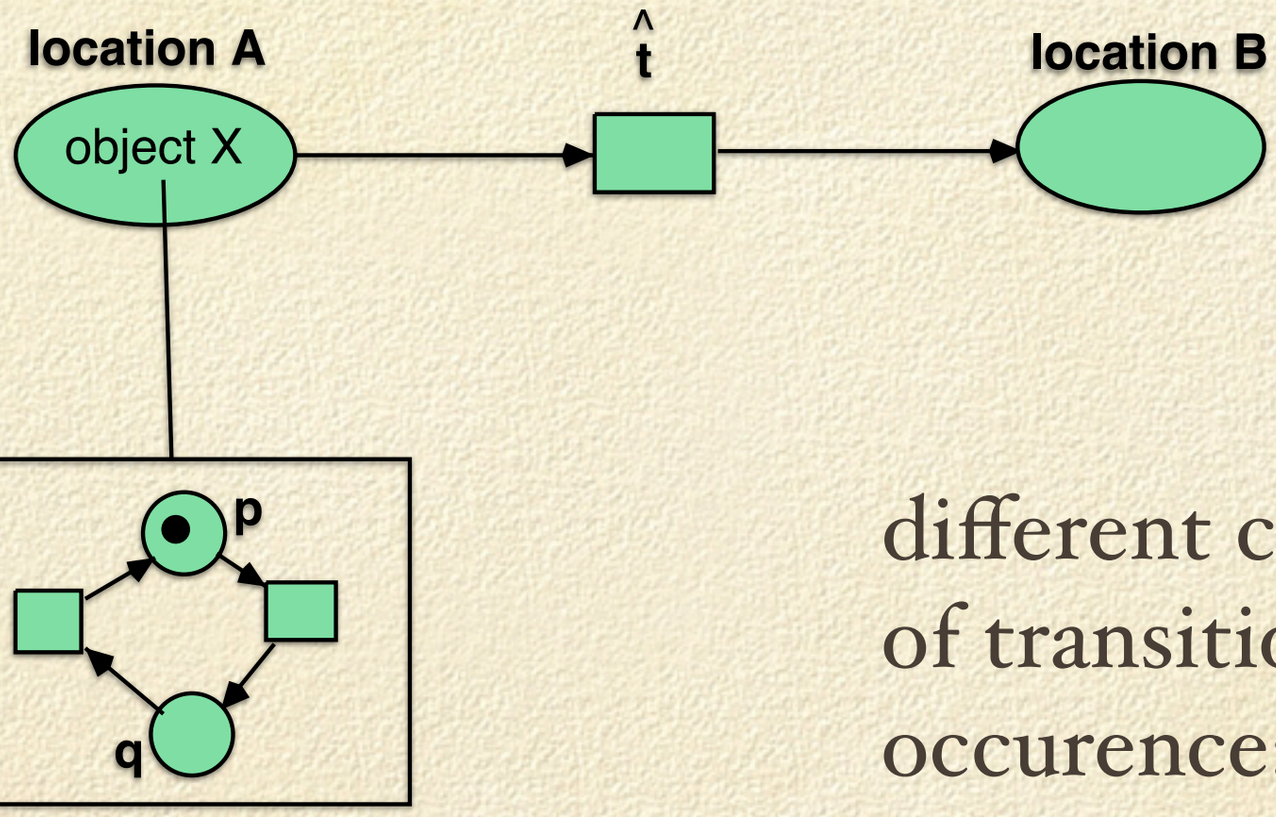
flexible manufacturing



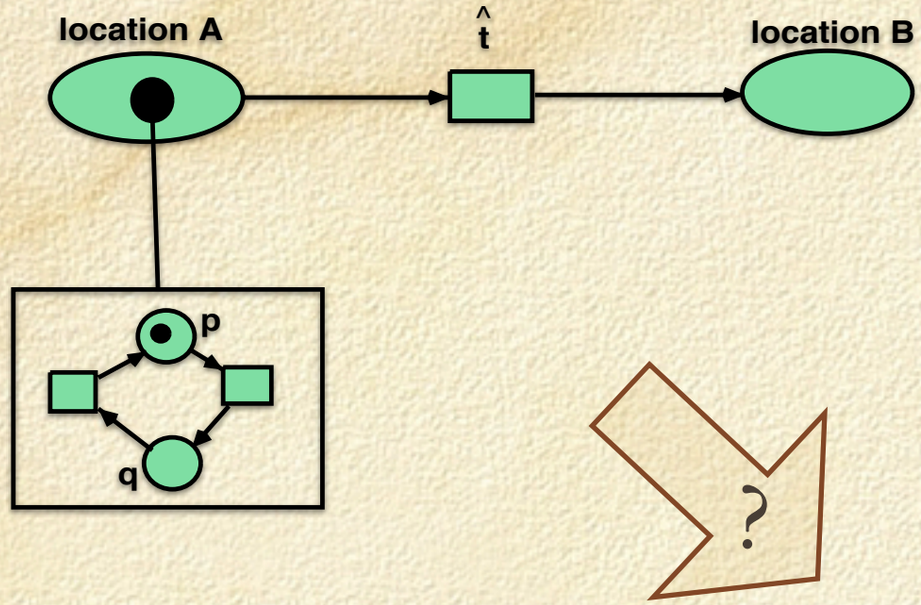
workflow



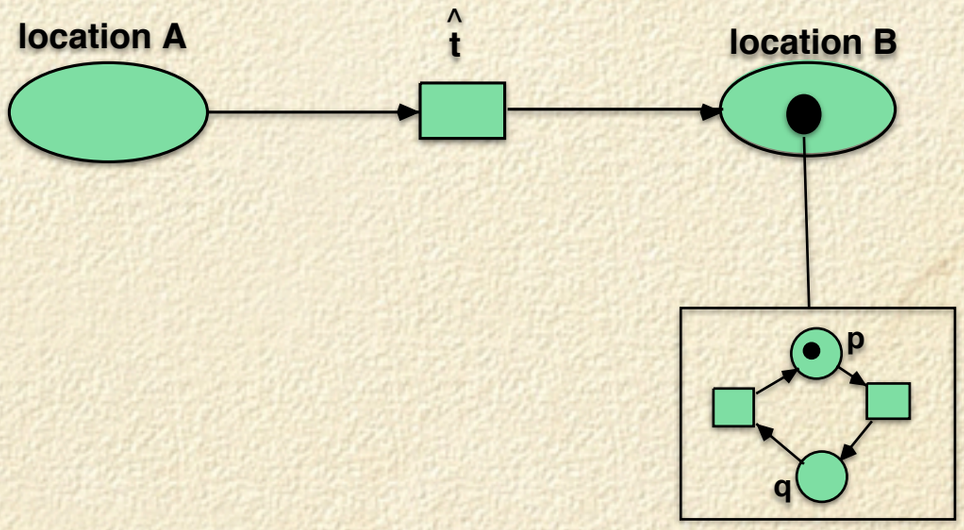
location A location B

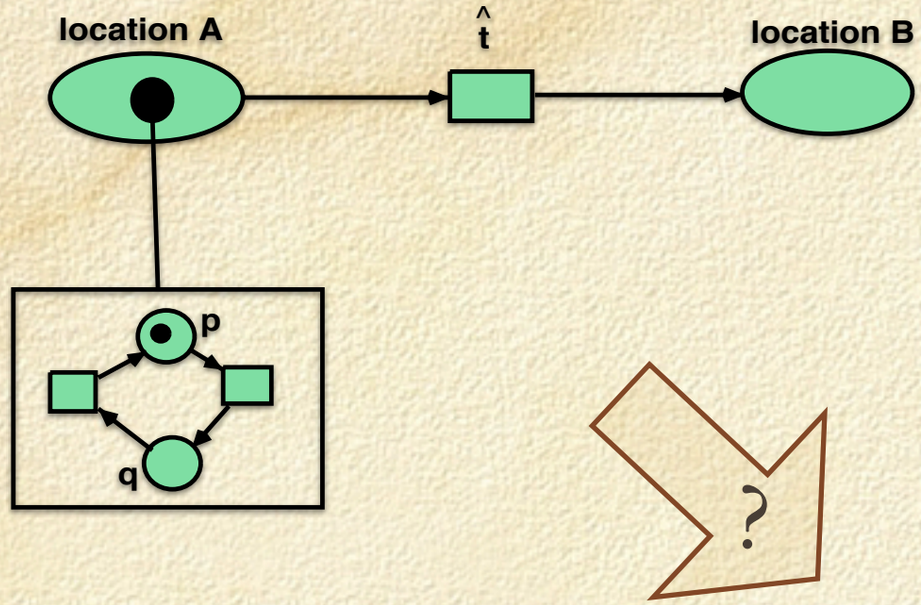


different cases of transition occurence:

