Business Processes Modelling

MPB (6 cfu, 295AA)

Roberto Bruni

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

12 - Analysis of WF nets

Object

We study suitable soundness properties of Workflow nets

Bondedness, liveness

$$(P, T, F, M_0)$$

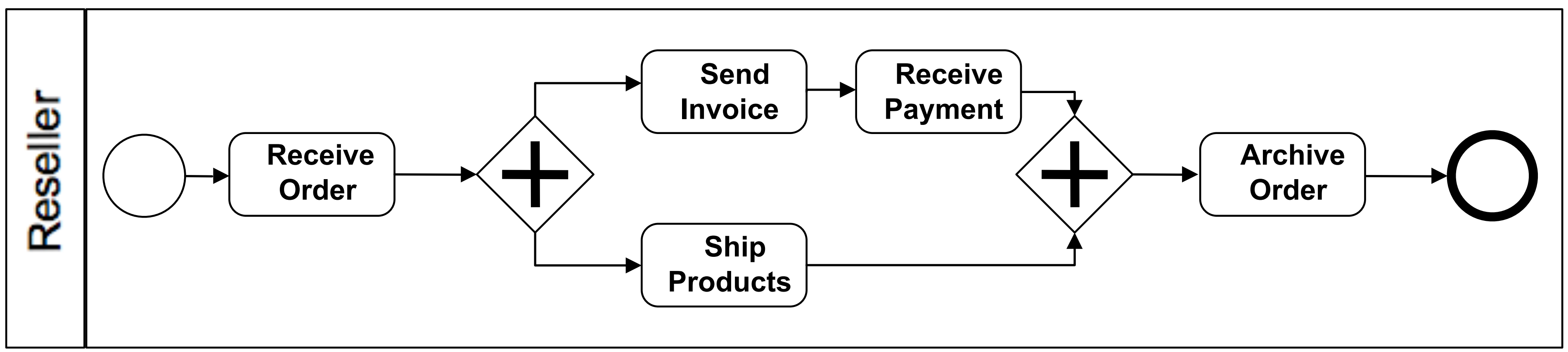
$$\exists k \in \mathbb{N}, \quad \forall p \in P, \quad \forall M \in [M_0\rangle, \quad M(p) \leq k$$

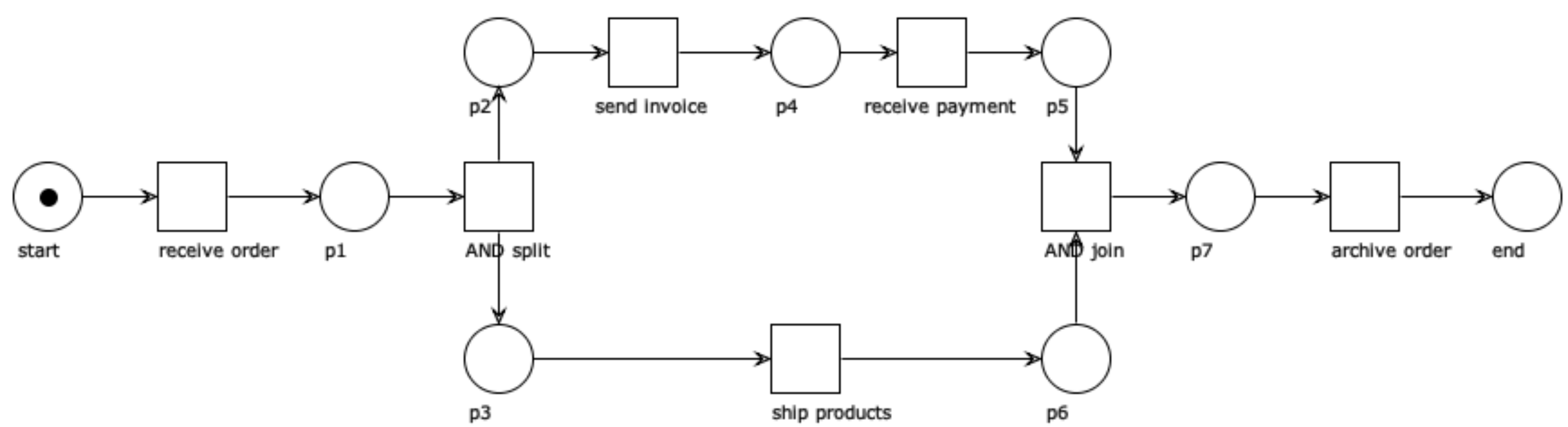
Boundedness? Liveness?

$$\forall t \in T, \quad \forall M \in [M_0\rangle, \quad \exists M' \in [M\rangle, \quad M' \xrightarrow{t}$$

Soundness informally

Example: Reseller





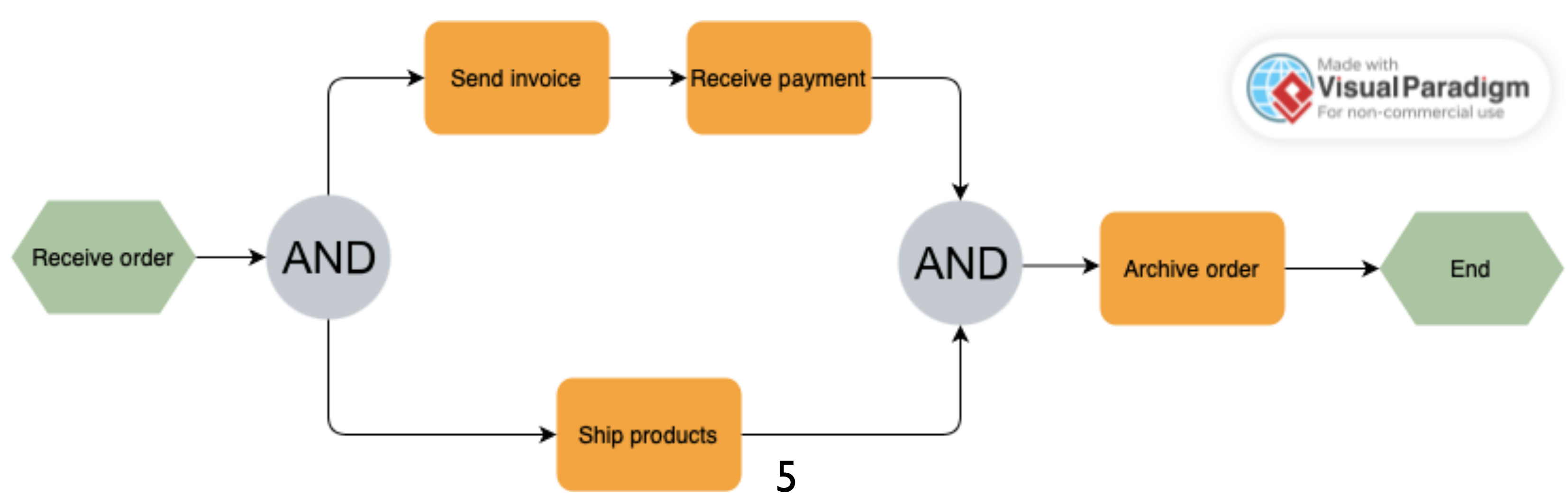
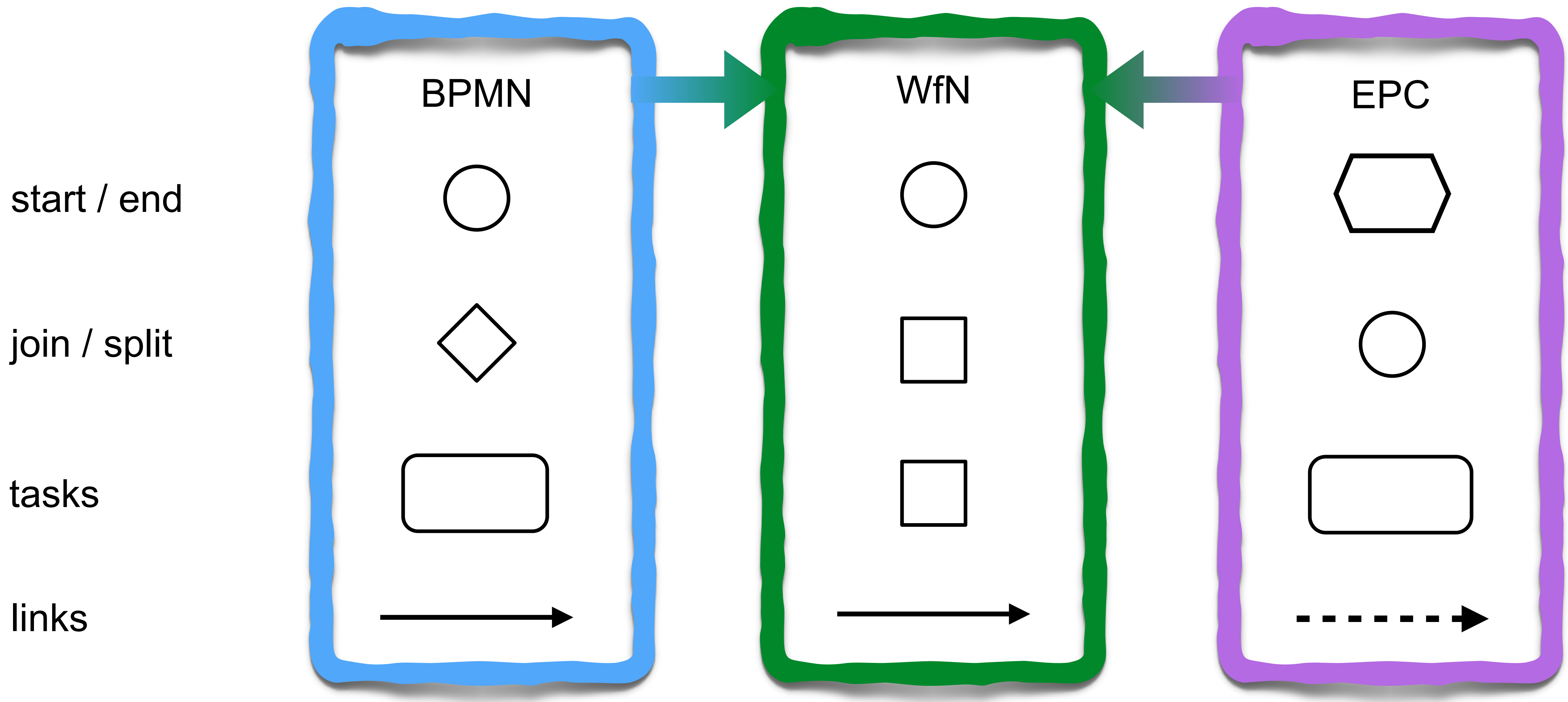
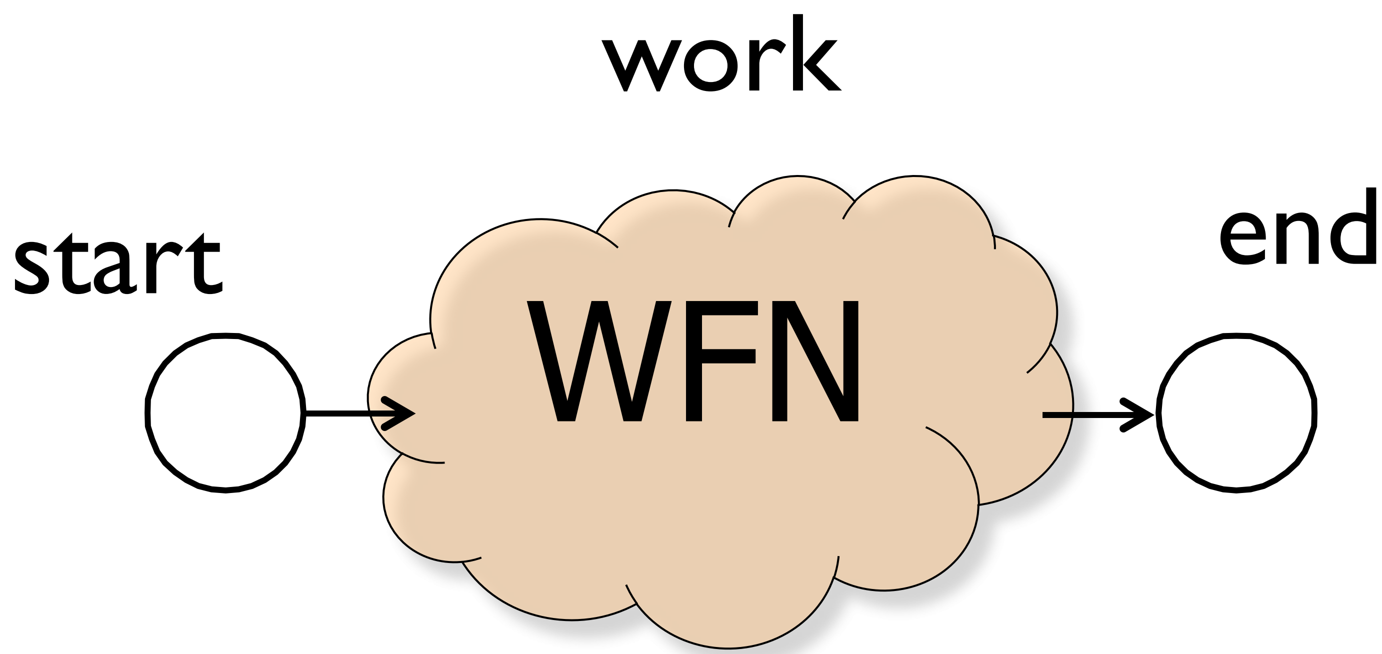


Diagram verification



Workflow net: idea

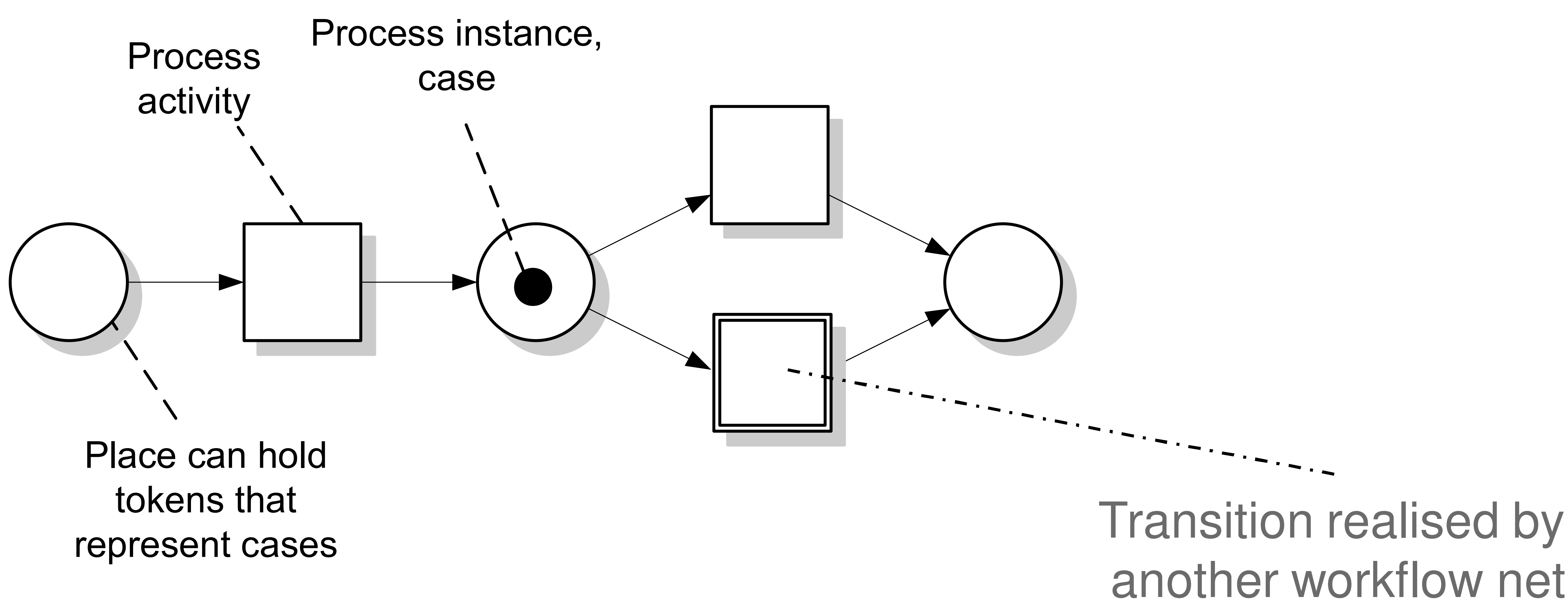


Workflow net

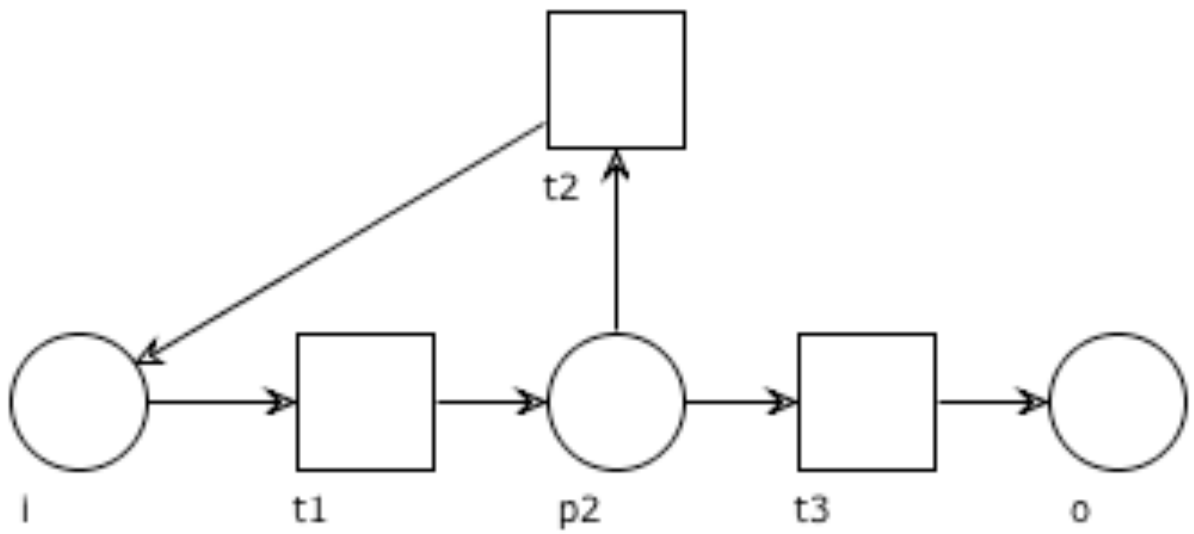
Definition:
A Petri net (P, T, F) is called **workflow net** if:

1. there is a distinguished *initial place* $i \in P$ with $\bullet i = \emptyset$
2. there is a distinguished *final place* $o \in P$ with $o\bullet = \emptyset$
3. every other place and transition belongs to a path from i to o

WF nets as business processes



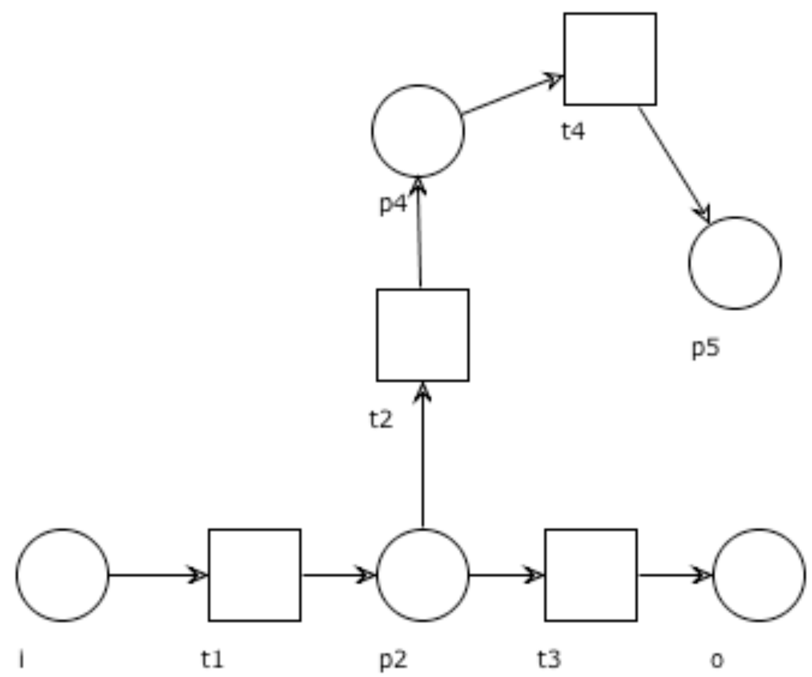
Structural analysis



No distinguished entry / exit point

no entry: when should the case start?
no exit: when should the case end?
not a workflow net!

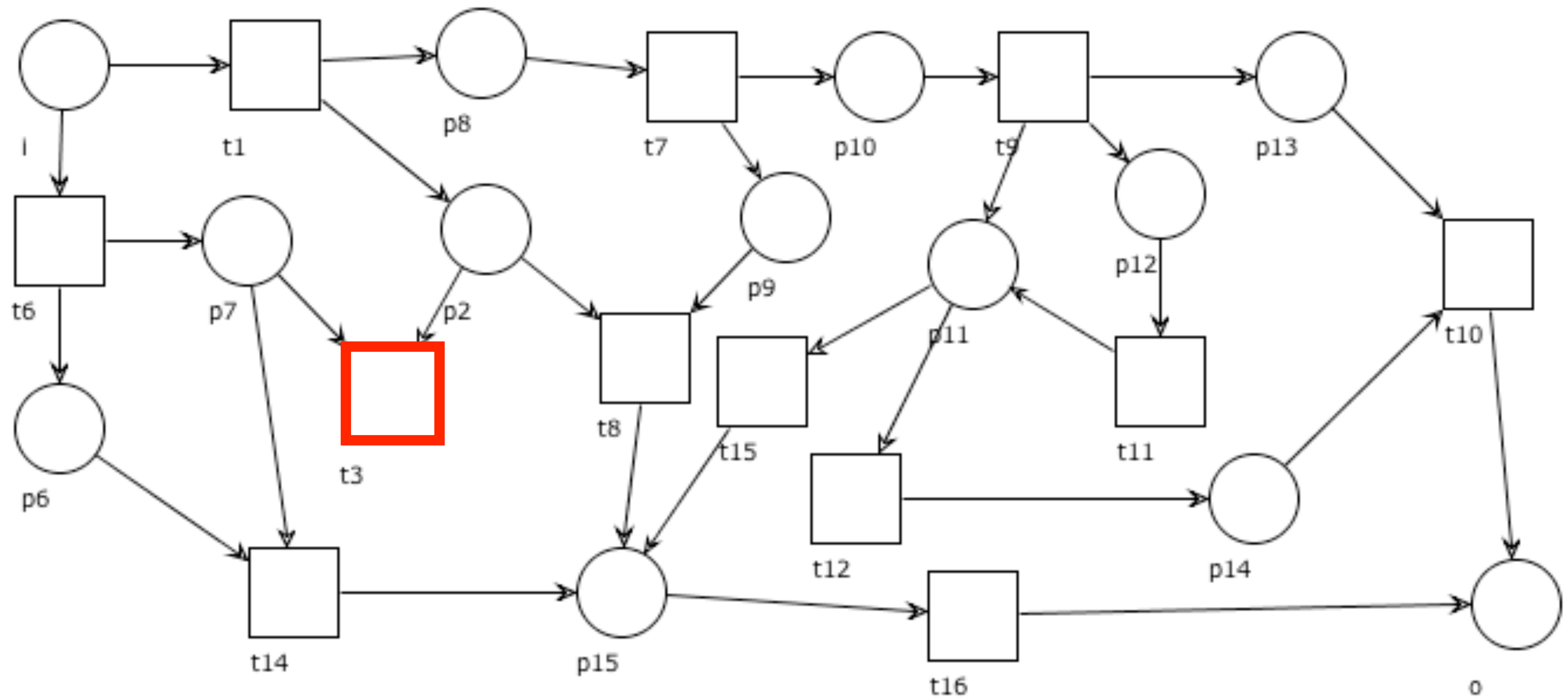
Structural analysis



Multiple entry / exit points

multiple entries: when should the case start?
multiple exit: when should the case end?
not a workflow net!

Structural analysis



Tasks t without incoming and/or outgoing arcs

no input: when should t be carried out?
no output: t does not contribute to case completion
not a workflow net!

Structural analysis

Wrong decorations of transitions

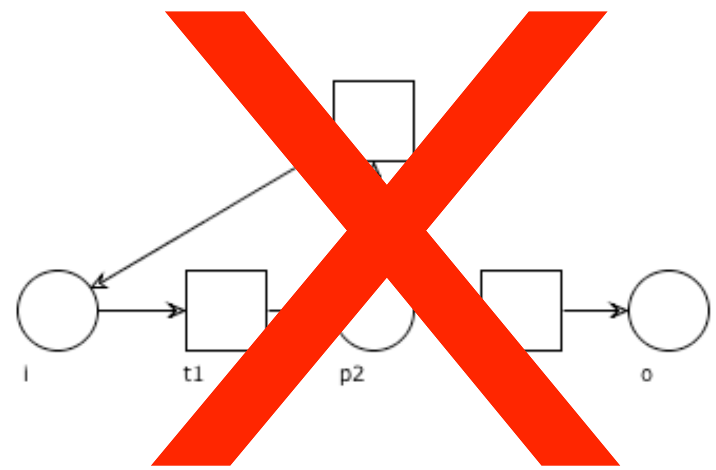
split with only one outgoing arc

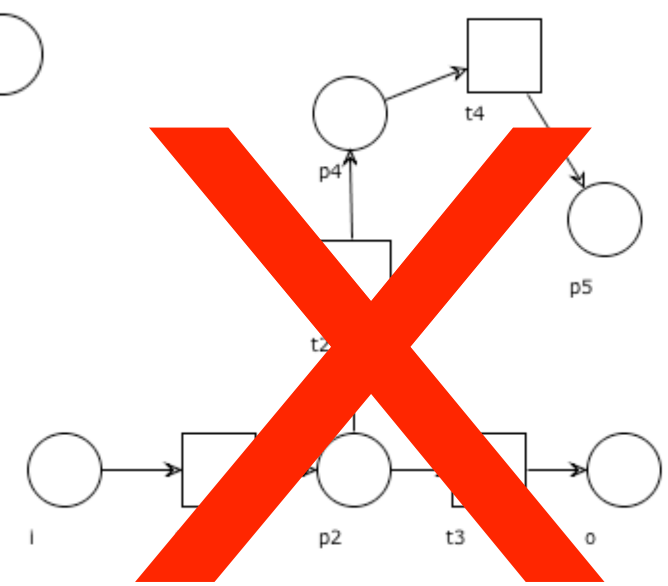
join with only one incoming arc

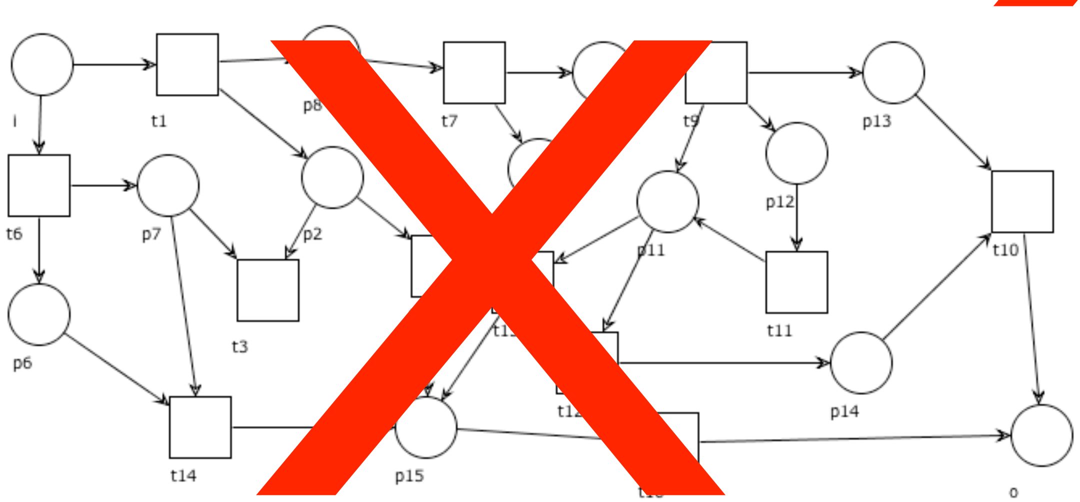
non-sense: left to designer responsibility

Structural analysis

The definition of Workflow nets is purely structural but already rules out many erroneous models







Structural properties

All the properties we have seen so far are
structural (or **static**)
(i.e., they depend on the shape of the graph,
on its connectivity or topology,
but NOT on the initial marking and enabled firings)

We also care about **behavioural** properties
(e.g., how the system can evolve,
which firing sequences will be possible,
which markings will be reachable)

A matter of terminology

To better reflect the above distinction, it is frequent:

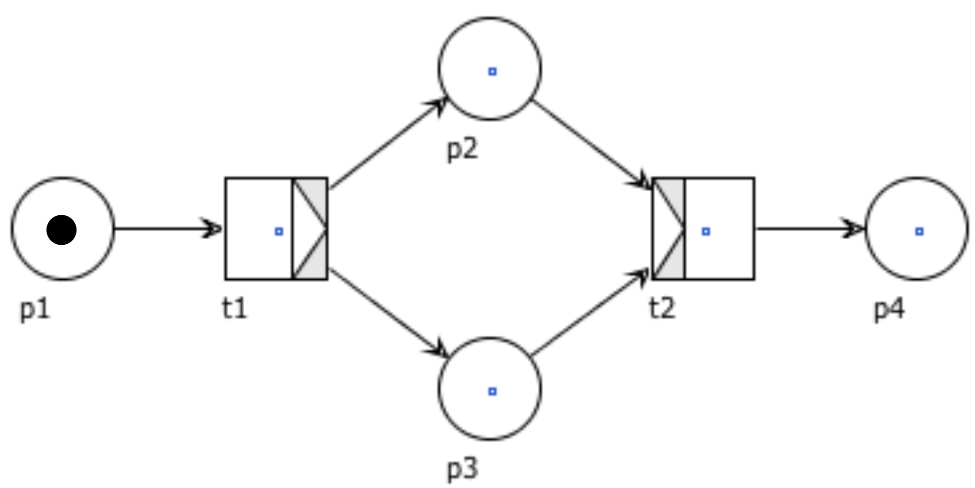
to use the term **net system** for denoting a Petri net **with** a given initial marking
(we study behavioural properties of systems)

to use the term **net** for denoting a Petri net **without** specifying any initial marking
(we study structural properties of nets)

even if, in the case of workflow nets, the initial markings will consist of one token in the initial place

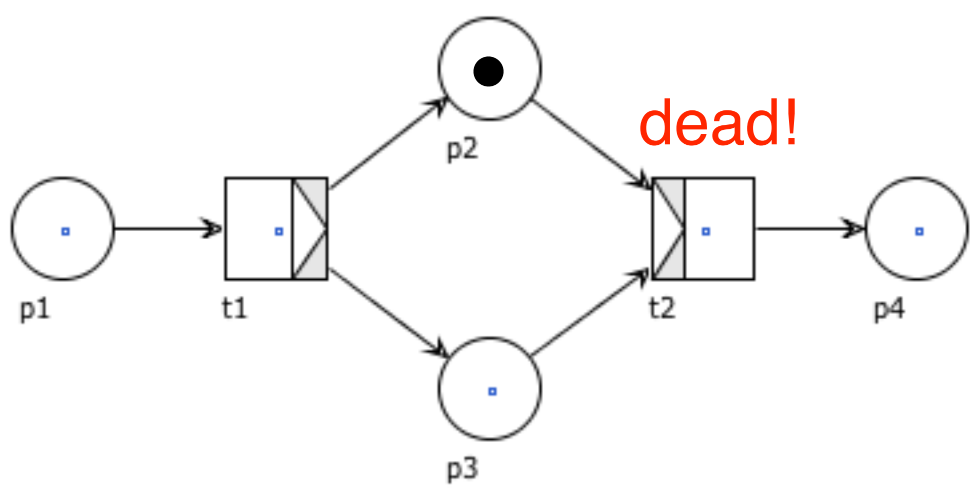
Behavioural analysis

Structural correctness cannot rule out many other problematic issues...



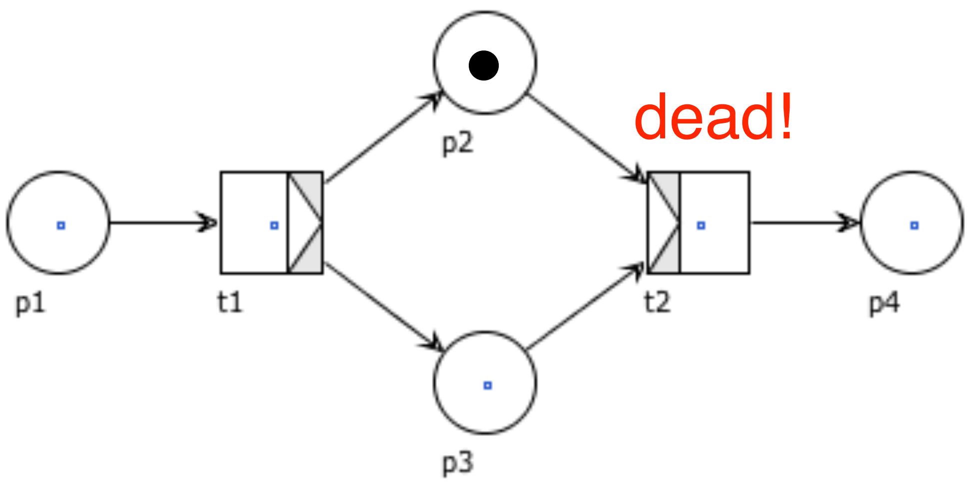
Behavioural analysis

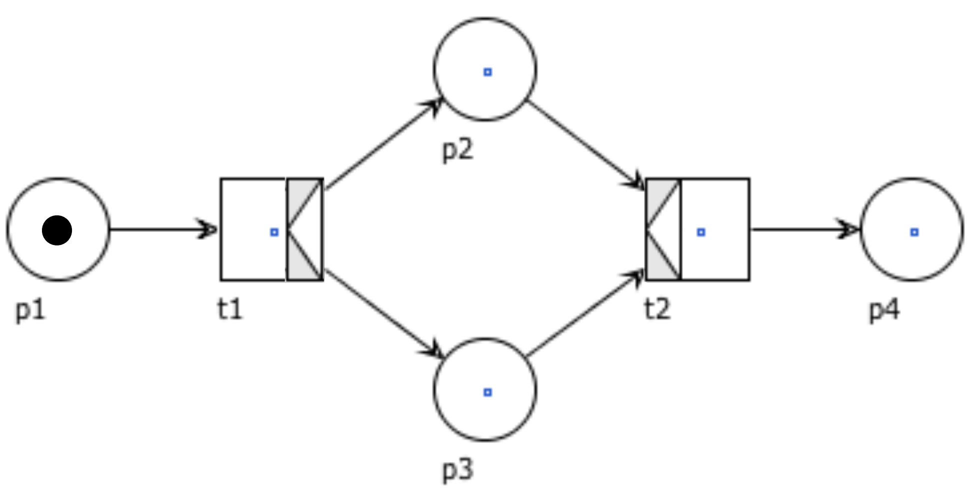
Structural correctness cannot rule out many other problematic issues...



Behavioural analysis

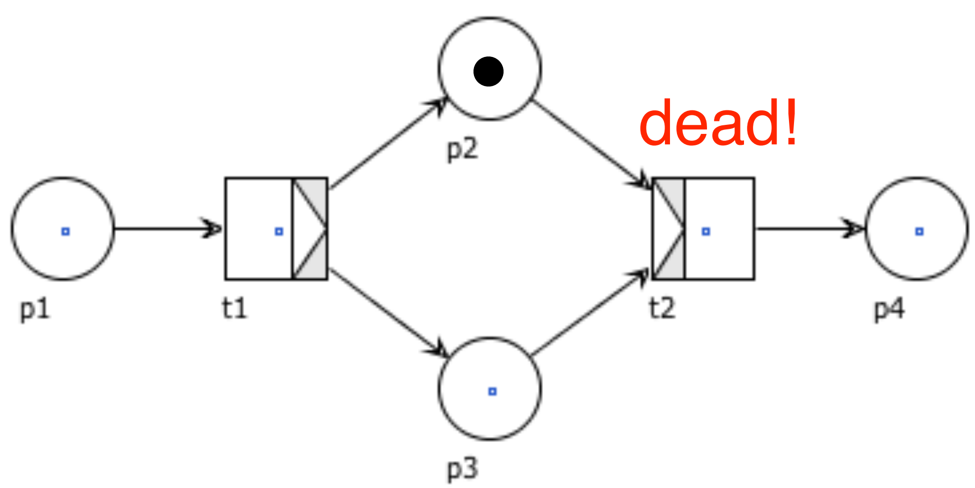
Structural correctness cannot rule out many other problematic issues...

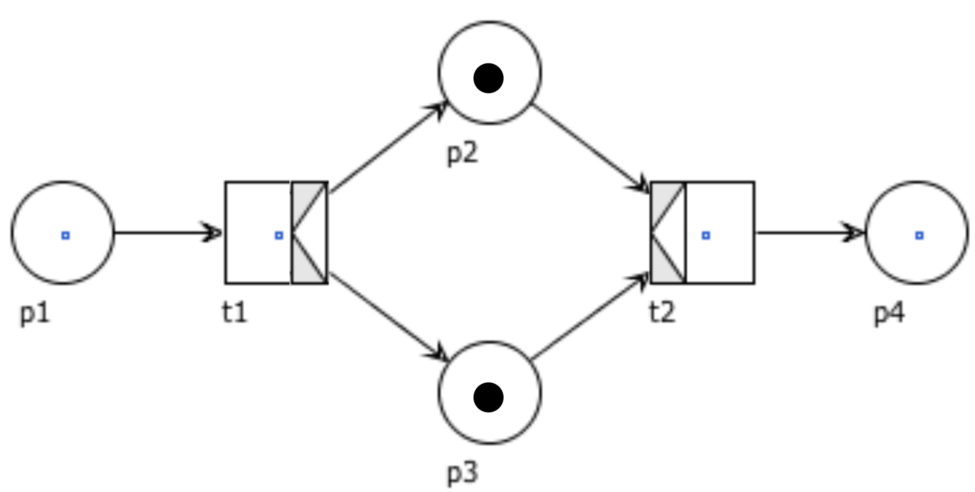




Behavioural analysis

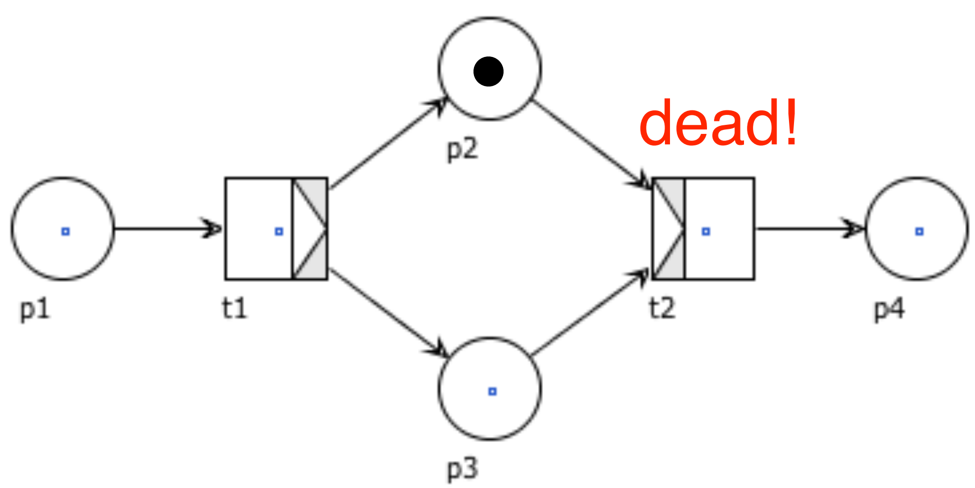
Structural correctness cannot rule out many other problematic issues...

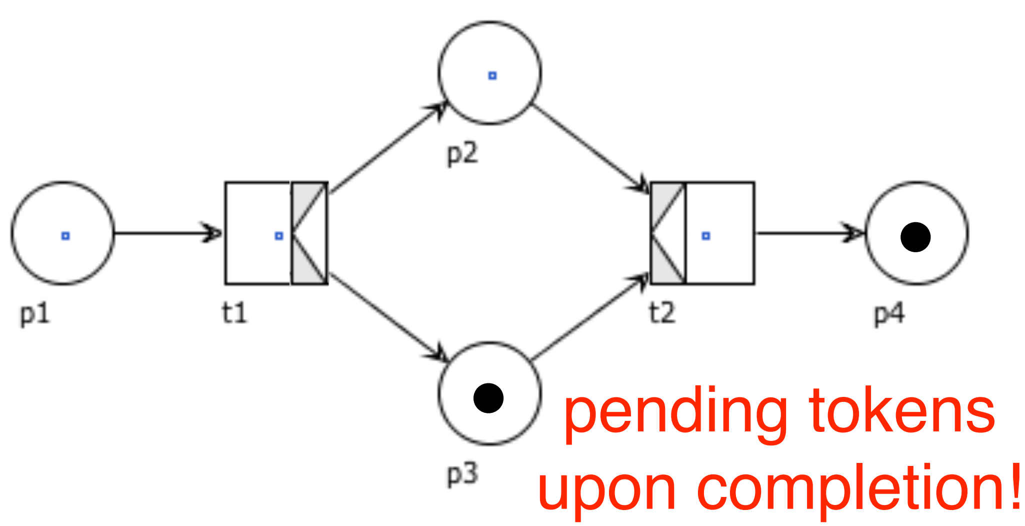




Behavioural analysis

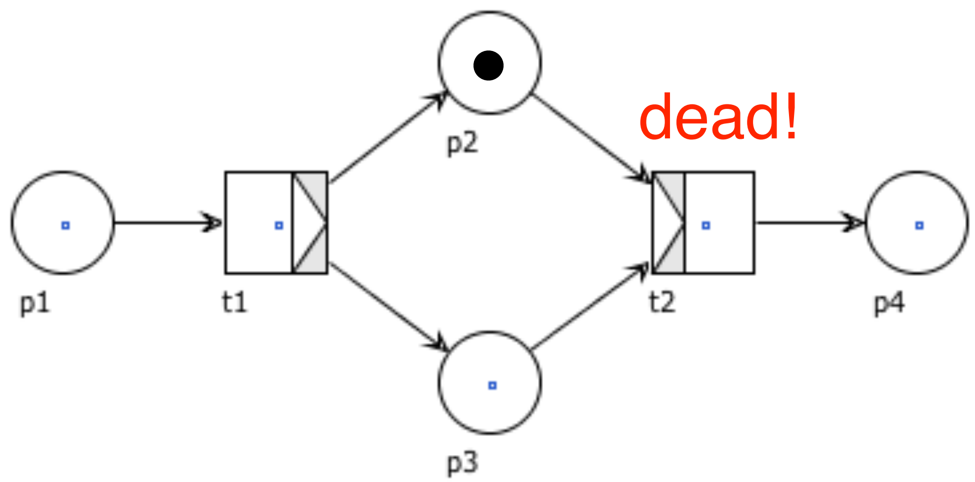
Structural correctness cannot rule out many other problematic issues...

