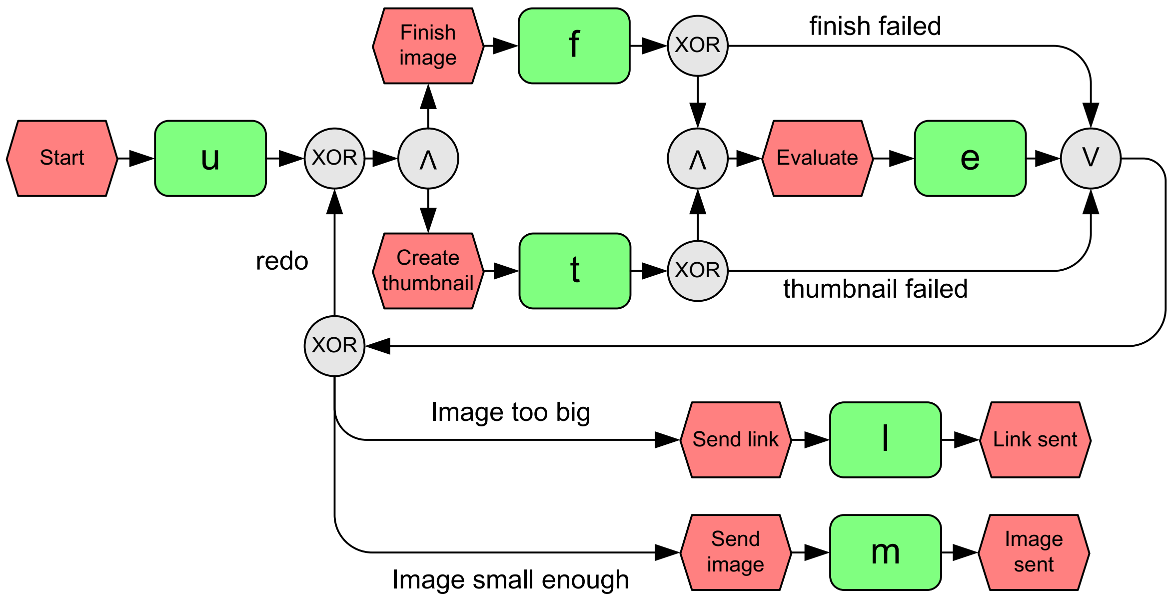
Business Processes Modelling

MPB (6 cfu, 295AA)

Roberto Bruni
http://www.di.unipi.it/~bruni

20 - EPC analysis

Object



We overview the main challenges that arise when analysing EPC diagrams with Petri nets

Ch. 6 of Business Process Management: Concepts, Languages, Architectures

EPC Diagrams

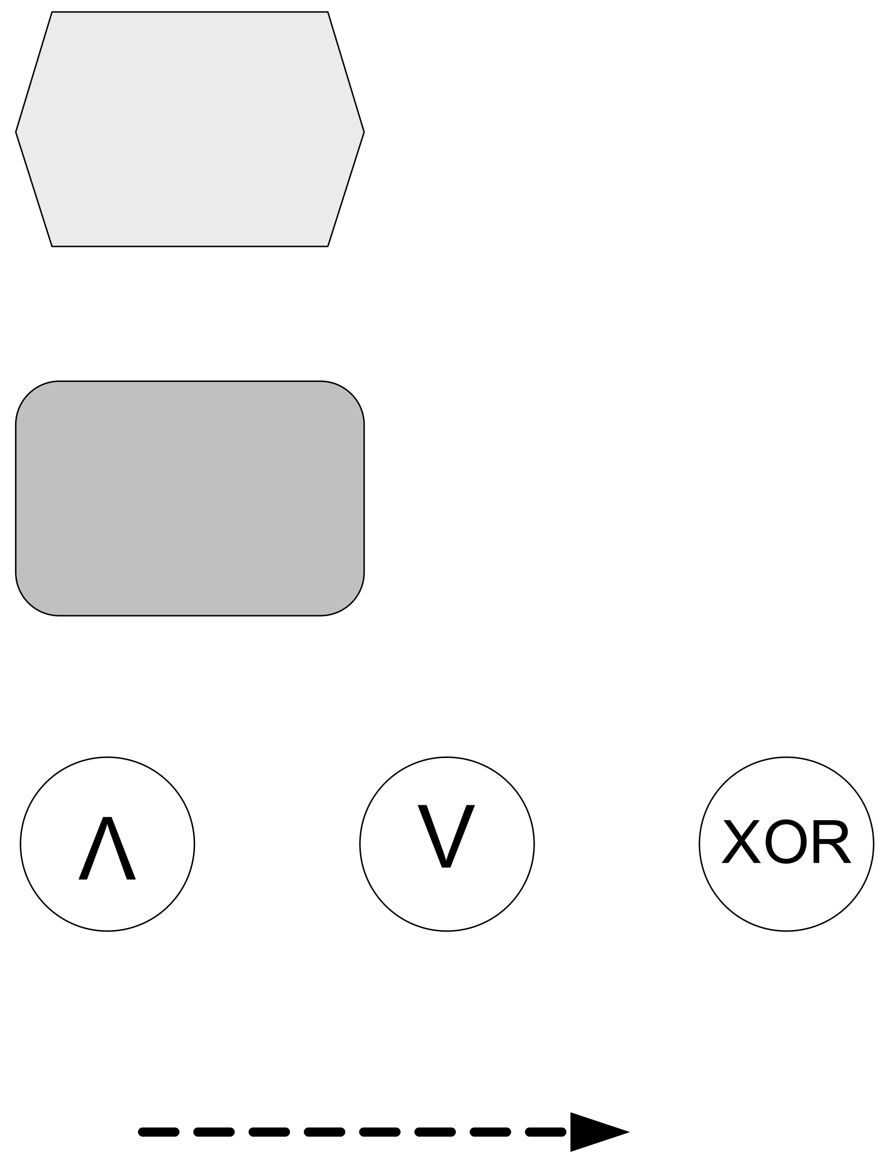
EPC ingredients at a glance

Event

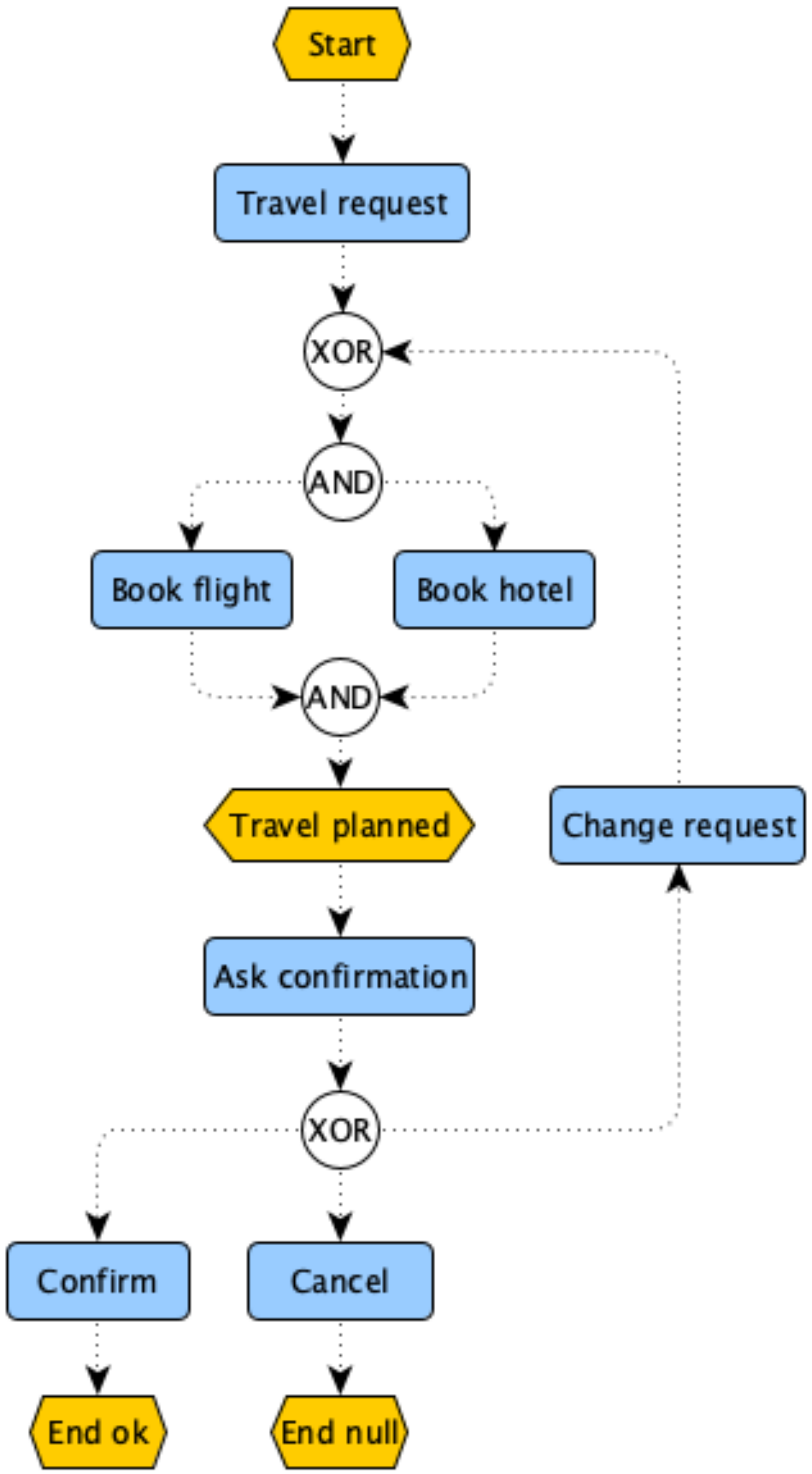
Function

Connectors

Control Flow



EPC: Example


Start
Travel request
XOR
AND
Book flight
Book hotel
AND
Travel planned
Change request
Ask confirmation
XOR
Confirm
Cancel
End ok
End null

EPC Semantics

Sound EPC diagrams

We exploit the formal semantics of nets
to give unambiguous semantics to EPC diagrams

We transform EPC diagrams to Workflow nets:
the EPC diagram is sound if its net is so

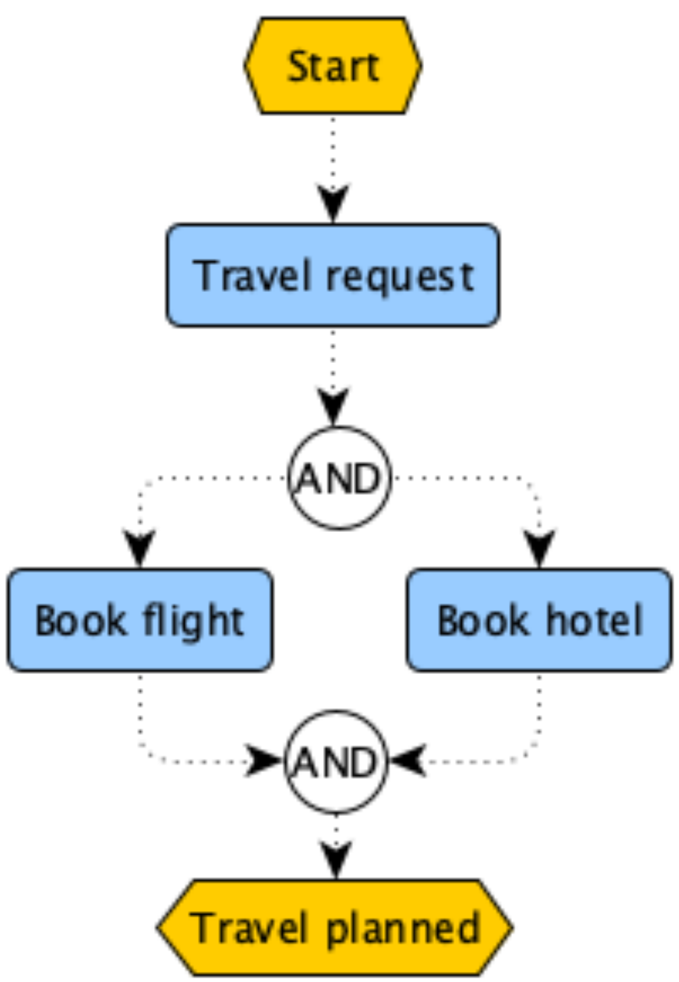
We can reuse the verification tools
to check if the net is sound

Is there a unique way to proceed? Not necessarily!

Translation of EPC to Petri nets

The idea

From EPC to wf nets in three steps



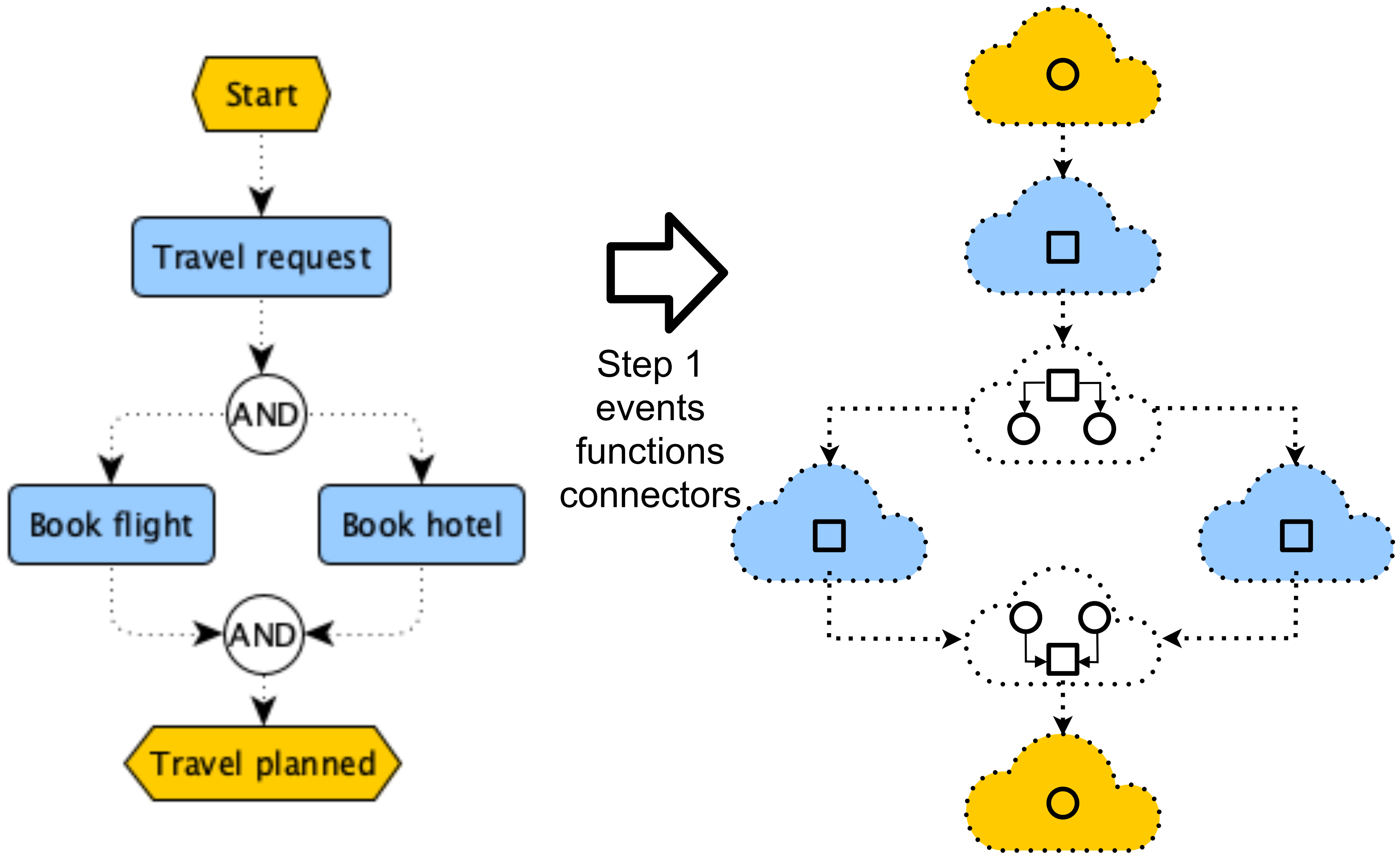
Step 1
convert each
- event
- function
- connector
to a net fragment

Step 2
connect
fragments
together

Step 3
enforce
initial place
final place

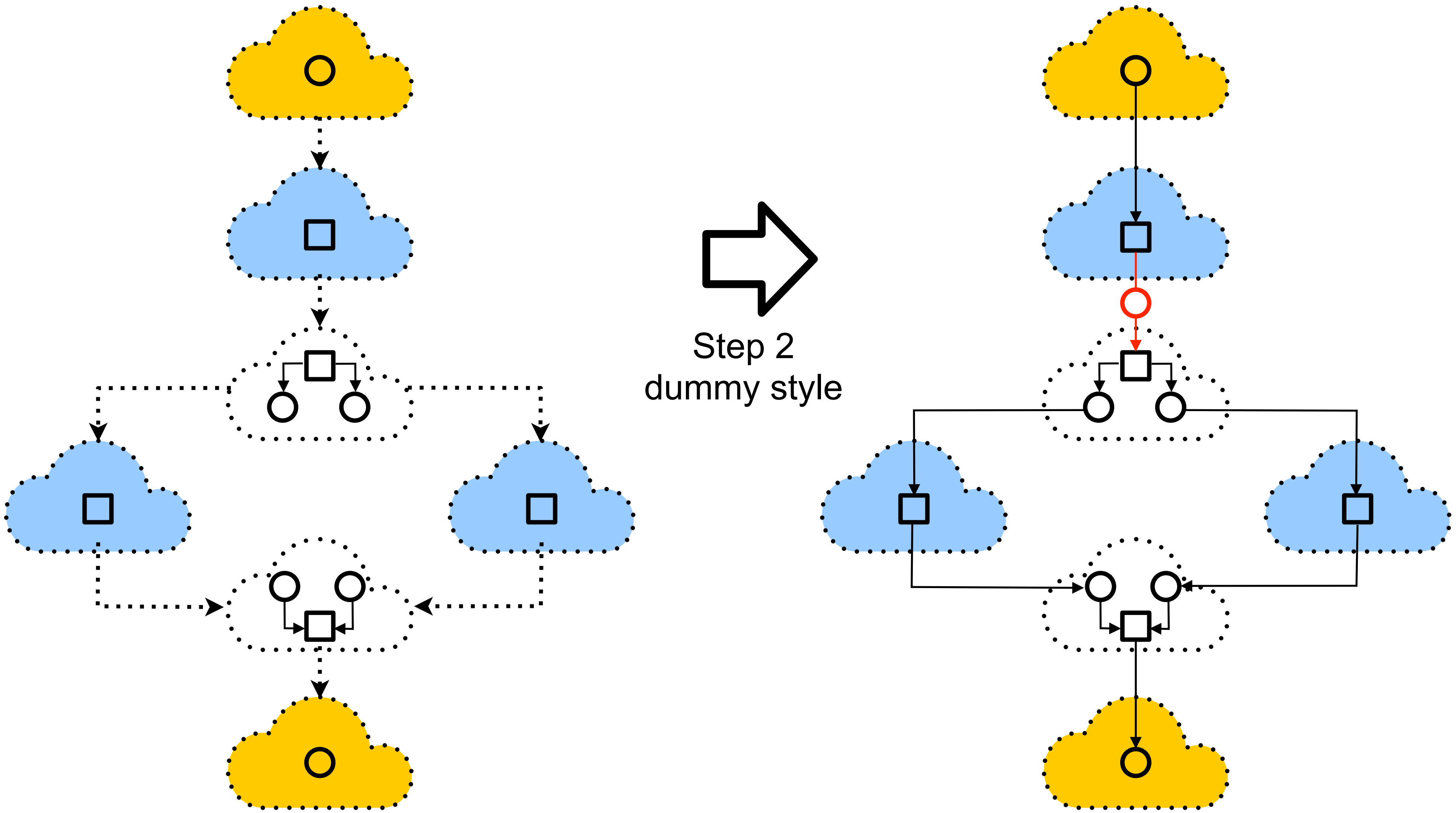
Step 1

We replace each event, function and connector separately with small net fragments



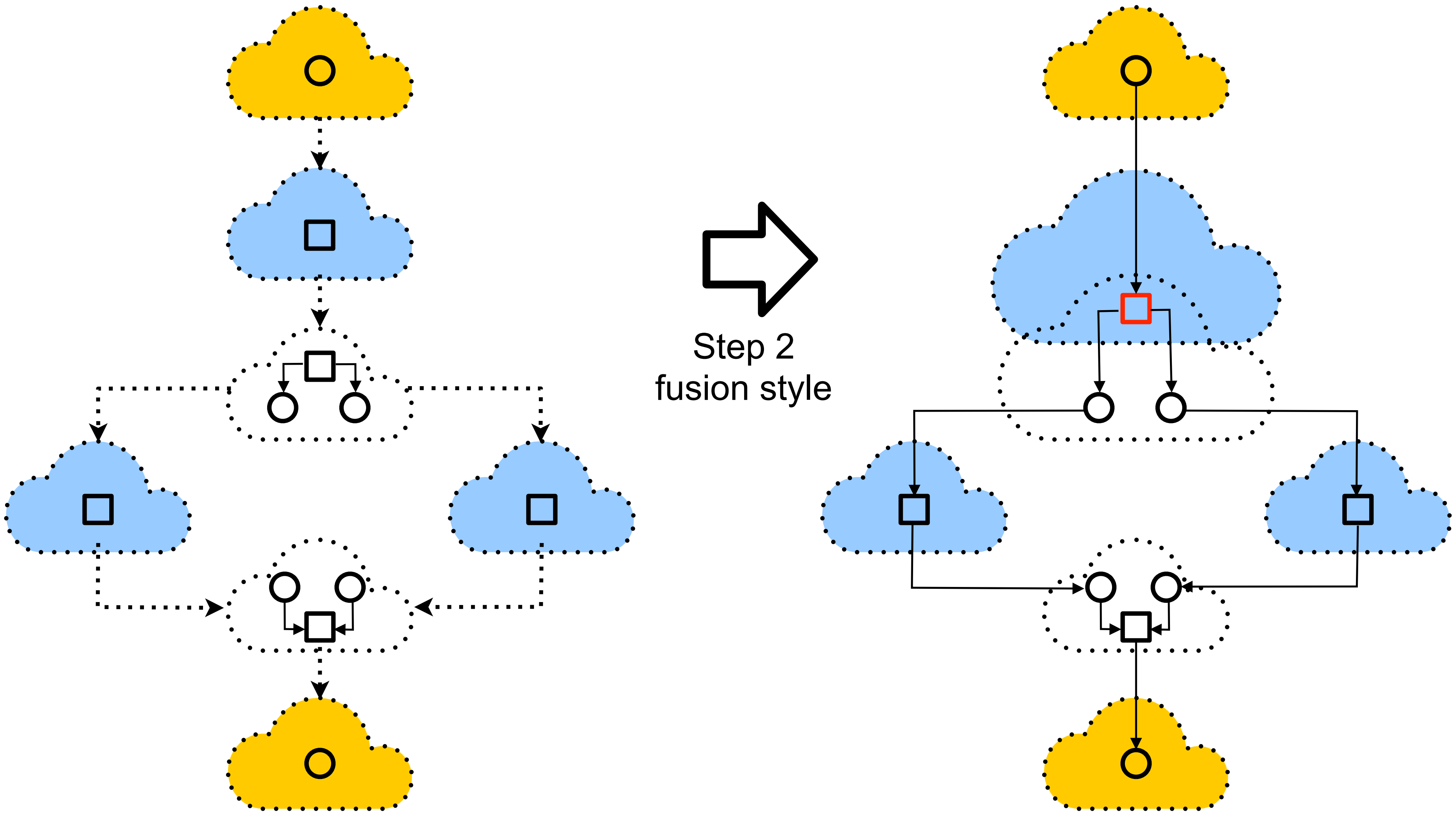
Step 2: dummy style

Then we connect the fragments together
(we may decide to introduce dummy places / transitions)

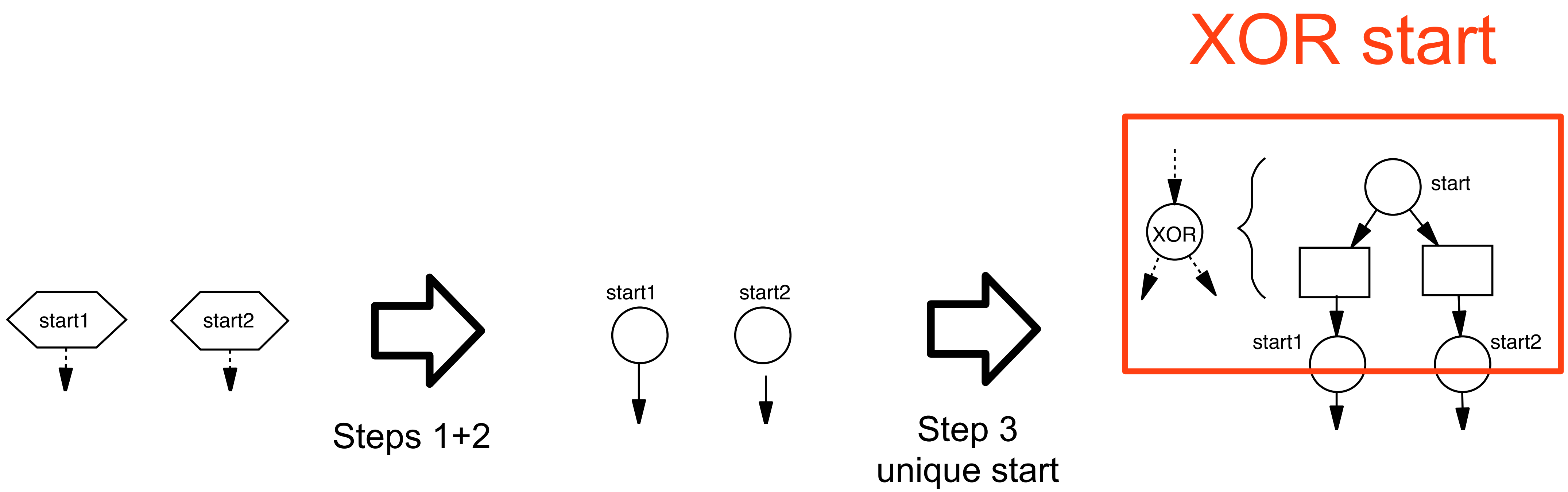


Step 2: fusion style

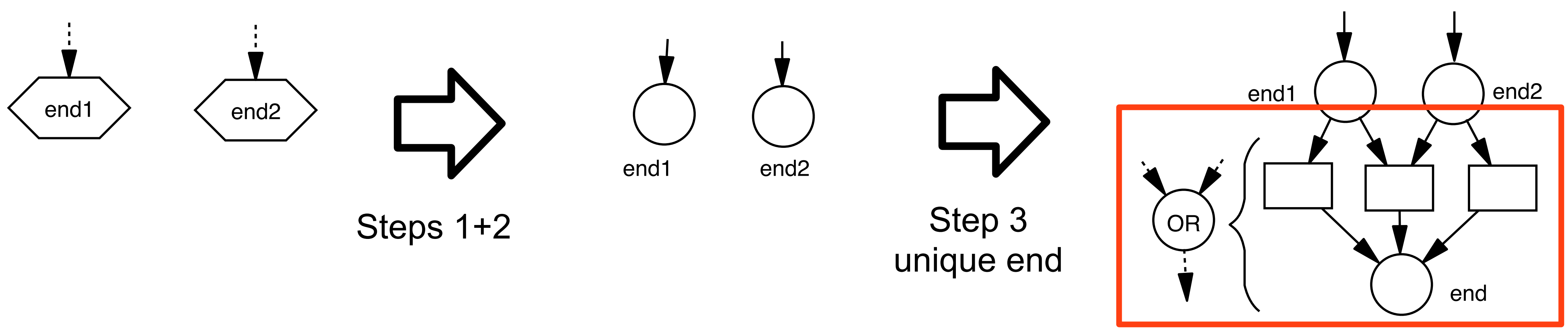
Then we connect the fragments together
(or we may decide to merge places / transitions)



Step 3: unique start



Step 3: unique end



OR end

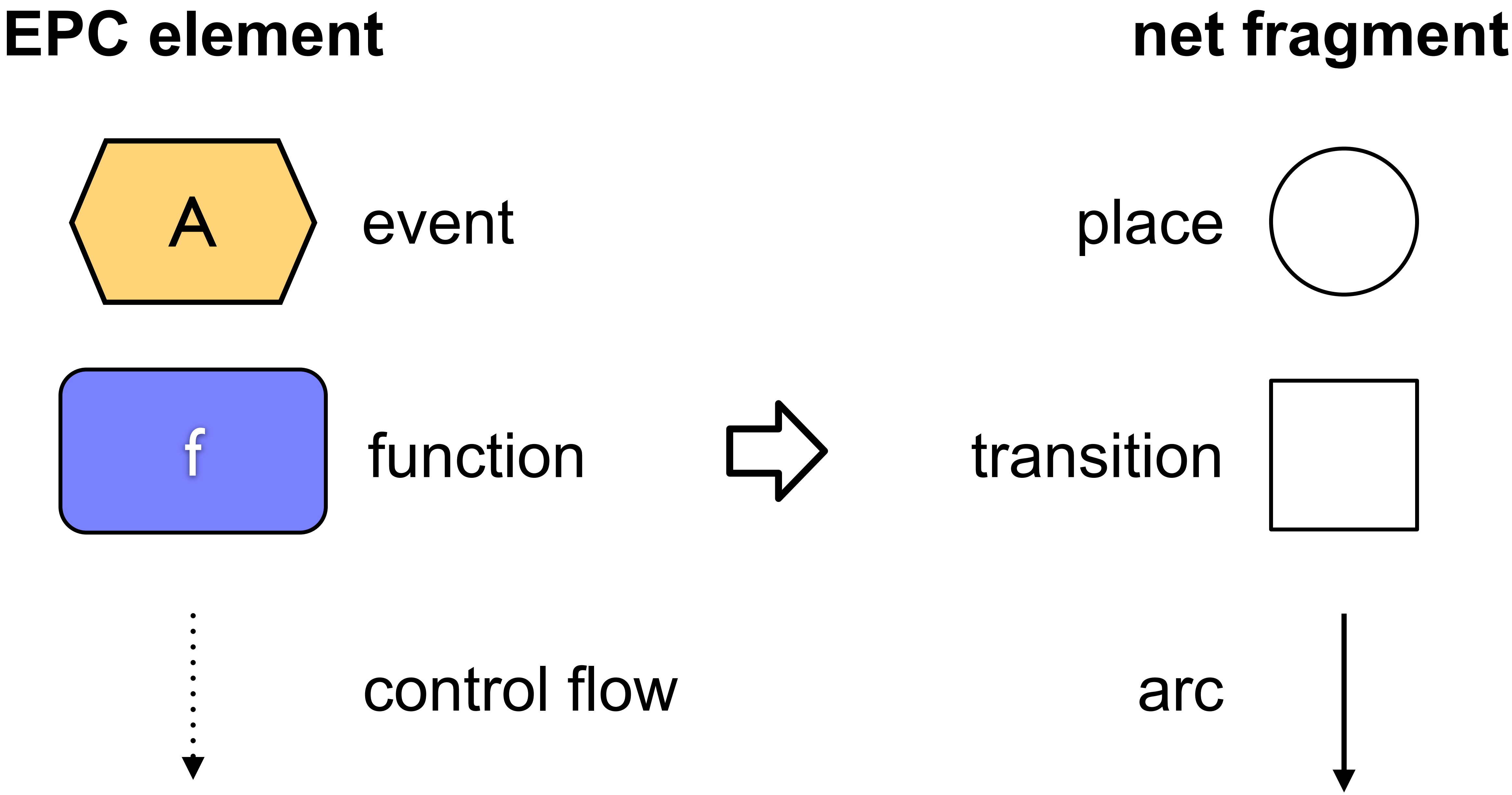
(sometimes XOR/AND can be preferred)

Three approaches

We overview **three** different translations

n.	**ingenuity**	**style**	**applicability**	**outcome**
1st	easy	fusion	any EPC	likely unsound, (relaxed soundness)
2nd	medium, context dependent	(dummy)	simplified EPC: event function alternation, no OR connectors	free-choice net
3rd	hard, context dependent	dummy	decorated EPC: join-split correspondence, OR policies	accurate analysis

Commonalities



First attempt (straight translation)

Relaxed Soundness of Business Processes

Juliane Dehnert[1,*] and Peter Rittgen[2]

[1] Institute of Computer Information Systems, Technical University Berlin, Germany
dehnert@cs.tu-berlin.de

[2] Institute of Business Informatics, University Koblenz-Landau, Germany
rittgen@uni-koblenz.de

K.R. Dittrich, A. Geppert, M.C. Norrie (Eds.): CAiSE 2001, LNCS 2068, pp. 157–170, 2001.

Rationale

EPC success is due to its **simplicity**

EPC diagrams lack a consistent semantics:
ambiguous and flawed process descriptions
can arise in the **design phase**

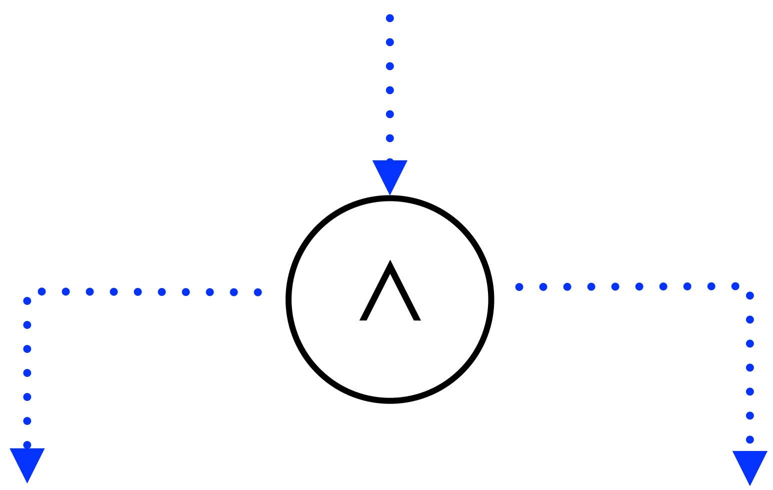
it is important to find out **flaws** as **soon** as possible