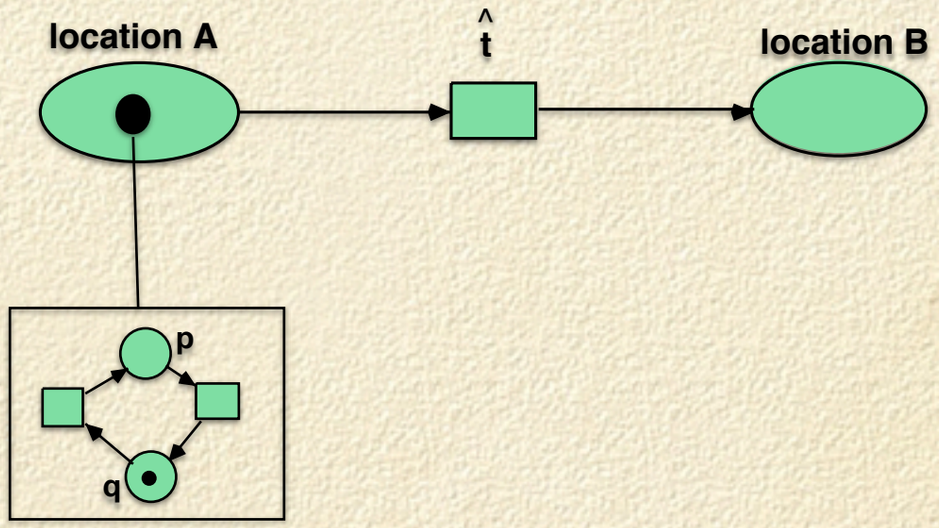


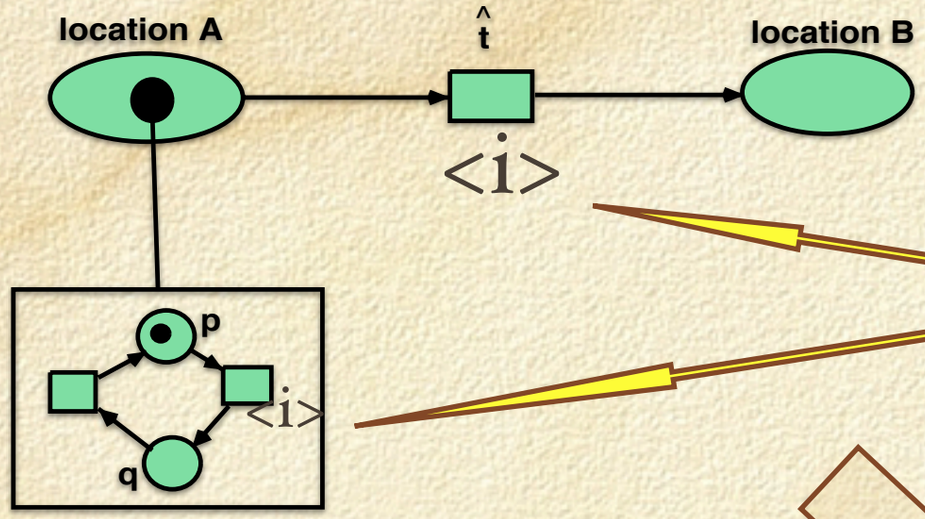
transport



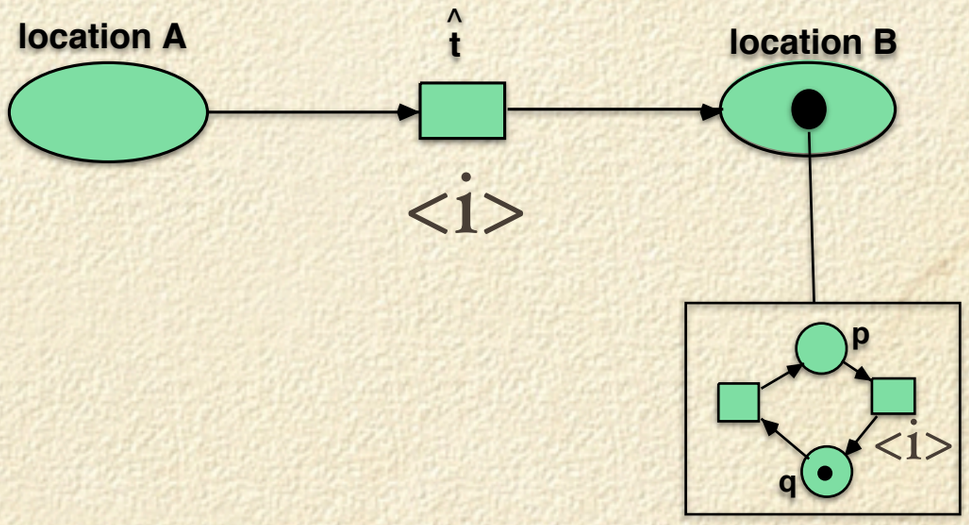


autonomous action



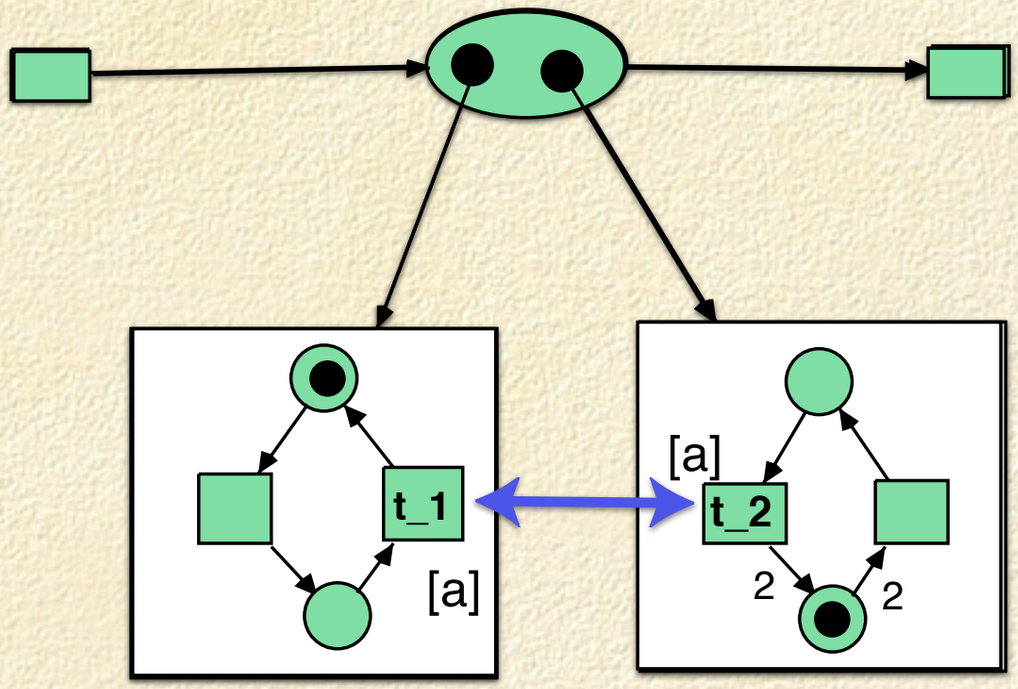


interaction



object net marking has changed

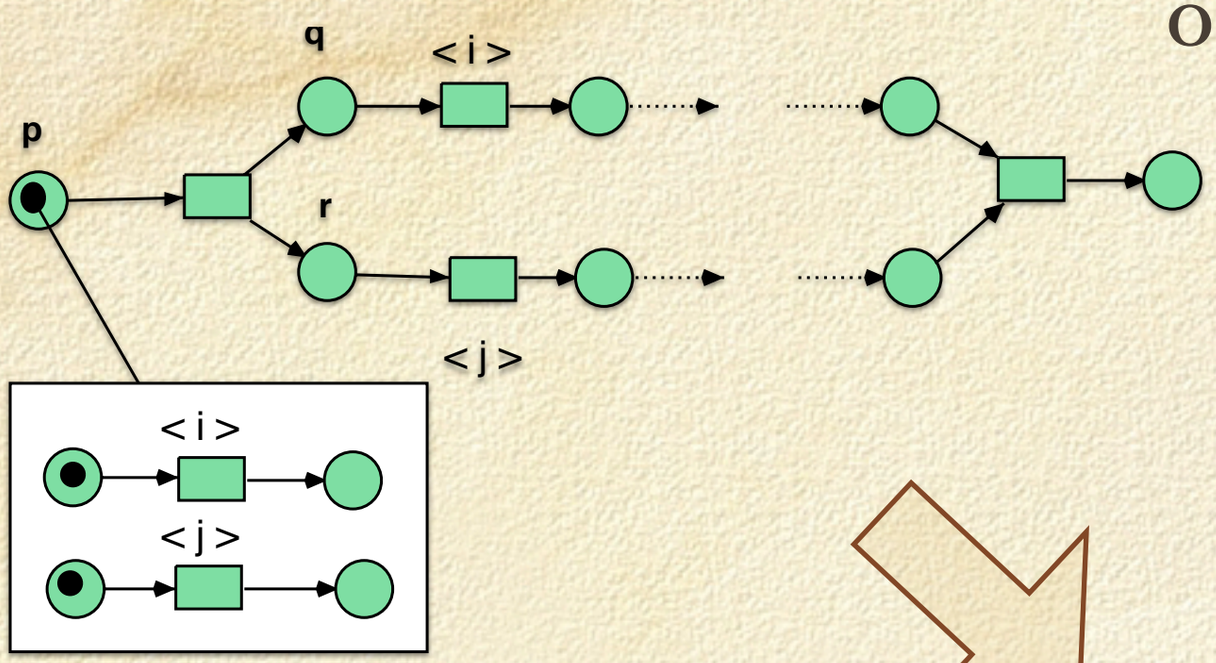
object interaction



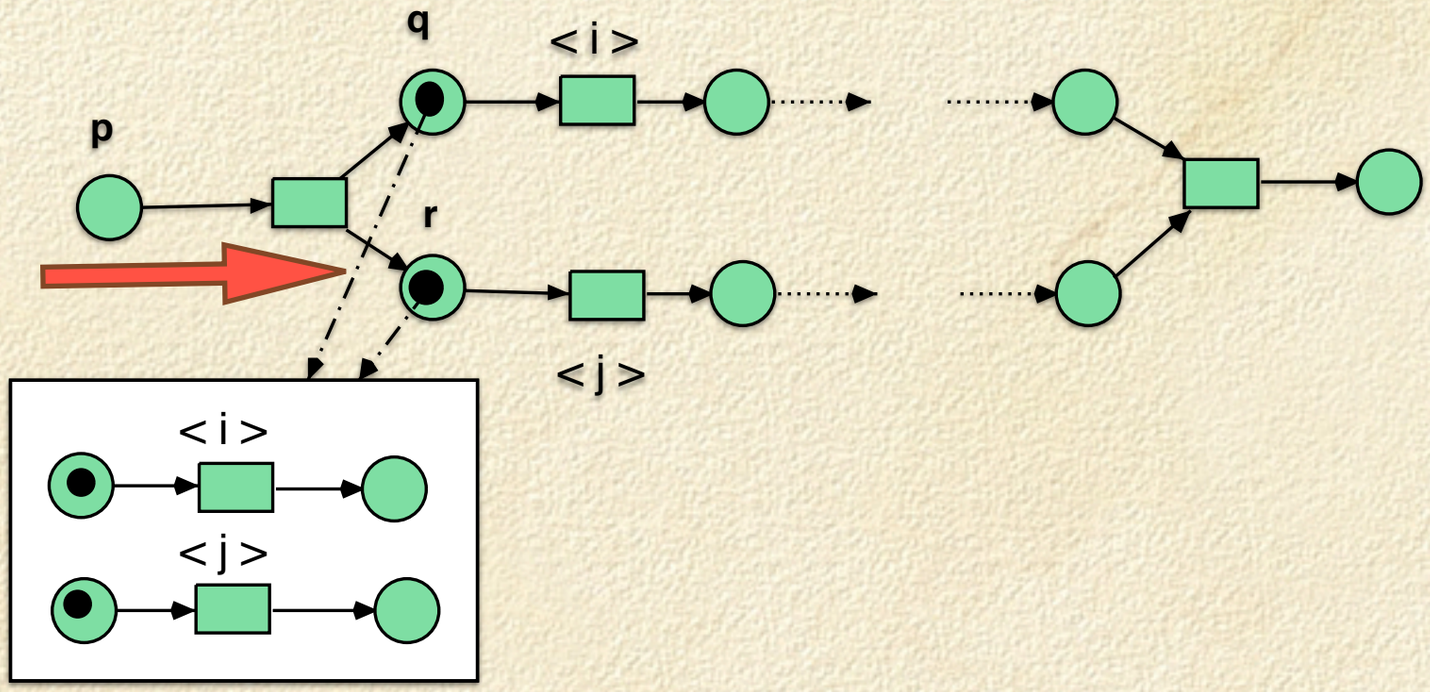
$$(t_1, t_2) \in \sigma$$

object nets interact
without changing
environment

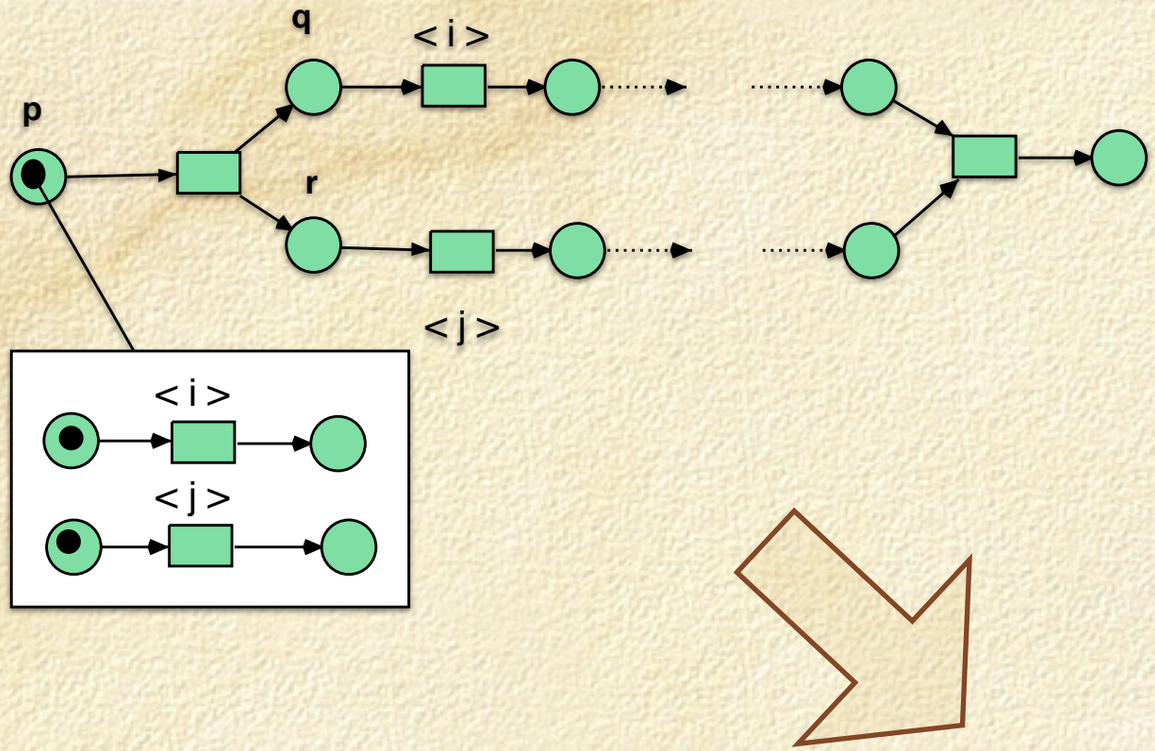
creating distributed objects



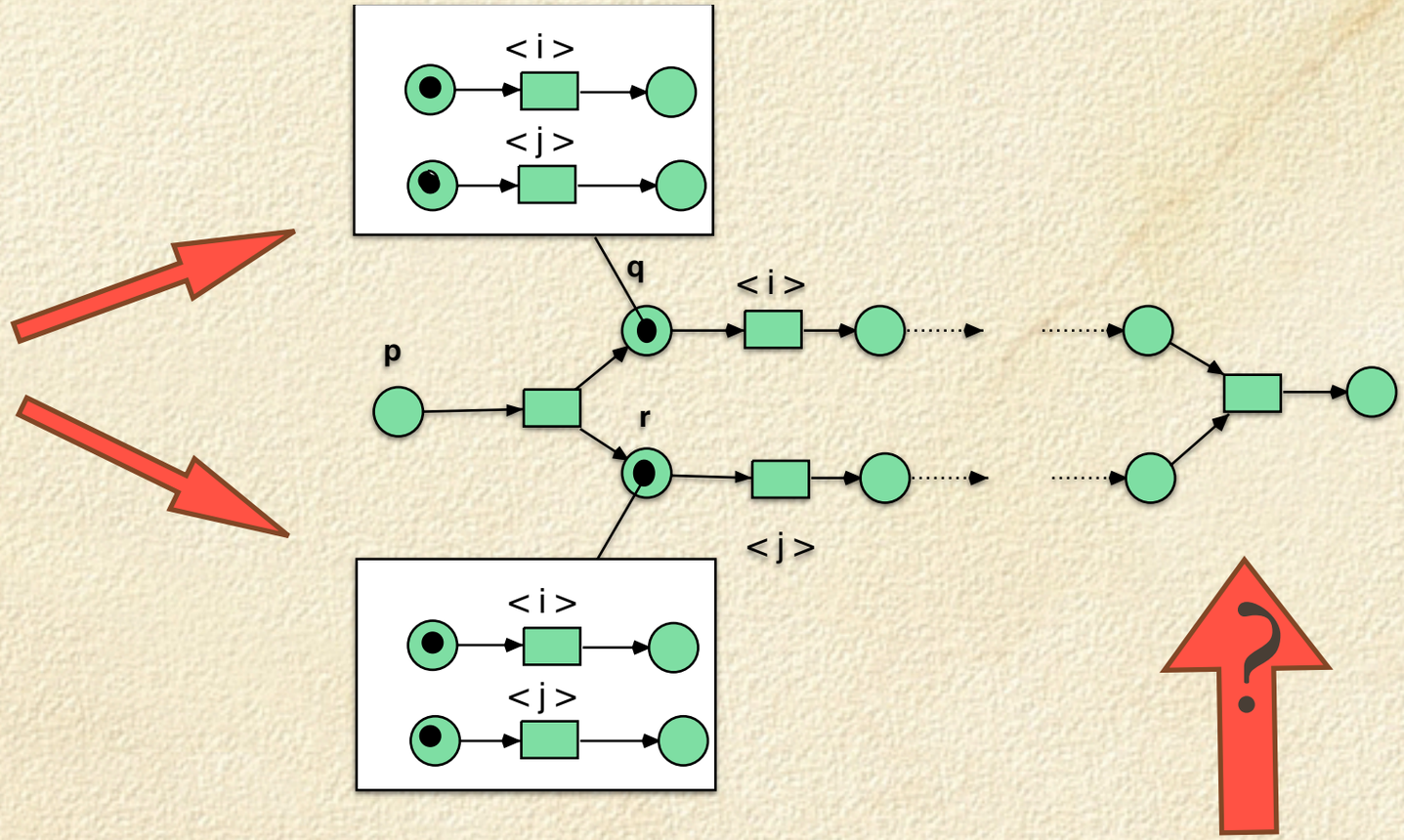
reference semantics



creating distributed objects

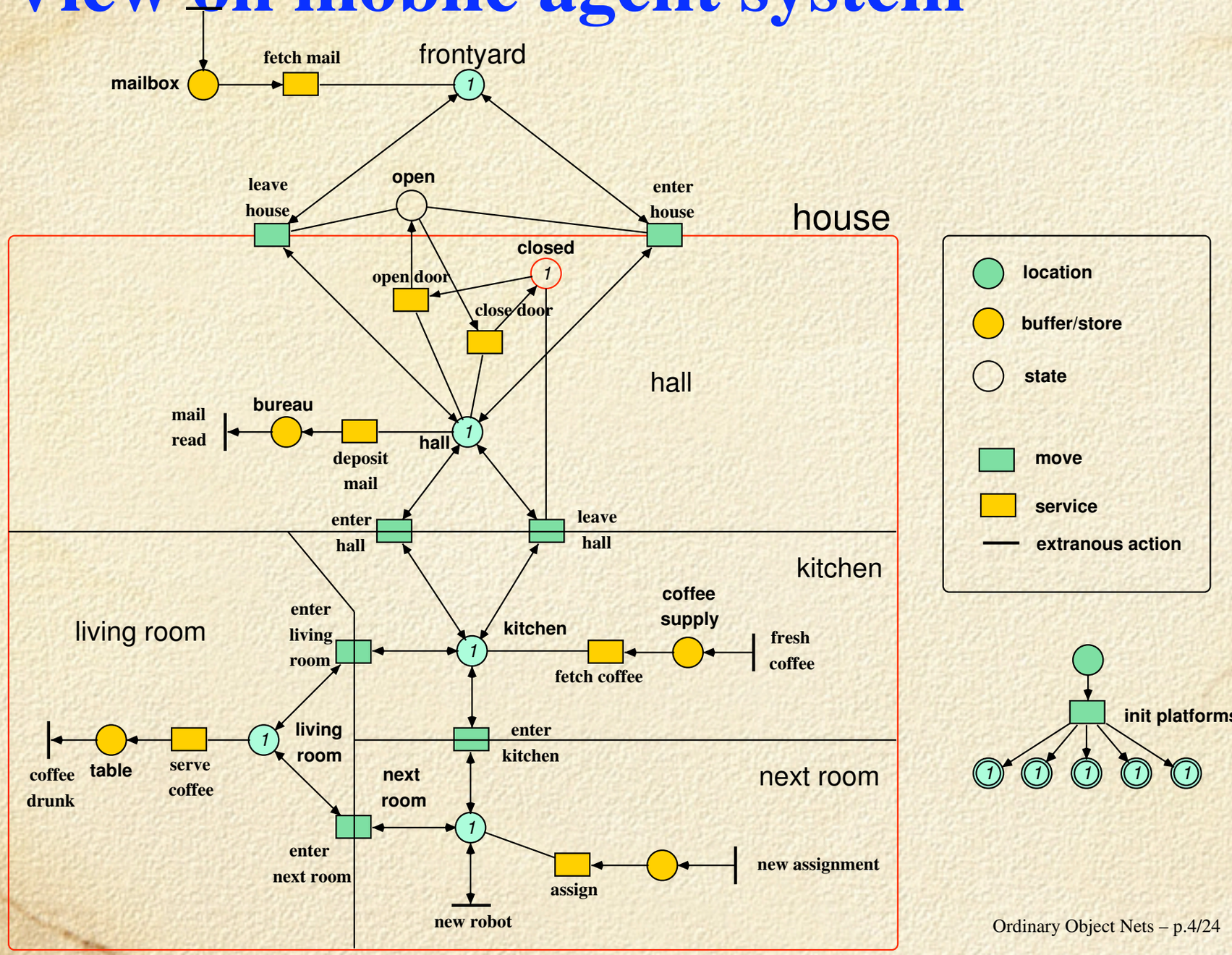


value semantics

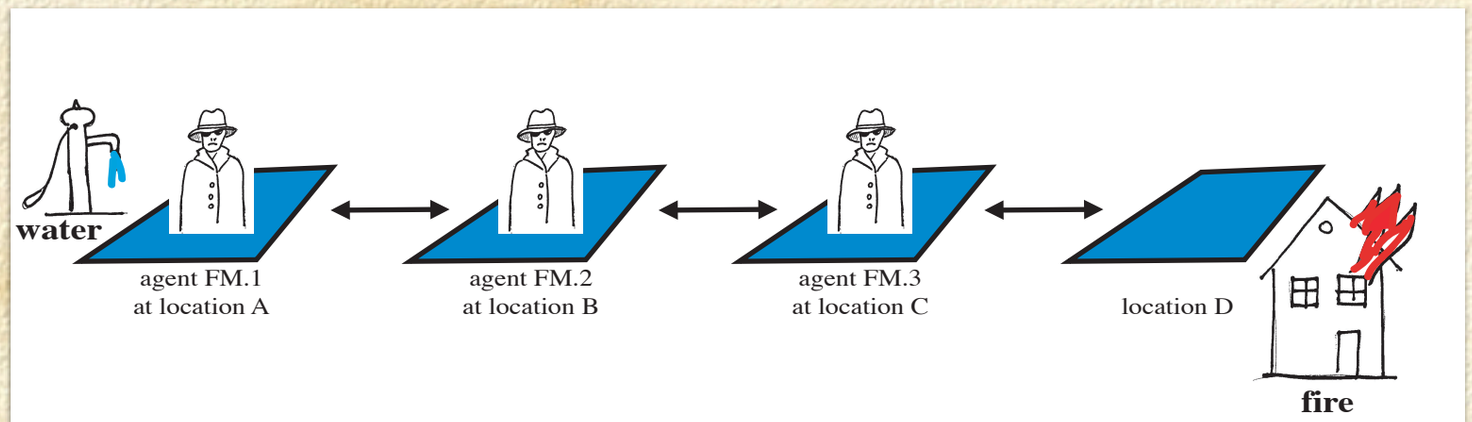


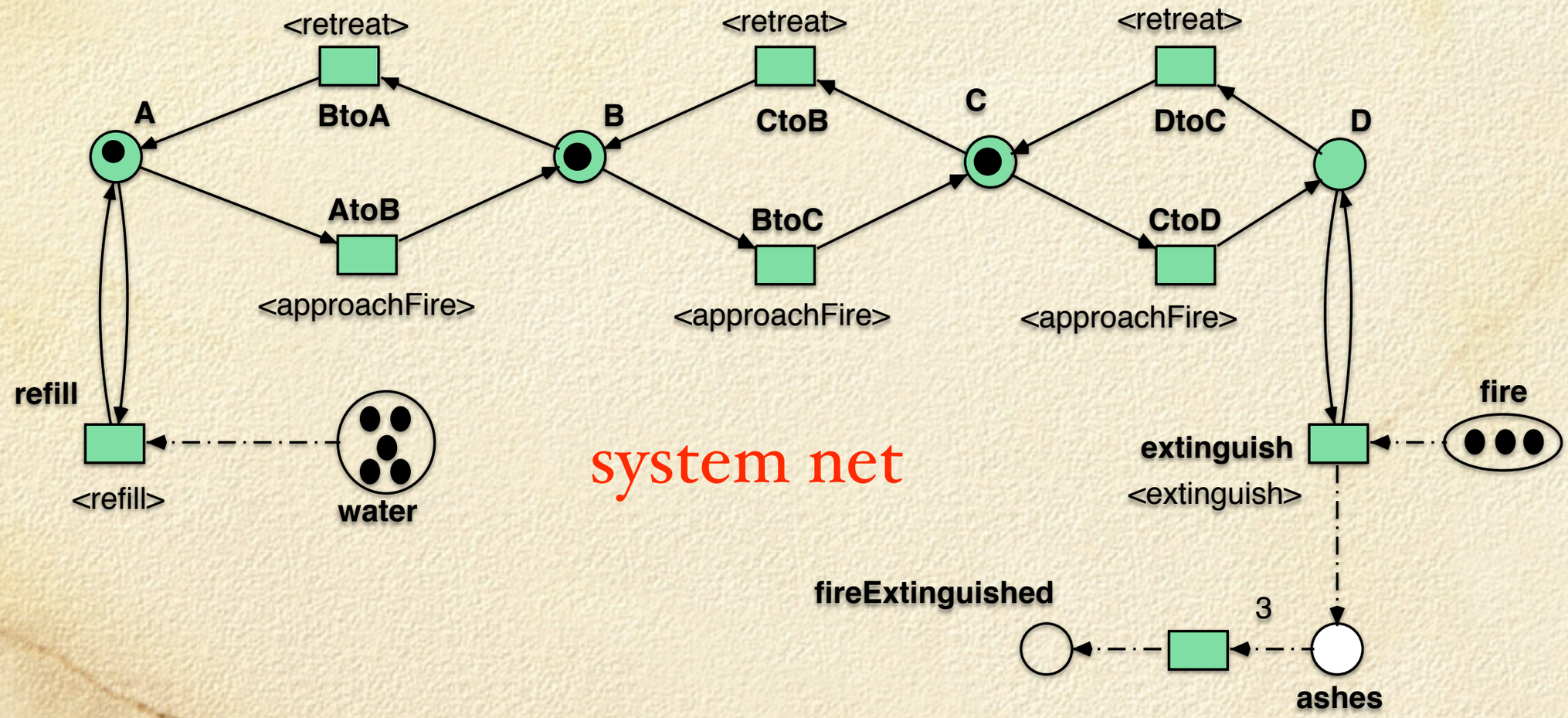
Reference semantics

View on mobile agent system



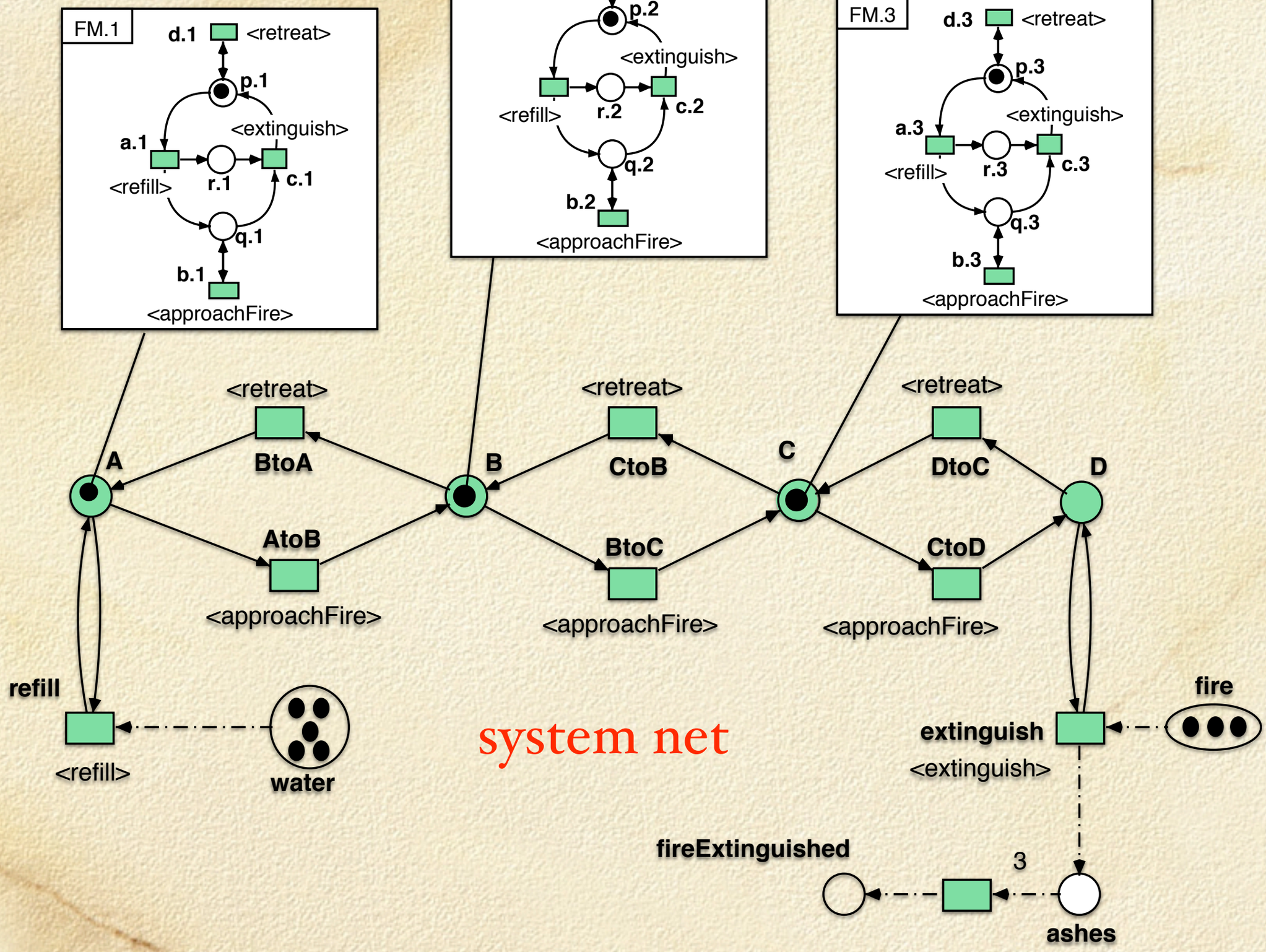
Ordinary Object Nets – p.4/24


water
agent FM.1
at location A
agent FM.2
at location B
agent FM.3
at location C
location D
fire


<retreat>
<retreat>
<retreat>
A
BtoA
B
CtoB
C
DtoC
D
AtoB
BtoC
CtoD
<approachFire>
<approachFire>
<approachFire>
refill
<refill>
water
fire
extinguish
<extinguish>
fireExtinguished
3
ashes