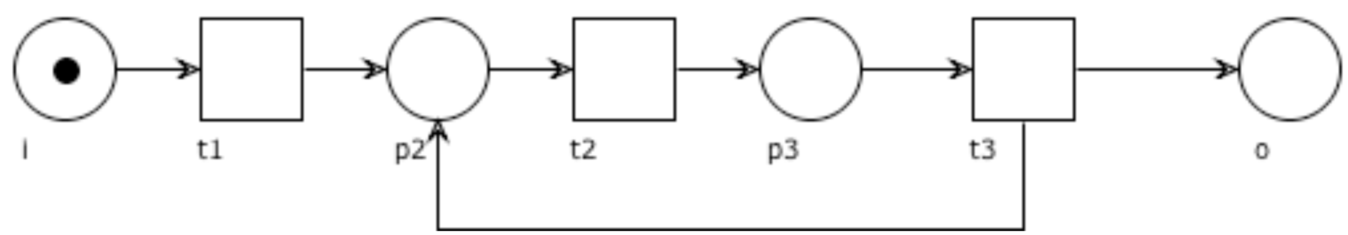


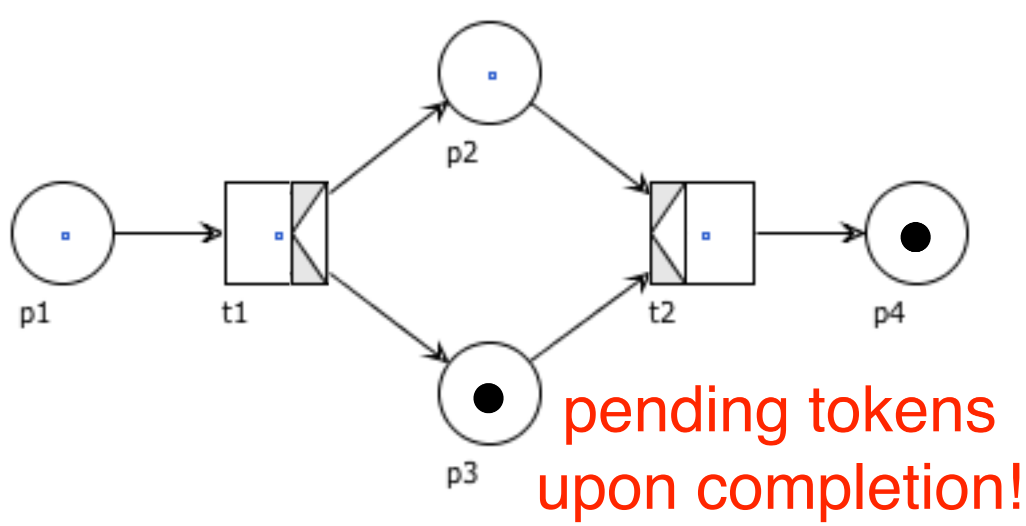


Behavioural analysis

Structural correctness cannot rule out many other problematic issues...

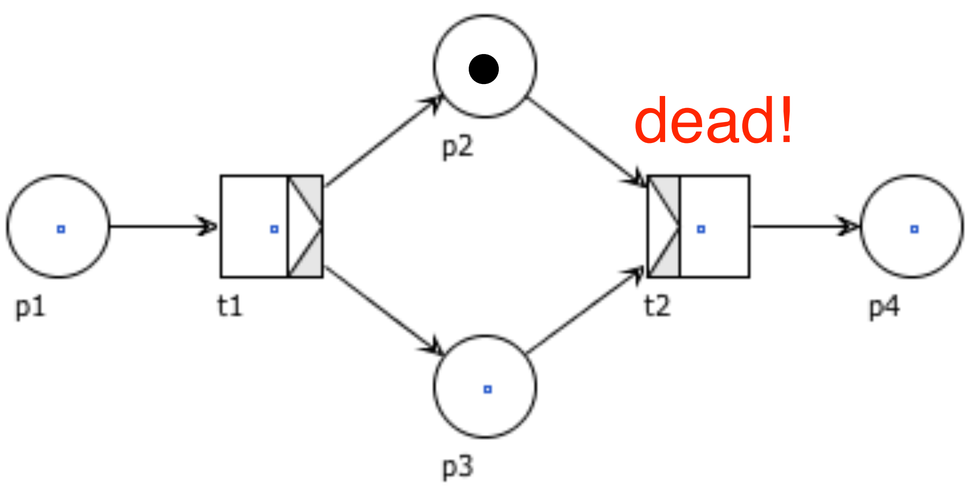


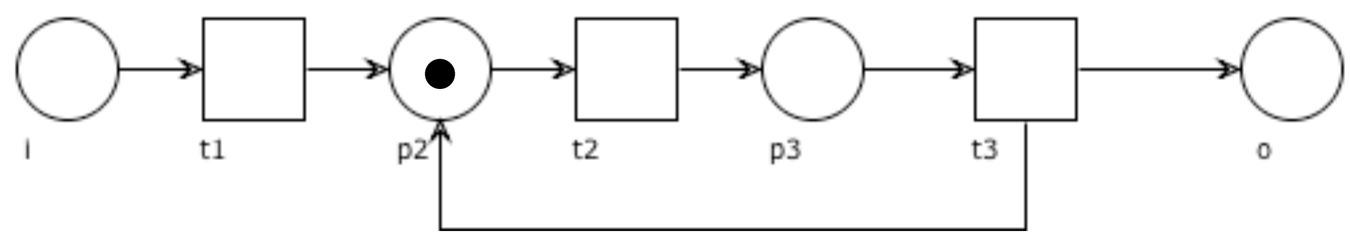


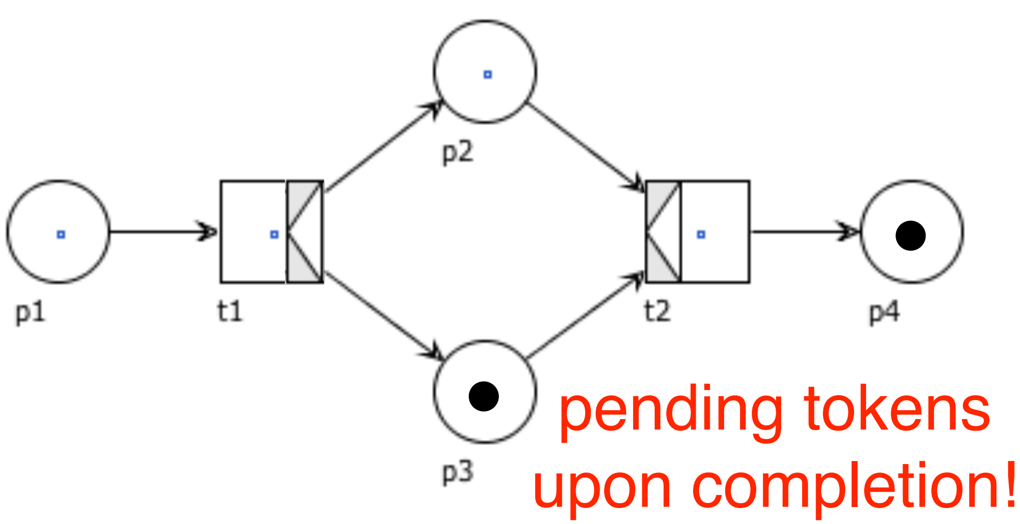


Behavioural analysis

Structural correctness cannot rule out many other problematic issues...

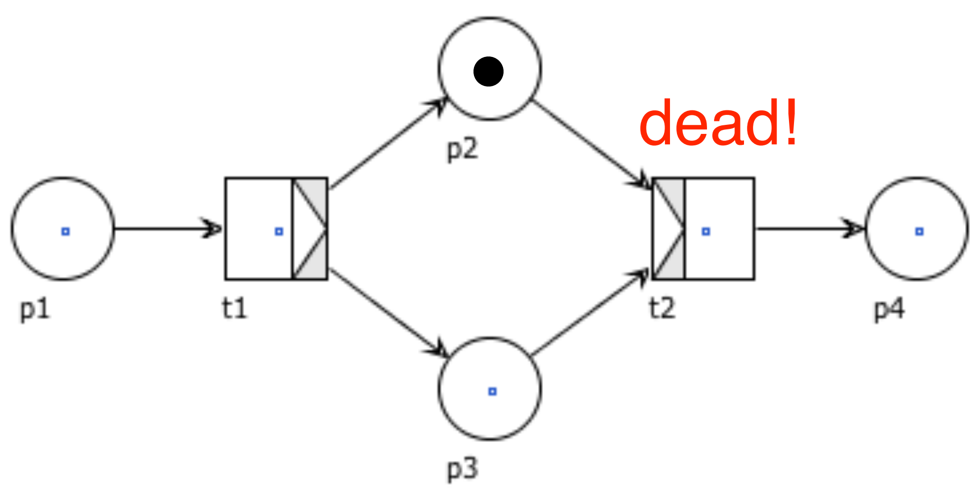


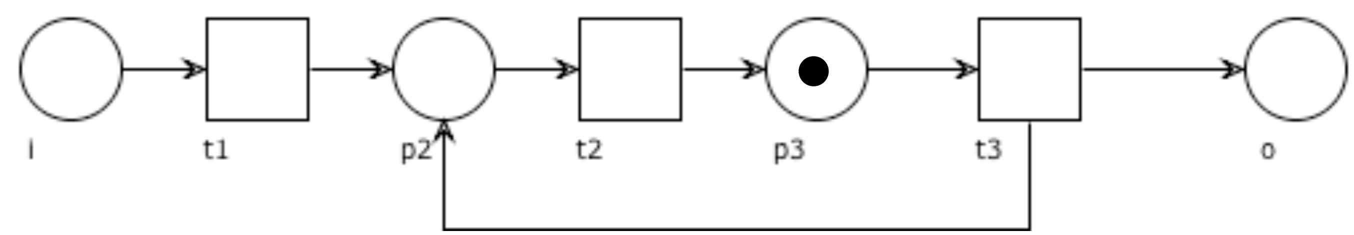


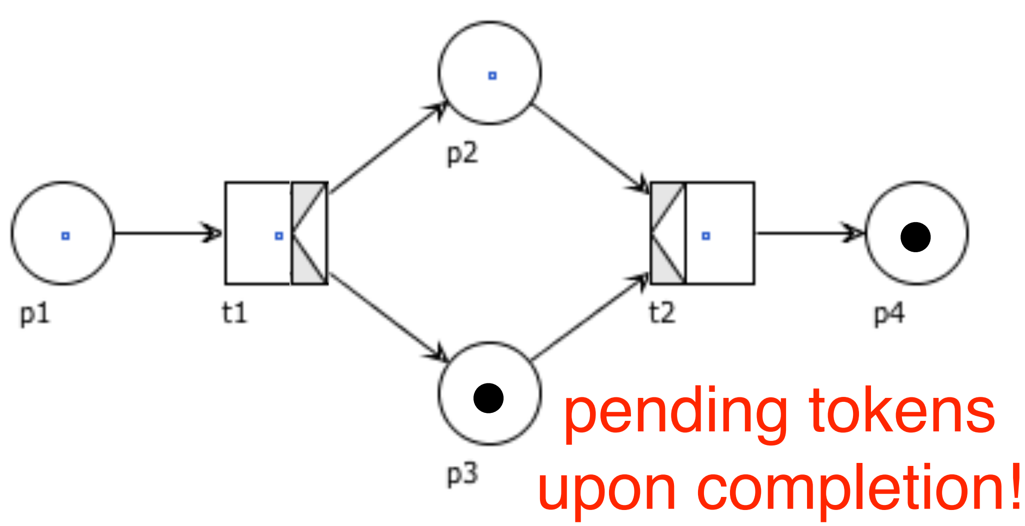


Behavioural analysis

Structural correctness cannot rule out many other problematic issues...

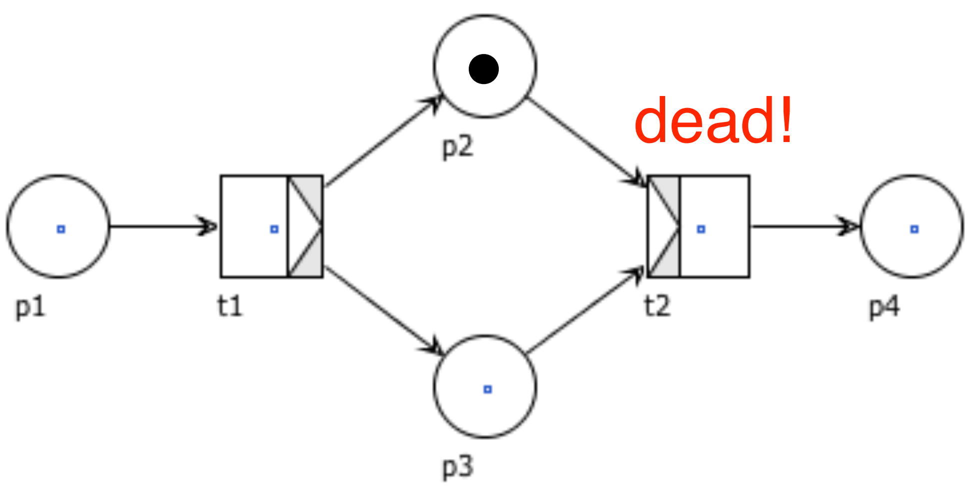






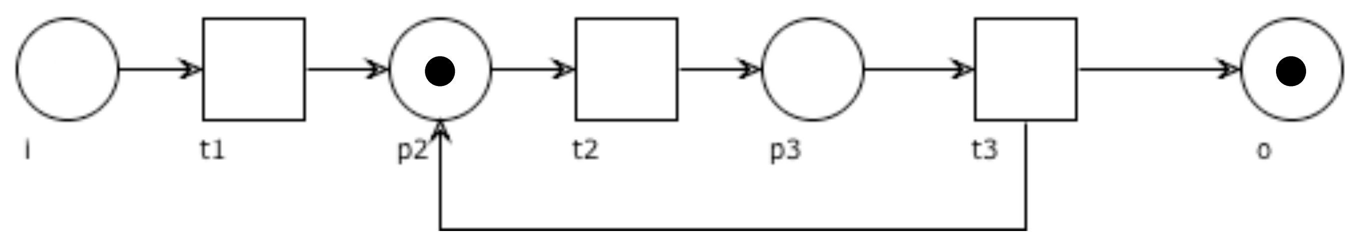
Behavioural analysis

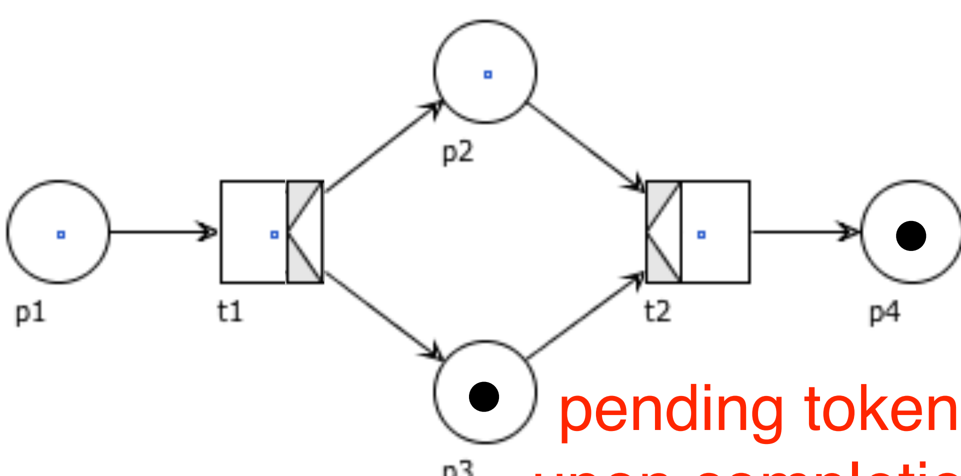
Structural correctness cannot rule out many other problematic issues...



dead!

pending tokens and activities
upon completion!



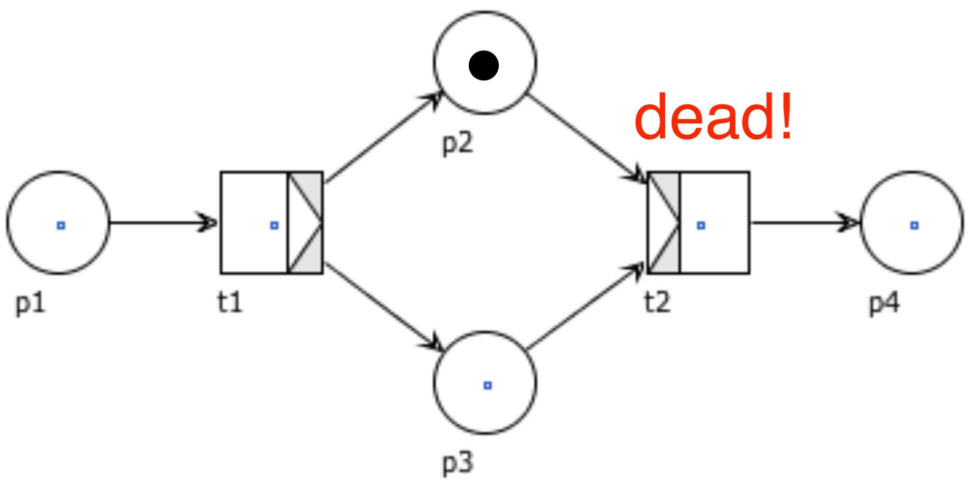


pending tokens
upon completion!

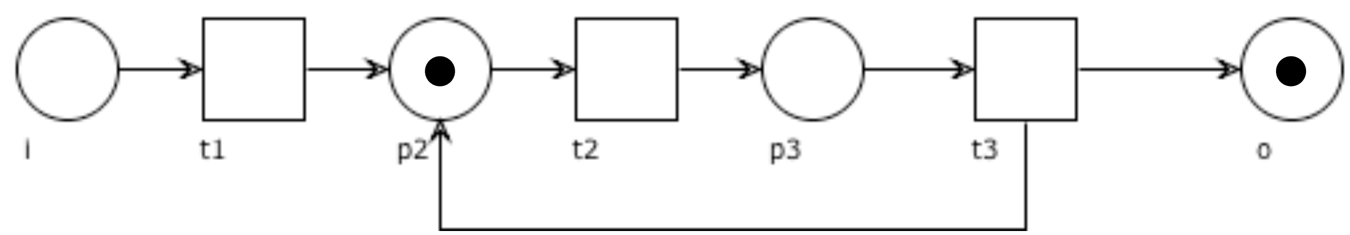
Behavioural analysis

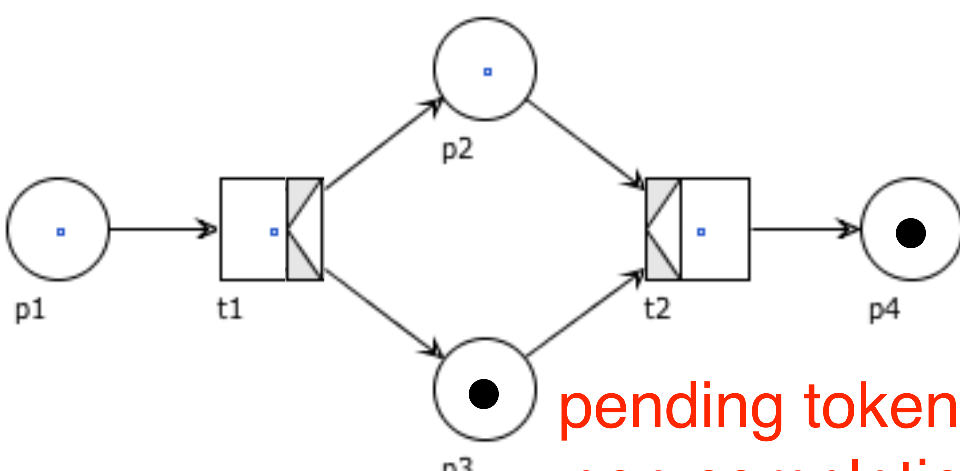
Structural correctness cannot rule out many other problematic issues...

dead!

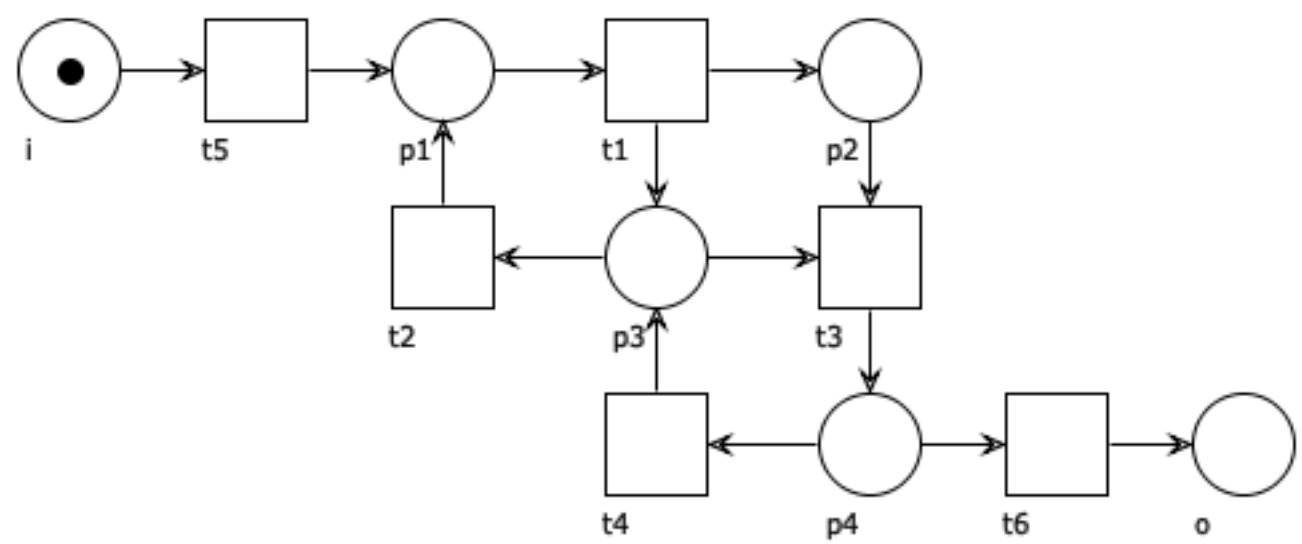


pending tokens and activities upon completion!



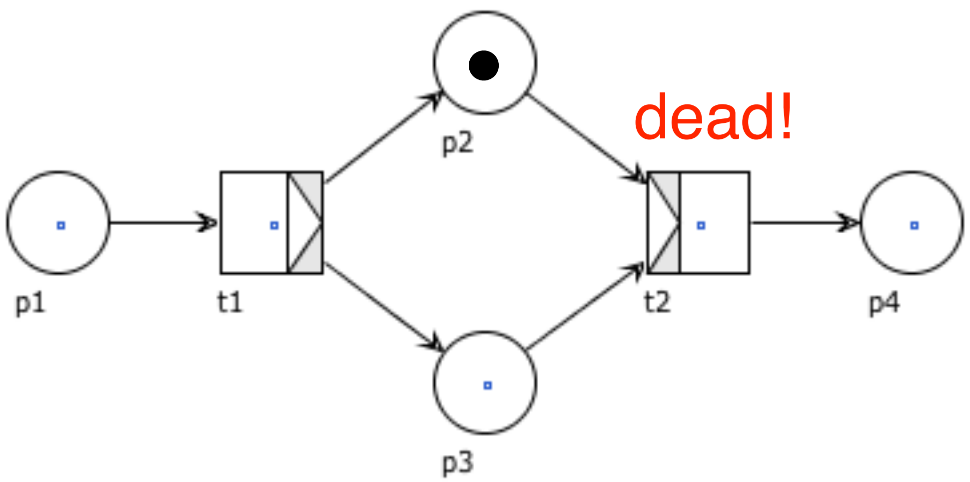


pending tokens upon completion!



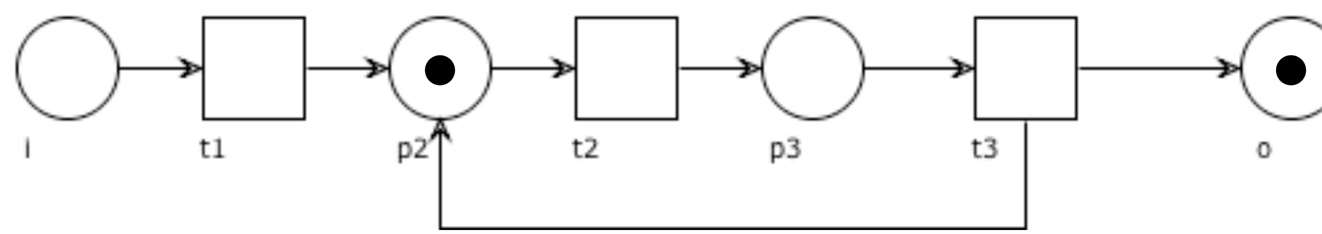
Behavioural analysis

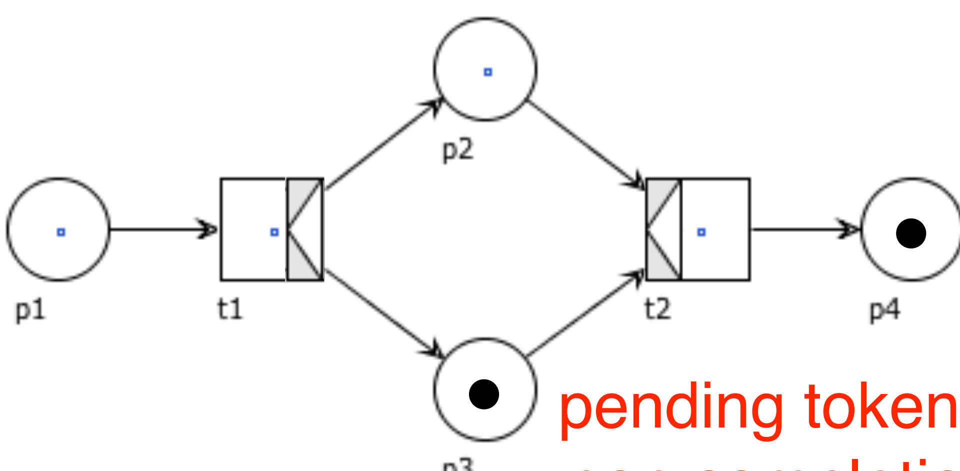
Structural correctness cannot rule out many other problematic issues...



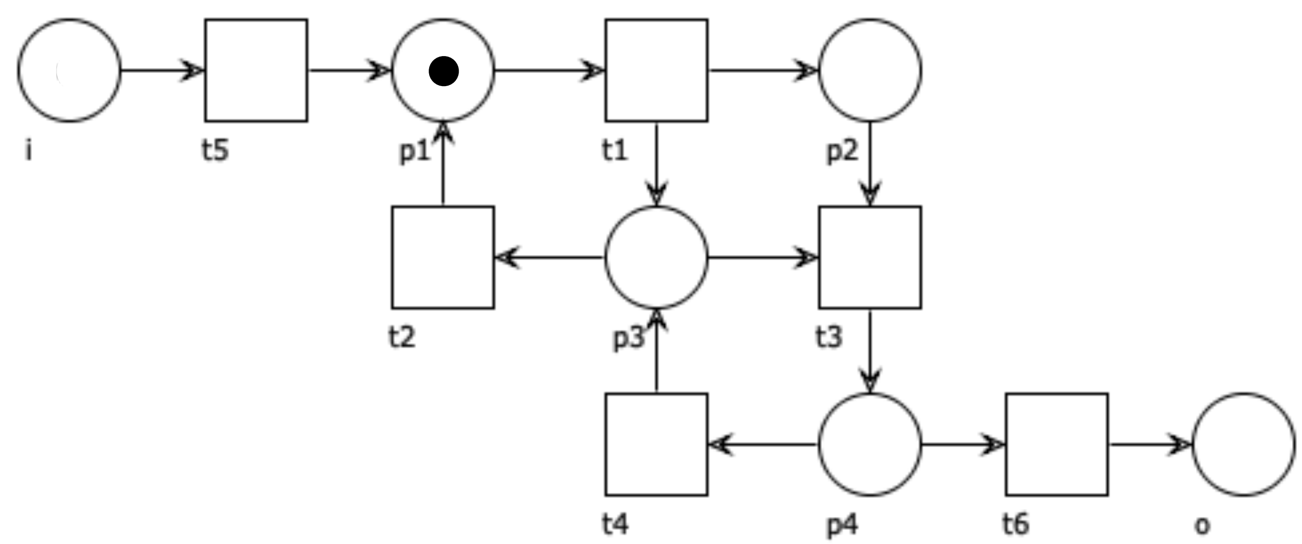
dead!

pending tokens and activities upon completion!



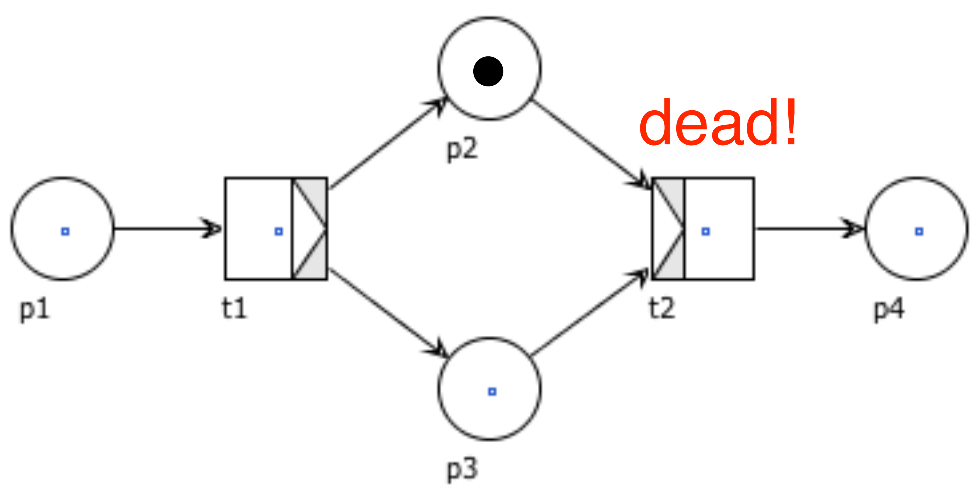


pending tokens upon completion!



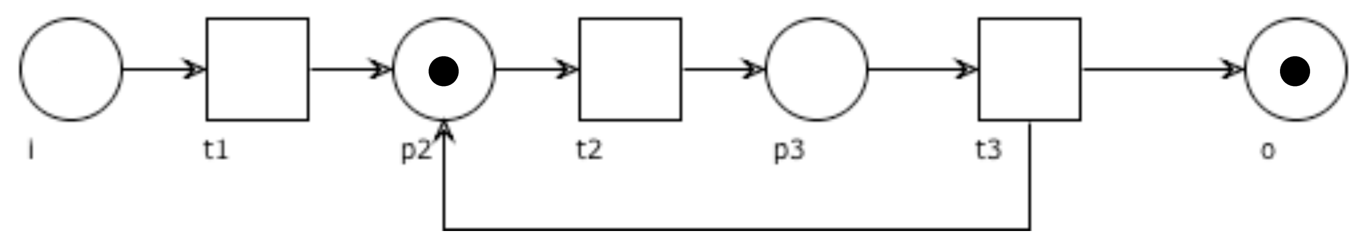
Behavioural analysis

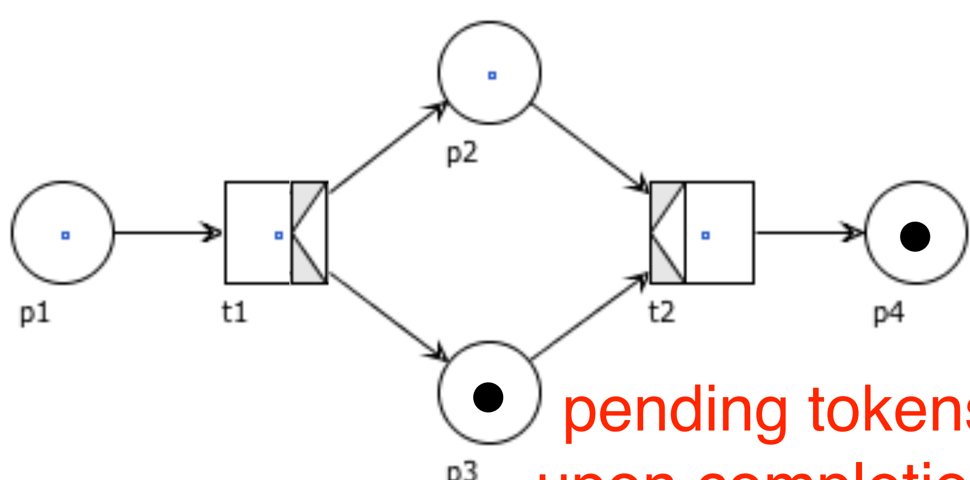
Structural correctness cannot rule out many other problematic issues...



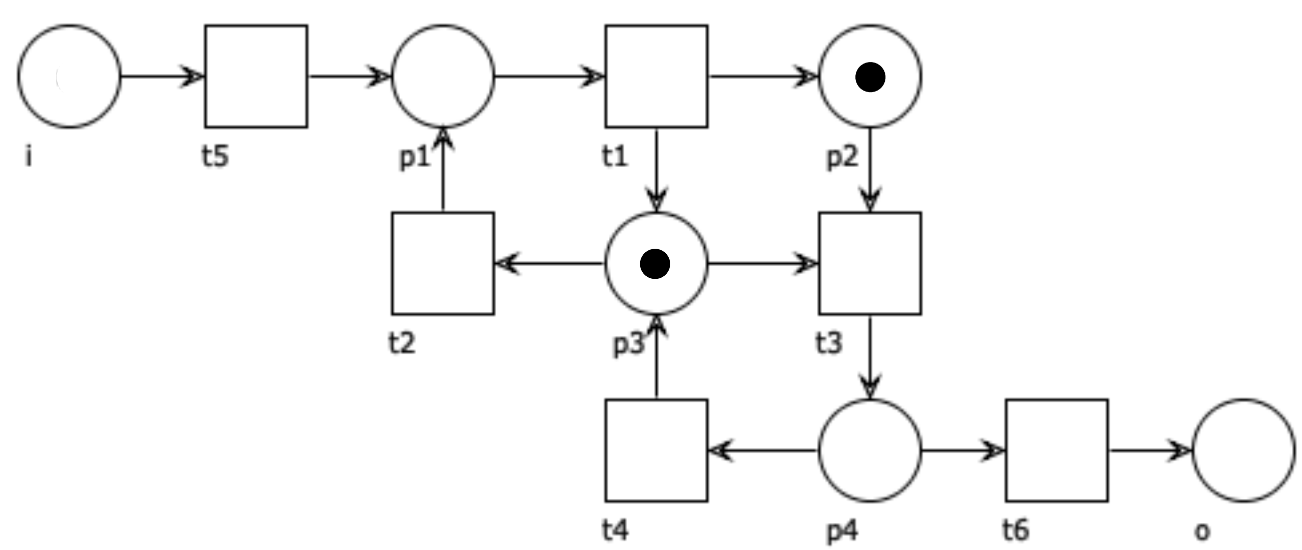
dead!

pending tokens and activities upon completion!



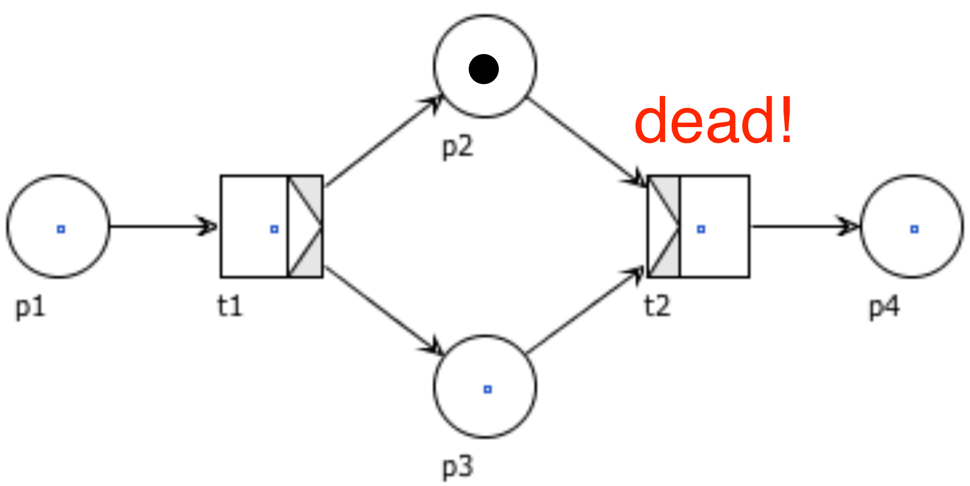


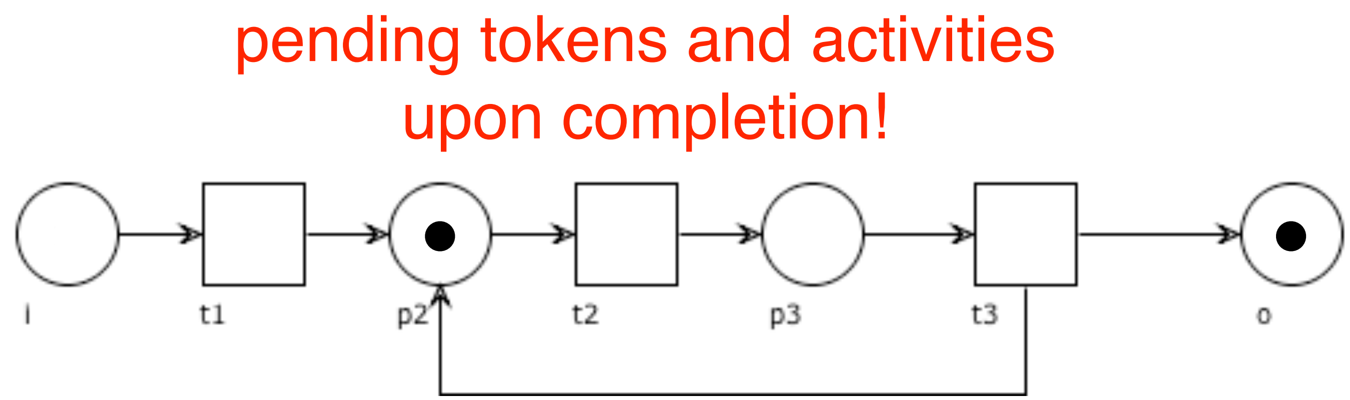
pending tokens upon completion!