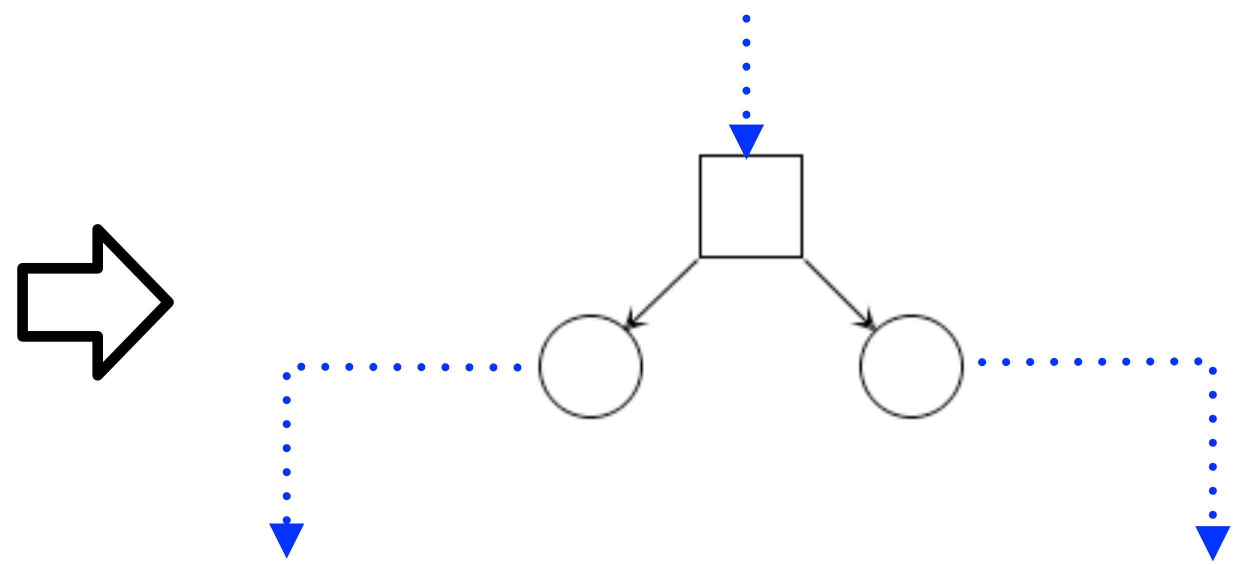
therefore

we need to fix a **formal representation**
that **preserves all ambiguities**

Step 1: AND split

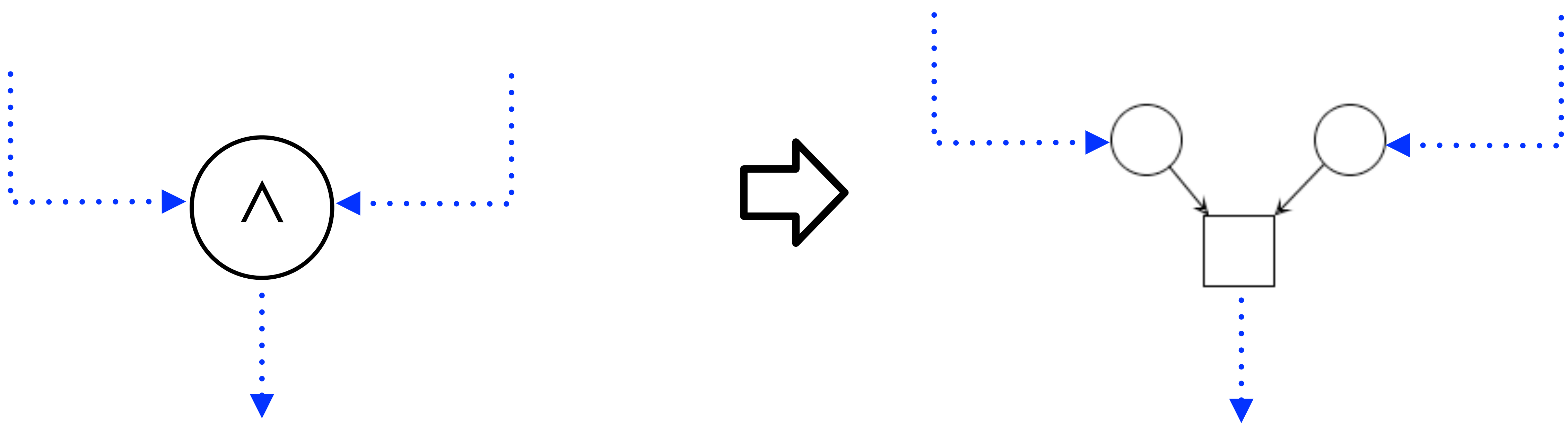
EPC element **net fragment**





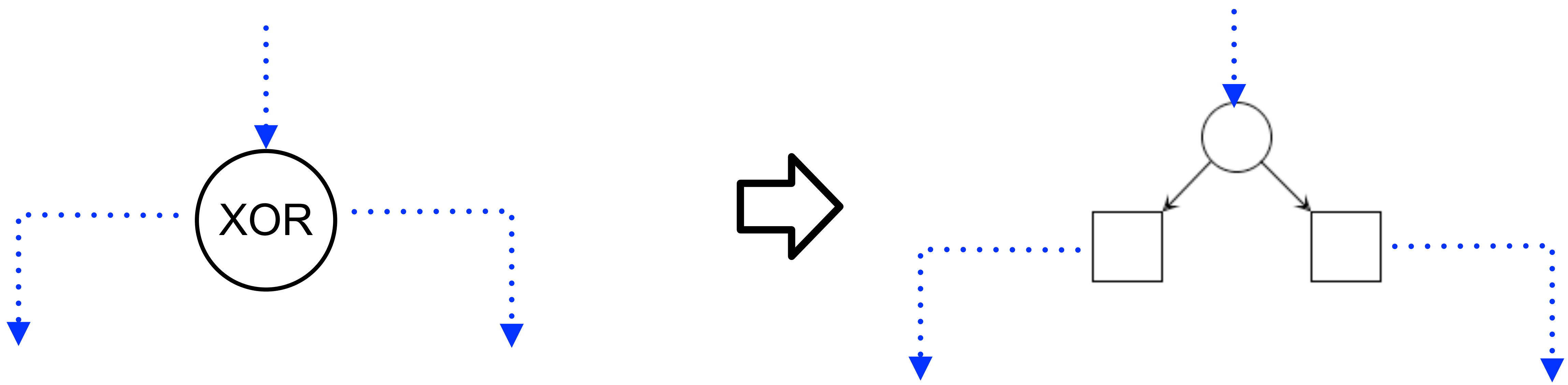
Step 1: AND join

EPC element **net fragment**



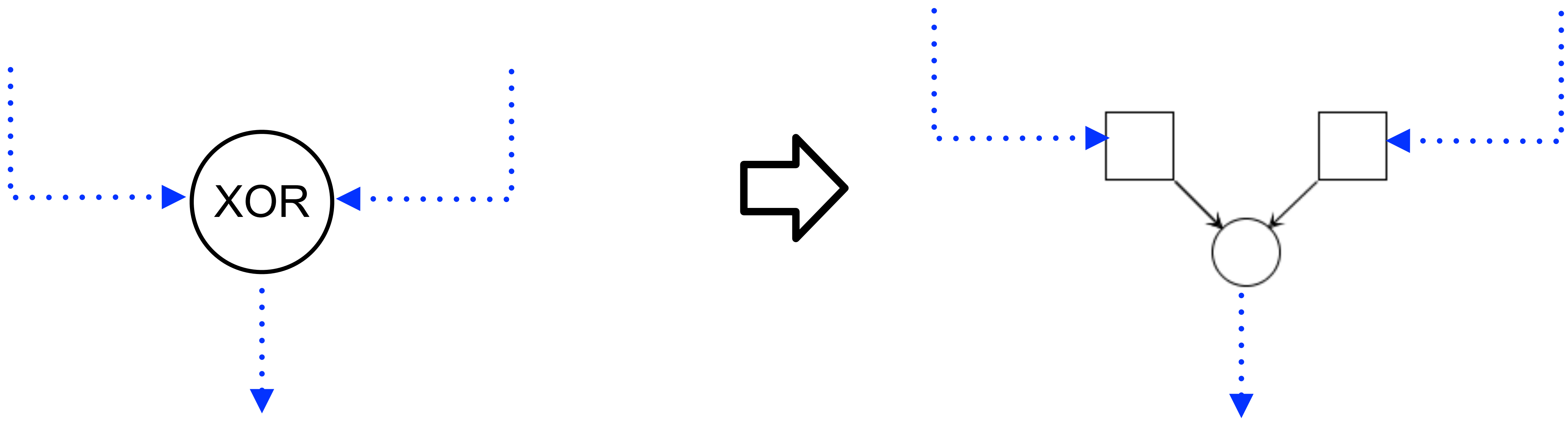
Step 1: XOR split

EPC element **net fragment**

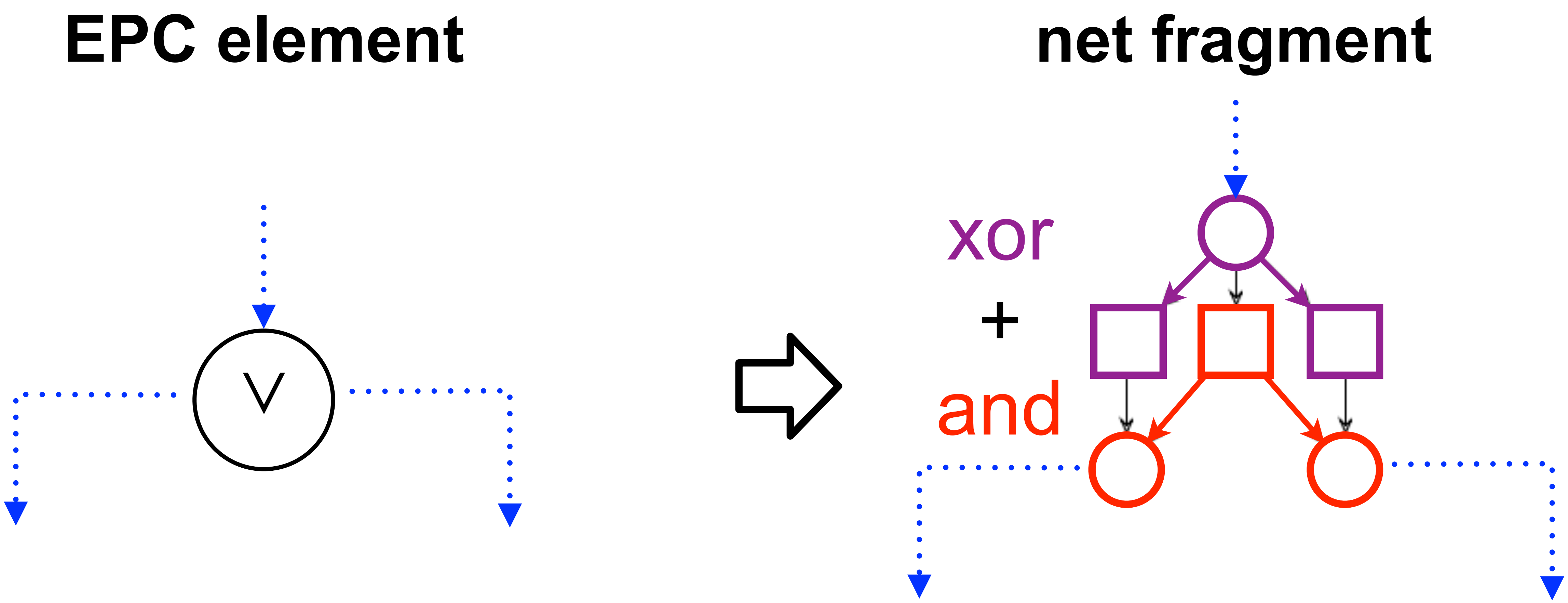


Step 1: XOR join

EPC element **net fragment**

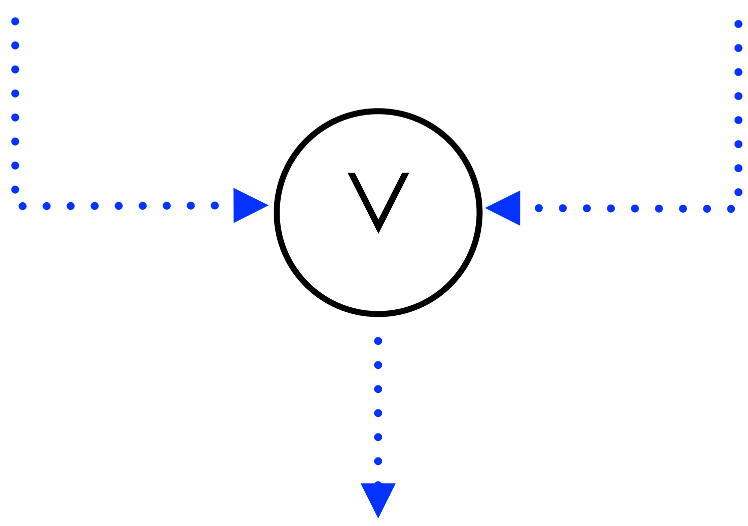


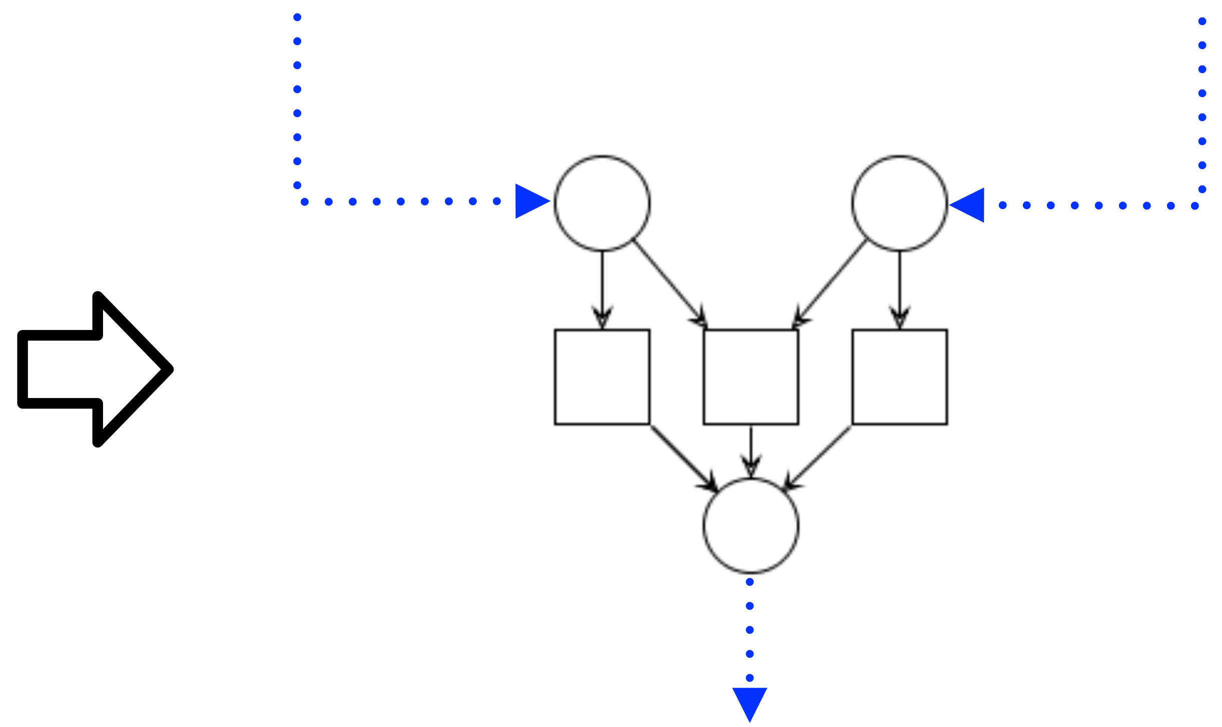
Step 1: OR split



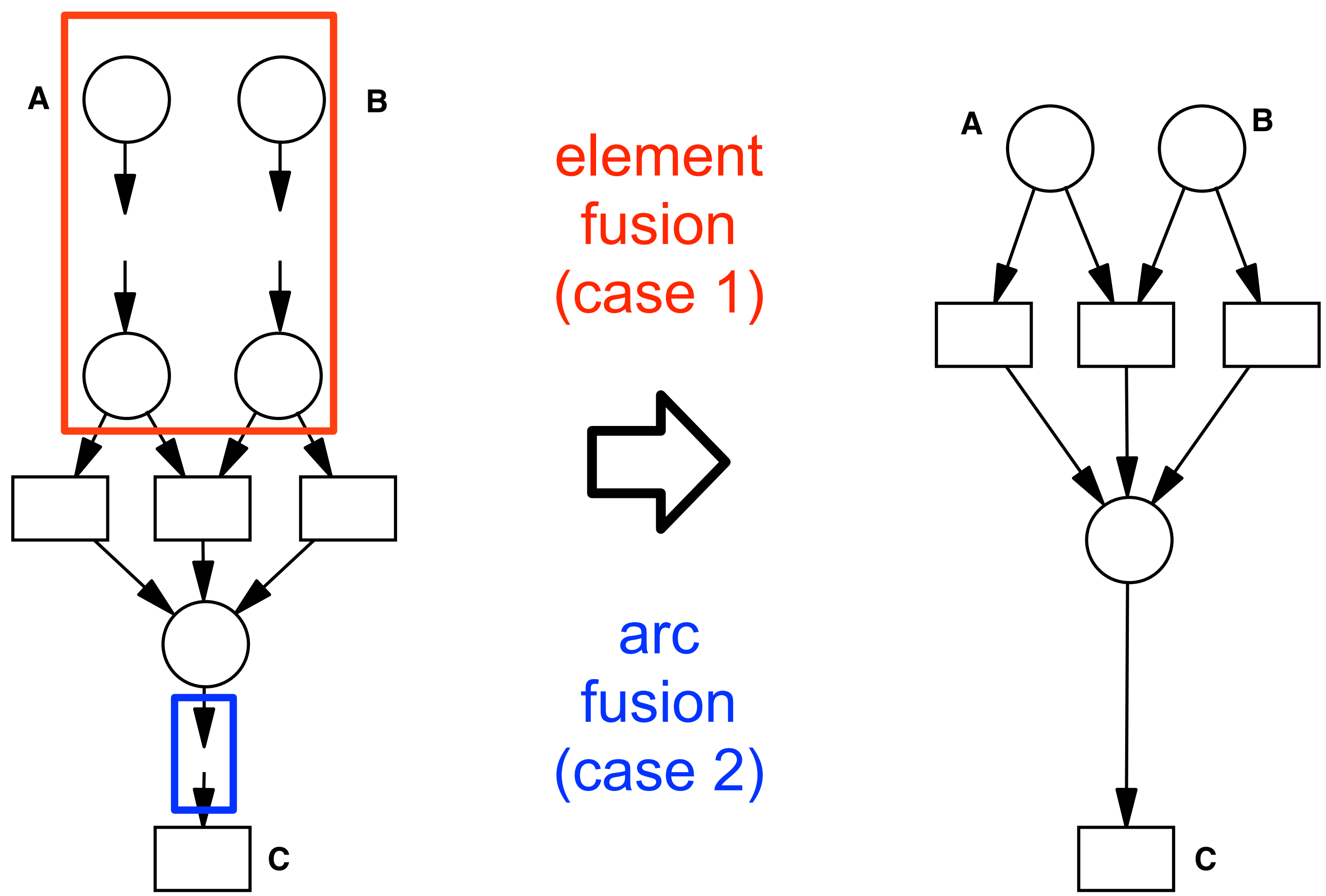
Step 1: OR join

EPC element **net fragment**

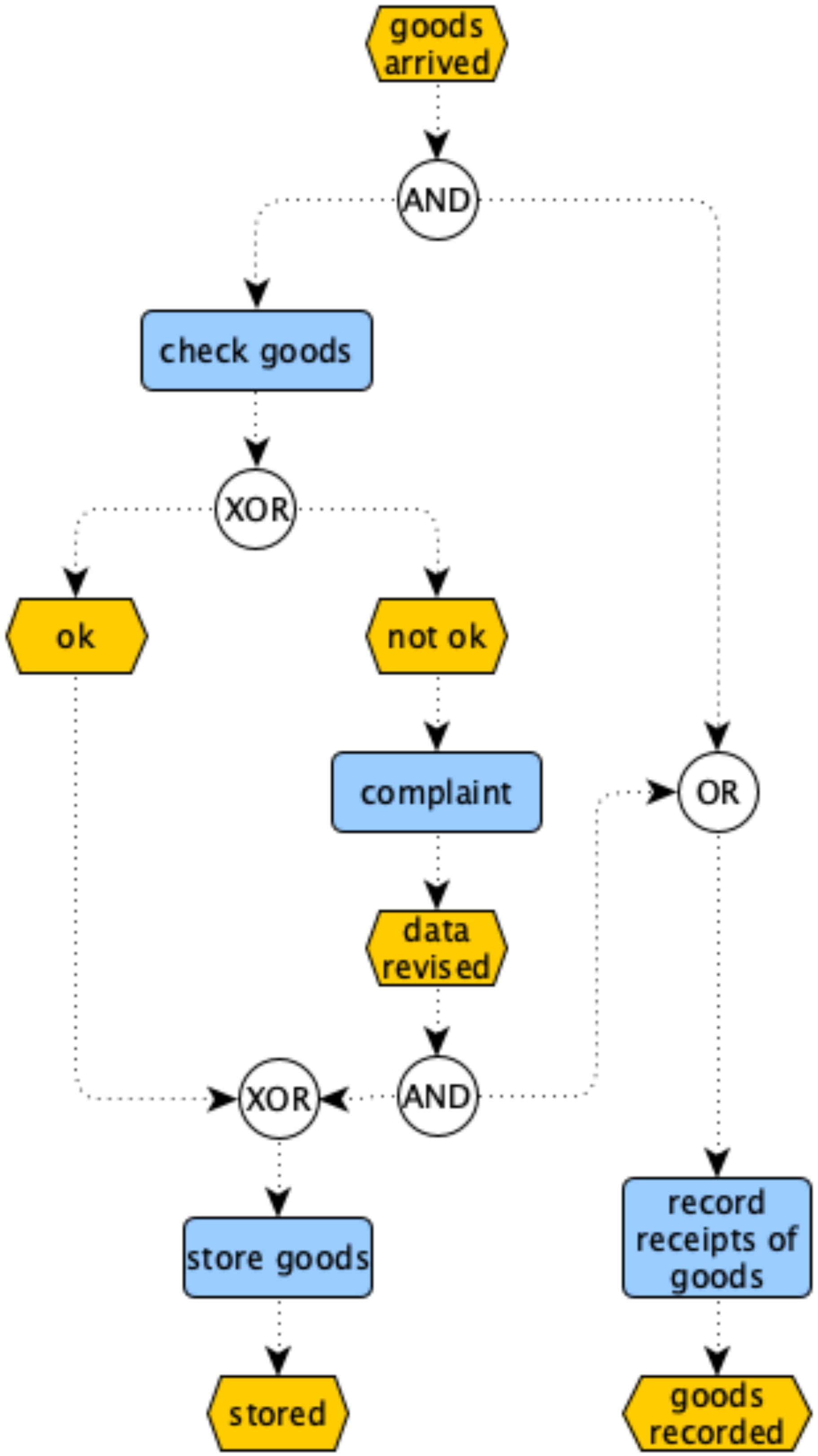




Step 2: fusion style

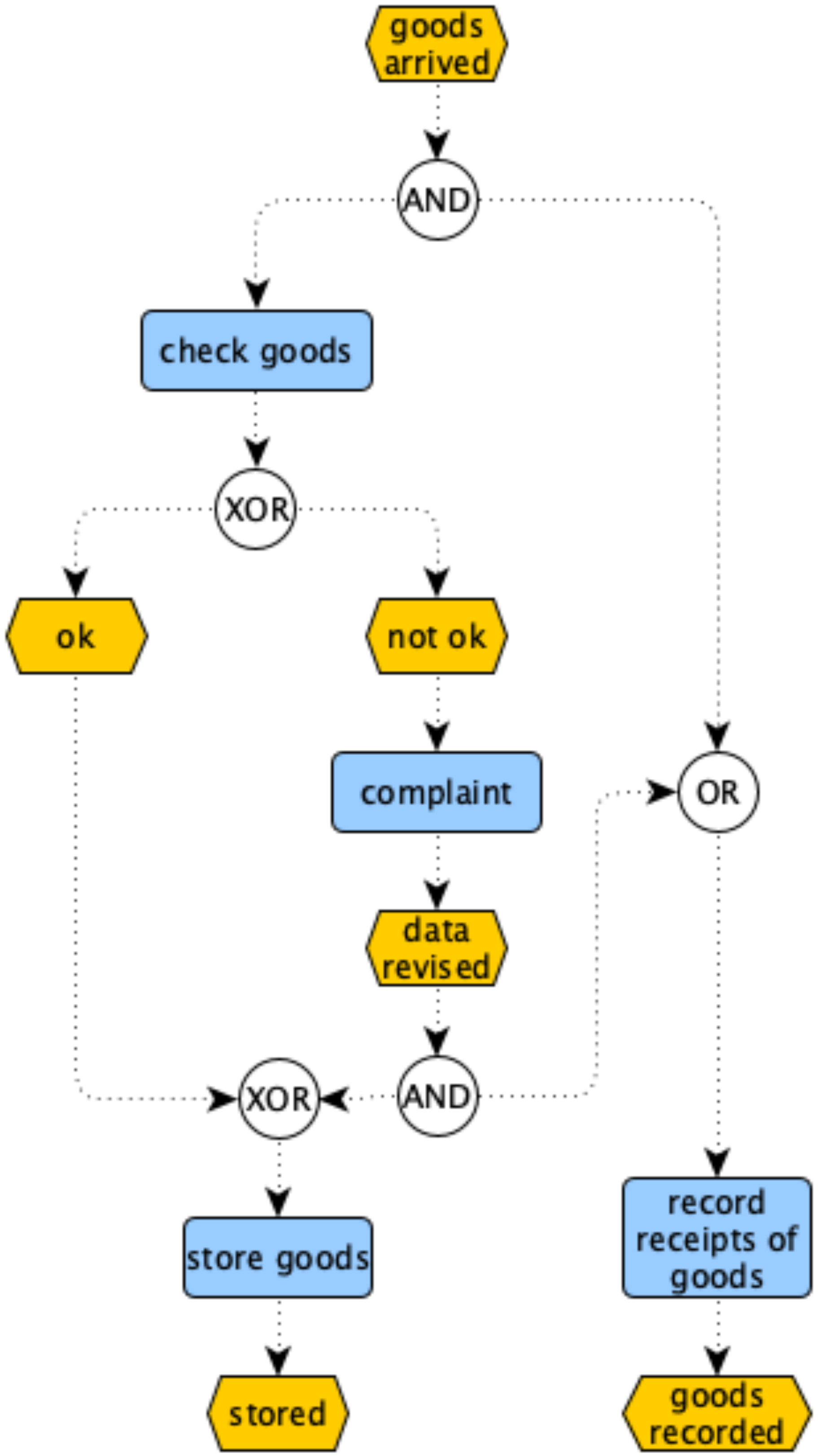


Example

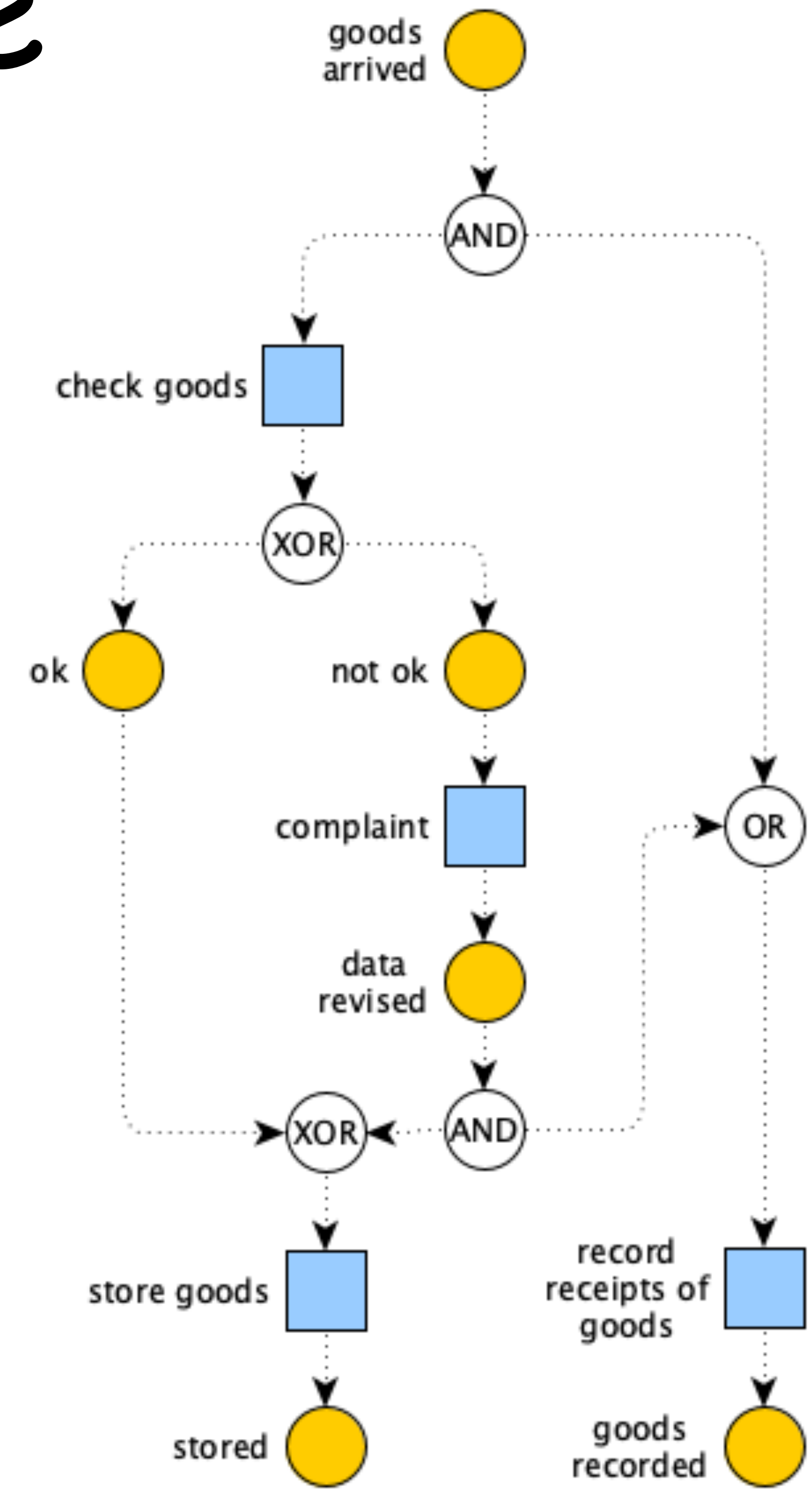


Sound?

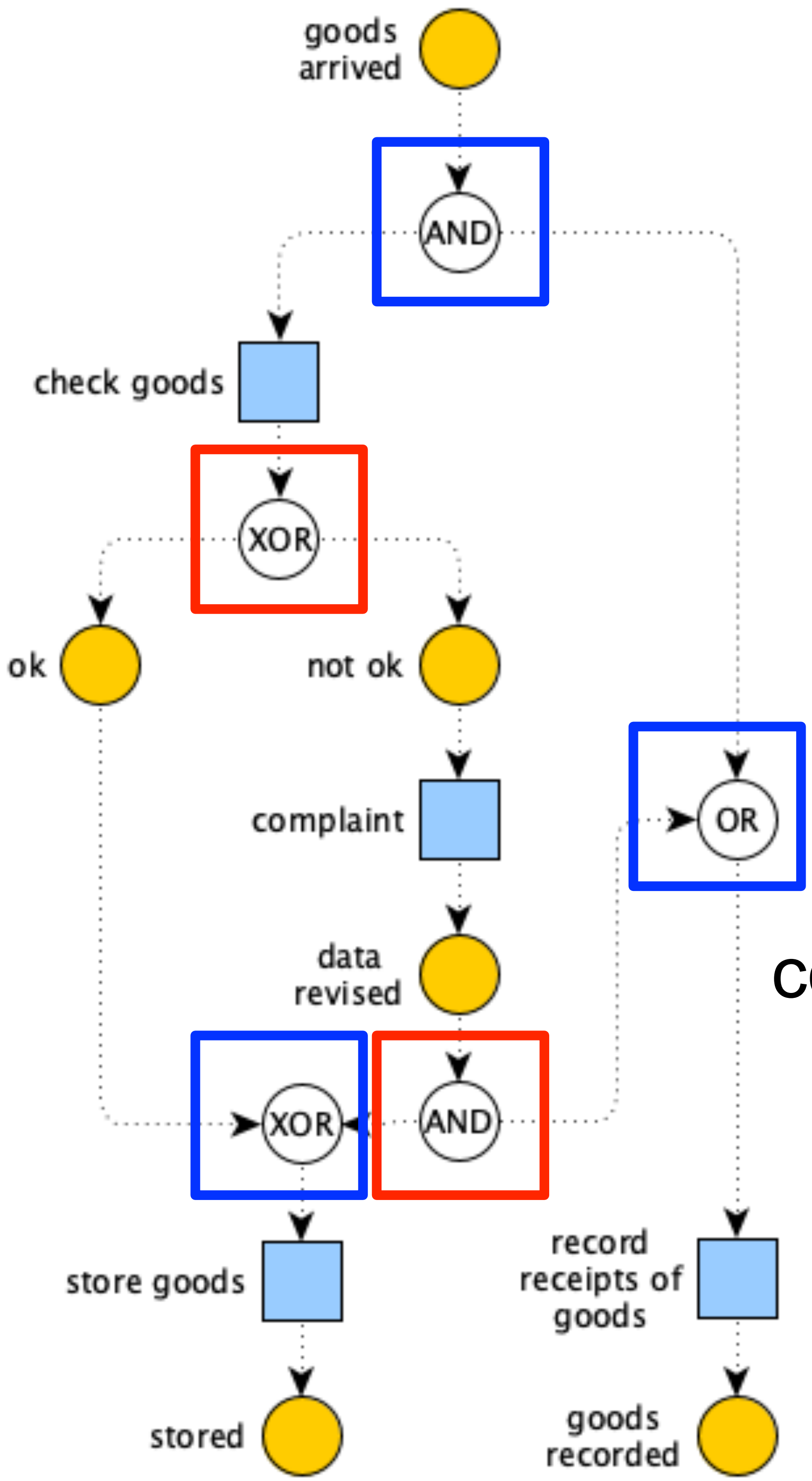
Example



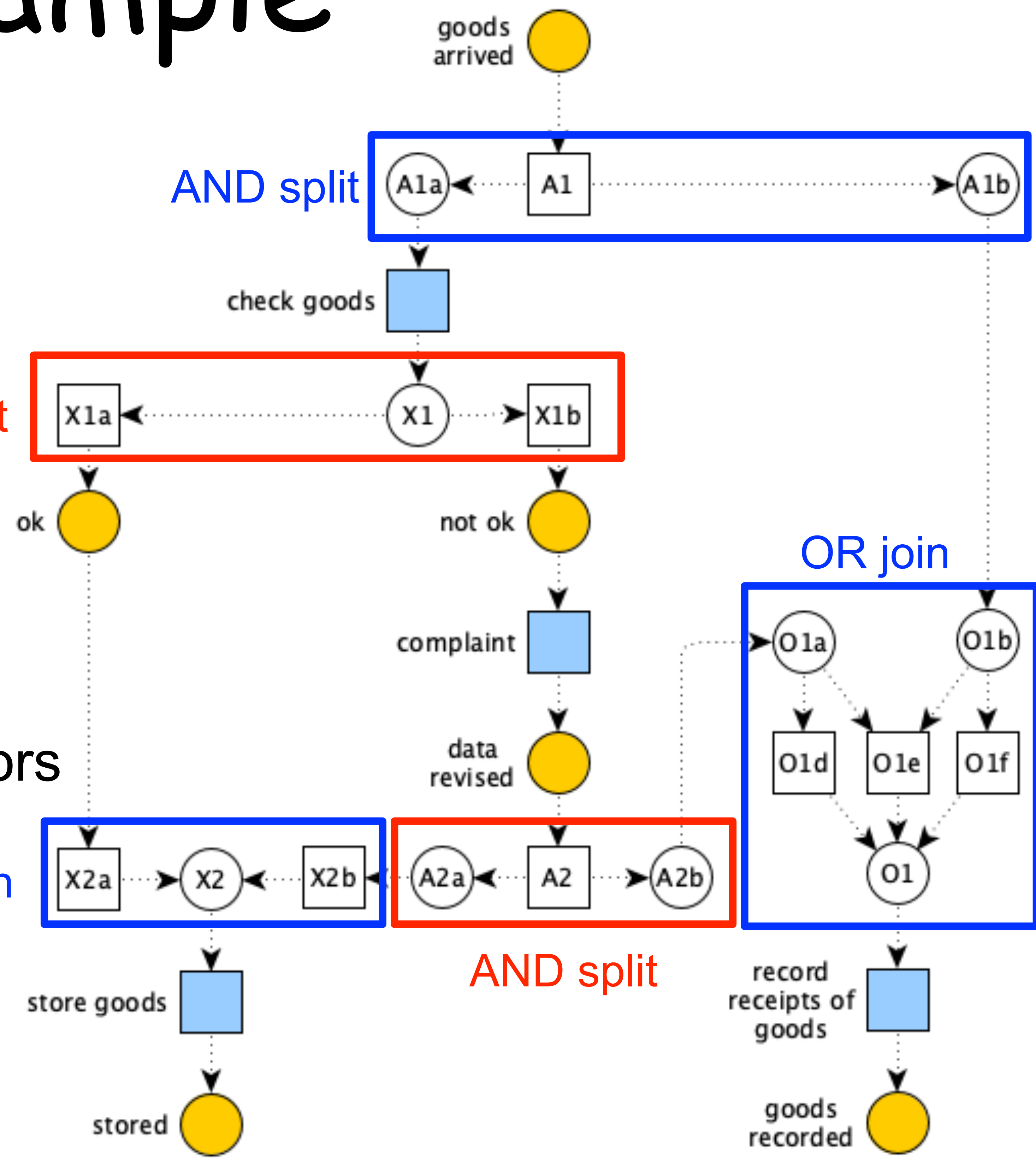
Step 1
events and
functions



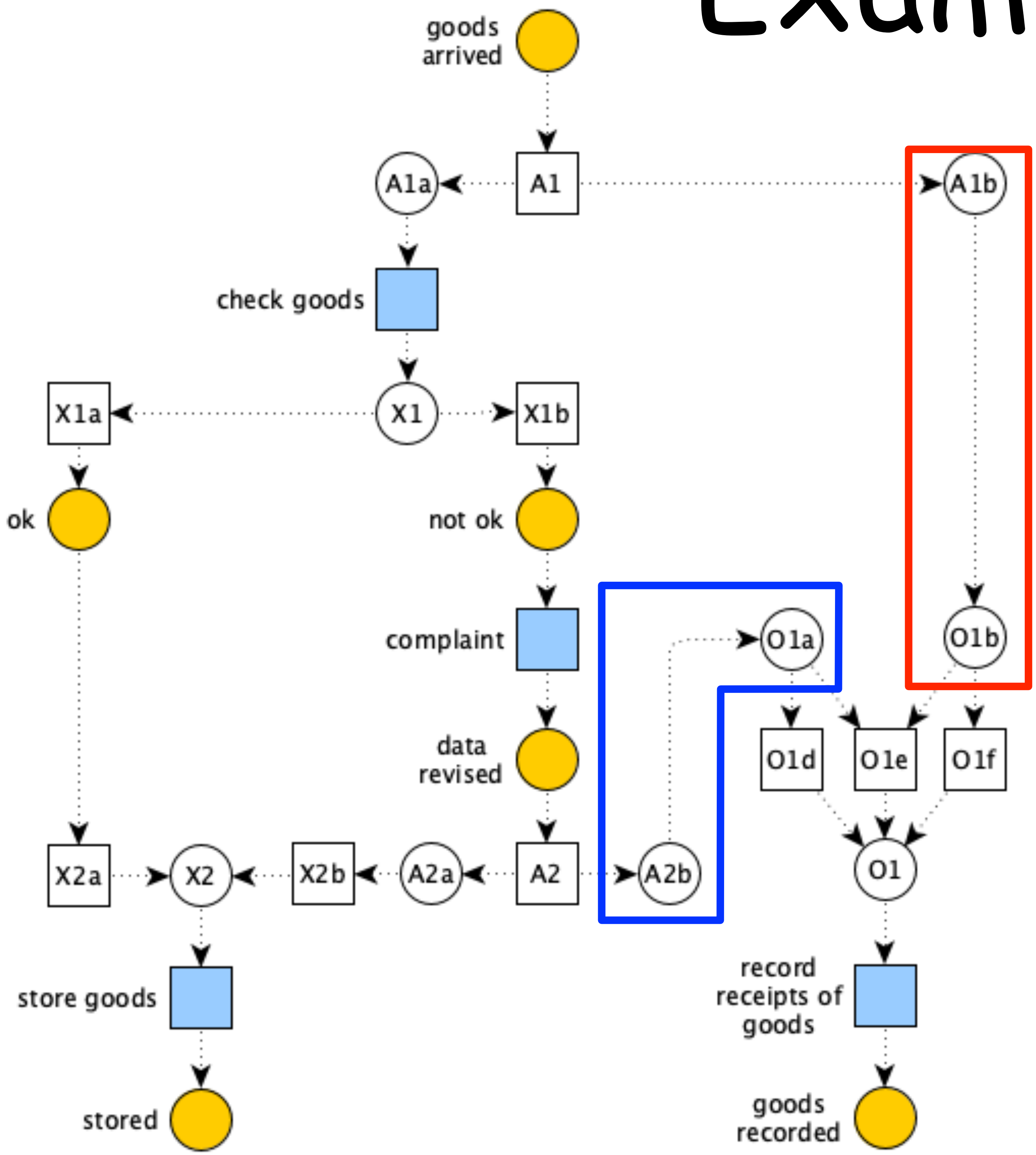
Example



Step 1
connectors

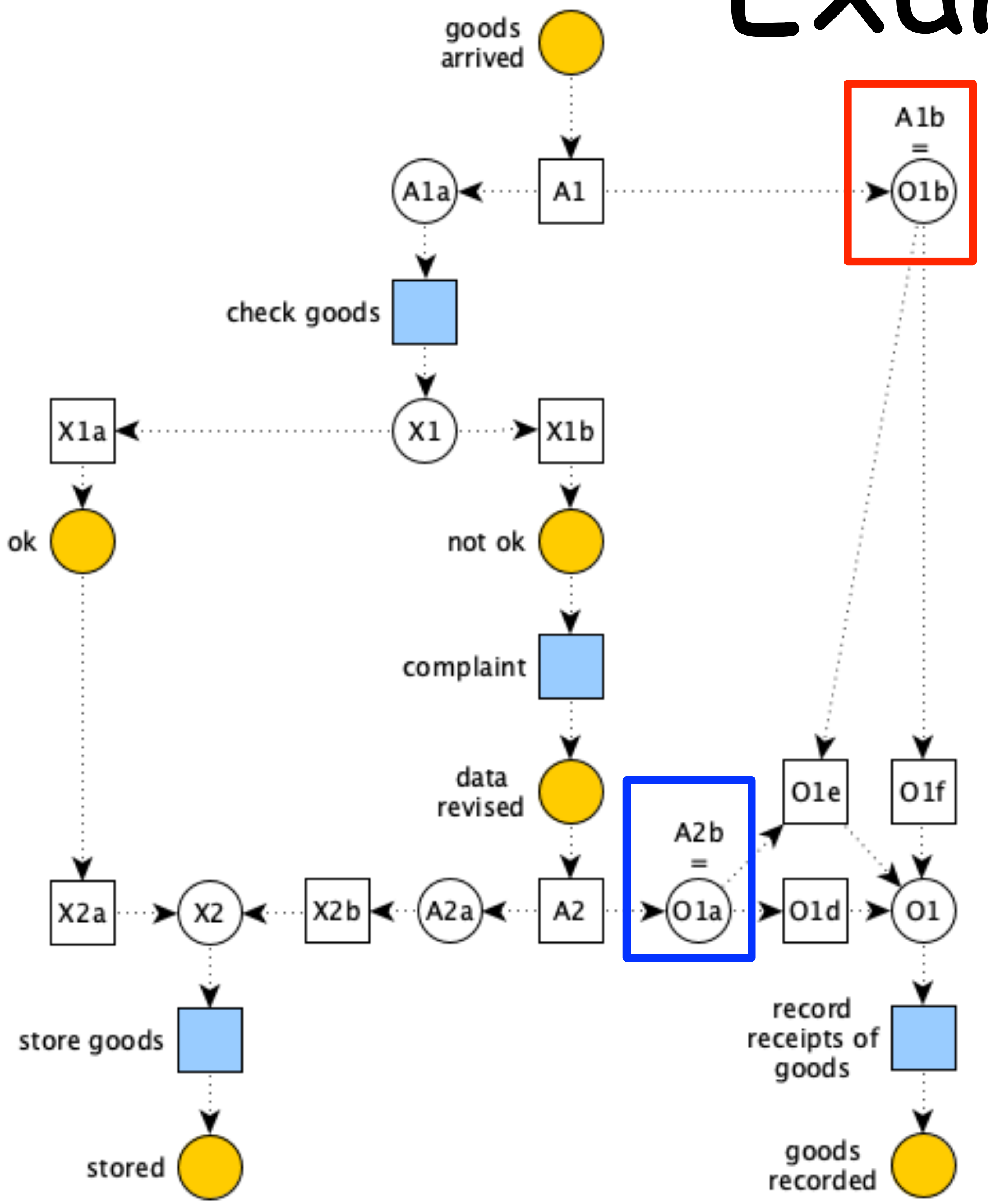


Example



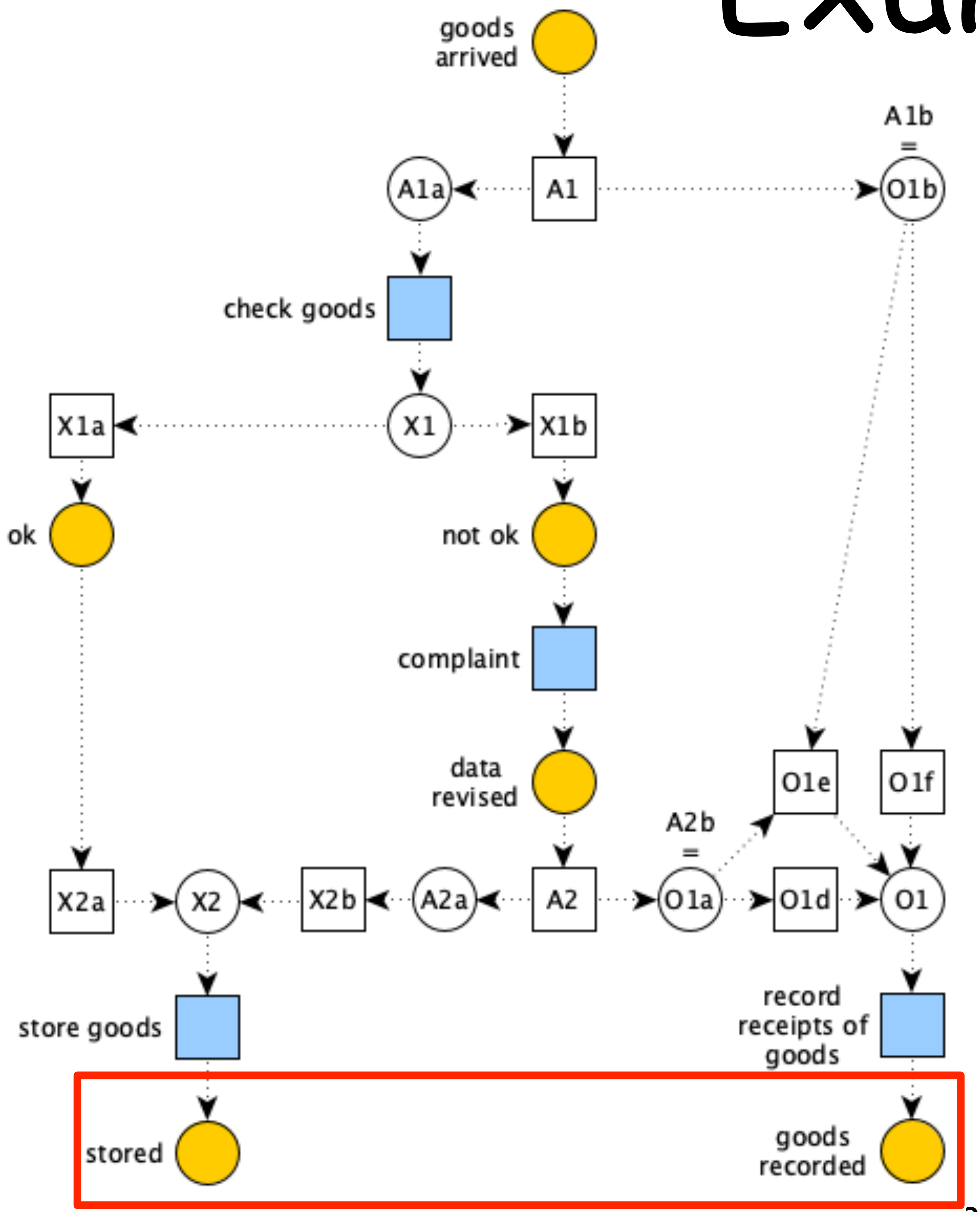
Step 2 fusion

Example



Step 2 fusion

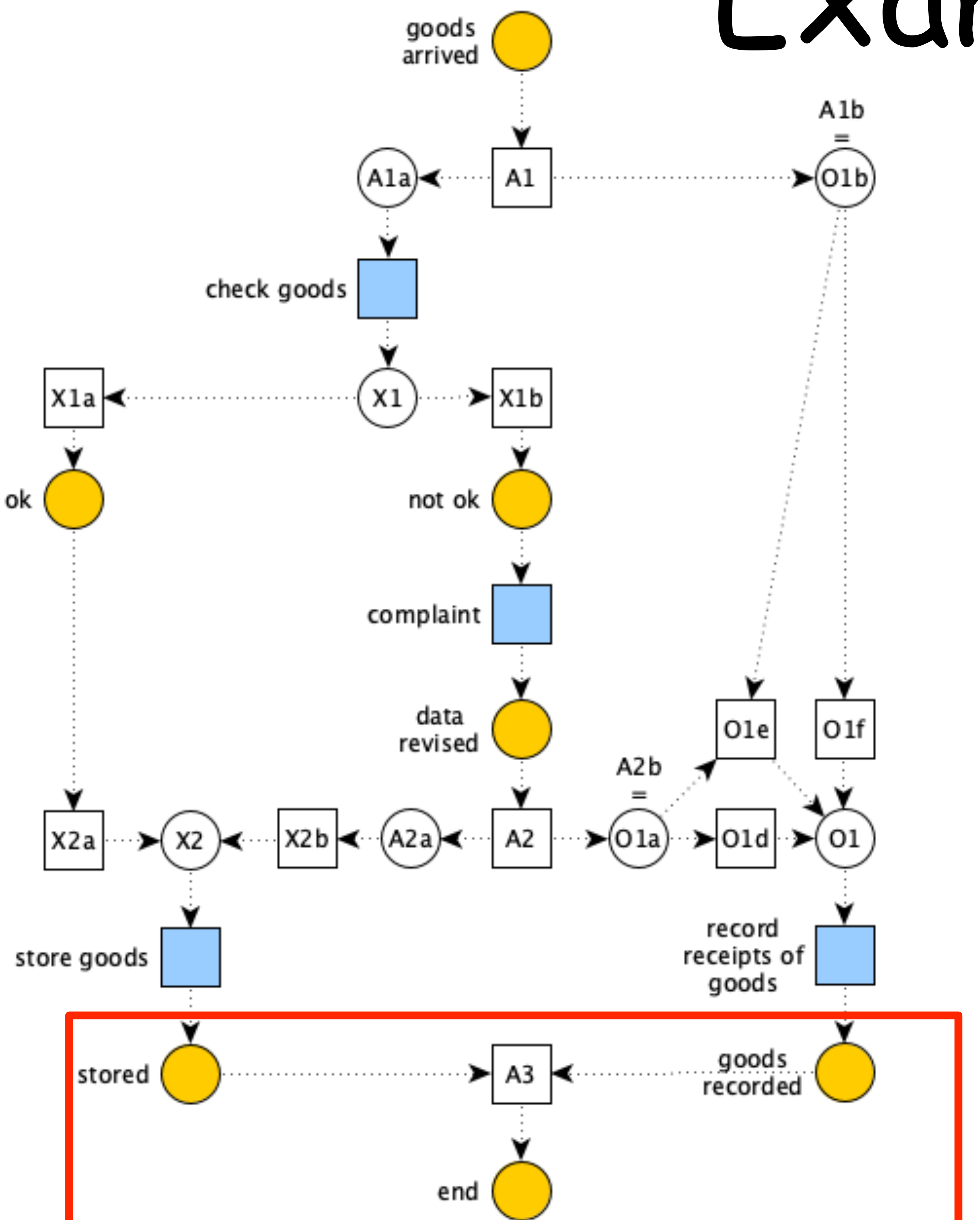
Example



Step 3
unique end

implicit AND join (because of A2)

Example

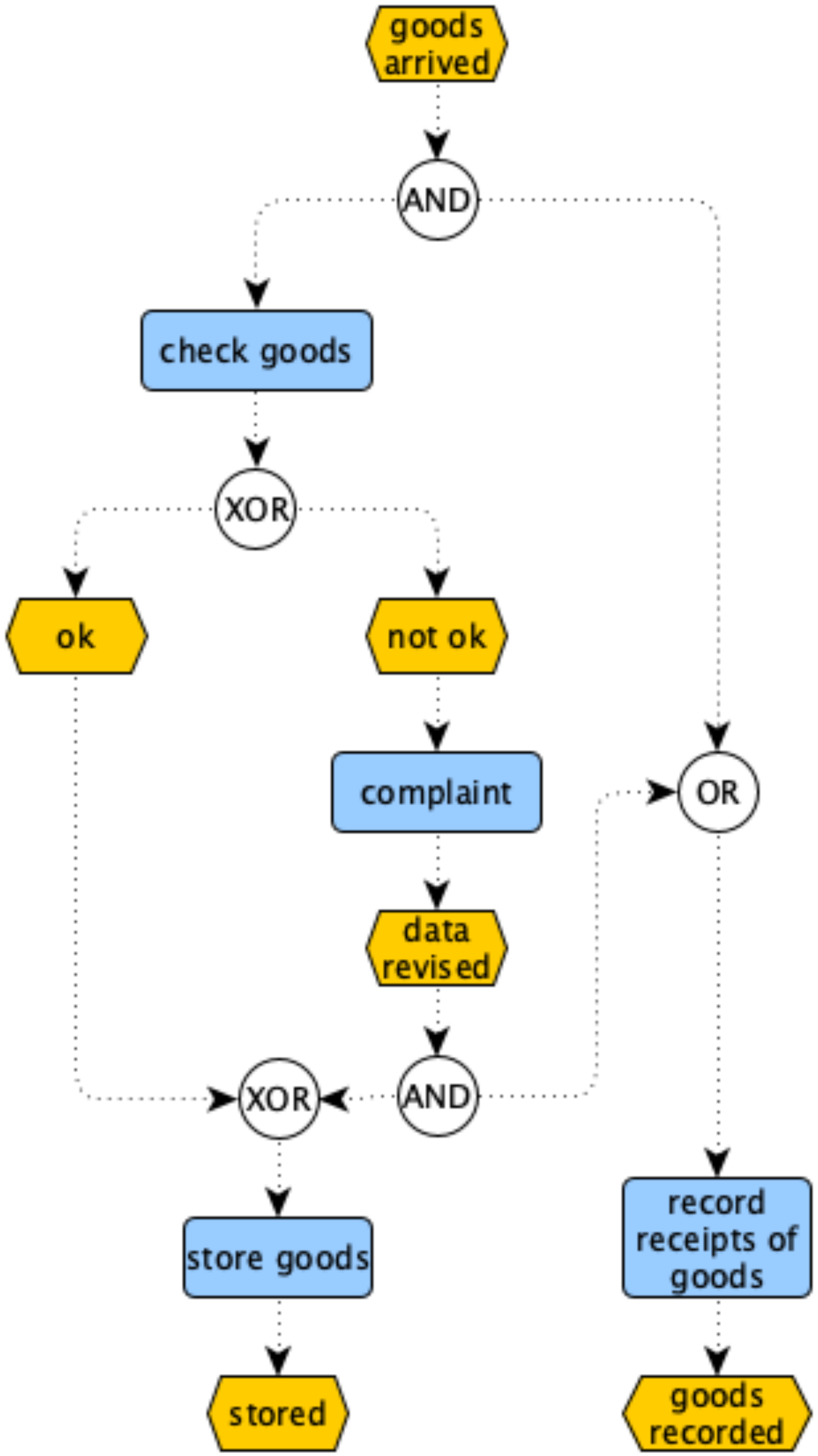


Step 3
unique end

implicit AND join (because of A2)

Example

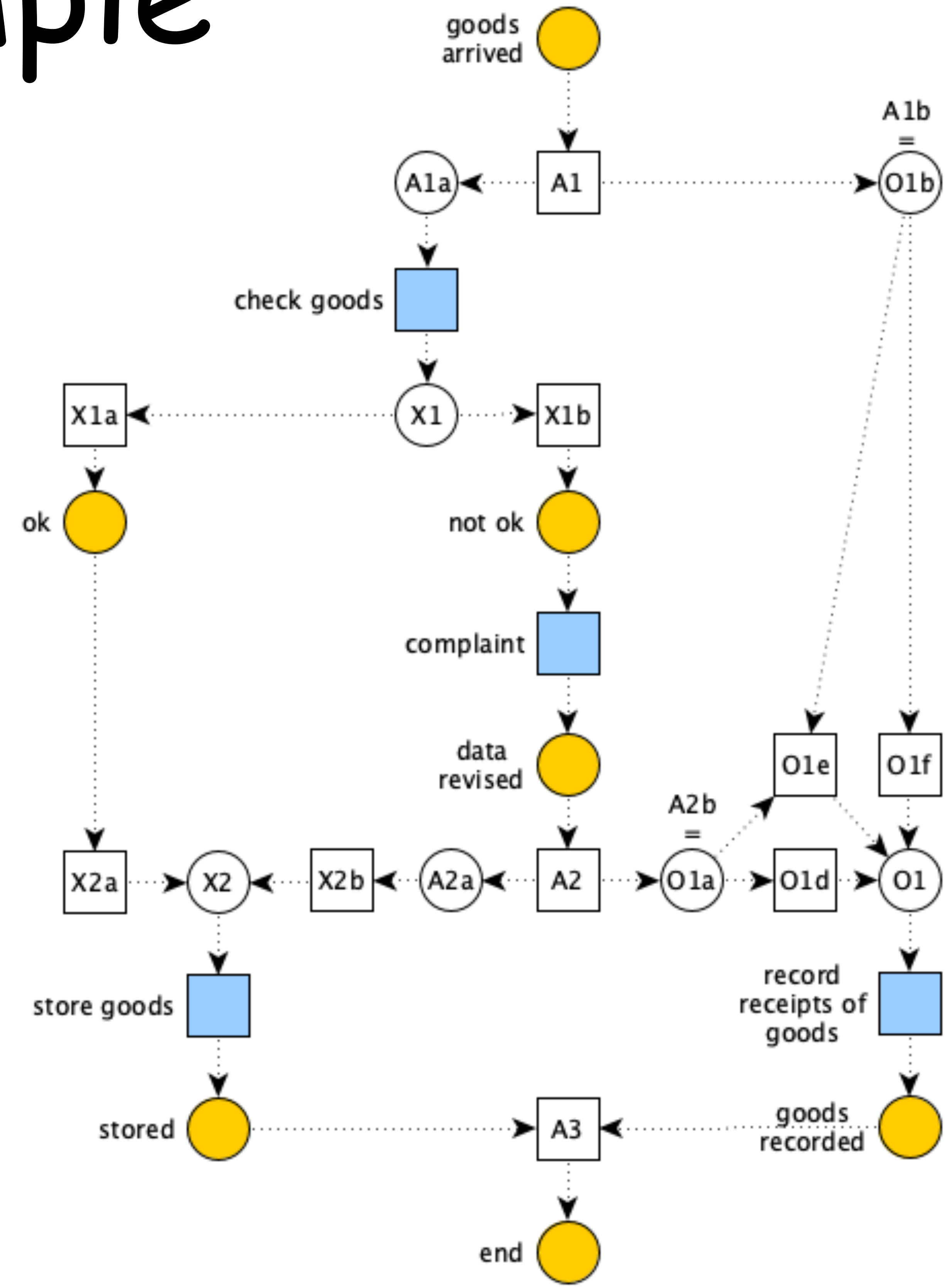
EPC



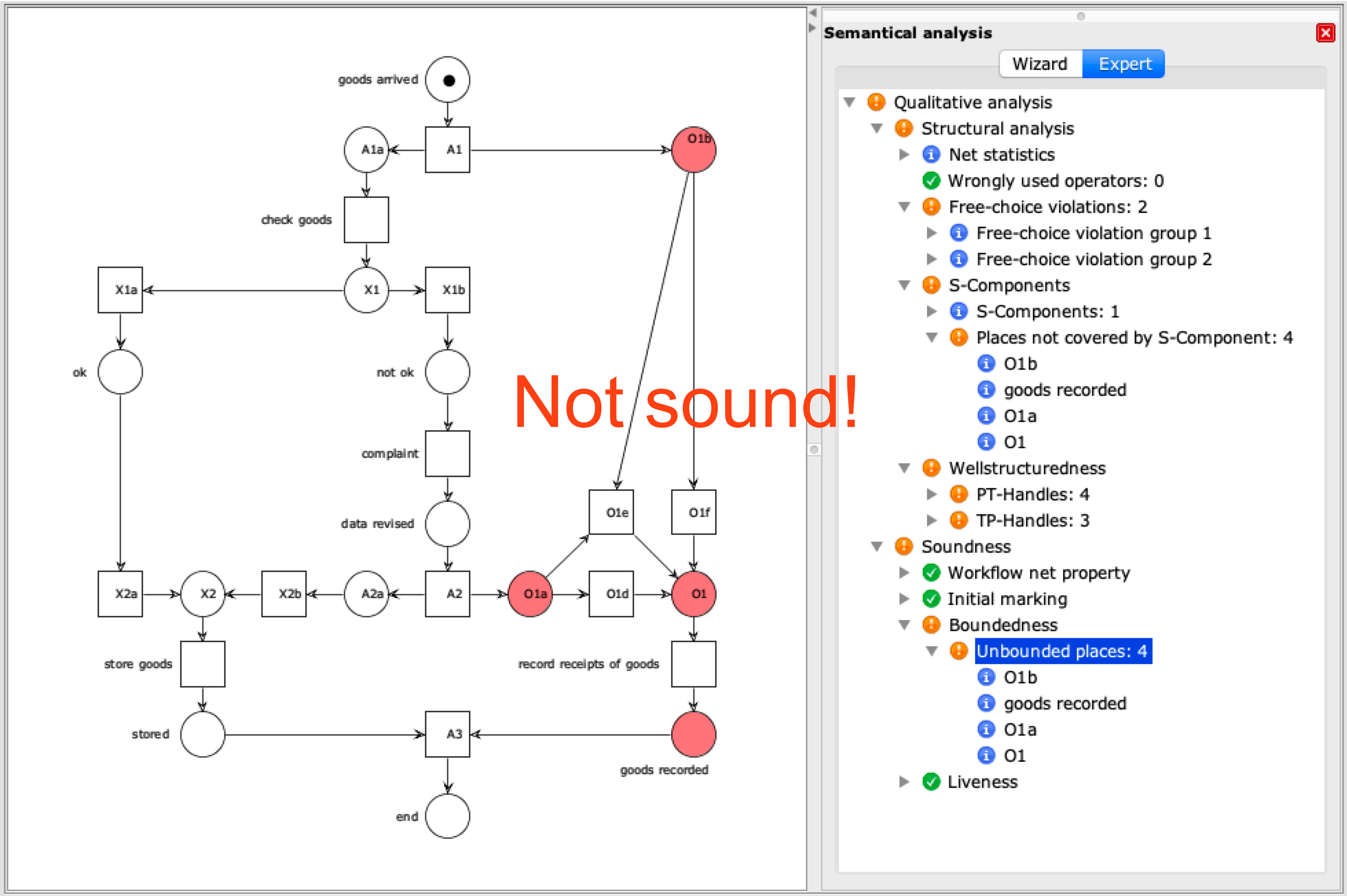
Sound?