system net

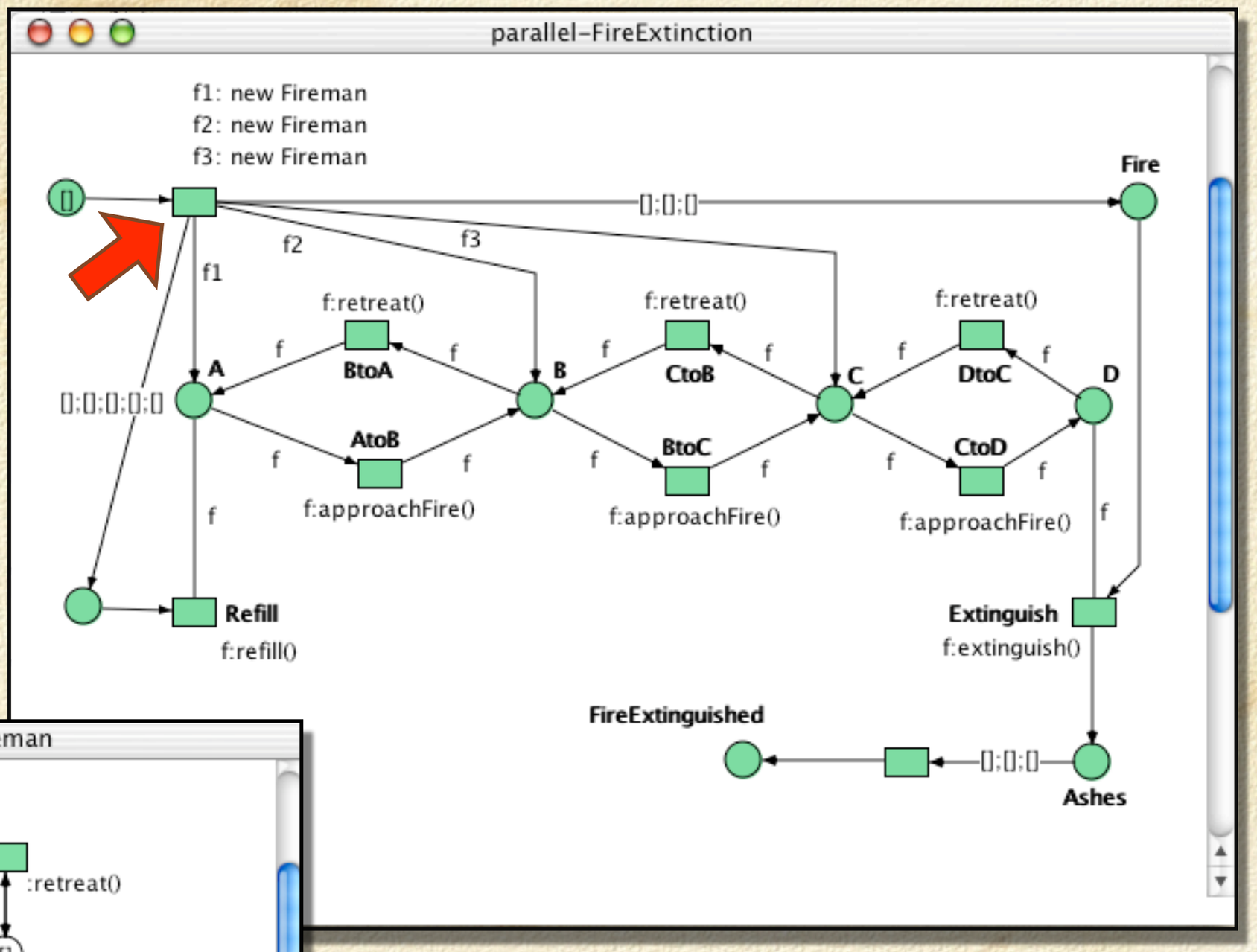
3 object nets

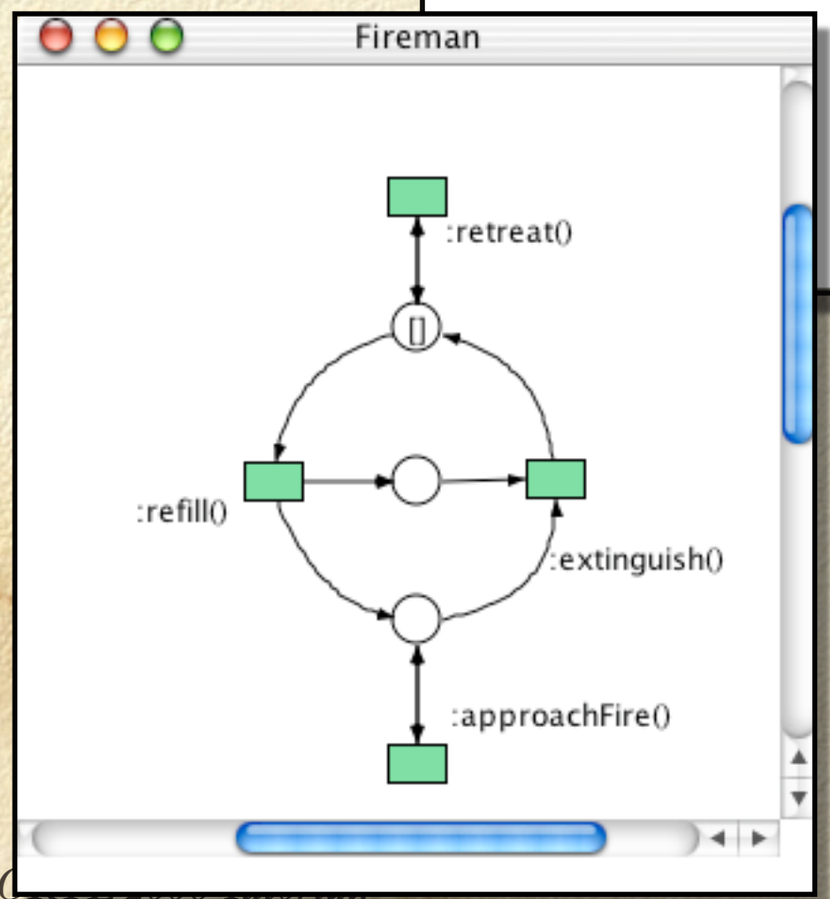


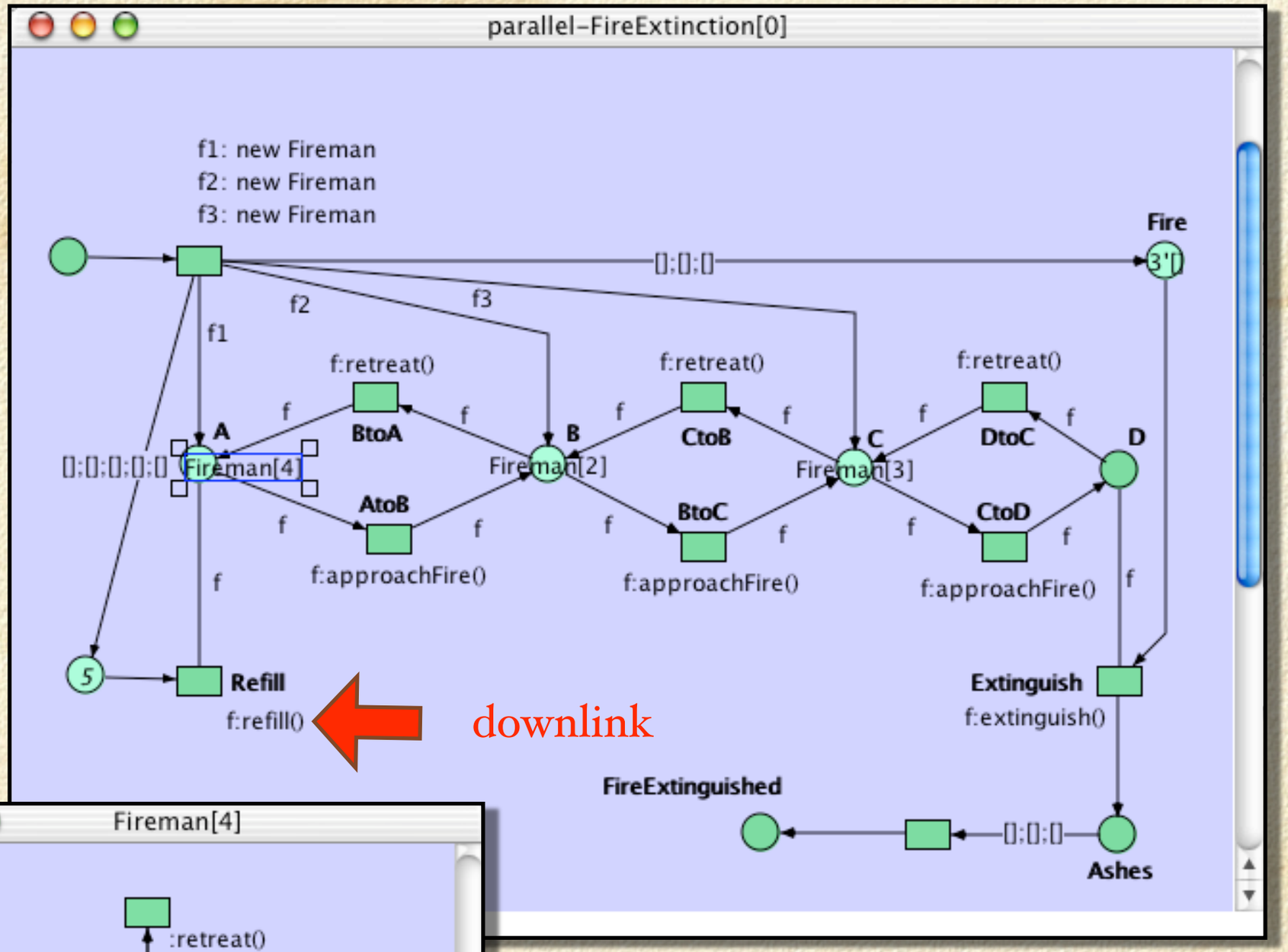
system net

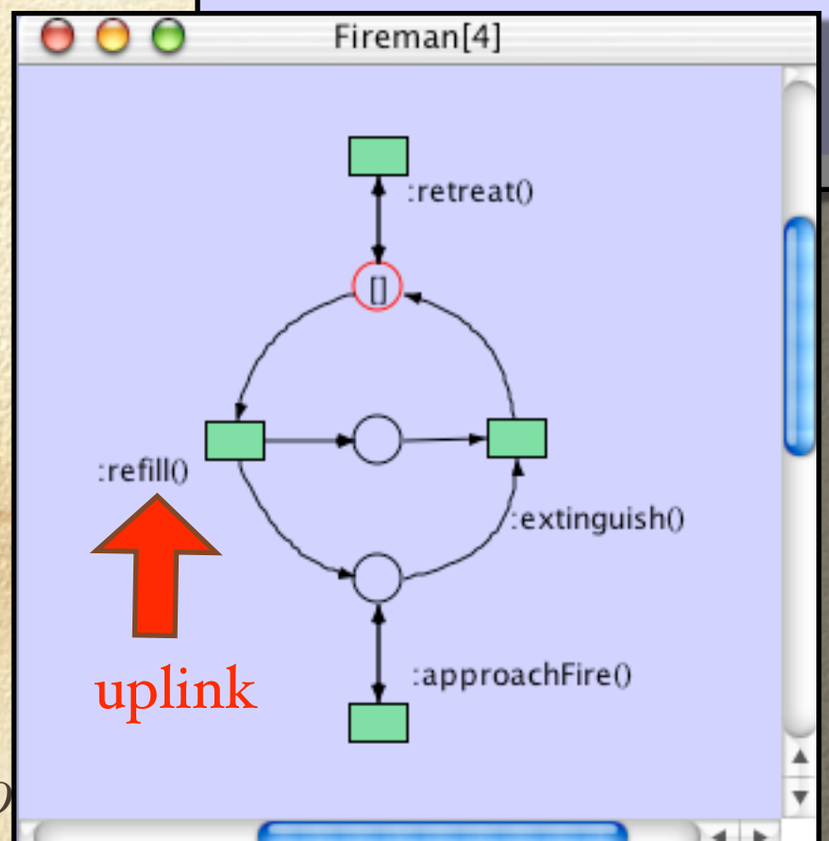
parallel fire extinction/formal model

The RENEW tool

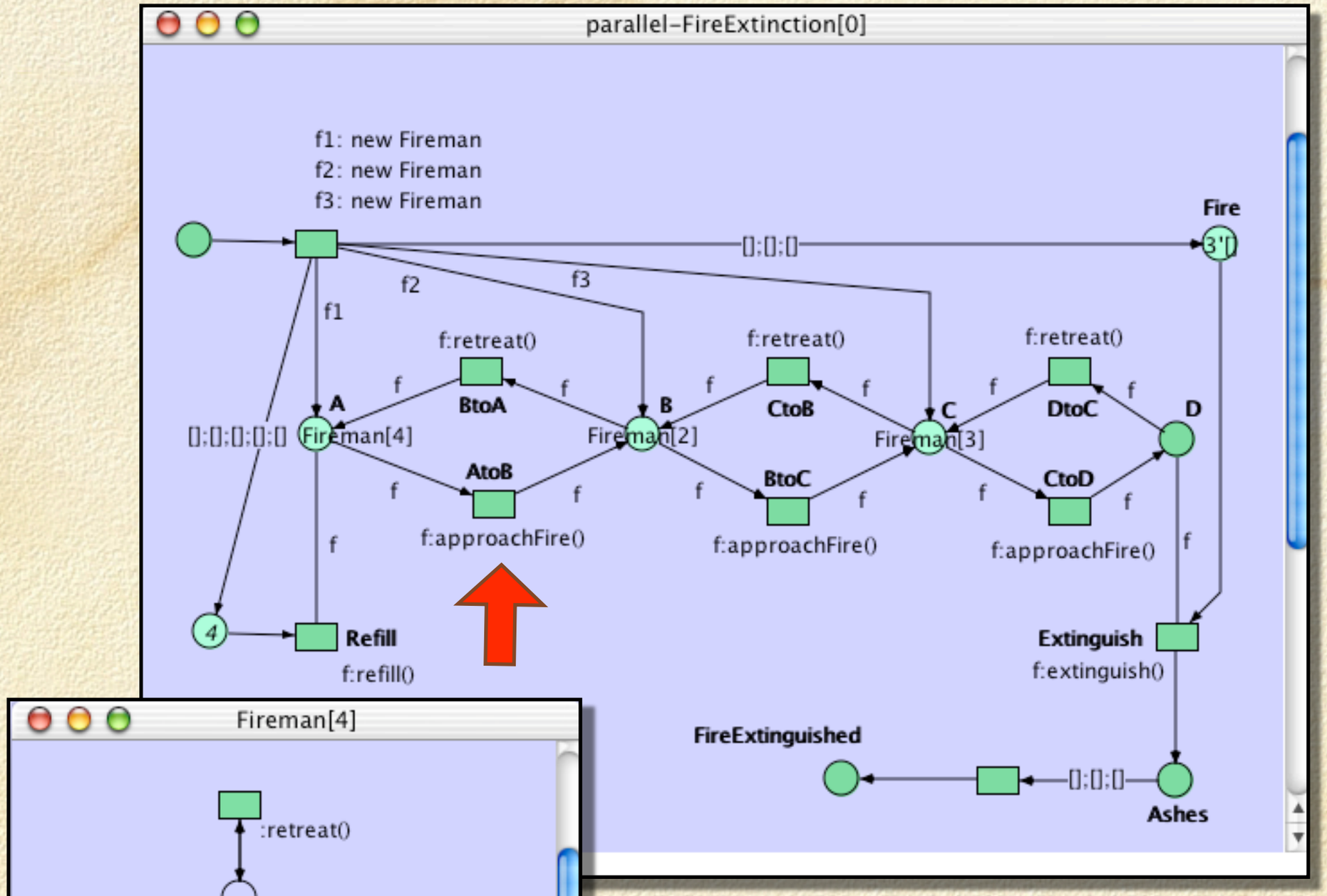

parallel-FireExtinction
f1: new Fireman
f2: new Fireman
f3: new Fireman
[]
[];[];[]
Fire
f1
f2
f3
f:retreat()
f:retreat()
f:retreat()
A
BtoA
B
CtoB
C
DtoC
D
[];[];[];[];[]
AtoB
BtoC
CtoD
f:approachFire()
f:approachFire()
f:approachFire()
Refill
f:refill()
Extinguish
f:extinguish()
FireExtinguished
[];[];[]
Ashes

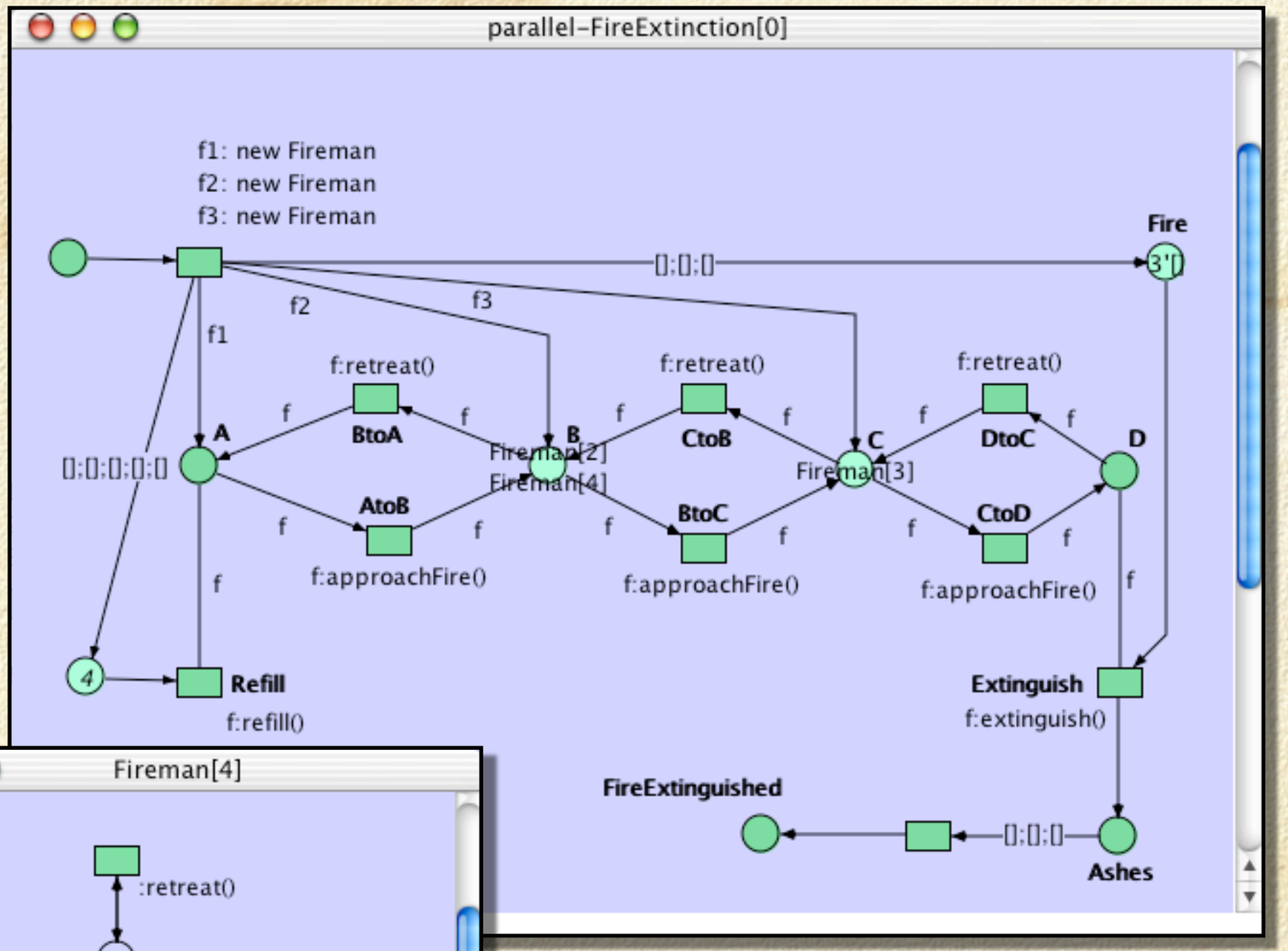

Fireman
:retreat()
[]
:refill()
:extinguish()
:approachFire()

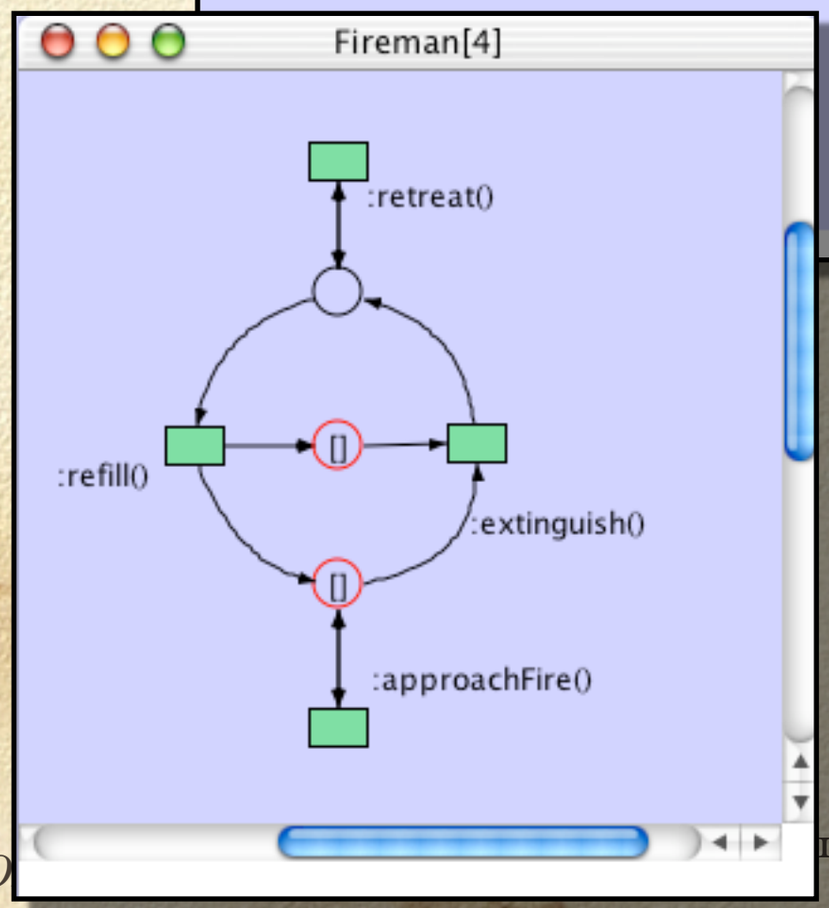




parallel fire extinction/renew

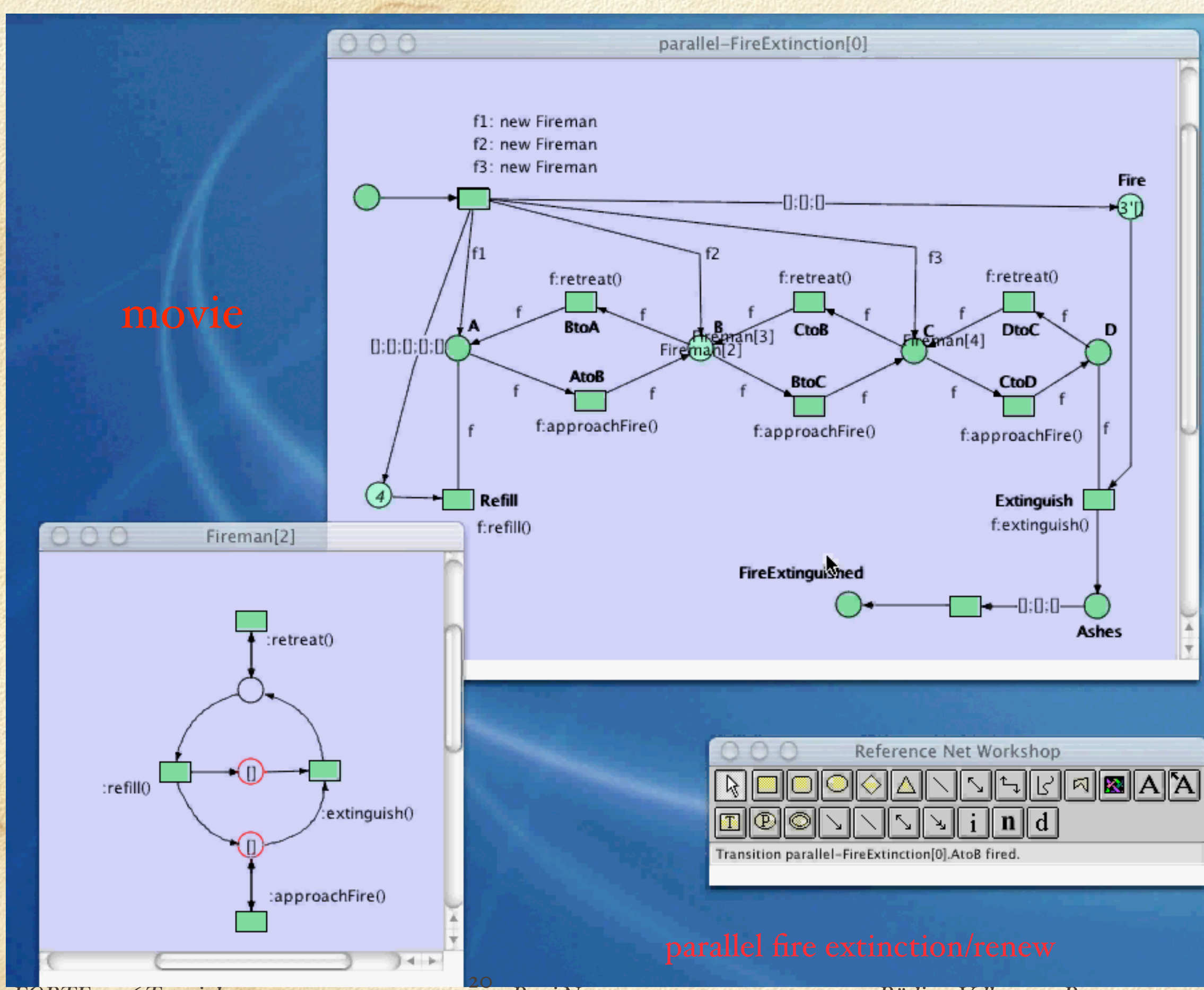

parallel-FireExtinction[0]
f1: new Fireman
f2: new Fireman
f3: new Fireman
[];[];[]
Fire
3'[]
f1
f2
f3
f:retreat()
f:retreat()
f:retreat()
A
B
C
D
BtoA
CtoB
DtoC
[];[];[];[];[]
Fireman[4]
Fireman[2]
Fireman[3]
AtoB
BtoC
CtoD
f:approachFire()
f:approachFire()
f:approachFire()
4
Refill
f:refill()
Extinguish
f:extinguish()
FireExtinguished
[];[];[]
Ashes
Fireman[4]
:retreat()
:refill()
:extinguish()
:approachFire()


parallel-FireExtinction[0]
f1: new Fireman
f2: new Fireman
f3: new Fireman
[];[];[]
Fire
3'[]
f1
f2
f3
f:retreat()
f:retreat()
f:retreat()
A
BtoA
B
CtoB
C
DtoC
D
[];[];[];[];[]
Fireman[2]
Fireman[4]
Fireman[3]
AtoB
BtoC
CtoD
f:approachFire()
f:approachFire()
f:approachFire()
4
Refill
f:refill()
Extinguish
f:extinguish()
FireExtinguished
[];[];[]
Ashes