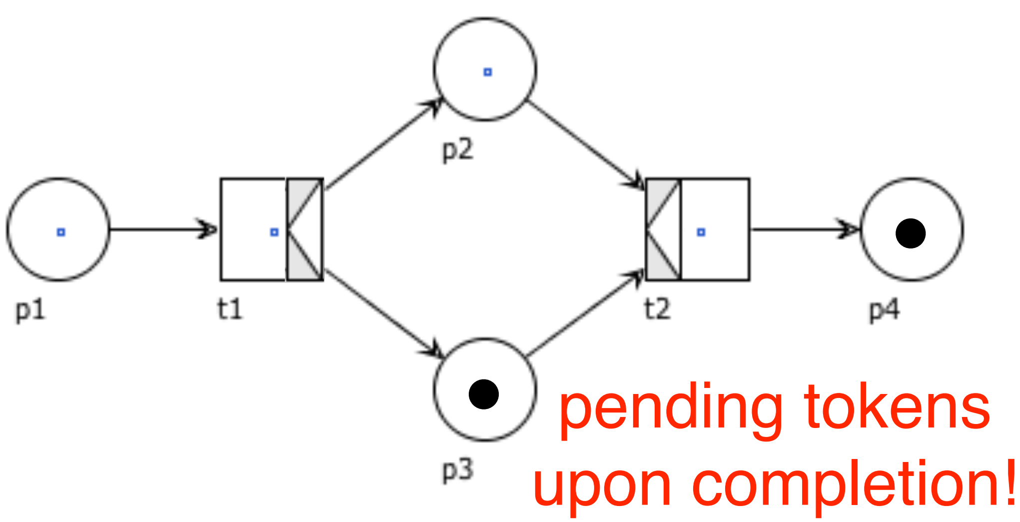


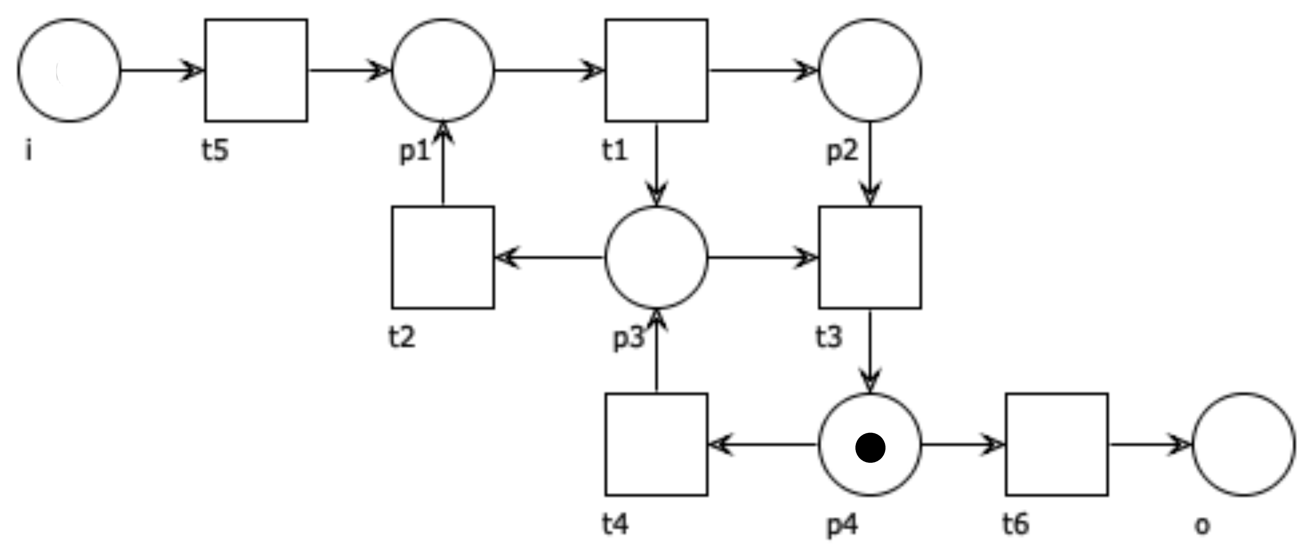
Behavioural analysis

Structural correctness cannot rule out many other problematic issues...



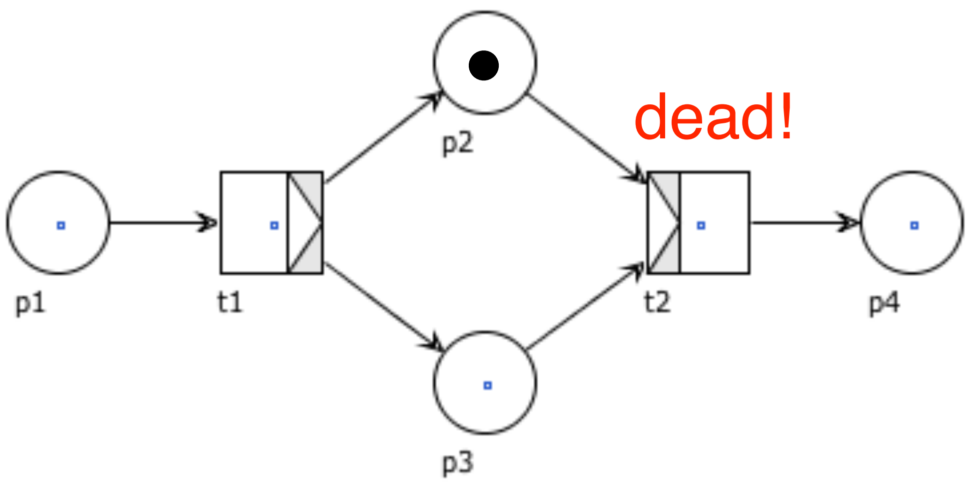






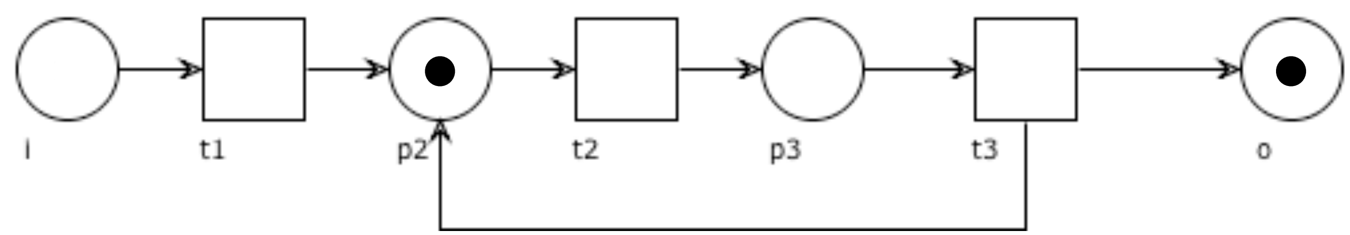
Behavioural analysis

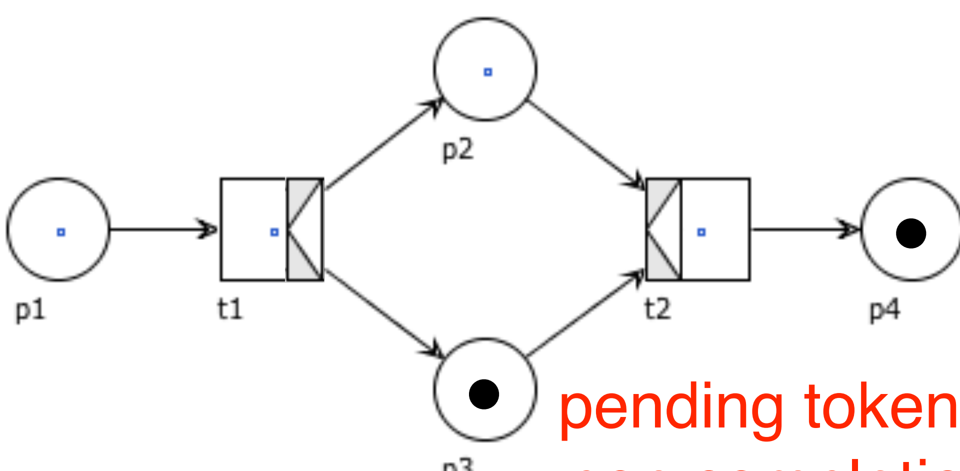
Structural correctness cannot rule out many other problematic issues...



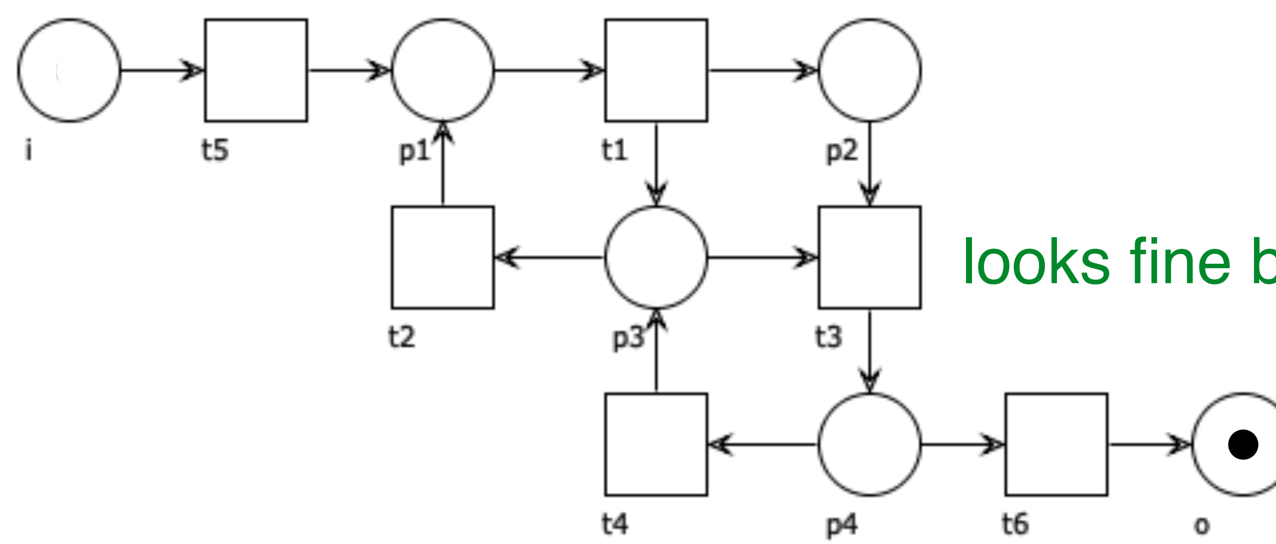
dead!

pending tokens and activities upon completion!





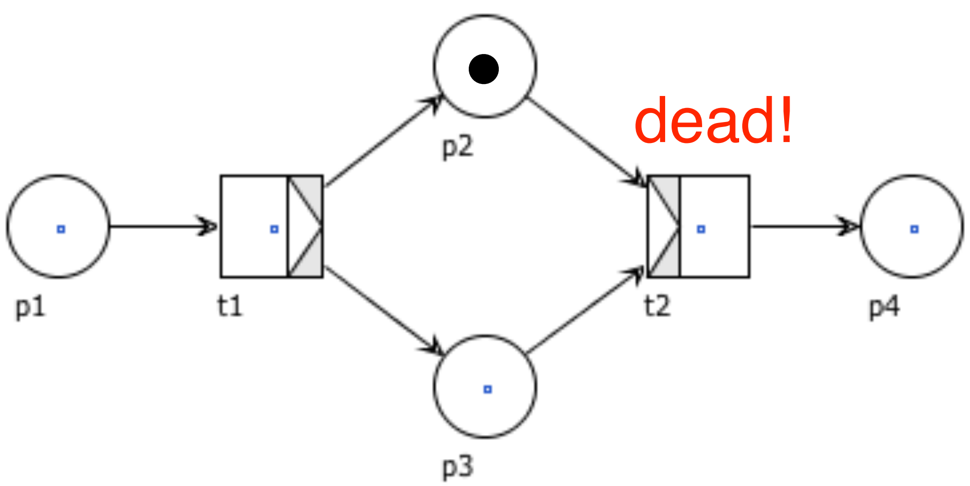
pending tokens upon completion!



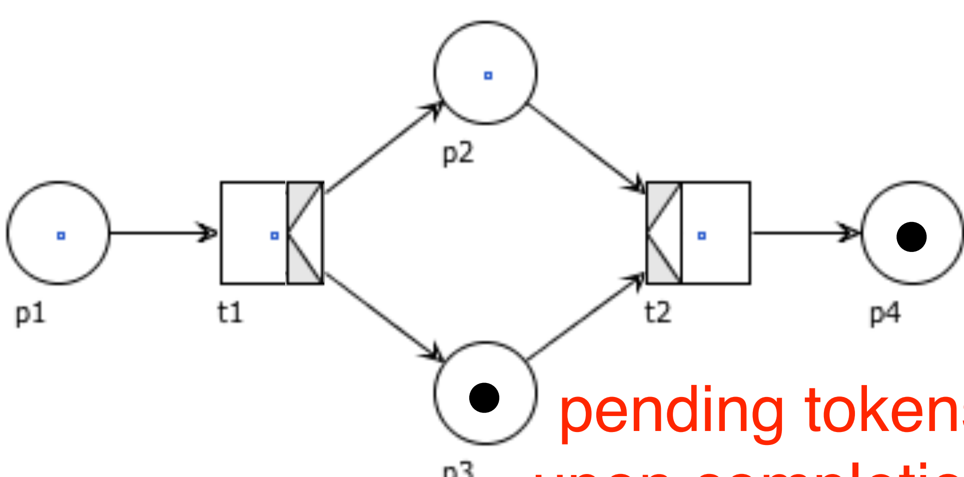
looks fine but...

Behavioural analysis

Structural correctness cannot rule out many other problematic issues...

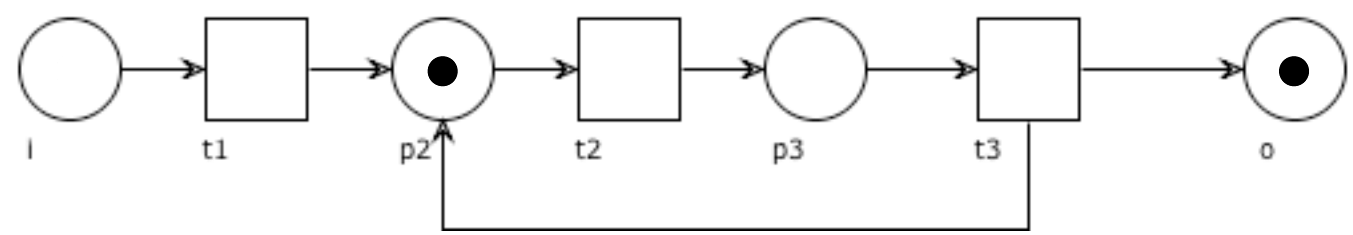


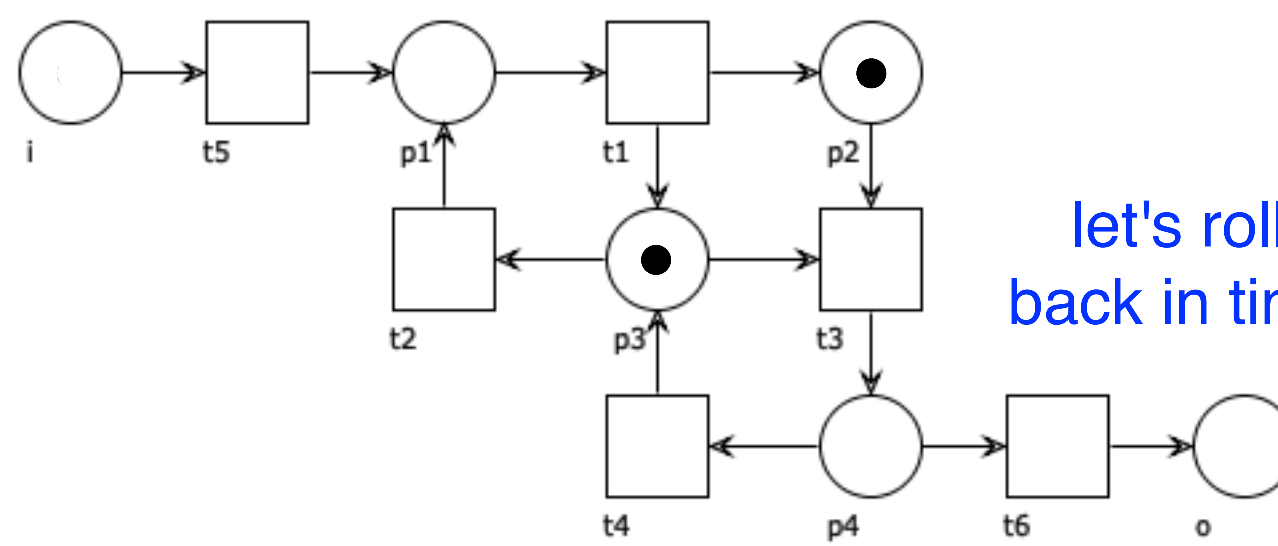
dead!



pending tokens
upon completion!

pending tokens and activities
upon completion!

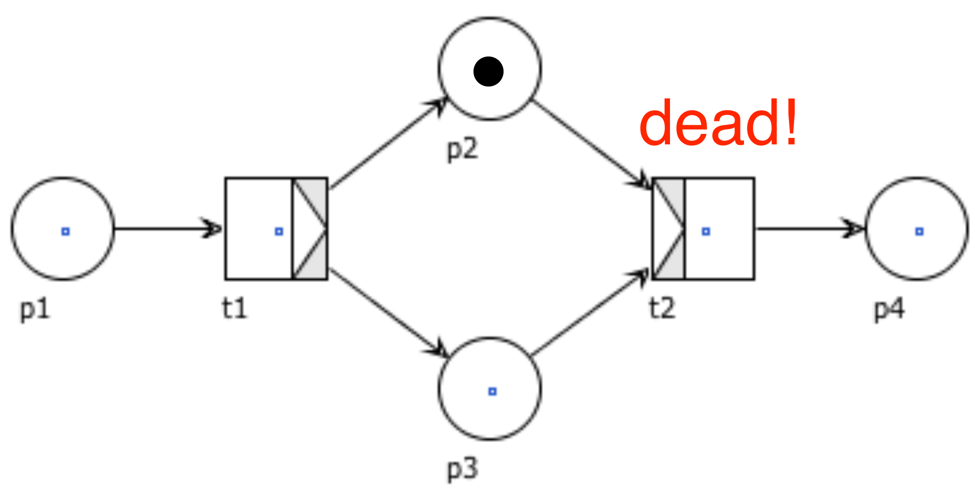


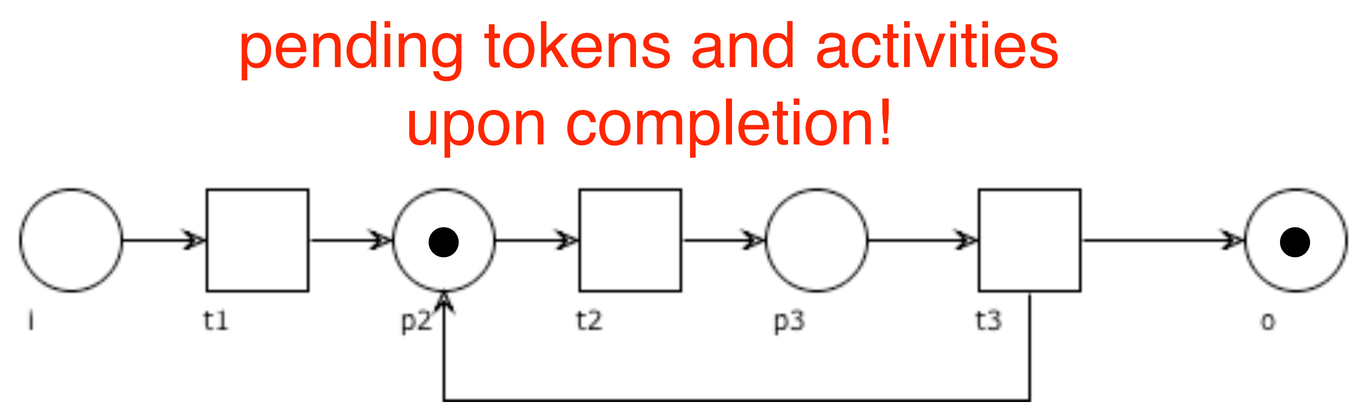


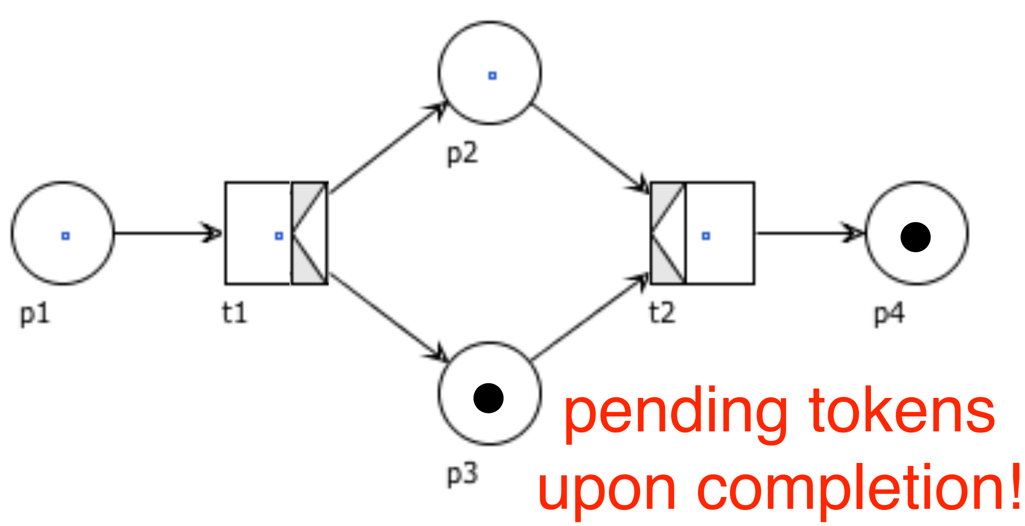
let's roll
back in time

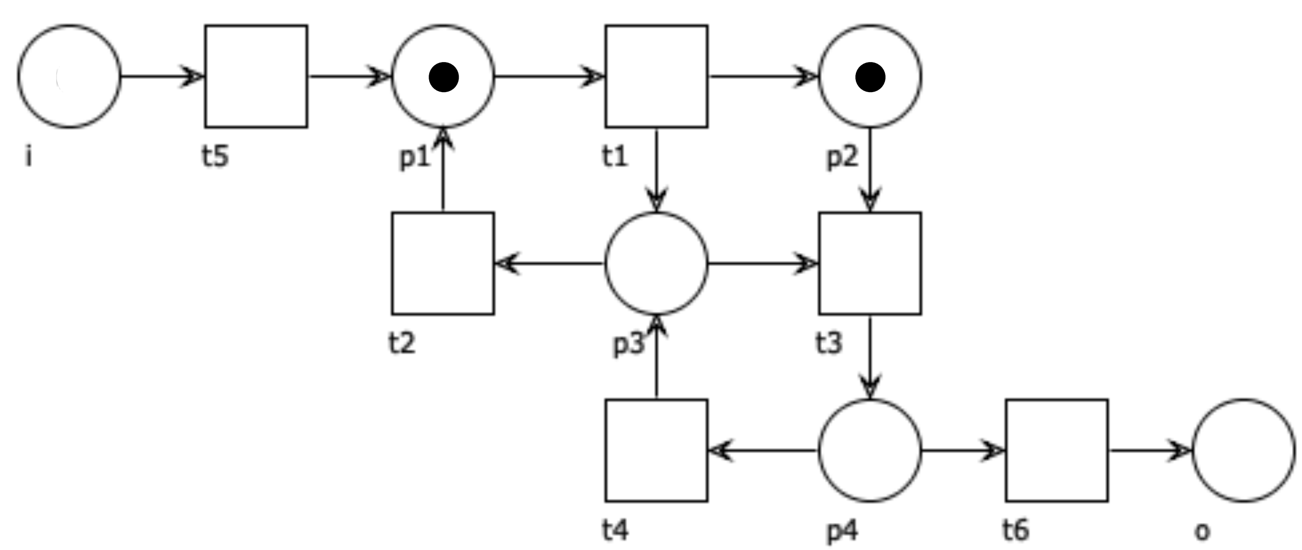
Behavioural analysis

Structural correctness cannot rule out many other problematic issues...



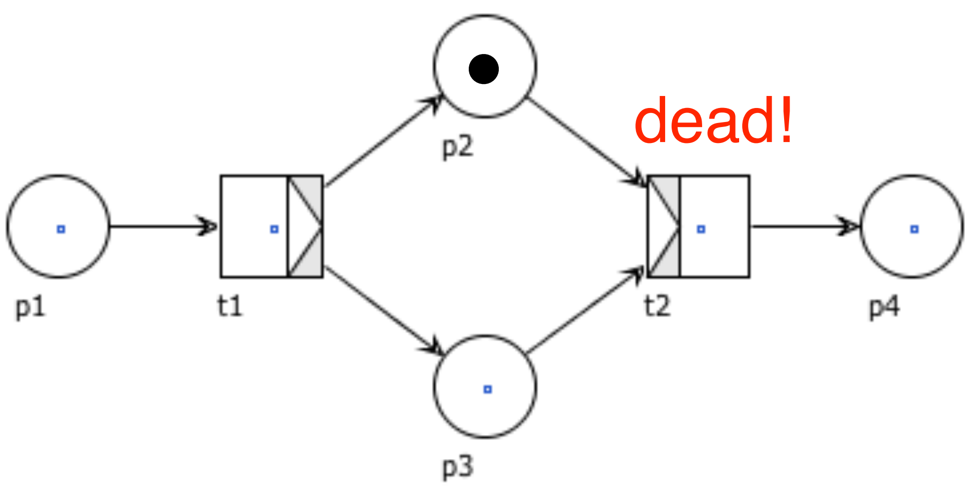






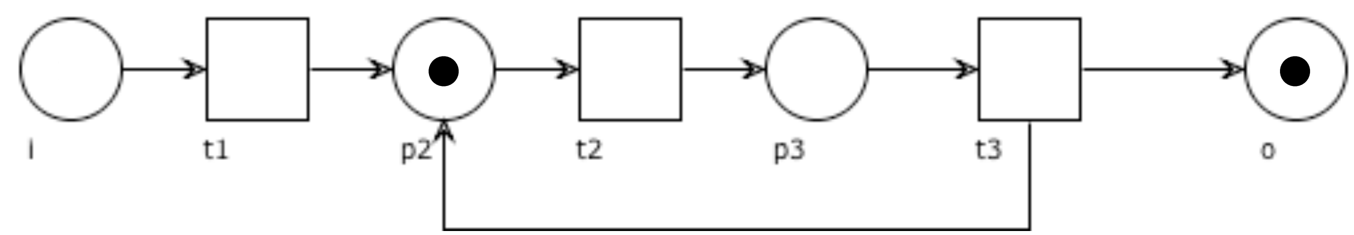
Behavioural analysis

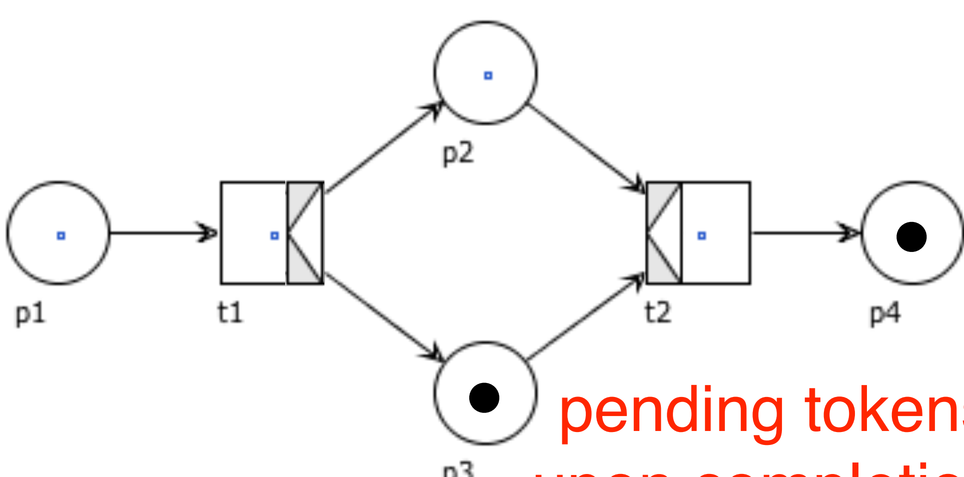
Structural correctness cannot rule out many other problematic issues...



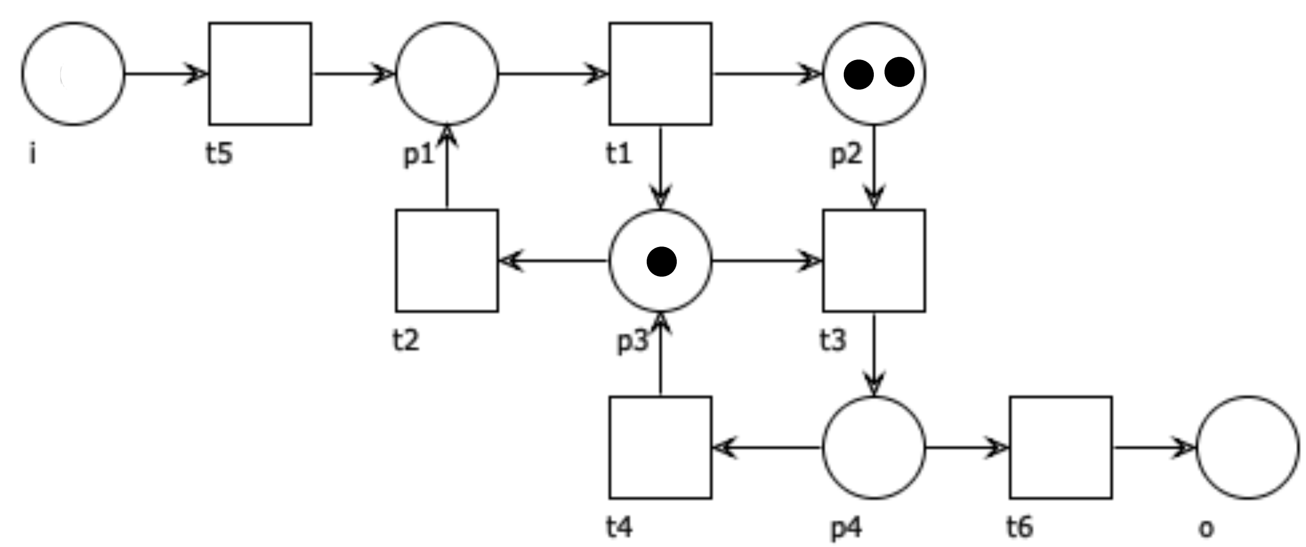
dead!

pending tokens and activities upon completion!



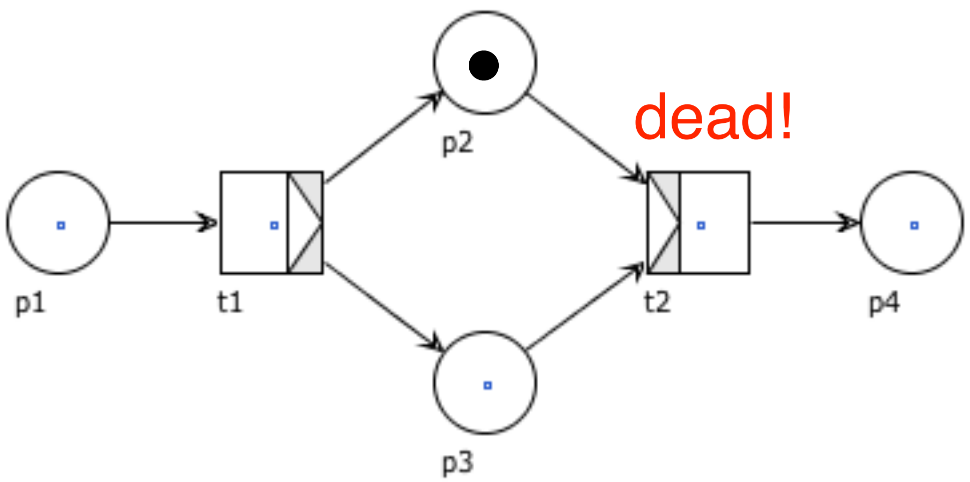


pending tokens upon completion!



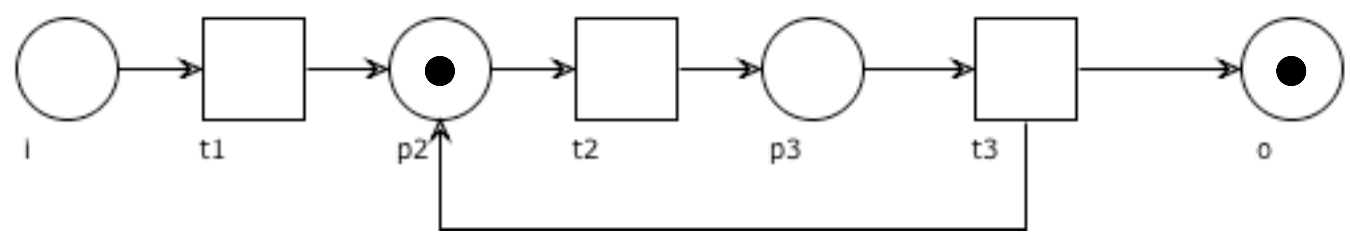
Behavioural analysis

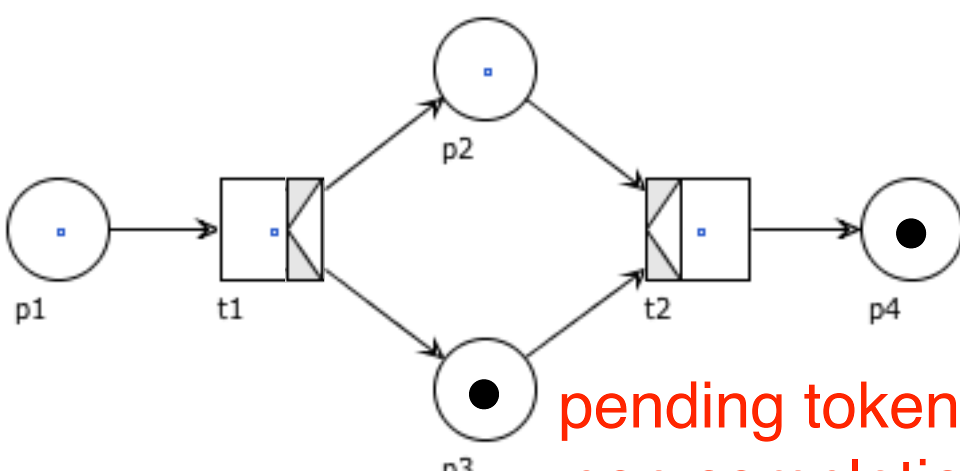
Structural correctness cannot rule out many other problematic issues...



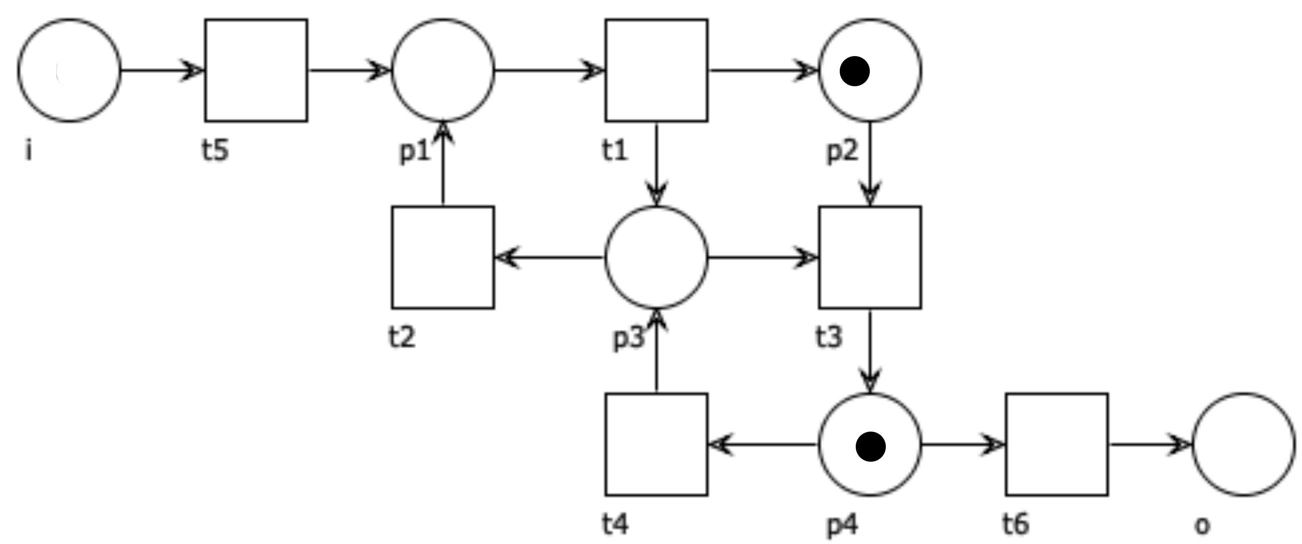
dead!

pending tokens and activities upon completion!



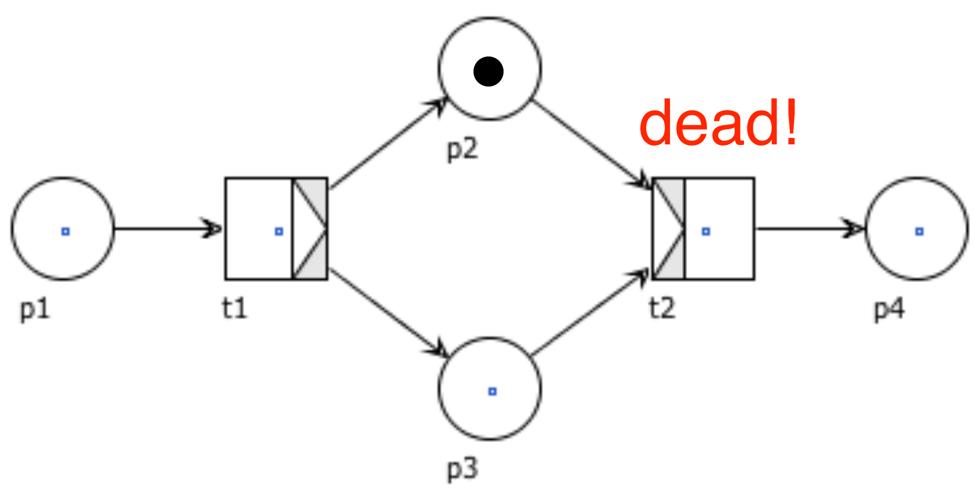


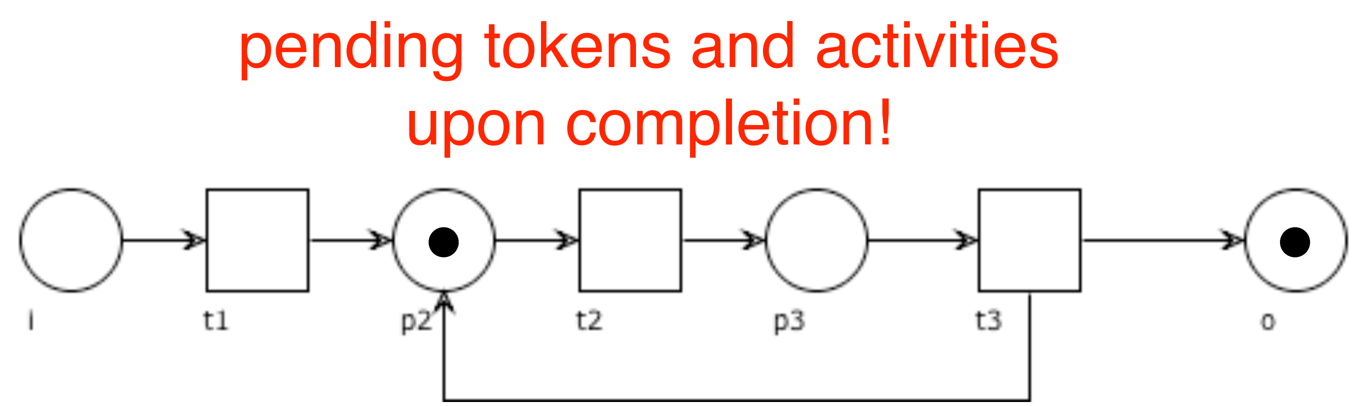
pending tokens upon completion!

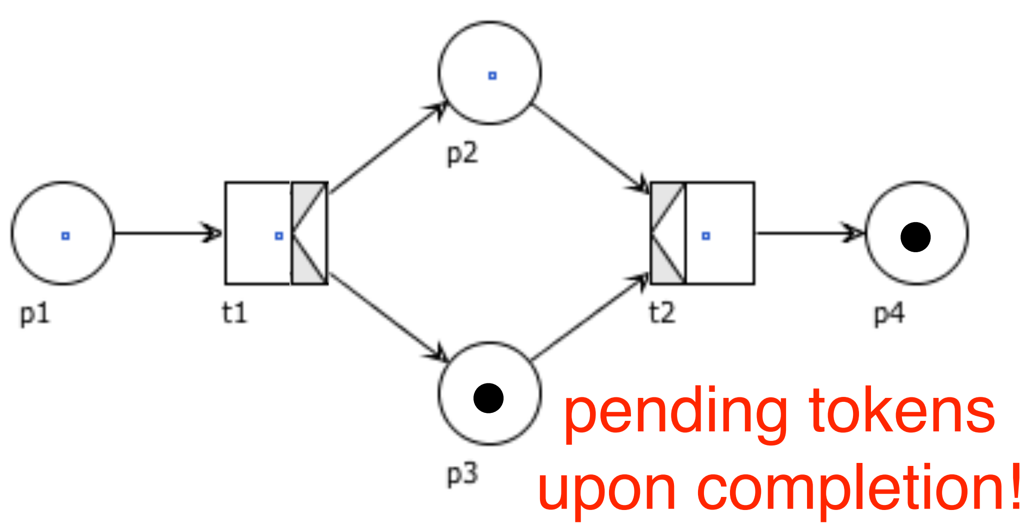


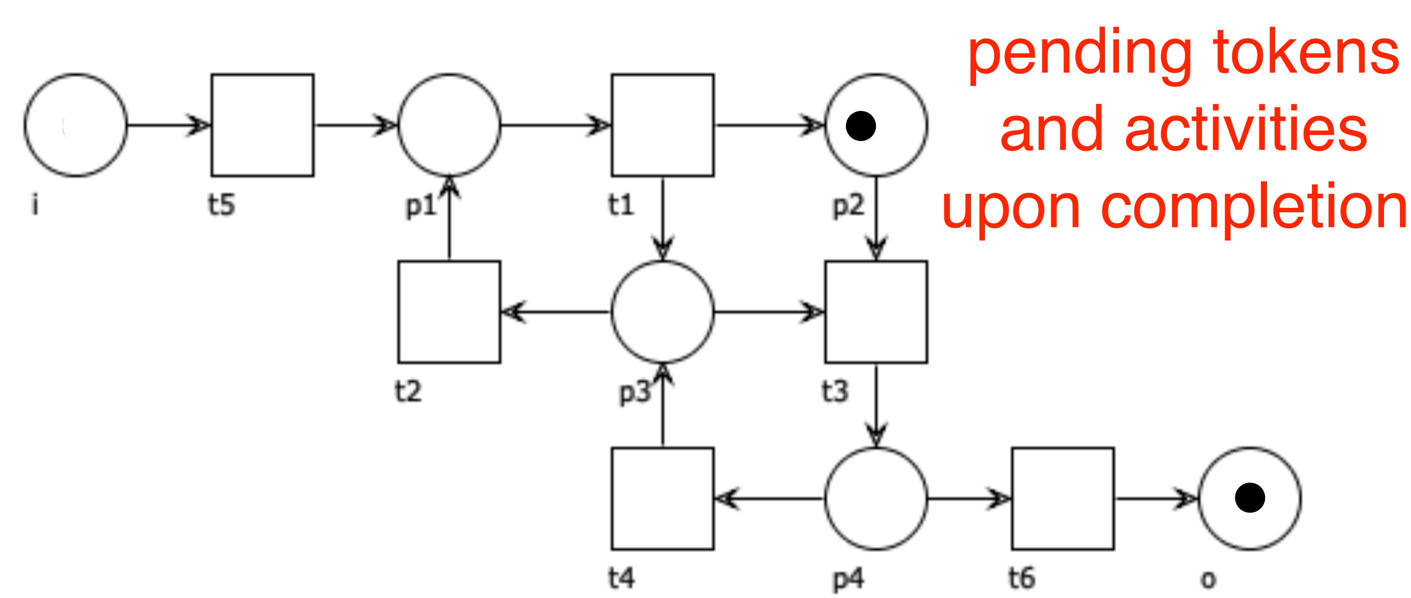
Behavioural analysis

Structural correctness cannot rule out many other problematic issues...