Steps
1+2+3

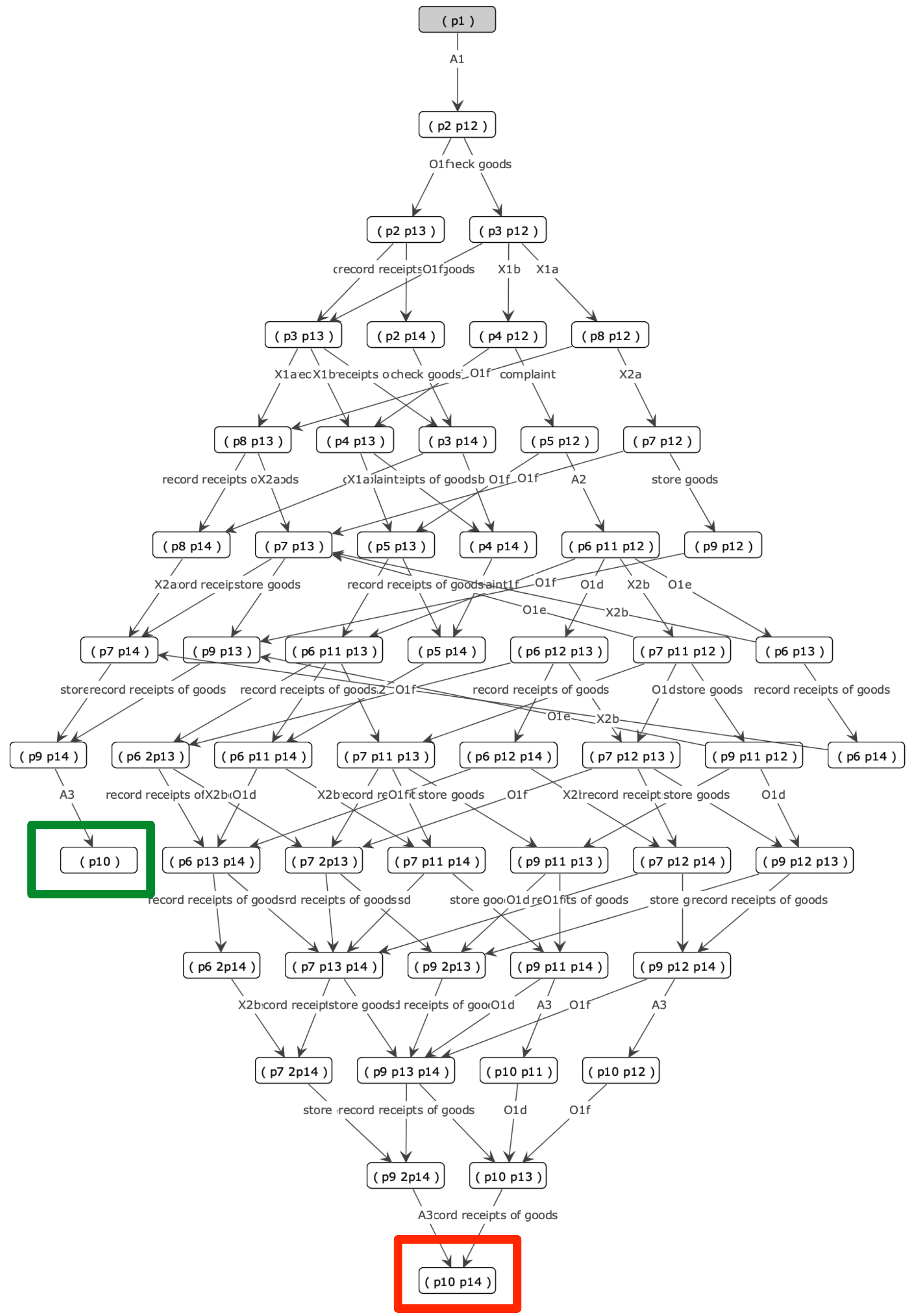
wf net



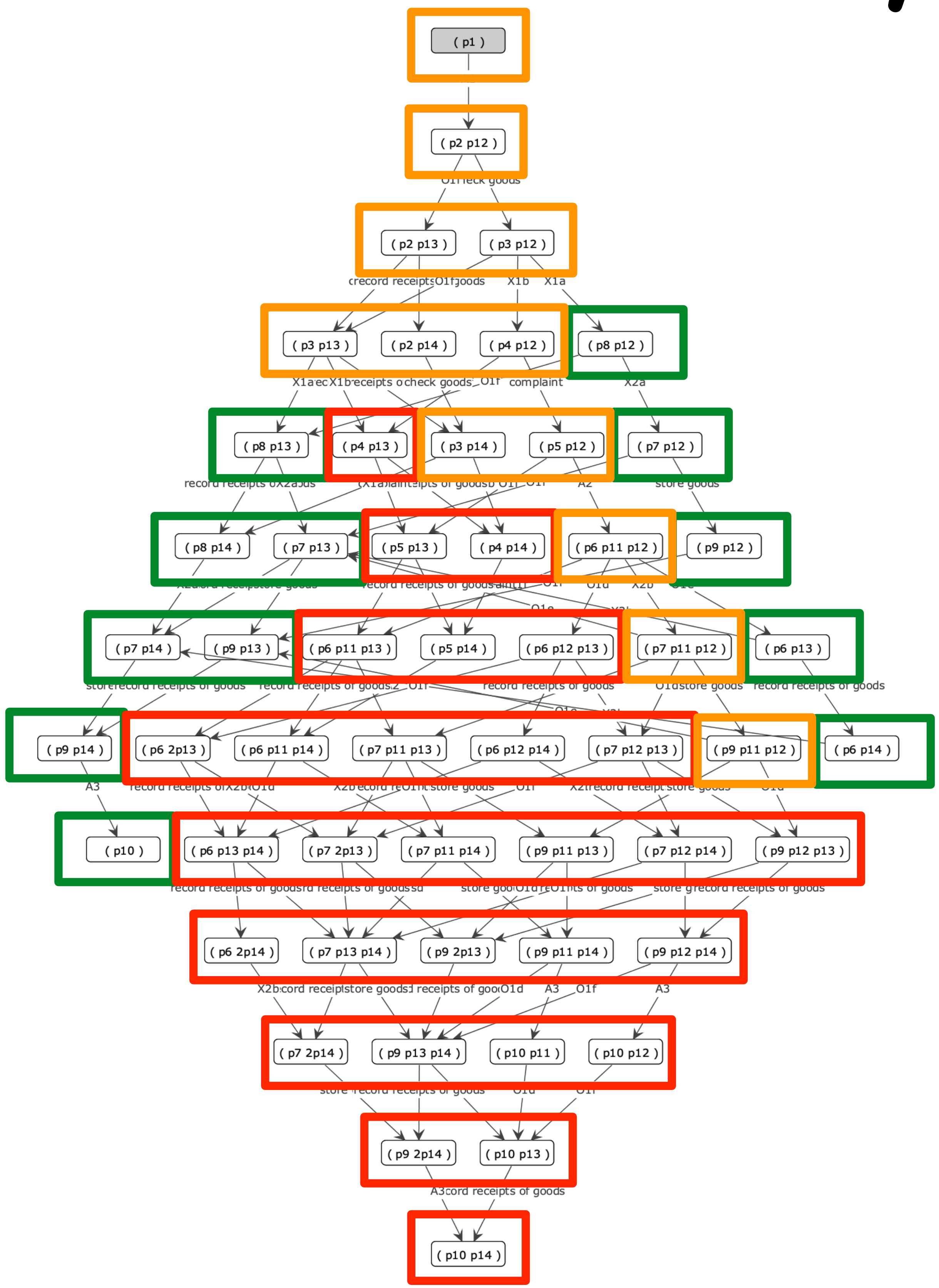
Soundness analysis



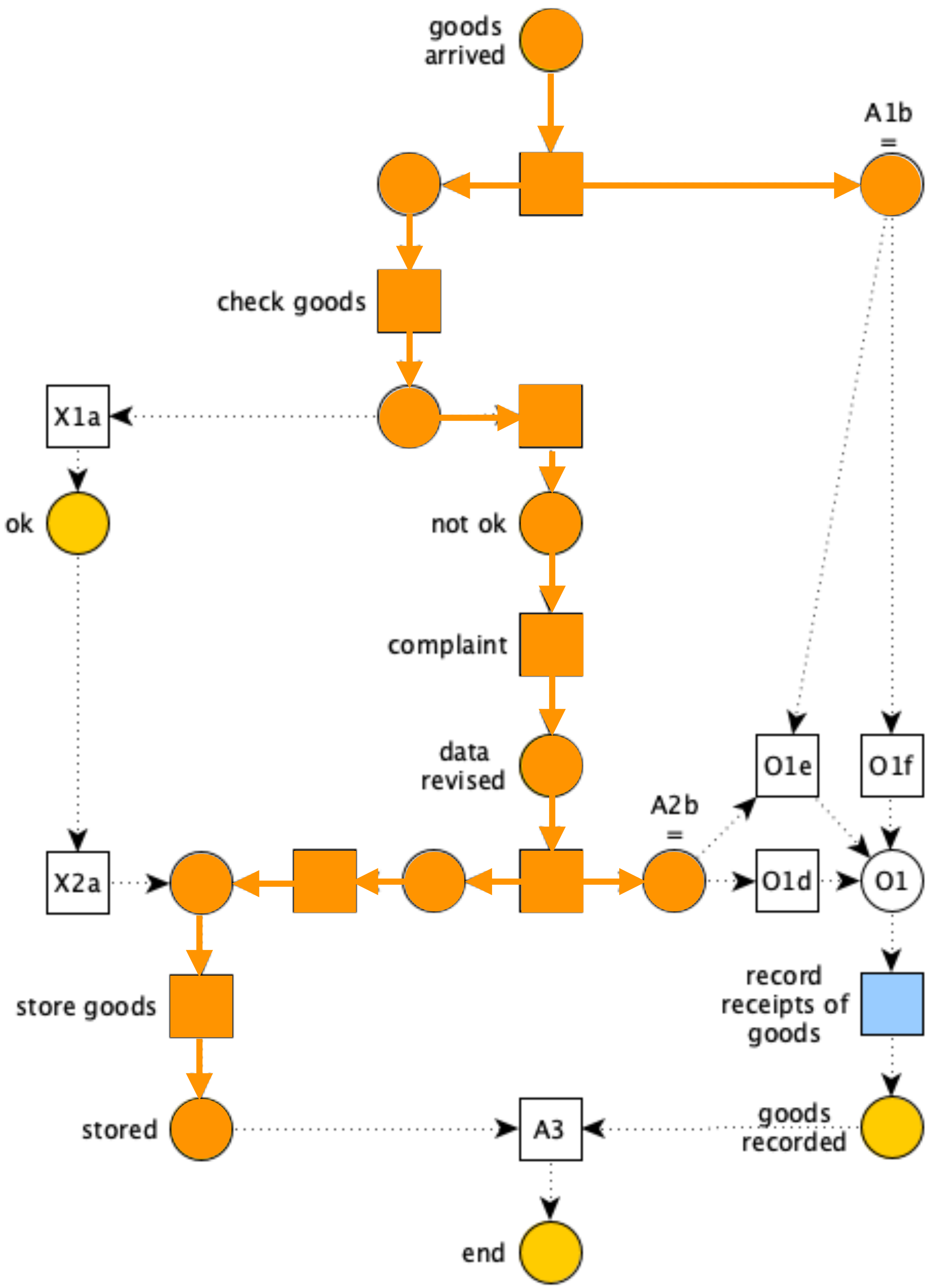
Soundness analysis



Soundness analysis

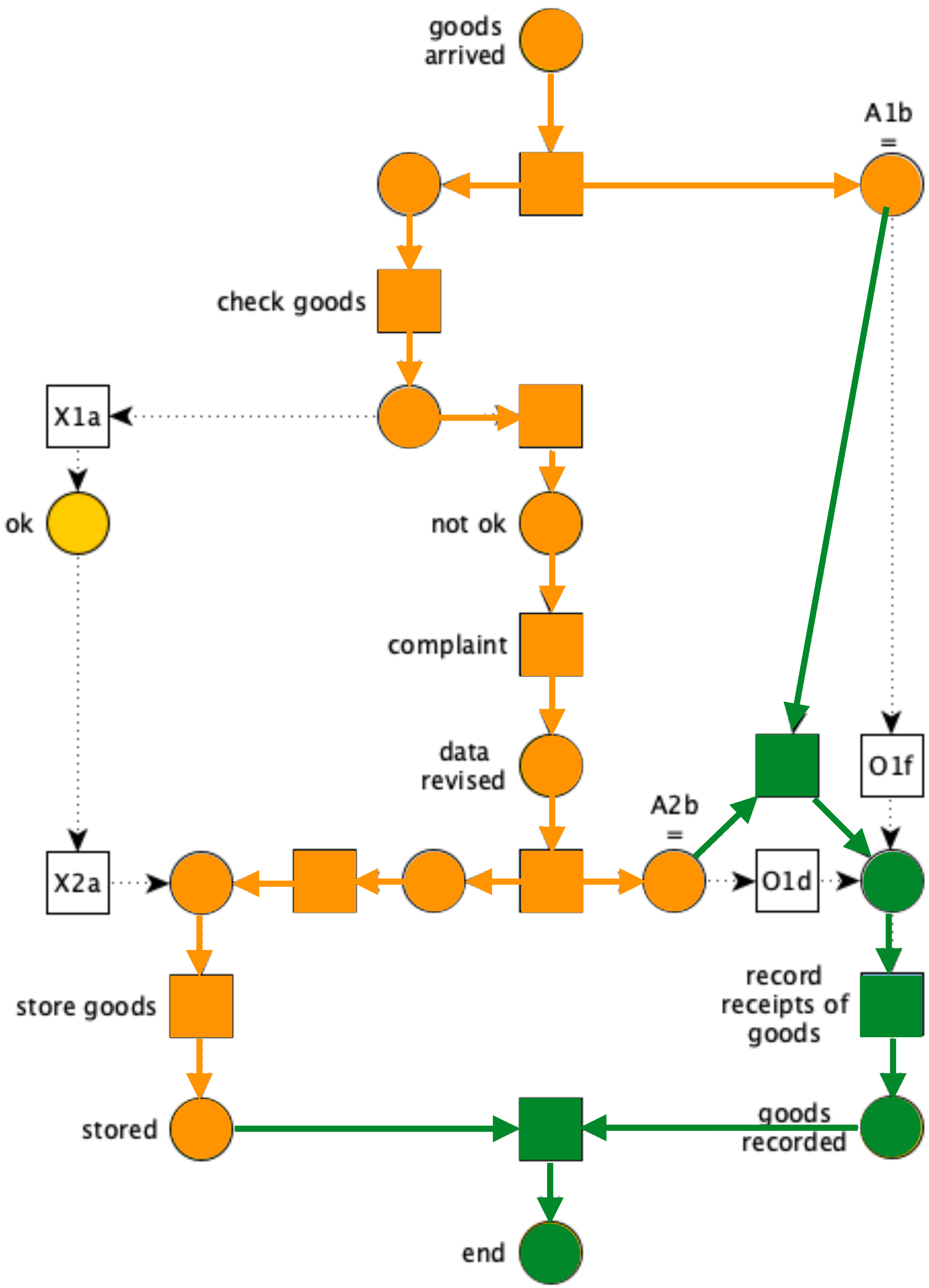


Soundness analysis



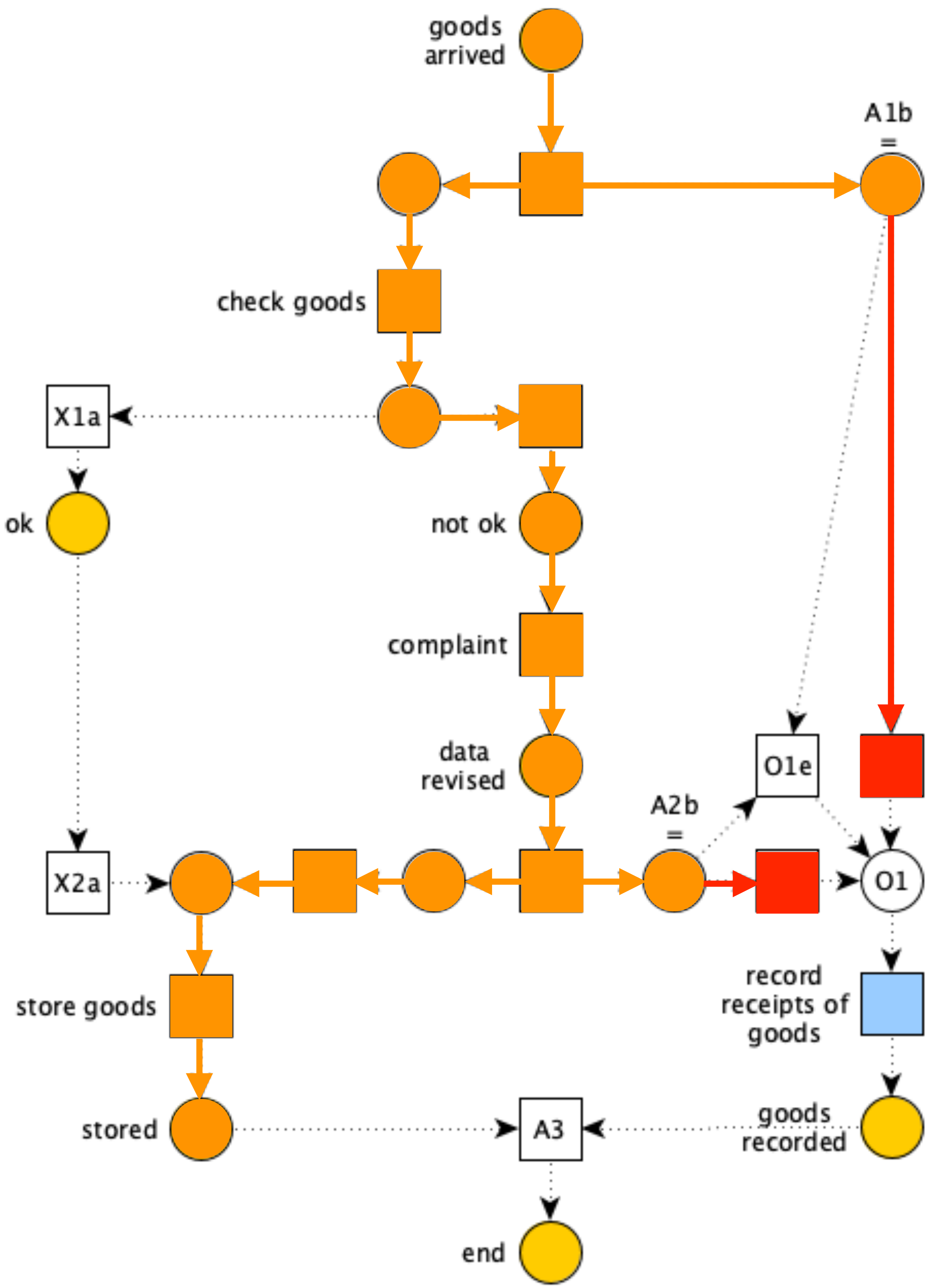
the right thing to do would be to fire O1e

Soundness analysis



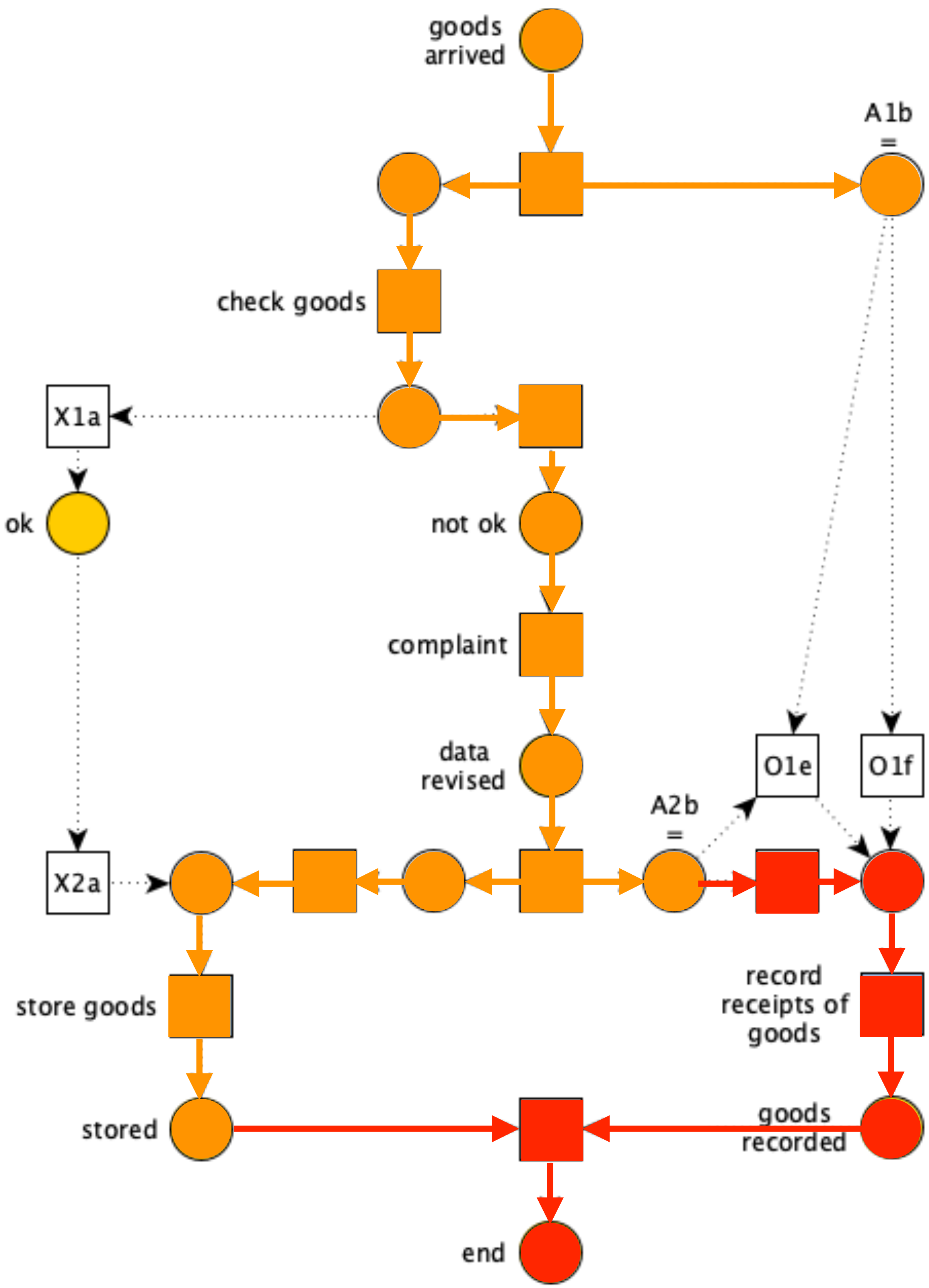
the right thing to do would be to fire O1e

Soundness analysis



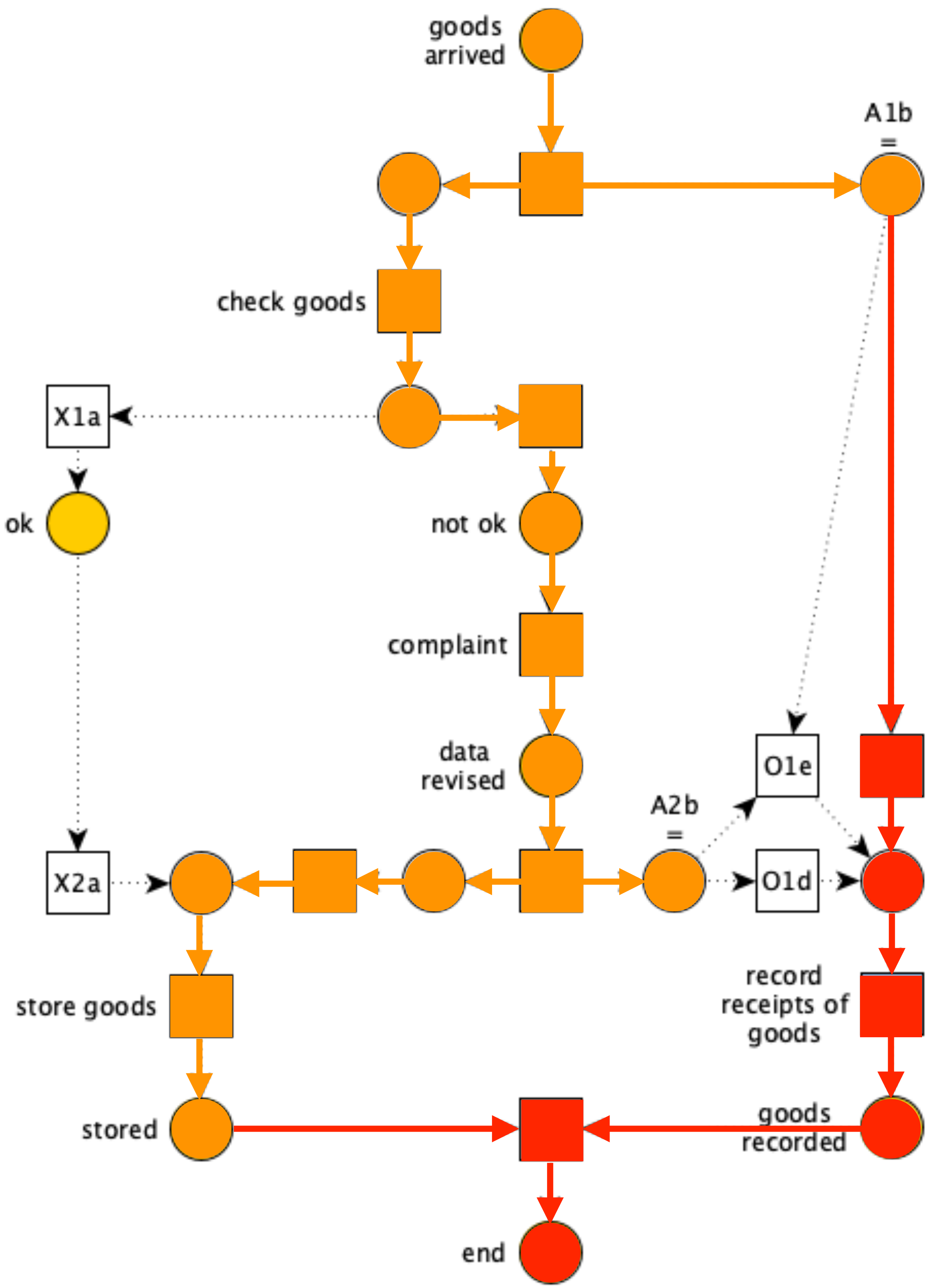
but O1f and O1d are enabled as well (OR semantics!)

Soundness analysis



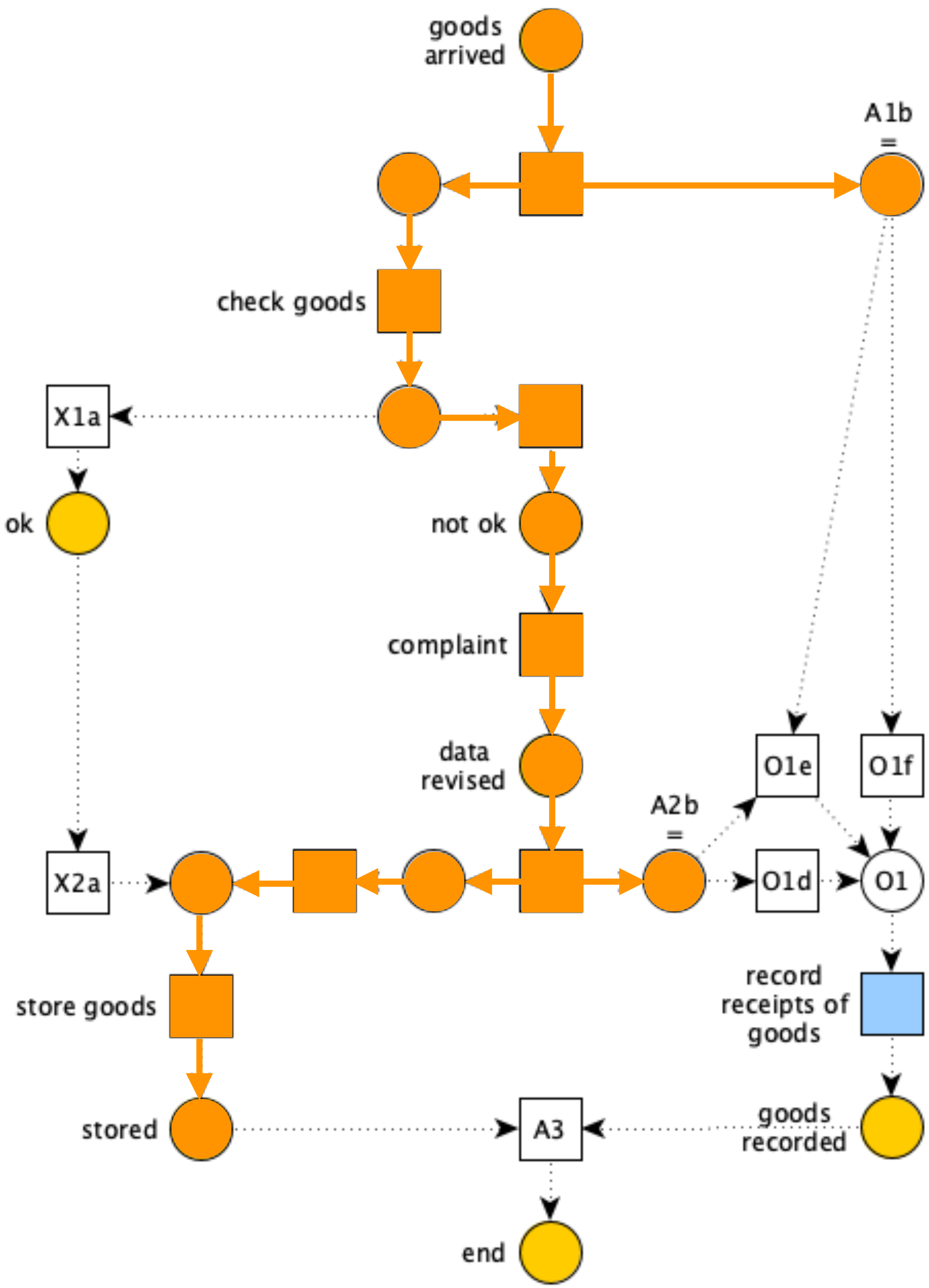
proper completion is not guaranteed (N* unbounded)

Soundness analysis



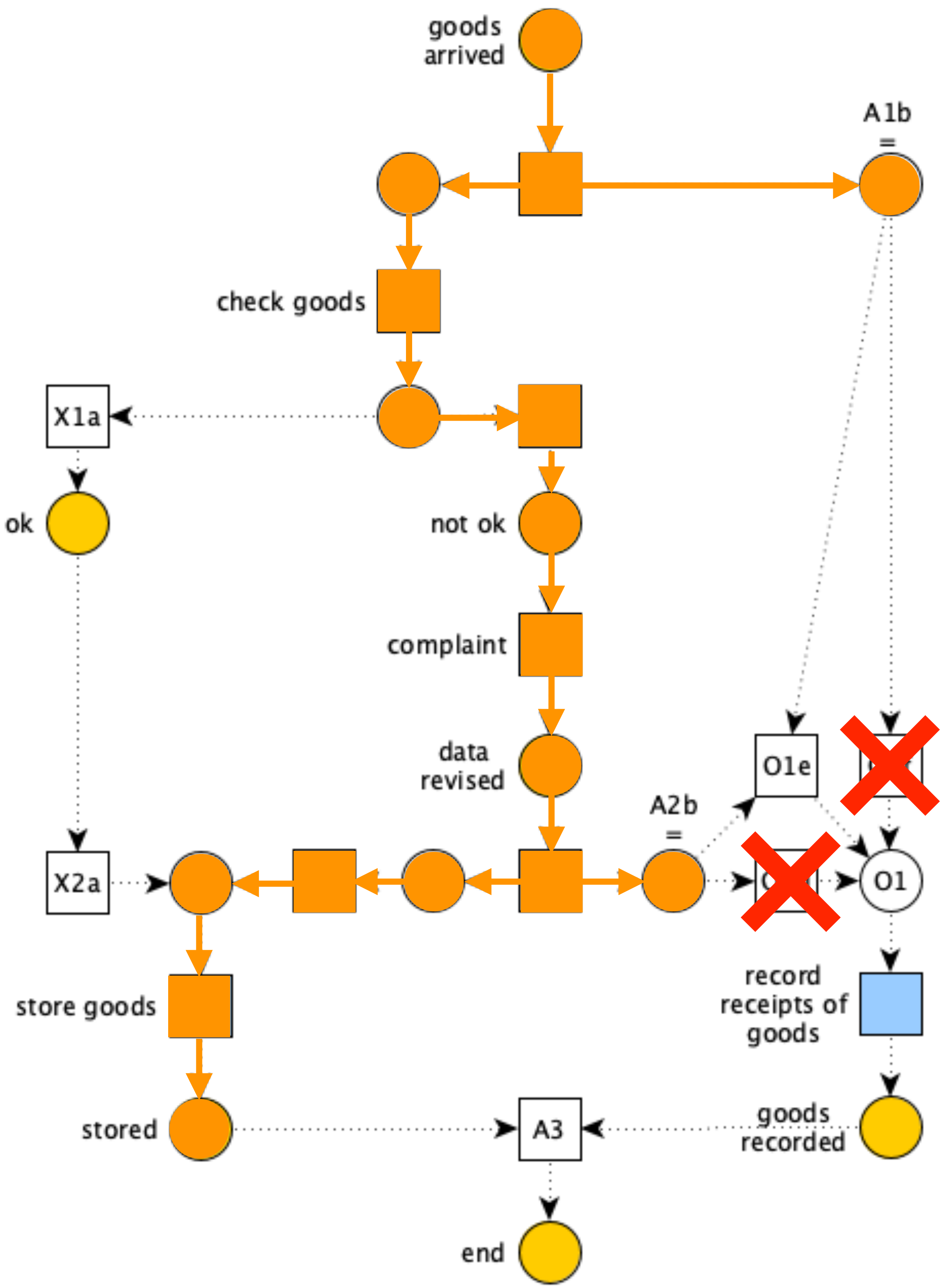
proper completion
is not guaranteed
(N* unbounded)

Soundness analysis



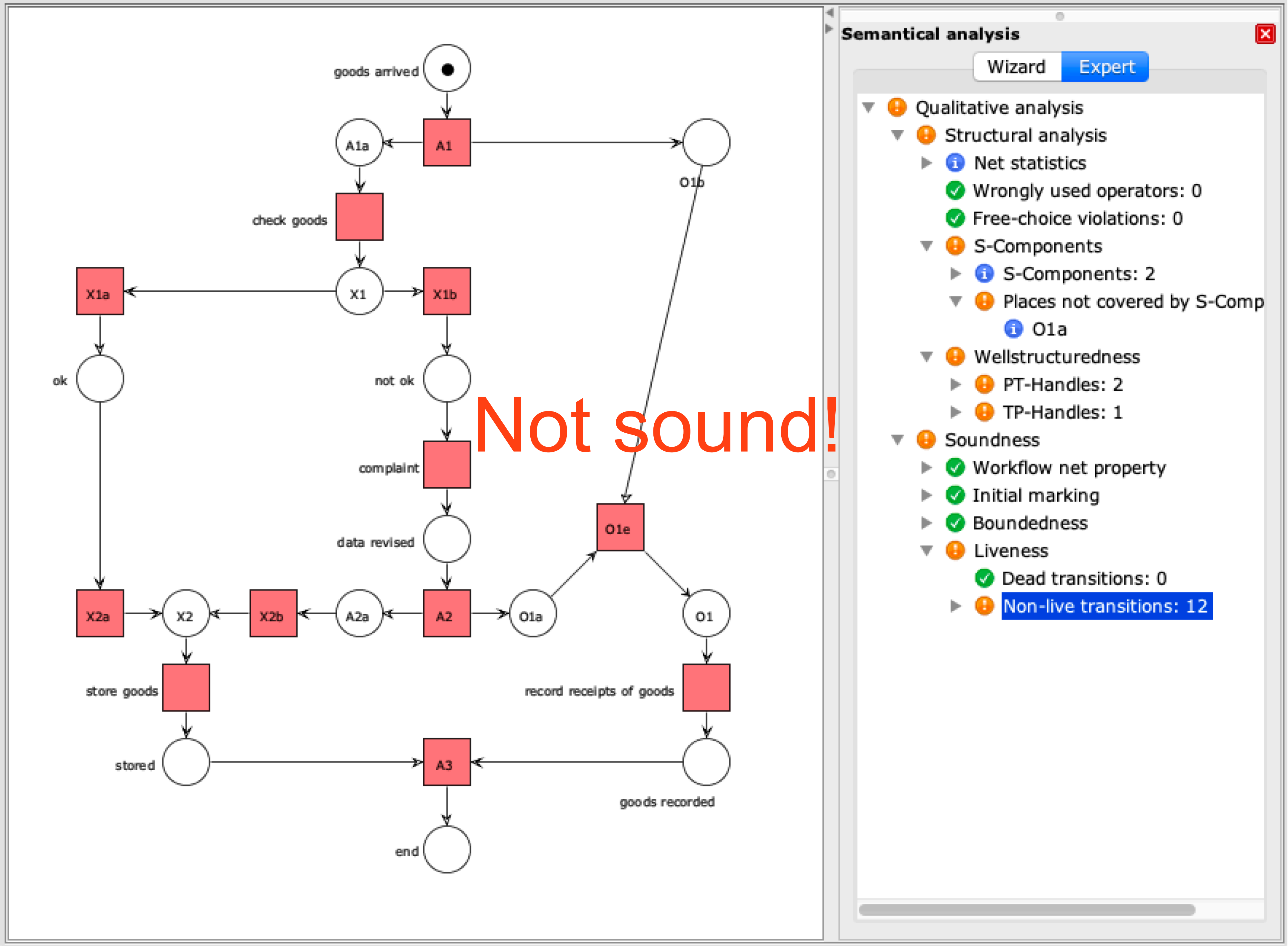
Can we repair the model?

Soundness analysis

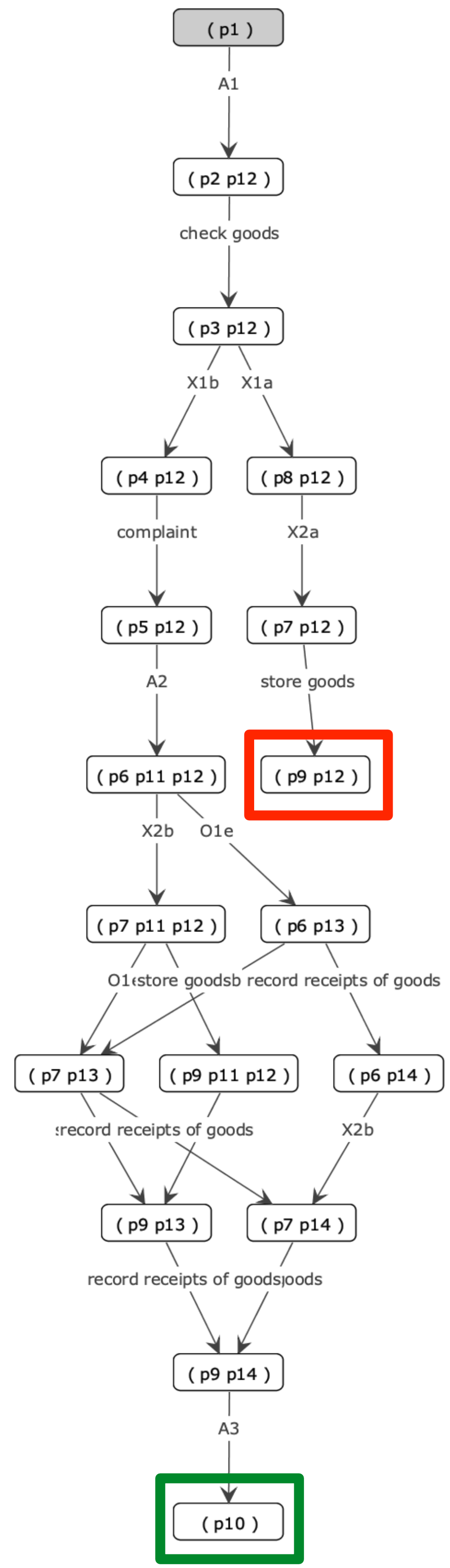


AND join instead of OR join?

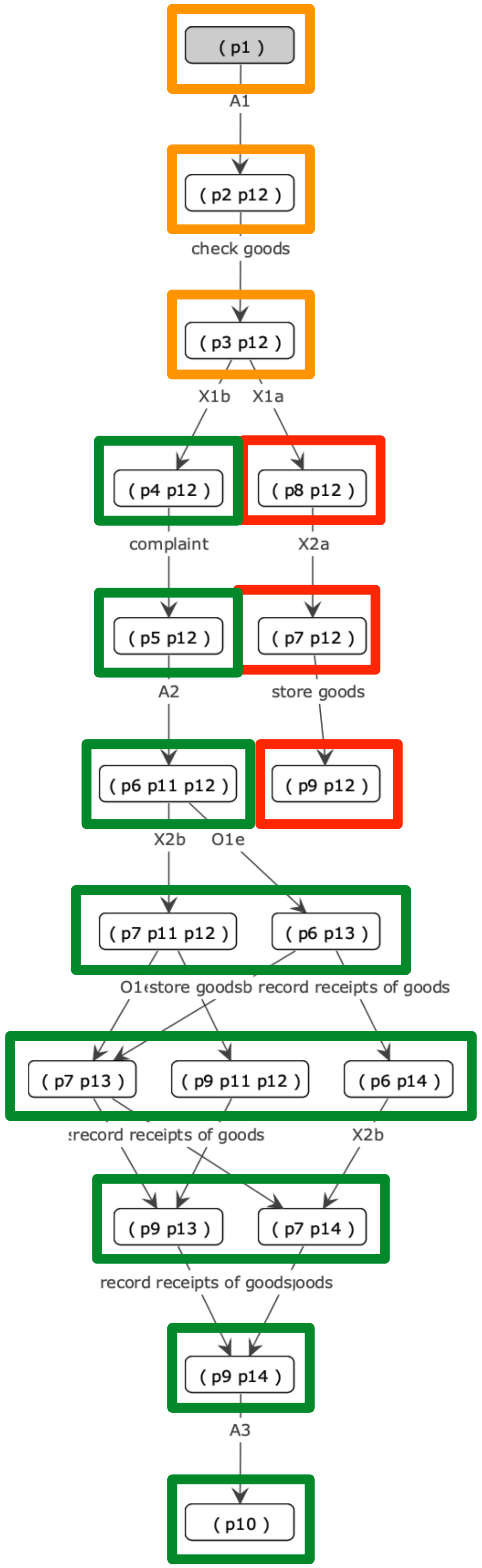
Soundness analysis



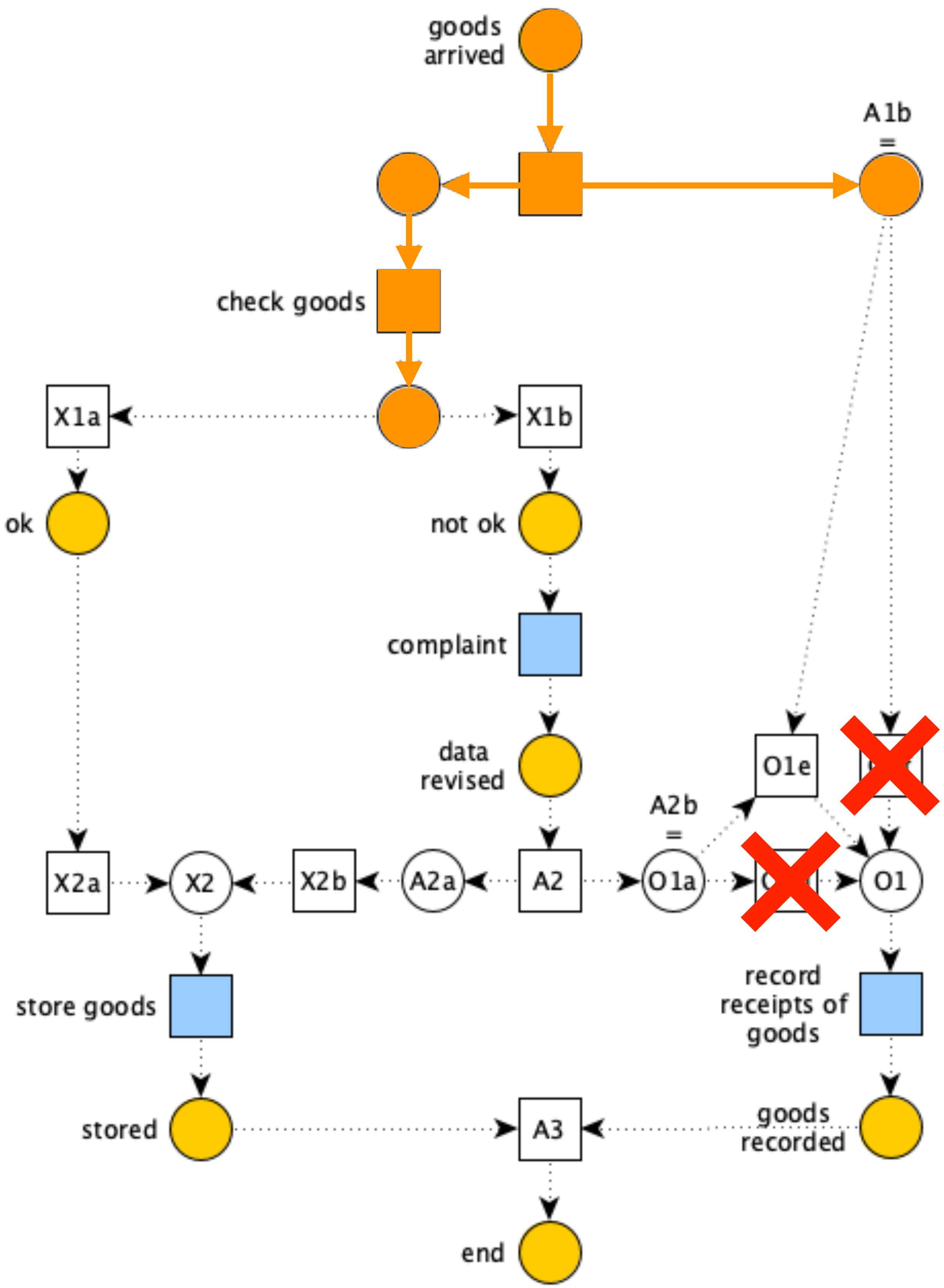
Soundness analysis



Soundness analysis



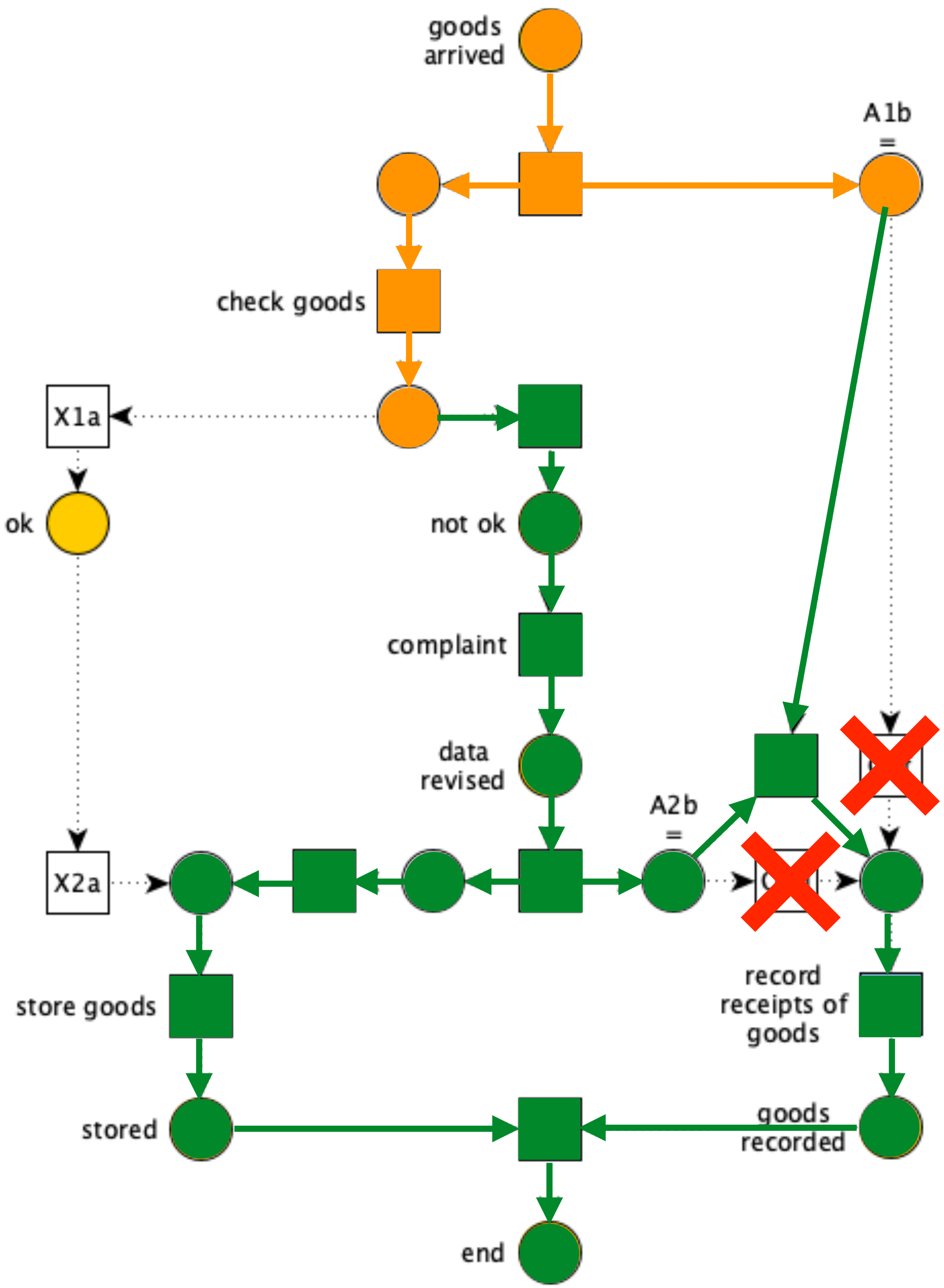
Soundness analysis



the right thing to do would be to fire X1b

AND join instead of OR join?

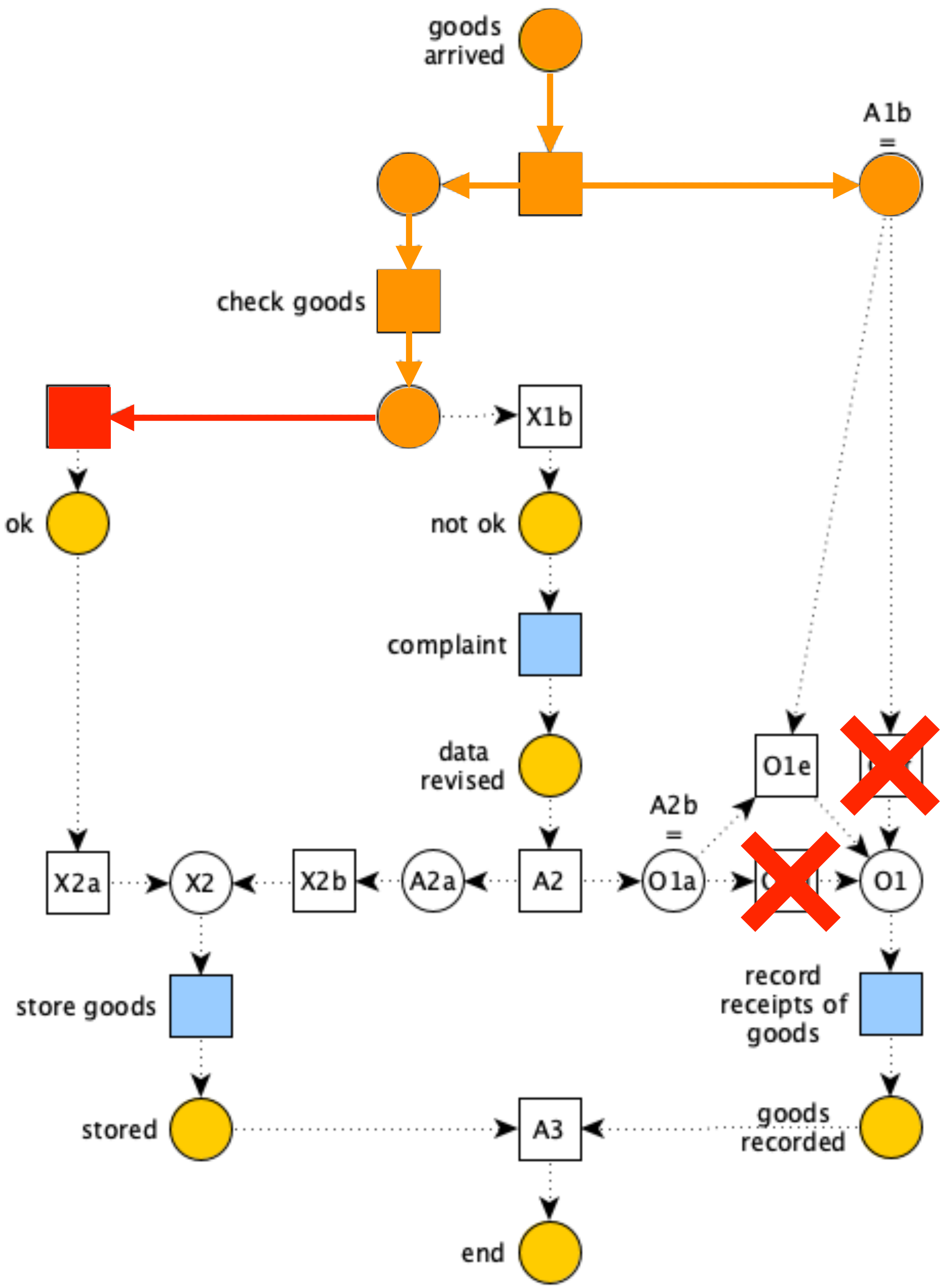
Soundness analysis



the right thing to do would be to fire X1b

AND join instead of OR join?

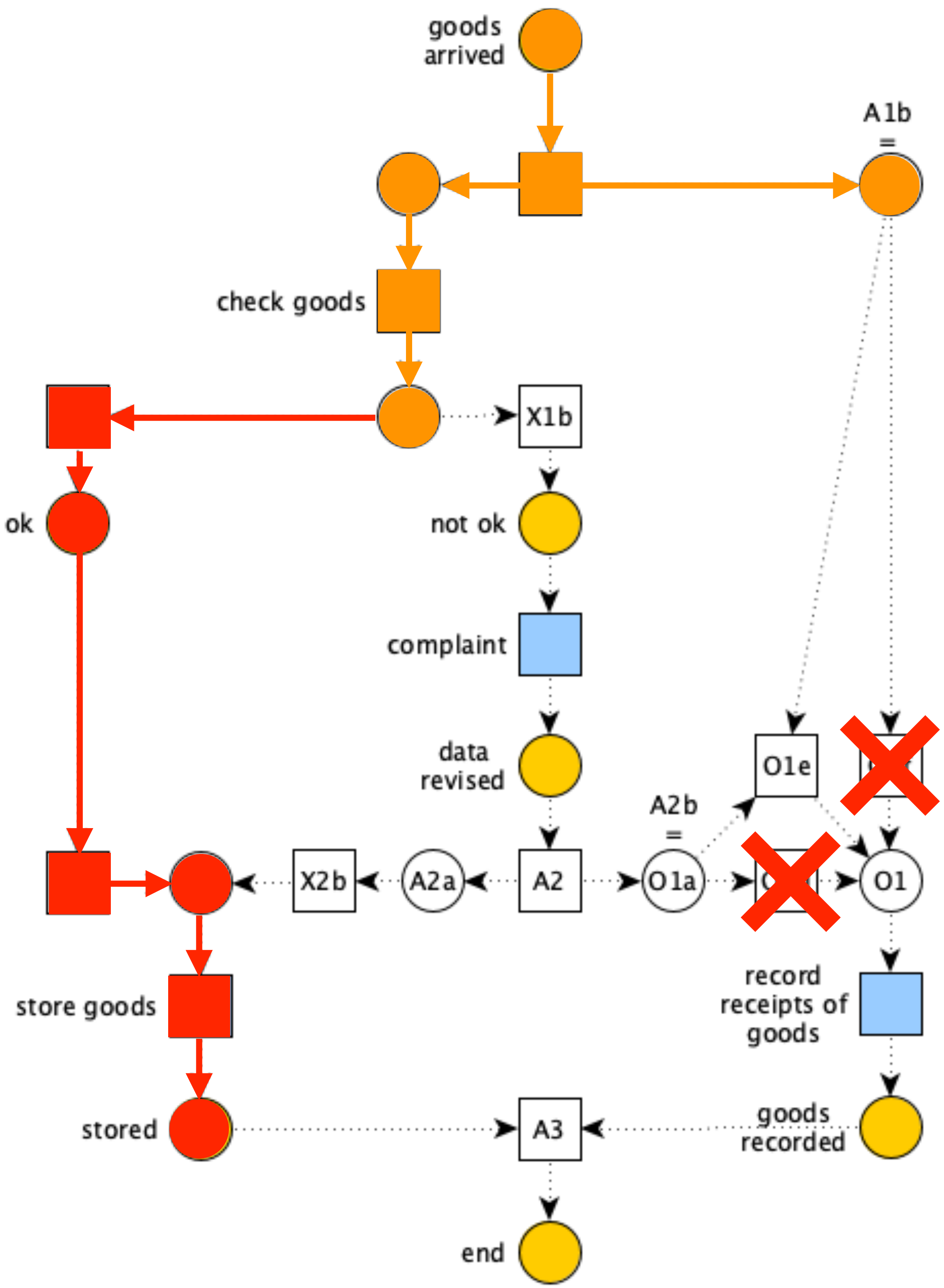
Soundness analysis



but X1a
is enabled as well

AND join
instead of
OR join?