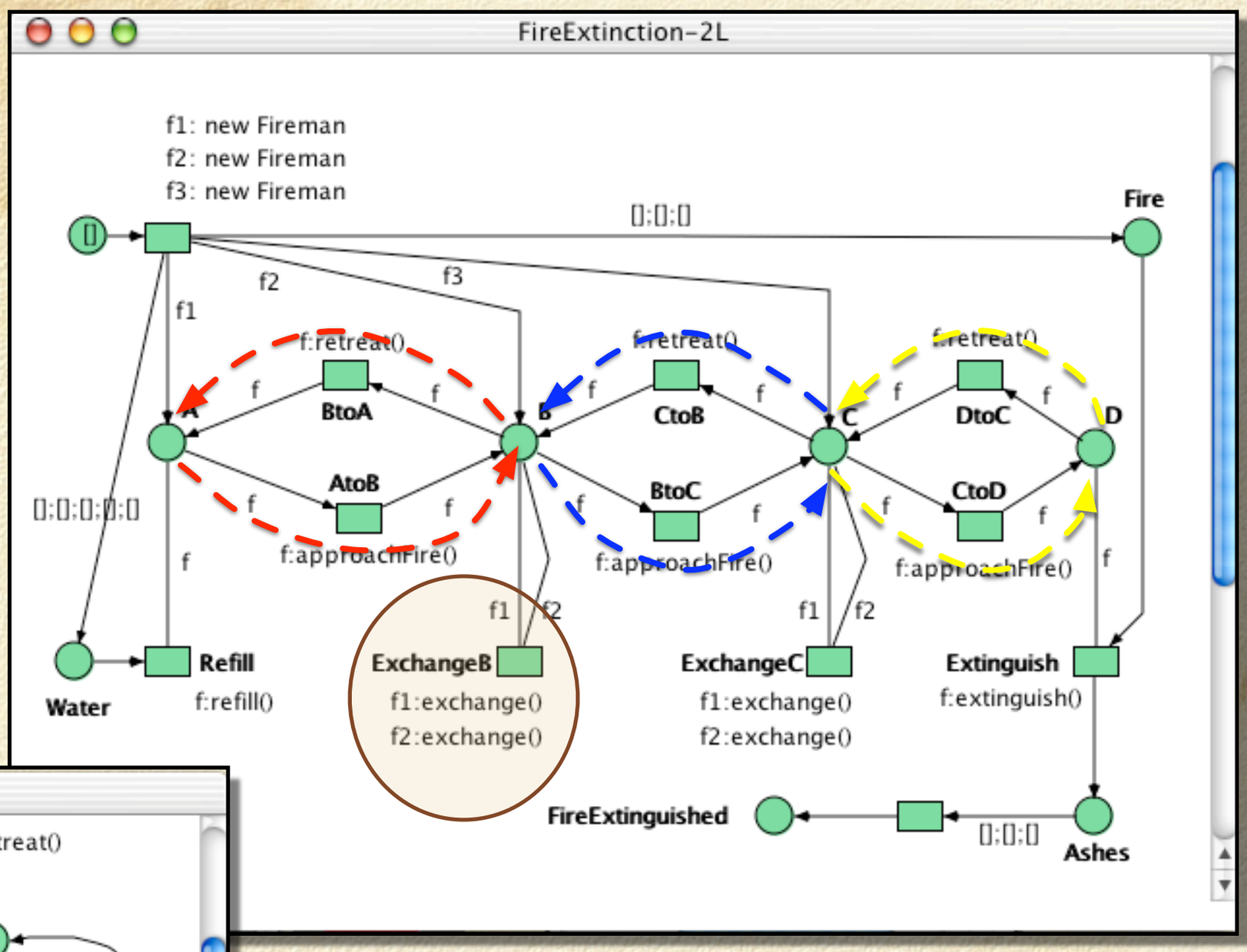

Fireman[4]
:retreat()
:refill()
:extinguish()
:approachFire()

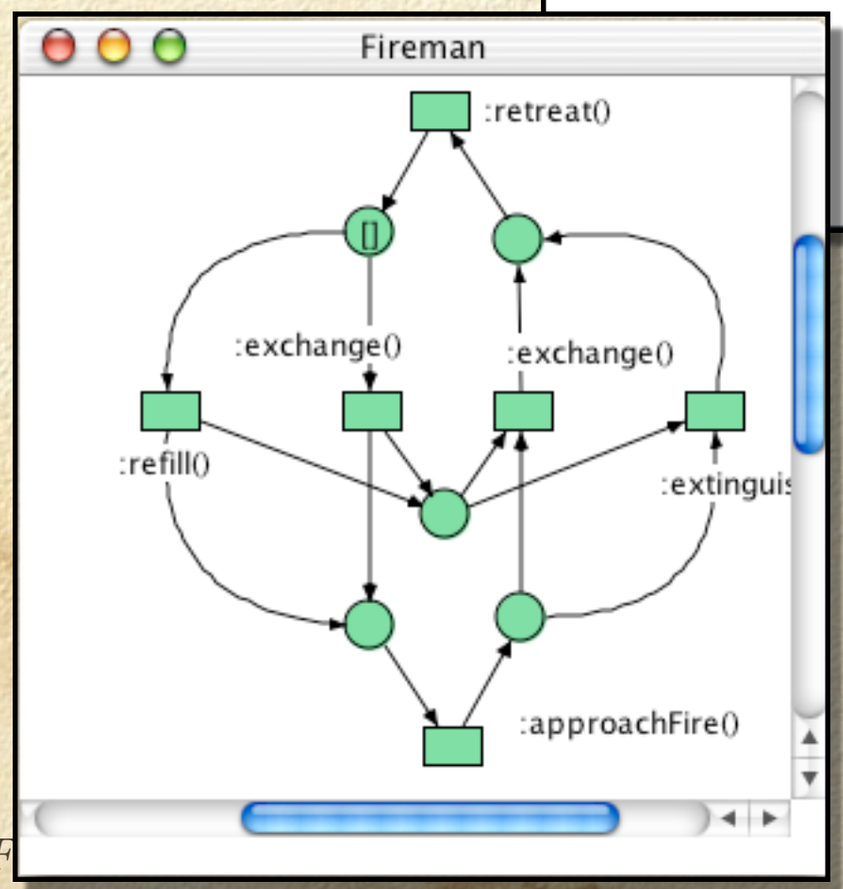
movie

movie

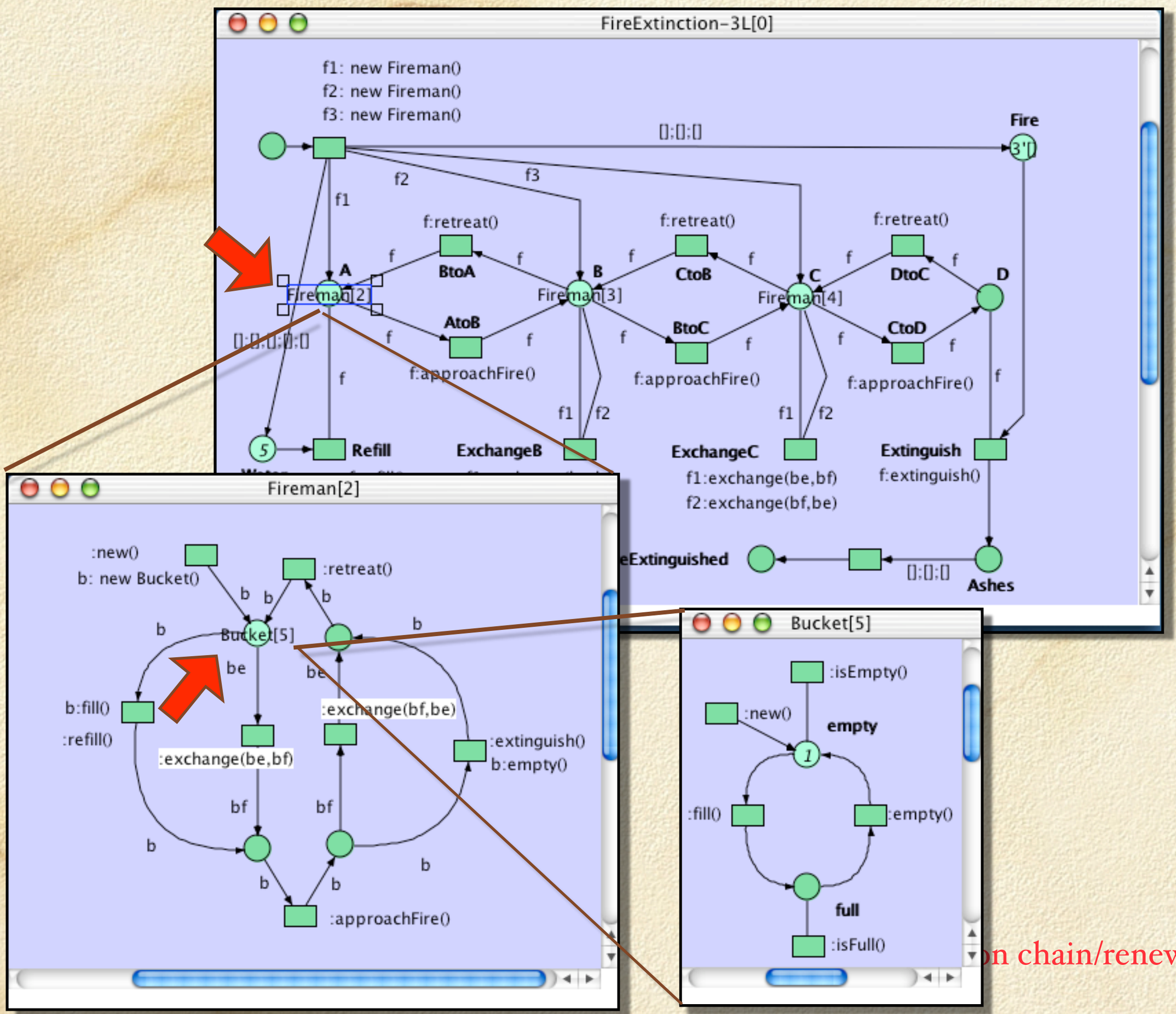


parallel fire extinction/renew





fire extinction **chain**/renew


FireExtinction-3L[0]
f1: new Fireman()
f2: new Fireman()
f3: new Fireman()
[];[];[]
Fire
3'[]
f1
f2
f3
f:retreat()
f:retreat()
f:retreat()
A
B
C
D
BtoA
CtoB
DtoC
AtoB
BtoC
CtoD
Fireman[2]
Fireman[3]
Fireman[4]
[];[];[];[];[]
f:approachFire()
f:approachFire()
f:approachFire()
5
Refill
ExchangeB
ExchangeC
Extinguish
f:extinguish()
f1:exchange(be,bf)
f2:exchange(bf,be)
eExtinguished
[];[];[]
Ashes
Fireman[2]
:new()
b: new Bucket()
:retreat()
Bucket[5]
b:fill()
:refill()
:exchange(be,bf)
:exchange(bf,be)
:extinguish()
b:empty()
:approachFire()
Bucket[5]
:isEmpty()
:new()
empty
1
:fill()
:empty()
full
:isFull()

Garbage Can

"socionics" =

informatics/multi agent systems

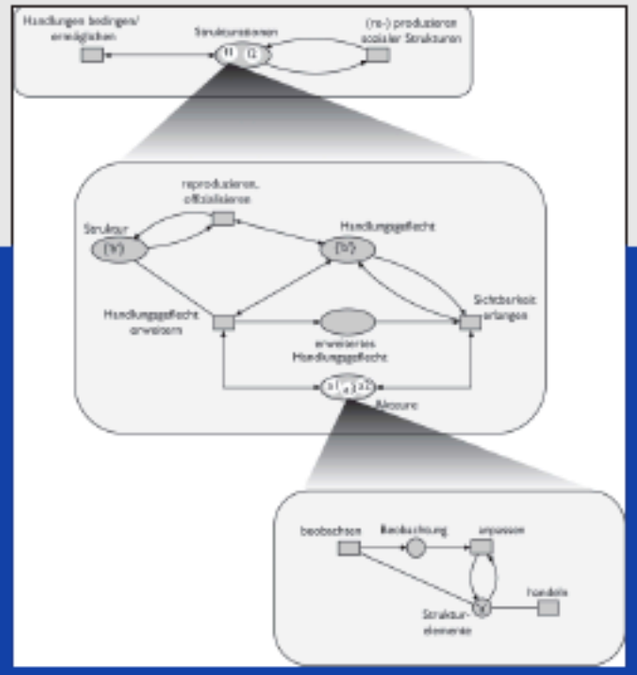
+

sociology

Rolf v. Lüde, Daniel Moldt, Rüdiger Valk

Sozionik – Modellierung soziologischer Theorie

Unter Mitarbeit von Michael Köhler, Roman Langer, Heiko Rölke, Daniela Spresny



WIRTSCHAFT – ARBEIT – TECHNIK

PERSPEKTIVEN GESELLSCHAFTLICHEN WANDELS
HG. ROLF v. LÜDE

LIT

did appear this month

M.D. Cohen J.G. March and J.P. Olsen.

A garbage can model of organizational choice.

Administrative Science Quarterly 17:1--25, 1972.

Consider organized anarchies. These are organizations - or decision situations - characterized by three general properties.

The first is **problematic preferences**. In the organization it is difficult to impute a set of preferences to the decision situation that satisfies the standard consistency requirements for a theory of choice.

The organization operates on the basis of a variety of inconsistent and ill defined preferences. It can be described better by a loose collection of ideas than as a coherent structure; it discovers preferences through action more than it acts on the basis of preferences,

The second property is **unclear technology**. Although the organization manages to survive and even produce, its own processes are not understood by its members. It operates on the basis of simple trial-and-error procedures, the residue of learning from accidents of past experience, and pragmatic inventions of necessity.

The third property is **fluid participation**. Participants vary in the amount of time and effort they devote to different domains; involvement varies from one time to another. As a result, the boundaries of the organization are uncertain and changing; the audiences and decision makers for any particular kind of choice change capriciously.

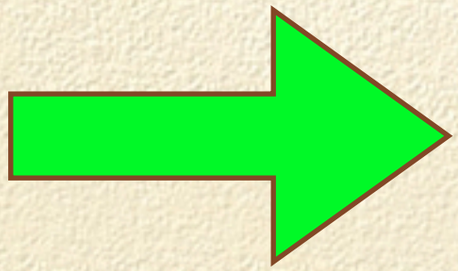
Problems. Problems are the concern of people inside and outside the organization ....

Solutions. A solution is somebody's product. .... Despite the dictum that you cannot find the answer until you have formulated the question well, you often do not know what the question is in organizational problem solving until you know the answer.

Participants. Participants come and go. ..... Substantial variation in participation stems from other demands on the participants' time (rather than from features of the decision under study) .

Choice opportunities. These are occasions when an organization is expected to produce behavior that can be called a decision. Opportunities arise regularly and any organization has ways of declaring an occasion for choice. Contracts must be signed; people hired, promoted, or fired; money spent; and responsibilities allocated.

A stream of choices
A stream of problems
A rate of flow of solutions
A stream of energy from participants



garbage can

M.Masuch and P. LaPotin.

Beyond Garbage Cans:

An AI Model of Organizational Choice.

Administrtive Science Quarterly 36: 38--67, 1989.

To get an approximate impression of the model, reconsider the finale of the James Bond movie, "A View to a Kill." Agent 007 balances on the main cable of the Golden Gate Bridge, a woman in distress clinging to his arm, a blimp approaching for rescue.

In terms of the garbage can model,
the blimp is a solution,
007 a choice opportunity, and
the woman a problem.

In the picture's happy ending, the hero is finally picked up, together with the woman, and a solution by resolution takes place; the problem is solved

As women, as well as blimps, make their choices simultaneously, but independently of each other, a light hero, on the verge of rescue, may suddenly find himself overburdened. Heavy heroes, in turn, may become rescuable all of a sudden as their women desert them. This mechanism, called "**fluid participation**", creates the possibility of nonsensical solutions or nonsolutions.

Women may switch heroes too swiftly and end up with an overburdened hero each time; then, problems are not solved. Or heroes are rescued just as all women have left; then, a "**decision by flight**" is made.

Finally, heroes may be salvaged upon arrival at the scene before any woman in distress has a chance to grab their arm; then "**decisions by oversight**" are said to be made.

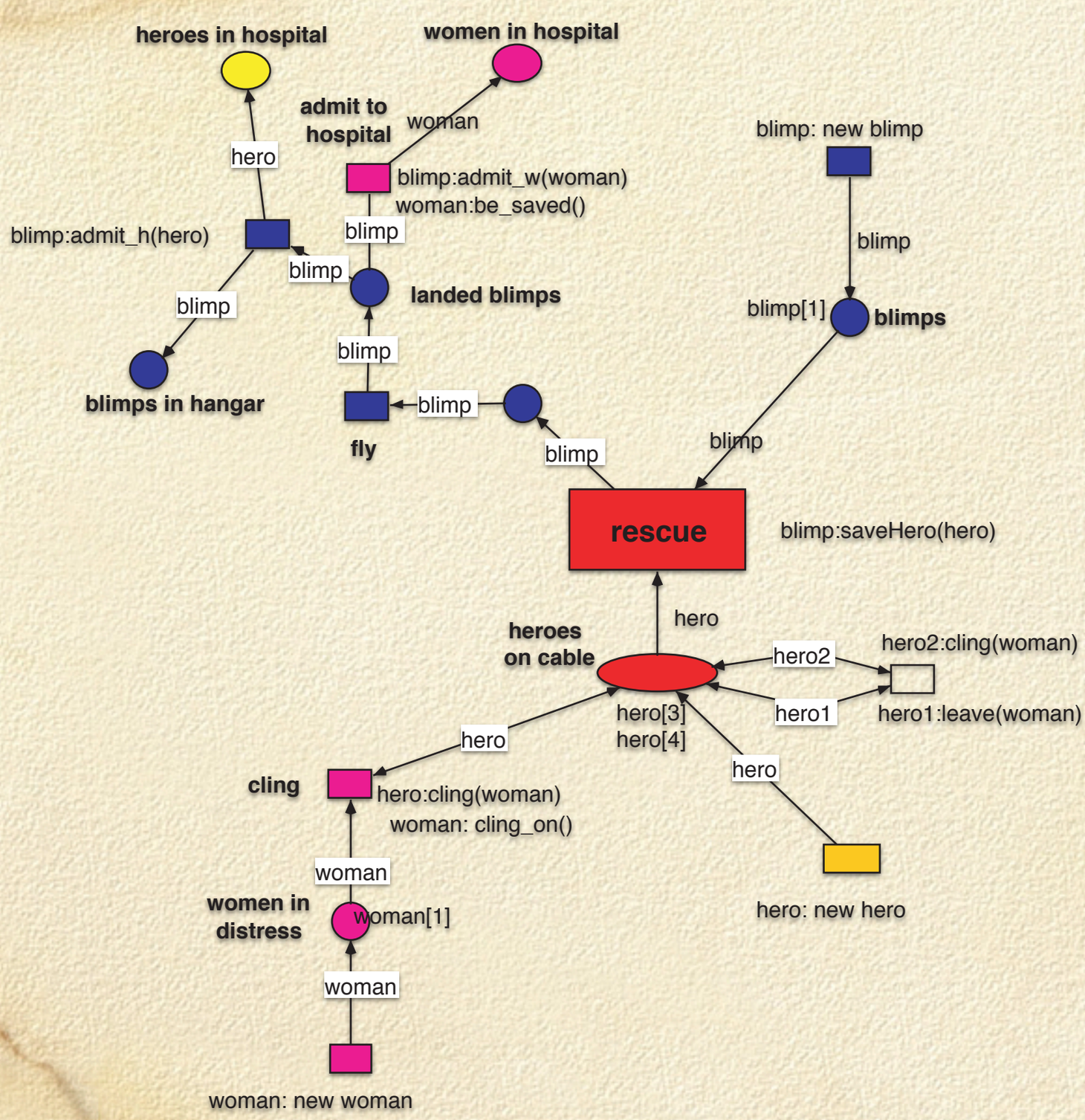
Nevertheless, decisions by resolution do occur.