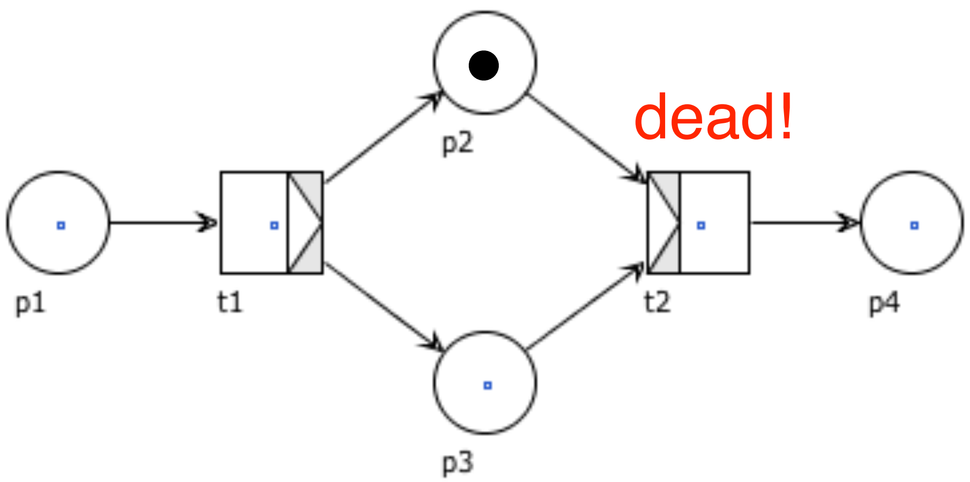






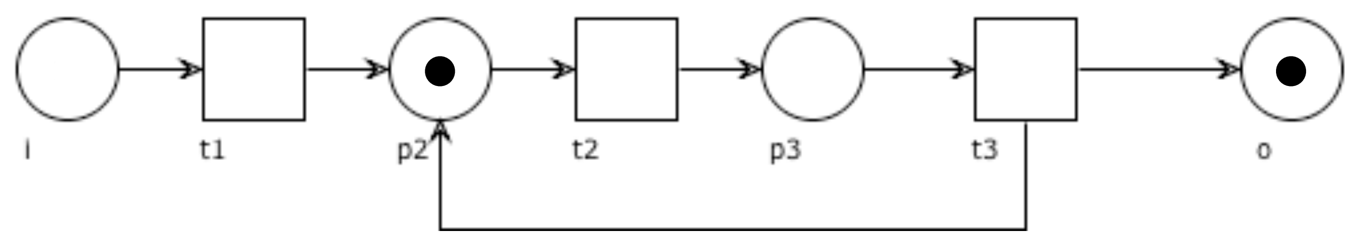
Behavioural analysis

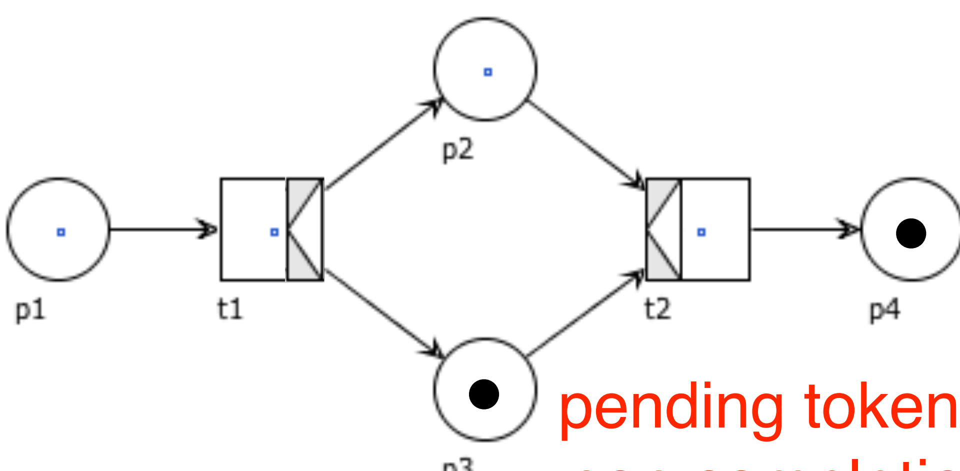
Structural correctness cannot rule out many other problematic issues...



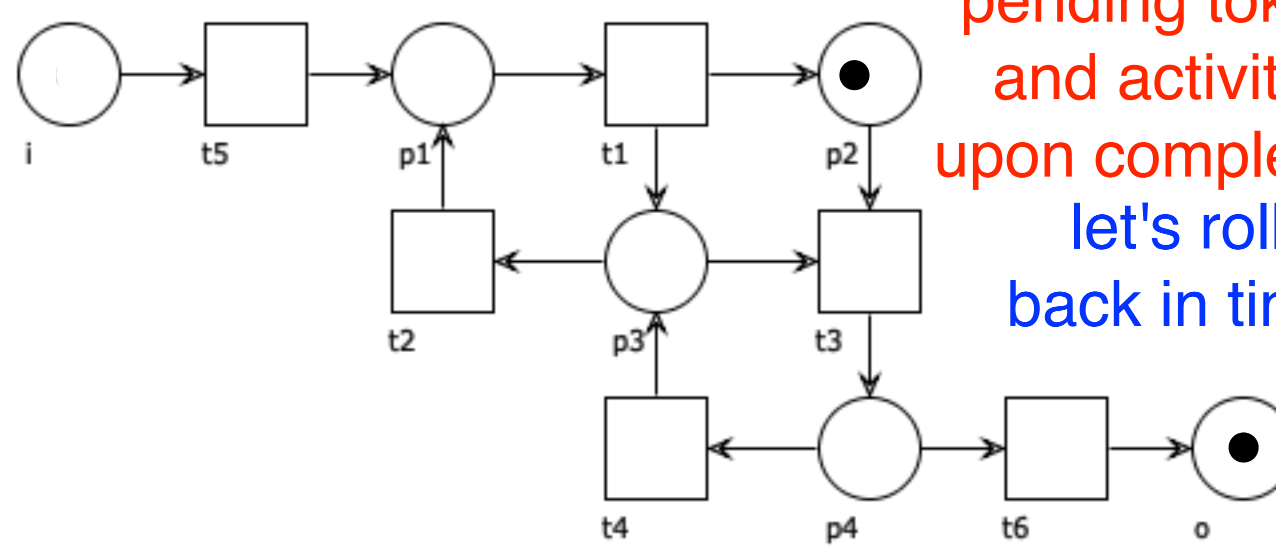
dead!

pending tokens and activities upon completion!





pending tokens upon completion!

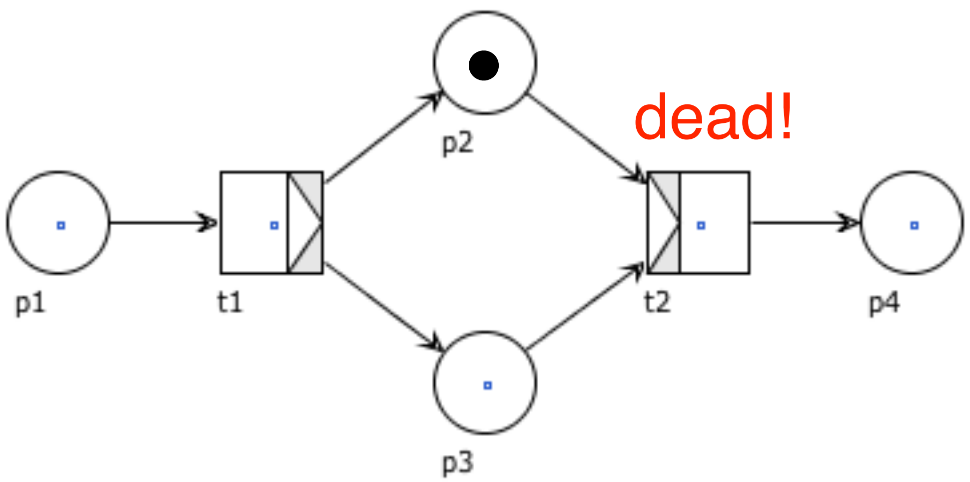


pending tokens and activities upon completion!

let's roll back in time

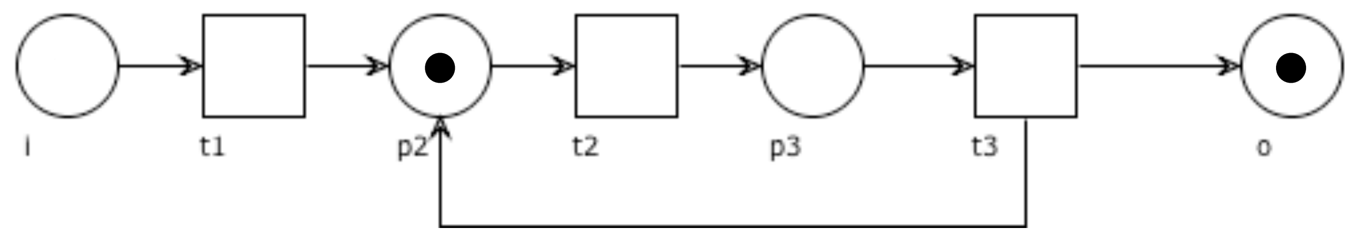
Behavioural analysis

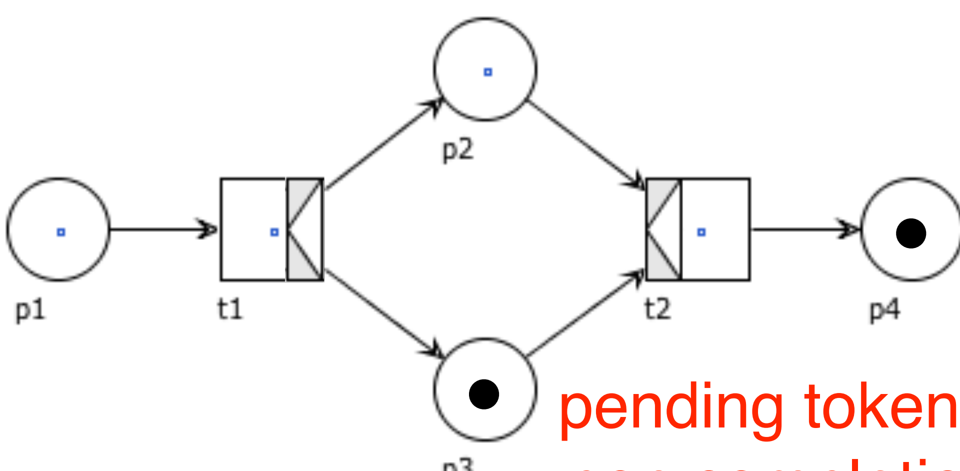
Structural correctness cannot rule out many other problematic issues...



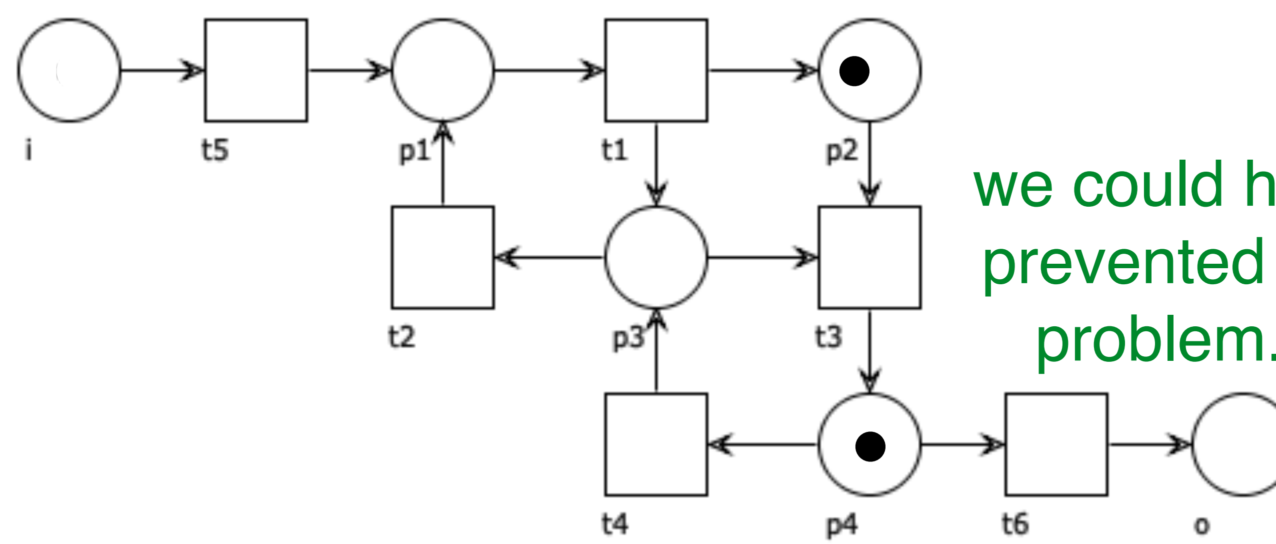
dead!

pending tokens and activities upon completion!





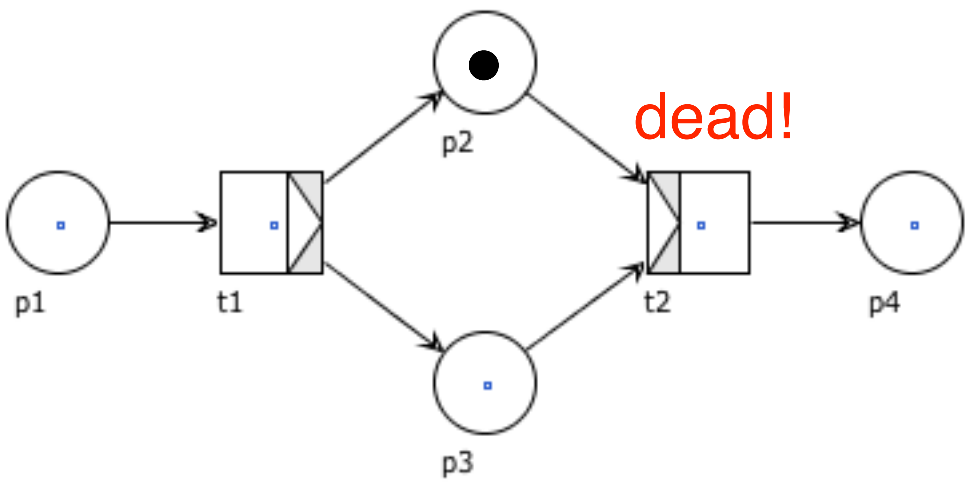
pending tokens upon completion!



we could have prevented the problem...

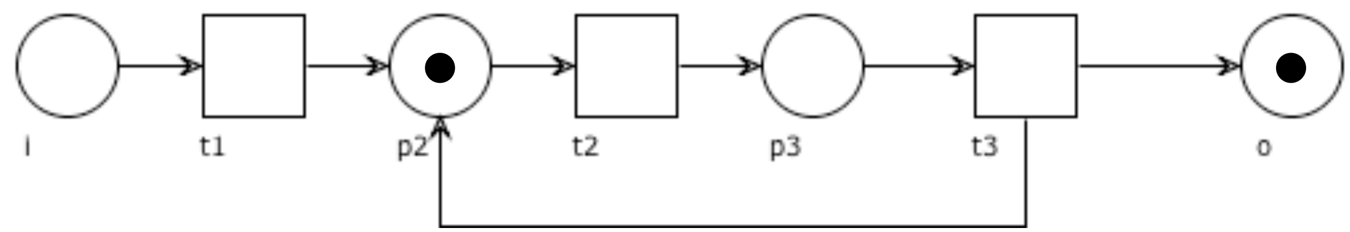
Behavioural analysis

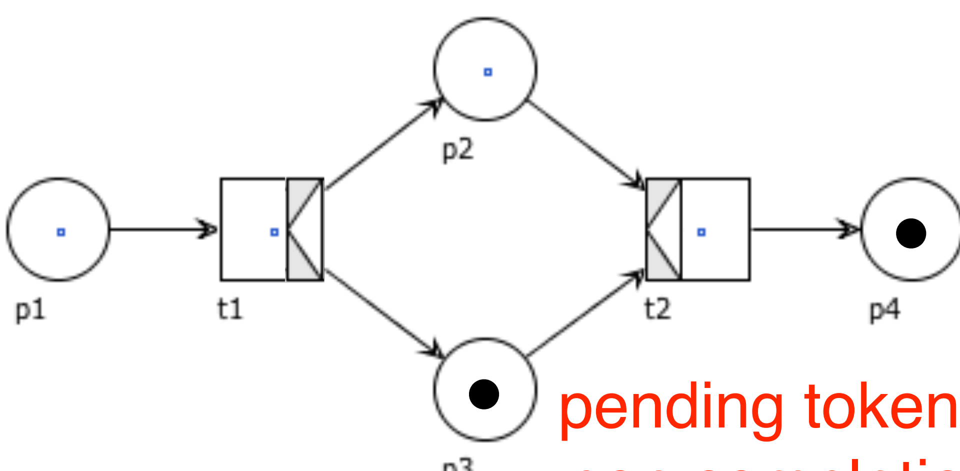
Structural correctness cannot rule out many other problematic issues...



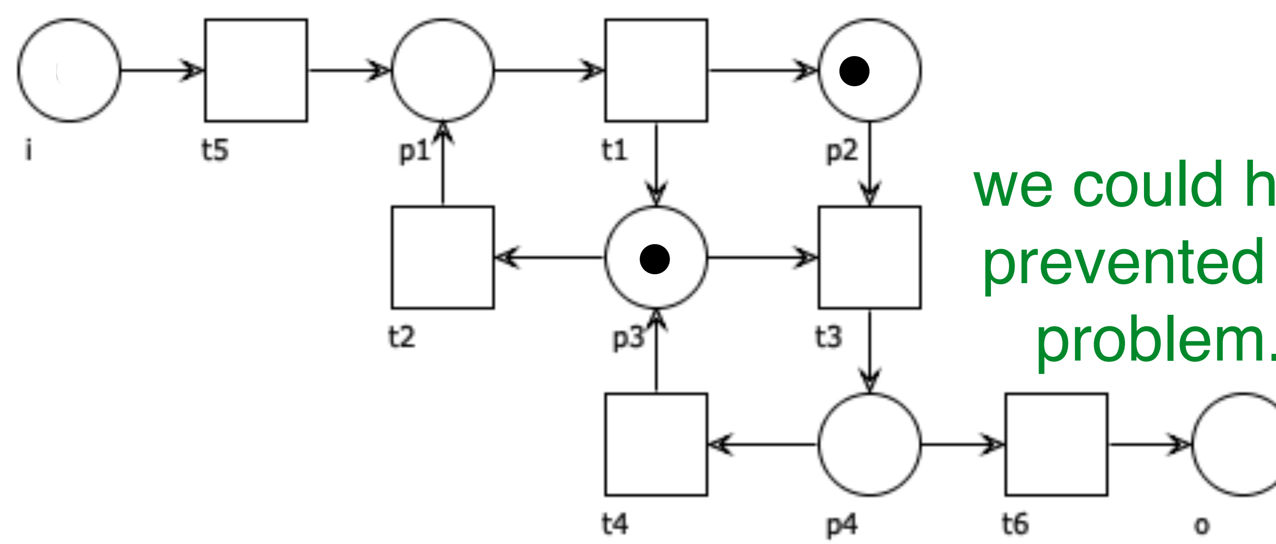
dead!

pending tokens and activities upon completion!





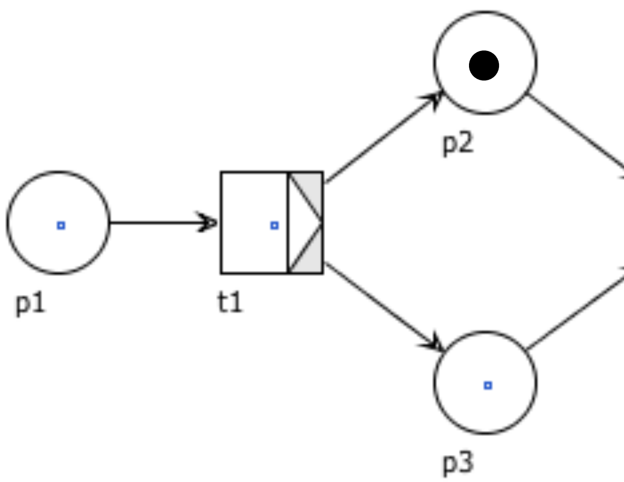
pending tokens upon completion!



we could have prevented the problem...

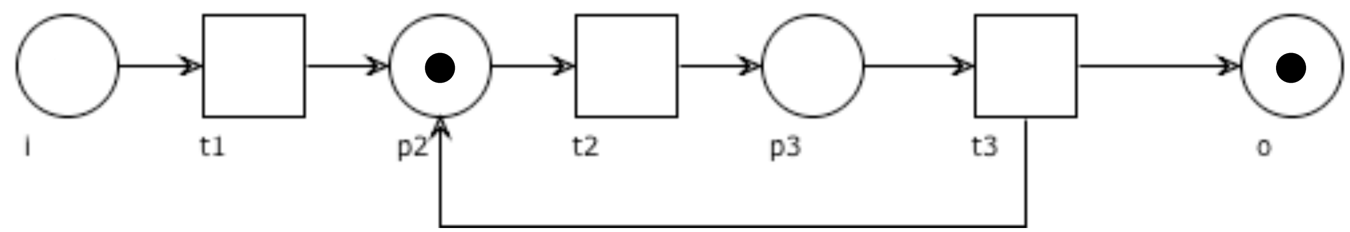
Behavioural analysis

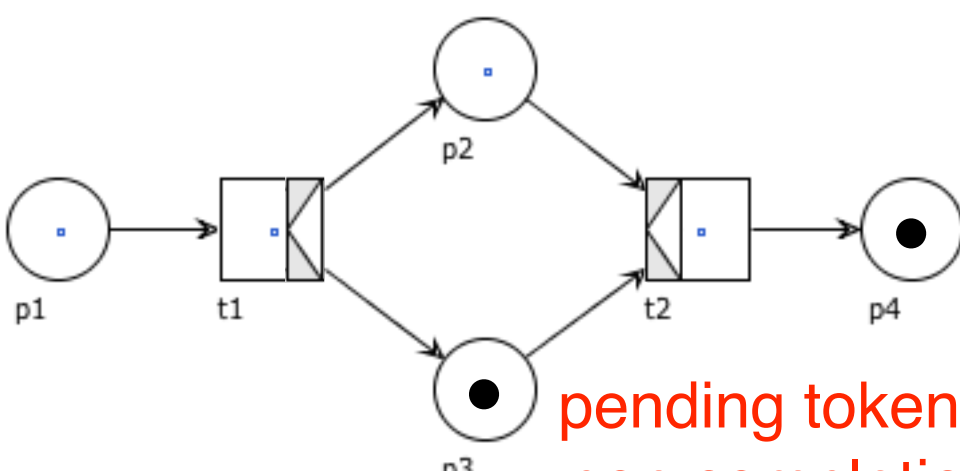
Structural correctness cannot rule out many other problematic issues...



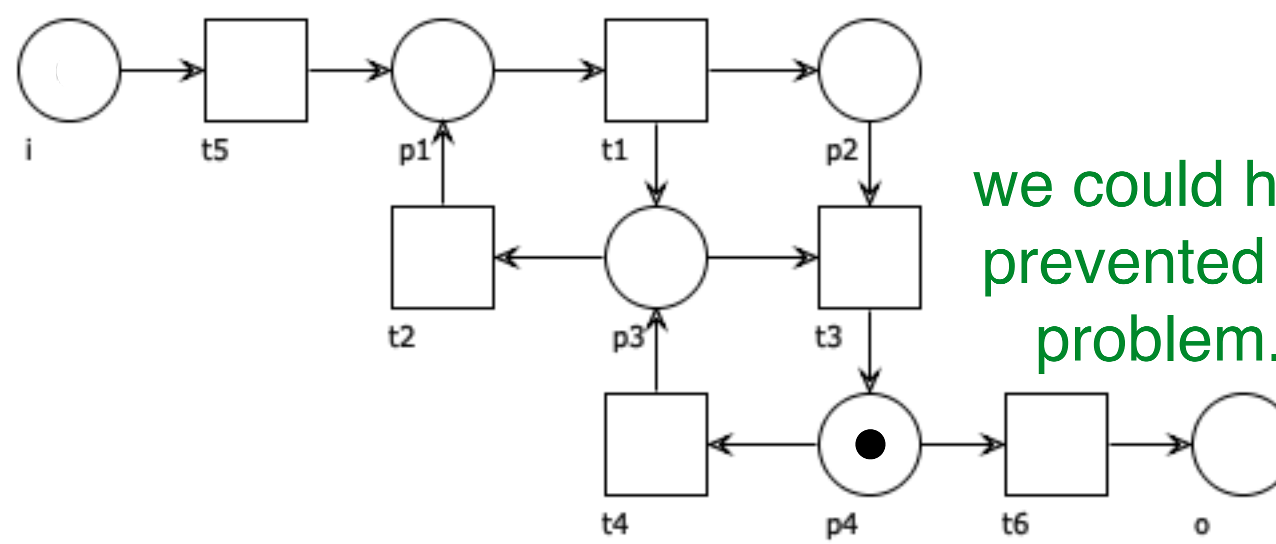
dead!

pending tokens and activities upon completion!





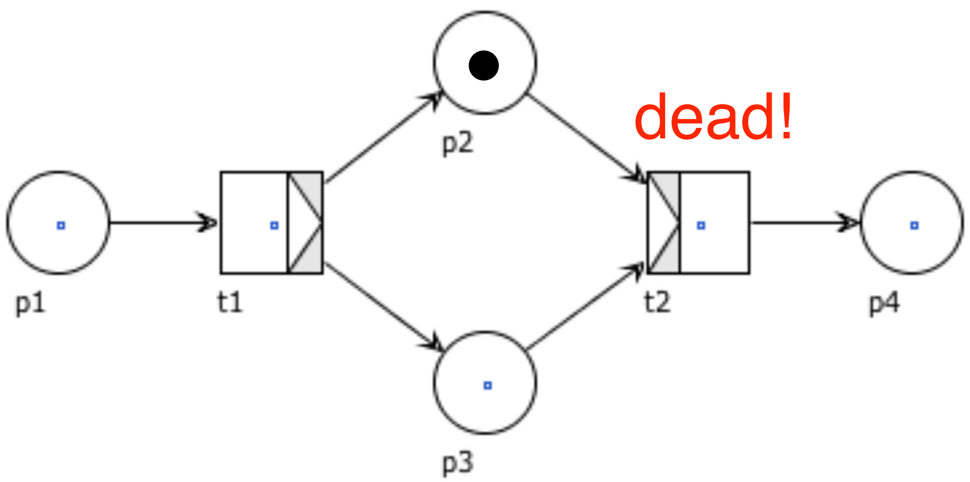
pending tokens upon completion!



we could have prevented the problem...

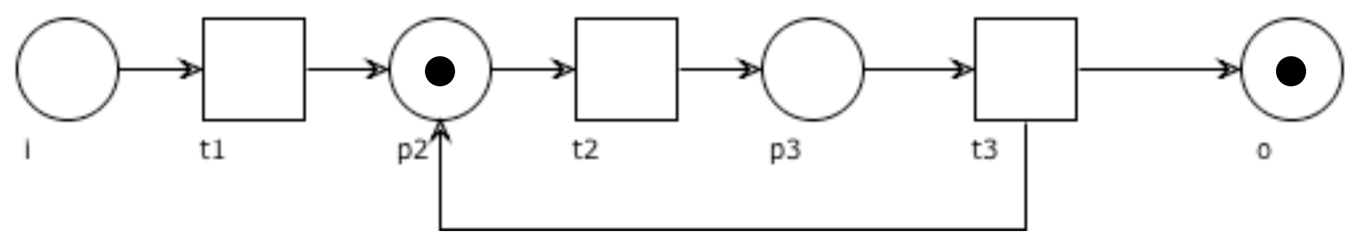
Behavioural analysis

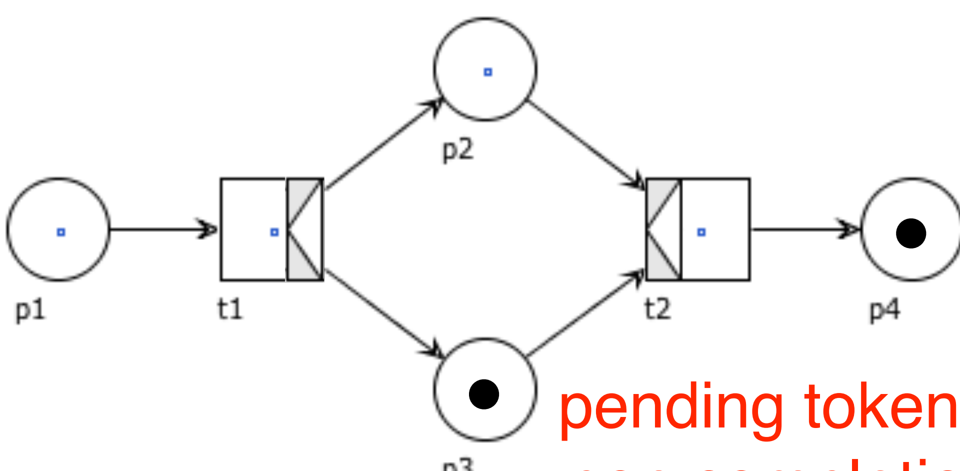
Structural correctness cannot rule out many other problematic issues...



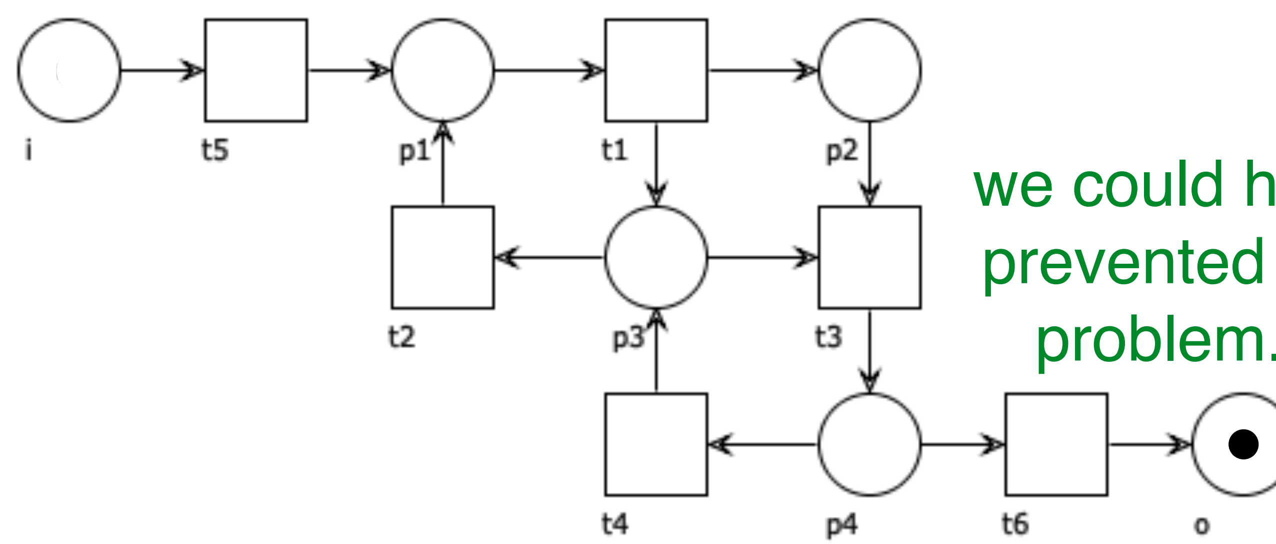
dead!

pending tokens and activities upon completion!





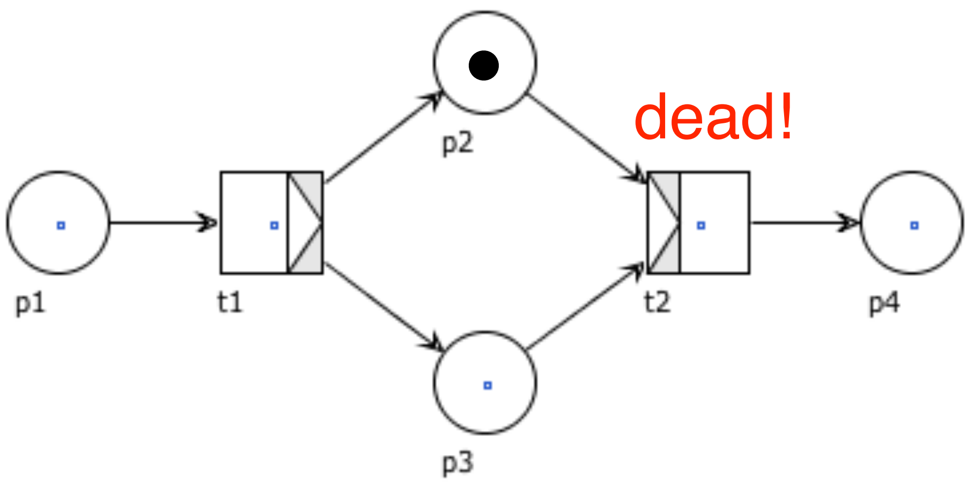
pending tokens upon completion!



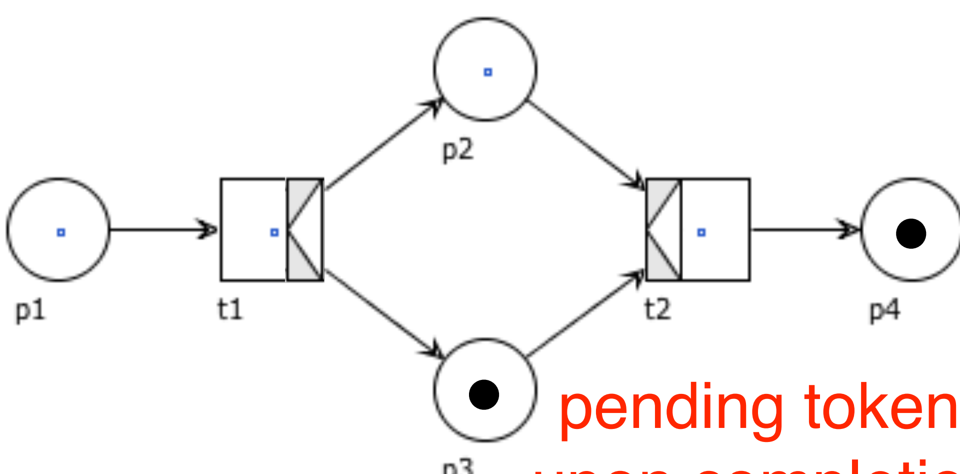
we could have prevented the problem...

Behavioural analysis

Structural correctness cannot rule out many other problematic issues...

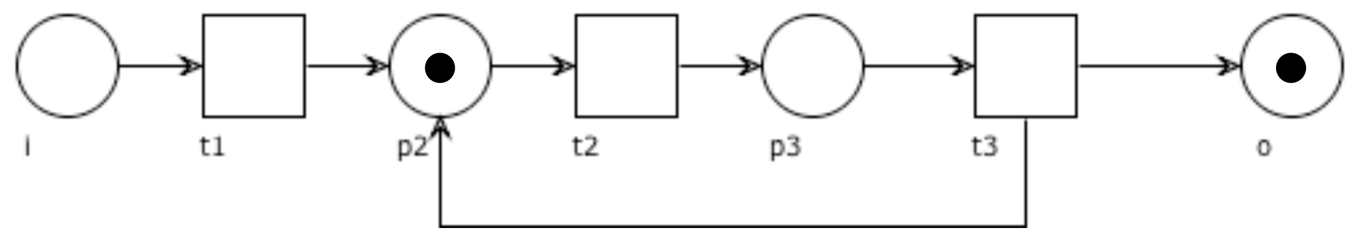


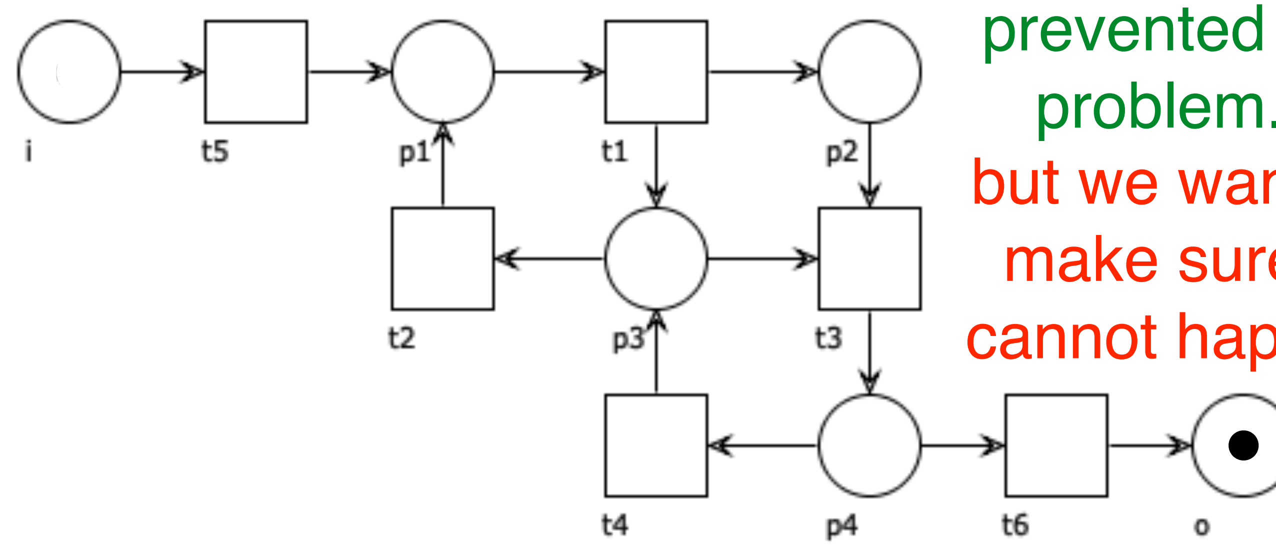
dead!



pending tokens upon completion!

pending tokens and activities upon completion!