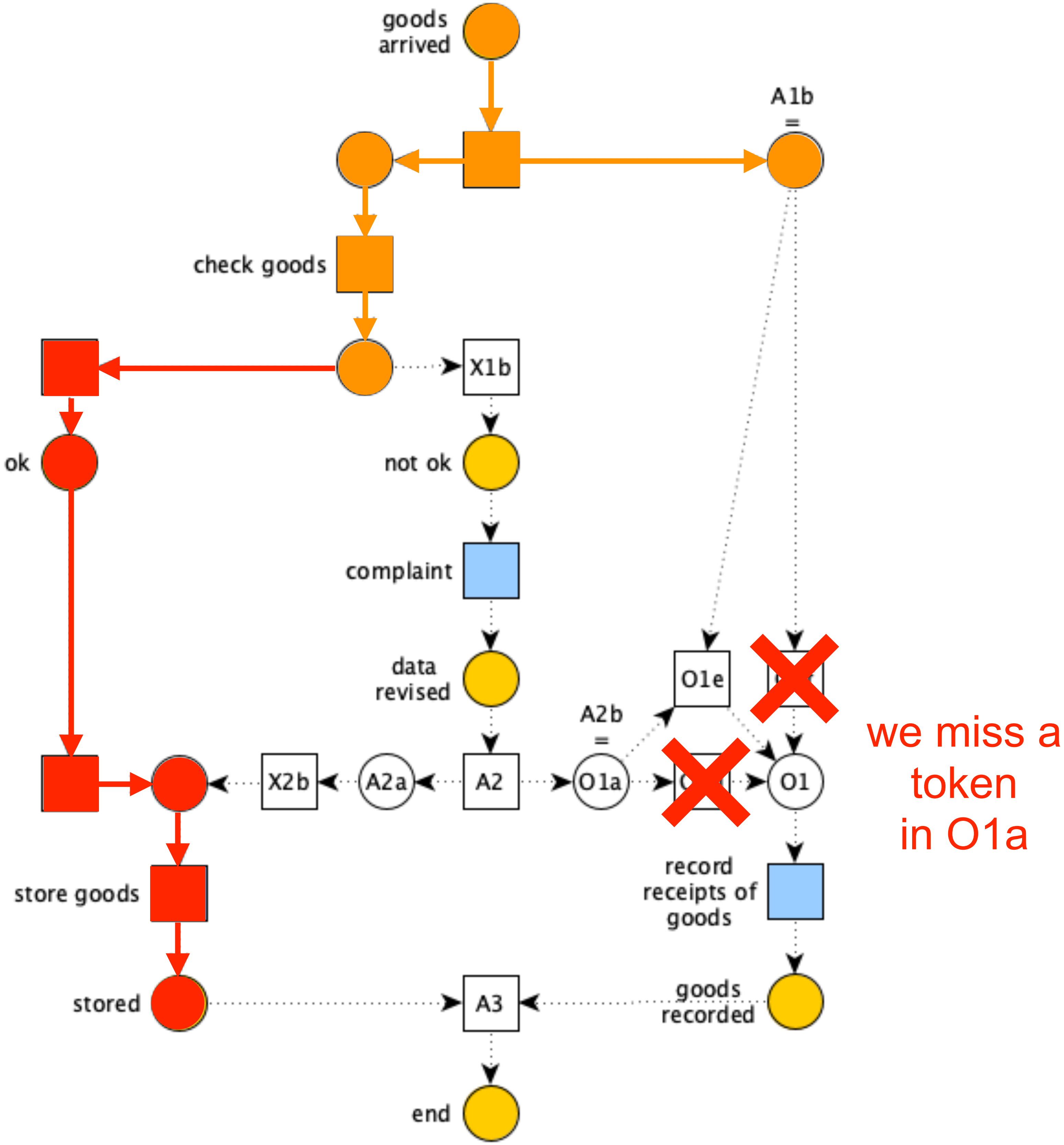
Soundness analysis



AND join instead of OR join?

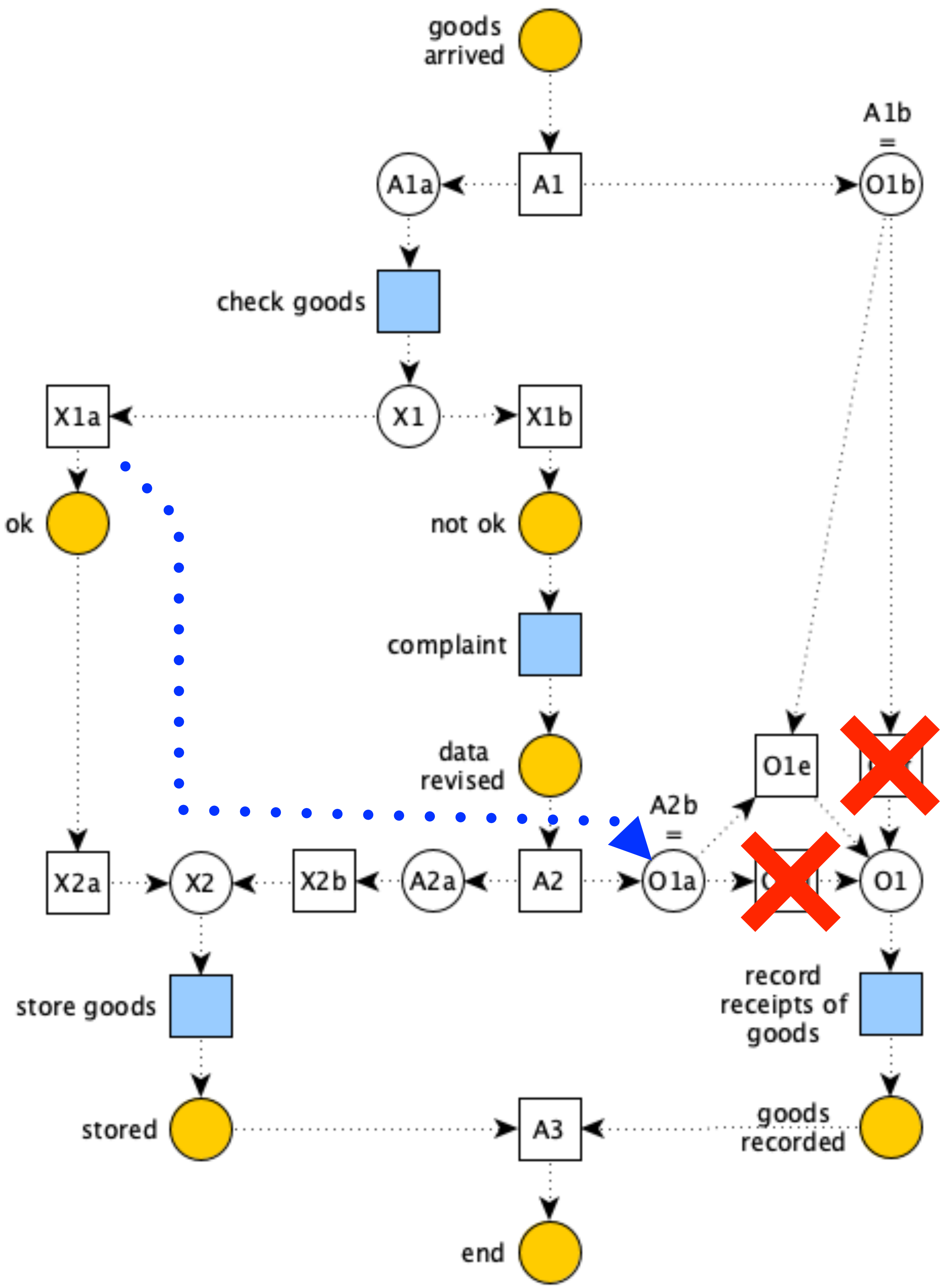
possible deadlock!
option to complete is not guaranteed
(N* non-live)

Soundness analysis



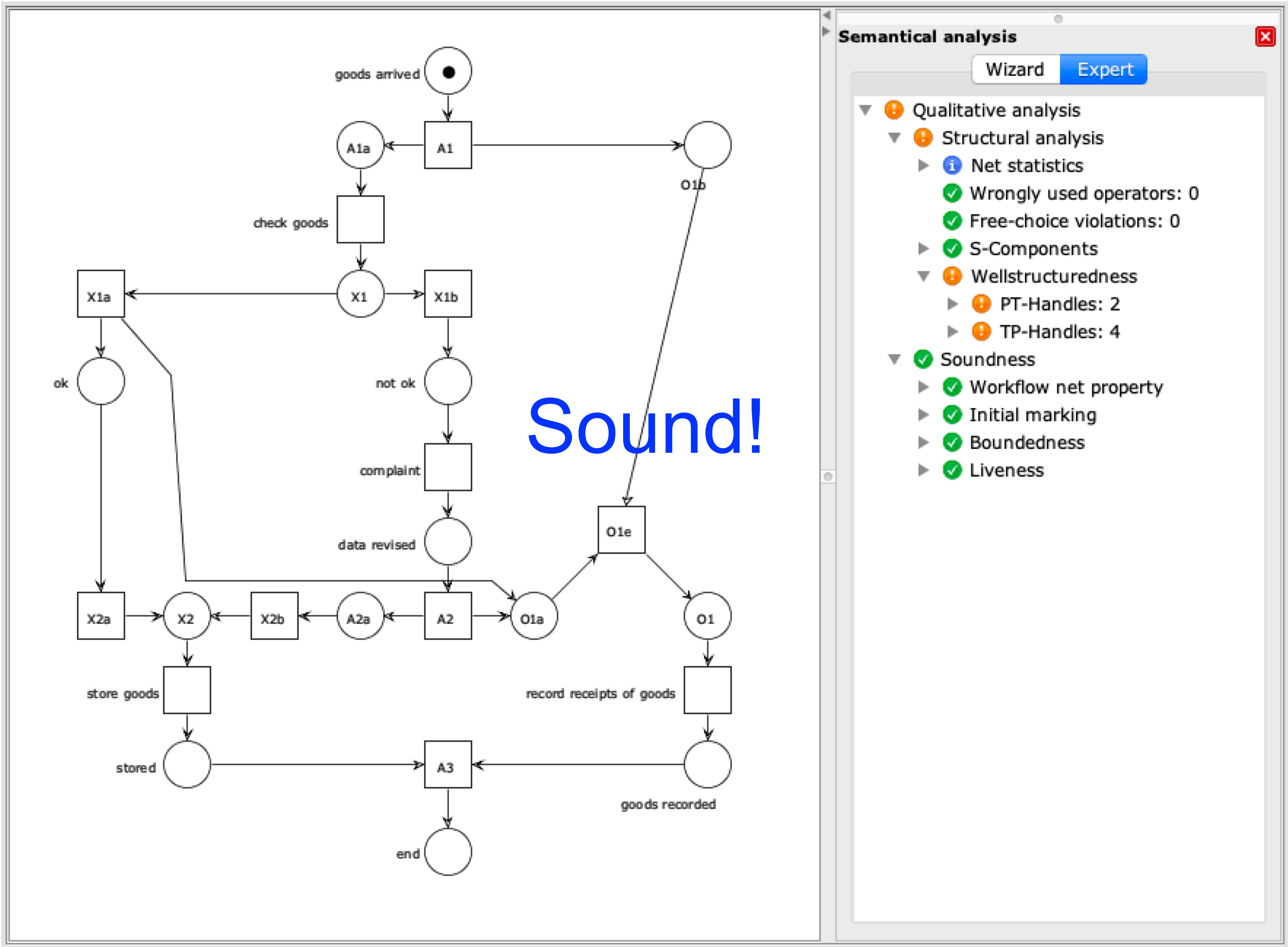
AND join instead of OR join + ad hoc flow?

Soundness analysis

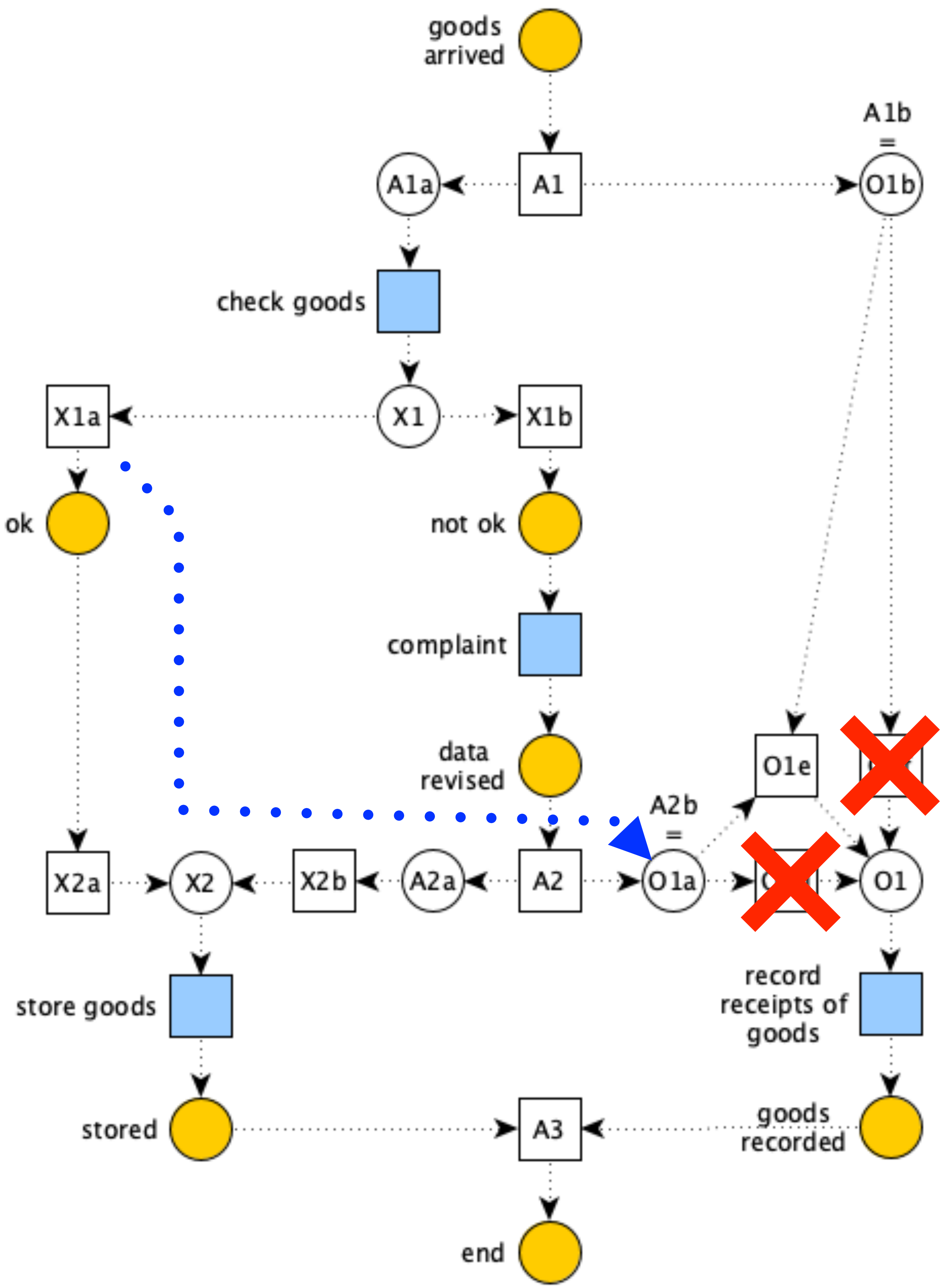


AND join
instead of
OR join
+ ad hoc flow?

Soundness analysis



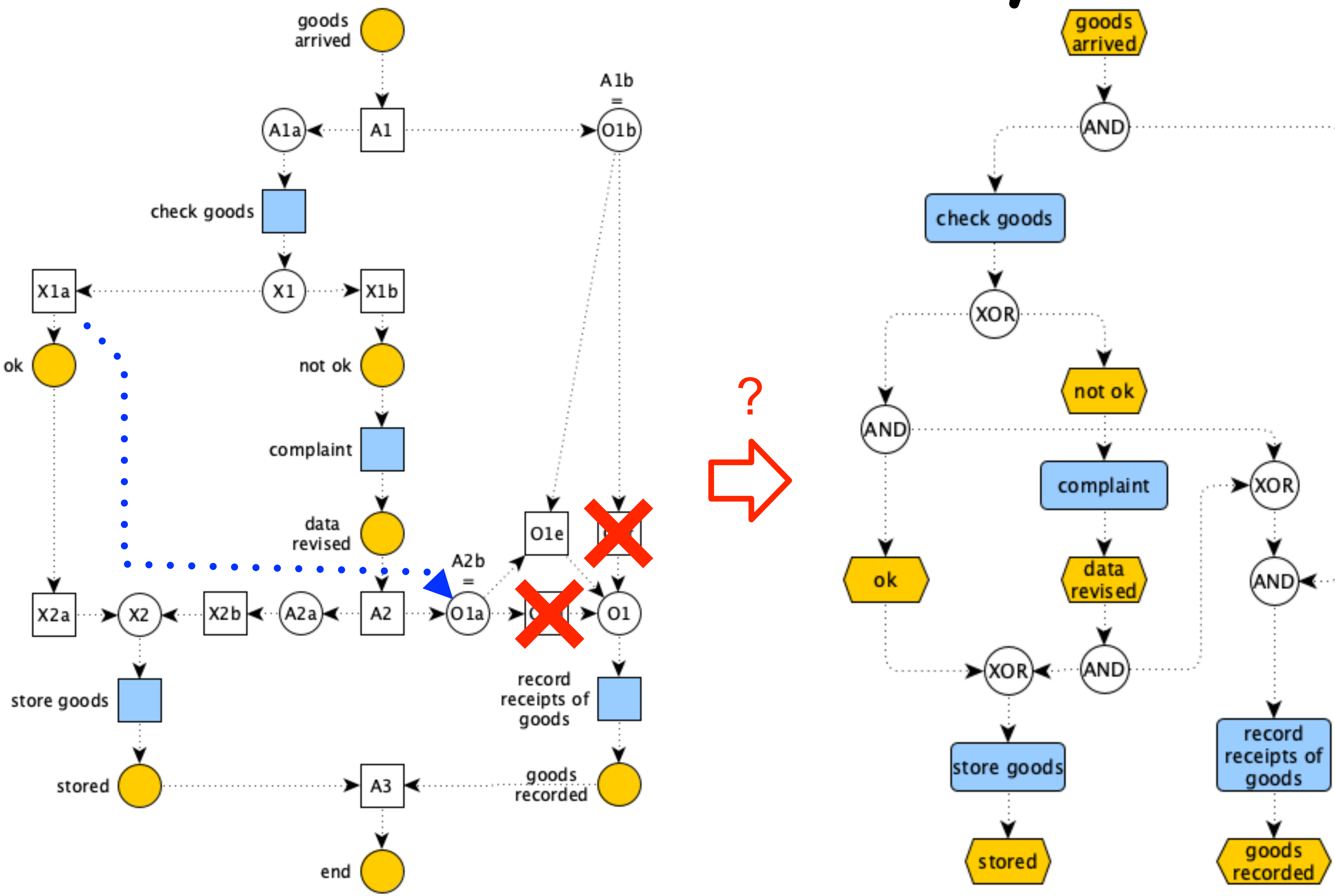
Soundness analysis



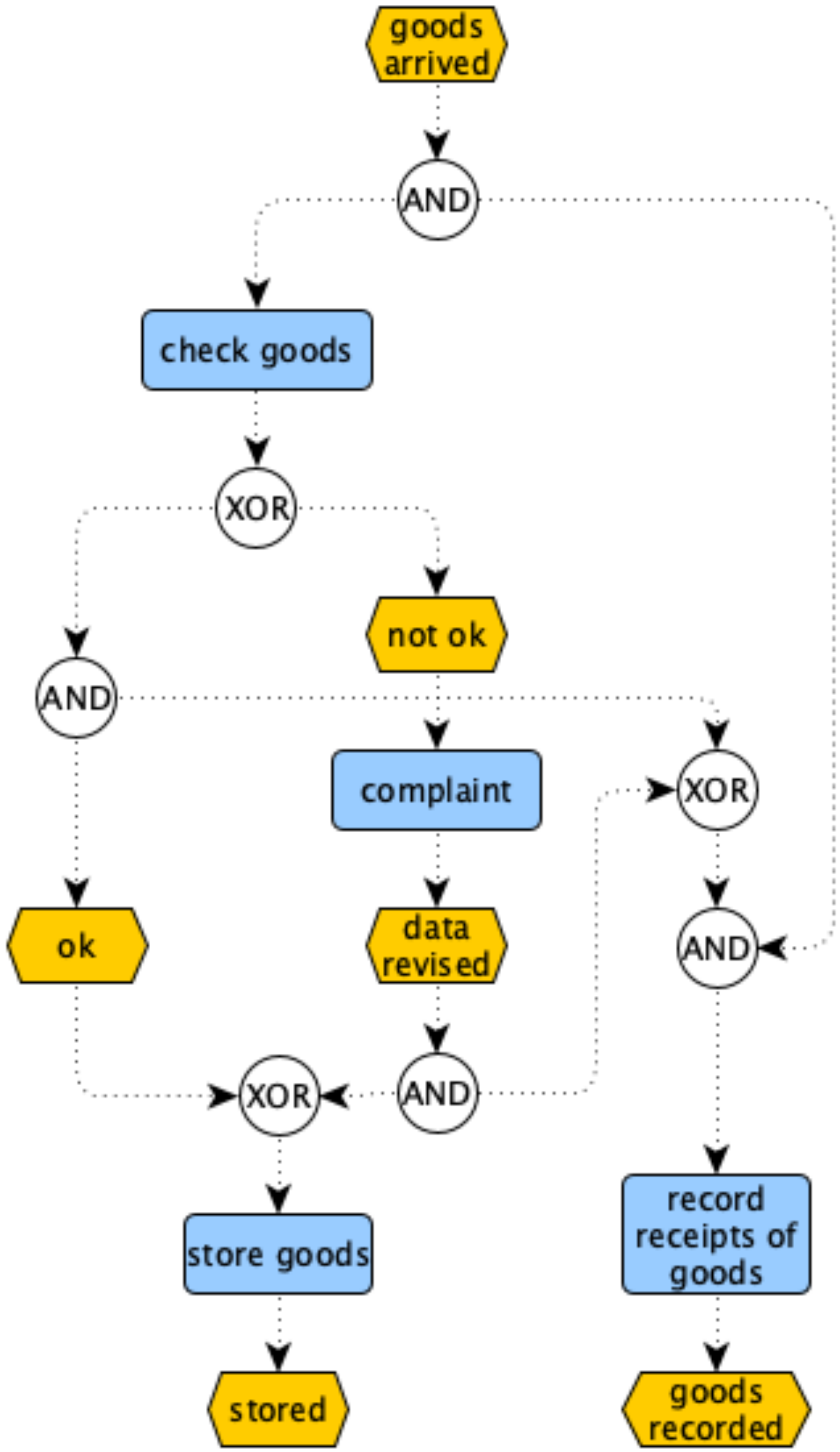
Sound, but…

we have repaired the wf net,
not the original EPC diagram!

Soundness analysis



Soundness analysis



The diagram is now
more complex
and less readable
than the original one!

Are we sure that its translation is the same sound wf net that we have designed ad hoc?

Are we sure it is sound?

Need to restart the analysis!!

Relaxed Soundness (optional reading)

Problem

EPC is widely adopted
also at early stages of design

WF nets offer a useful tool

but

Soundness can be too demanding at early stages

(Un)sound behaviours

A **sound** behaviour:
we move from a start event to an end event
so that nothing blocks or remains undone

The language of the net collects all and only its sound behaviours

$$L(N) = \{\sigma \mid i \xrightarrow{\sigma} o\}$$

Execution paths leading to **unsound** behaviours can be used to infer potential mistakes

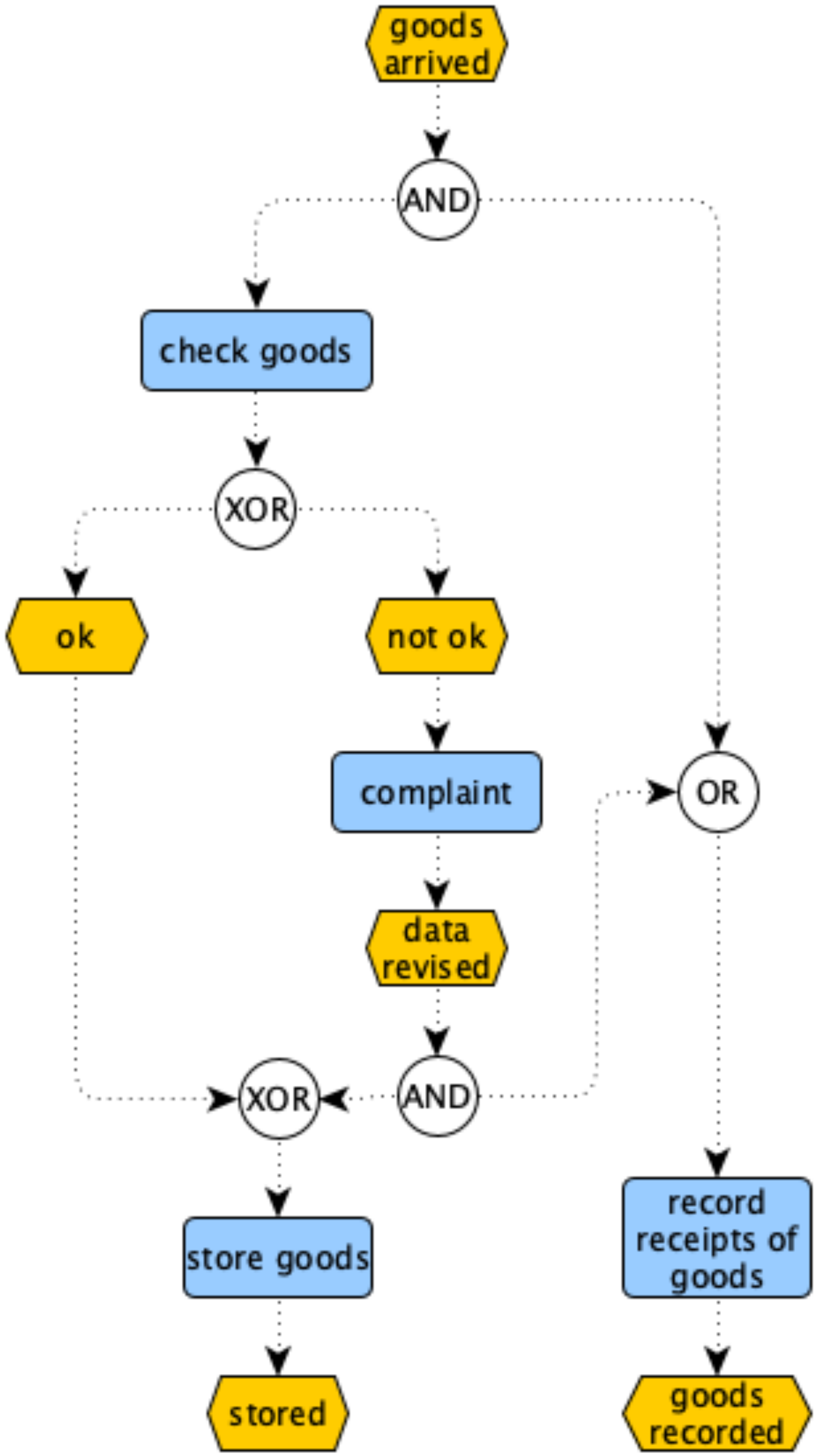
Relaxed soundness

If some unsound behaviour is possible
but any transition can take part to one sound execution,
then the process is called **relaxed sound**

Definition: A WF net is **relaxed sound** if
every transition belongs to a firing sequence
that starts in state i and ends in state o
(i.e. it appears in the language of the net)

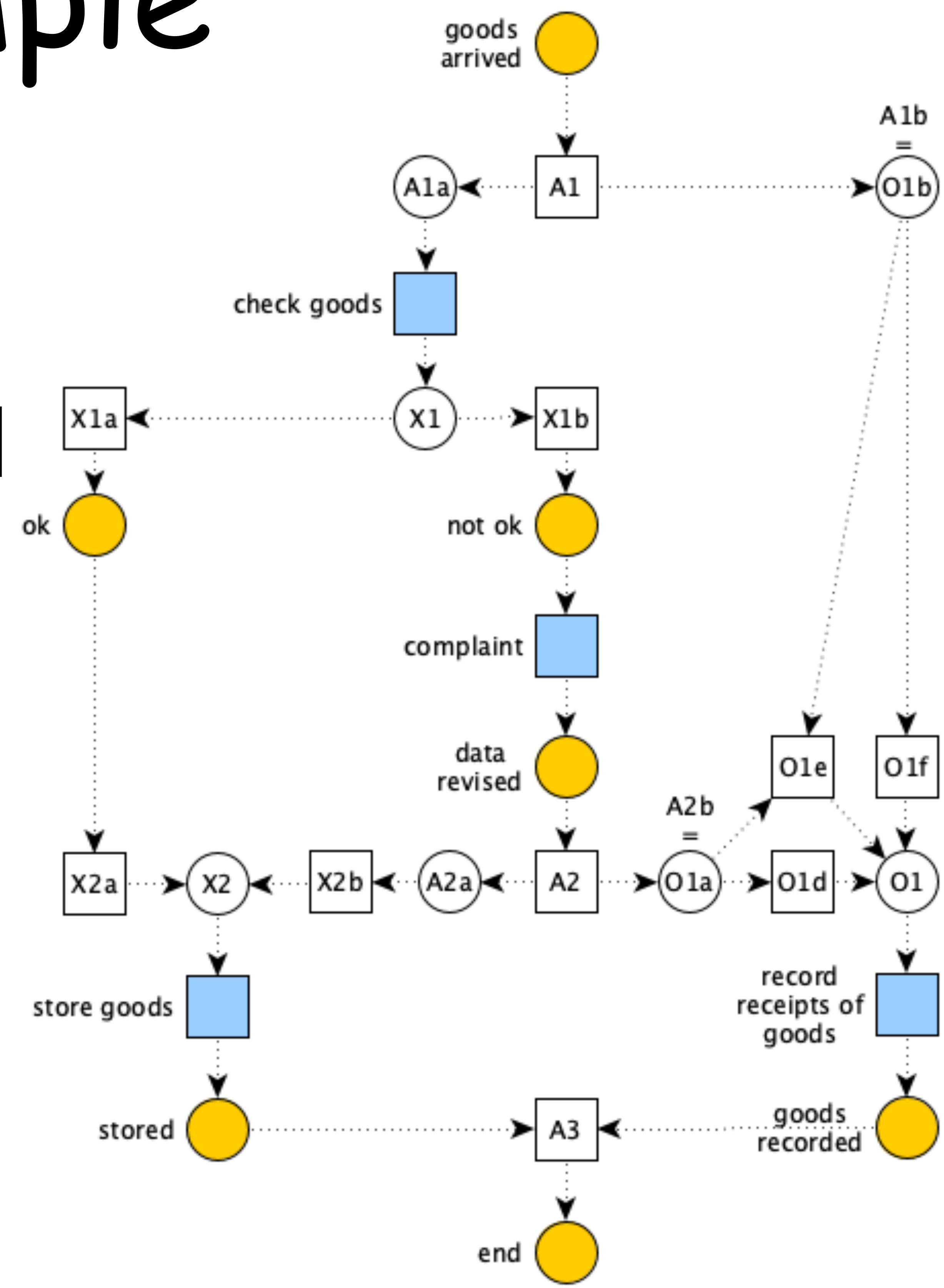
$$\forall t \in T.\ \exists \sigma \in L(N).\ \vec{\sigma}(t) > 0$$

Example

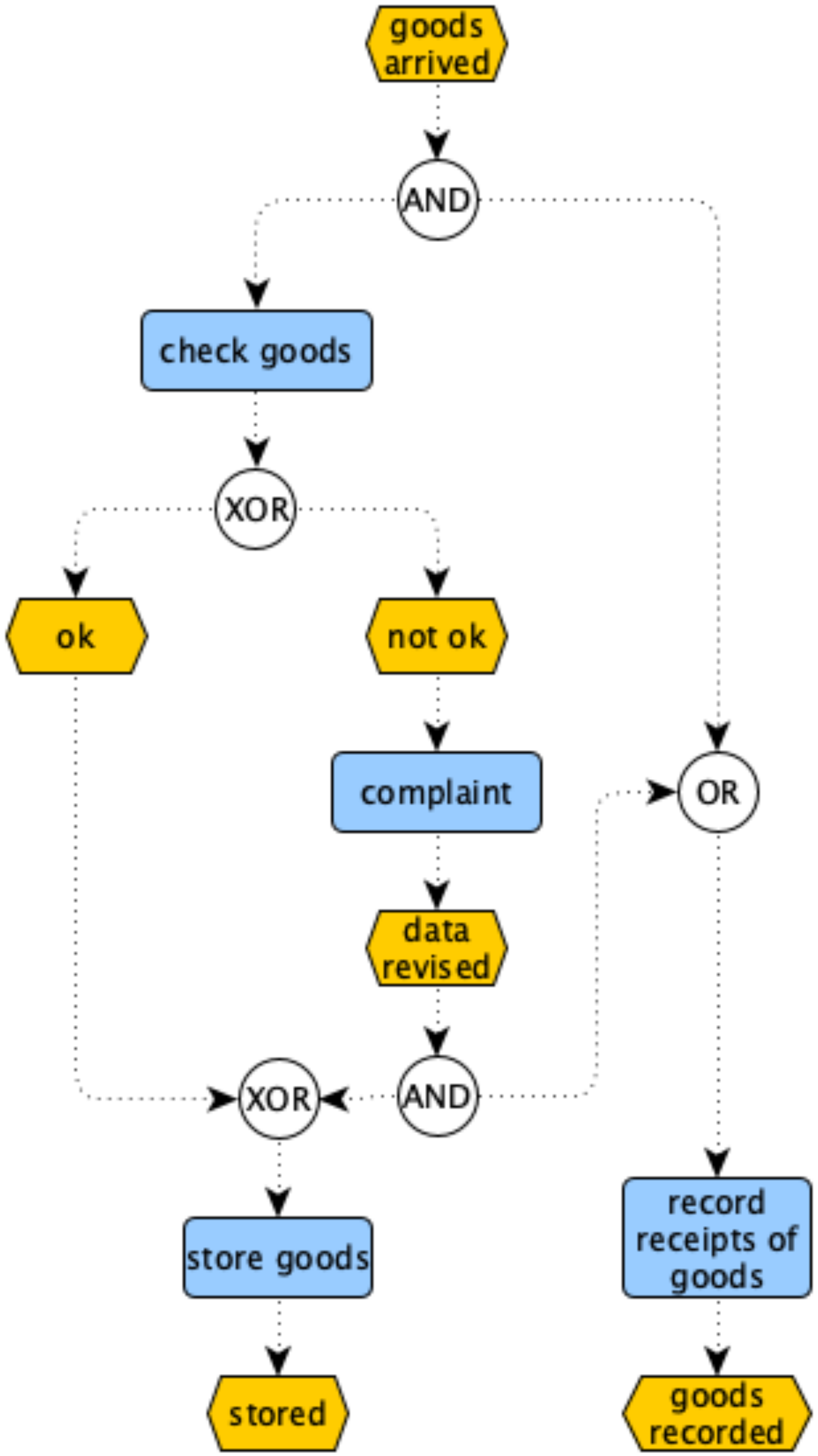


Relaxed sound?

Steps 1+2+3



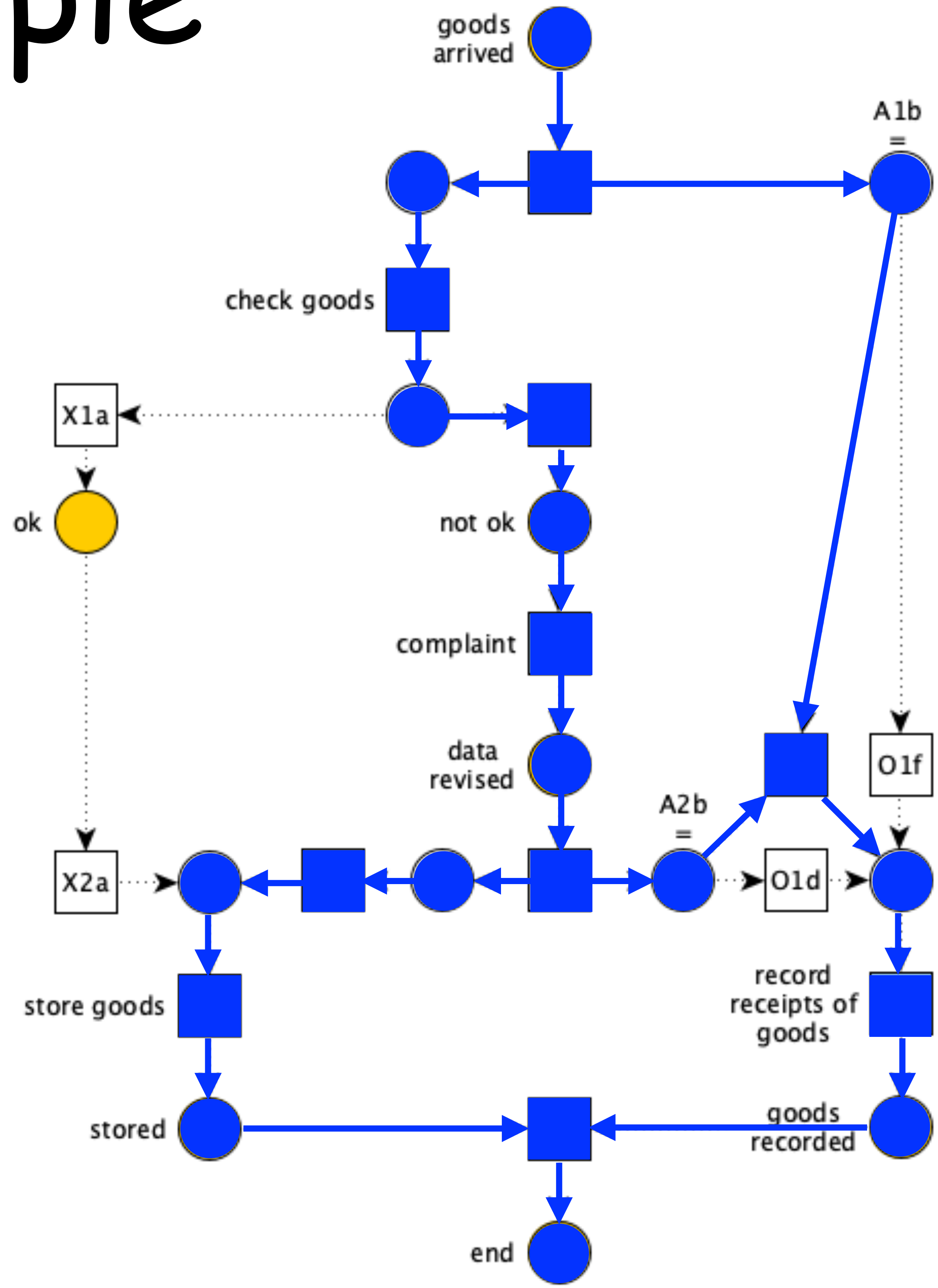
Example



Relaxed sound?

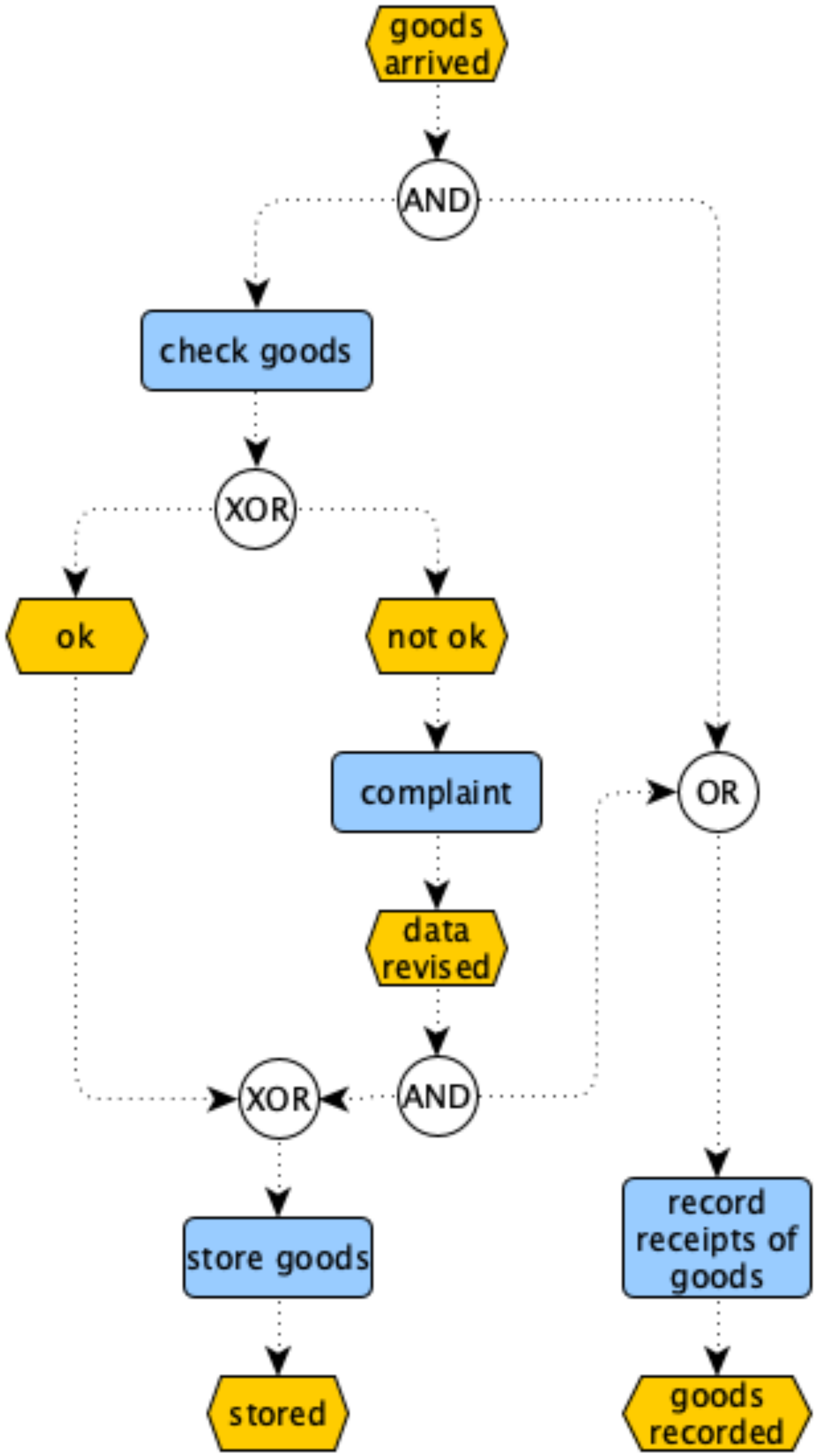
Steps 1+2+3

a sound execution



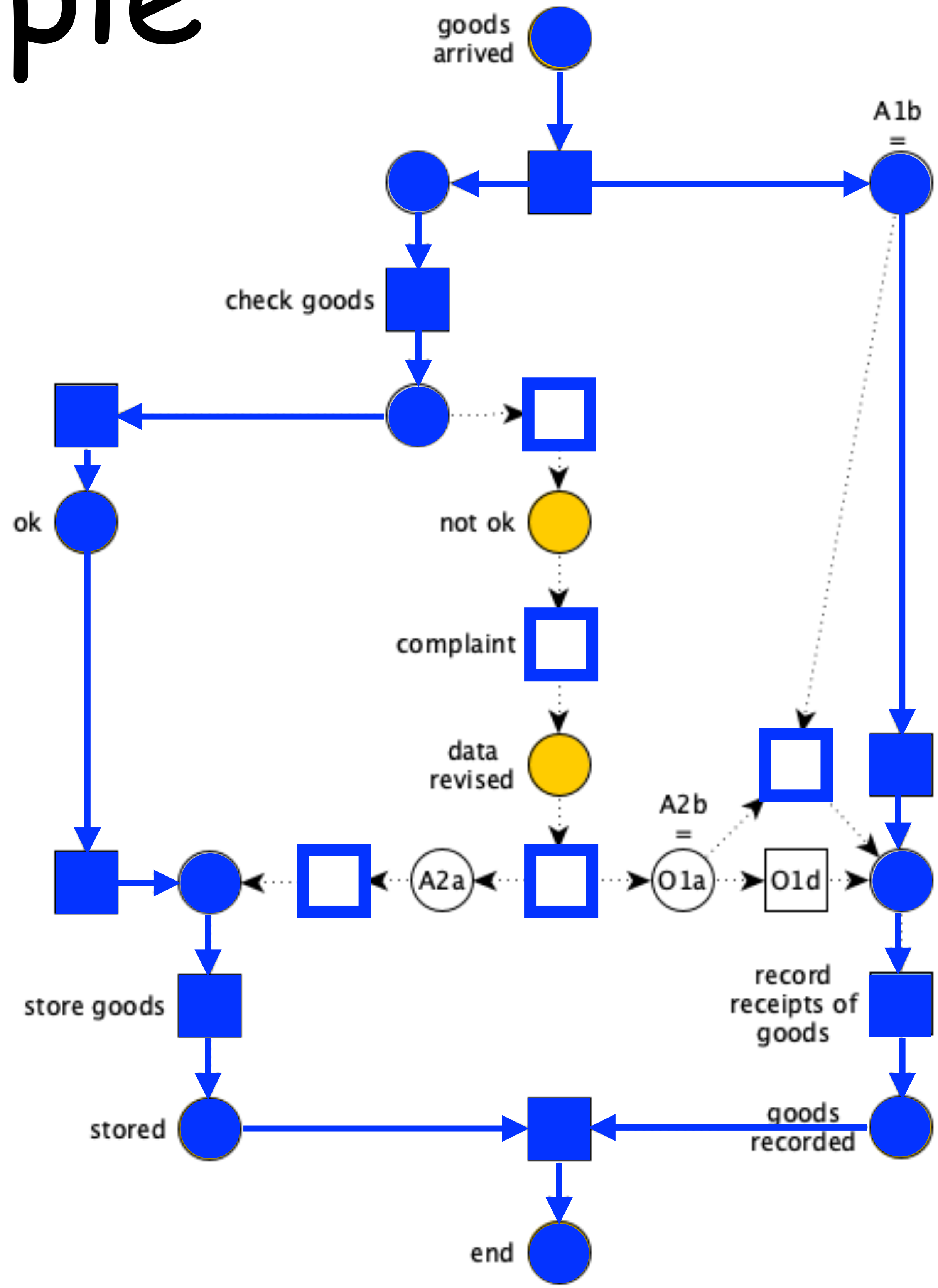
Example

another sound execution



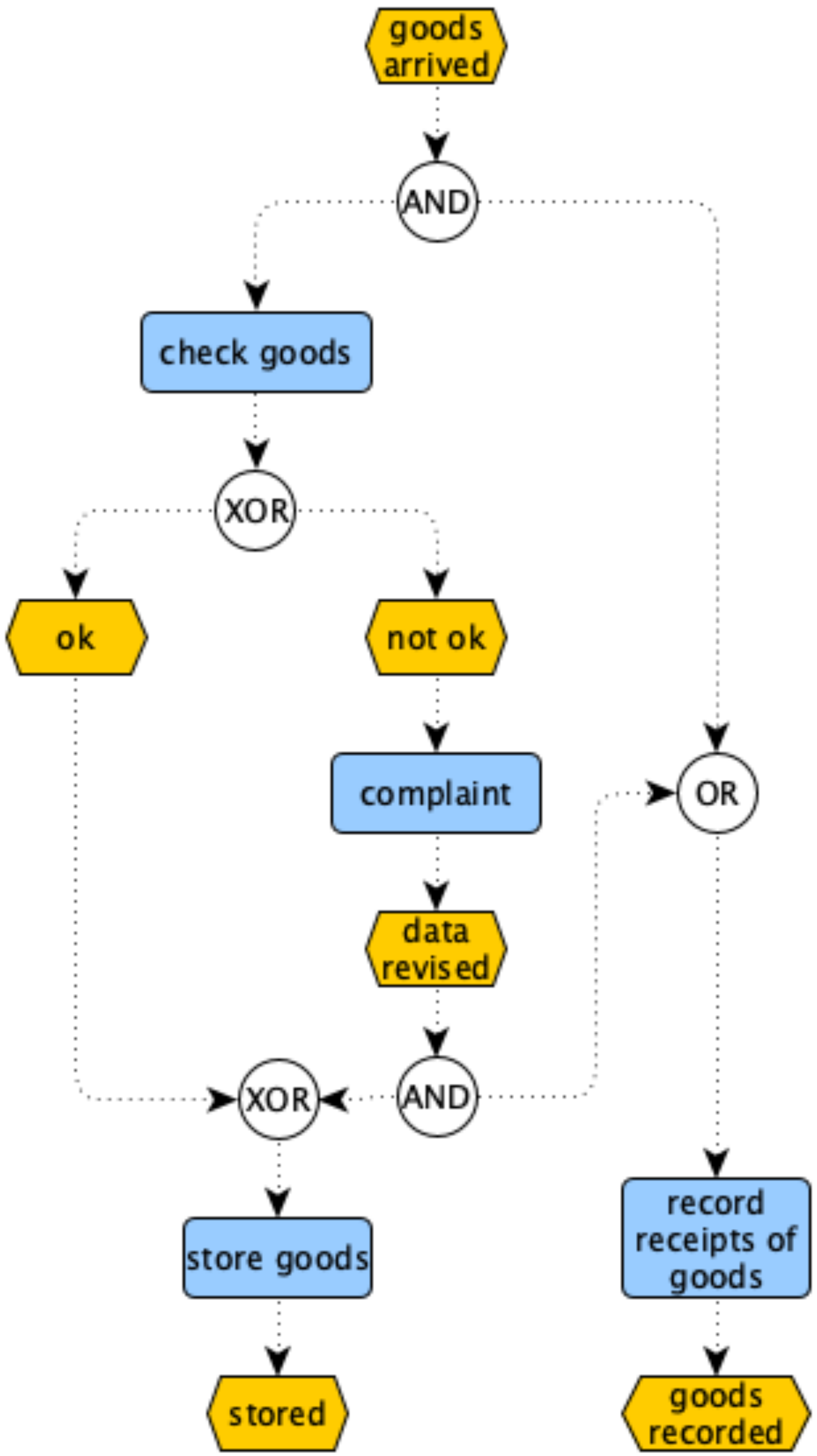
Relaxed sound?