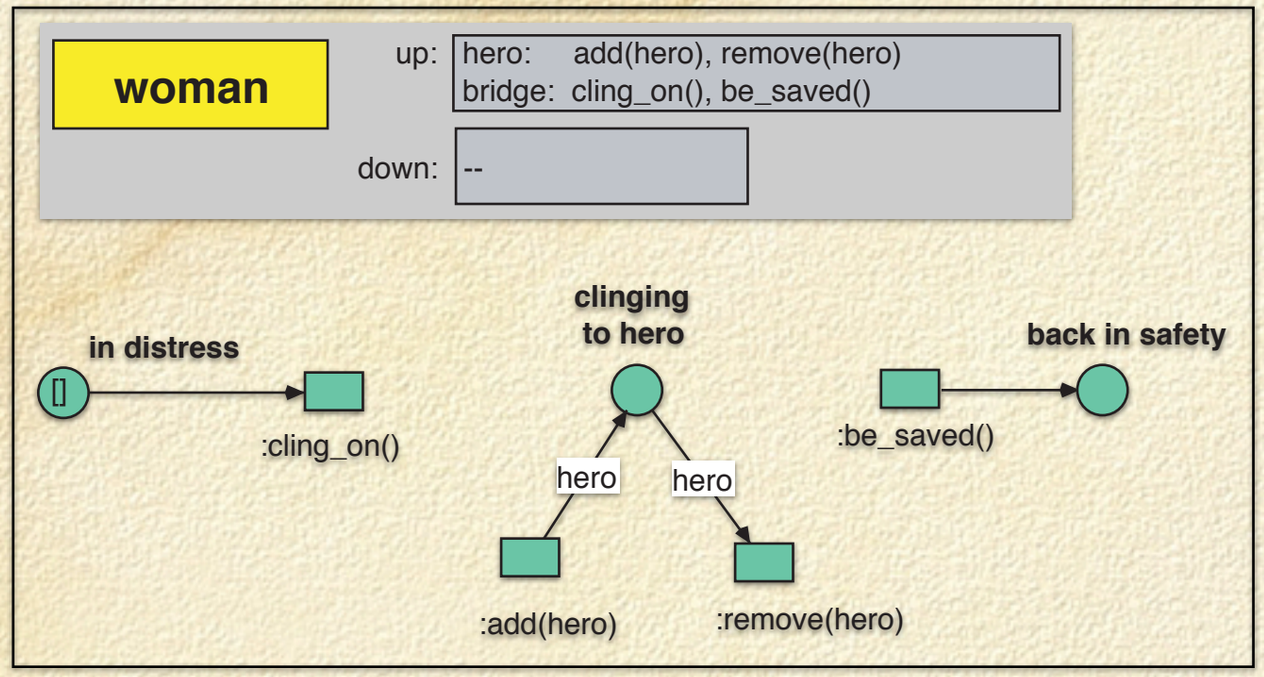
bridge

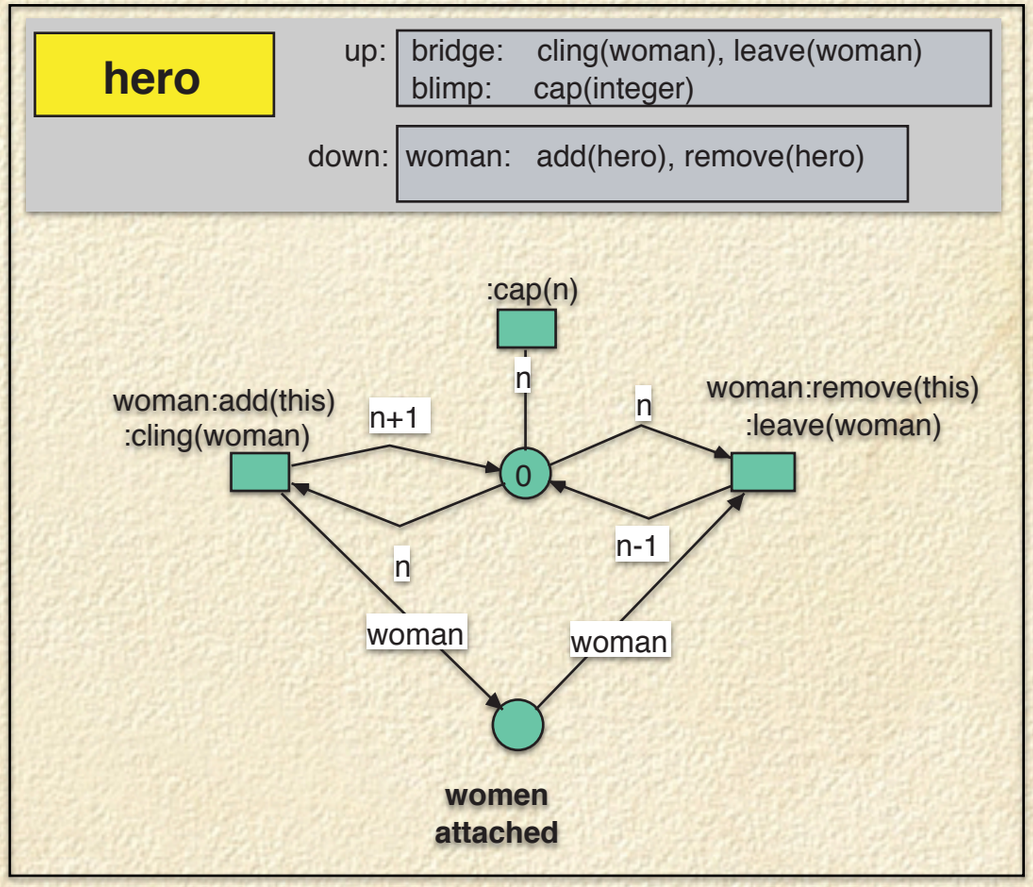
up: --

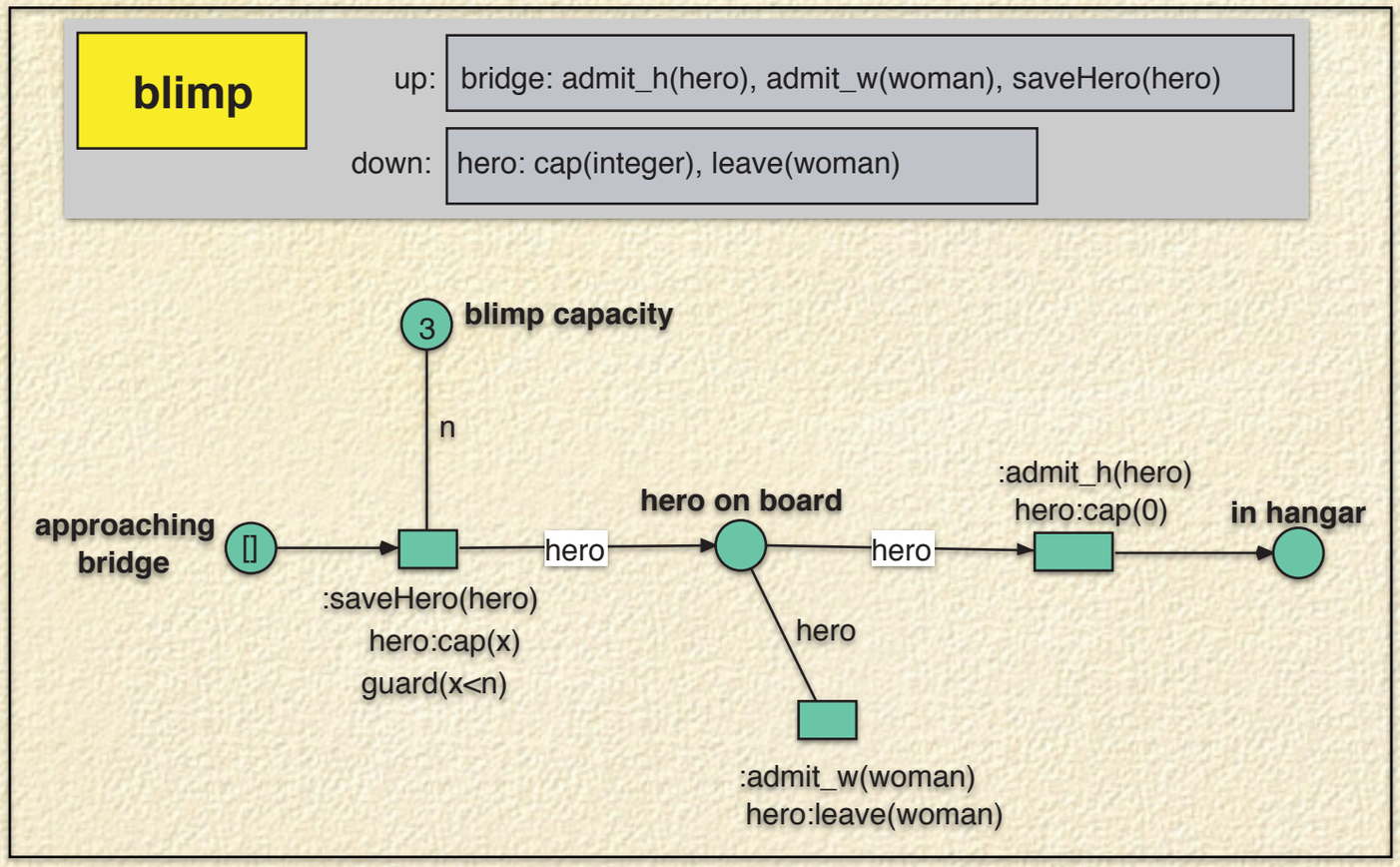
down:

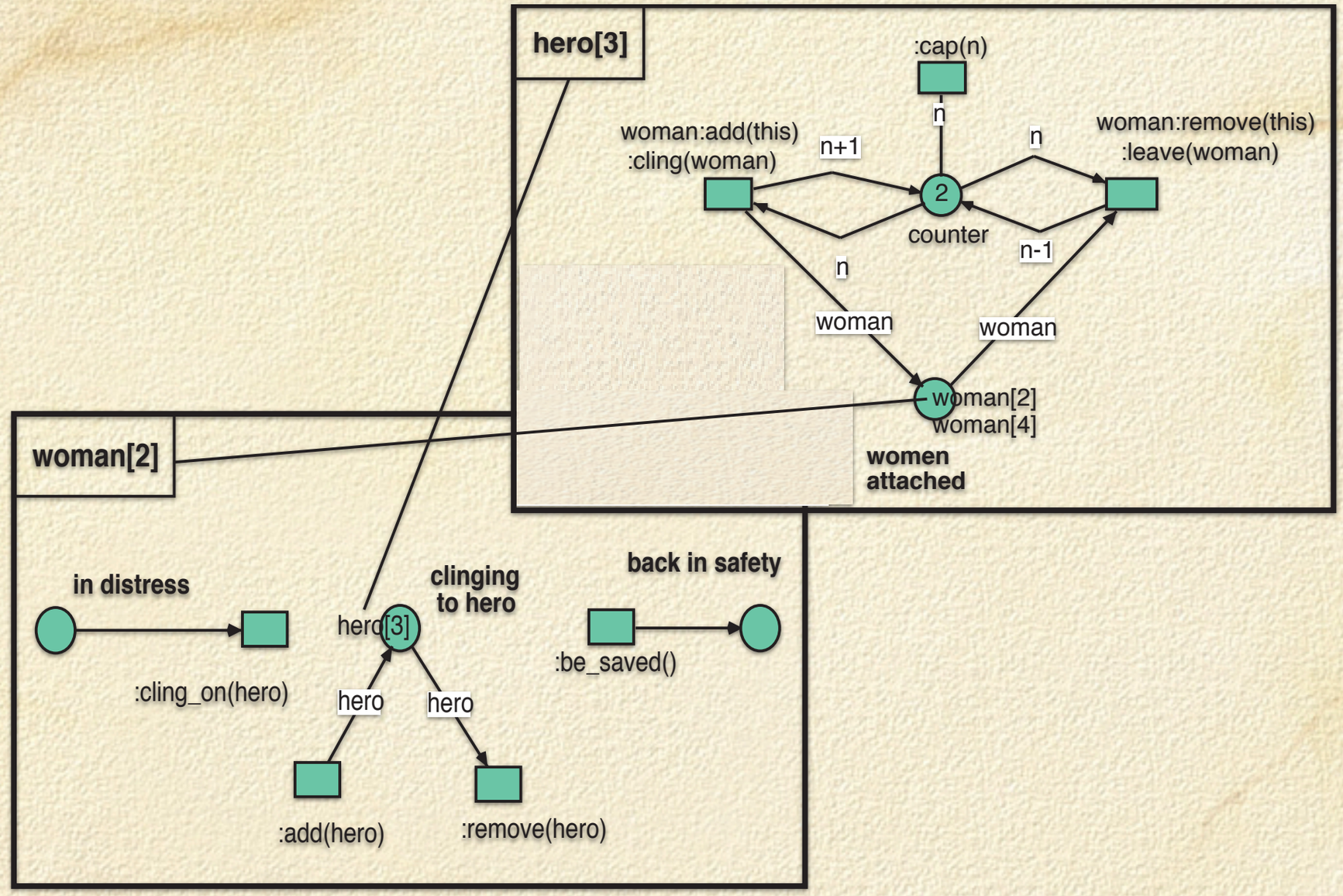
blimp:	admit_h(hero), admit_w(woman),saveHero(hero)
hero:	leave(woman), cling(woman)
woman:	cling_on(), be_saved()

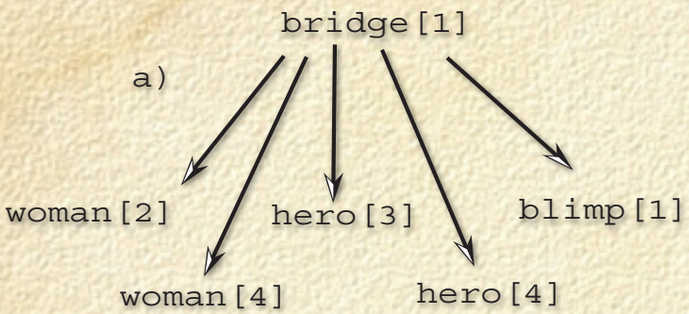


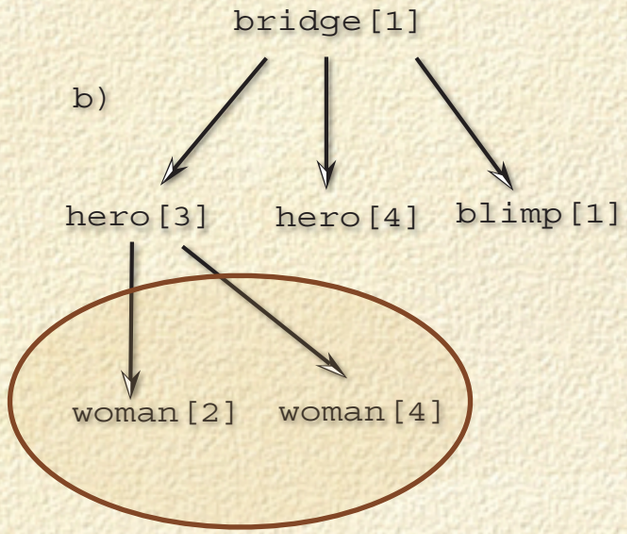

woman
up:
hero: add(hero), remove(hero)
bridge: cling_on(), be_saved()
down:
--
in distress
[]
:cling_on()
clinging
to hero
hero
hero
:add(hero)
:remove(hero)
back in safety
:be_saved()

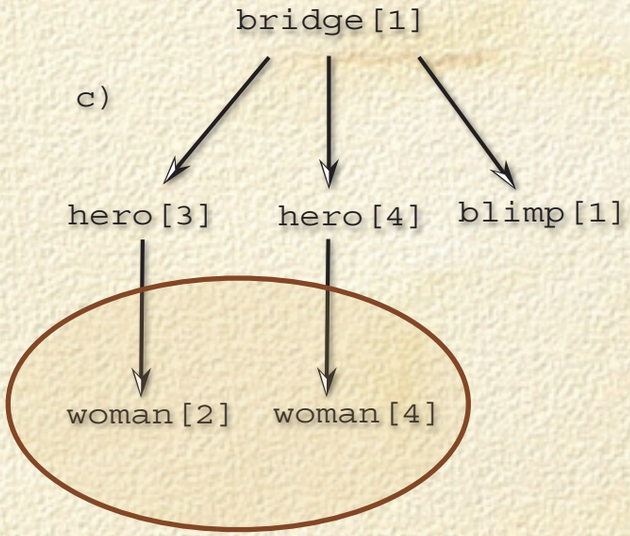

hero
up:
bridge: cling(woman), leave(woman)
blimp: cap(integer)
down:
woman: add(hero), remove(hero)
:cap(n)
n
woman:add(this)
:cling(woman)
n+1
0
n
woman:remove(this)
:leave(woman)
n-1
n
woman
woman
women
attached

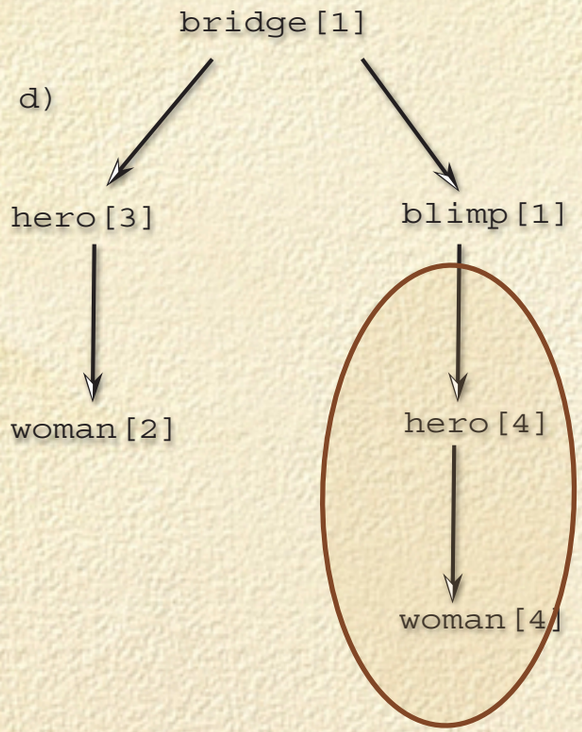

blimp
up:
bridge: admit_h(hero), admit_w(woman), saveHero(hero)
down:
hero: cap(integer), leave(woman)
3
blimp capacity
n
approaching bridge
[]
:saveHero(hero)
hero:cap(x)
guard(x<n)
hero
hero on board
hero
:admit_h(hero)
hero:cap(0)
in hangar
hero
:admit_w(woman)
hero:leave(woman)

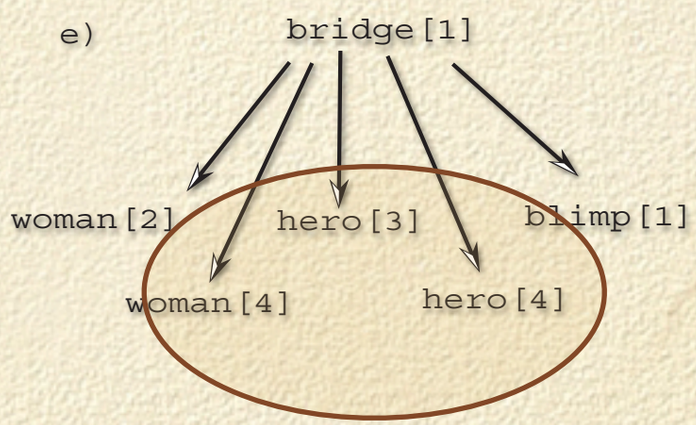

hero[3]
:cap(n)
n
woman:add(this)
:cling(woman)
n+1
n
woman:remove(this)
:leave(woman)
2
counter
n-1
n
woman
woman
woman[2]
woman[4]
women
attached
woman[2]
in distress
clinging
to hero
back in safety
hero[3]
:be_saved()
:cling_on(hero)
hero
hero
:add(hero)
:remove(hero)


a)
bridge[1]
woman[2]
woman[4]
hero[3]
hero[4]
blimp[1]


b)
bridge[1]
hero[3]
hero[4]
blimp[1]
woman[2]
woman[4]

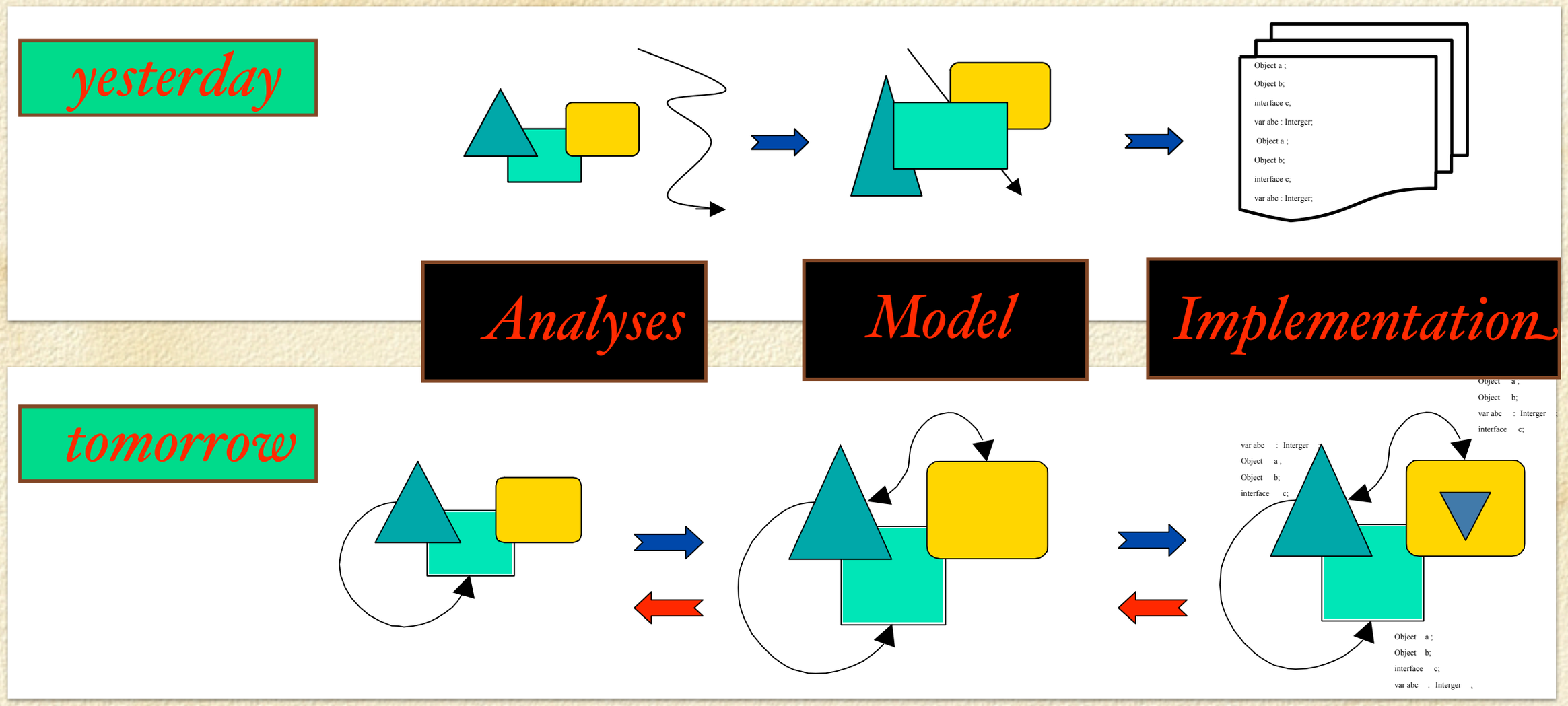

c)
bridge[1]
hero[3]
hero[4]
blimp[1]
woman[2]
woman[4]


d)
bridge[1]
hero[3]
woman[2]
blimp[1]
hero[4]
woman[4]