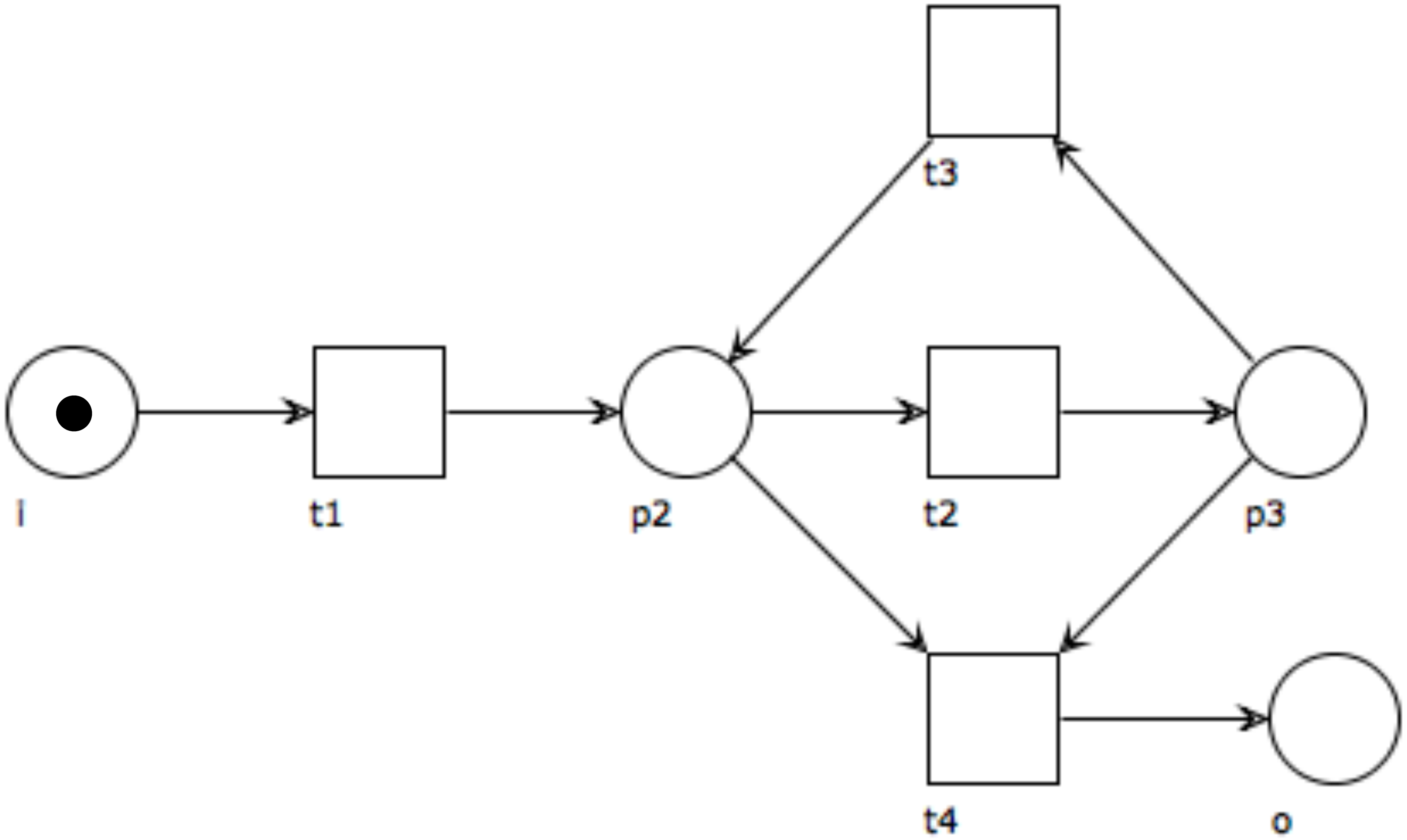


we could have prevented the problem...

but we want to make sure it cannot happen

Livelock

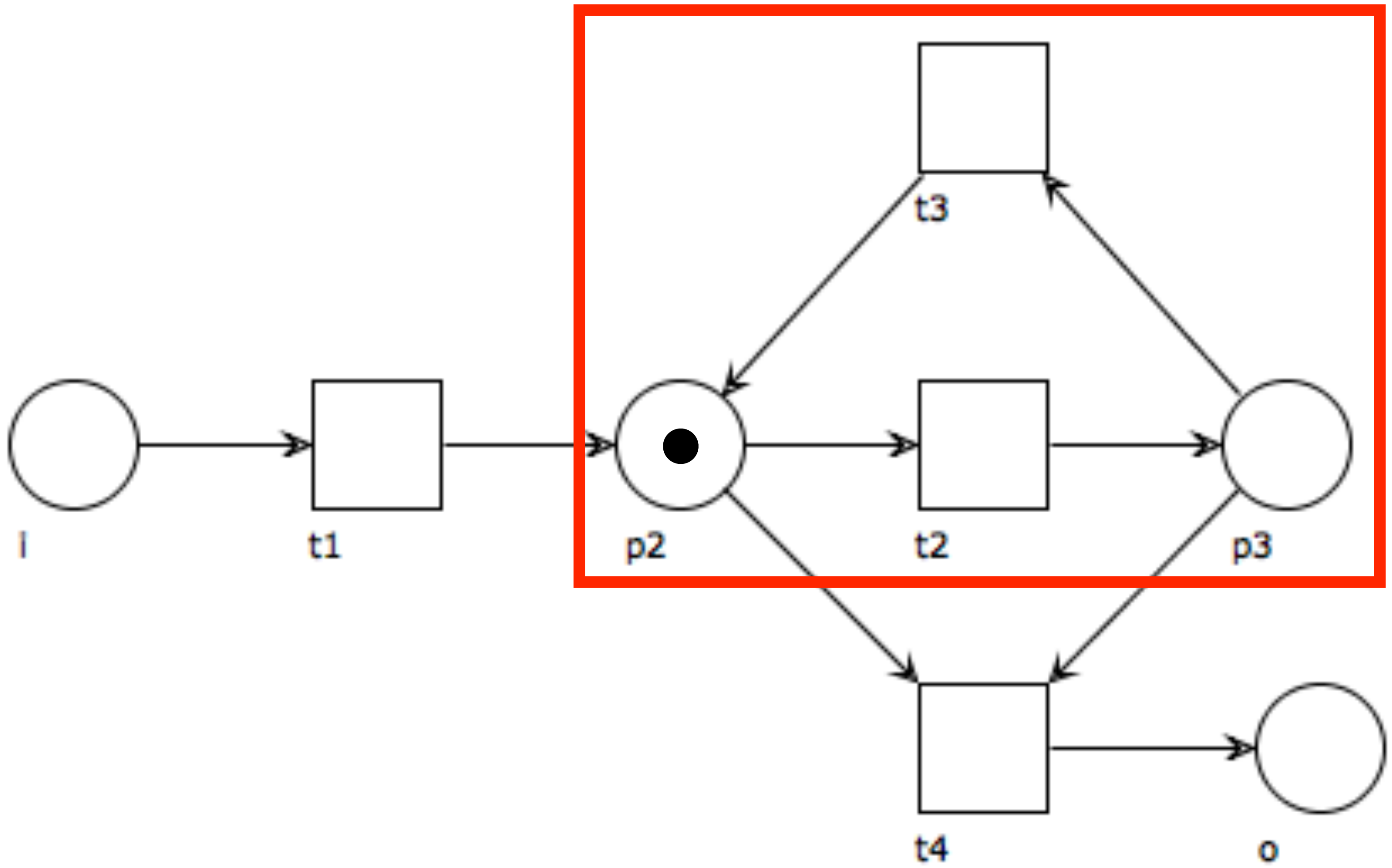
A case is trapped in a cycle with no opportunity to end



can arise in workflow nets

Livelock

A case is trapped in a cycle with no opportunity to end



can arise in workflow nets

Remark

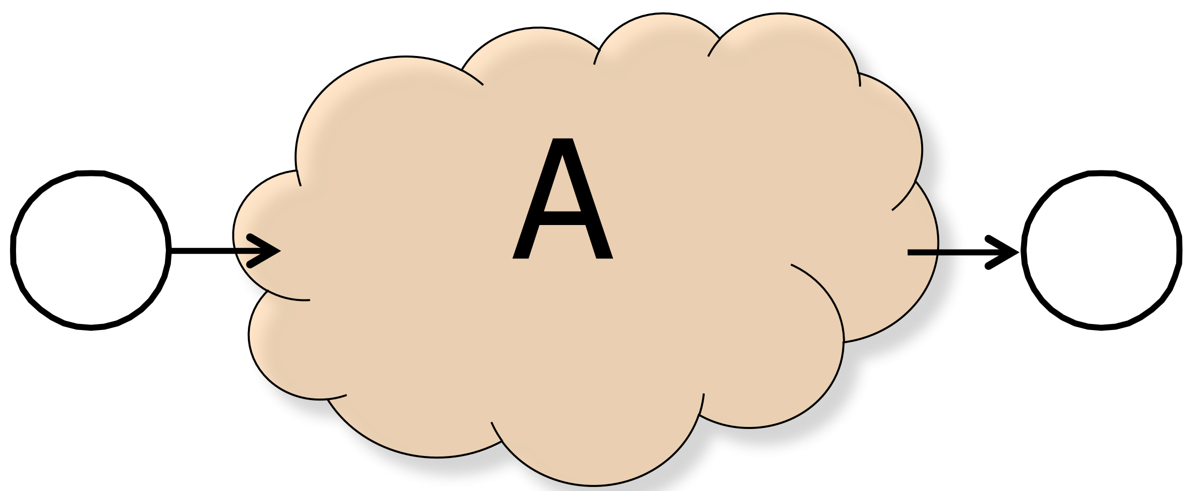
All the previous flaws are typical errors that can be detected **without any knowledge** about the actual goal of the Business Process

System validation and verification

Validation is concerned with
the relation between the model and the reality
How does a model fit log files?
Which model does fit better?

Verification aims to answer qualitative questions
Is there a deadlock possible?
Is it possible to successfully handle a specific case?
Will all cases terminate eventually?
Is it possible to execute a certain task?

Language of a workflow net

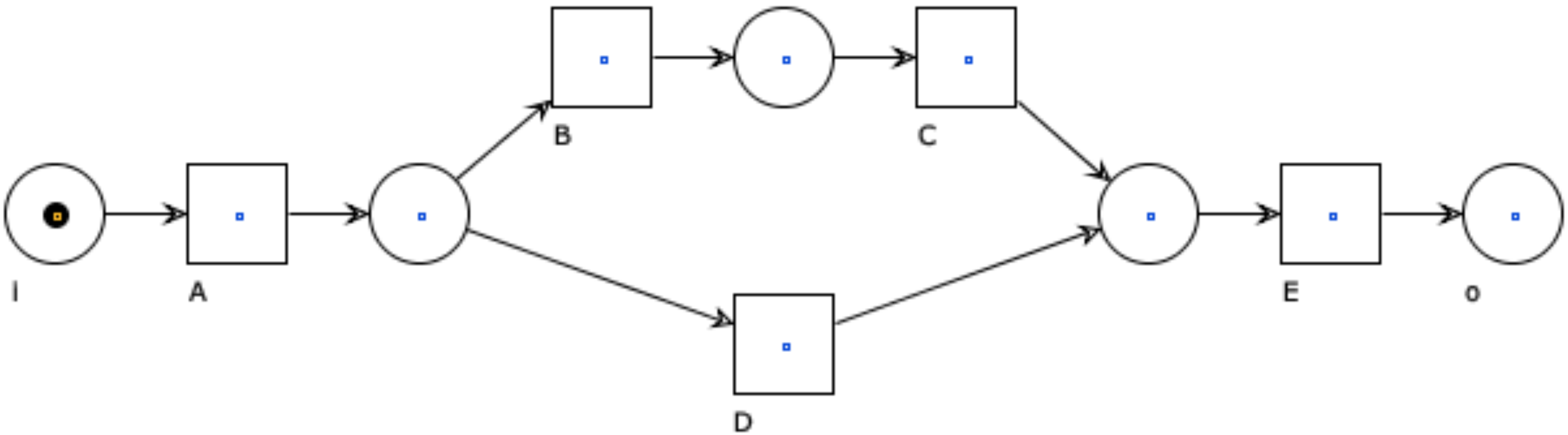


The language of a workflow net is the set of firing sequences that lead from marking i to marking o

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

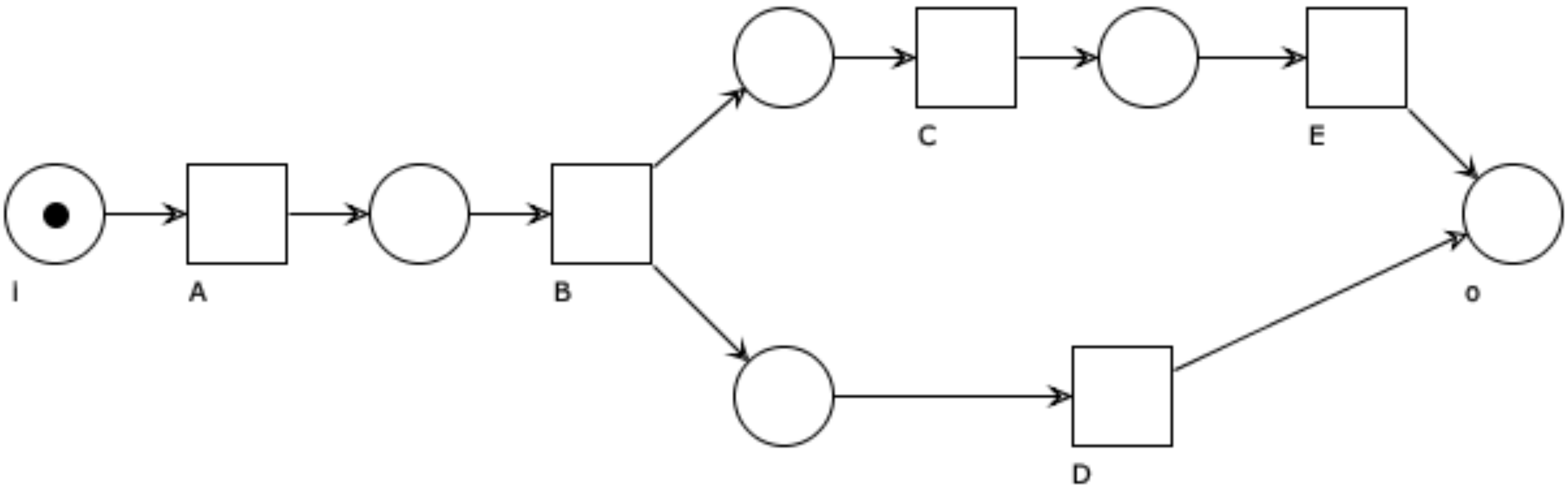
$L(N)$ defines all the admissible traces of the workflow

Question time: L(N)



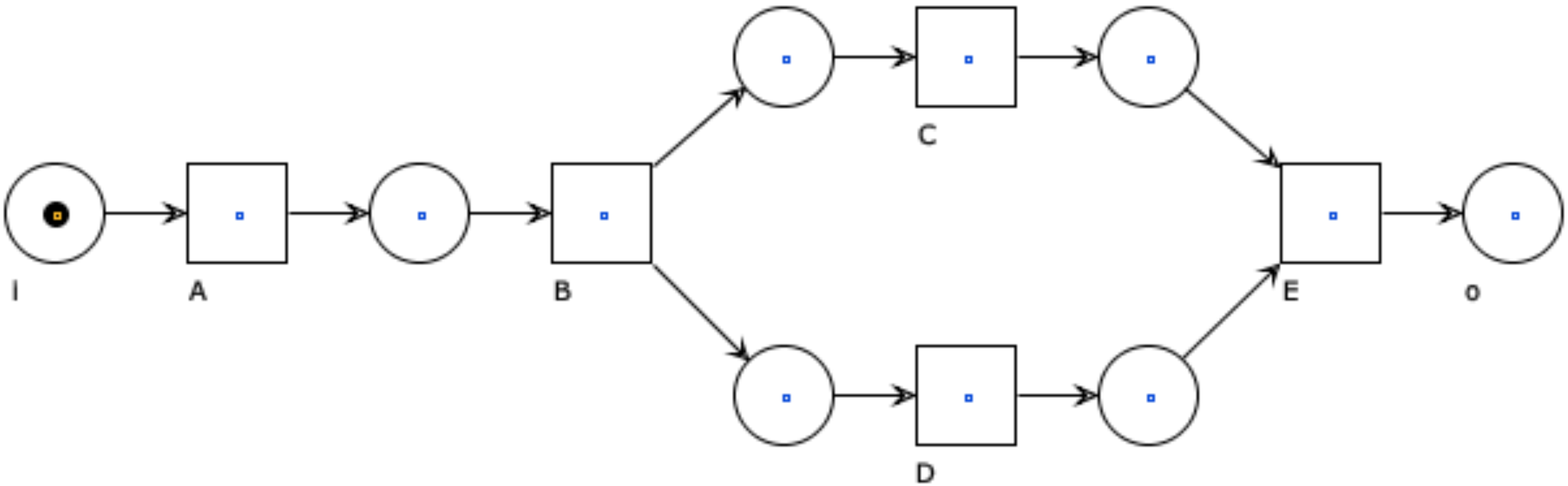
$$L(N) = \{\ \text{ABCE}\ ,\ \text{ADE}\ \}$$

Question time: L(N)



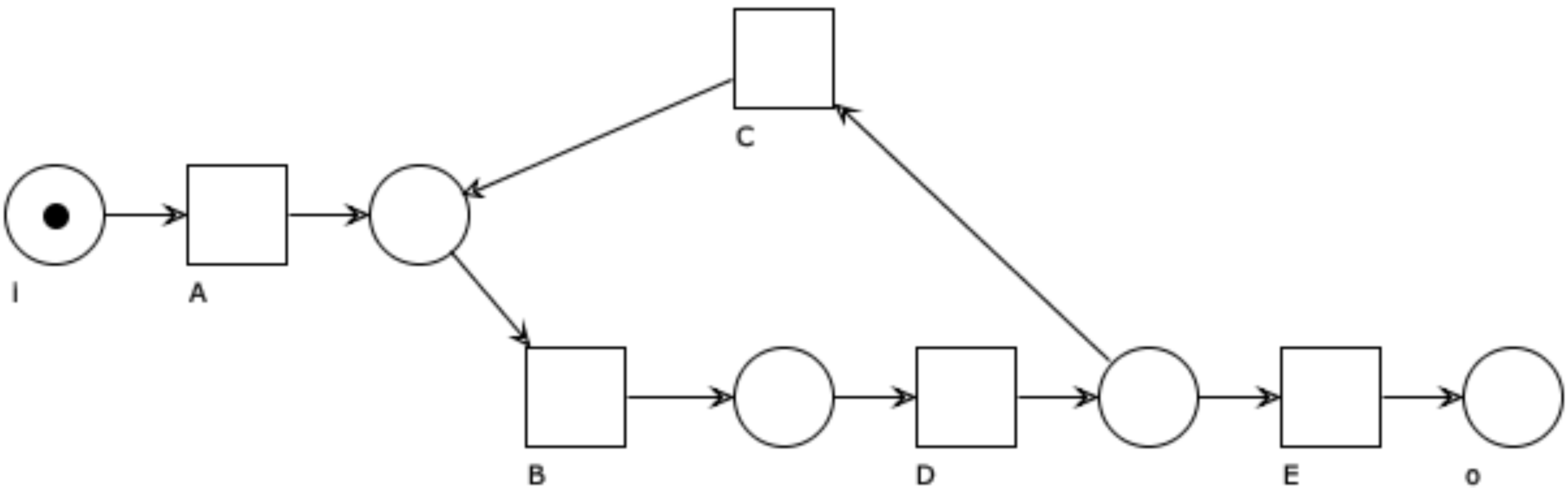
$$L(N) = \emptyset$$

Question time: L(N)



$L(N) = \{$ ABCDE , ABDCE $\}$

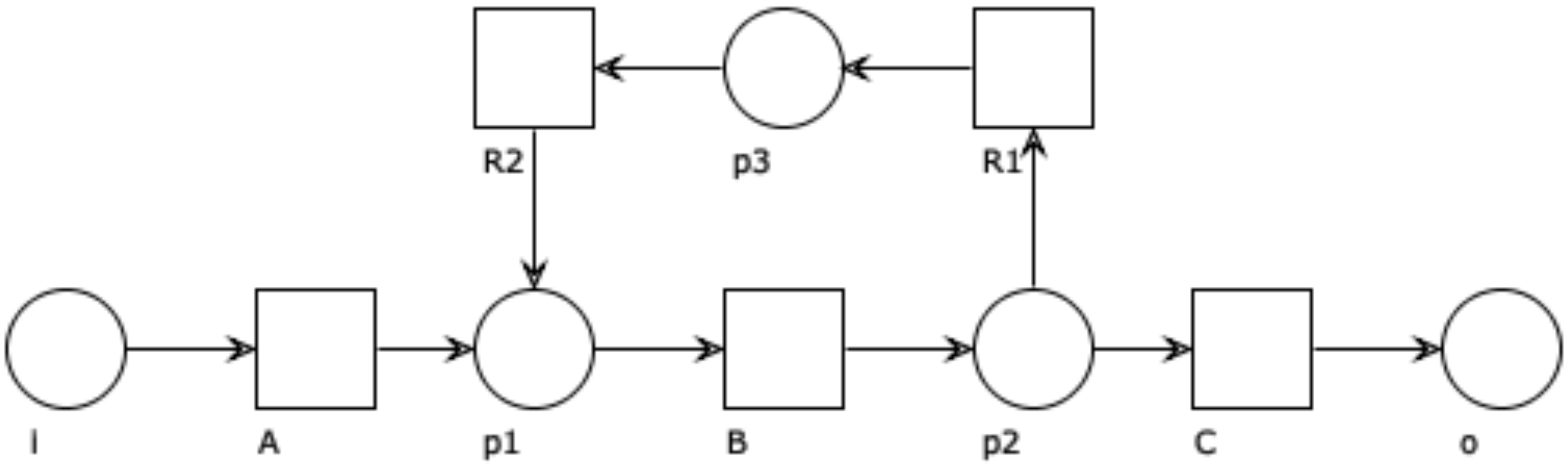
Question time: L(N)



$L(N) =$ { ABDE , ABDCBDE , ABDCBDCBDE , ABDCBDCBDCBDE , ABDCBDCBDCBDCBDE , ... }

$L(N) =$ { ABD(CBD)^{k}E $\mid k \geq 0$}

Question time: L(N)



$L(N) \stackrel{?}{=} \{\text{A (B R1 R2)}^k \text{ C} \mid k \geq 0\}$ No

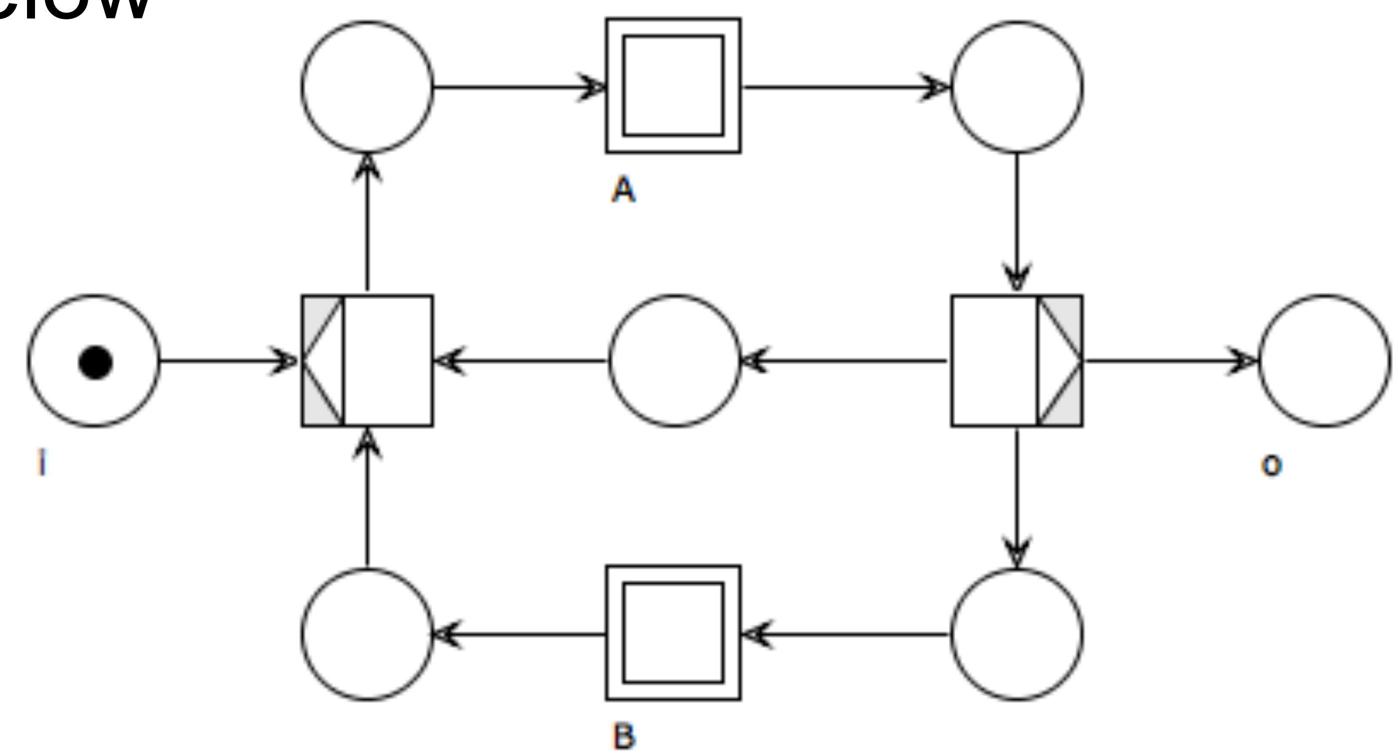
$L(N) \stackrel{?}{=} \{\text{A (B R1 R2 B)}^k \text{ C} \mid k \geq 0\}$ No

$L(N) \stackrel{?}{=} \{\text{A B (R1 R2 B)}^k \text{ C} \mid k \geq 0\}$ Yes

$L(N) \stackrel{?}{=} \{\text{A (B R1 R2)}^k \text{ B C} \mid k \geq 0\}$ Yes

Question time

Consider the workflow net below



How many times can A be executed?

How many times can B be executed?

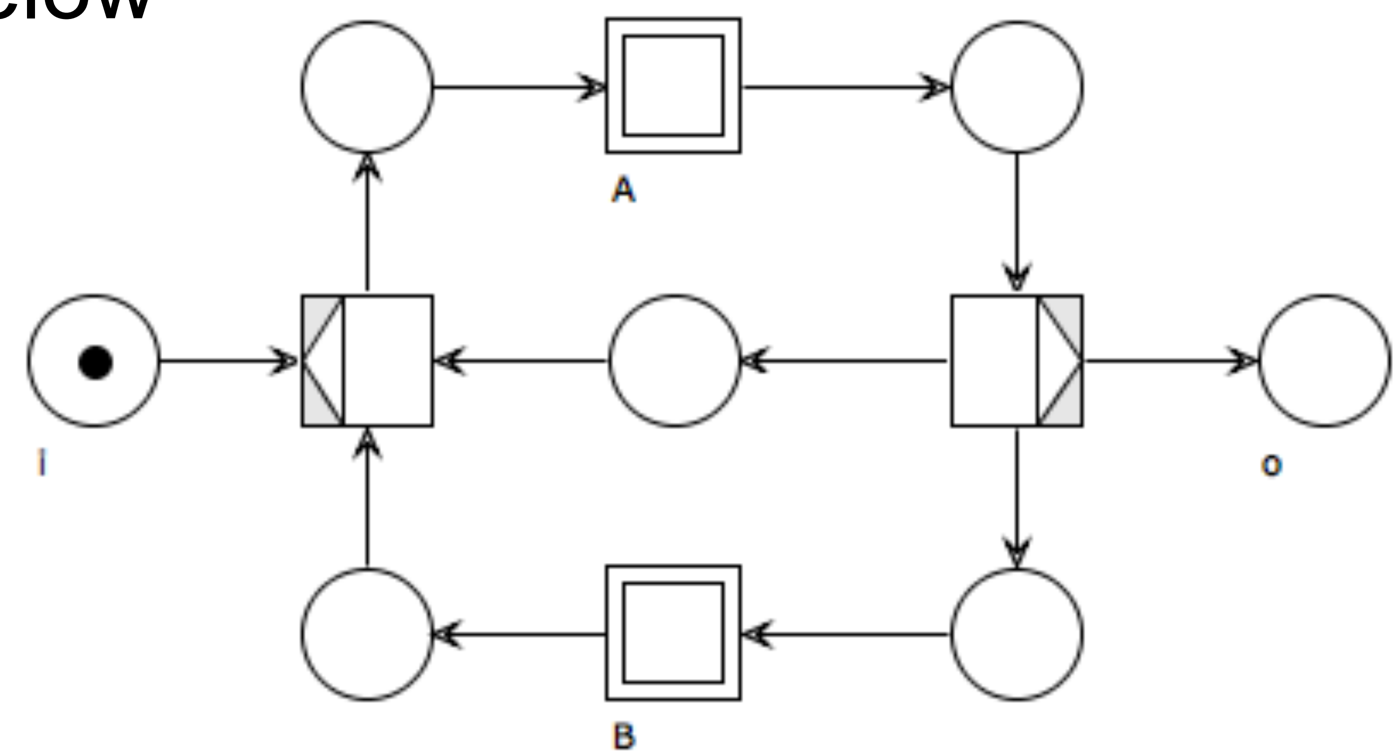
Can a firing sequence contain two As in a row?

Can a firing sequence contain two Bs in a row?

Can a firing sequence contain more Bs than As?

Question time

Consider the workflow net below



How many times can A be executed? 1 or more

How many times can B be executed? 0 or more

Can a firing sequence contain two As in a row? yes

Can a firing sequence contain two Bs in a row? no

Can a firing sequence contain more Bs than As? no

Simulation

Test analysis
Try and see which firing sequences are possible

Using WoPeD:
Play (forward and backward) with net tokens
Record certain runs (to replay or explain)
Randomly select alternatives

Problem: how to make sure that all possible runs have been examined?

Reachability analysis

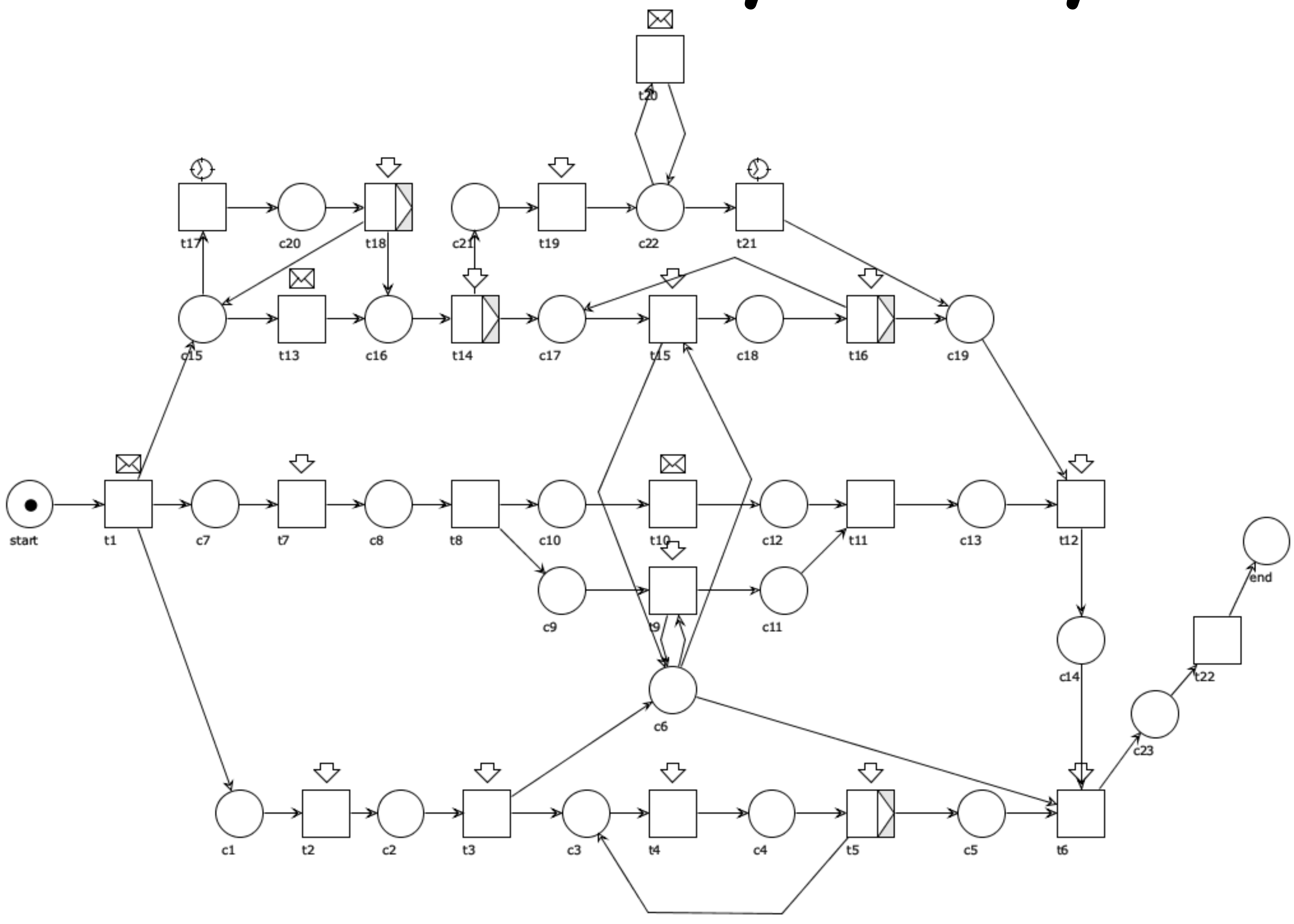
All possible runs of a workflow net are represented in its Reachability / Coverability Graph

Using WoPeD:
all reachable states are shown
(a single run does not necessarily visit all nodes)
End states are evident (no outgoing arc)

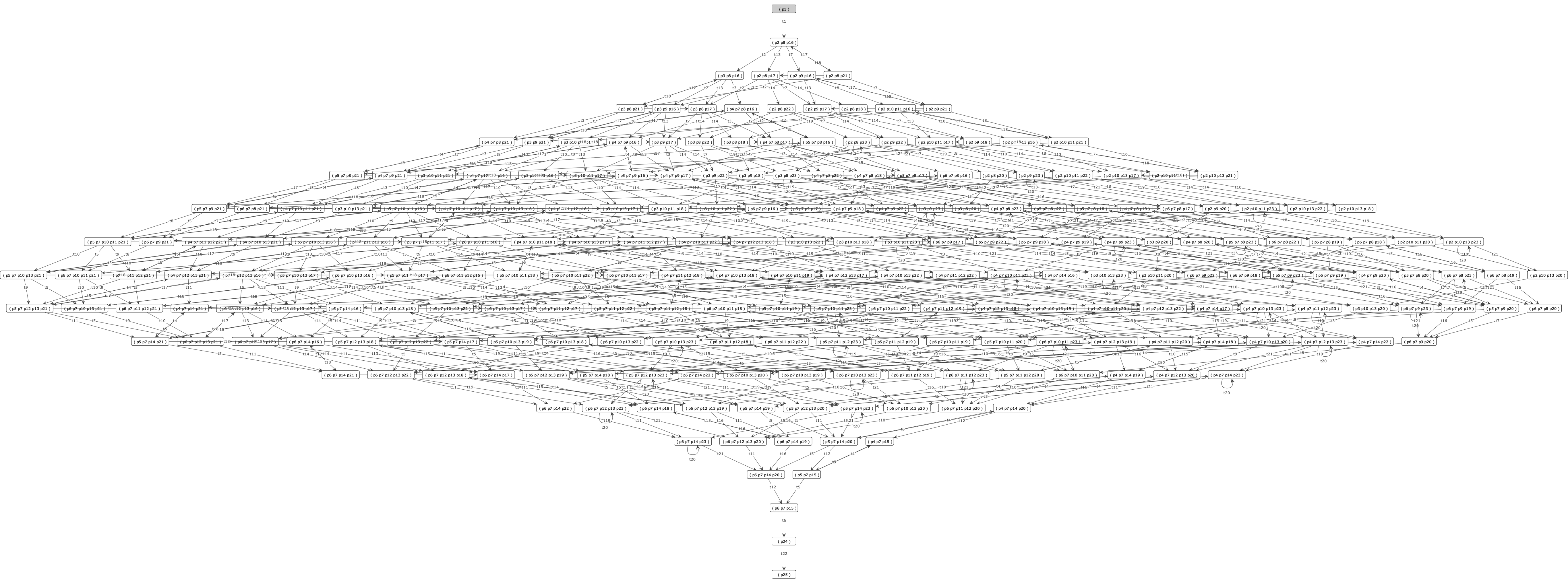
Useful to check if dangerous or undesired states can arise
(e.g. the green-green state in the two-traffic-lights)

Problem: state explosion

Reachability analysis



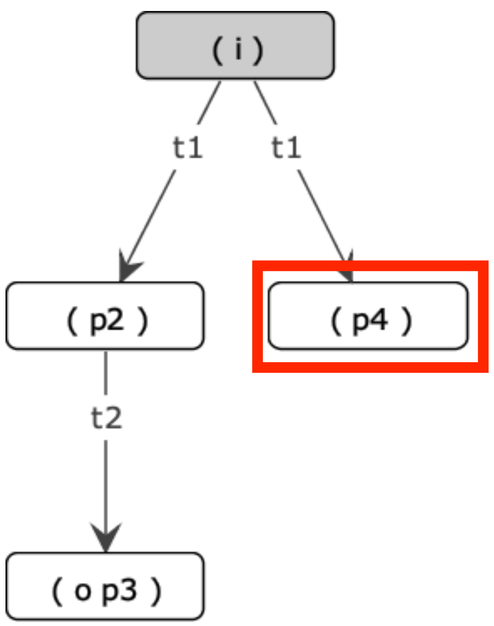
Reachability analysis

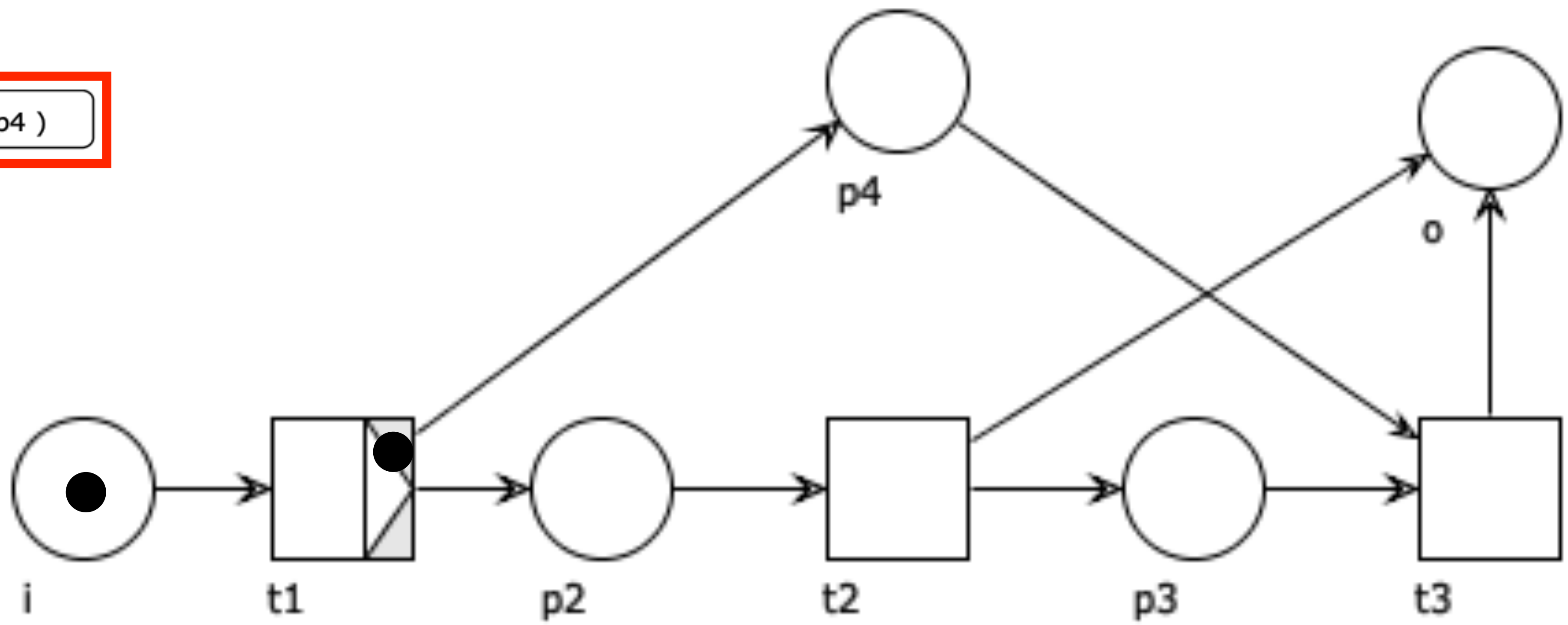


Problem: state explosion

Exercise

Do you see any problem in the workflow net below?

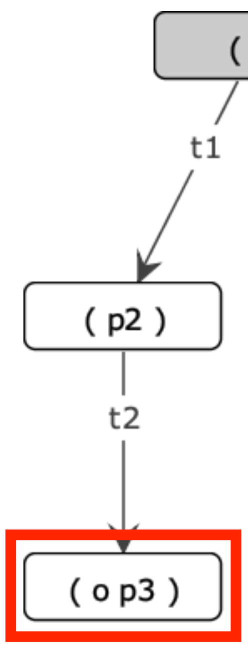


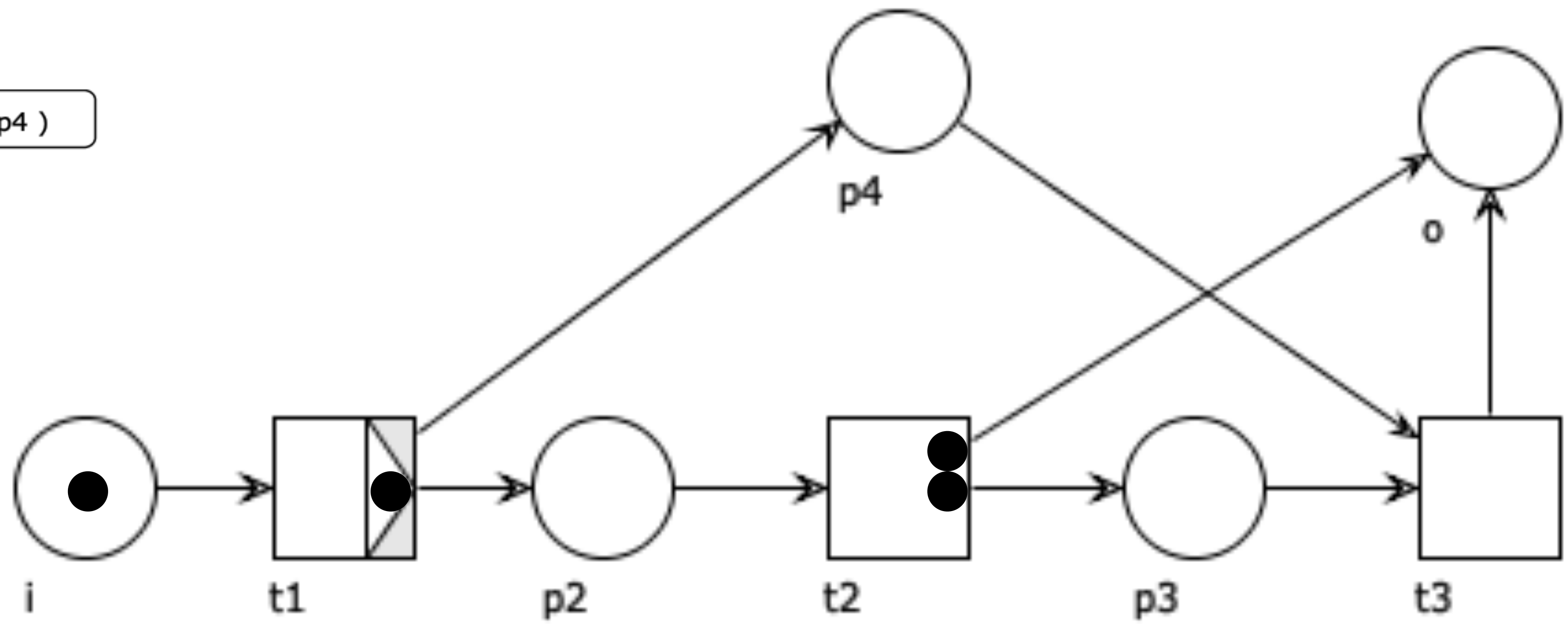


Deadlock

Exercise

Do you see any problem in the workflow net below?