Steps 1+2+3



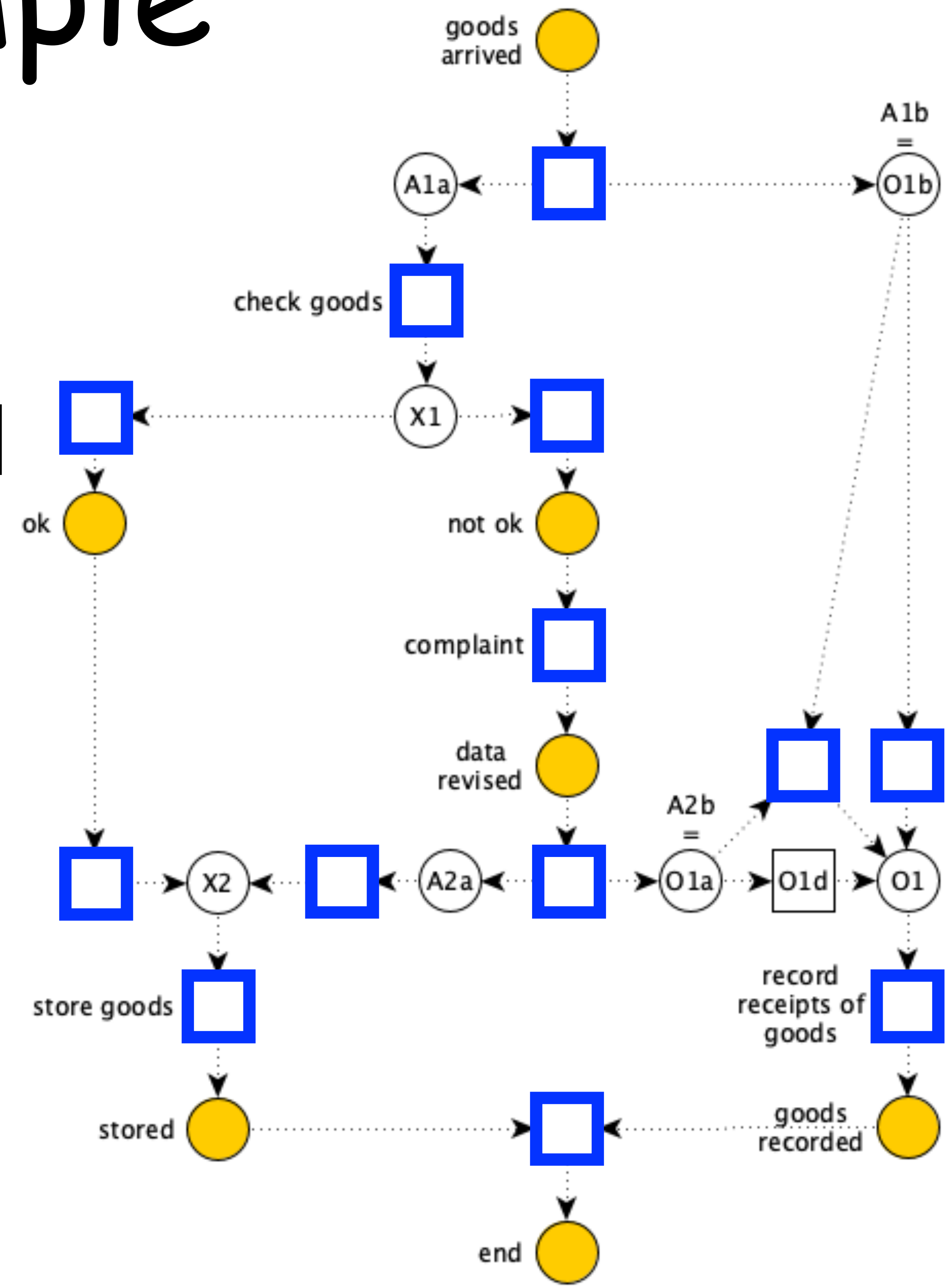
Example

tasks involved in some sound execution



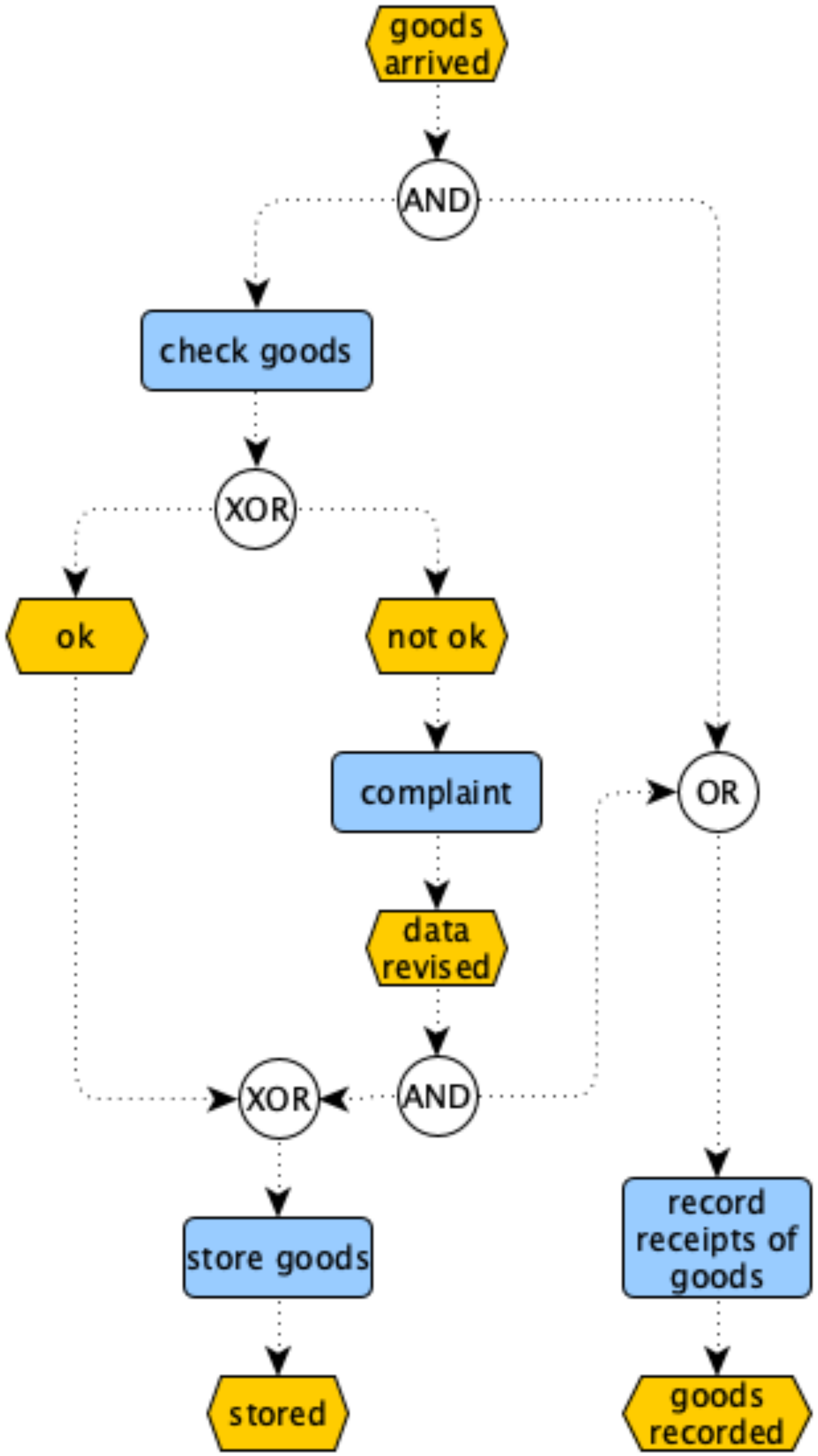
Relaxed sound?

Steps 1+2+3



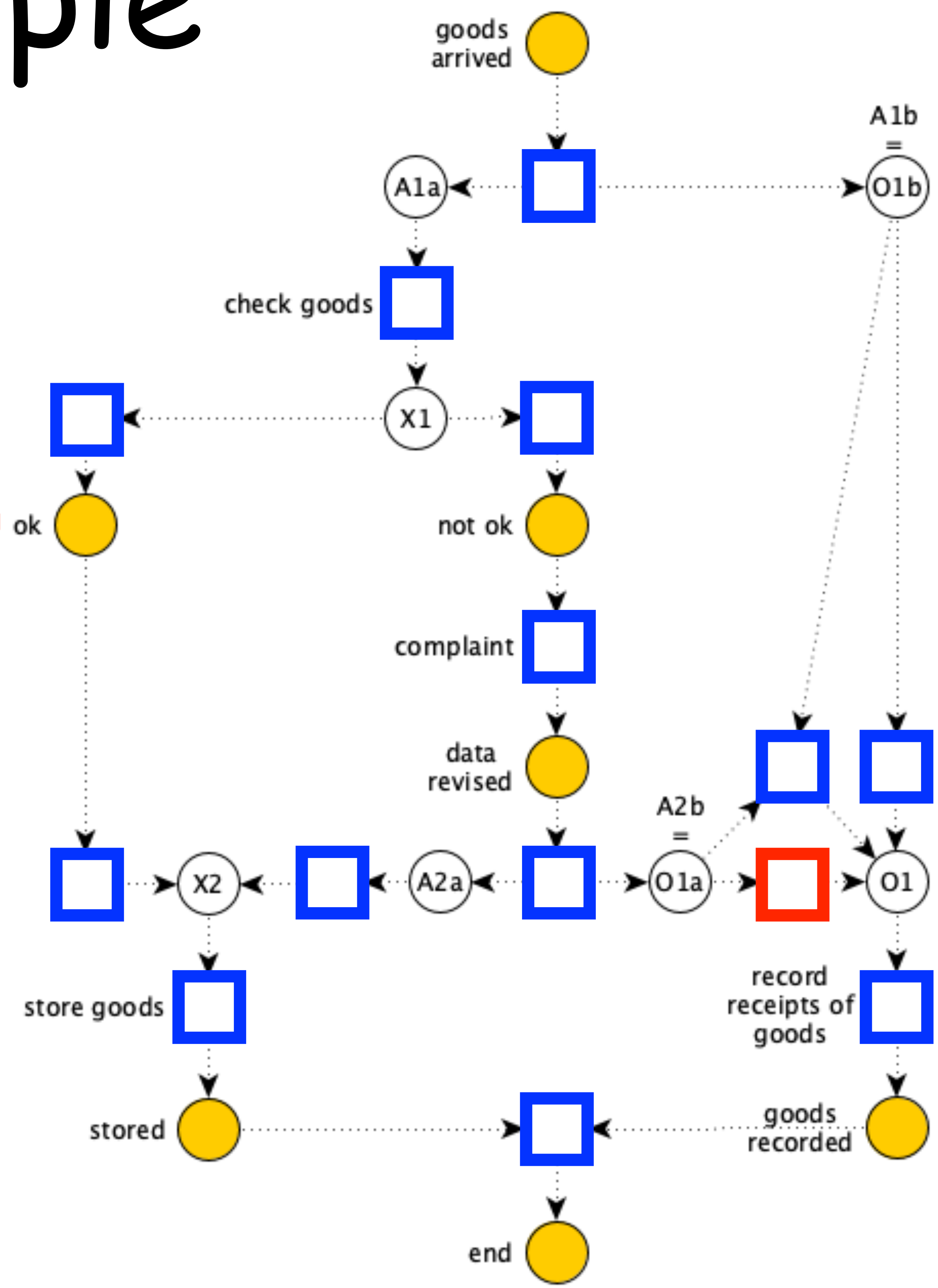
Example

one task not involved in some sound execution



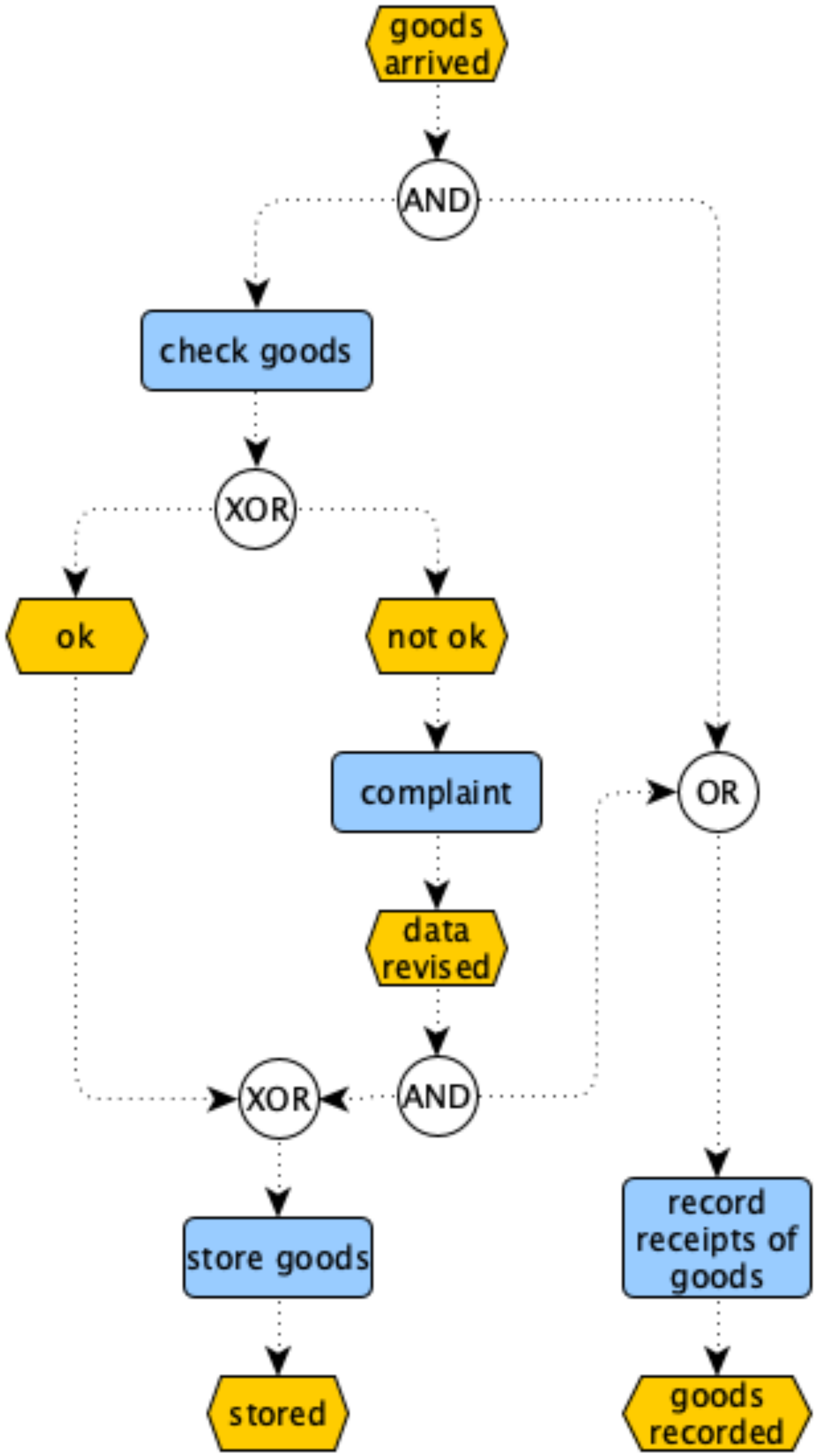
Not relaxed sound as a net!

Steps 1+2+3



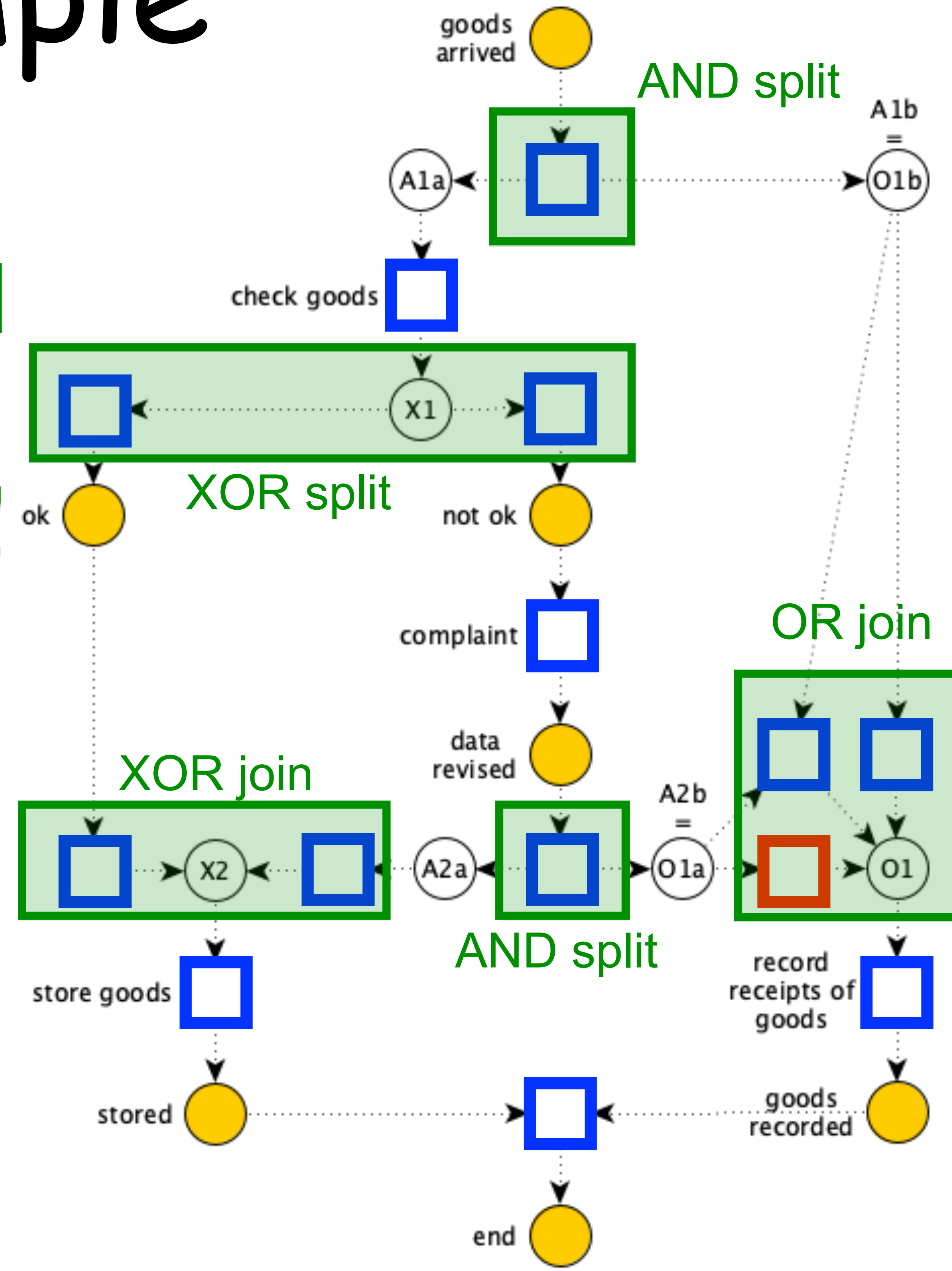
Example

all EPC nodes involved in some sound execution



Relaxed sound as EPC!

Steps 1+2+3



Relaxed soundness?

If the WF net is **not relaxed sound** there are transitions that are not involved in sound executions (not included in a firing sequence of L(N))

Their EPC counterparts may need improvements

Relaxed soundness can be proven only by enumeration (of enough firing sequences of L(N))

Open problem

No equivalent characterization is known that is more convenient to check

Second attempt (no OR connectors)

Formalization and Verification of Event-driven Process Chains

W.M.P. van der Aalst

Department of Mathematics and Computing Science, Eindhoven University of Technology, P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands, telephone: -31 40 2474295, e-mail: wsinwa@win.tue.nl

Simplified EPC

We restrict the analysis to a sub-class of EPC diagrams

We require:

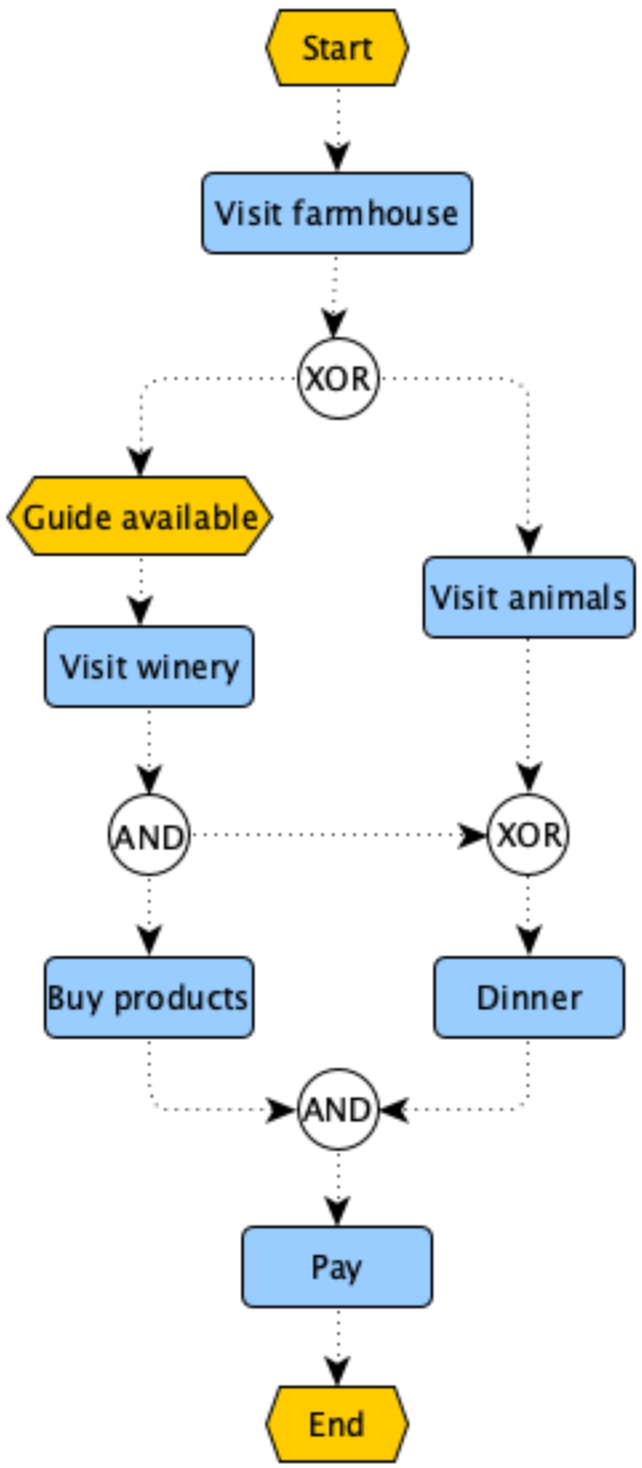
event / function alternation
(also along paths between two connectors)
(fusion not needed, dummy places/transitions not needed)

OR-connectors are not present
(avoid intrinsic problems with OR join)

OR-connectors are not present

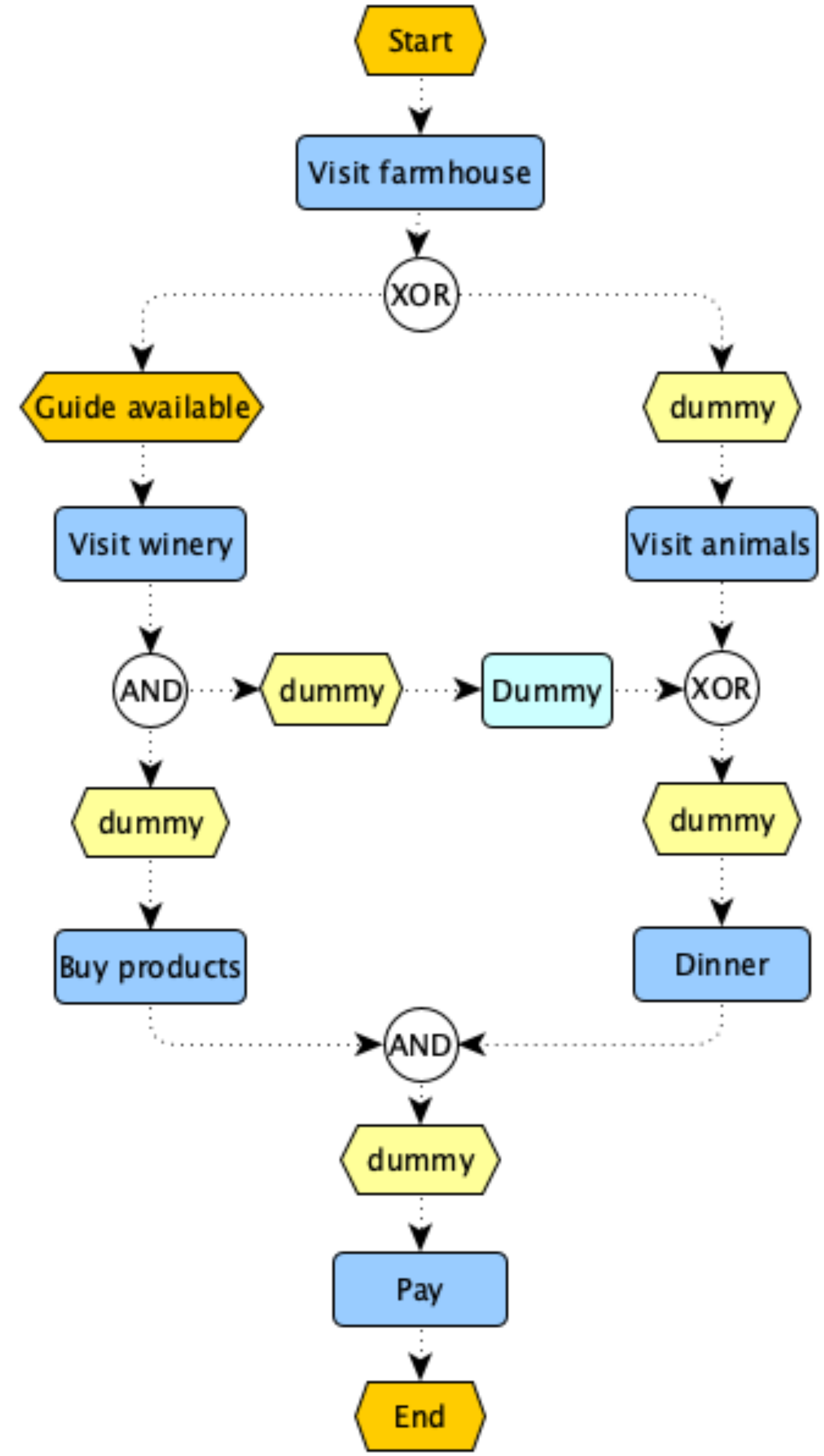
alternation is not satisfied

Example

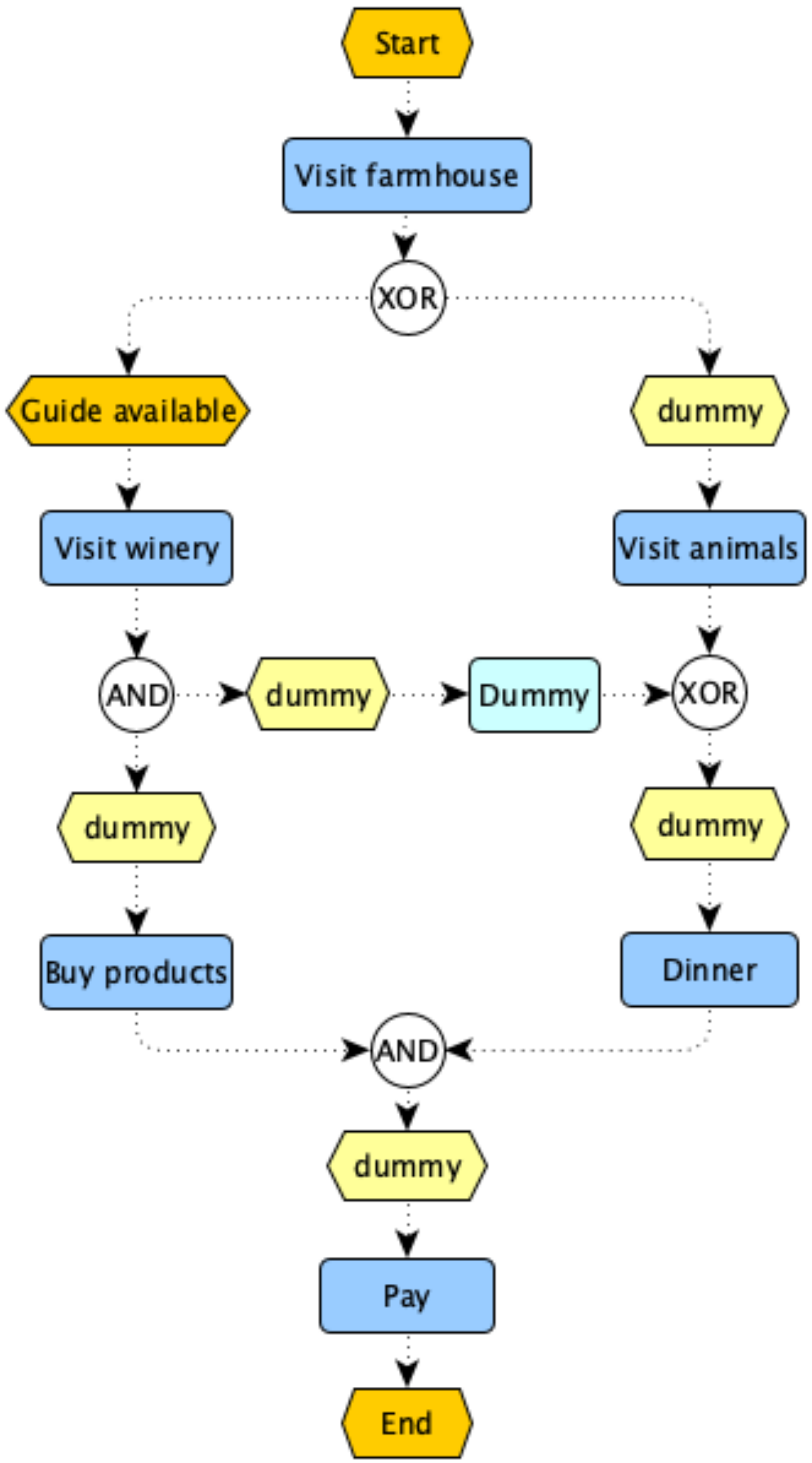


Add dummy events and functions to force alternation

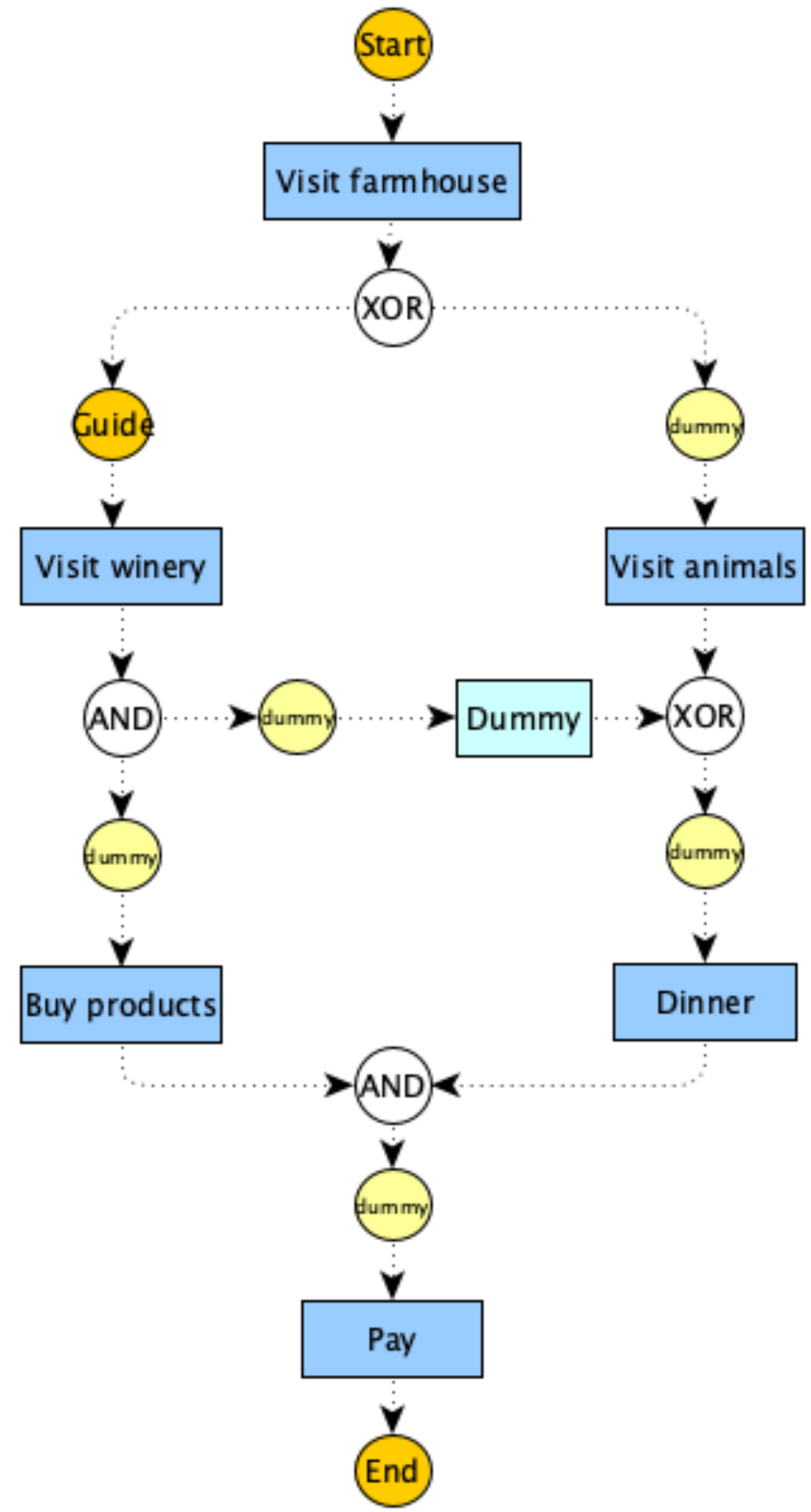
Step 0



Example



Step 1
events and
functions



Step 1: split/join connectors

The translation of logical connectors **depends on the context**:

if a connector connects **functions to events** we apply a certain translation

if it connects **events to functions** we apply a different translation

Step 1: split/join connectors

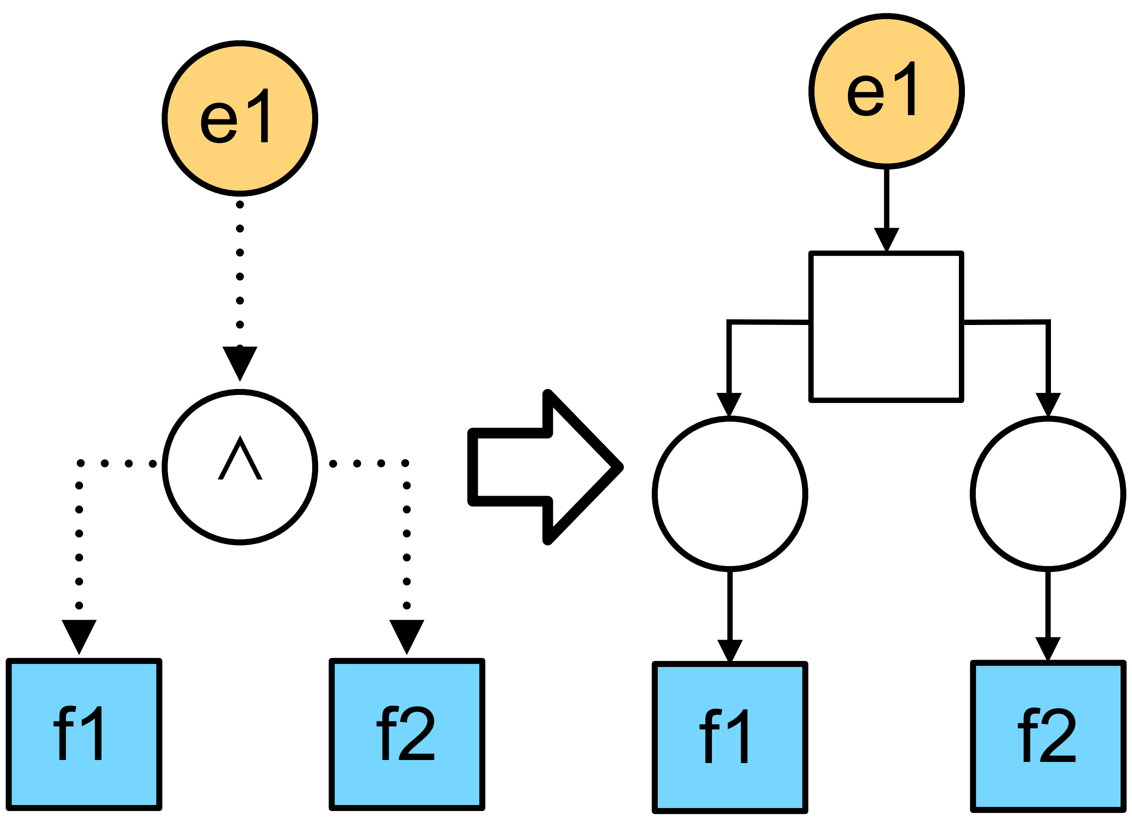
The translation of logical connectors
depends on the context:

if a connector connects **transitions to places**
we apply a certain translation

if it connects **places to transitions**
we apply a different translation

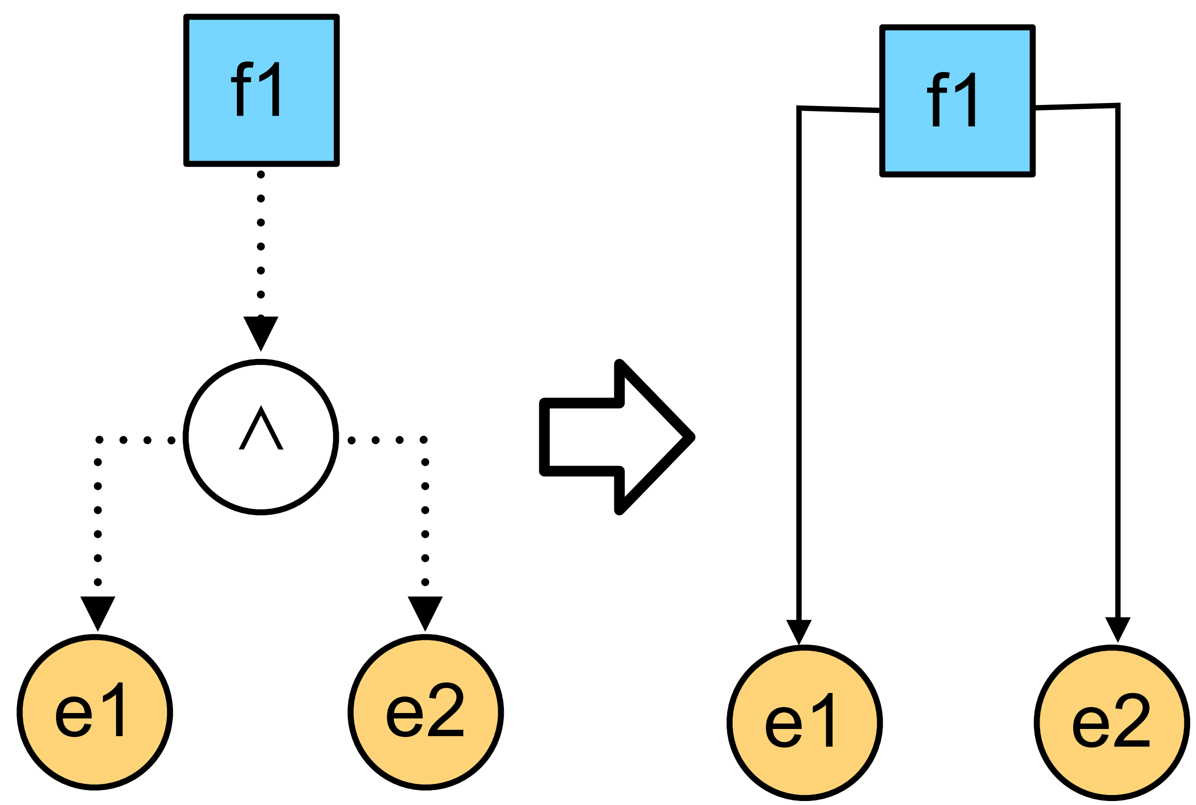
Step 1: AND split

EPC **net fragment**



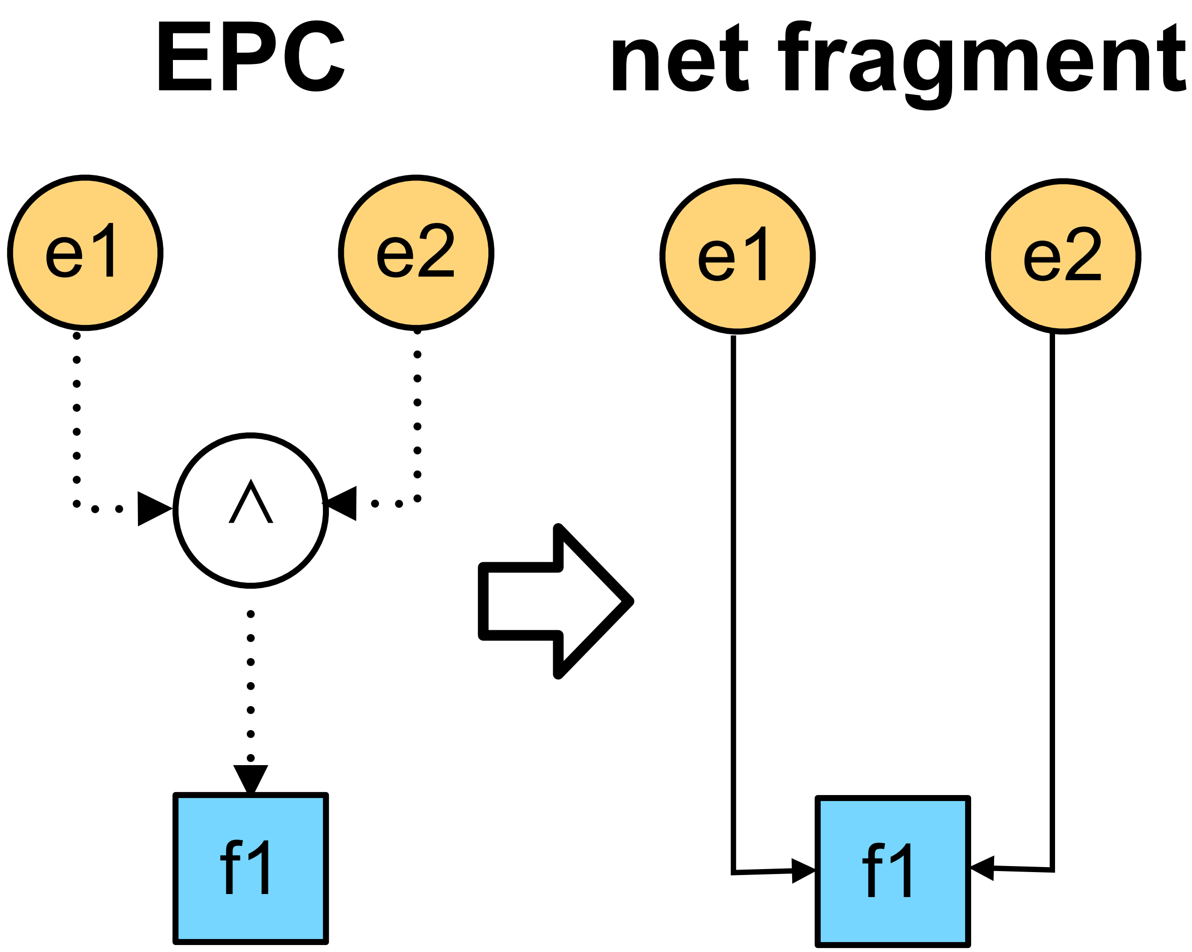
(event to functions)

EPC **net fragment**

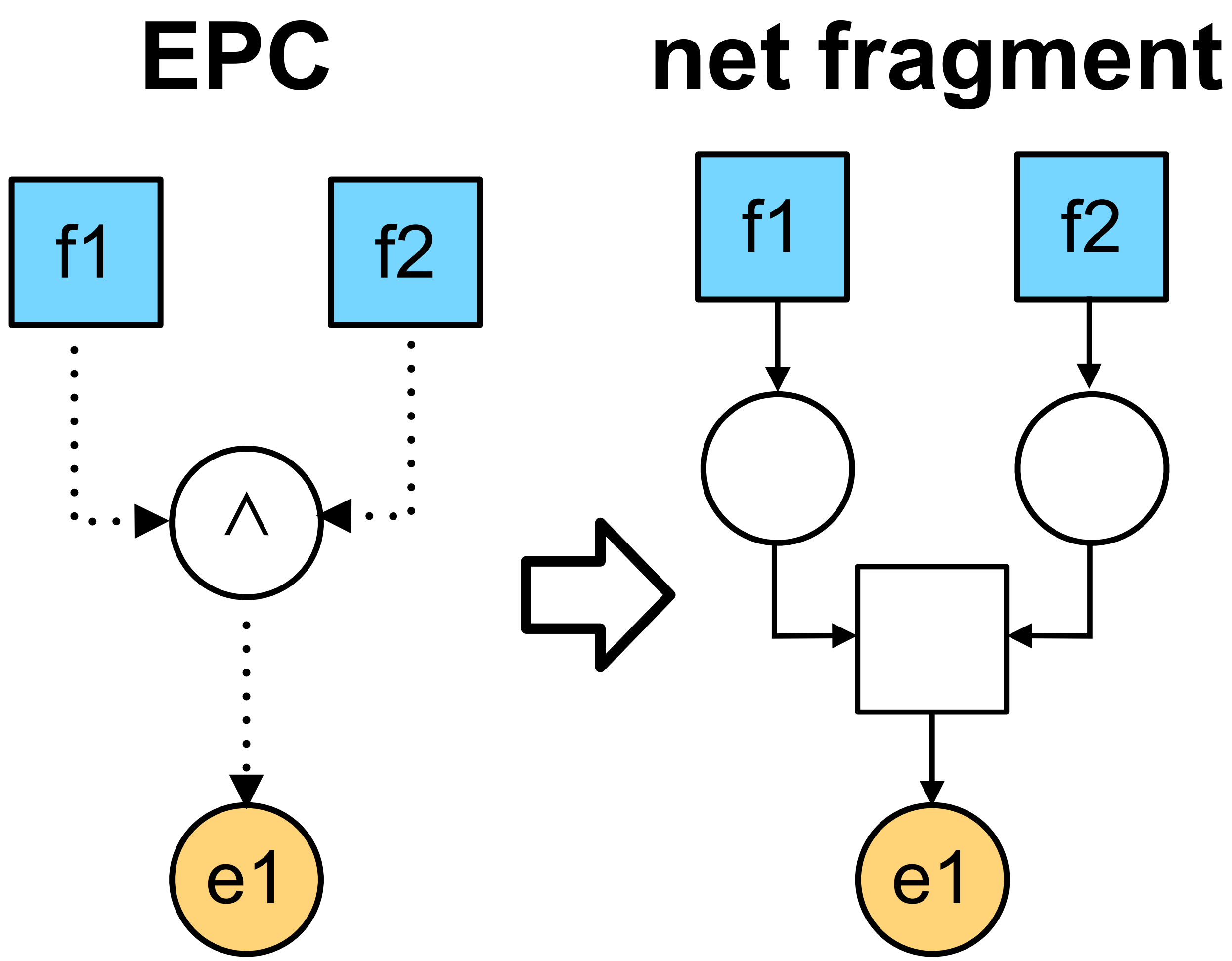


(functions to events)

Step 1: AND join

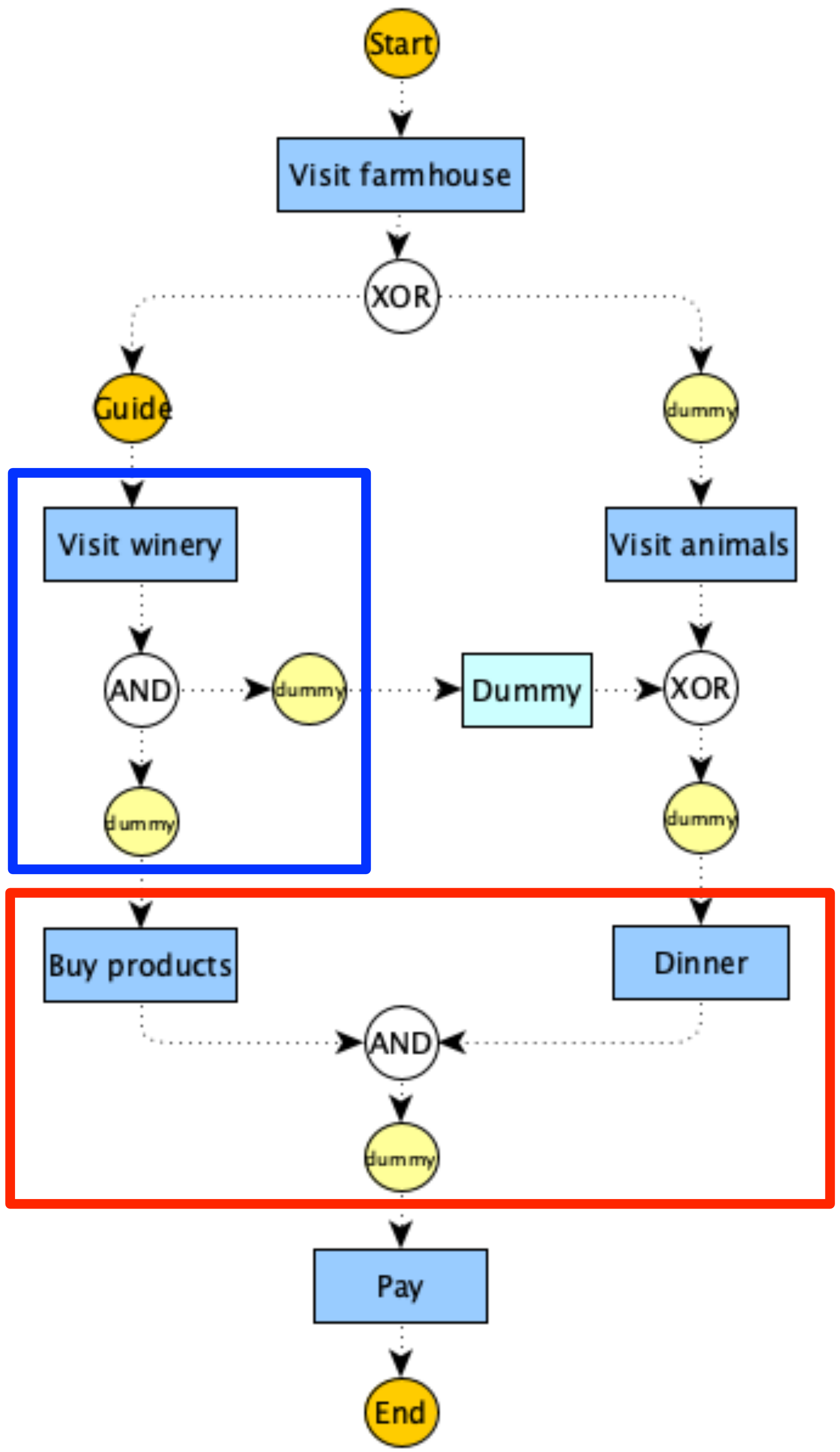


(event to functions)

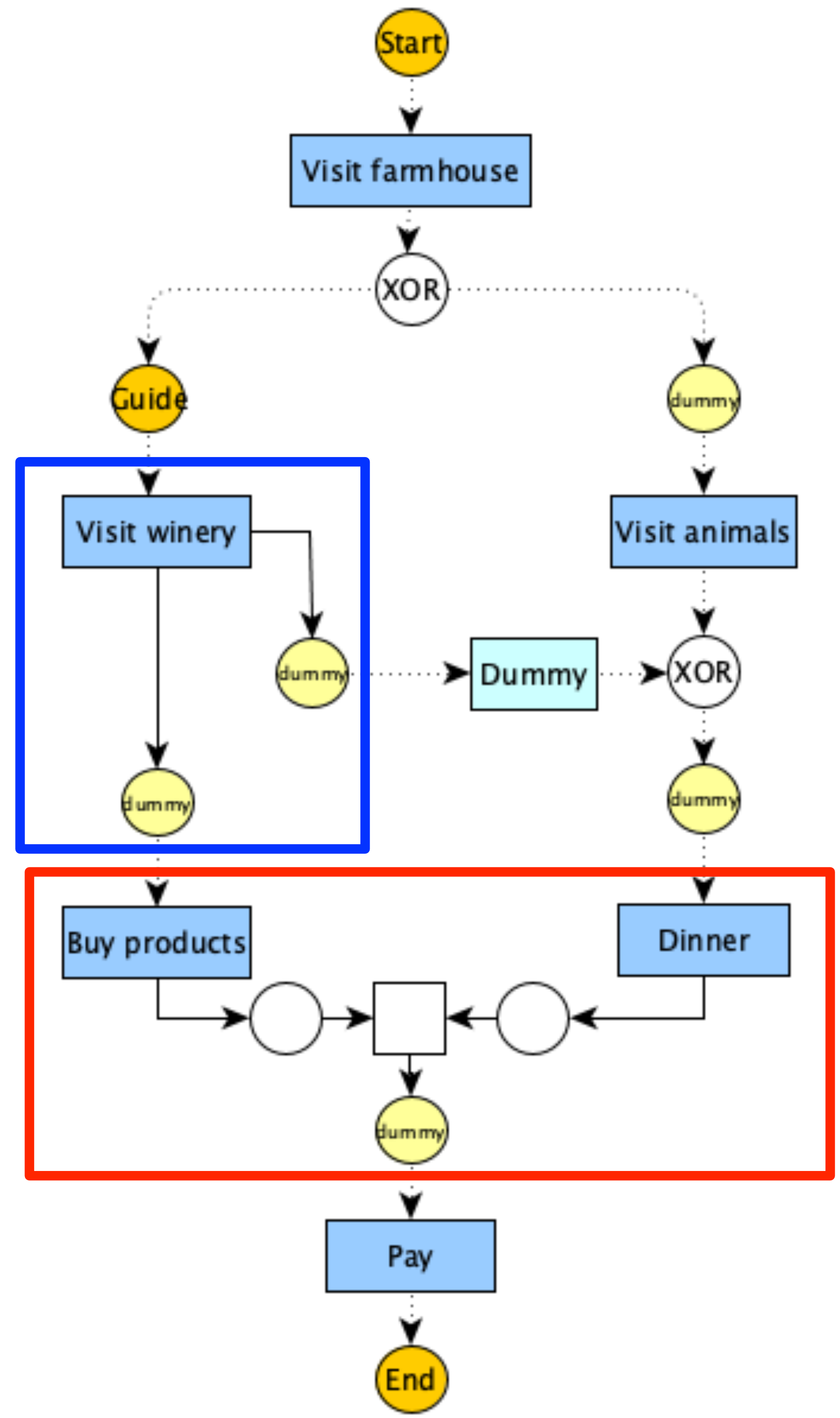


(functions to events)