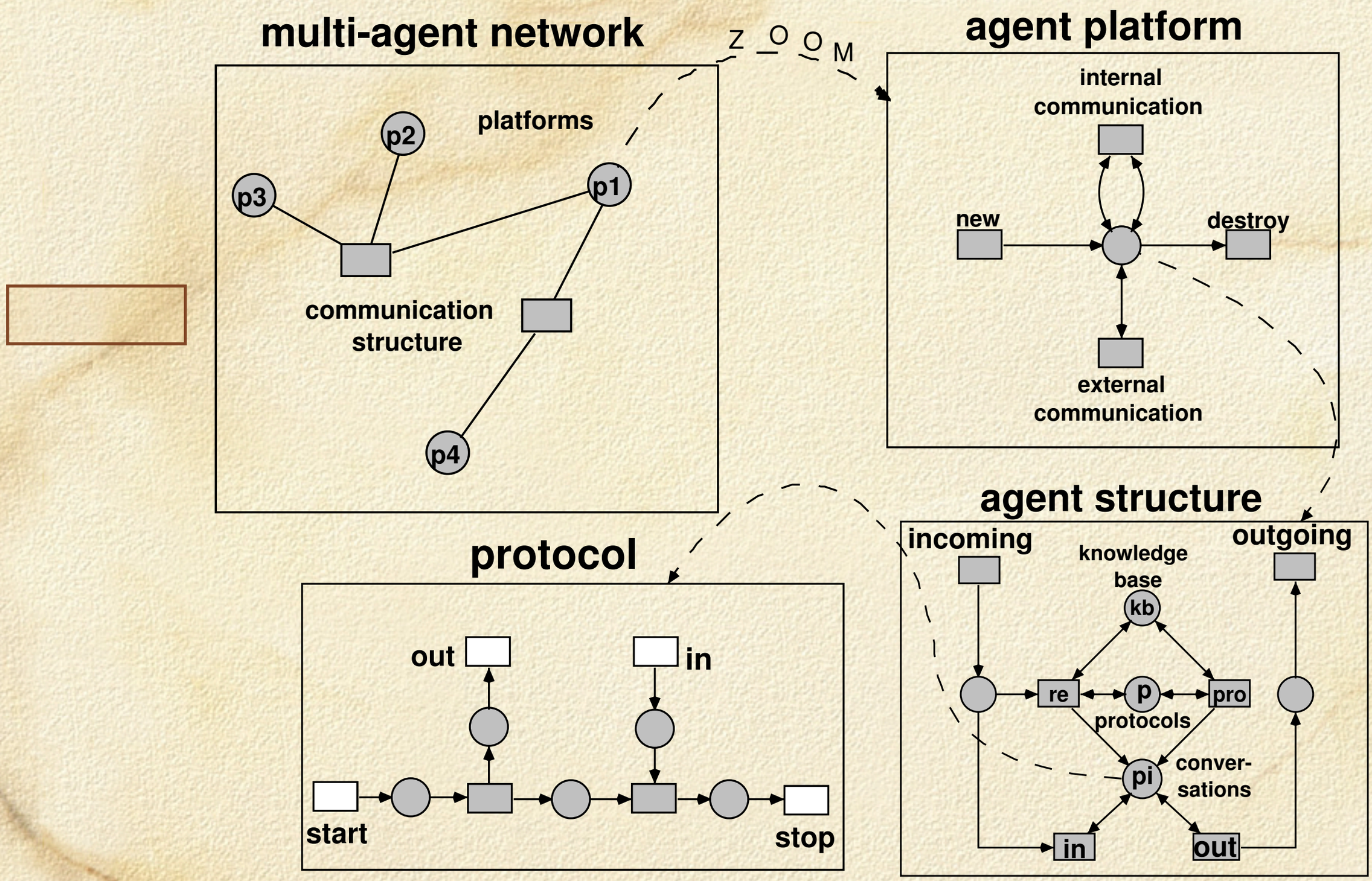

e)
bridge[1]
woman[2]
woman[4]
hero[3]
hero[4]
blimp[1]

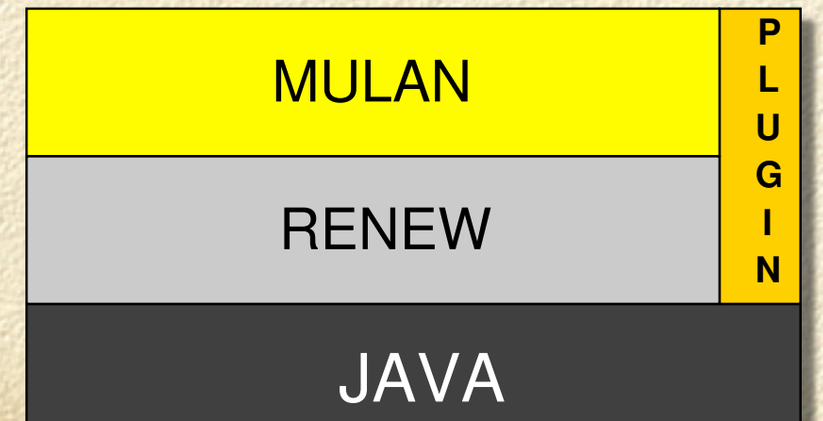
Agents & Components

software engineering approach



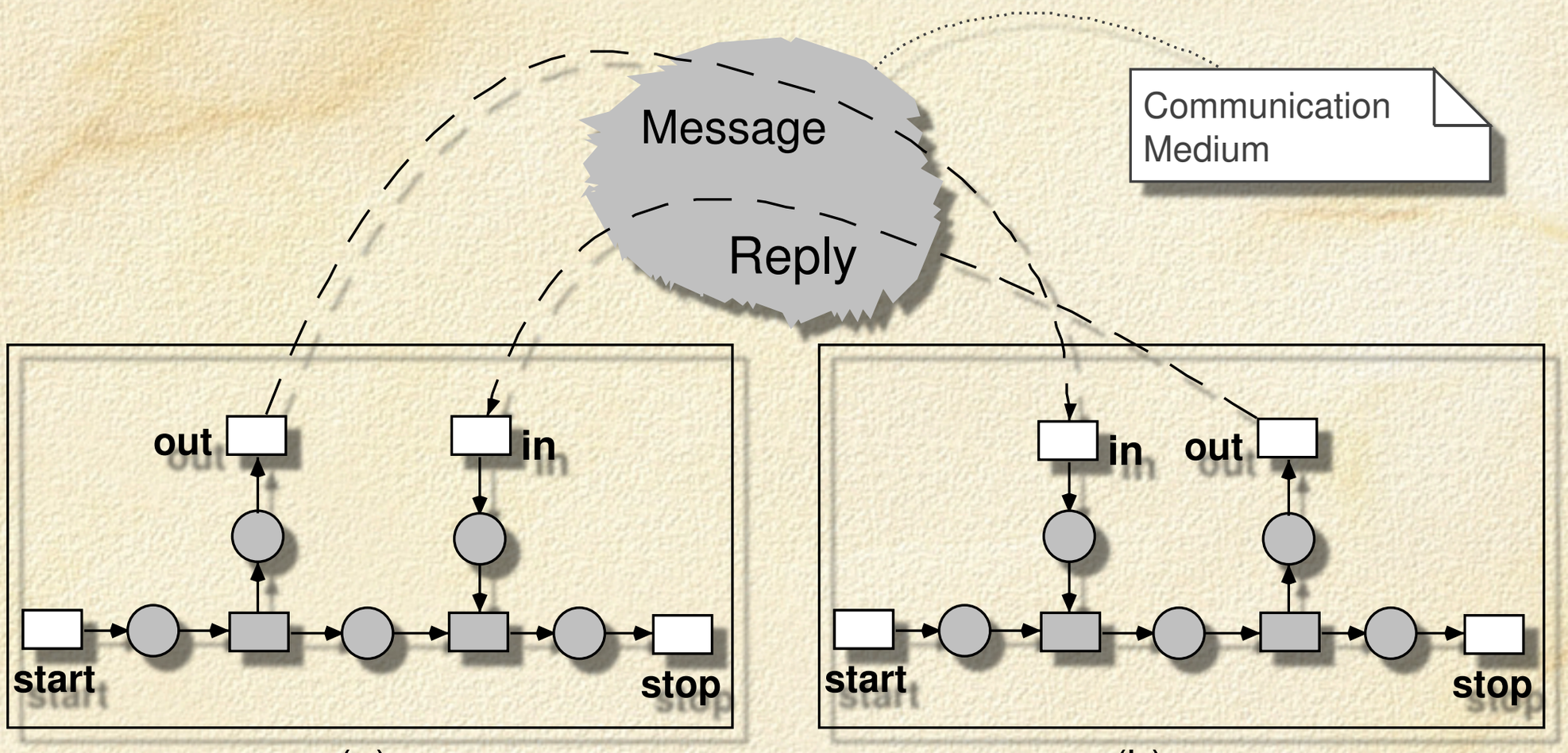
round-trip engineering

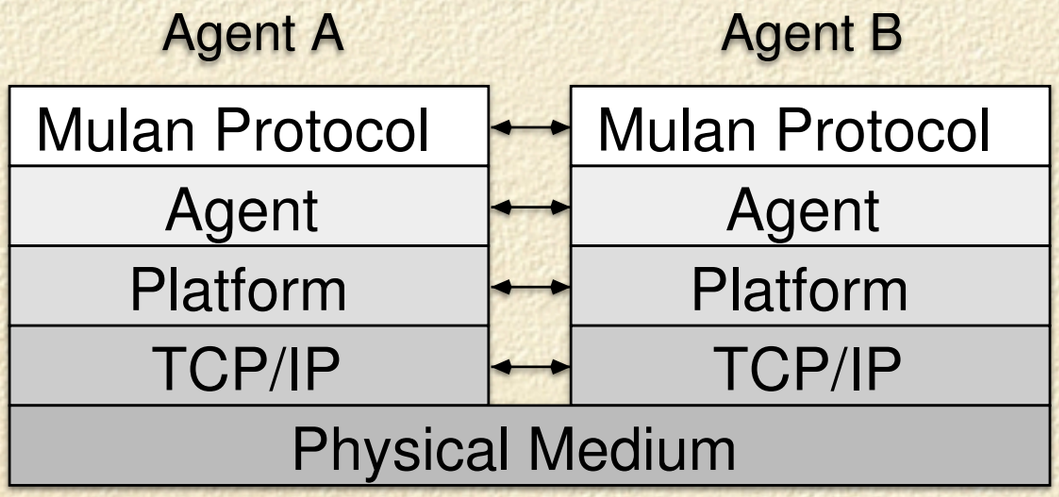


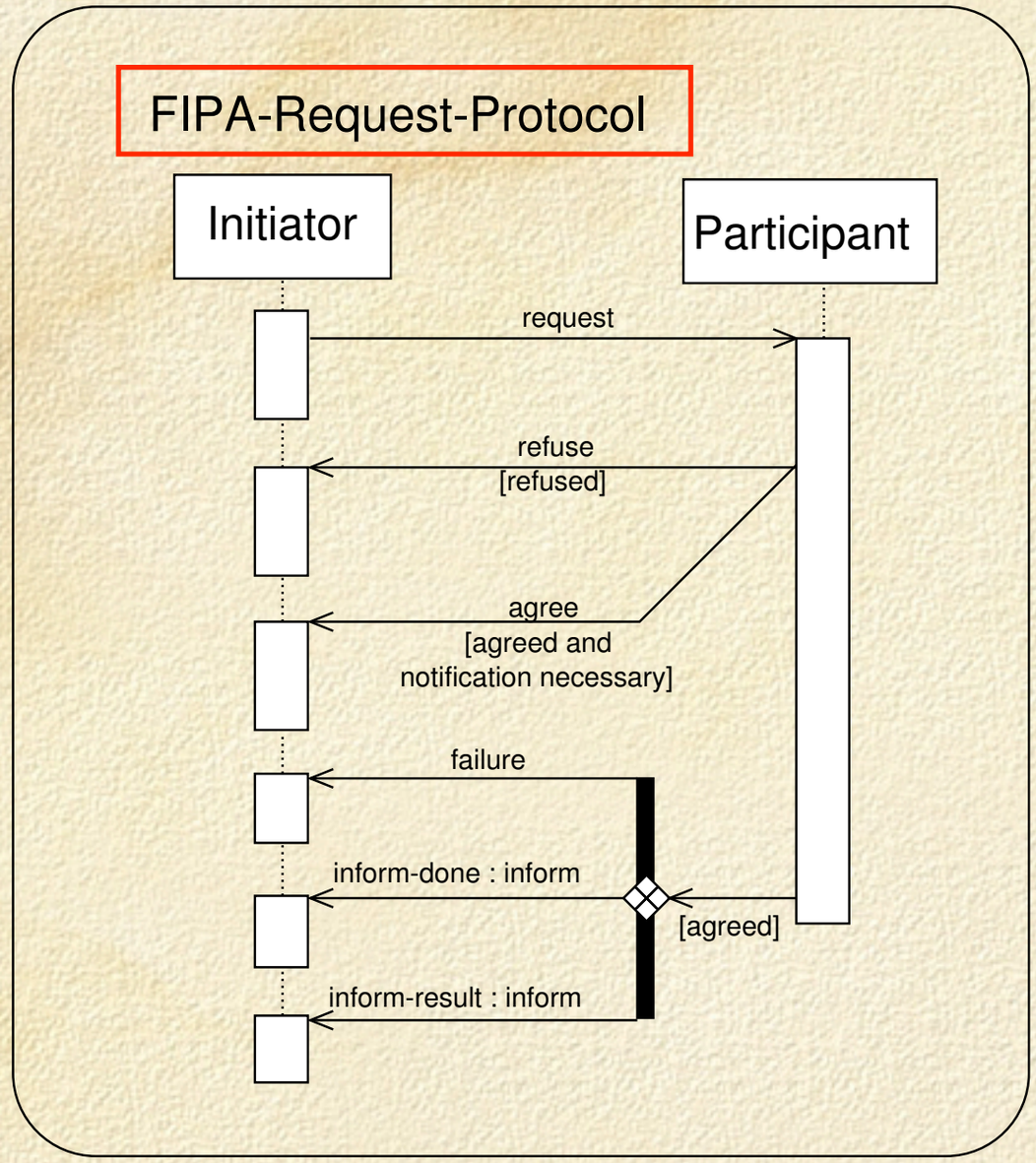


MULtiAgentenNetze

REferenceNEtWorkshop


Message
Reply
Communication Medium
out
in
start
stop
(a)
in
out
start
stop
(b)


Agent A
Agent B
Mulan Protocol
Mulan Protocol
Agent
Agent
Platform
Platform
TCP/IP
TCP/IP
Physical Medium



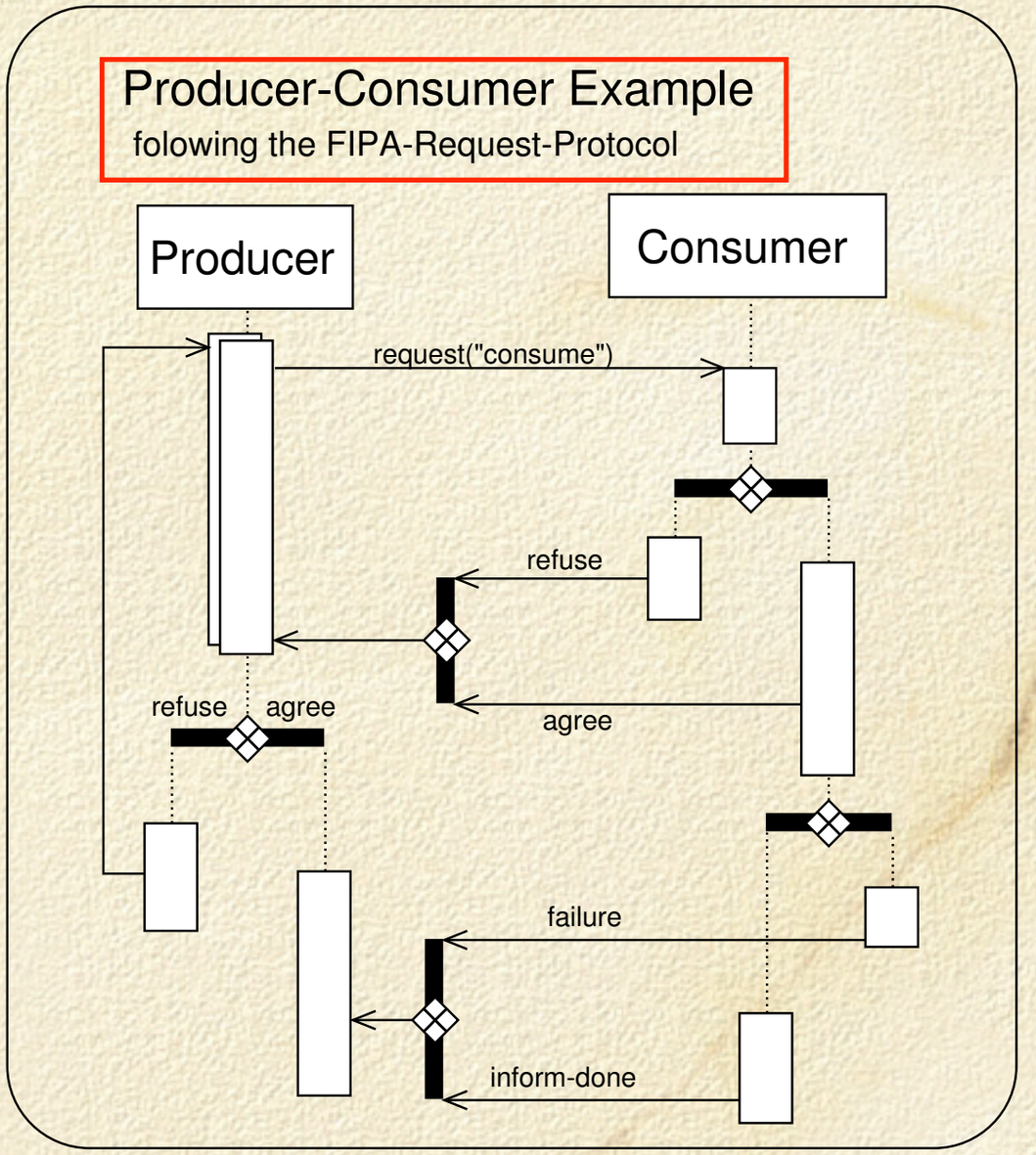


Fig. 11: Interaction protocol diagram of the FIPA Request and a FIPA Request-compliant *producer-consumer* example.

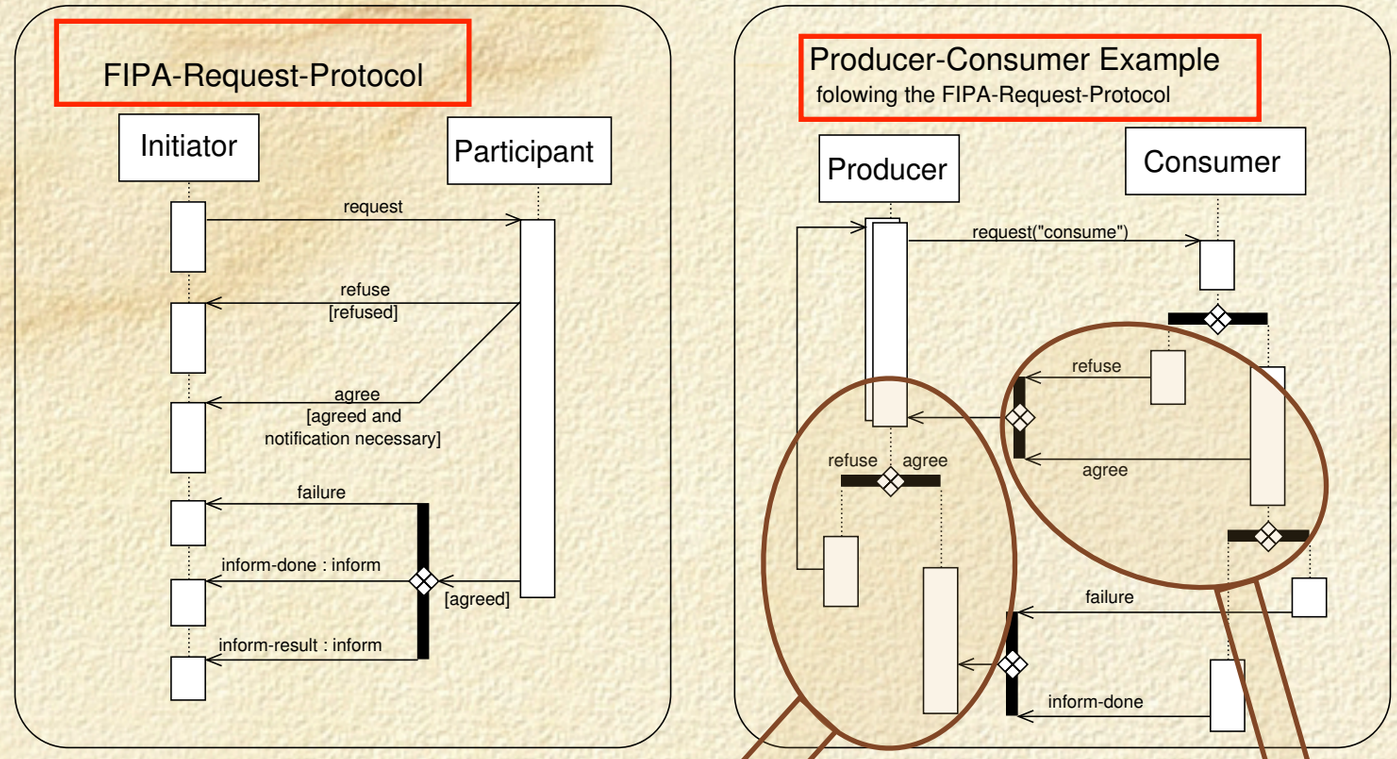


Fig. 11: Interaction protocol diagram of the FIPA Request and a FIPA Request-compliant *producer-consumer* example.

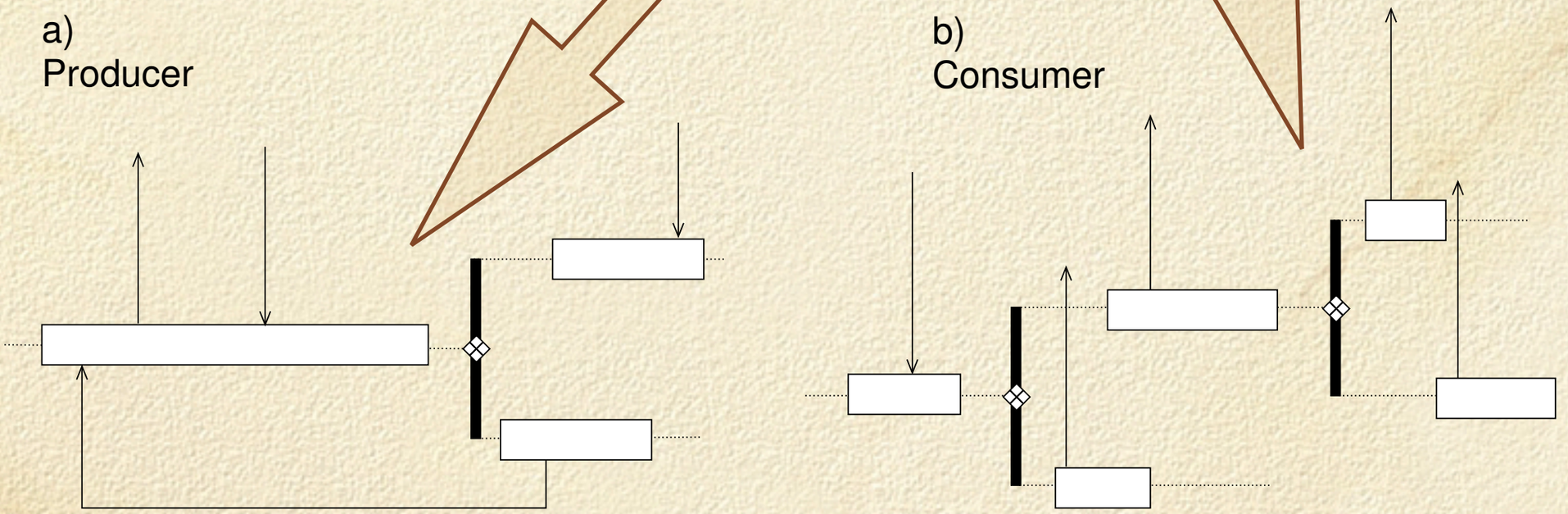


Fig. 12: Two parts of the example diagram; one for each agent.