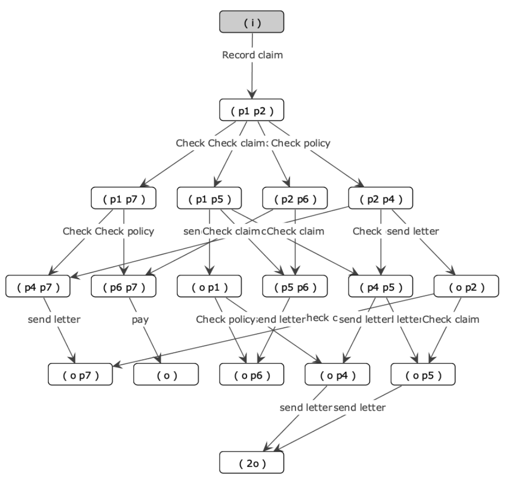


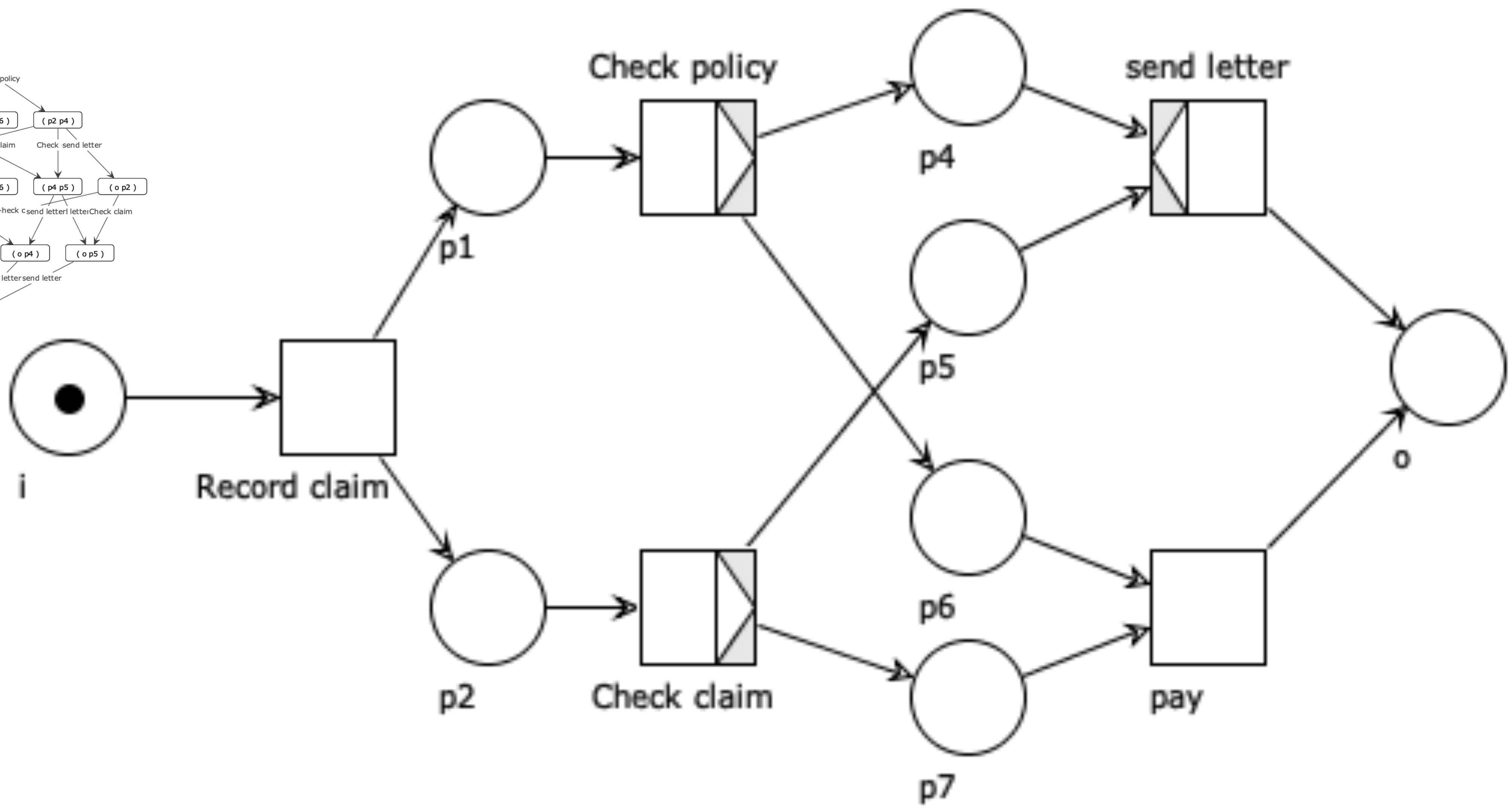


Some tokens left in the net after case completion

Exercise

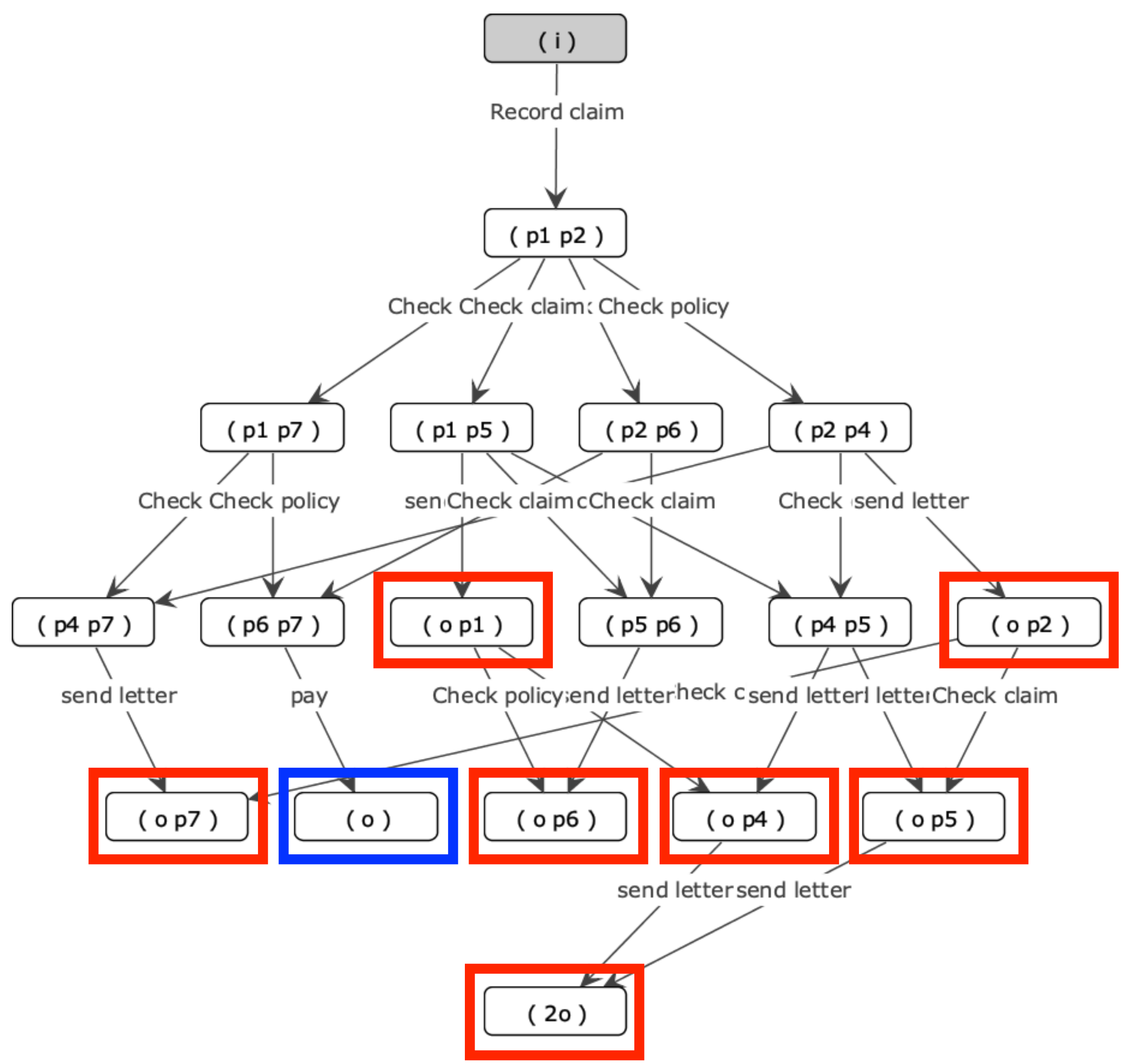
Which problem(s) in the workflow net below?
How would you redesign the business process?





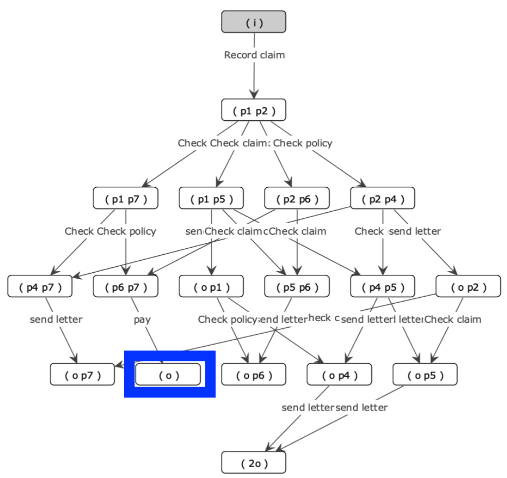
Exercise

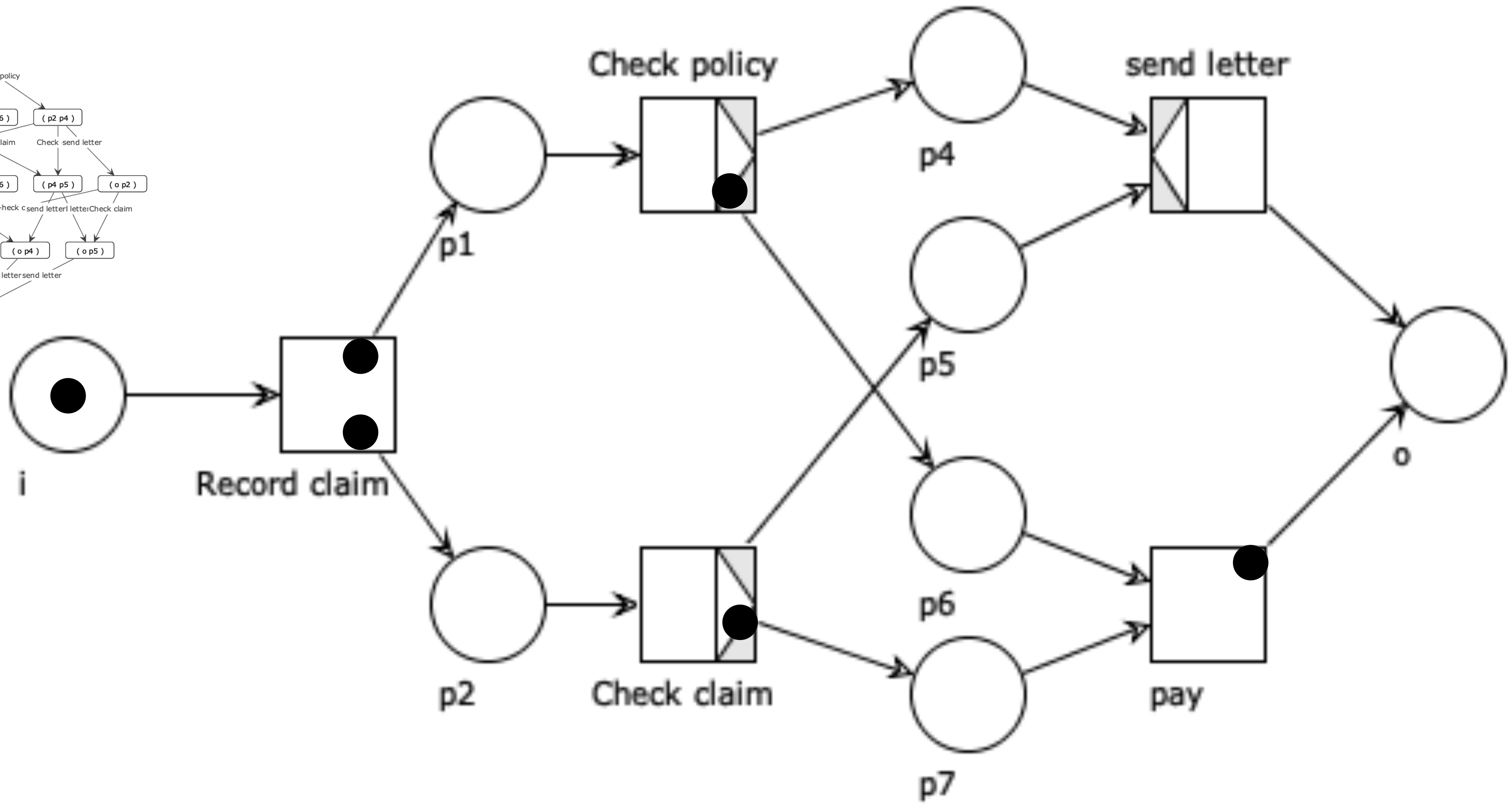
Which problem(s) in the workflow net below?
How would you redesign the business process?



Exercise

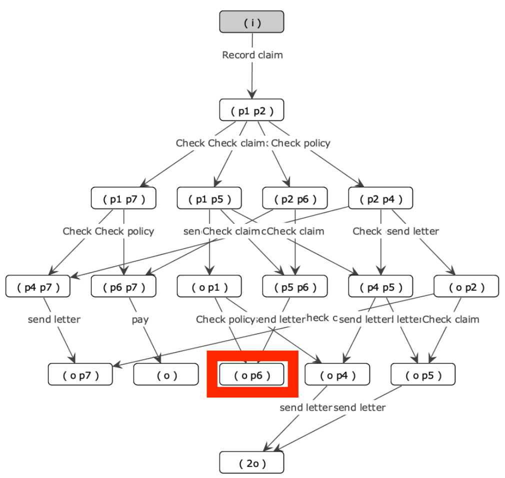
Which problem(s) in the workflow net below?
How would you redesign the business process?

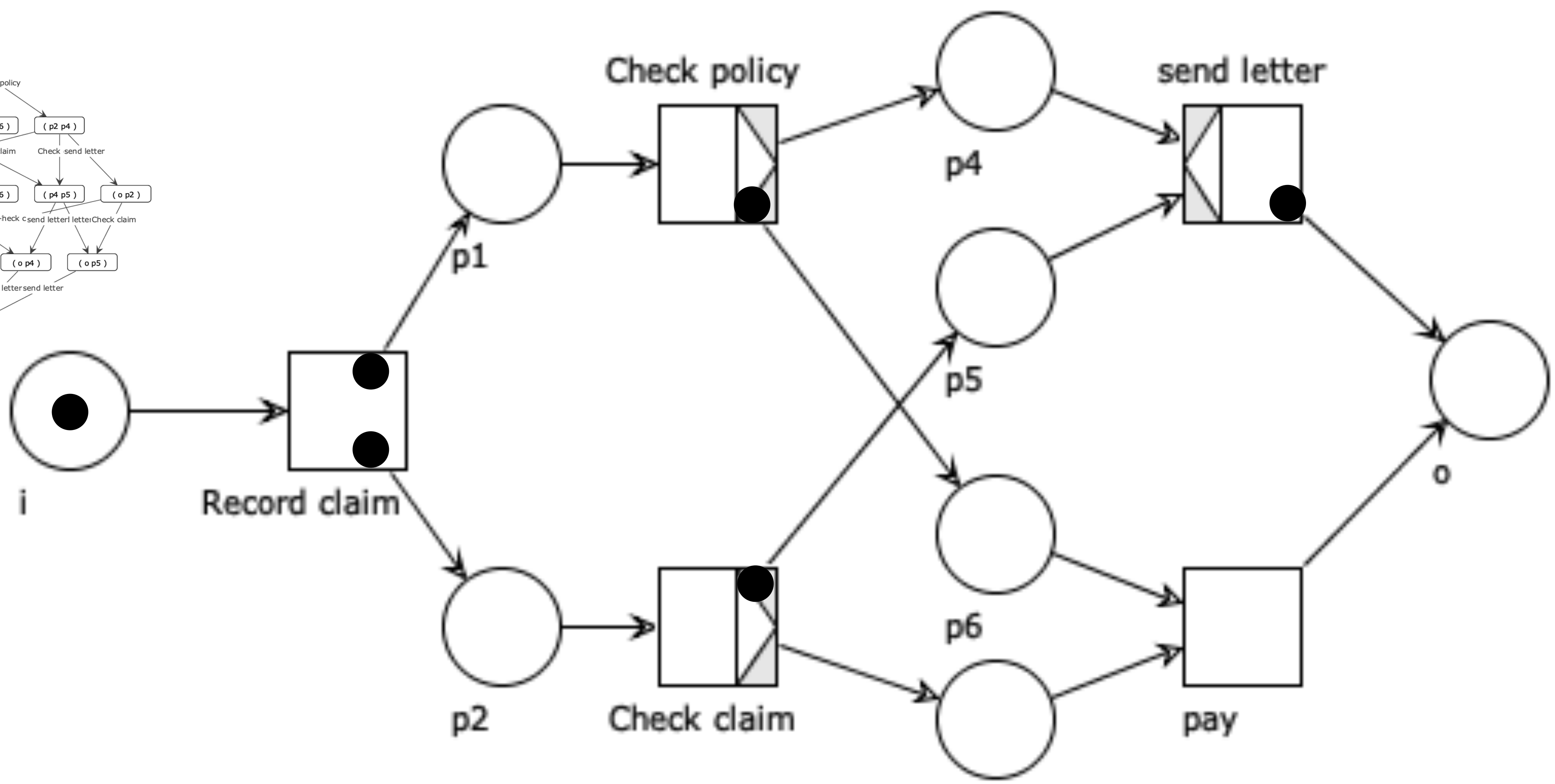




Exercise

Which problem(s) in the workflow net below?
How would you redesign the business process?

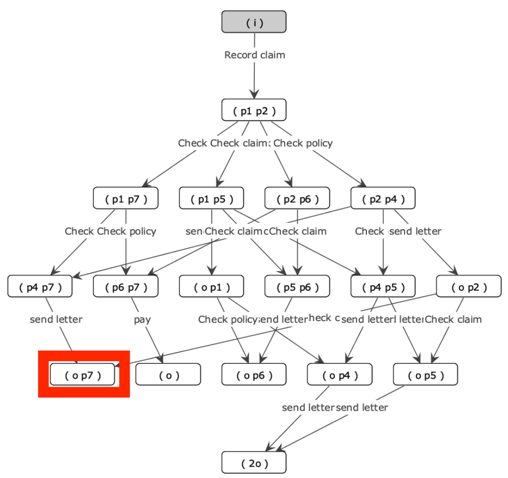


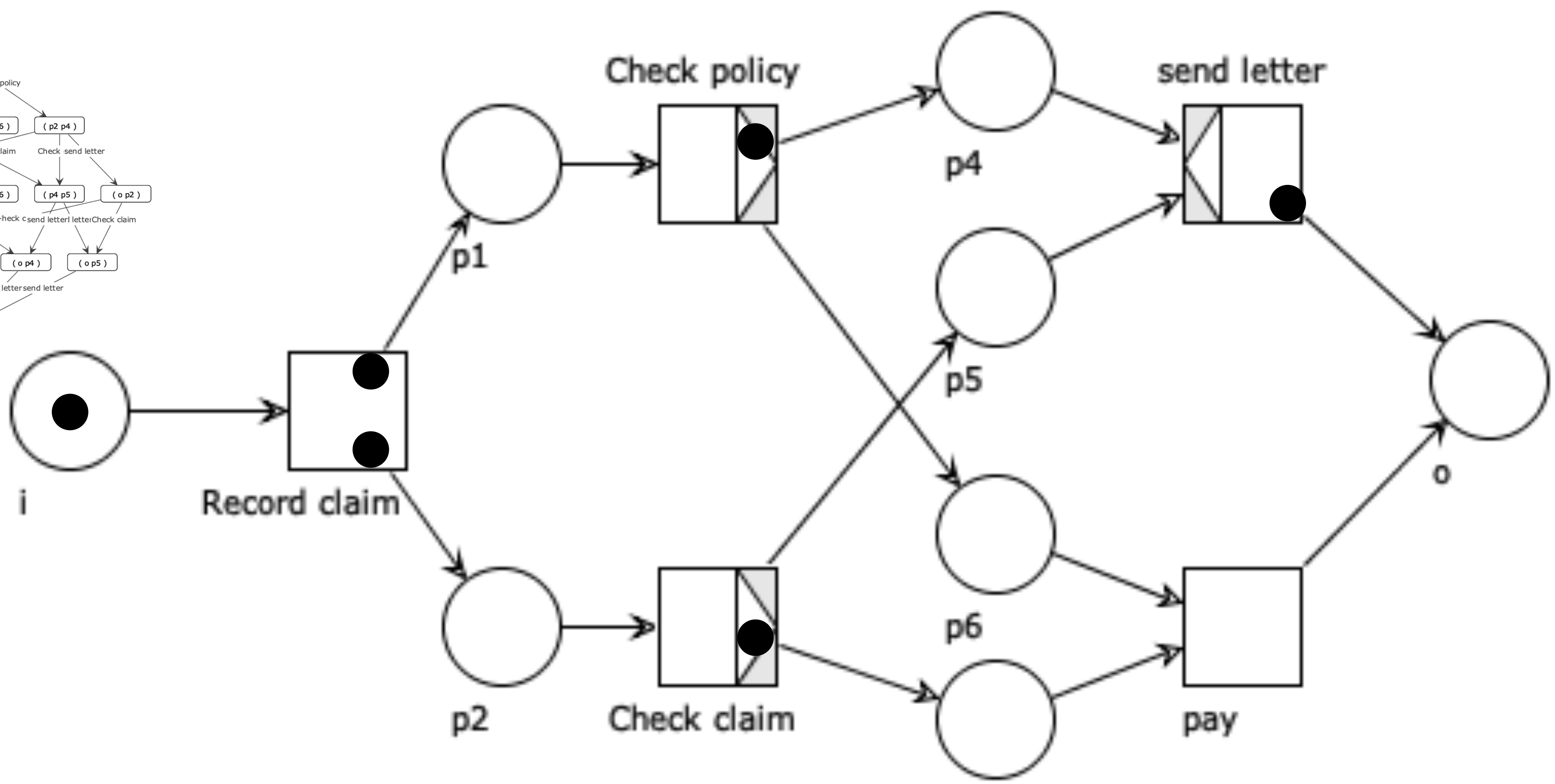


Some tokens left in the net after case completion

Exercise

Which problem(s) in the workflow net below?
How would you redesign the business process?

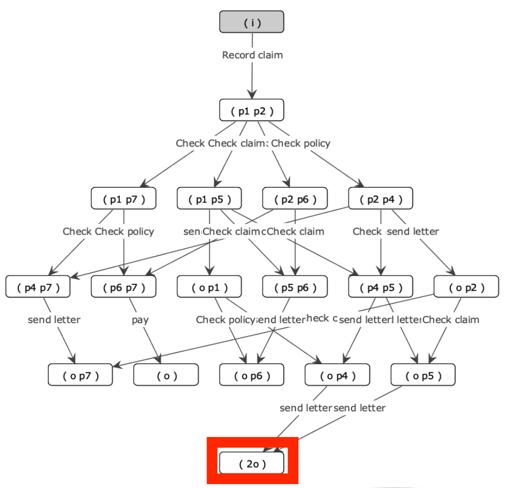


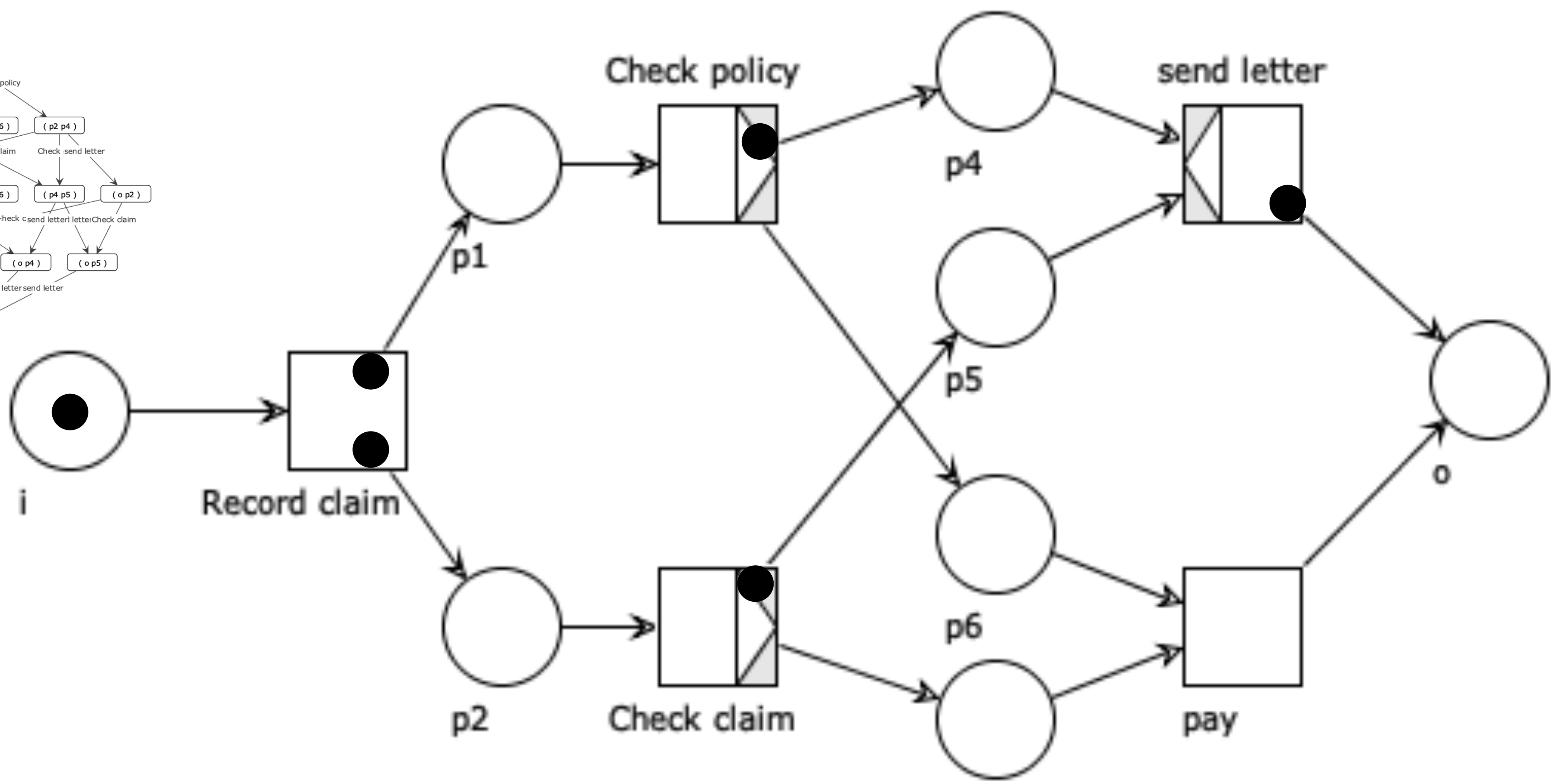


Some tokens left in the net after case completion

Exercise

Which problem(s) in the workflow net below?
How would you redesign the business process?

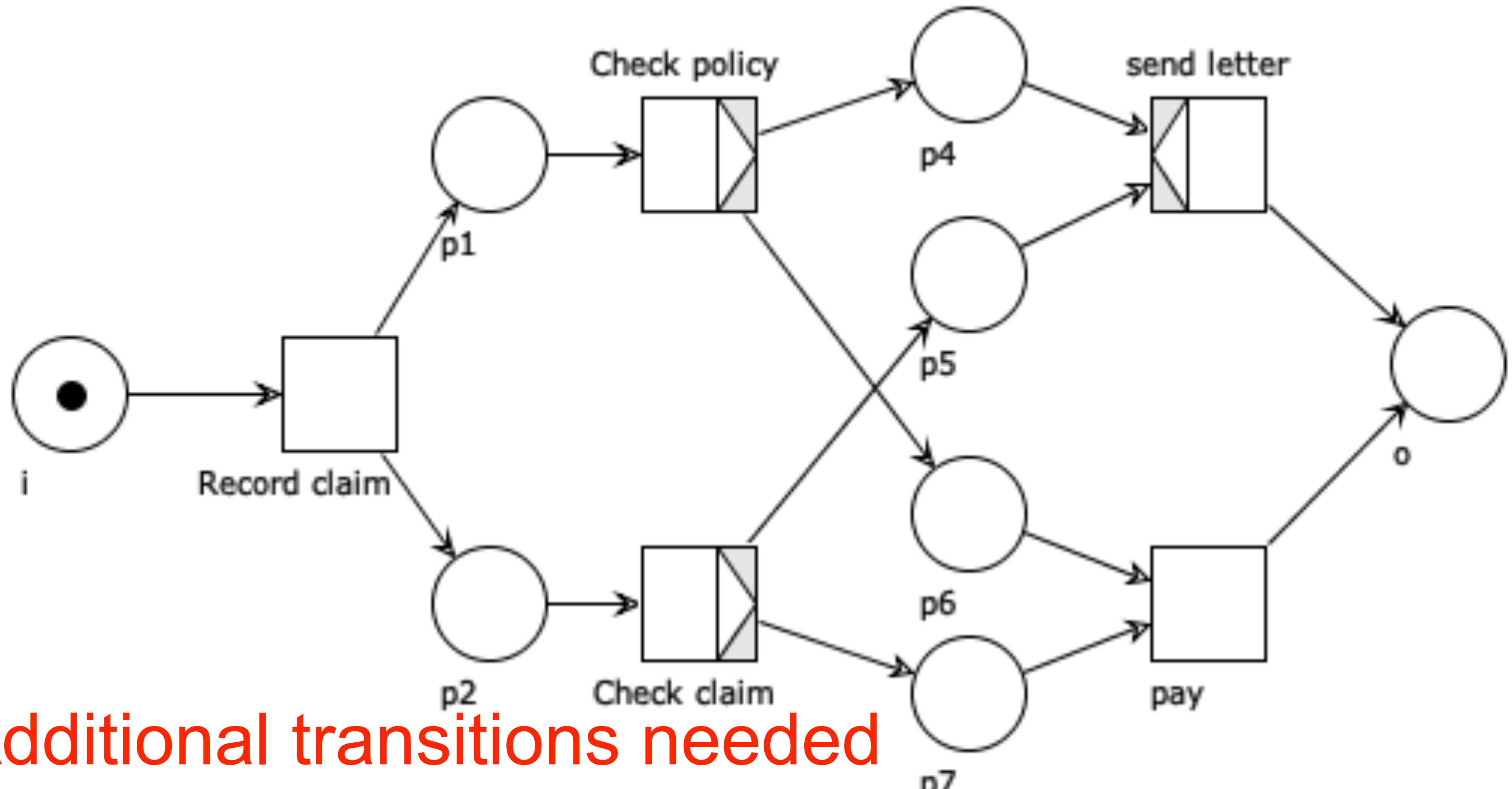




Some activities take place after case completion

Exercise

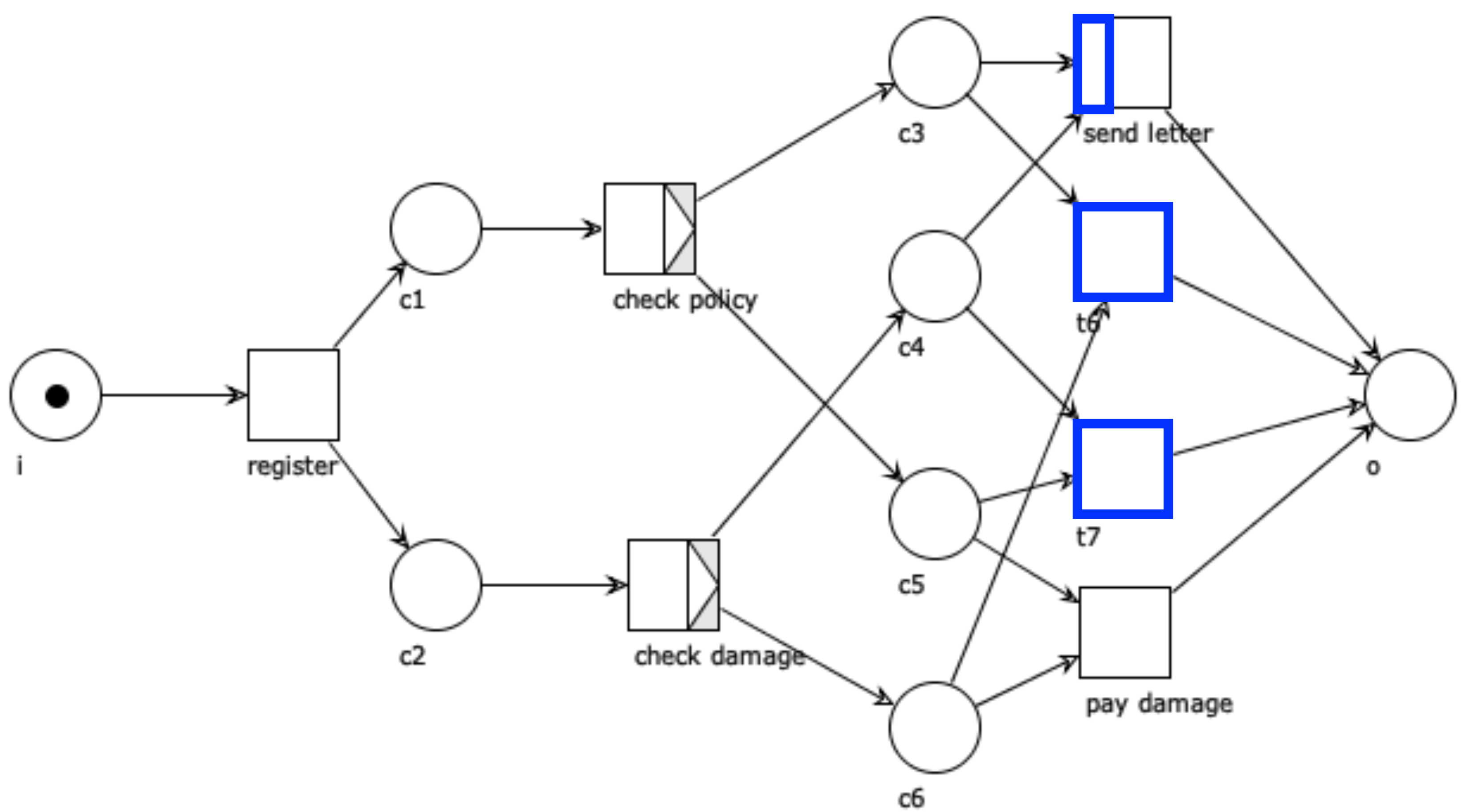
Which problem(s) in the workflow net below?
How would you redesign the business process?



Additional transitions needed
(dealing with all check failures)

Exercise

Which problem(s) in the workflow net below?
How would you redesign the business process?