Example

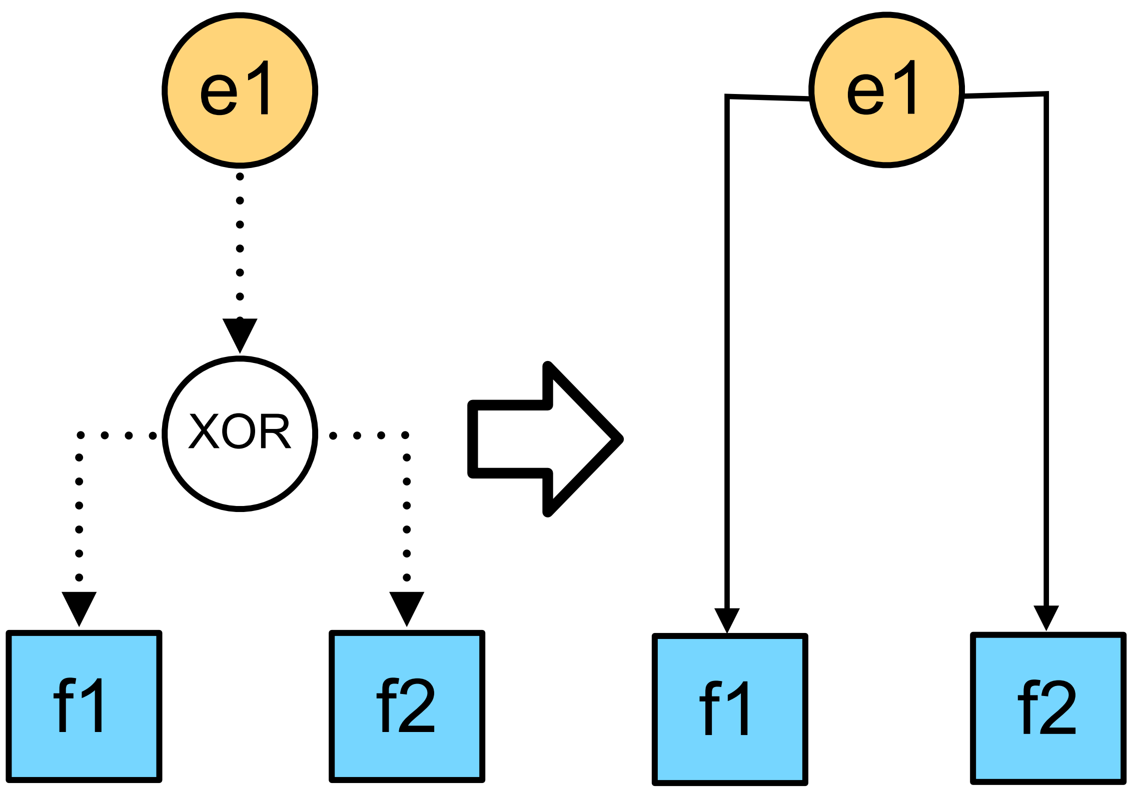


Step 1
AND
connectors



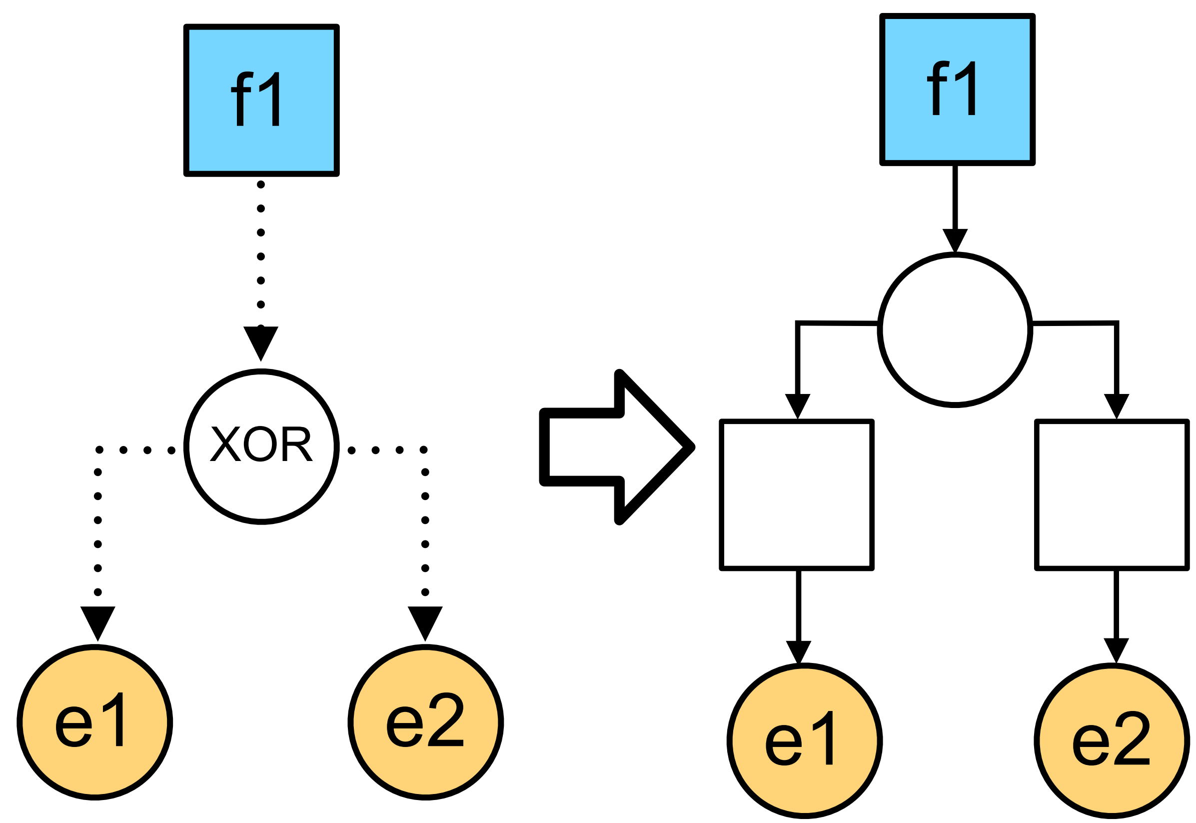
Step 1: XOR split

EPC **net fragment**



(event to functions)

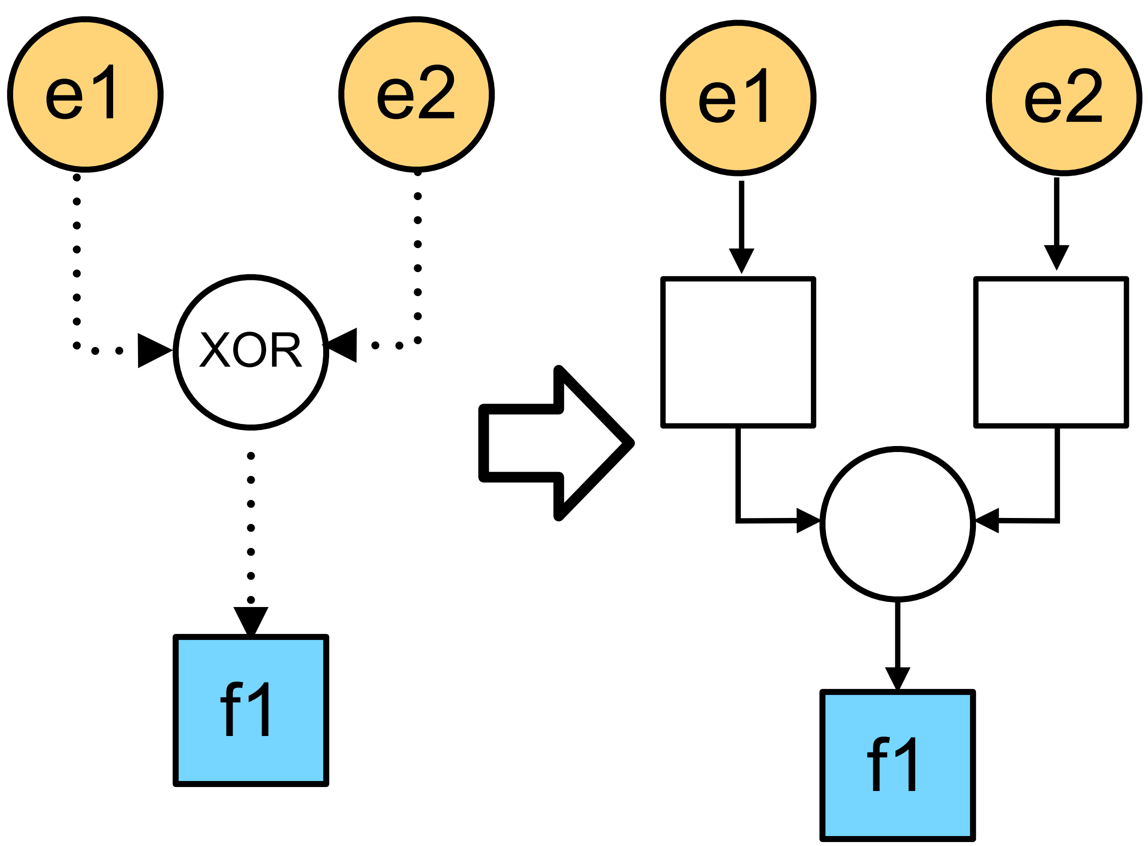
EPC **net fragment**



(functions to events)

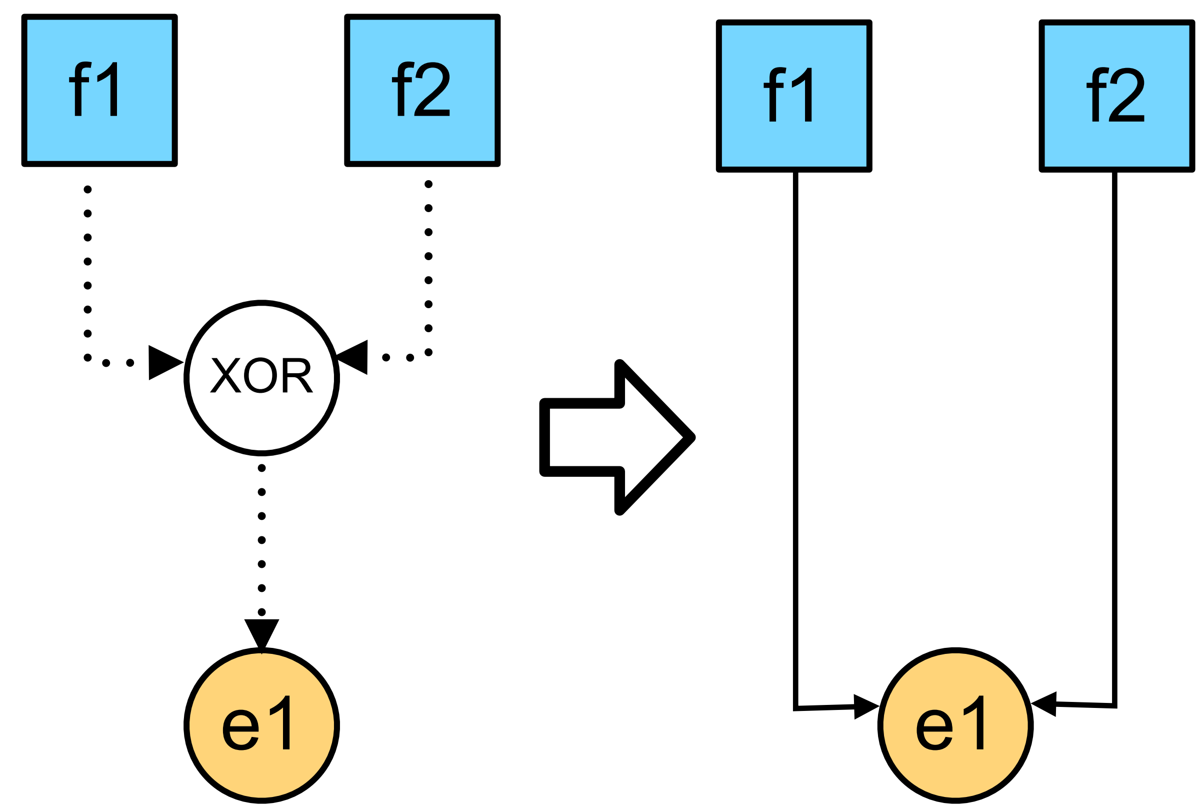
Step 1: XOR join

EPC **net fragment**



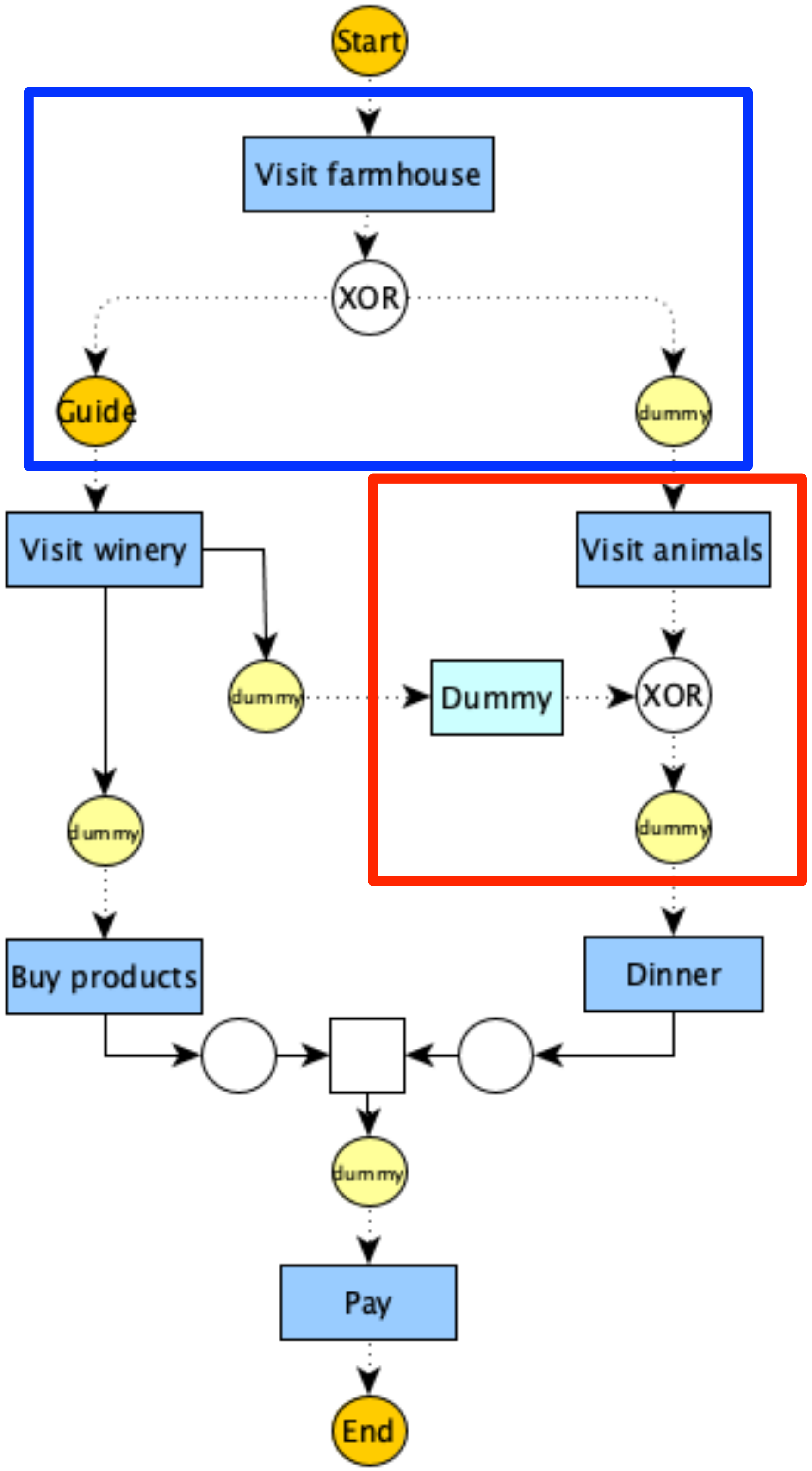
(event to functions)

EPC **net fragment**

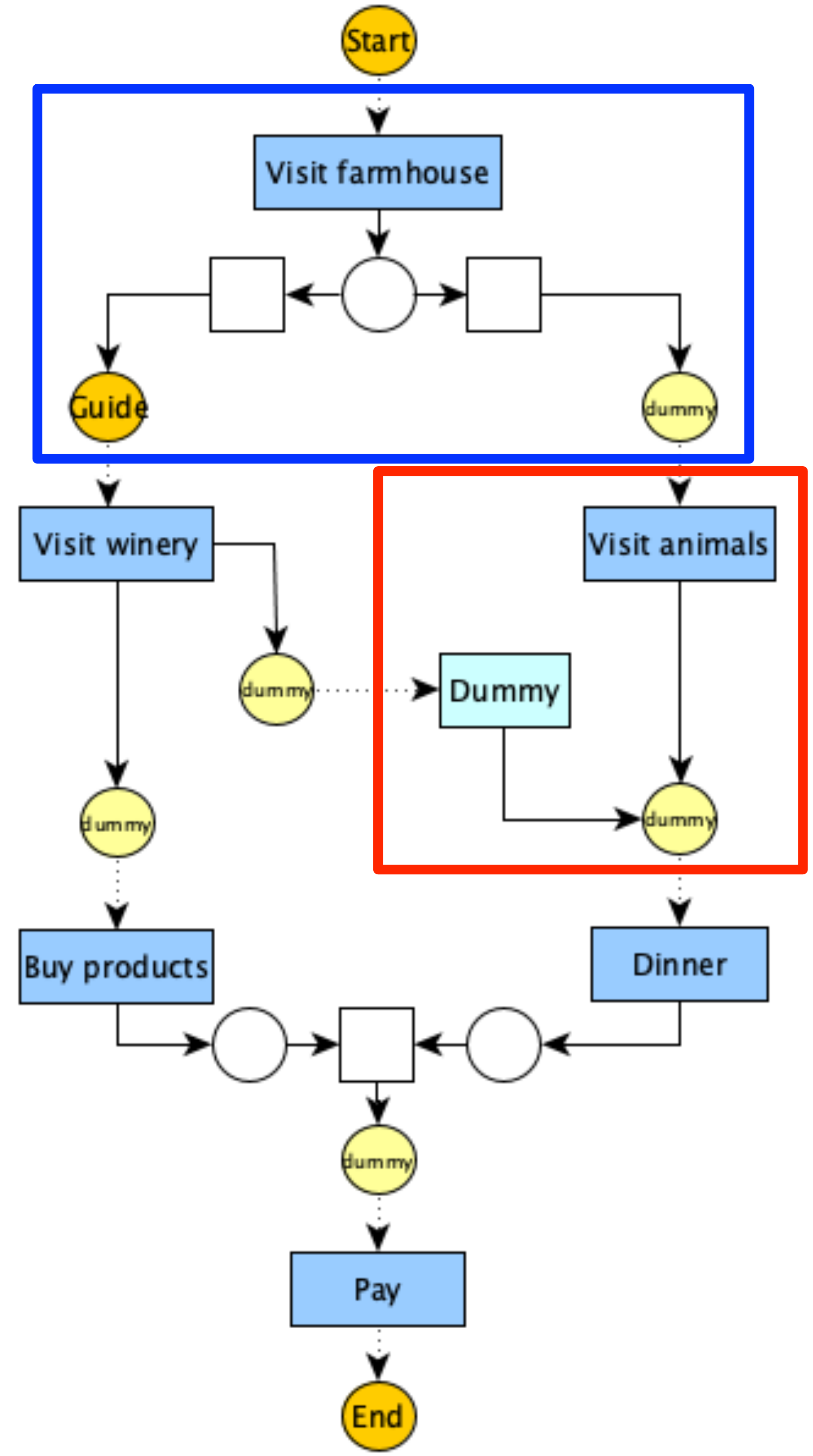


(functions to events)

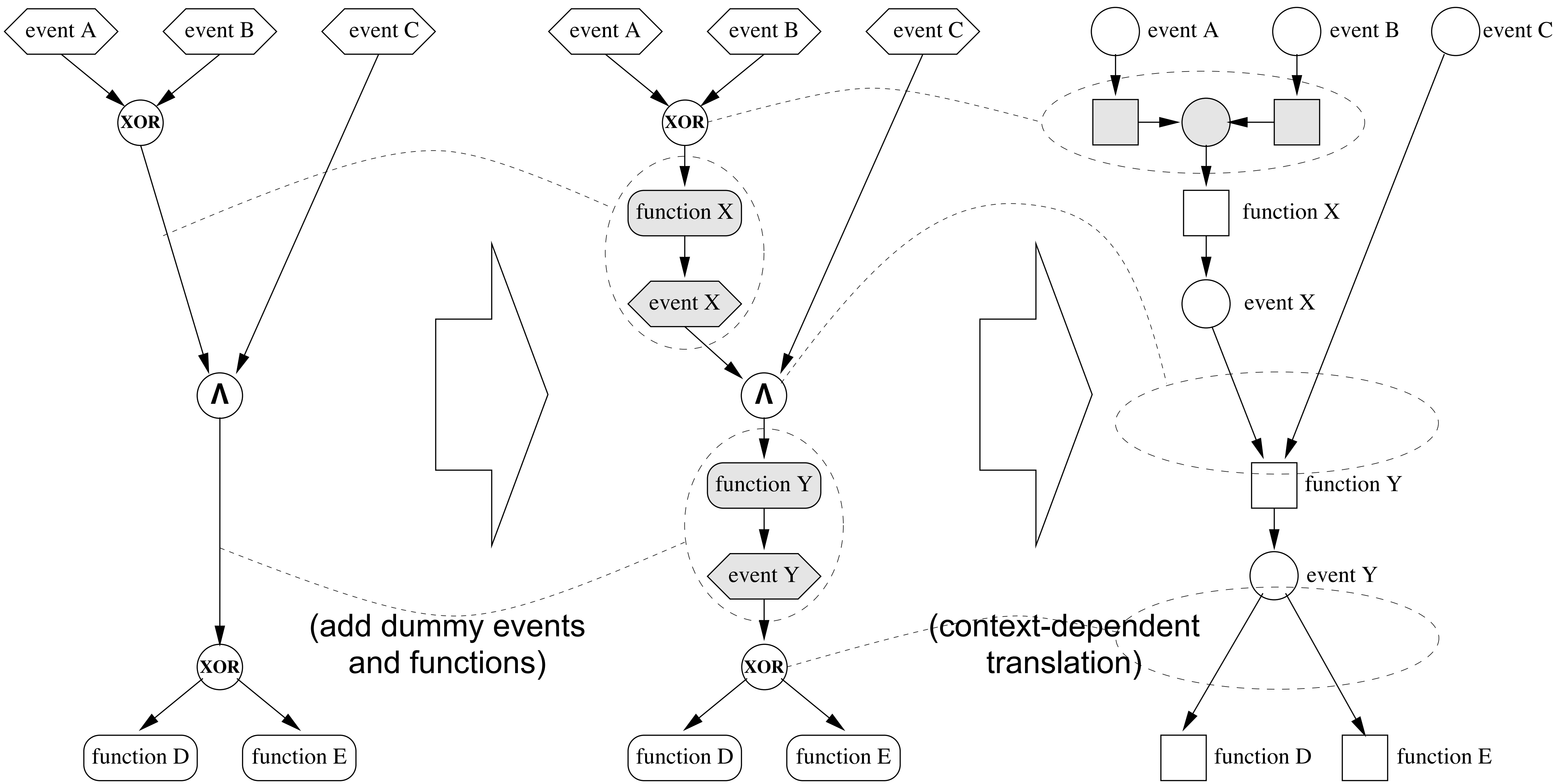
Example



Step 1
XOR
connectors

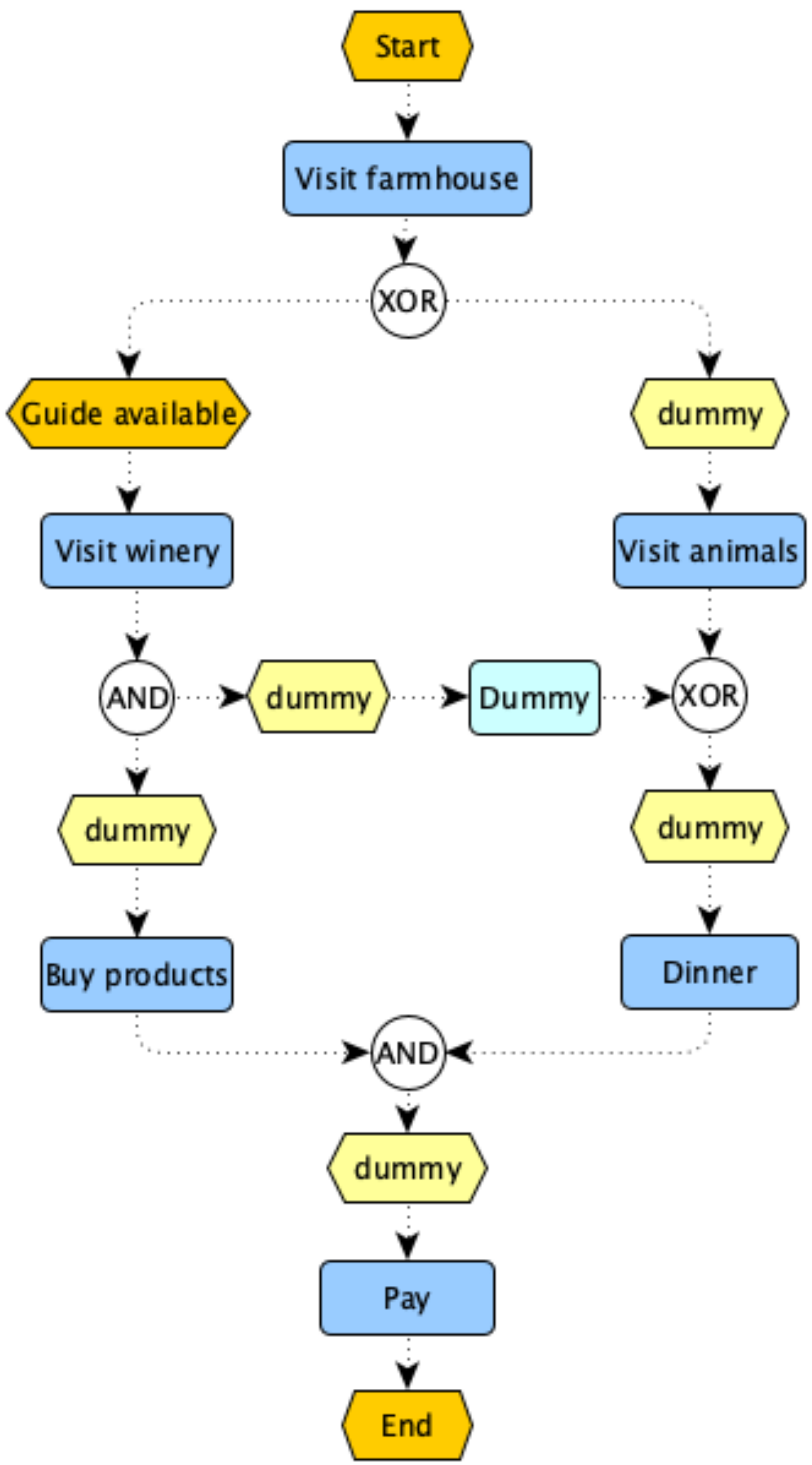


Overall strategy



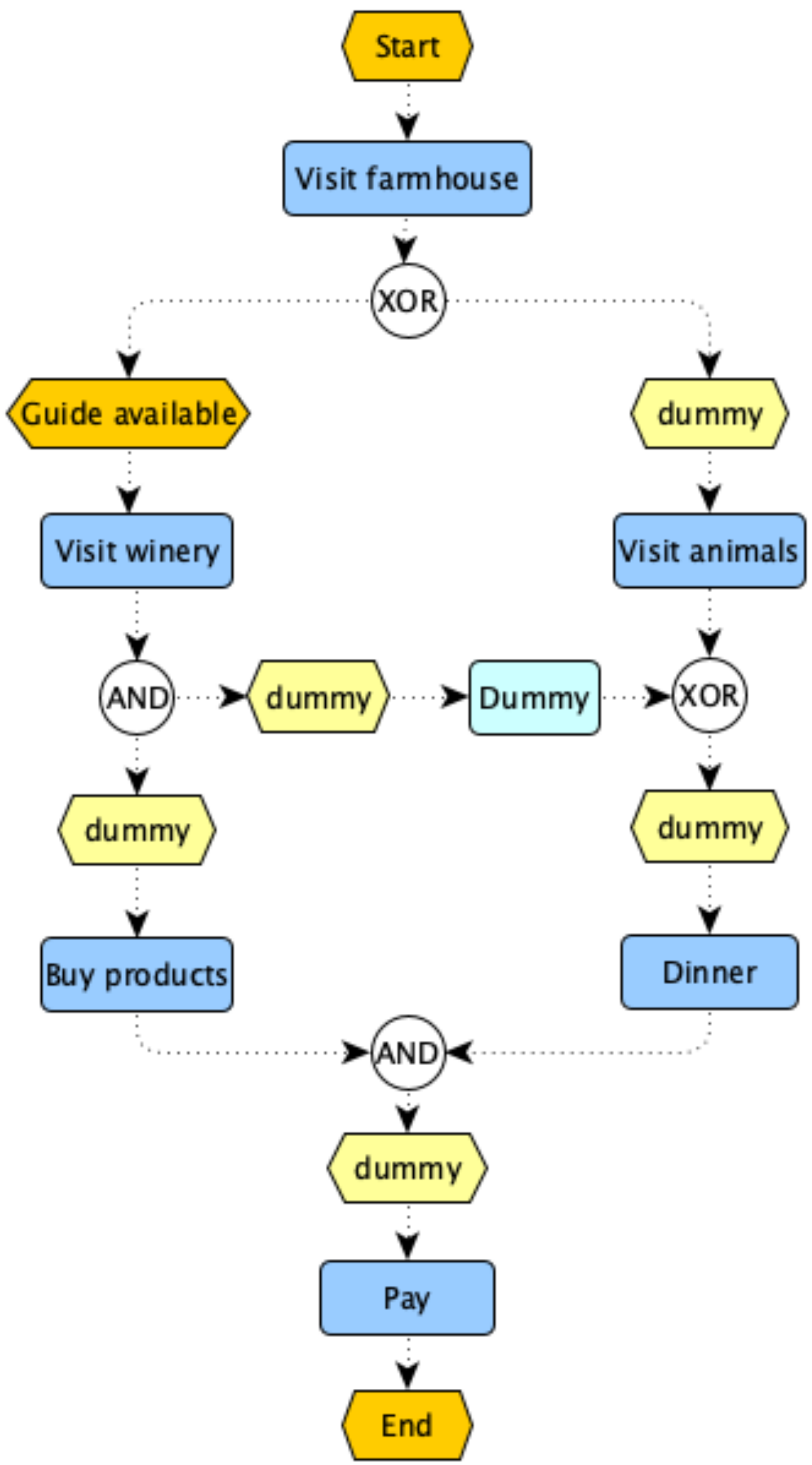
From any EPC we derive a free-choice net

Example



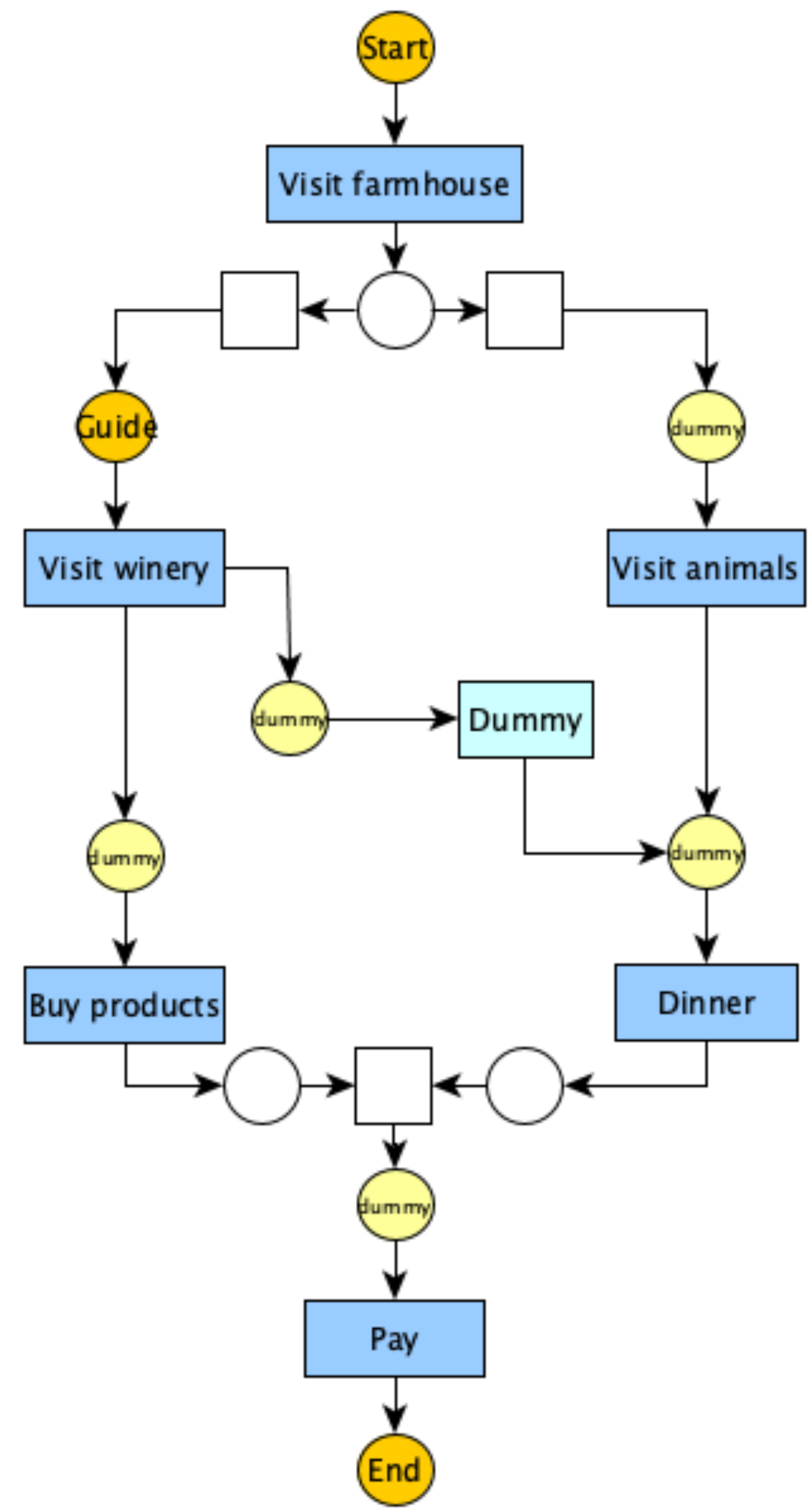
Sound?

Example

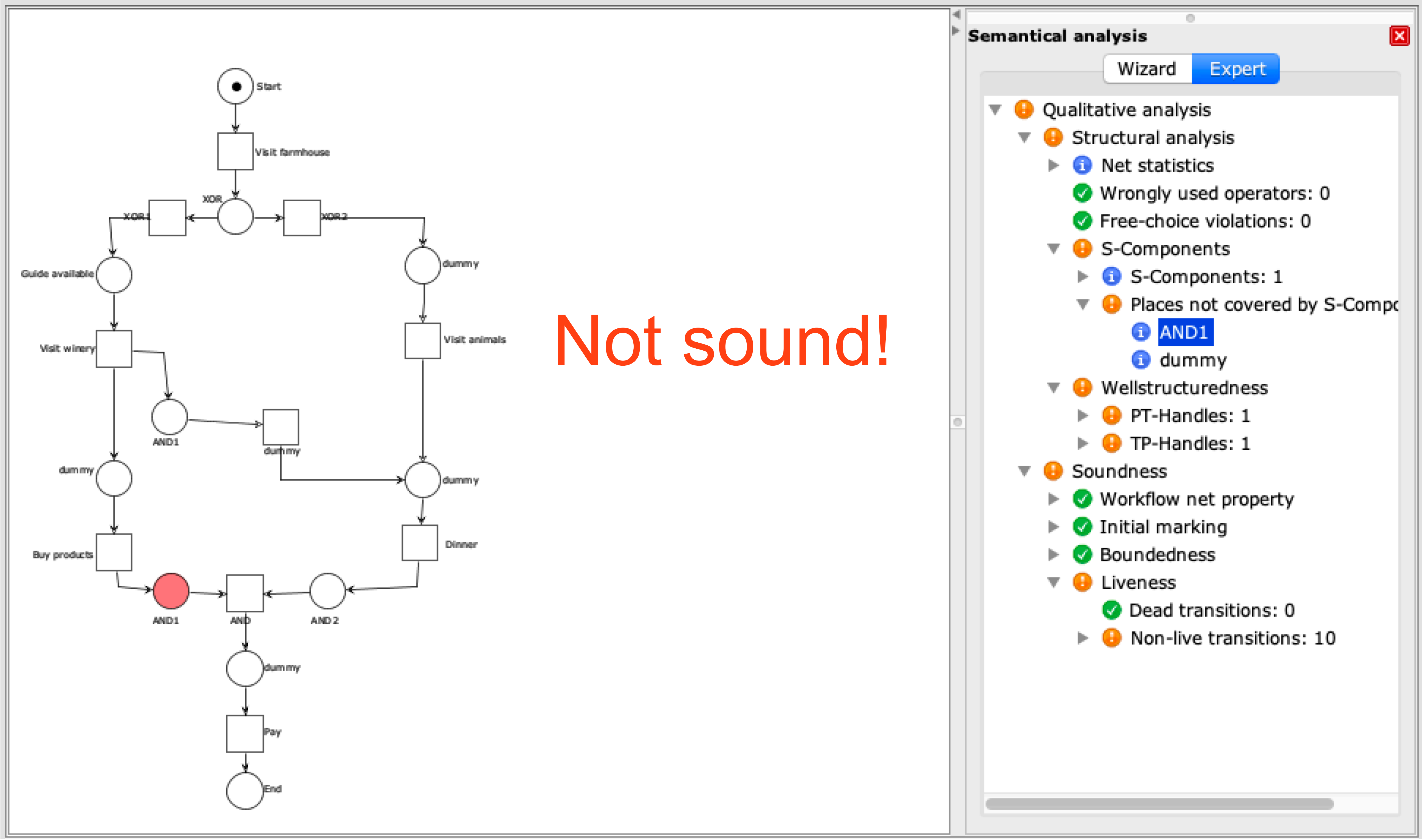


Sound?

Steps
1+2(+3)



Example


Start
Visit farmhouse
XOR
XOR1
XOR2
Guide available
dummy
Visit winery
Visit animals
AND1
dummy
dummy
dummy
Buy products
Dinner
AND1
AND
AND2
dummy
Pay
End
Not sound!
Semantical analysis
Wizard
Expert
Qualitative analysis
Structural analysis
Net statistics
Wrongly used operators: 0
Free-choice violations: 0
S-Components
S-Components: 1
Places not covered by S-Compo
AND1
dummy
Wellstructuredness
PT-Handles: 1
TP-Handles: 1
Soundness
Workflow net property
Initial marking
Boundedness
Liveness
Dead transitions: 0
Non-live transitions: 10

Third attempt (decorated EPC)

UNIVERSITÄT KOBLENZ · LANDAU

iwi Institut für Wirtschaftsinformatik
Fachbereich Informatik
Universität Koblenz-Landau

Decorated EPC

Applicable to any EPC diagram, provided that its designer add some information

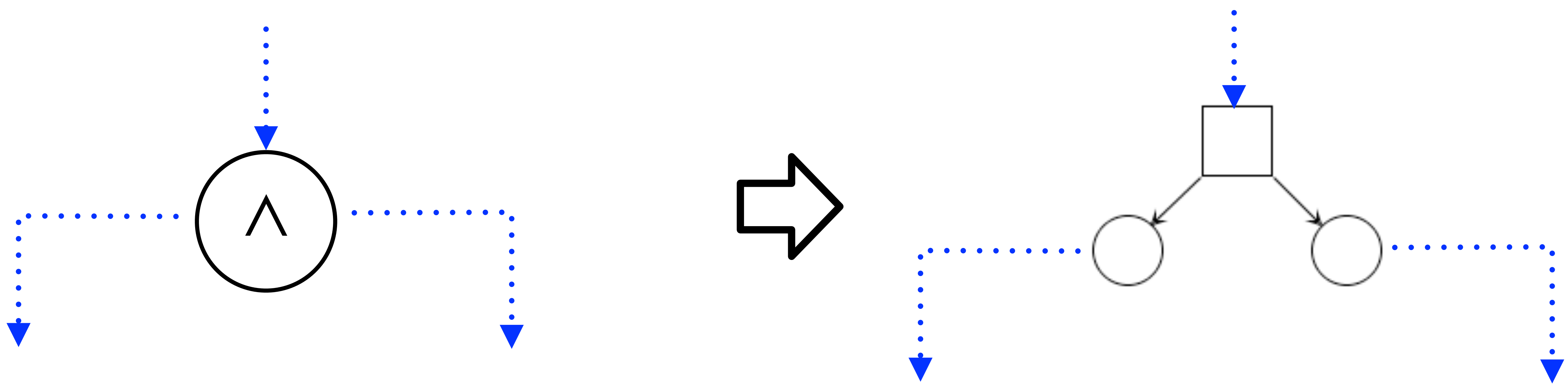
We require:

every (X)OR join is paired with a corresponding split
(possibly of the same type)

OR-joins are decorated with a policy
(avoid OR join ambiguous behaviour)

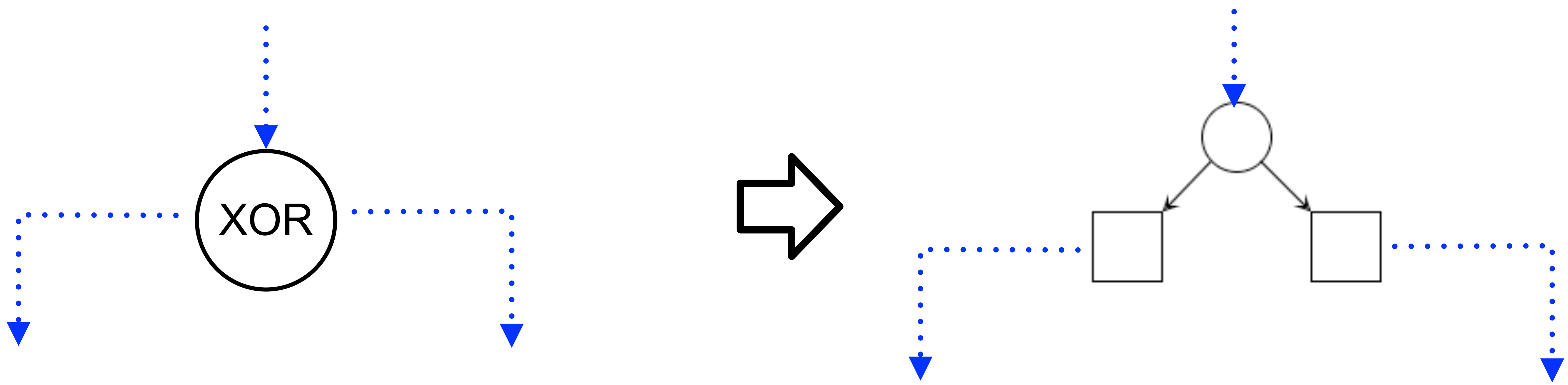
Step 1: AND split

EPC element **net fragment**

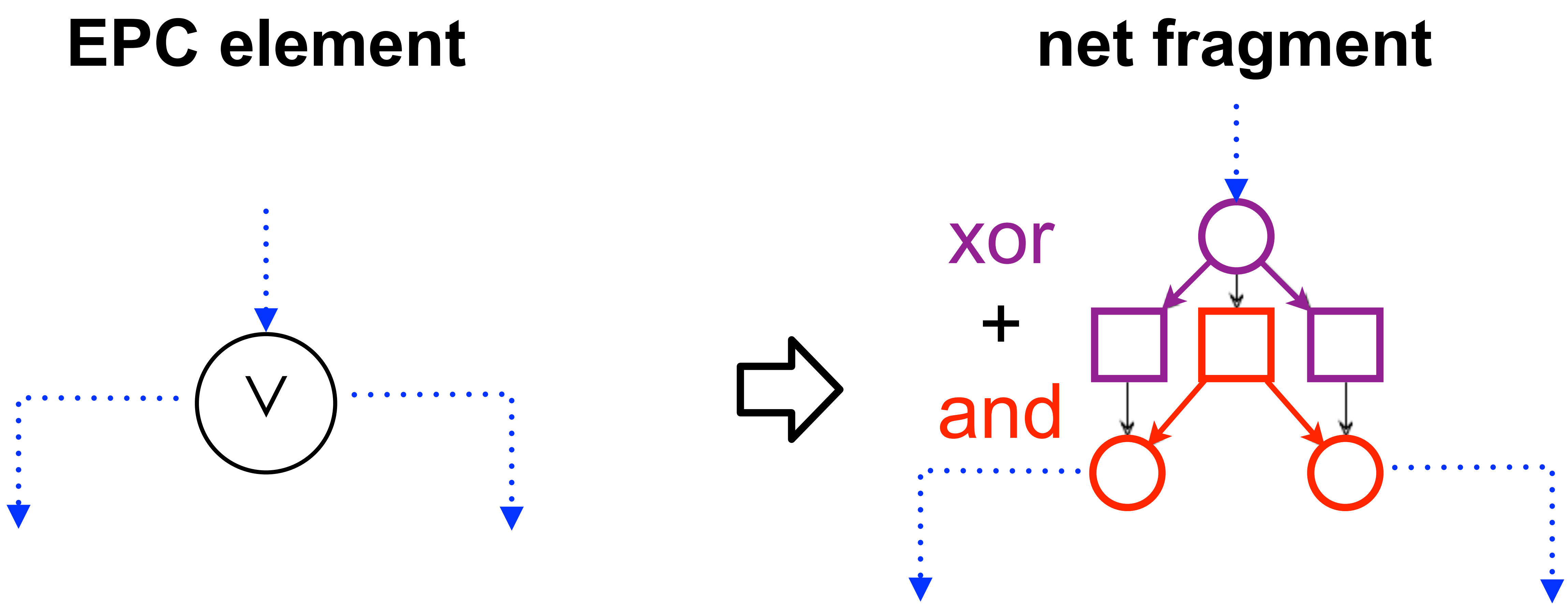


Step 1: XOR split

EPC element **net fragment**

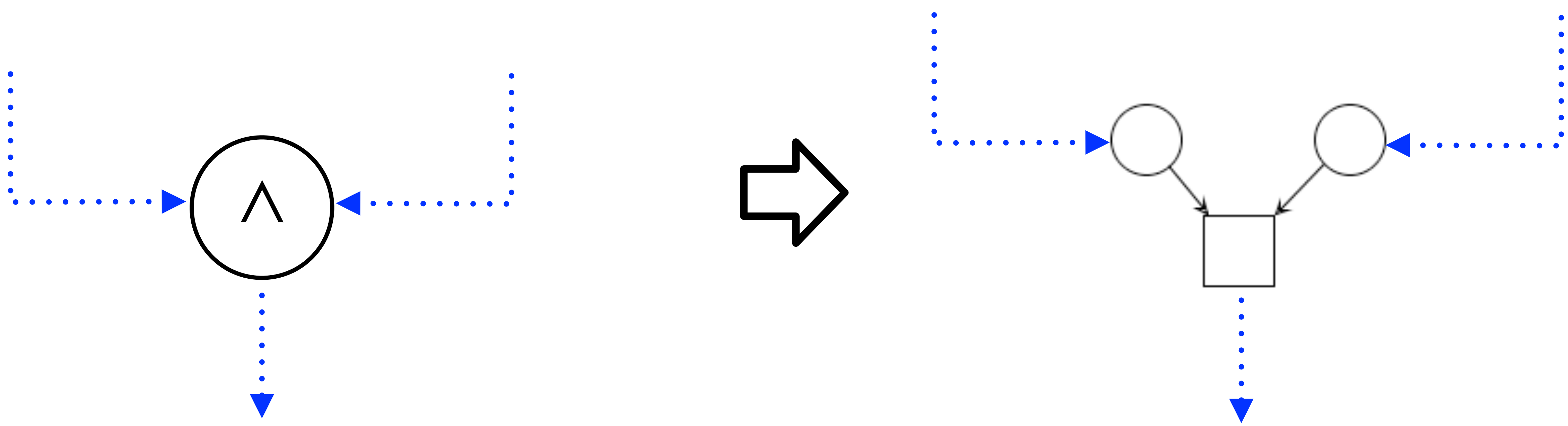


Step 1: OR split

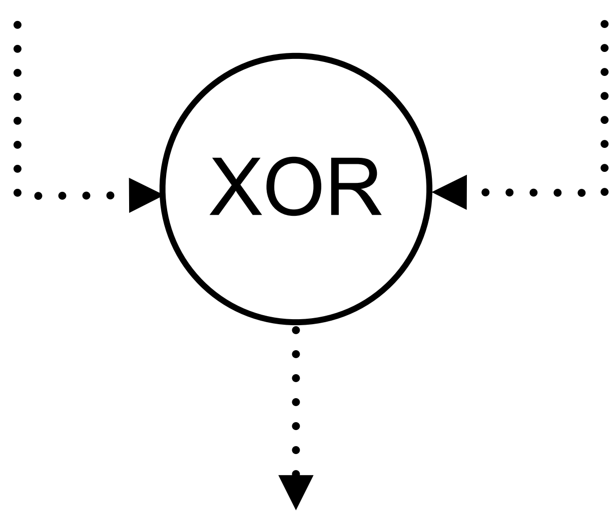


Step 1: AND join

EPC element **net fragment**



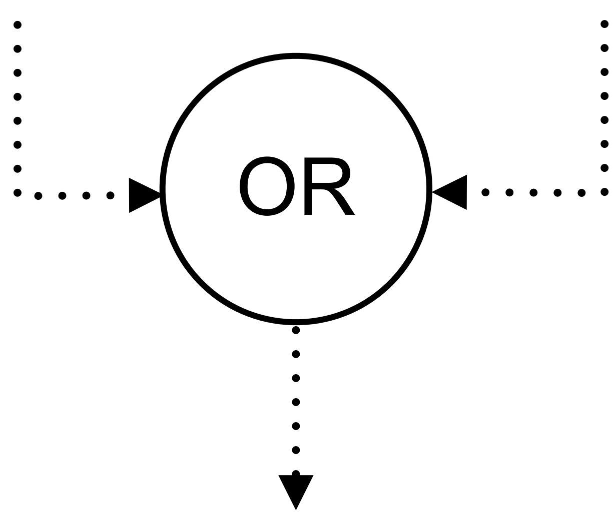
XOR join: intended meaning



if both inputs arrive,
it should block the flow

if one input arrives,
it cannot proceed unless
it is informed that
the other input will never arrive

OR join: intended meaning



if only one input arrives,
it should release the flow

if both inputs arrive,
it should release only one output

if one input arrives,
it must wait until the other arrives or
it is guaranteed that the other will never arrive

OR join: assumption

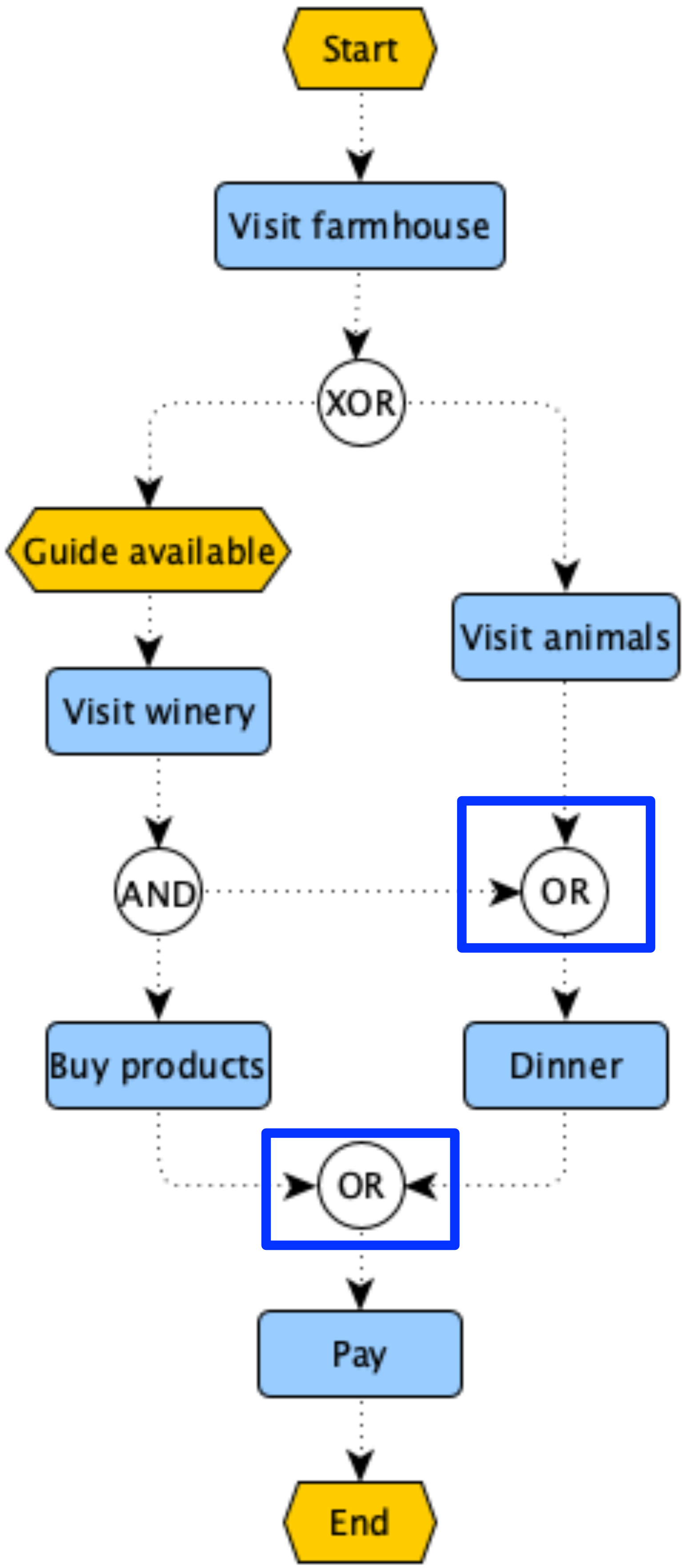
If an OR join has a **matching split**, its semantics is **wait-for-all**: wait for the completion of all *activated* paths