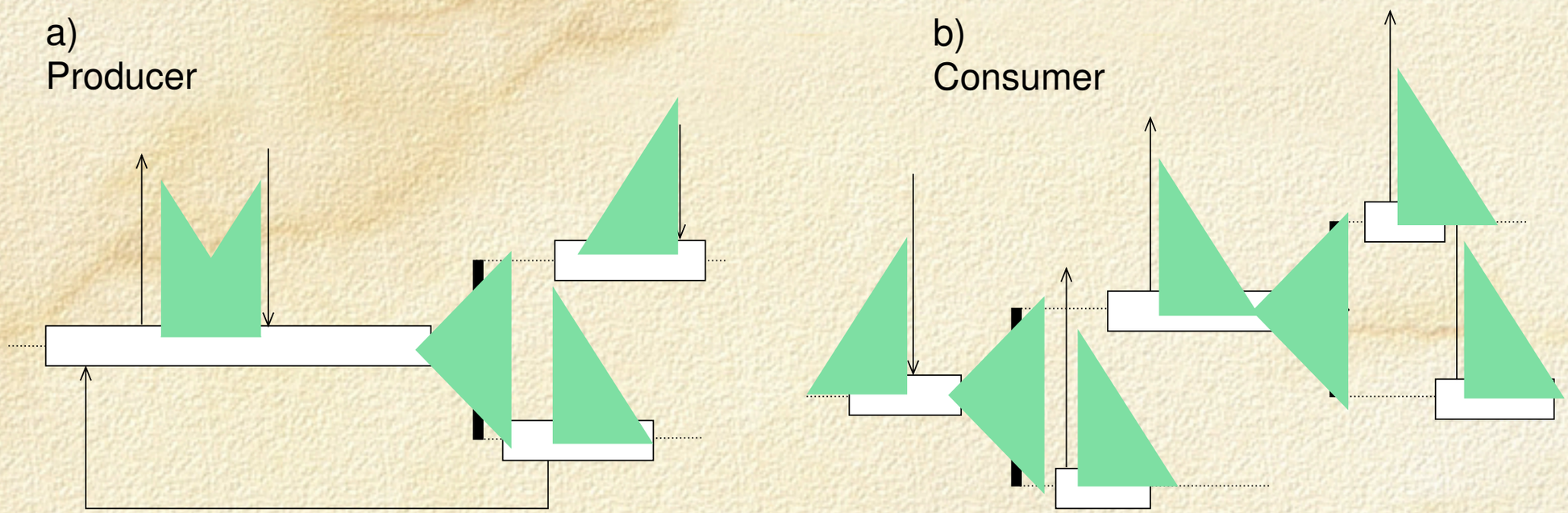


Fig. 13: Both parts augmented by the symbols for the net components.

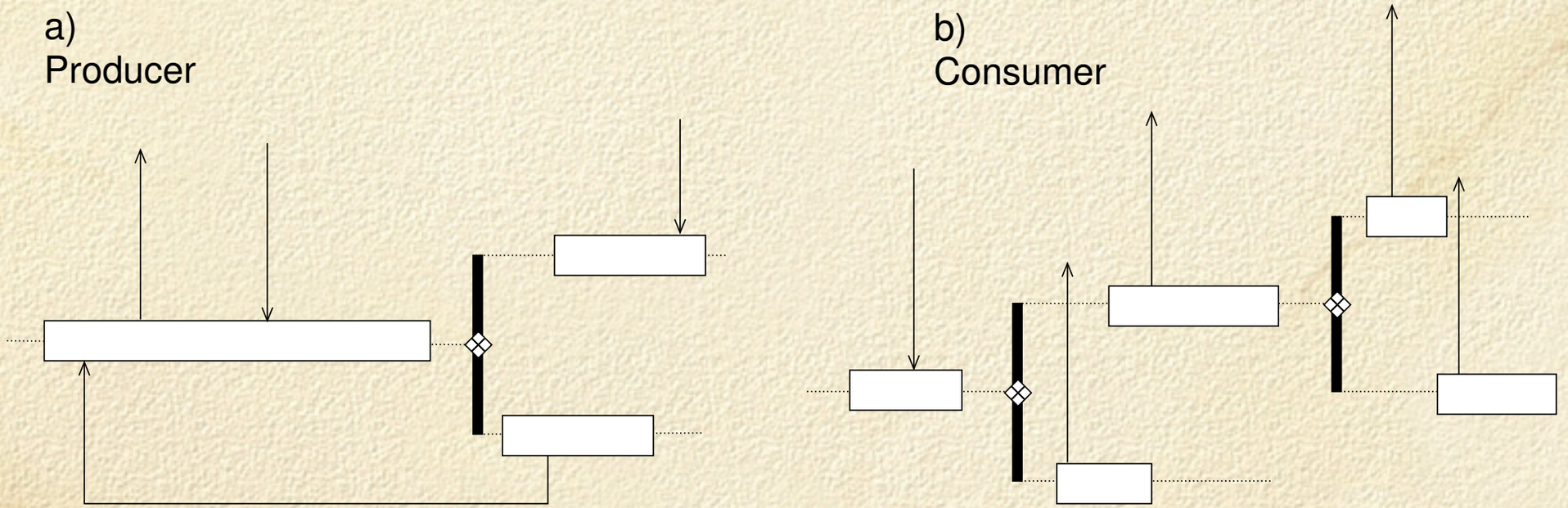
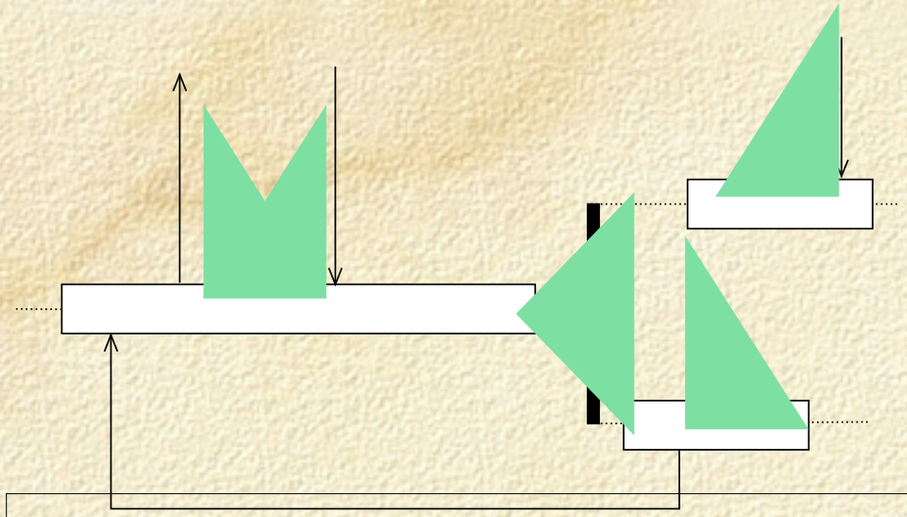
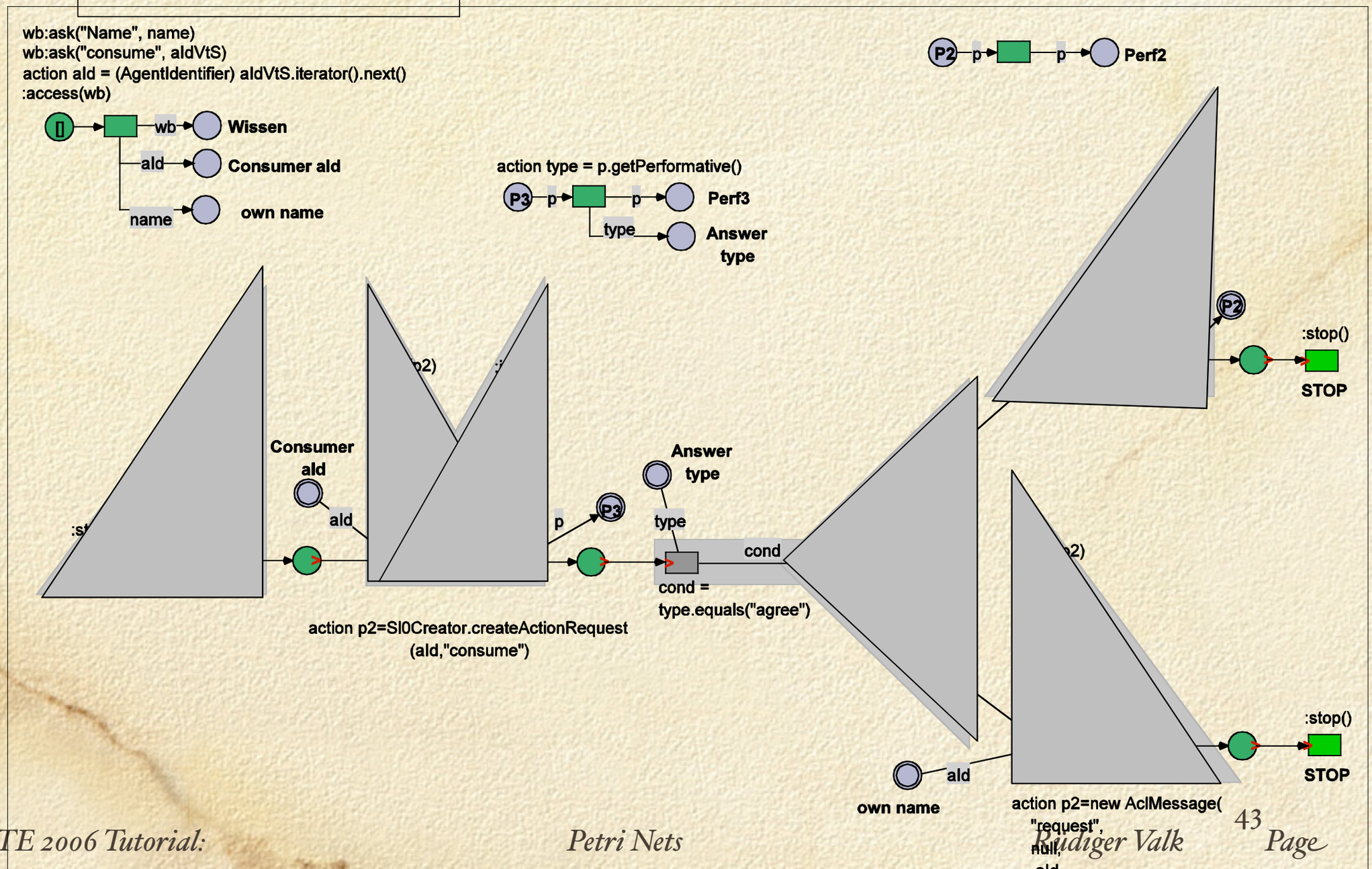


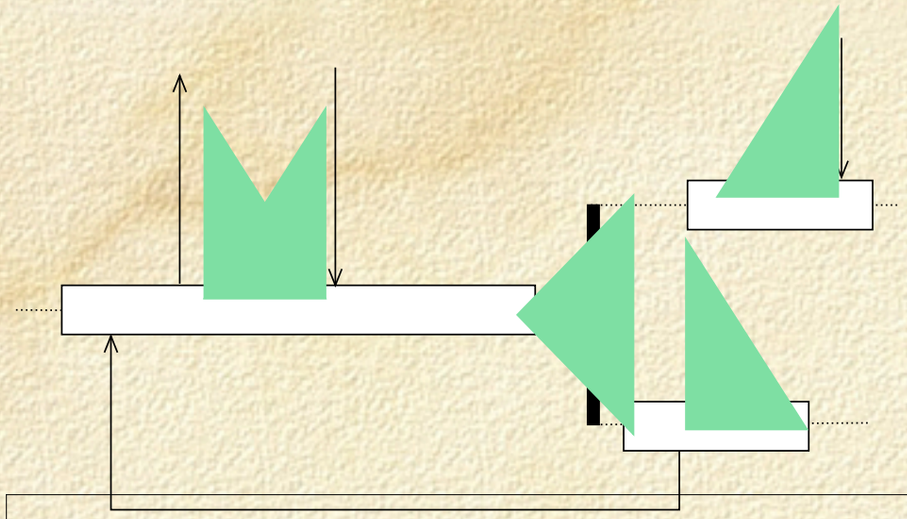
Fig. 12: Two parts of the example diagram; one for each agent.

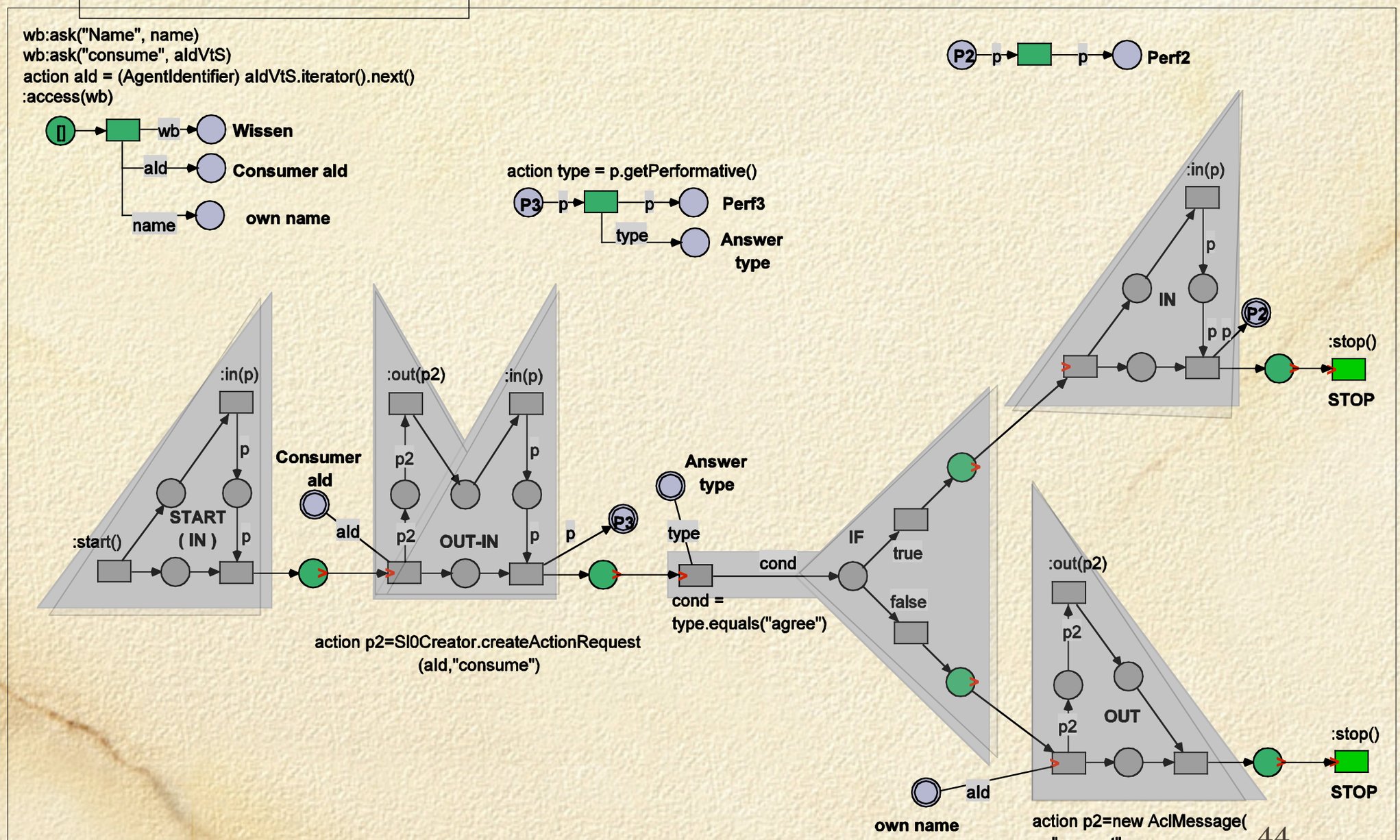
a)
Producer




wb:ask("Name", name)
wb:ask("consume", aIdVtS)
action aId = (AgentIdentifier) aIdVtS.iterator().next()
:access(wb)
[]
wb
Wissen
aId
Consumer aId
name
own name
action type = p.getPerformative()
P3
p
p
Perf3
type
Answer type
P2
p
p
Perf2
:stop()
STOP
Consumer aId
aId
p
P3
Answer type
type
cond
cond = type.equals("agree")
action p2=SI0Creator.createActionRequest (aId,"consume")
own name
aId
action p2=new AclMessage("request",
:stop()
STOP

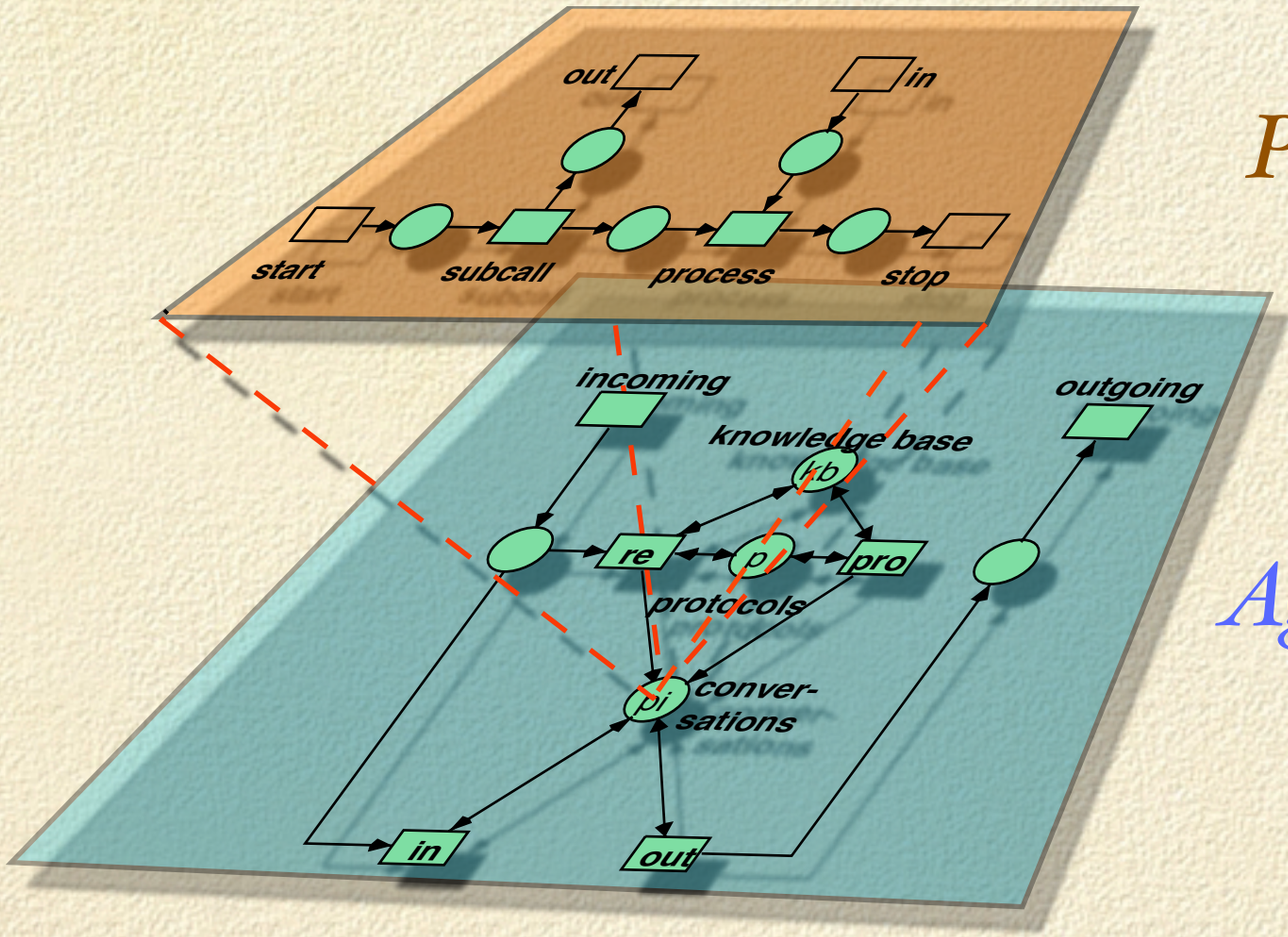
a)
Producer




wb:ask("Name", name)
wb:ask("consume", aIdVtS)
action aId = (AgentIdentifier) aIdVtS.iterator().next()
:access(wb)
[]
wb
Wissen
aId
Consumer aId
name
own name
action type = p.getPerformative()
P3
p
p
Perf3
type
Answer type
P2
p
p
Perf2
:in(p)
:out(p2)
:in(p)
:start()
START
(IN)
p
p
Consumer aId
aId
p2
p2
OUT-IN
p
p
P3
action p2=Sl0Creator.createActionRequest
(aId,"consume")
Answer type
type
cond
cond =
type.equals("agree")
IF
true
false
:in(p)
IN
p
p p
P2
:stop()
STOP
:out(p2)
p2
p2
OUT
:stop()
STOP
aId
own name
action p2=new AclMessage(
"request",
null,


The Settlers of Catan
A Game of Discovery, Settlement and Trade
LES COLONS DE CATANE
DIE SIEDLER VON CATAN


Jetzt bin ich dran - und ich habe eine "11" gewürfelt.
[Weiter]
Siegfried
2
Marlene
2
Vicky
2



Protokol

Agent Structure

Agent “Siegfried”

Agent “Insel”

Agent “Marlene”

Agent “Vicky”

Agent “Steuereinheit”

Agent “Bank”

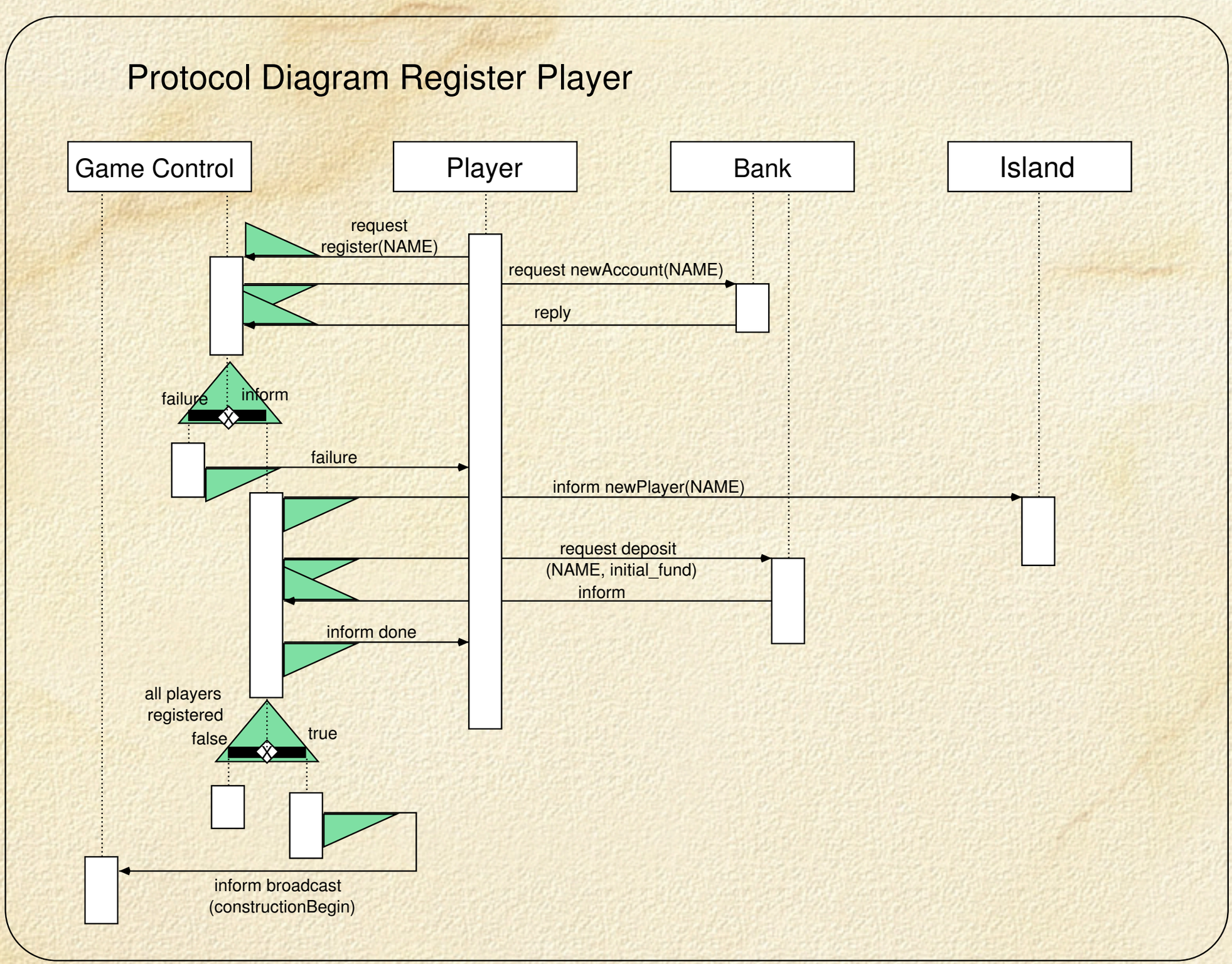


Fig. 17: Mulan conform-structured agent interaction protocol for the registration of a player.