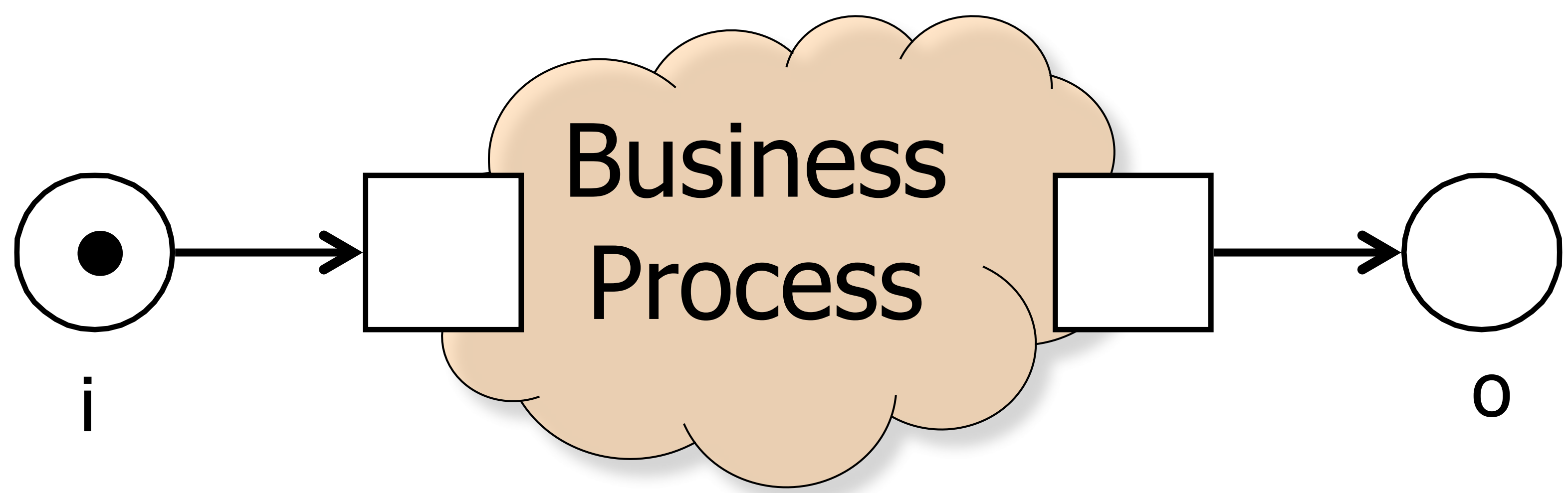
Soundness

Soundness of Business Processes

A process is called **sound** if

1. it contains no unnecessary tasks
2. every case, once started, can always be completed in full
3. no pending items are left upon case completion

Soundness of Business Processes



Soundness of Workflow nets

A workflow net is called **sound** if

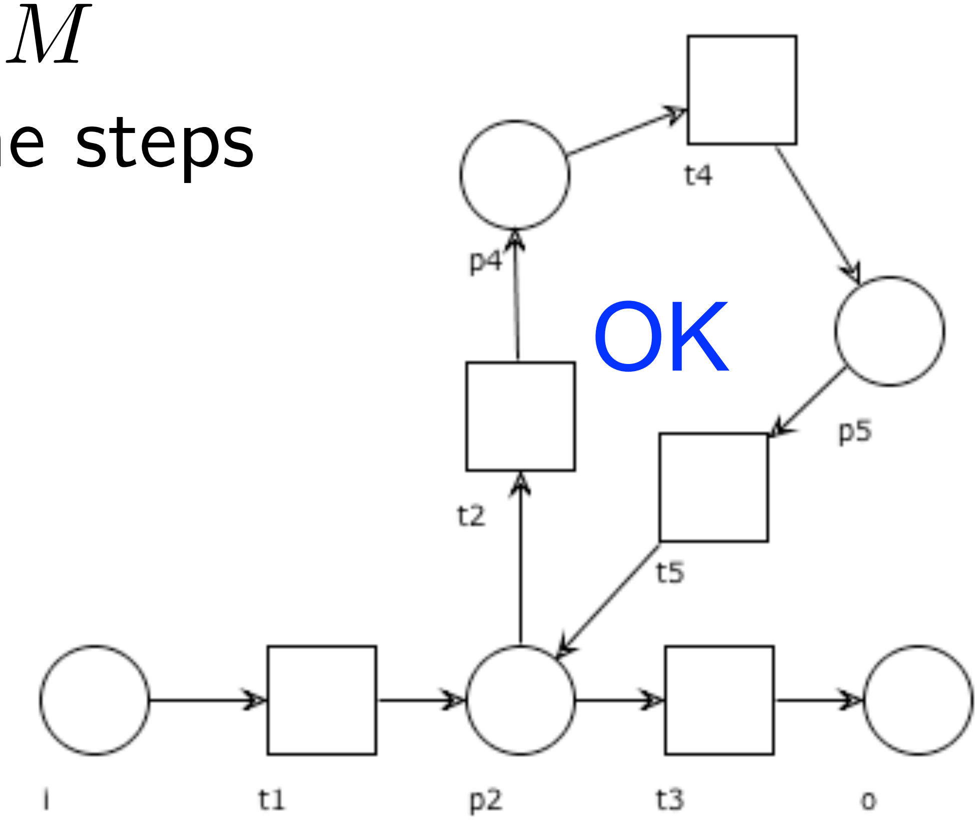
1. for each transition t,
   there is a marking M (reachable from i) that enables t
2. for each token put in place i,
   one token eventually appears in the place o
3. when a token is in place o, all other places are empty

Fairness assumption

Remark:
Condition 2 does not mean that iteration must be forbidden or bound

It says that from any reachable marking M
there must be possible to reach o in some steps

Fairness assumption:
A task cannot be postponed indefinitely



Soundness, Formally

A workflow net is called **sound** if

no dead task no transition is dead

$$\forall t \in T.\ \exists M \in [\, i \,\rangle.\ M \xrightarrow{t}$$

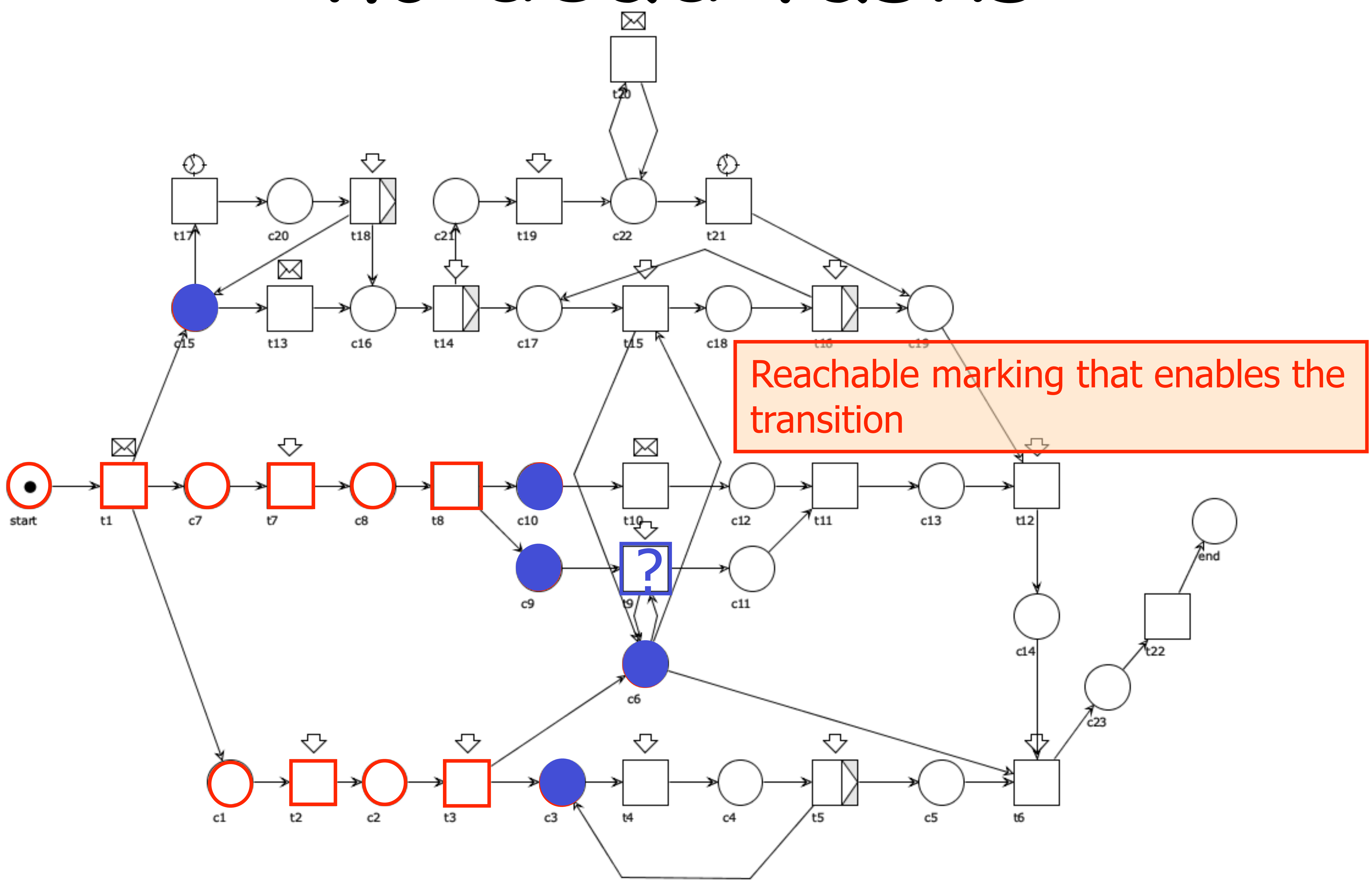
option to complete place o is eventually marked

$$\forall M \in [\, i \,\rangle.\ \exists M' \in [\, M \,\rangle.\ M'(o) \geq 1$$

proper completion when o is marked, no other token is left

$$\forall M \in [\, i \,\rangle.\ M(o) \geq 1 \Rightarrow M = o$$

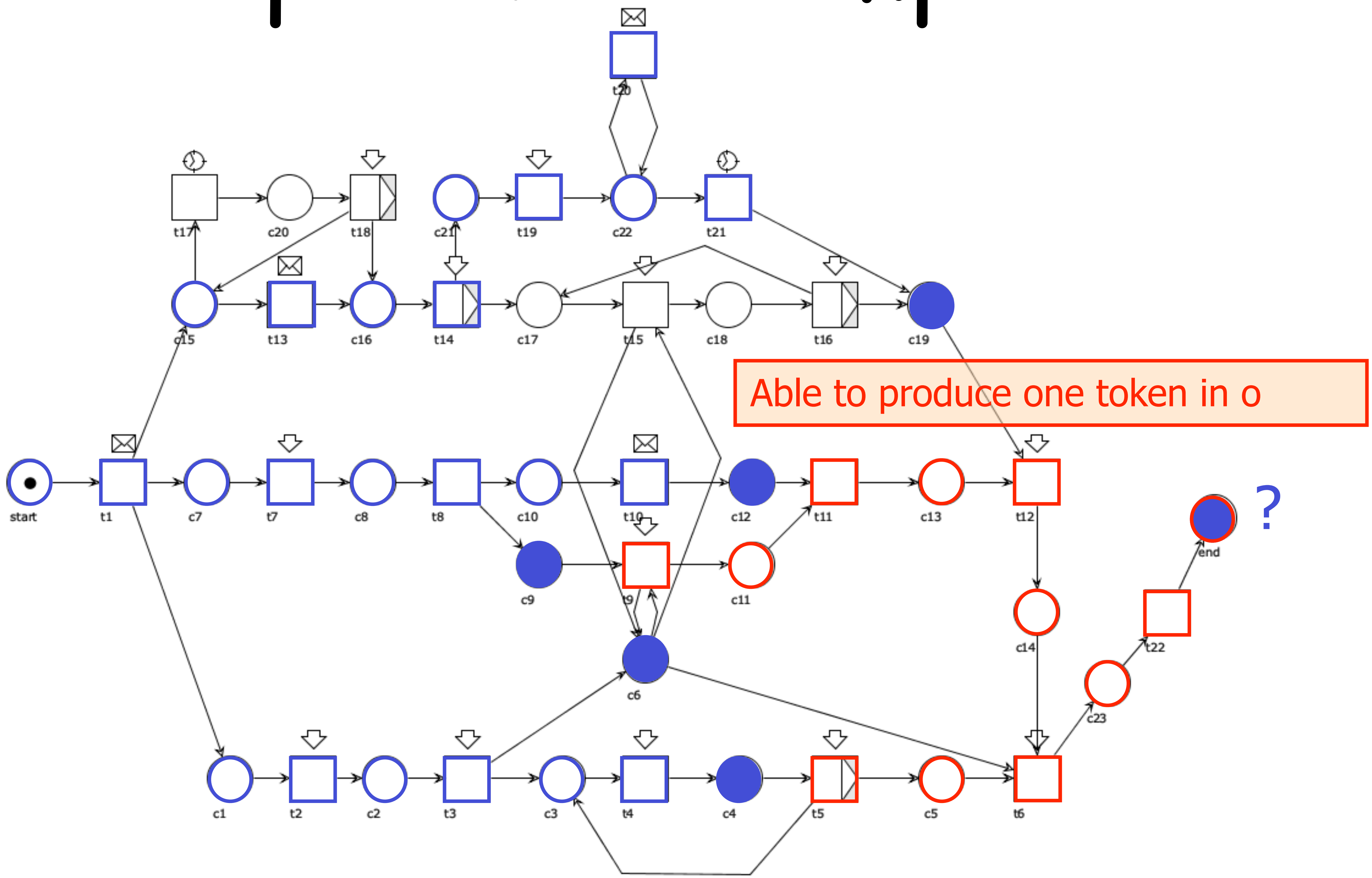
1: no dead tasks



1: no dead tasks

The check must be repeated for each task

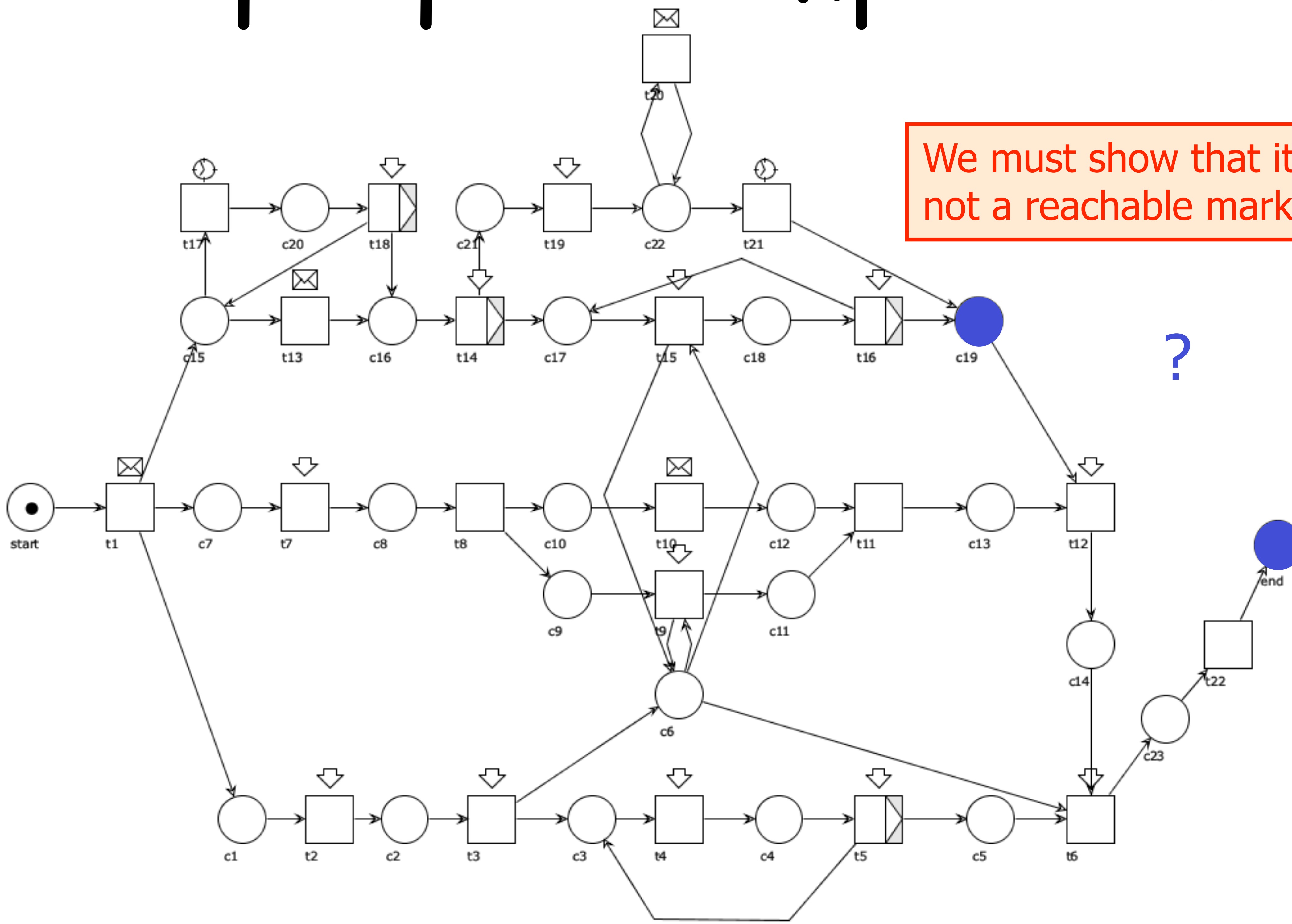
2: option to complete



2: option to complete

The check must be repeated for each reachable marking

3: proper completion



We must show that it is not a reachable marking

3: proper completion

The check must be repeated for each marking M such that $M > o$

Brute-force analysis

First, check if the Petri net is a workflow net
easy "structural" check

Second, check if it is sound (more difficult):
build the Reachability Graph
to check 1: for each transition t there must be an arc in the RG that is labelled with t
to check 2&3: the RG must have only one final state (sink), that consists of one token in o
and is reachable from any other state,
and no other marking has a token in o

Some Pragmatic Considerations

All checks can better be done automatically (computer aided)

but nevertheless RG construction...

1. can be computationally expensive for large nets (because of state explosion)
2. provides little support in repairing unsound processes
3. can be infinite (CG can be used, but it is not exact)

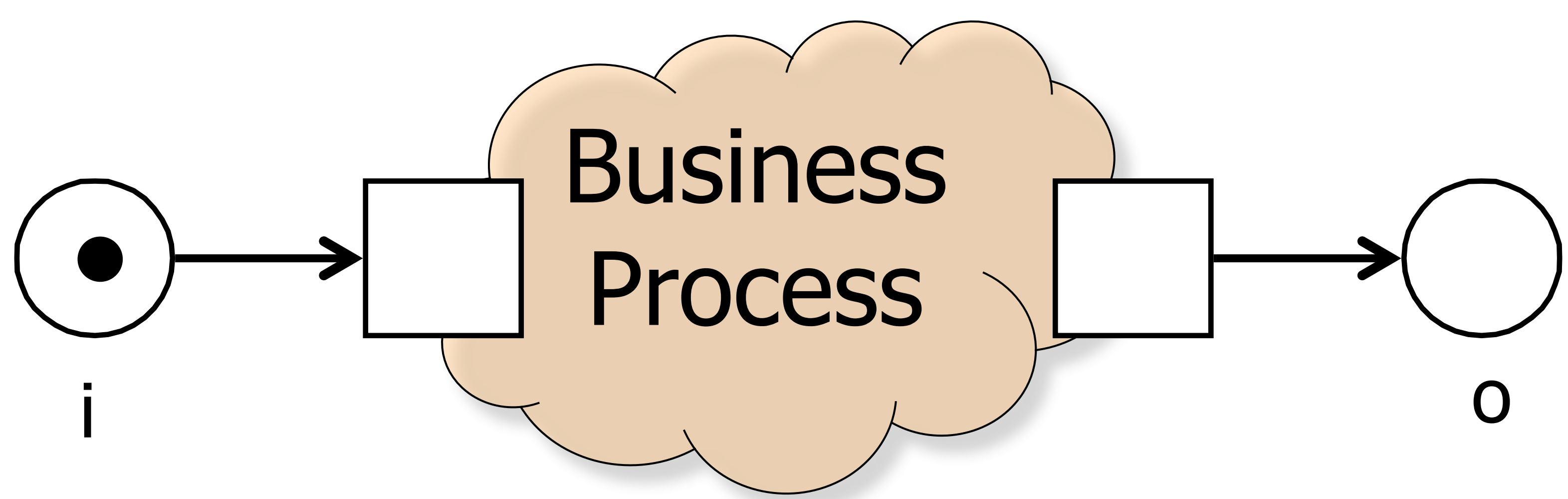
Advanced support

Translate soundness to other well-known properties that can be checked more efficiently:

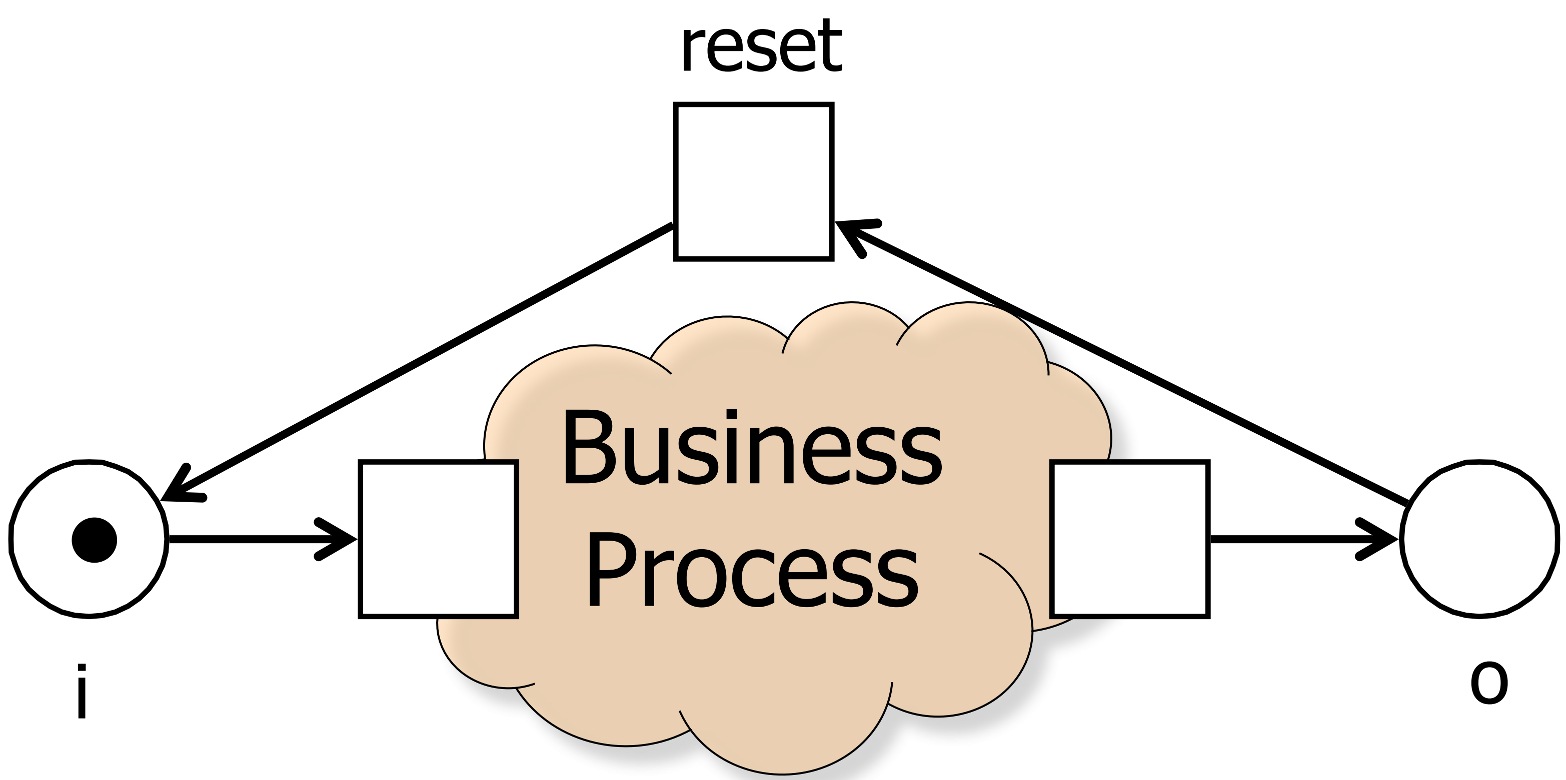
boundedness and liveness

N*

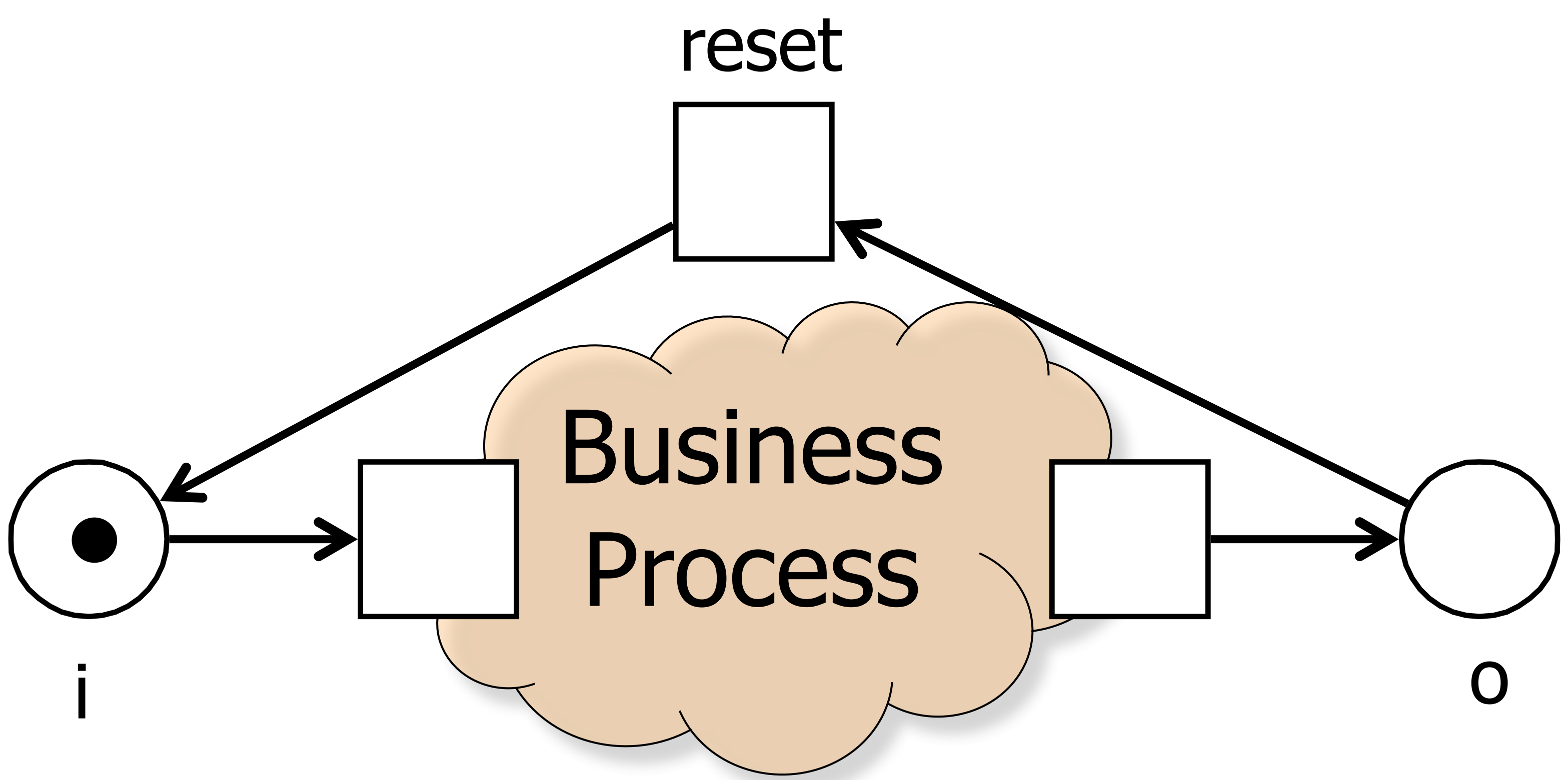
Play once



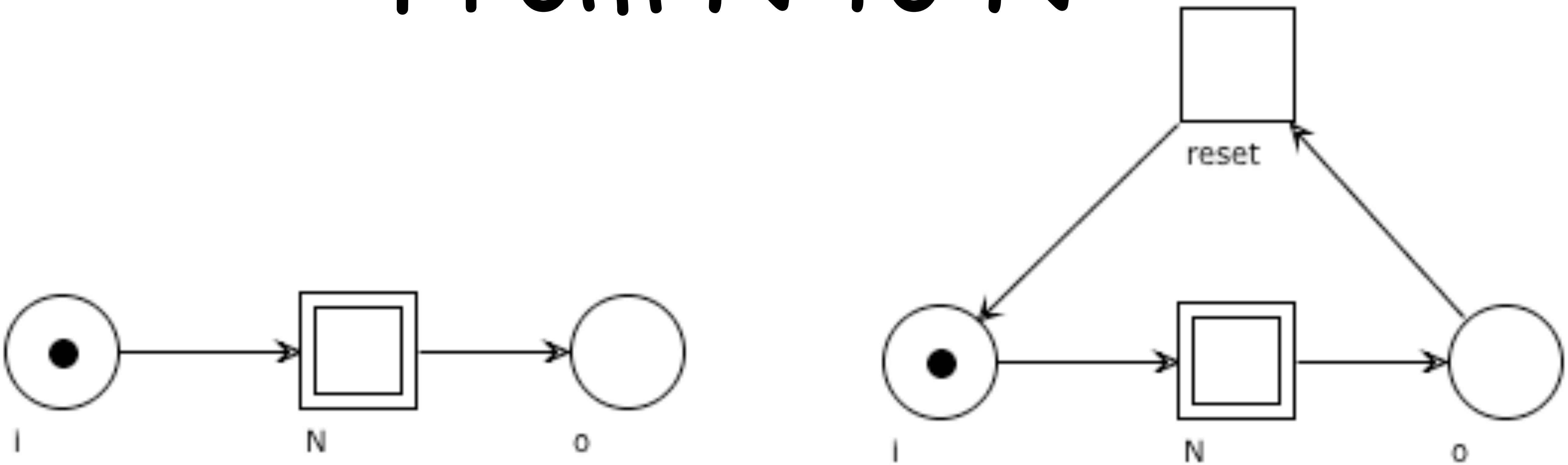
Play twice



Play any number of times



From N to N*



Let us denote by $N : i \to o$ a workflow net with entry place i and exit place o.

Let N^* be the net obtained by adding the "$reset$" transition to N $reset : o \to i$.

MAIN THEOREM

Theorem:
N is sound iff N^* is live and bounded

MAIN THEOREM

Theorem:

N is sound iff N^* is live and bounded

1 no dead tasks
2 option to complete
3 proper completion

$\Leftarrow^{1}$

$^{3}\Rightarrow$ at any reachable marking, every transition can fire in the future
and
$^{2}\Rightarrow$ for some k, every place will contain less than k tokens

Proof of MAIN THEOREM (1)

$(\Leftarrow)$

N^* **live and bounded implies** N **sound**:

Since N^* is **live**: for each $t \in T$ there is $M \in [\,i\,\rangle$. $M \stackrel{t}{\rightarrow}$

Take any $M \in [\,i\,\rangle$ enabling $reset : o \rightarrow i$, hence $M \supseteq o$

Let $M \stackrel{reset}{\longrightarrow} M'$. Then $M' \in [\,i\,\rangle$ and $M' \supseteq i$

Since N^* is bound, it must be $M' = i$ (and $M = o$)
Otherwise all places marked by $M' - i = M - o$ would be unbounded

Hence N^* just allows multiple runs of N:
"option to complete" and "proper completion" hold (see above)
"no dead task" holds because N^* is live

A technical lemma

Lemma:
If N is sound, M is reachable in N iff M is reachable in N^*

$\Rightarrow$) straightforward

$\Leftarrow$) Let $i \xrightarrow{\sigma} M$ in N^* for $\sigma = t_1 t_2 ... t_n$
We proceed by induction on the number r of instances of $reset$ in σ
If $r = 0$, then $reset$ does not occur in σ and M is reachable in N
If $r > 0$, let k be the least index such that $t_k = reset$
Let $\sigma = \sigma' t_k \sigma''$ with $\sigma' = t_1 t_2 ... t_{k-1}$ fireable in N
Since N is sound: $i \xrightarrow{\sigma'} o$ and $i \xrightarrow{\sigma''} M$
Since σ'' contains $r-1$ instances of $reset$:
by inductive hypothesis M is reachable in N

Proof of MAIN THEOREM (2)

$(\Rightarrow)$

N **sound implies** N^* **bounded** :
We proceed by contradiction, assuming N^* is unbounded

Since N^* is unbounded:
$\exists M, M'$ such that $i \rightarrow^* M \rightarrow^* M'$ with $M \subset M'$
Let $L = M' - M \neq \emptyset$

Since N is sound:
$\exists \sigma \in T^*$ such that $M \stackrel{\sigma}{\rightarrow} o$

By the monotonicity Lemma: $M' \stackrel{\sigma}{\rightarrow} o + L$ and thus $o + L \in [\, i \,\rangle$
Which is absurd, because N is sound

$(\Leftarrow)$

Proof of MAIN THEOREM (3)

N **sound implies** N^* **live**:

Take any transition t and let M be a marking reachable in N^*

By the technical lemma, M is reachable in N

Since N is sound: $\exists \sigma \in T^*$ with $M \xrightarrow{\sigma} o$

Since N is sound: $\exists \sigma' \in T^*$ with $i \xrightarrow{\sigma'} M'$ and $M' \xrightarrow{t}$

Let $\sigma'' = \sigma \, reset \, \sigma'$, then:

$M \xrightarrow{\sigma''} M'$ in N^* and $M' \xrightarrow{t}$

A theorem on strong connectedness (whose proof we omit)

Paths and circuits

$$(P, T, F)$$

A **path** is a non-empty sequence of nodes $x_1 x_2 ... x_k$ such that

$$(x_i, x_{i+1}) \in F \quad \text{for every } 1 \leq i \leq k$$

and we say it leads from x_1 to x_k

A path $x_1 x_2 ... x_k$ is called a **circuit** if

all its nodes are distinct and $(x_k, x_1) \in F$

since there is no node x with $(x, x) \in F$, any circuit has at least two nodes

Paths and circuits

$$(P, T, F)$$

An **undirected path** is a non-empty sequence of nodes $x_1 x_2 ... x_k$ s.t.

$$(x_i, x_{i+1}) \in (F \cup F^{-1}) \quad \text{for every } 1 \leq i \leq k$$

(denotes the inverse of a binary relation)

$$F^{-1} = \{\, (y, x) \mid (x, y) \in F \,\}$$

(a path where we disregard the orientation of arcs)

Connectedness

A net (P, T, F) is **weakly connected** if

there is an undirected path between any two distinct nodes

A net (P, T, F) is **strongly connected** if

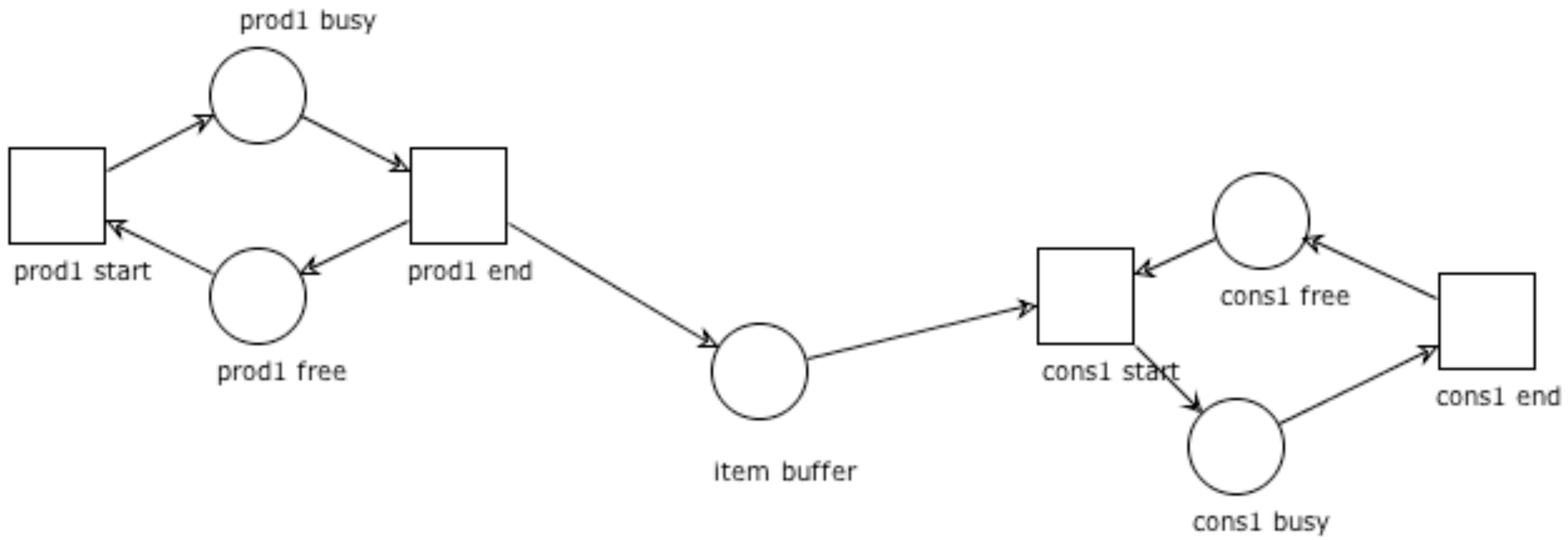
there is a path between any two distinct nodes

Connectedness, again

A net (P,T,F) is **weakly connected**
iff
it cannot be splitted in separated components

A weakly connected net is **strongly connected**
iff
for every arc (x,y) there is a path from y to x

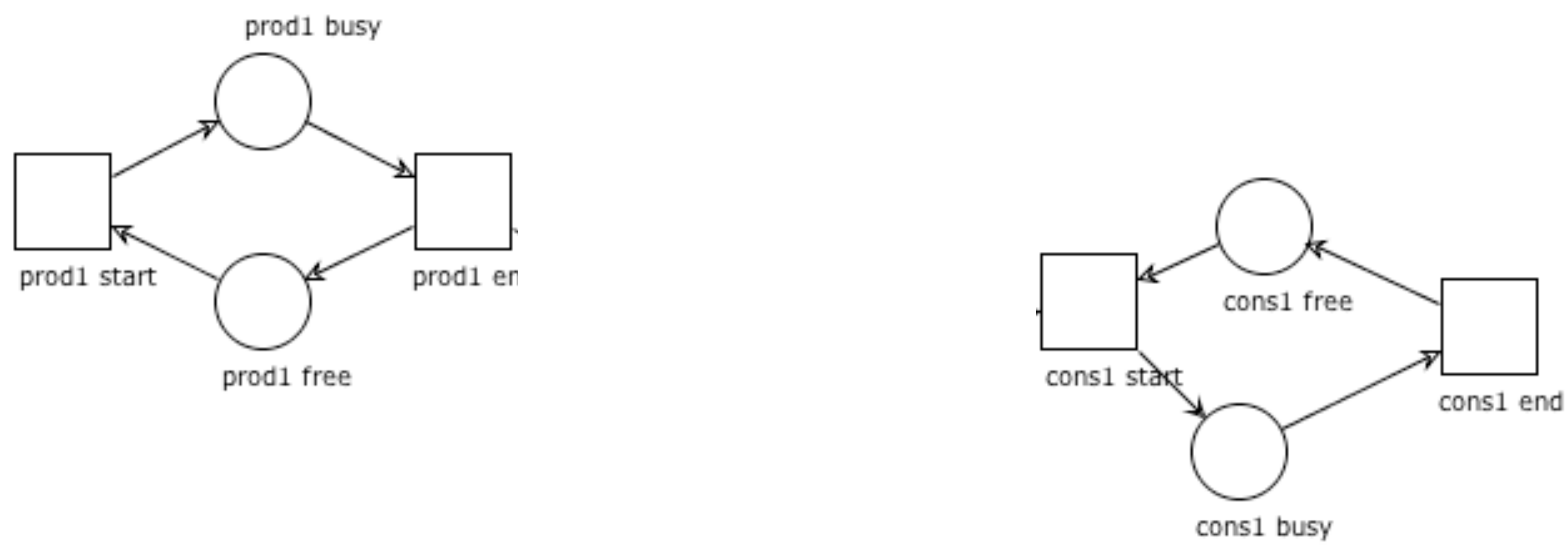
Examples



weakly connected

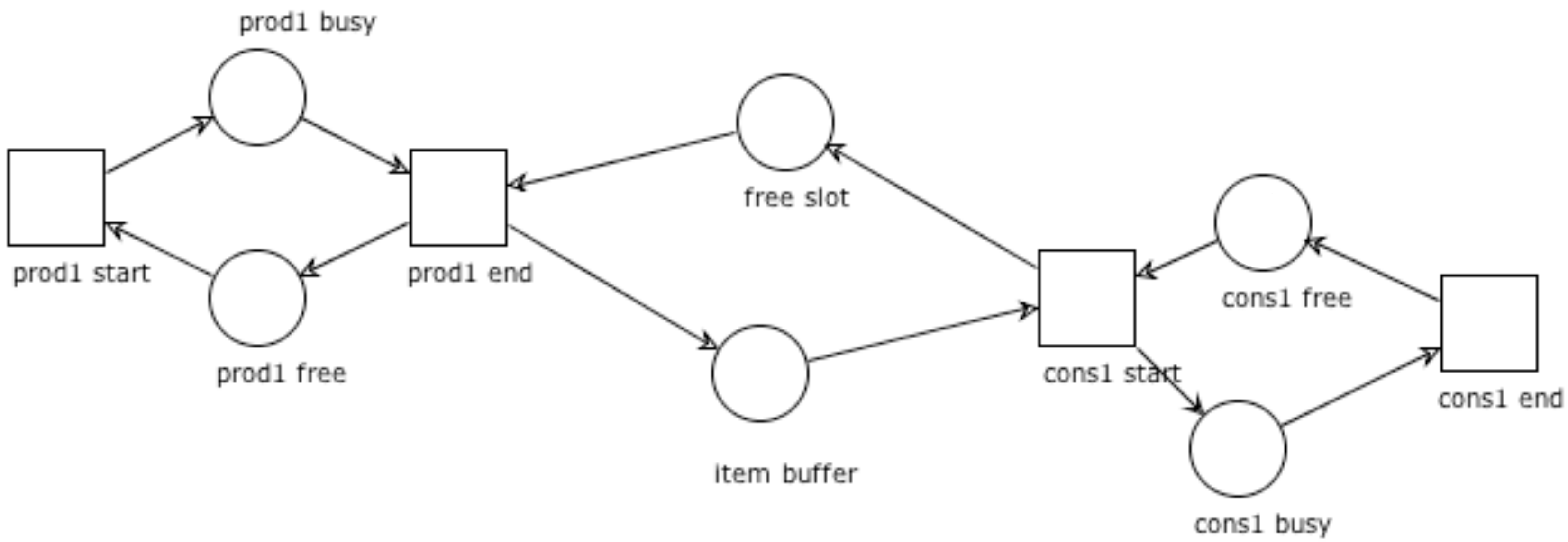
not strongly connected

Examples



not weakly connected
not strongly connected

Examples



weakly connected
strongly connected

Question time

?

IMPOSSIBLE !!!

not weakly connected
strongly connected

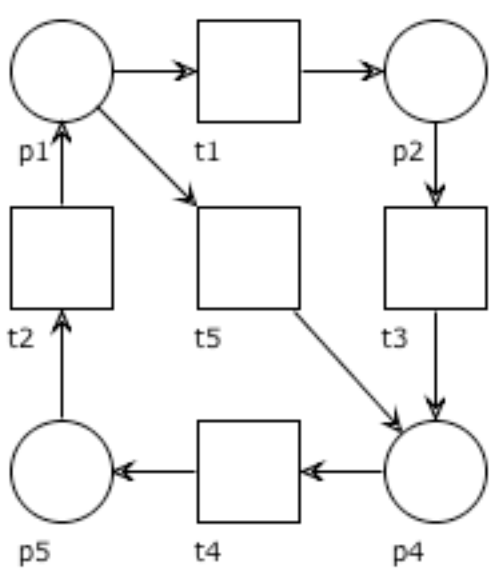
A note

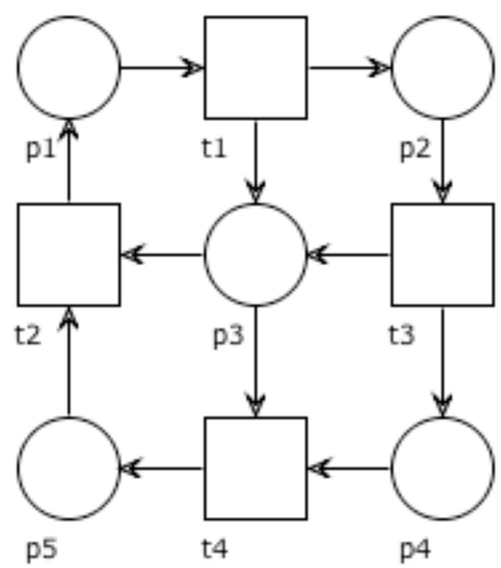
In the following we will consider (implicitly) weakly connected nets only

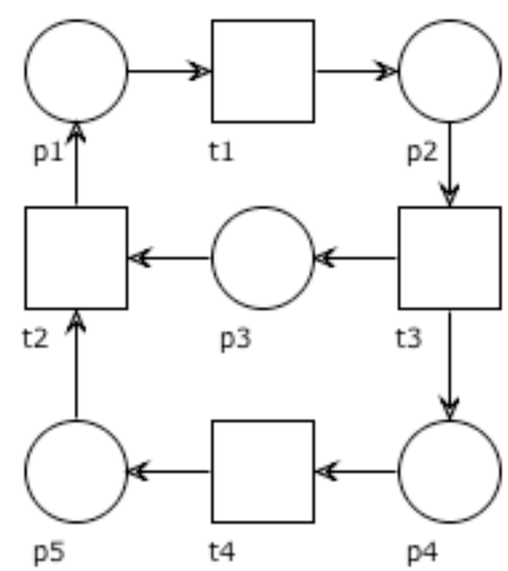
(if they are not, then we can study each of their subsystems separately)

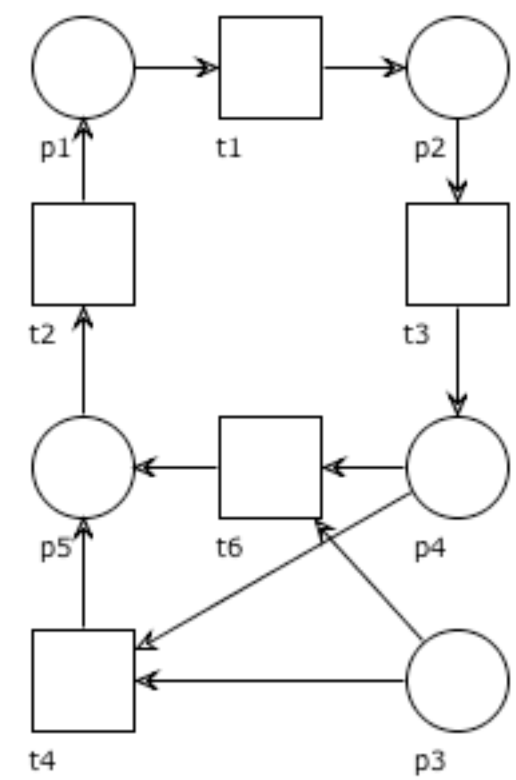
Question time

Is the net strongly connected?