Otherwise, also other policies can be chosen:

every-time: trigger the outgoing path on each input

first-come: wait for the first input and ignore the second

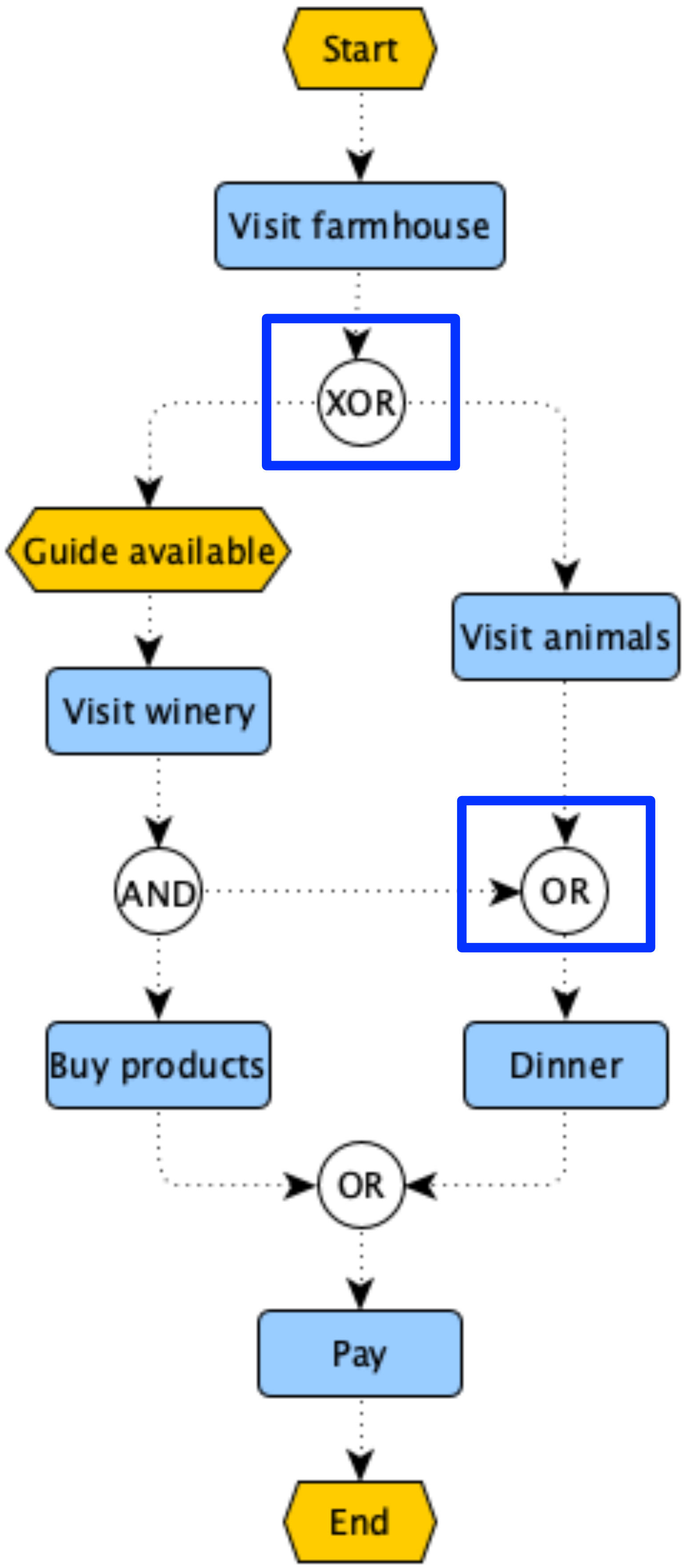
Assumption: every OR join is tagged with a policy (some suggested to have different trapezoid symbols)



Example

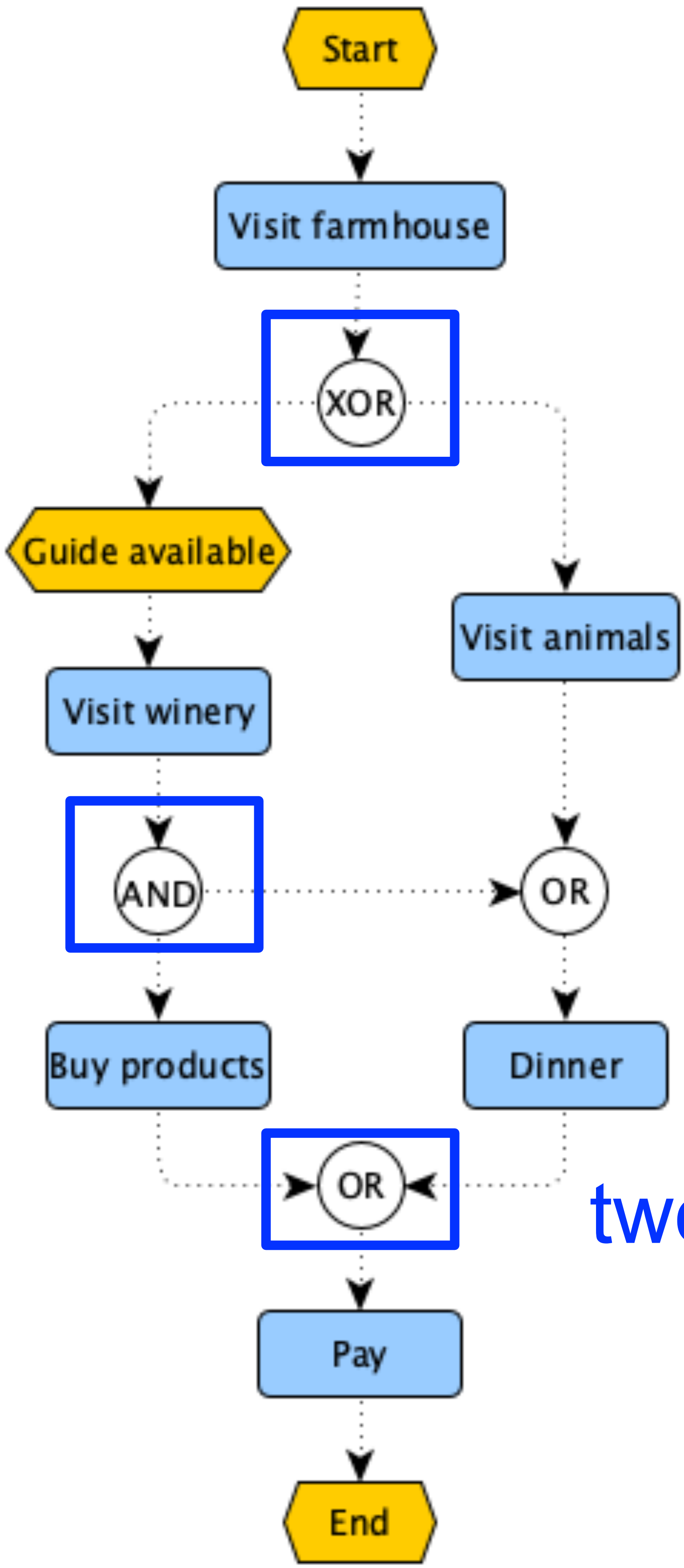
two OR joins
but no OR split

Example



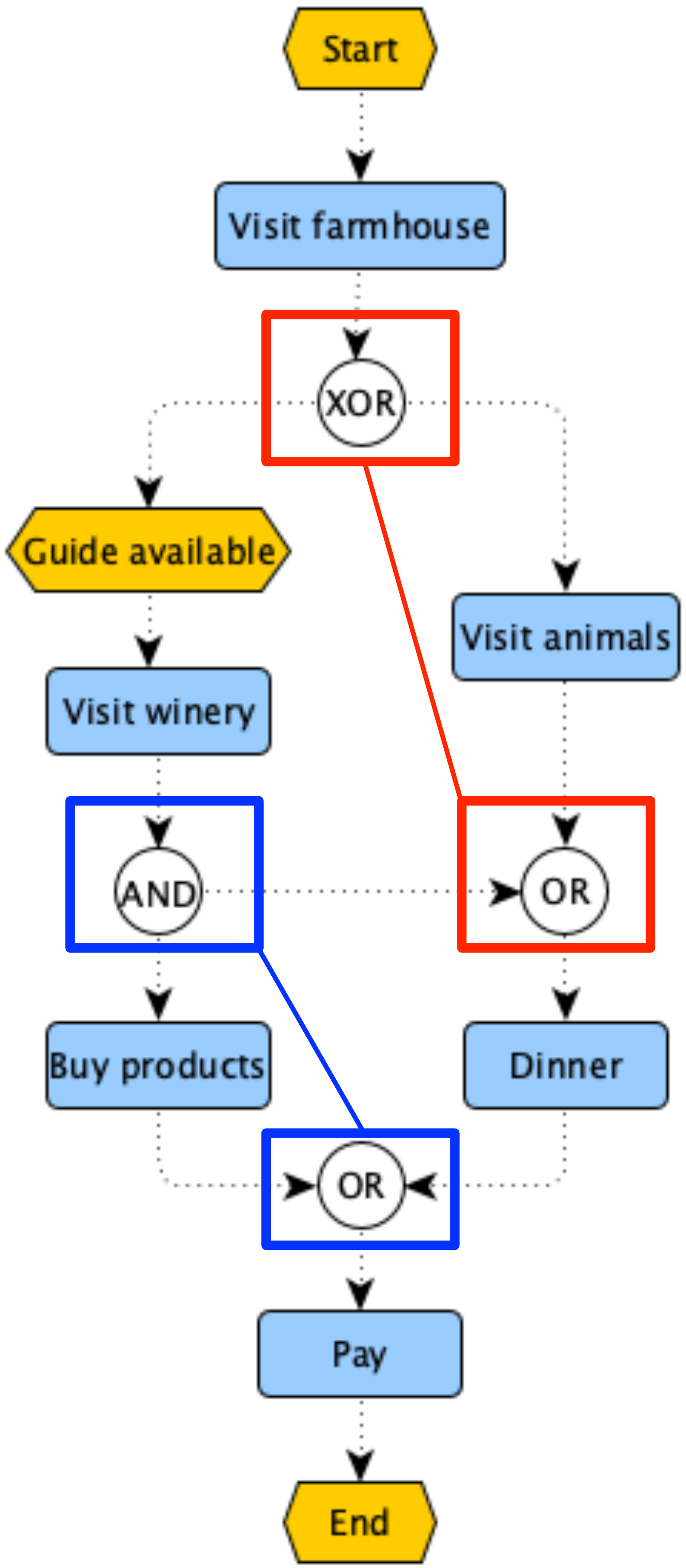
only one
candidate split

Example



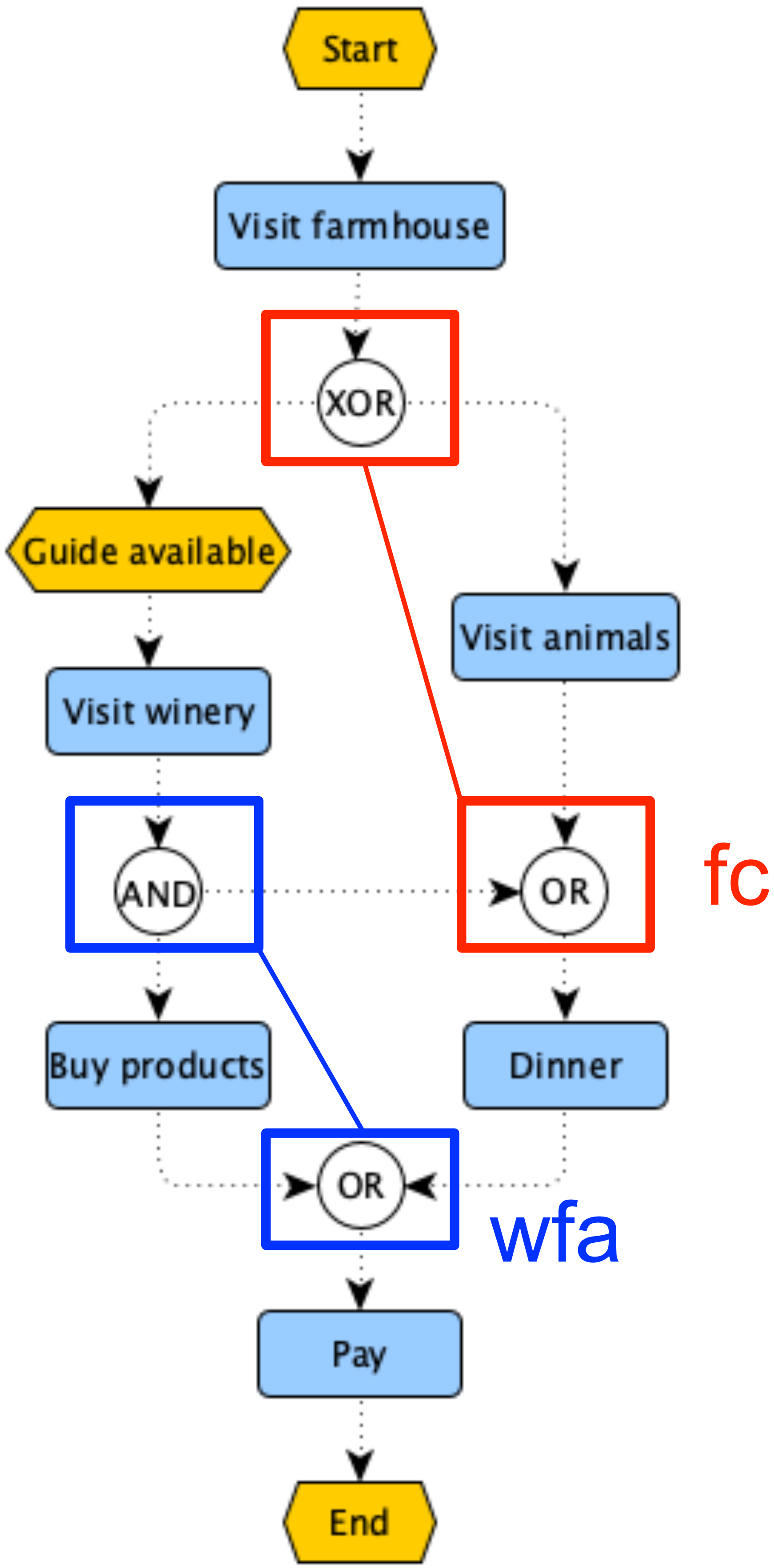
two candidate splits

Example



assign corresponding splits

Example



assign policies

Assumption

…

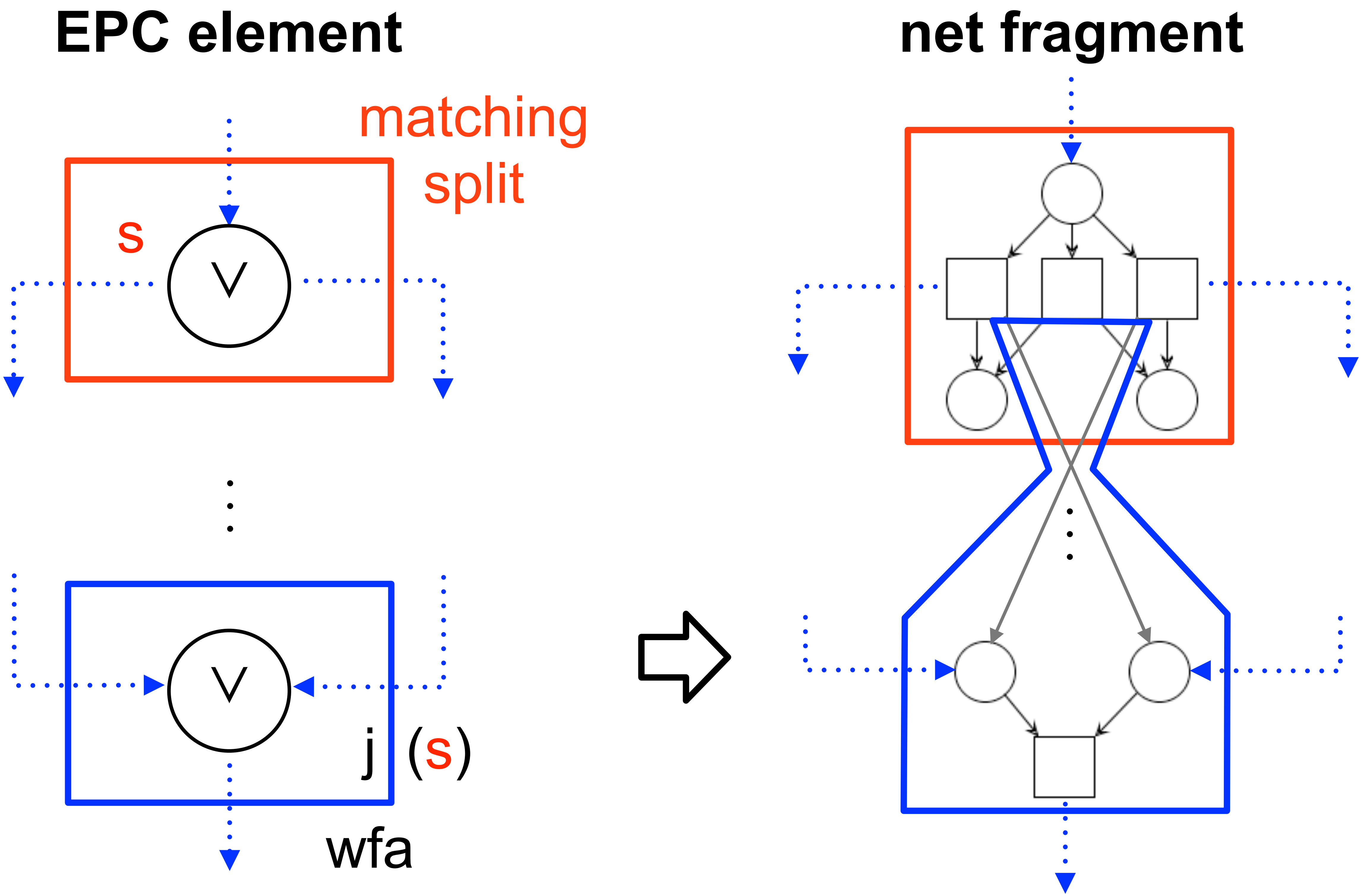
An OR join with **matching split uses wfa**

If an OR join has non-matching corresponding split
it is decorated with a policy (wfa, fc, et)

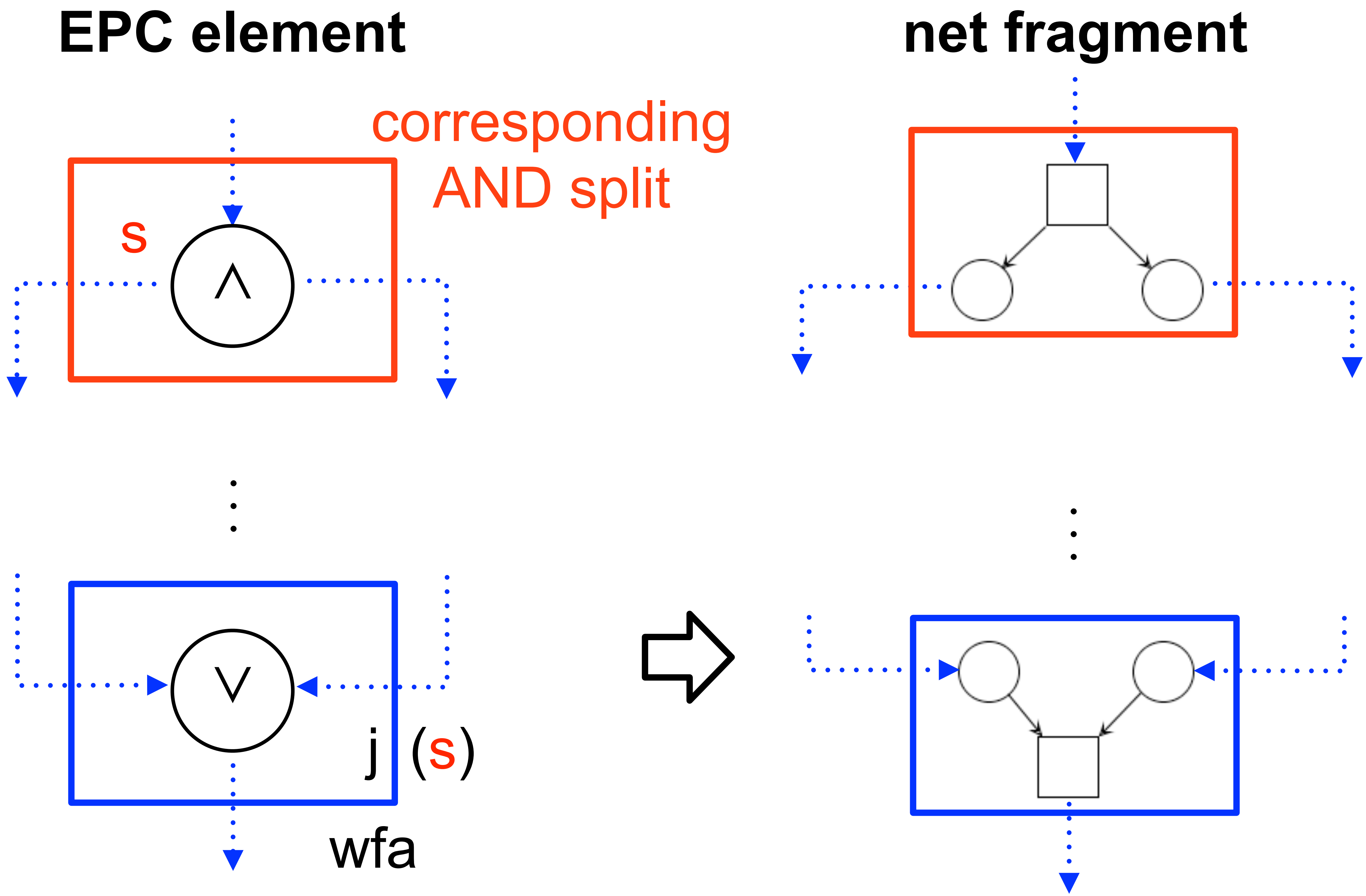
wfa: wait-for-all
works well with any corresponding split

…

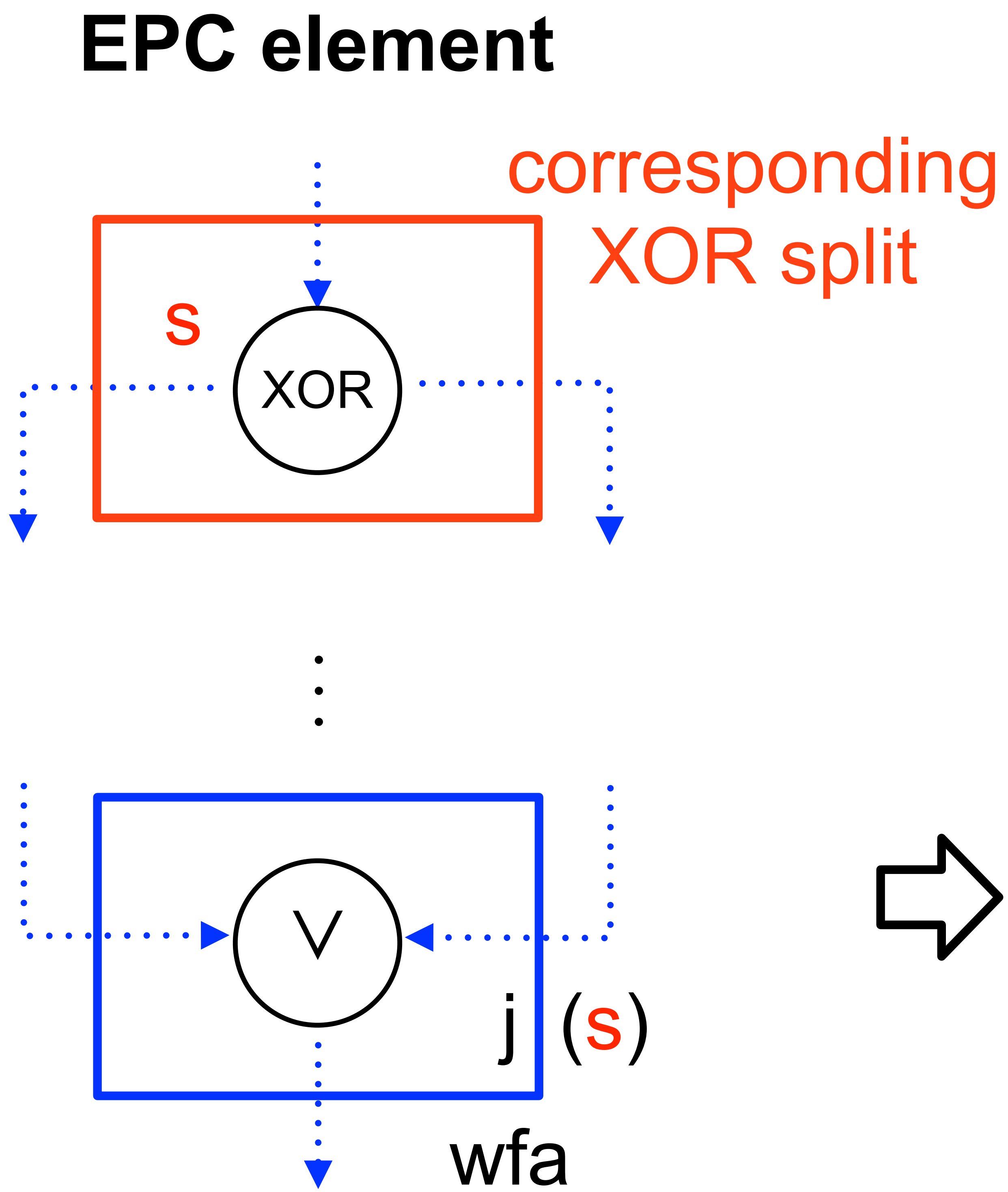
Step 1: OR join (wfa)

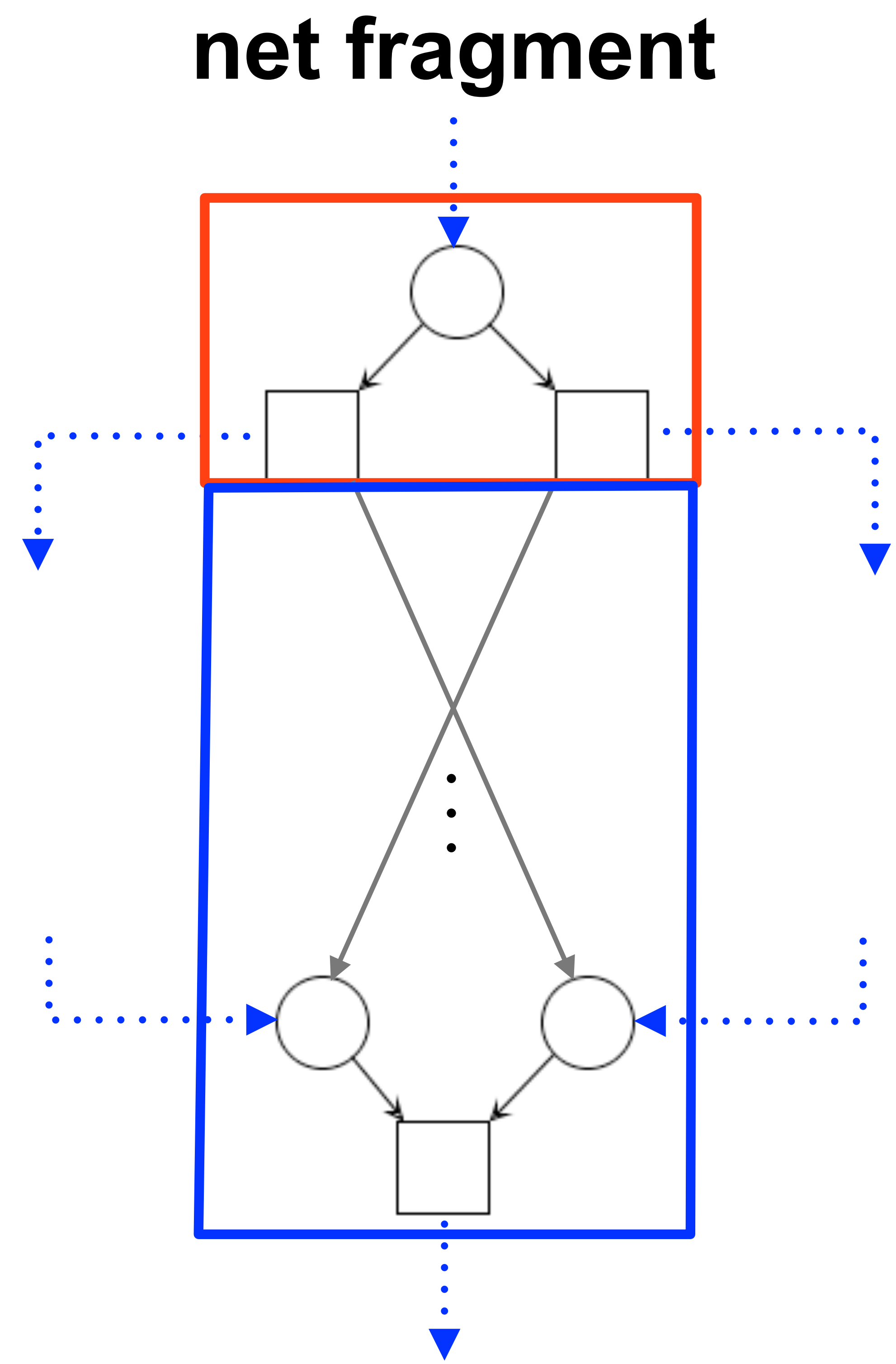


Step 1: OR join (wfa)



Step 1: OR join (wfa)





Assumption

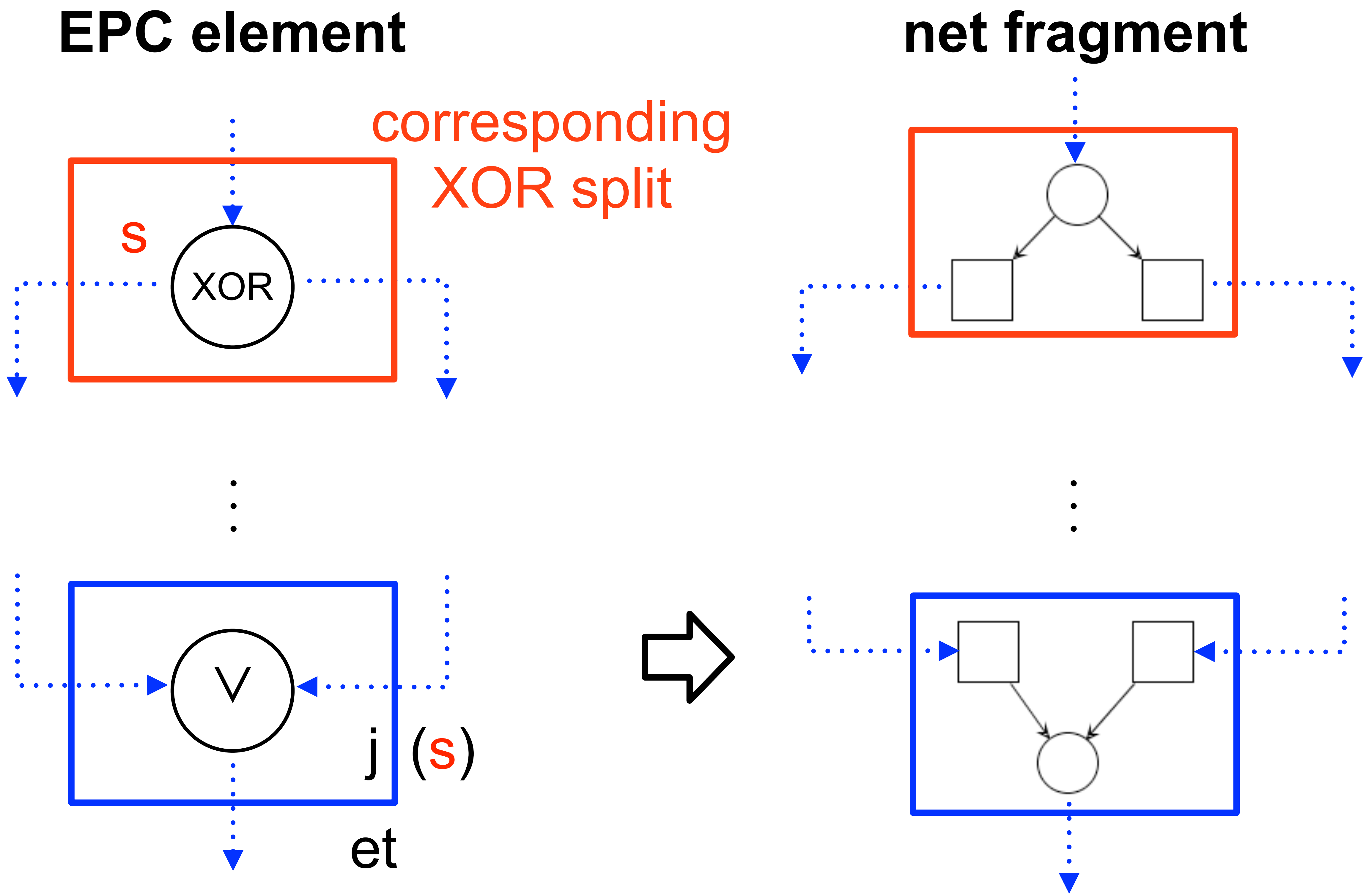
…

If an OR join has non-matching corresponding split
it is decorated with a policy (wfa, fc, et)

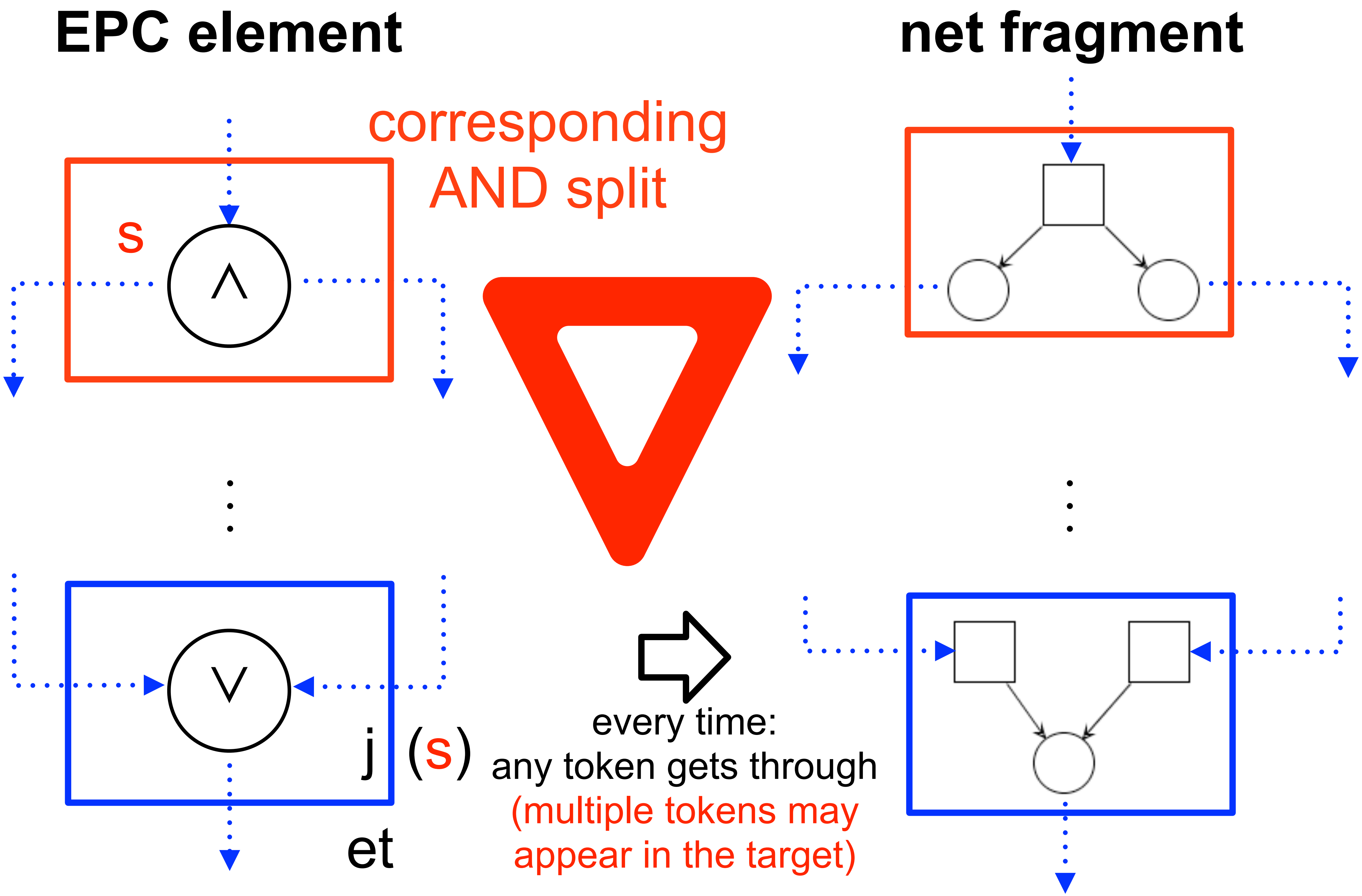
et: every-time
works well with corresponding XOR split

…

Step 1: OR join (et)



Step 1: OR join (et)



Assumption

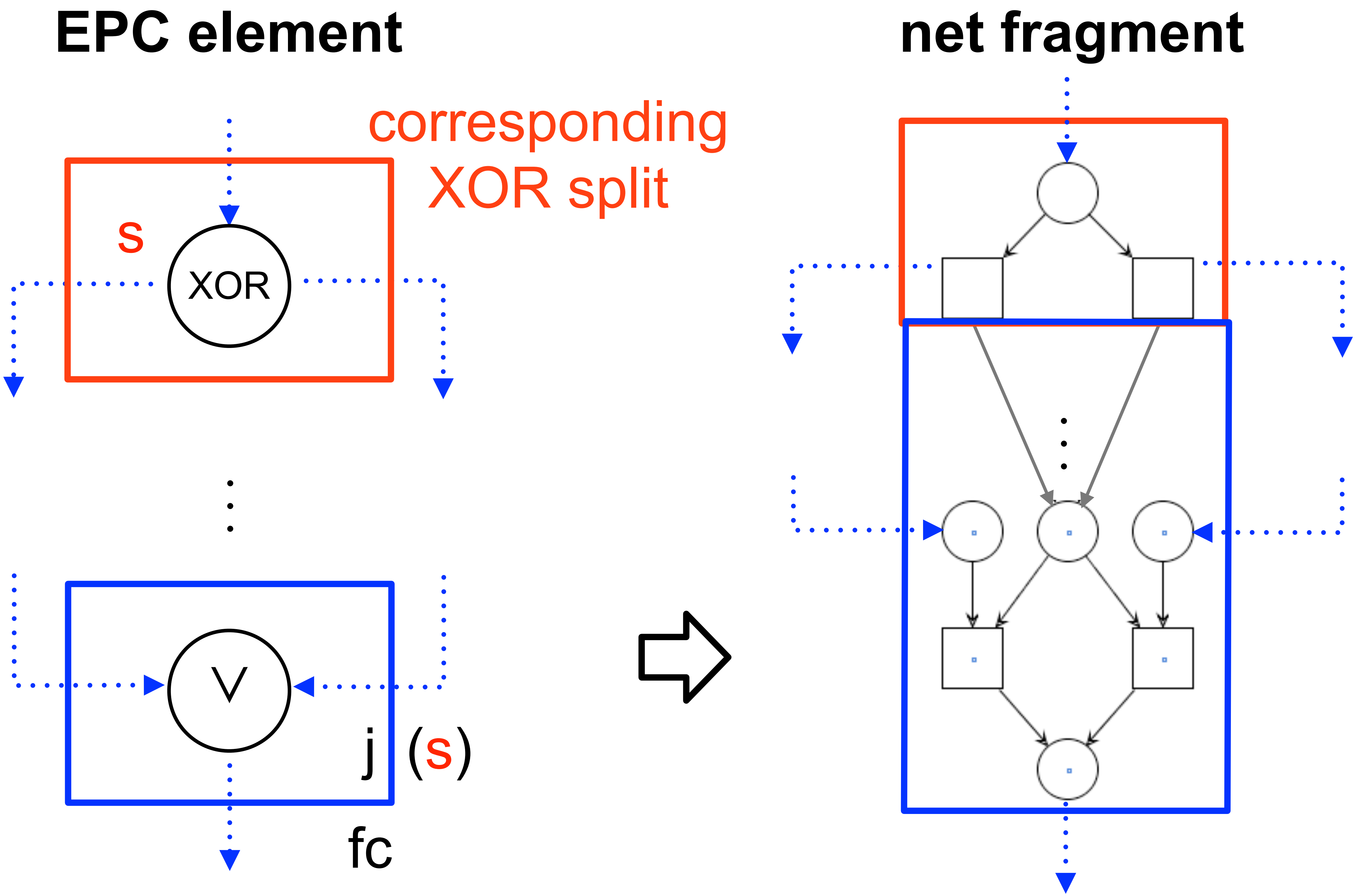
…

If an OR join has non-matching corresponding split
it is decorated with a policy (wfa, fc, et)

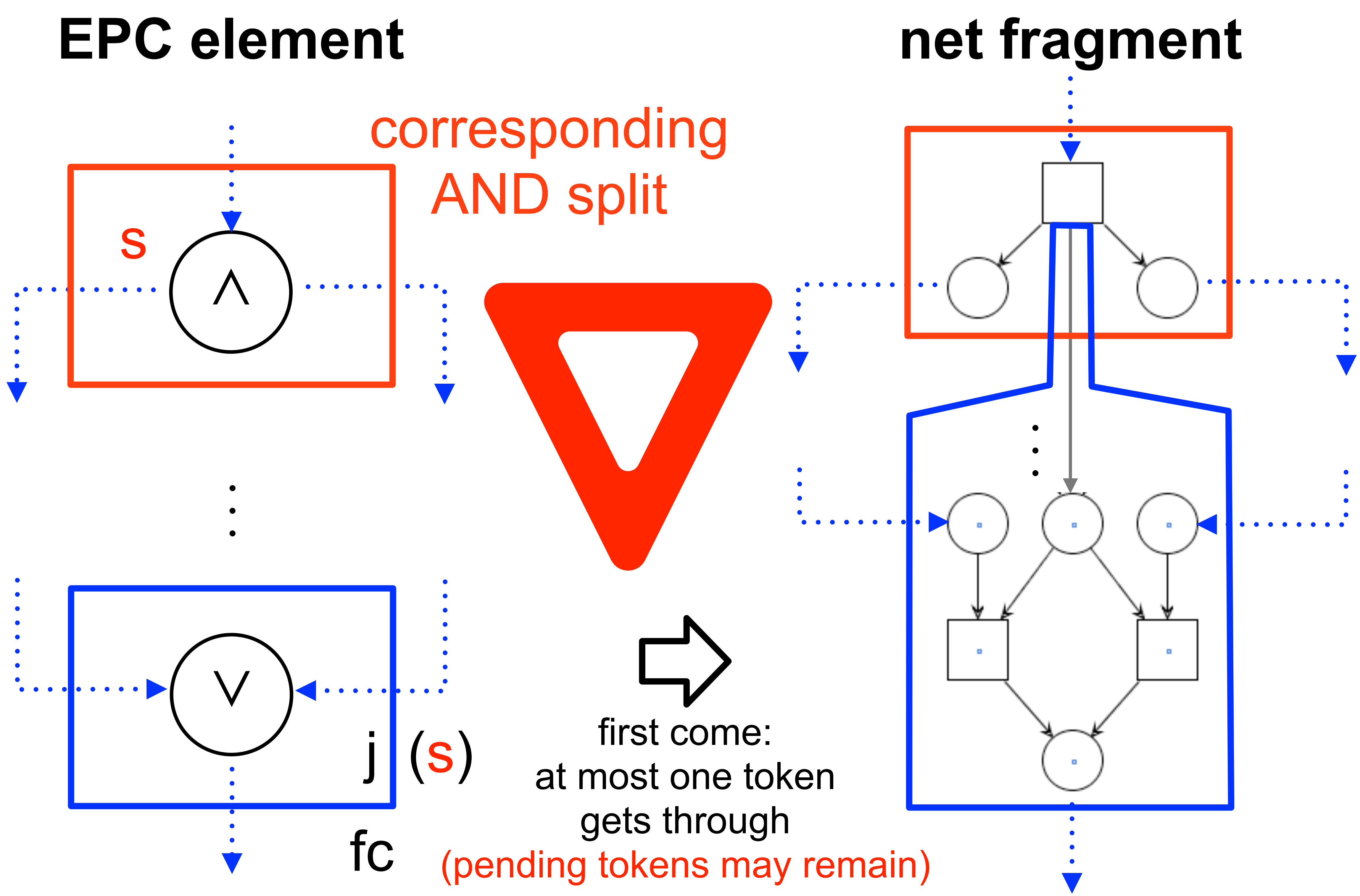
fc: first-come
works well with corresponding XOR split

…

Step 1: OR join (fc)



Step 1: OR join (fc)



XOR join: assumption

If a XOR join has a **matching split**, the semantics is:
“it blocks if both paths are activated and
it is triggered by a unique activated path”

Any policy (wait-for-all, first-come, every-time)
contradicts the exclusivity of XOR
(a token from one path can be accepted only if we make
sure that no second token will arrive via the other path)

Assumption: every XOR join has a matching split
(the implicit start split is allowed as a valid match)

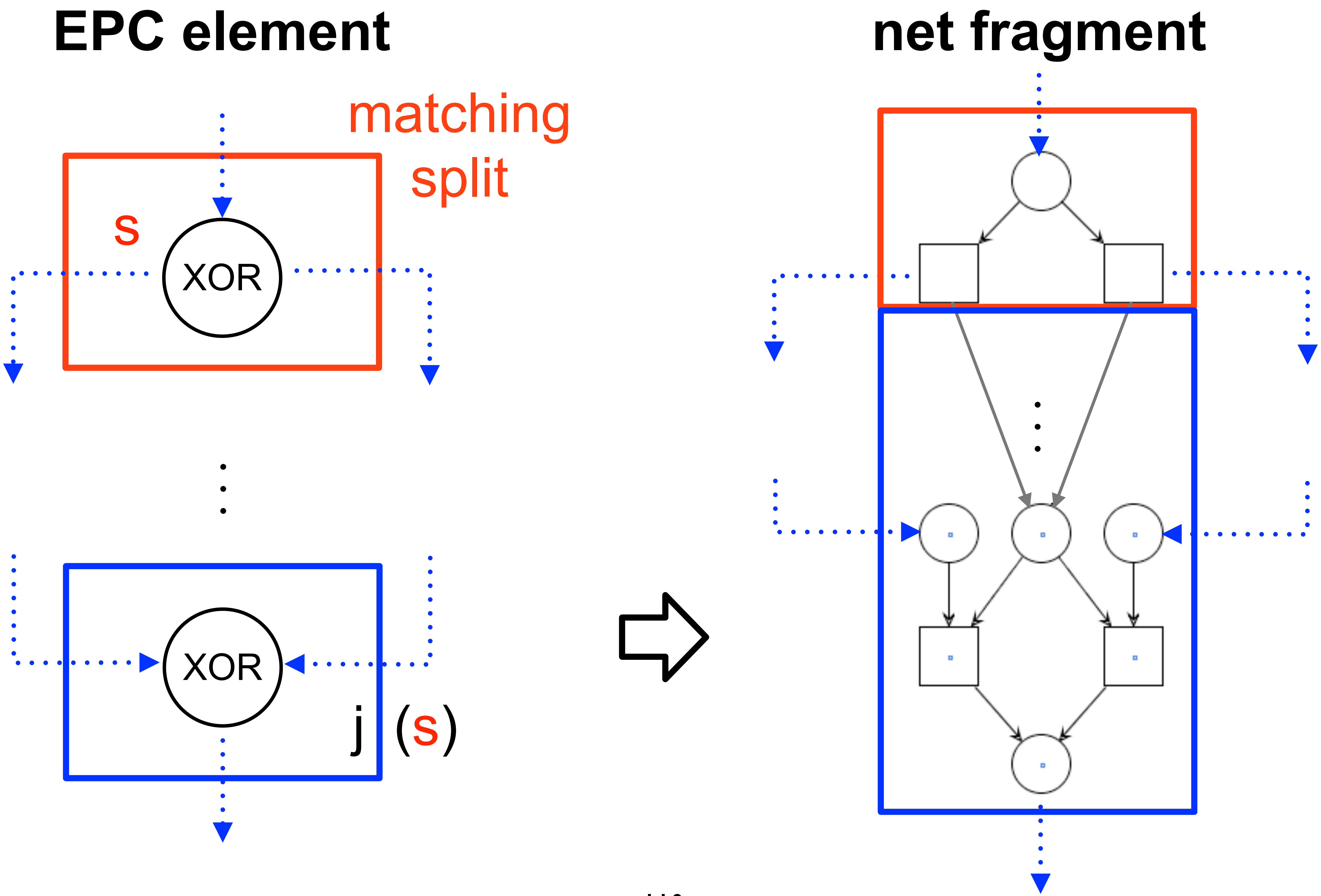
Assumption

…

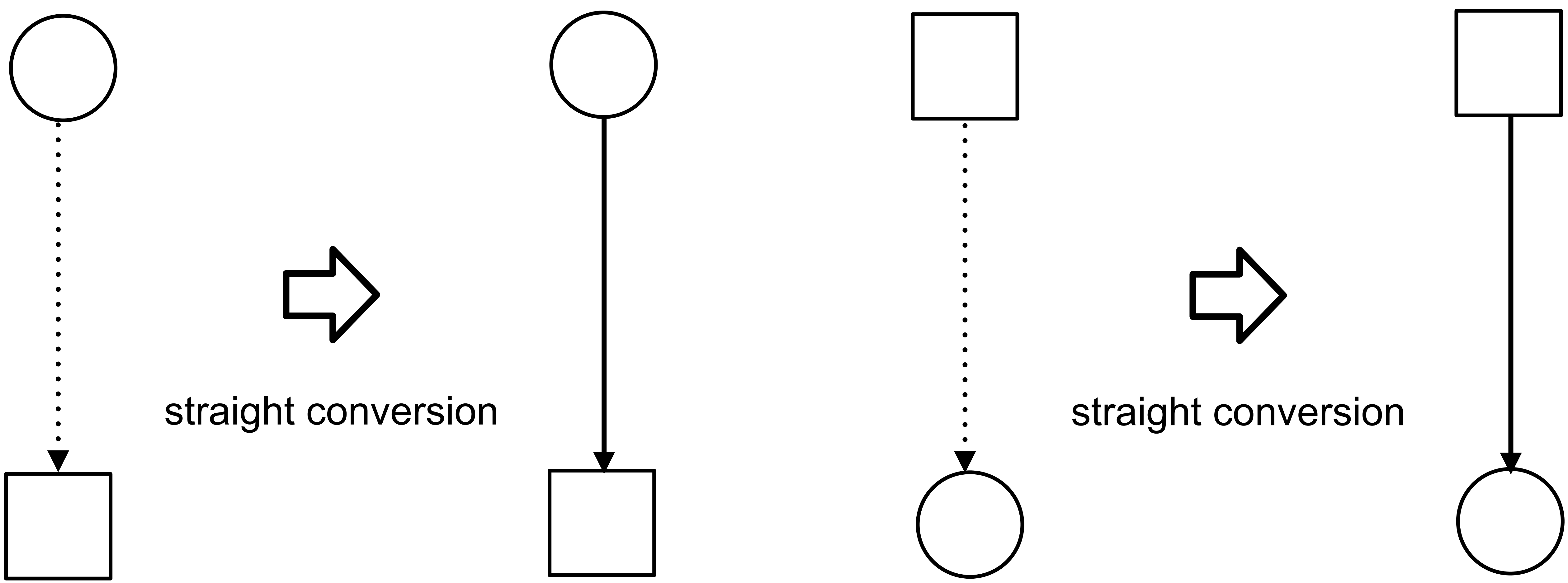
Any XOR join has a **corresponding matching split**