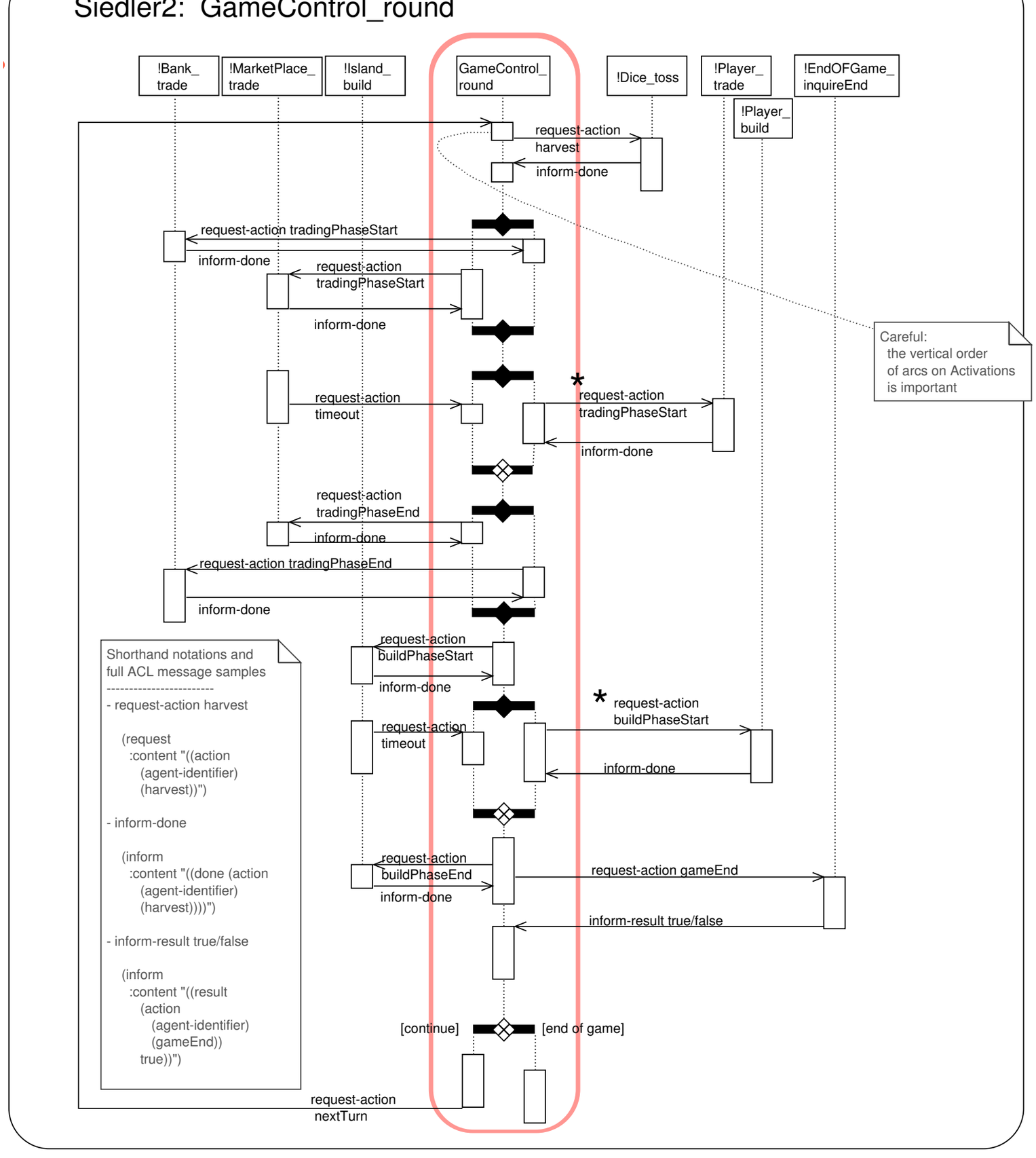


Figure 8.14: Agent interaction protocol diagram for the round of the *settler2* game.

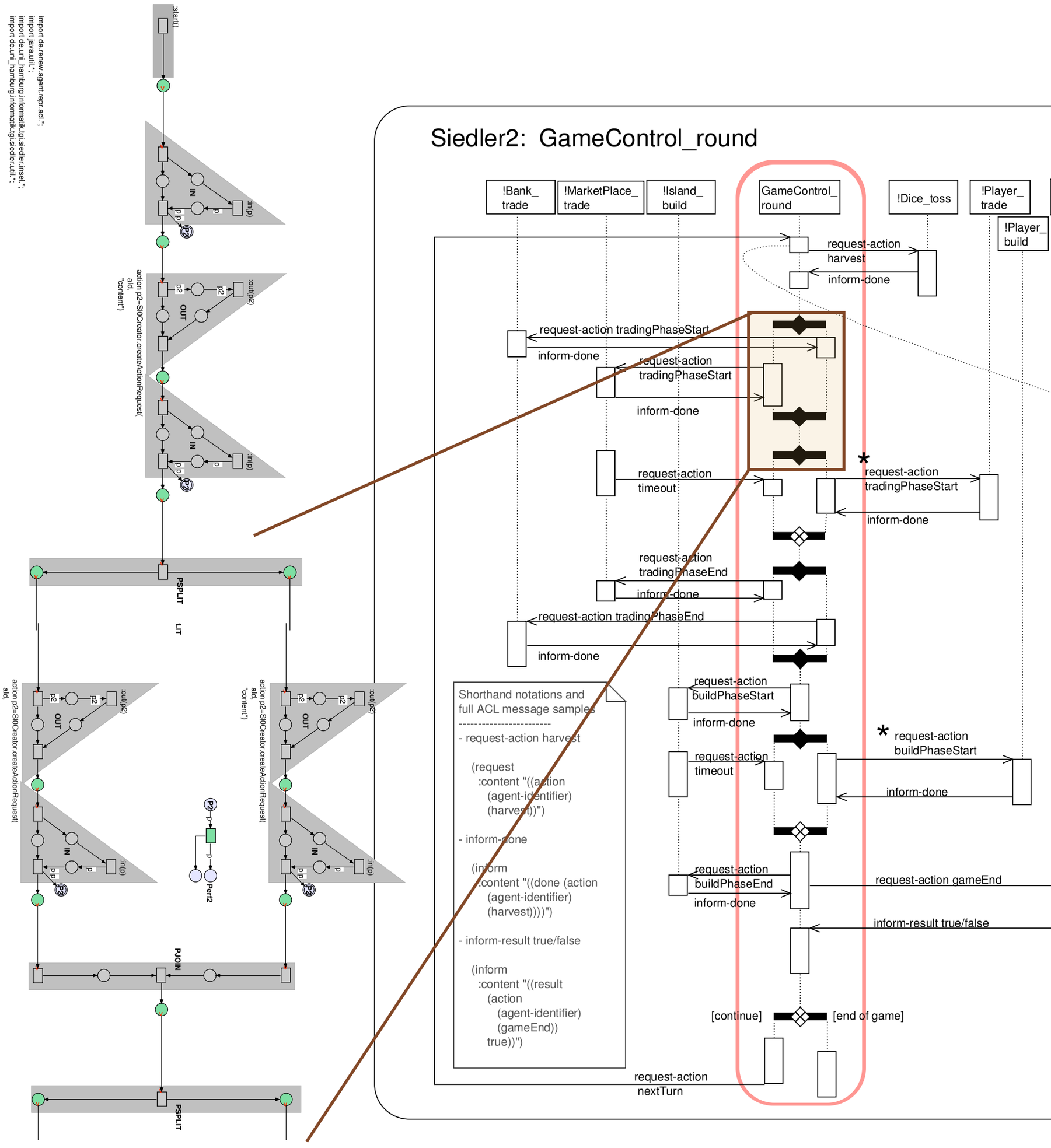
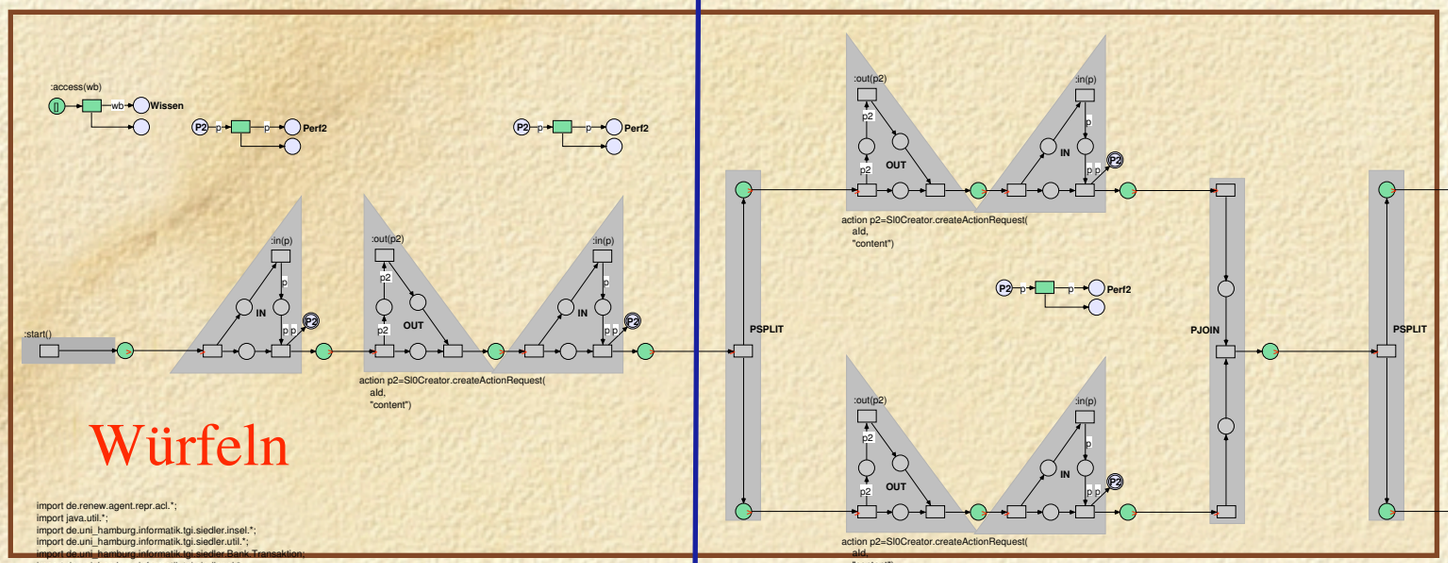


Figure 8.14: Agent interaction protocol diagram for the rou

eine Runde des Spiels

Würfelphase



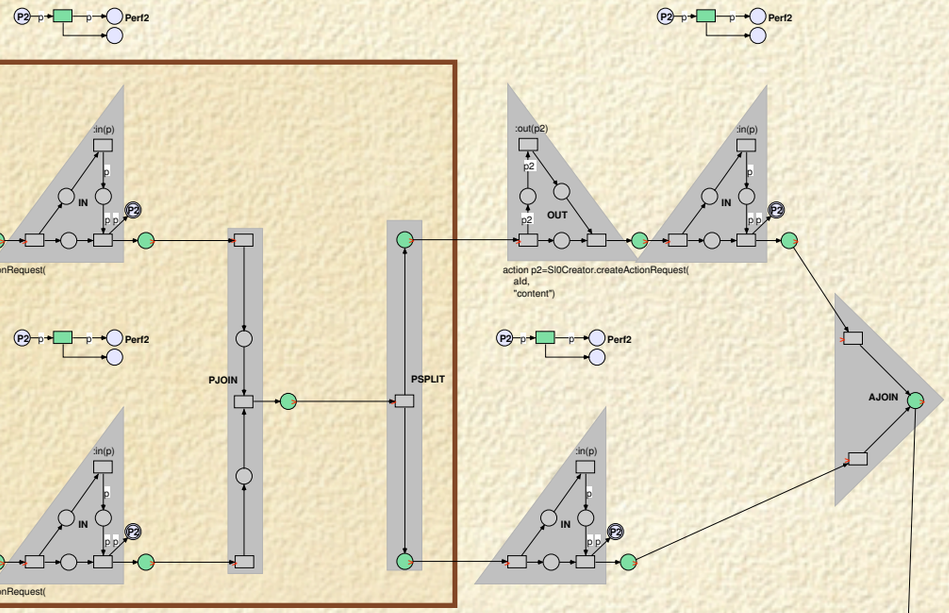
Würfeln

```
import de.renew.agent.repr.acl.*;
import java.util.*;
import de.uni_hamburg.informatik.tgi.siedler.insel.*;
import de.uni_hamburg.informatik.tgi.siedler.util.*;
import de.uni_hamburg.informatik.tgi.siedler.Bank.Transaktion;
import de.uni_hamburg.informatik.tgi.siedler.sl.*;
import de.renew.simulator.NetInstance;

NetInstance wb;
AclMessage p, p2,nachricht,ack ;
AgentIdentifier aId;
Insel insel;
Transaktion e;

boolean cond;
Object o;
Object[] os;
String s;
Vector v;
int y;
Iterator it;
```

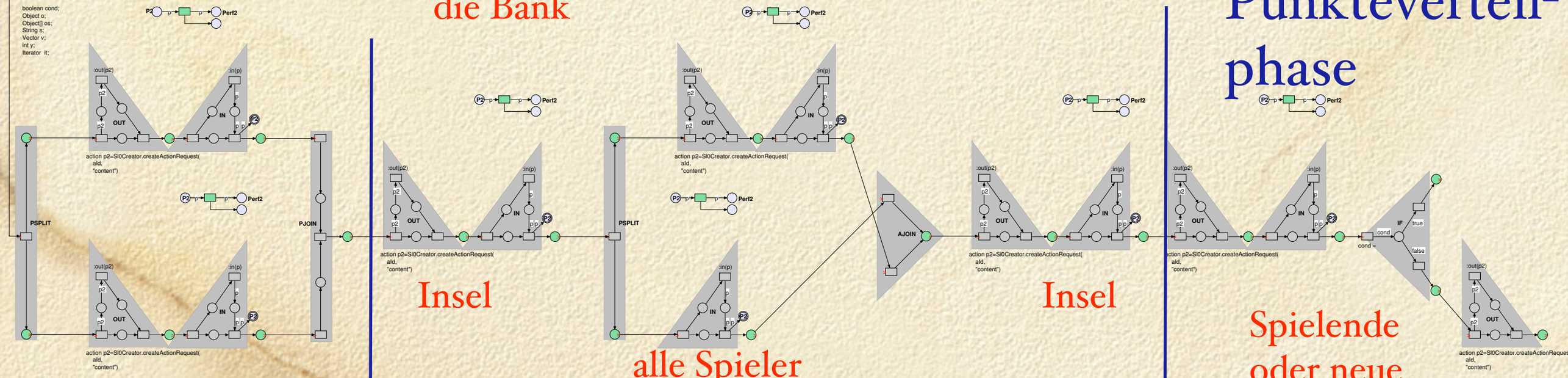
Handelsphase



Ausgabe der Güter durch die Bank

Handel der Spieler

Punkteverteil-phase



Ausgabe der Gewinne durch die Bank

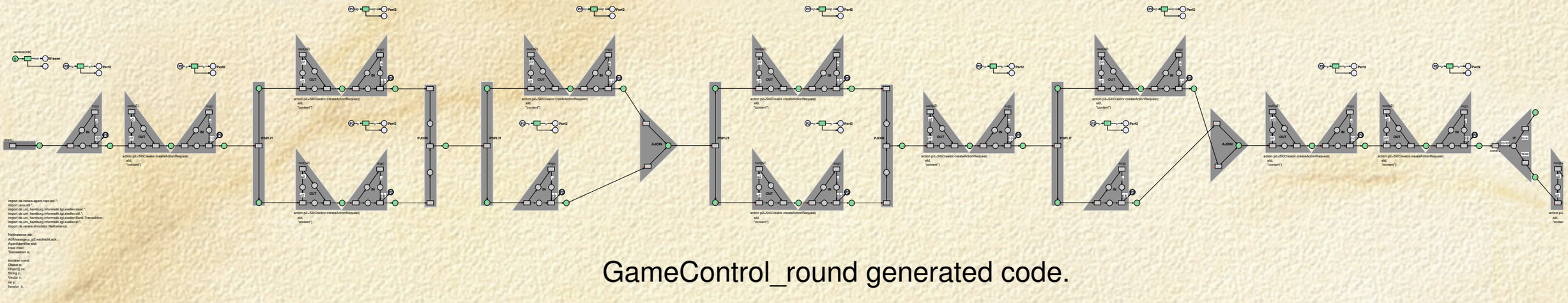
Insel

alle Spieler

Insel

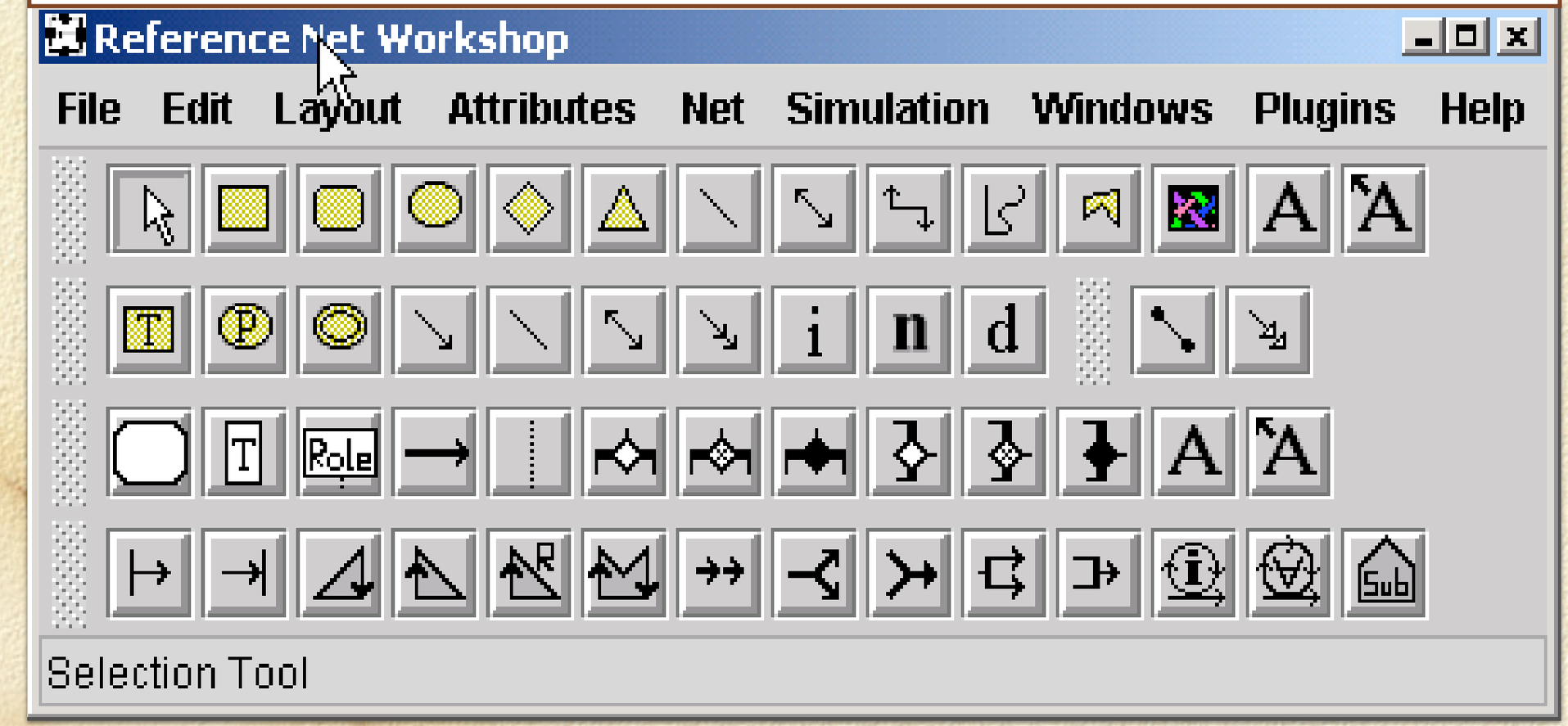
Spielende oder neue Runde

Bauphase



GameControl_round generated code.

Renew:
interaction diagrams & component plug-ins



Conclusion