…

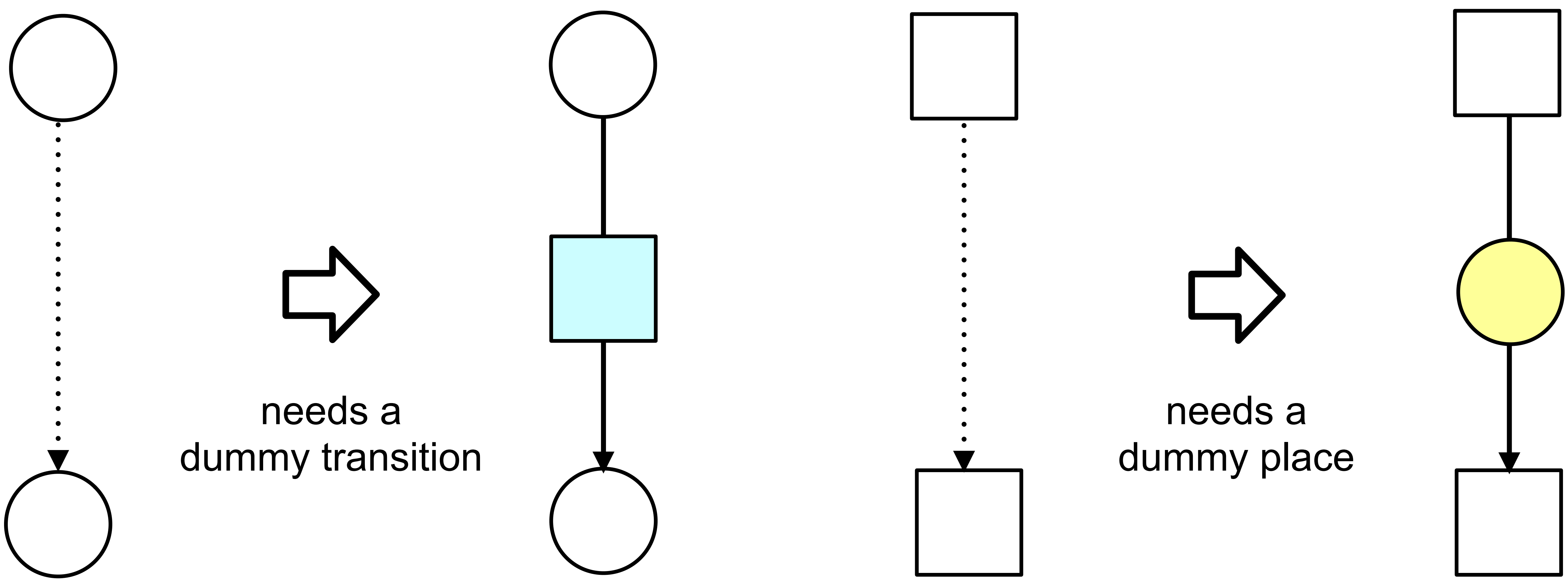
Step 1: XOR join



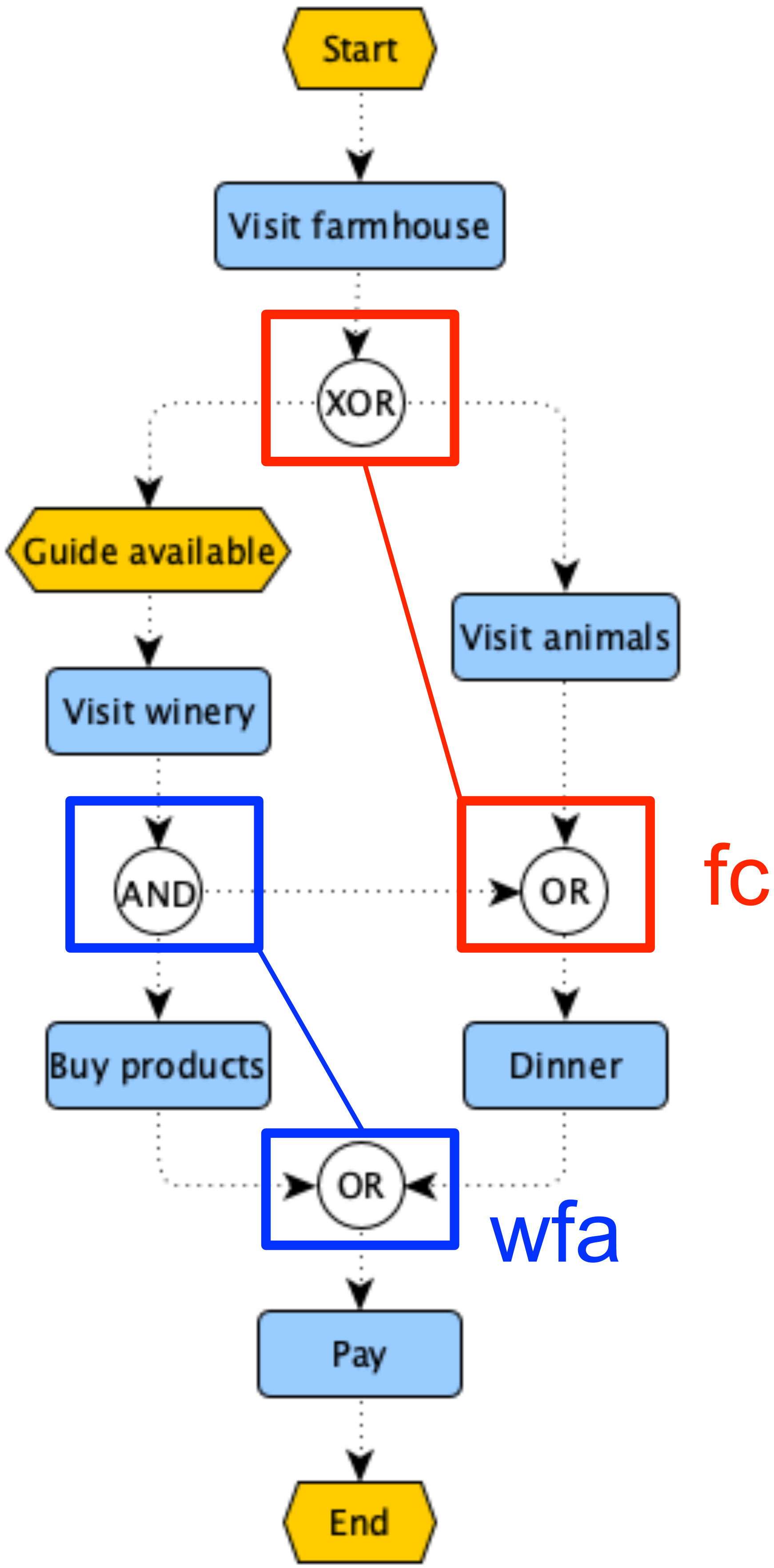
Step 2: dummy style



Step 2: dummy style

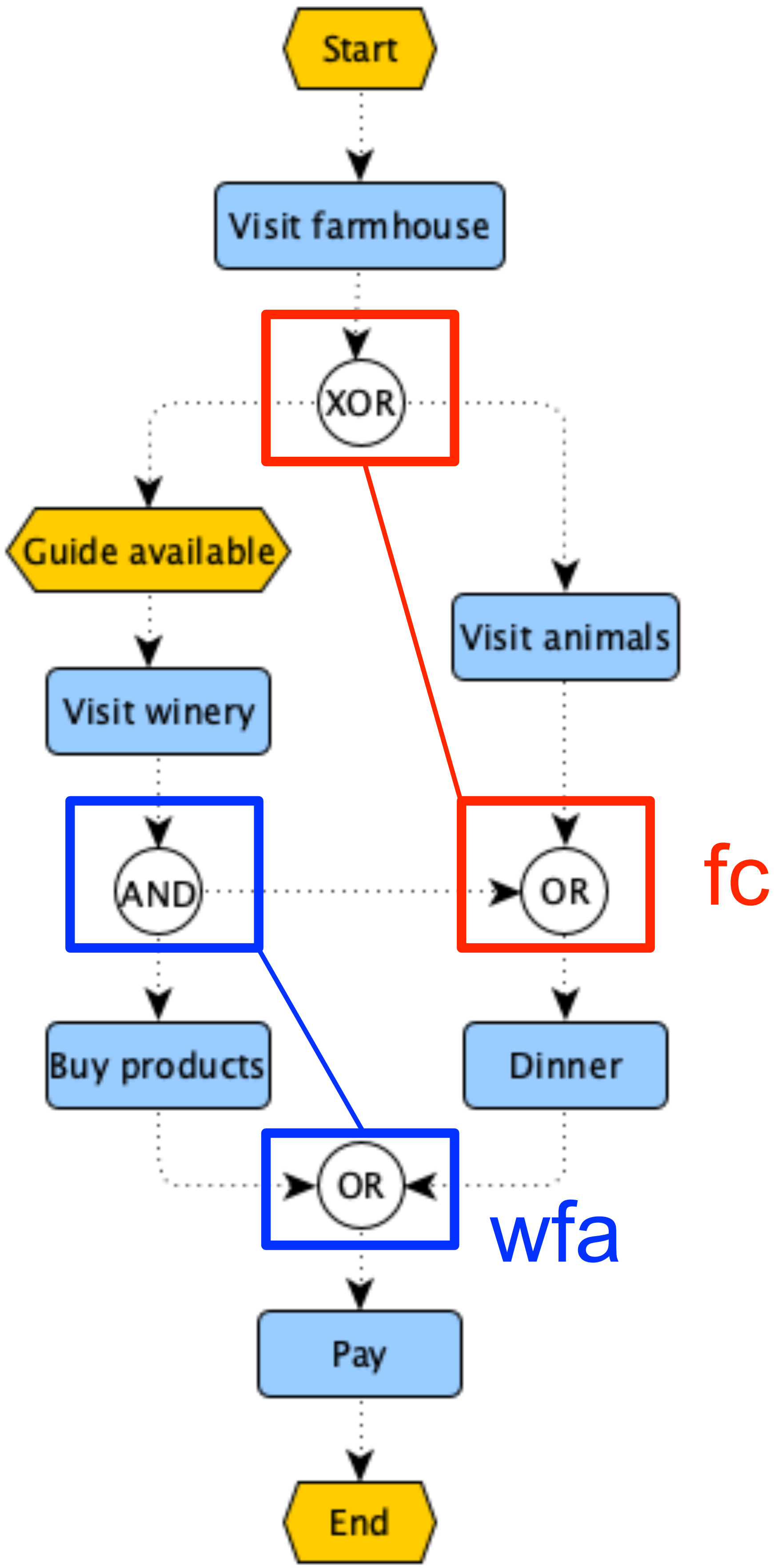


Example

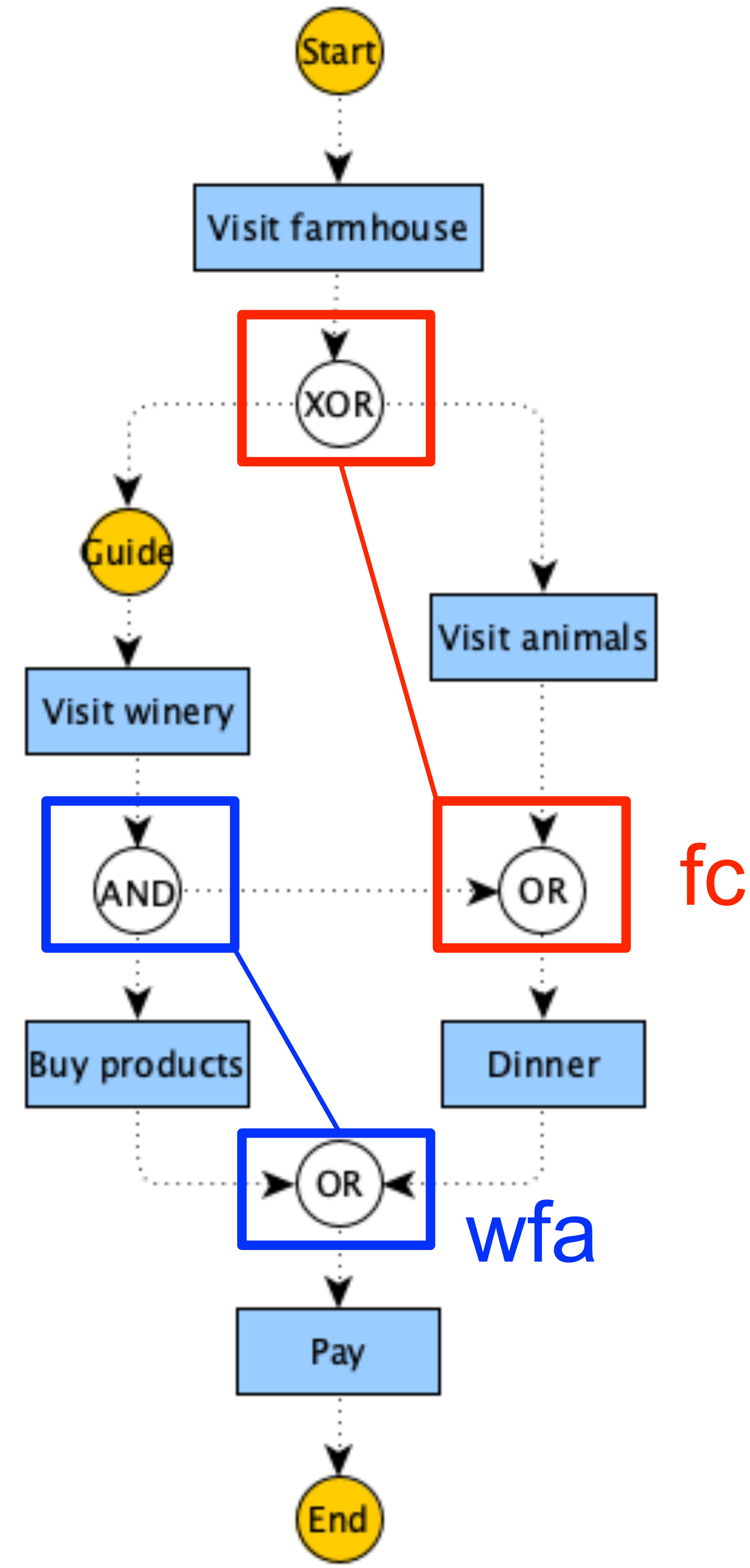


Sound?

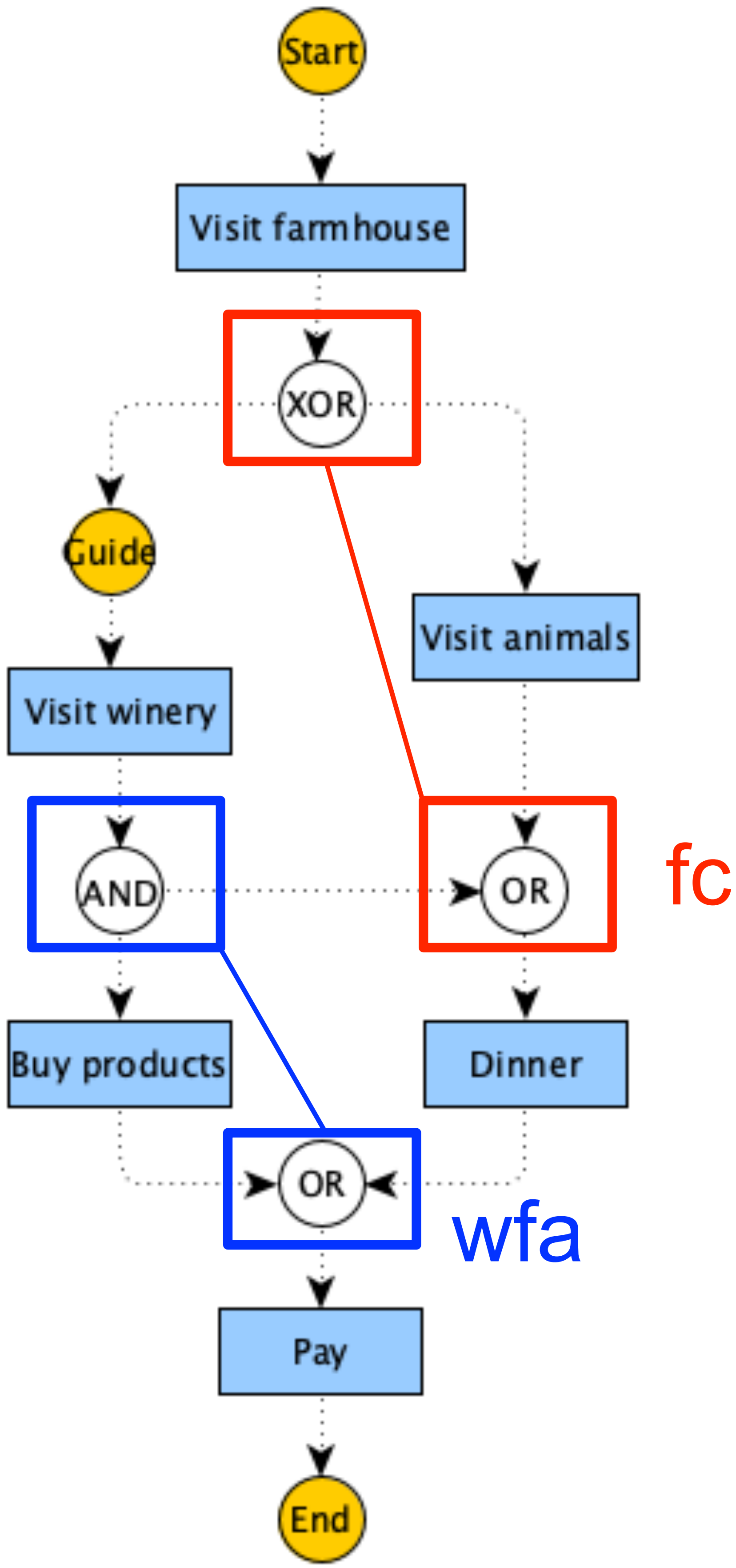
Example



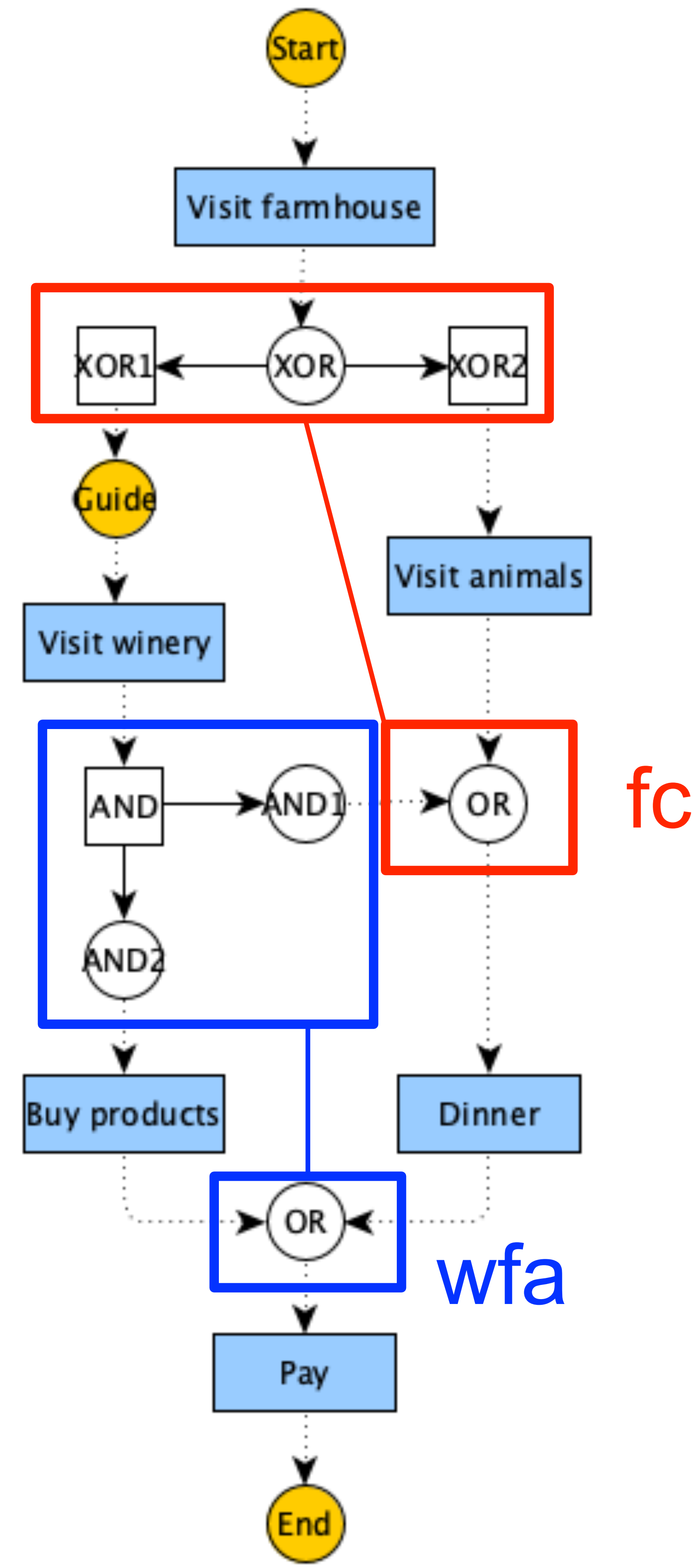
Step 1
events and
functions



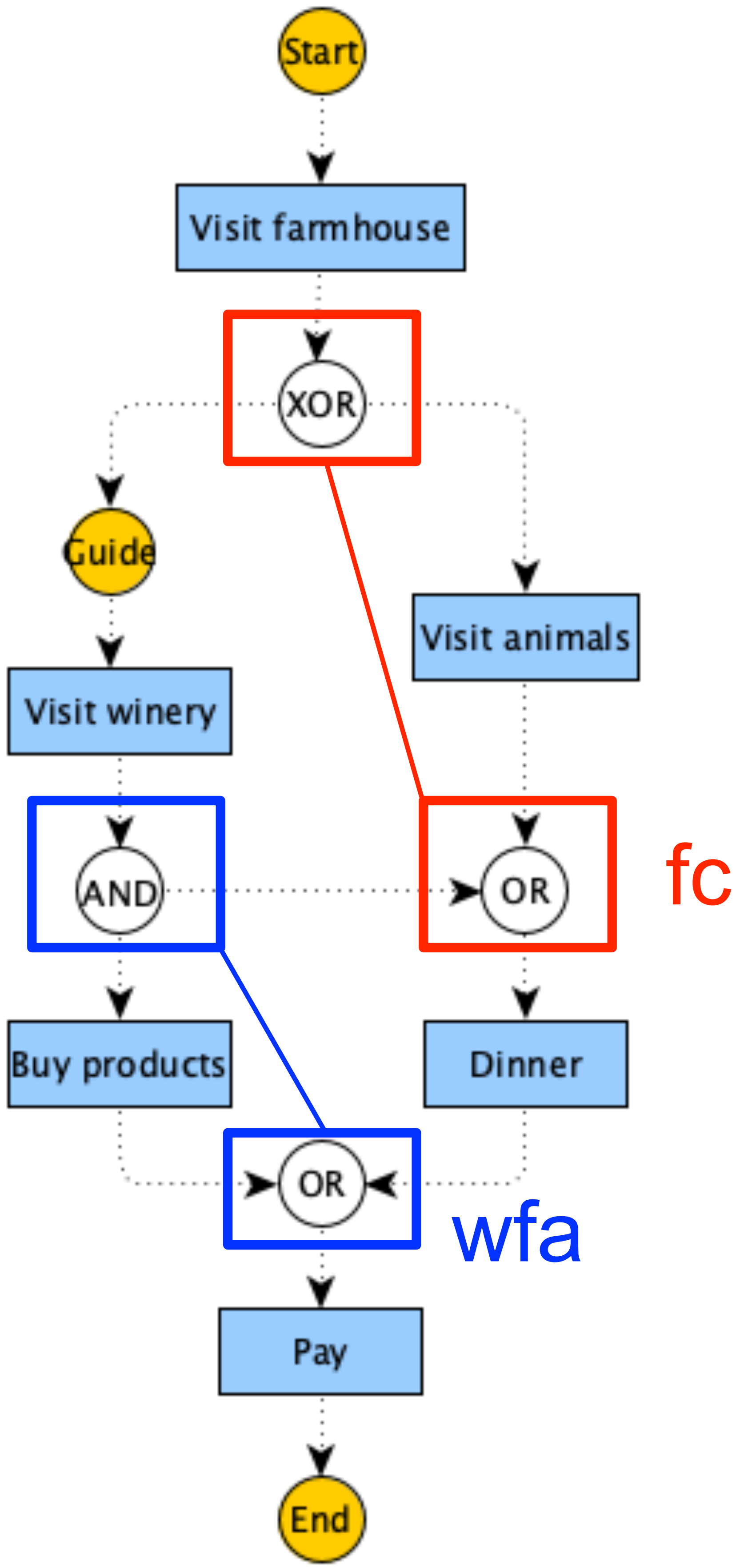
Example



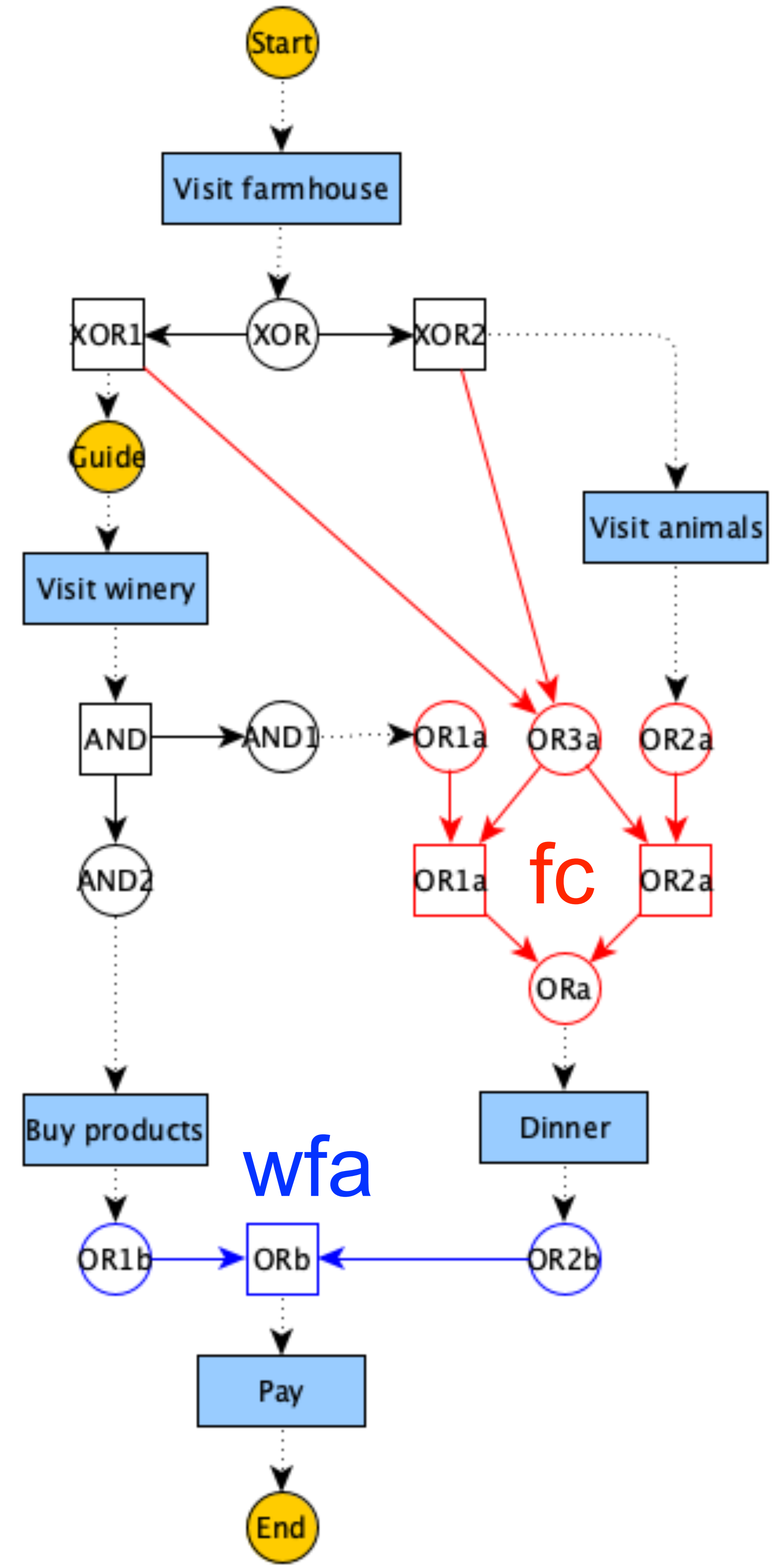
Step 1
splits



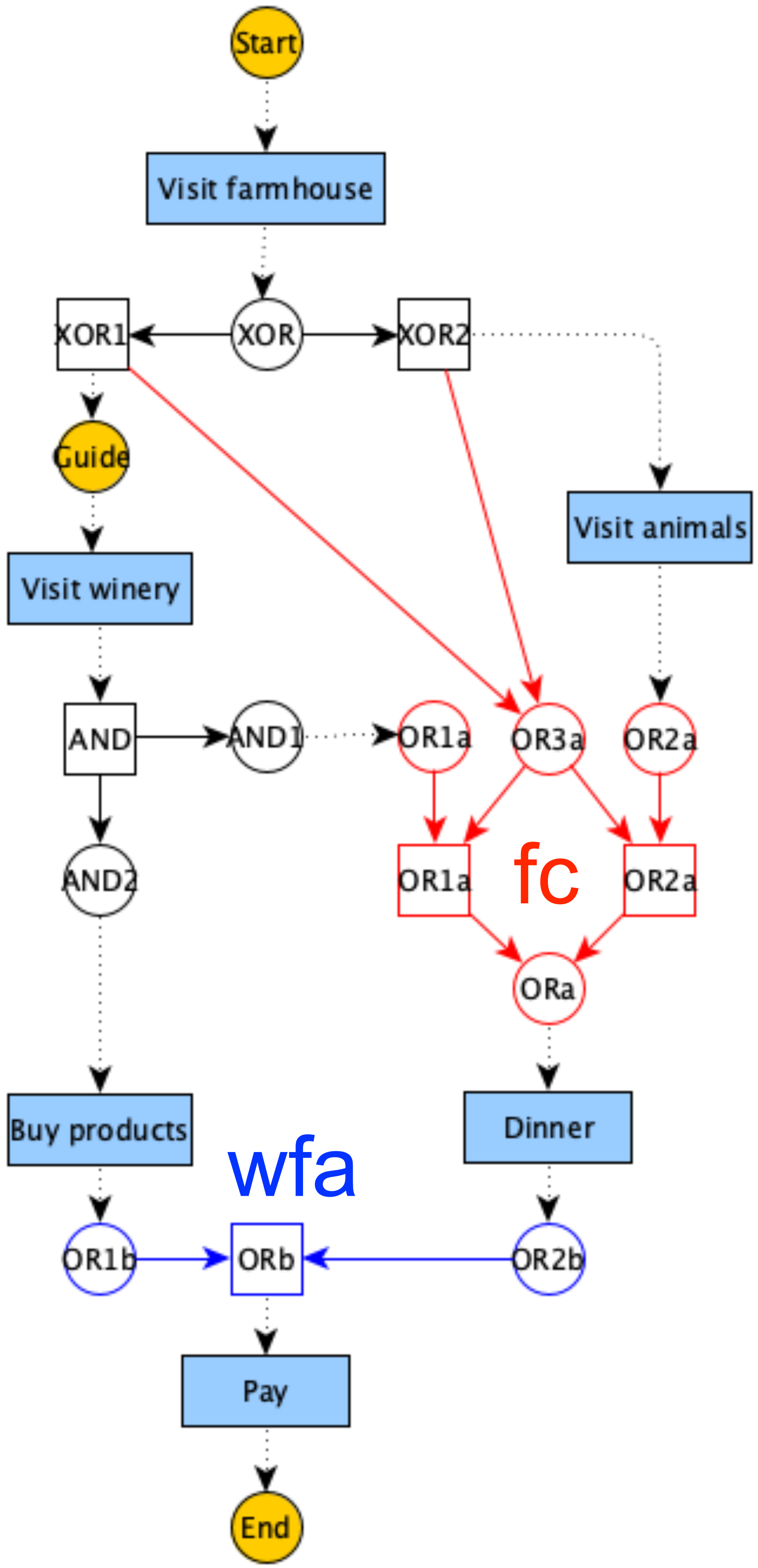
Example

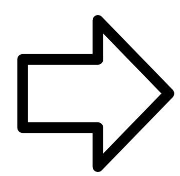


Step 1
splits and
joins

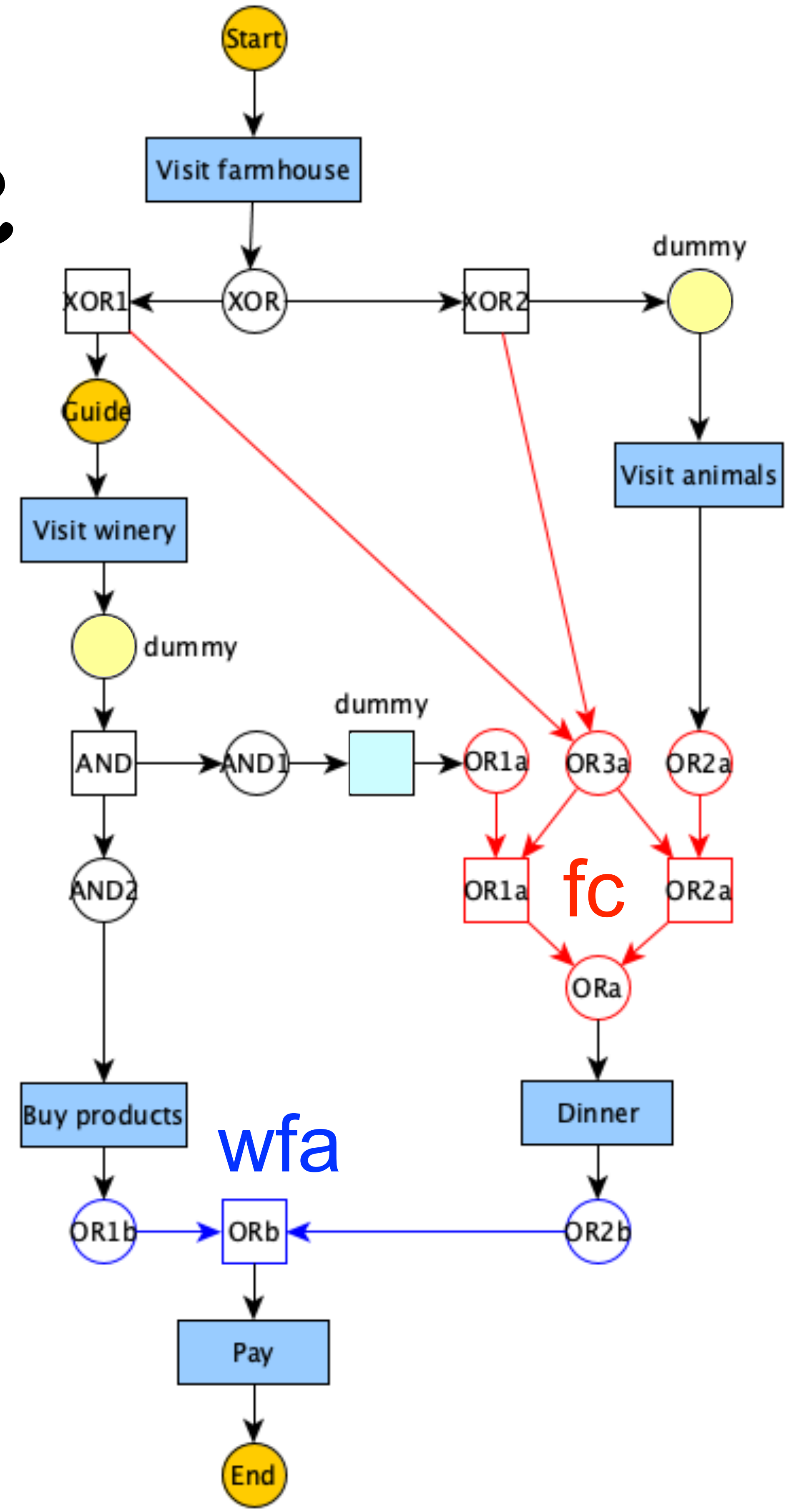


Example

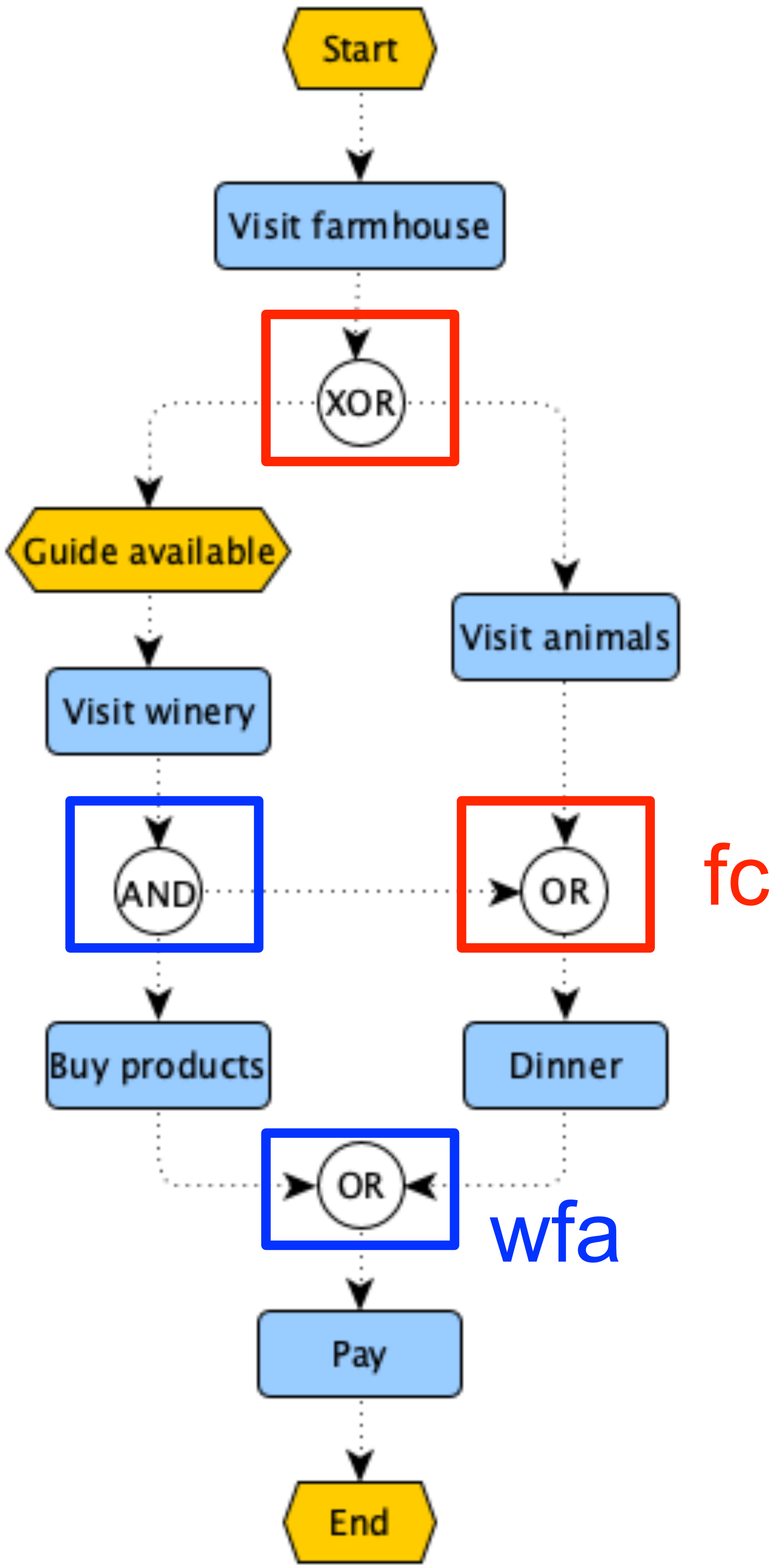




Step 2(+3)
dummy style

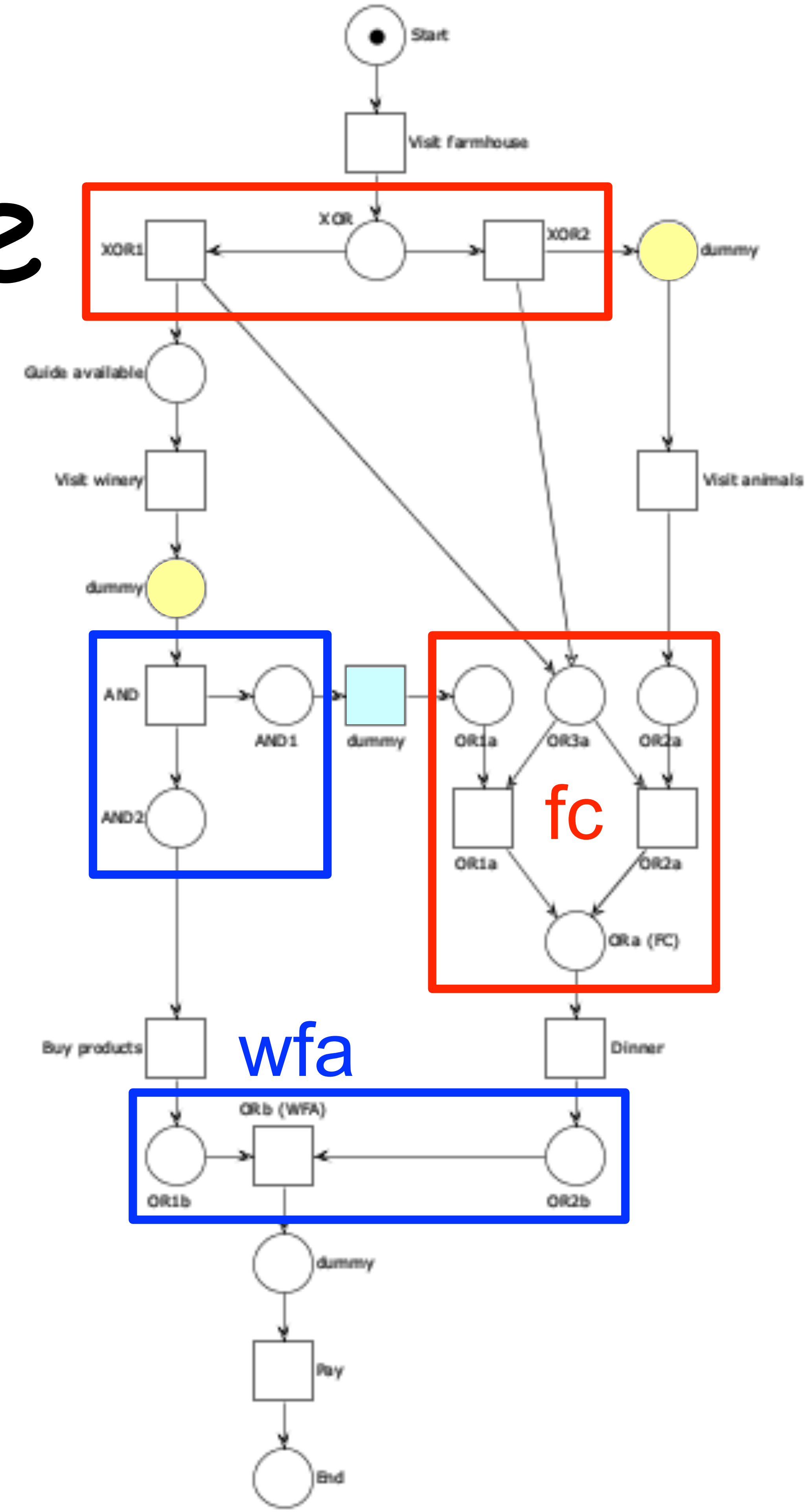


Example

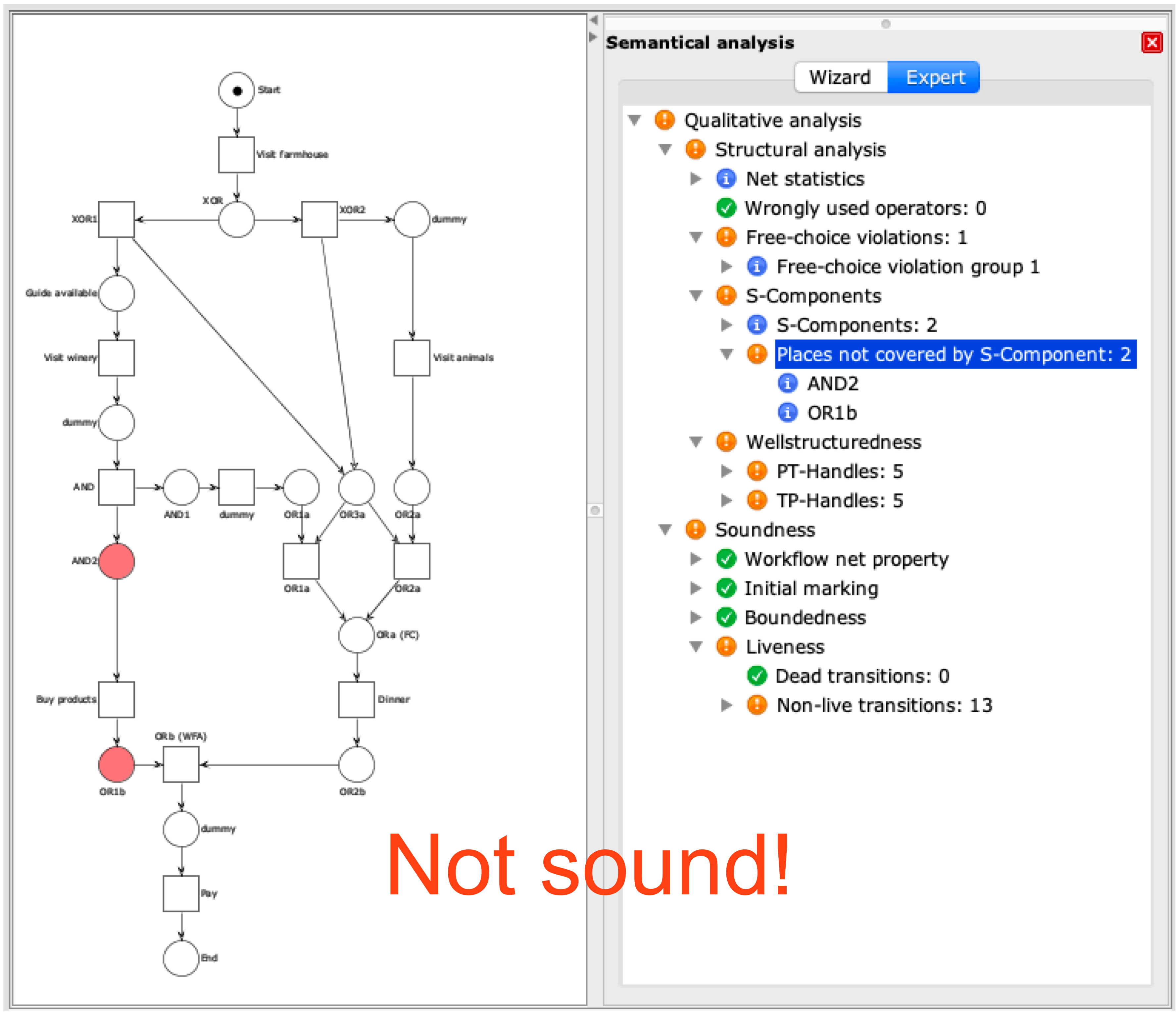


Sound?

Steps
1+2(+3)



Example



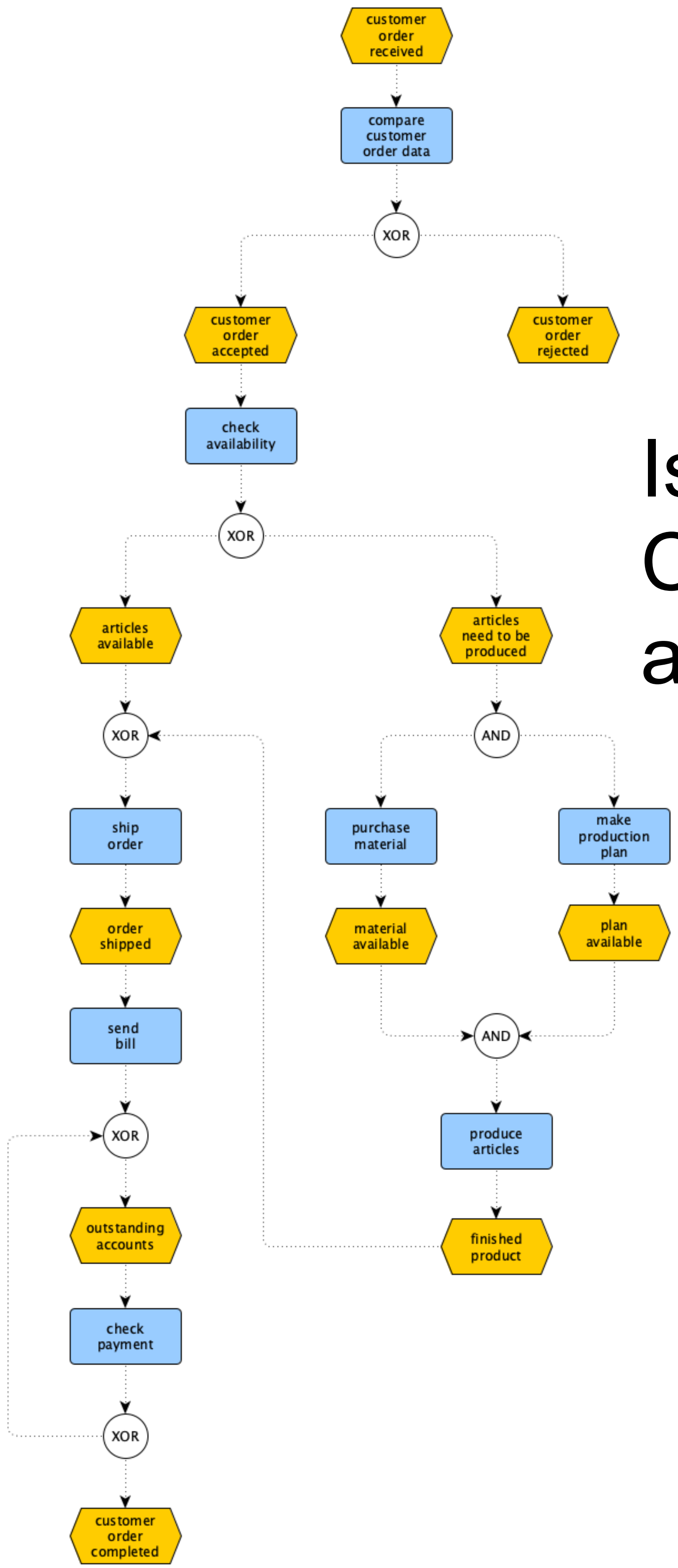
Not sound!

EPC pros and cons

You may **leave complete freedom**,
but most diagrams will not be sound

You may **constrain diagrams**,
but people like flexible syntax and ignore guidelines

You may **require to add decorations**,
but people will be lazy or misinterpret policies



Exercise

Is this EPC diagram sound?
Choose one of the three techniques seen and apply it to answer the above question