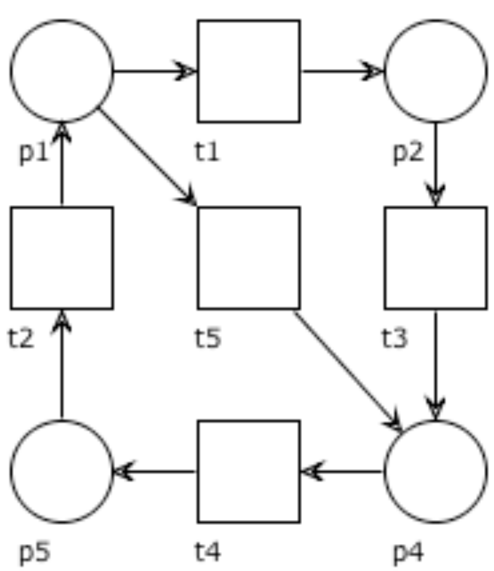


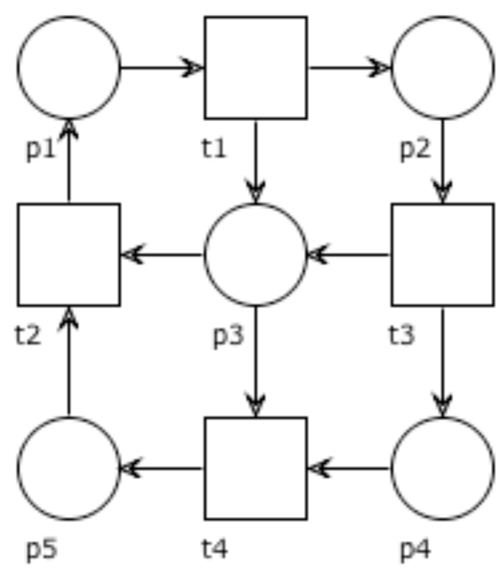


Question time

Is the net strongly connected?

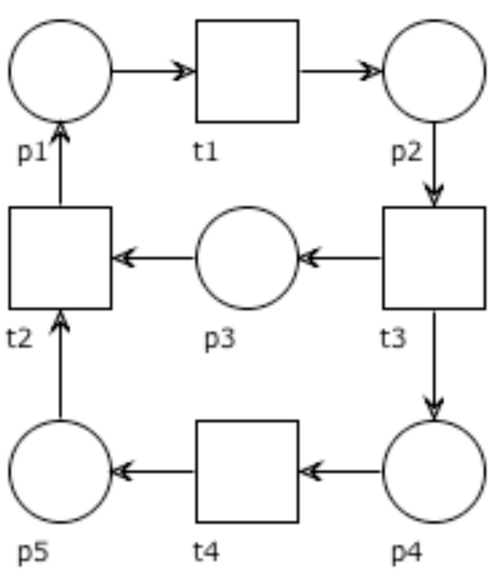
YES

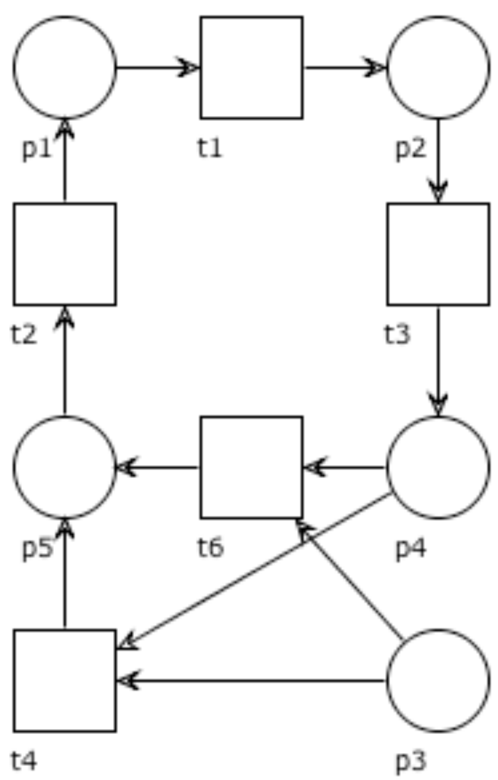




YES

YES





NO

Strong connectedness theorem

Theorem: If a weakly connected system is live and bounded then it is strongly connected

Consequences

If a (weakly-connected) net is not strongly connected

then

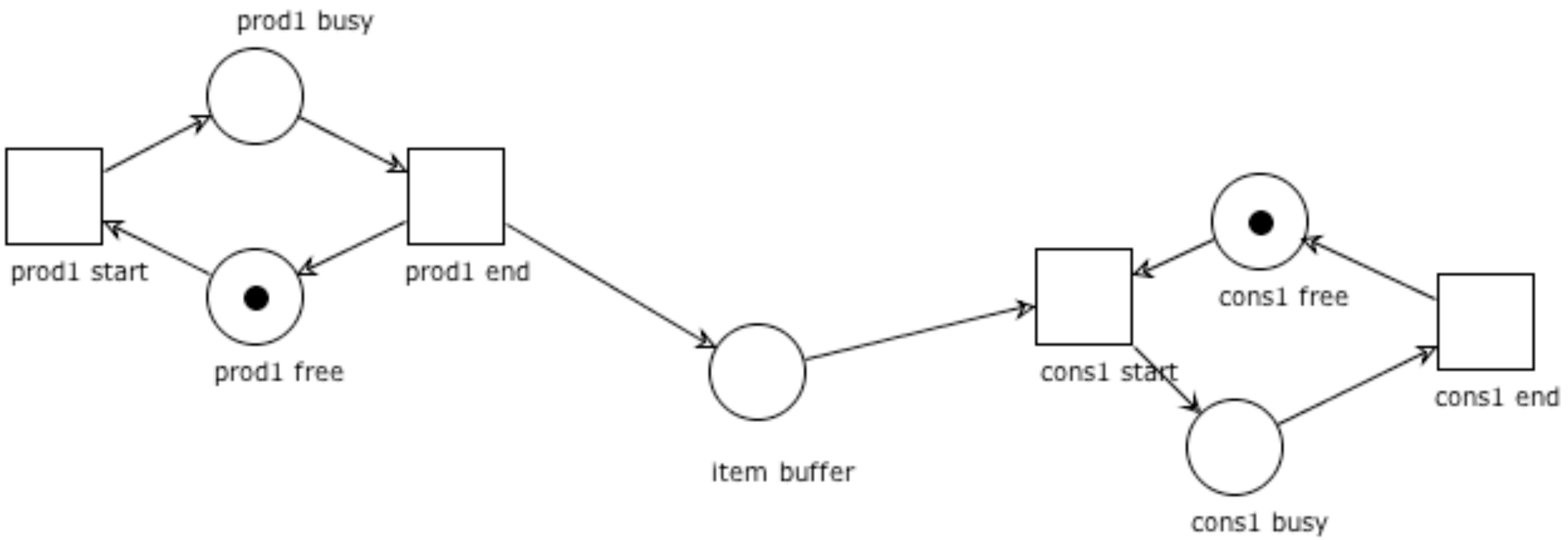
It is not “live and bounded”

If it is live, it is not bounded

If it is bounded, it is not live

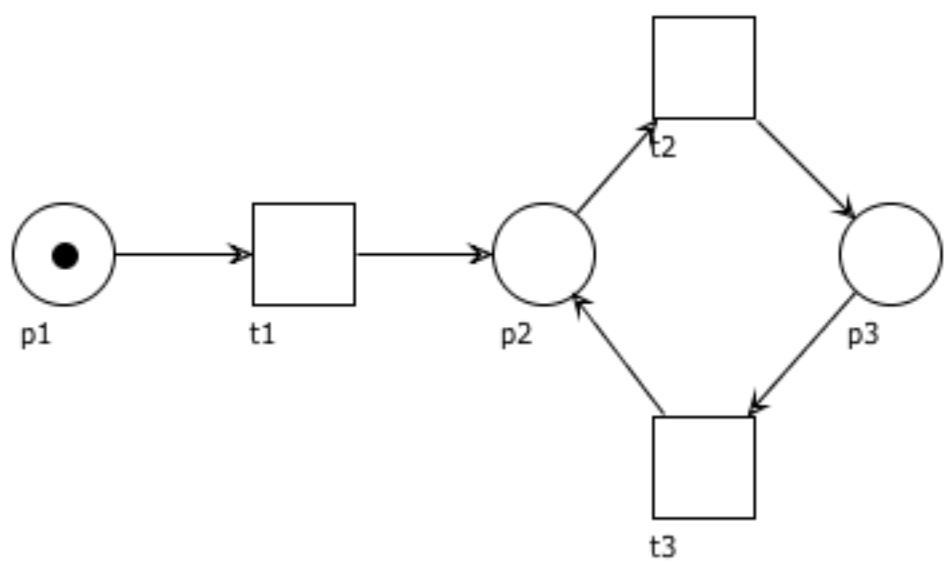
Example

It is now immediate to see that this system (weakly connected, not strongly connected) cannot be live and bounded (it is live but not bounded)



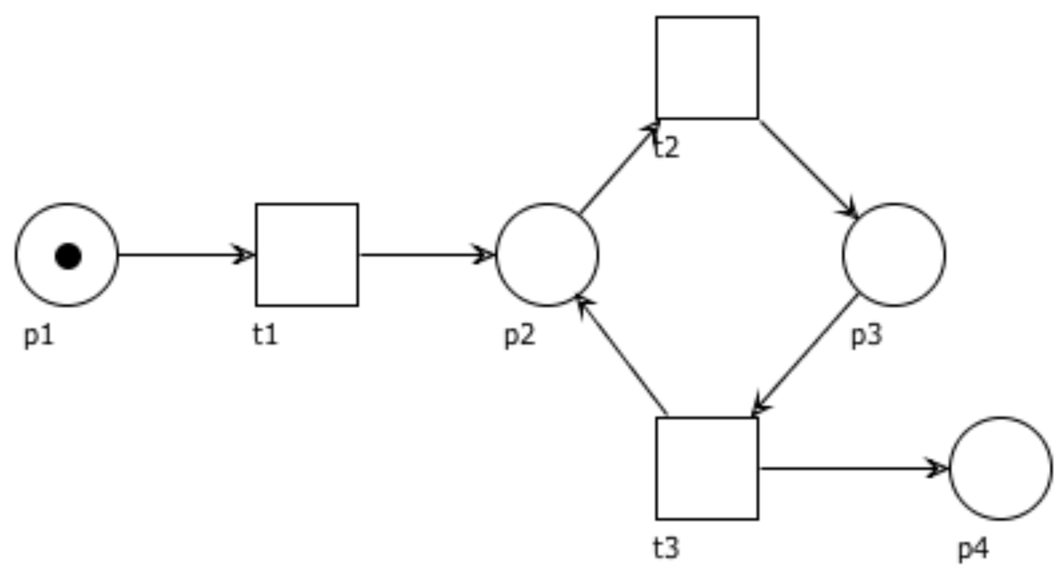
Example

It is now immediate to see that this system (weakly connected, not strongly connected) cannot be live and bounded
(it is bounded but not live)



Example

It is now immediate to see that this system (weakly connected, not strongly connected) cannot be live and bounded (it is neither bounded nor live)



Strong connectedness of N*

Proposition:
N^* is strongly connected.

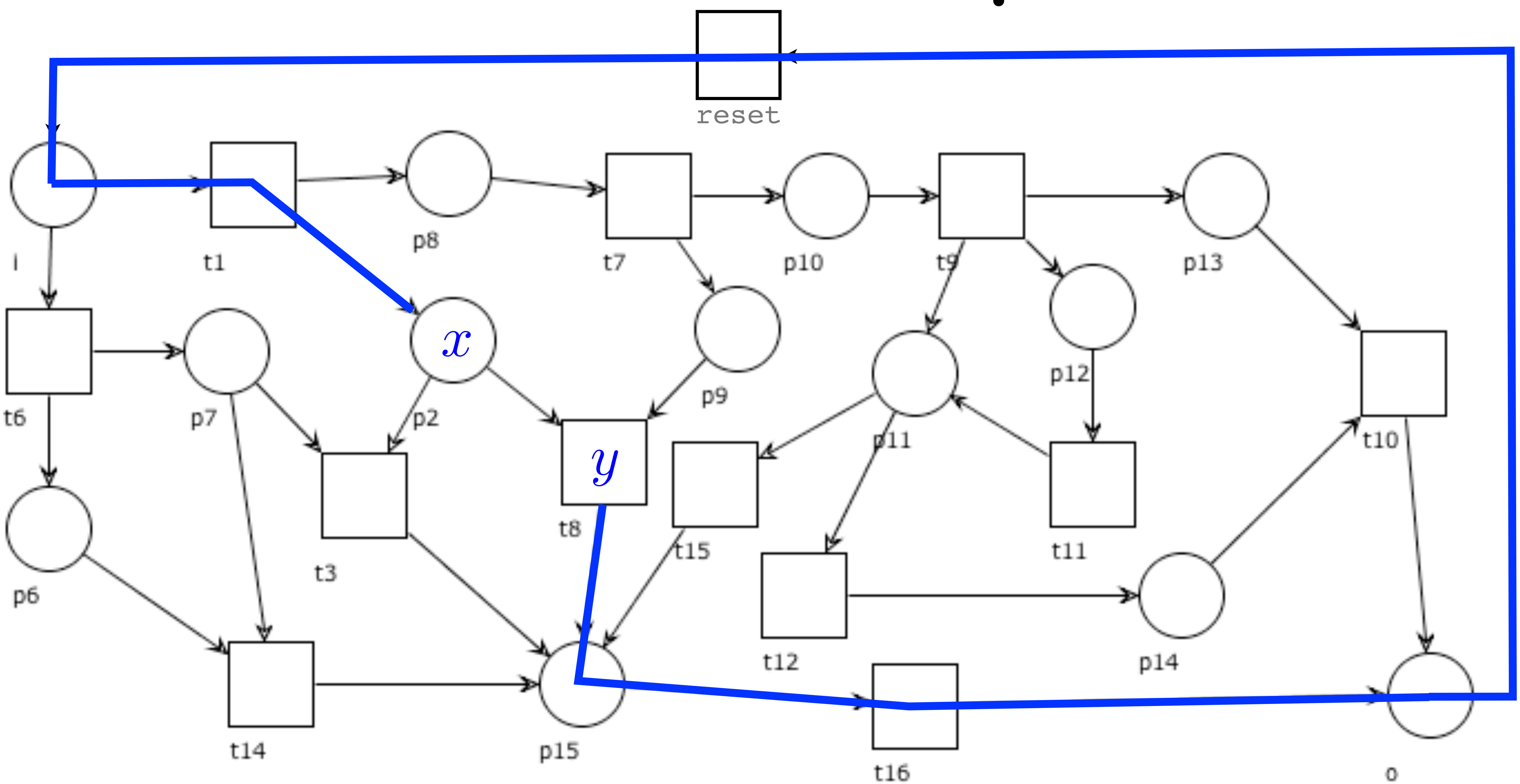
Take two nodes of $(x, y) \in F_{N^*}$,
we want to build a path from y to x

If $x, y \neq reset$, then
y lies on a path $i \to^* \boxed{y \to^* o}$, because N is a workflow net,
x lies on a path $\boxed{i \to^* x} \to^* o$, because N is a workflow net,
we combine the paths $y \to^* o \to reset \to i \to^* x$

If $x = reset, y = i$, then
take any path $i \to^* o$,
we build the path $i \to^* o \to reset$

If $x = o, y = reset$, then
take any path $i \to^* o$,
we build the path $reset \to i \to^* o$

Strong connectedness of N*: example

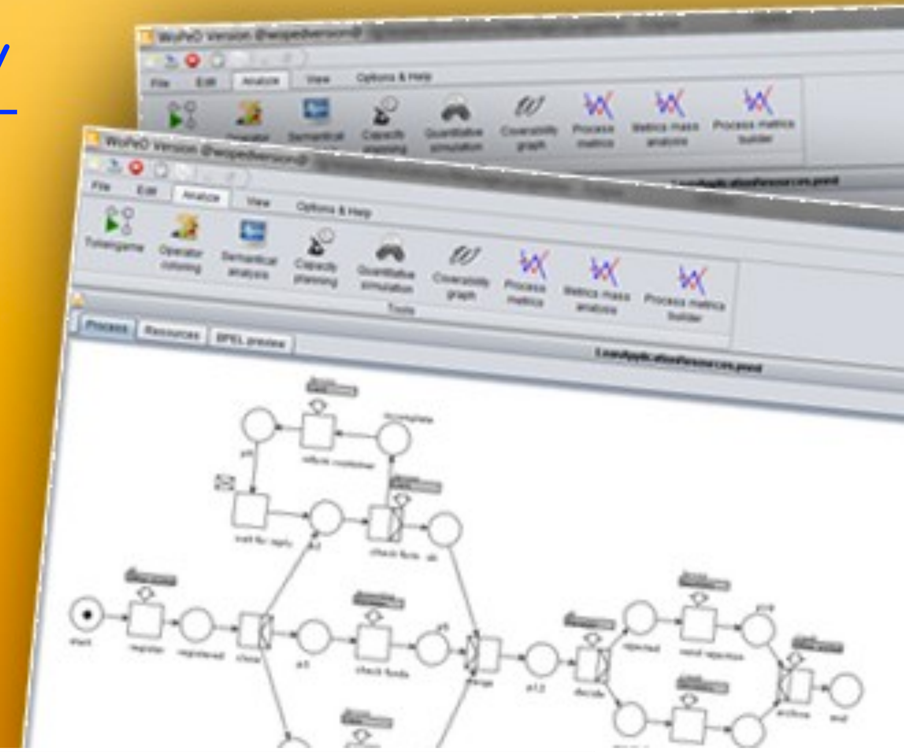


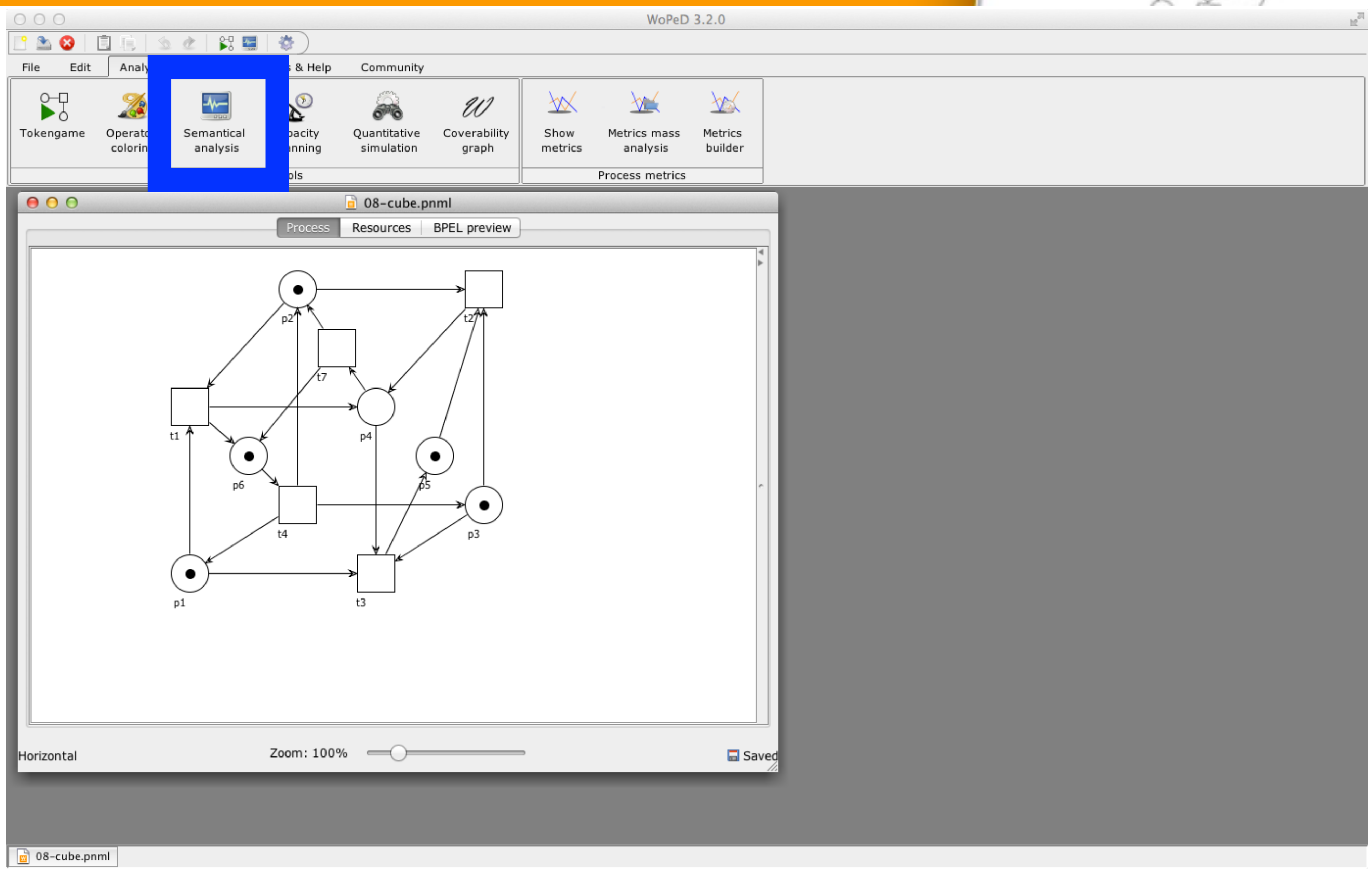
http://woped.dhbw-karlsruhe.de/woped/

WoPeD

Workflow Petri Net Designer

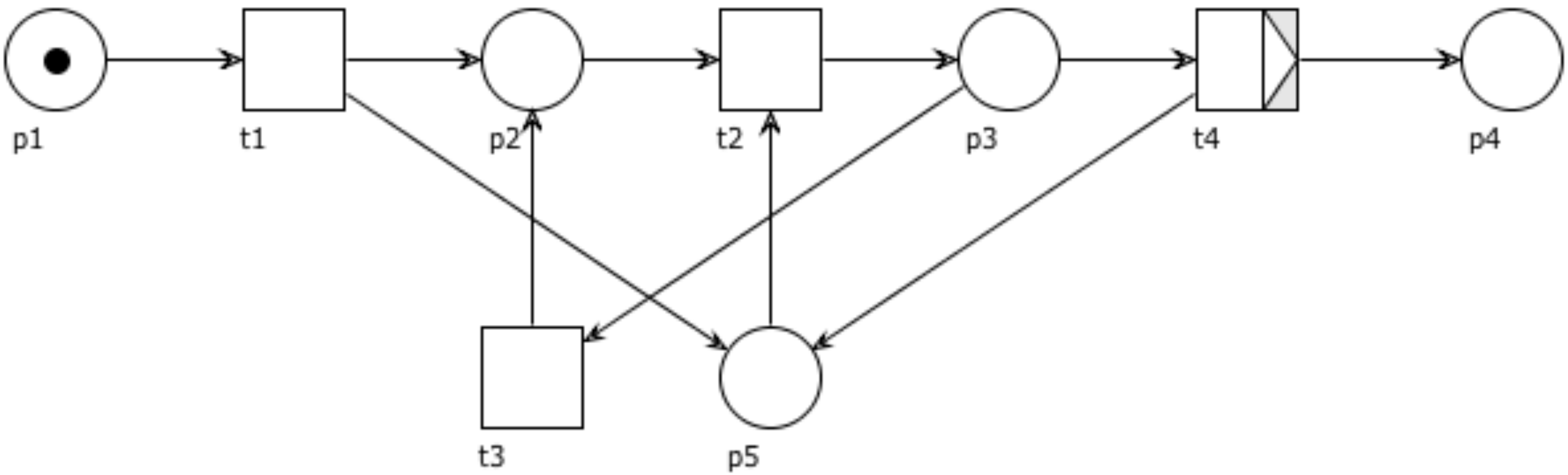
Download WoPeD at sourceforge!

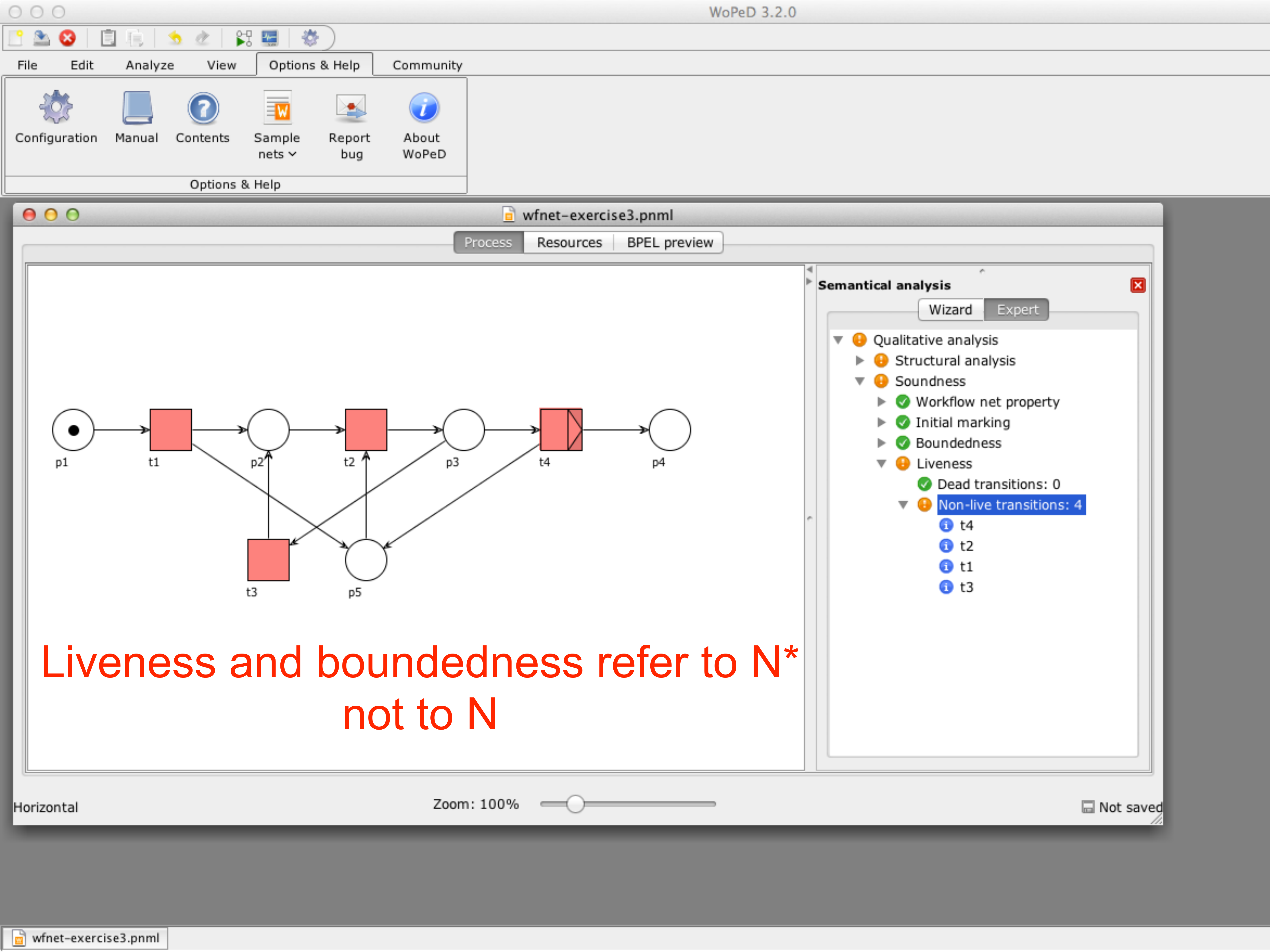


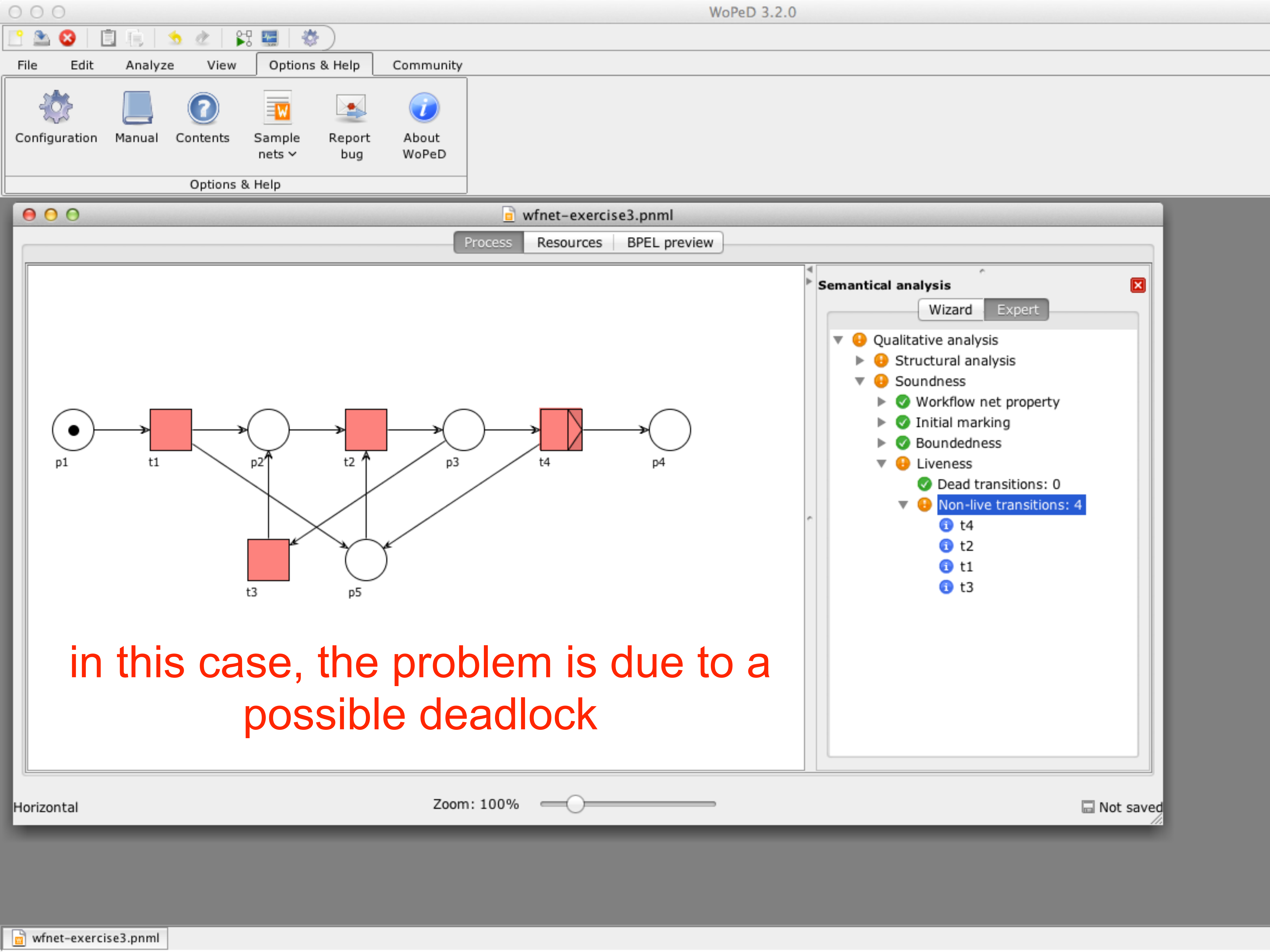


Exercise

Use some tools to check if the net below is a sound workflow net or not



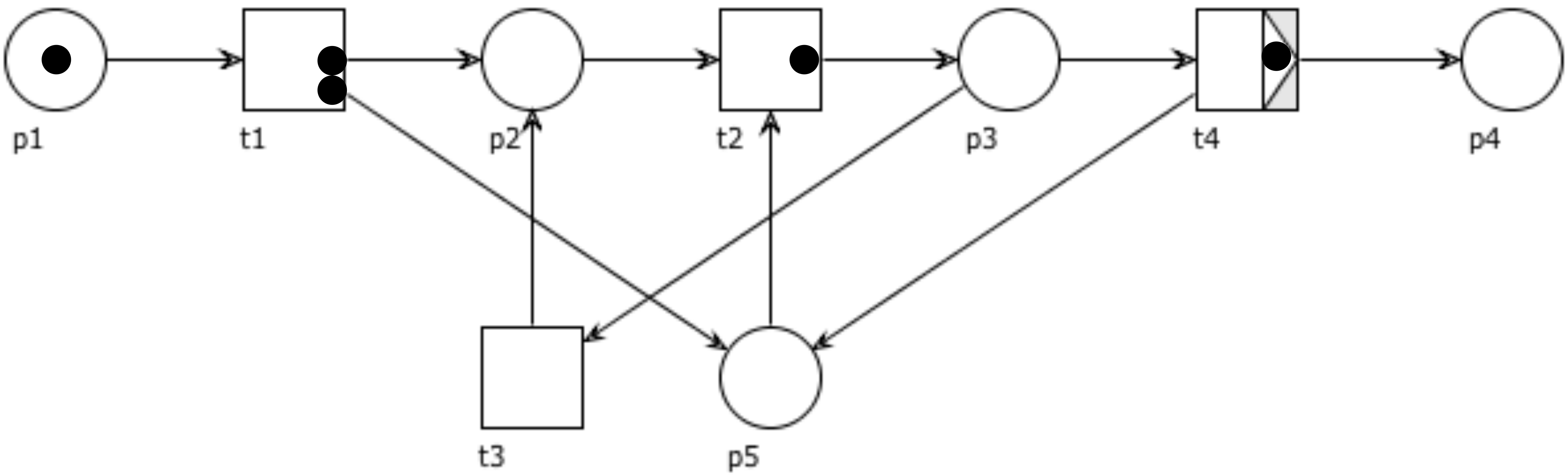

WoPeD 3.2.0
File
Edit
Analyze
View
Options & Help
Community
Configuration
Manual
Contents
Sample nets
Report bug
About WoPeD
Options & Help
wfnet-exercise3.pnml
Process
Resources
BPEL preview
p1
t1
p2
t2
p3
t4
p4
t3
p5
Liveness and boundedness refer to N* not to N
Semantical analysis
Wizard
Expert
Qualitative analysis
Structural analysis
Soundness
Workflow net property
Initial marking
Boundedness
Liveness
Dead transitions: 0
Non-live transitions: 4
t4
t2
t1
t3
Horizontal
Zoom: 100%
Not saved
wfnet-exercise3.pnml



in this case, the problem is due to a possible deadlock

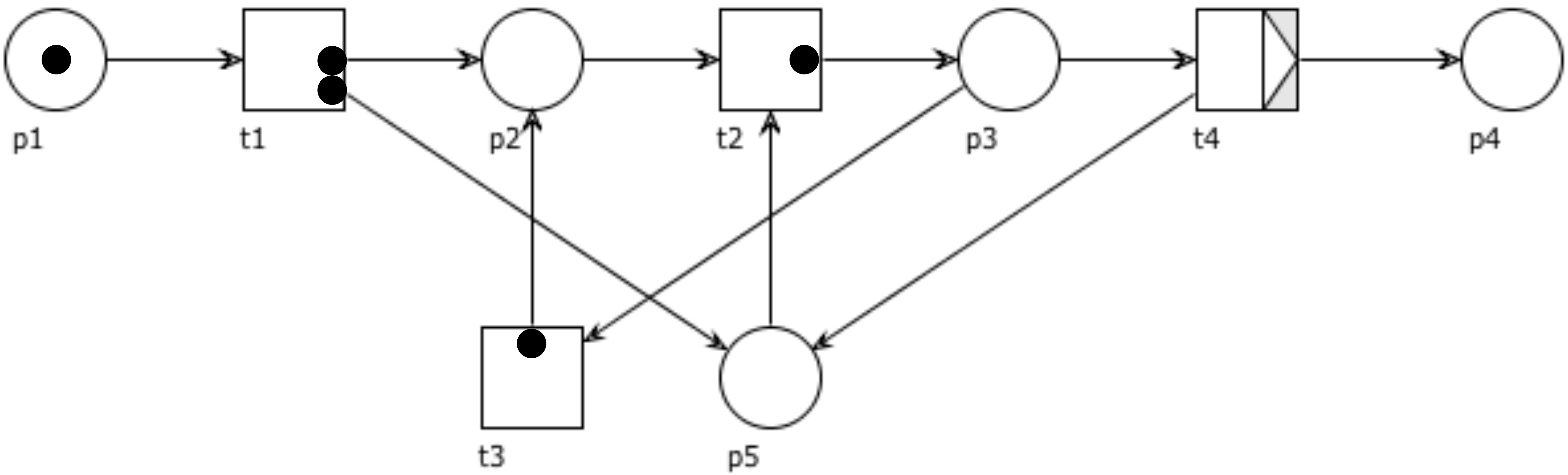
Exercise

Use some tools to check if the net below is a sound workflow net or not



Exercise

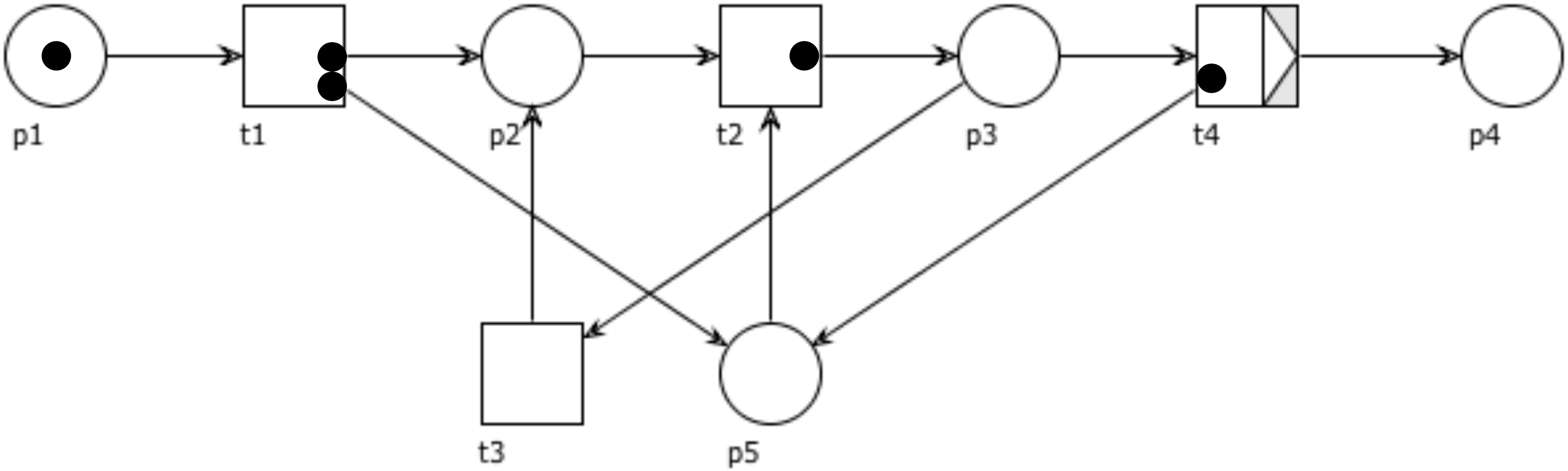
Use some tools to check if the net below is a sound workflow net or not



Deadlock

Exercise

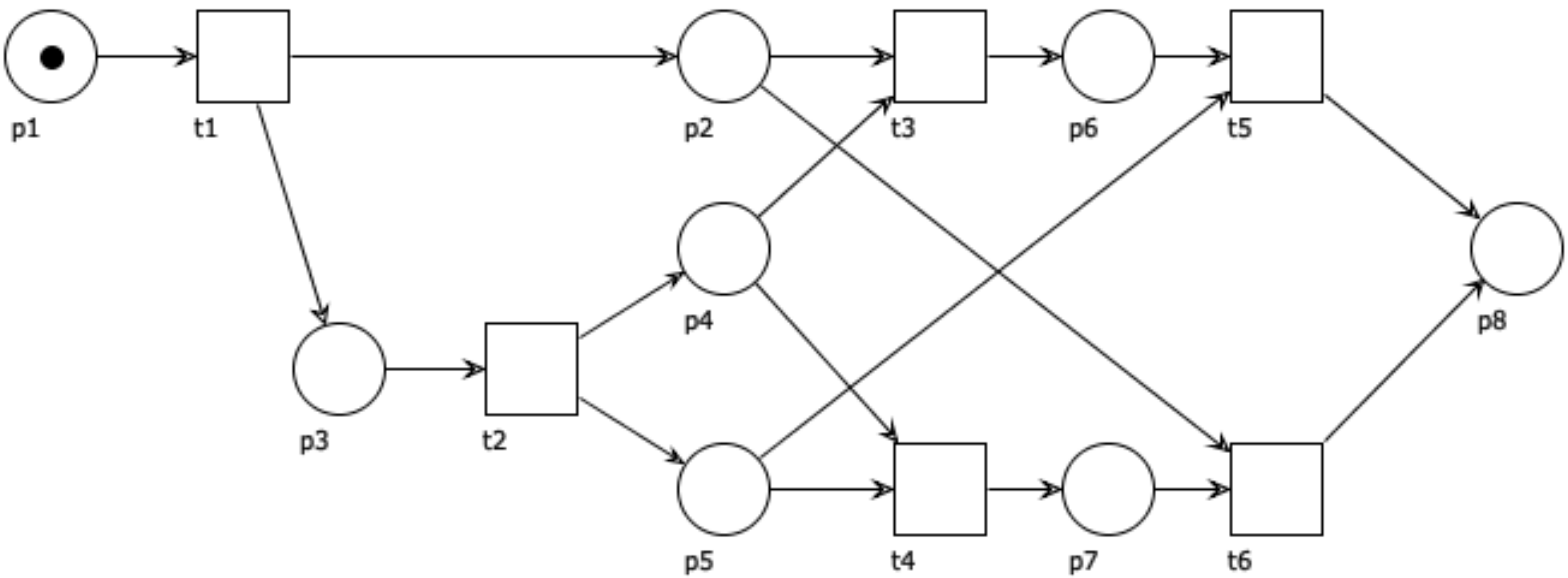
Use some tools to check if the net below is a sound workflow net or not

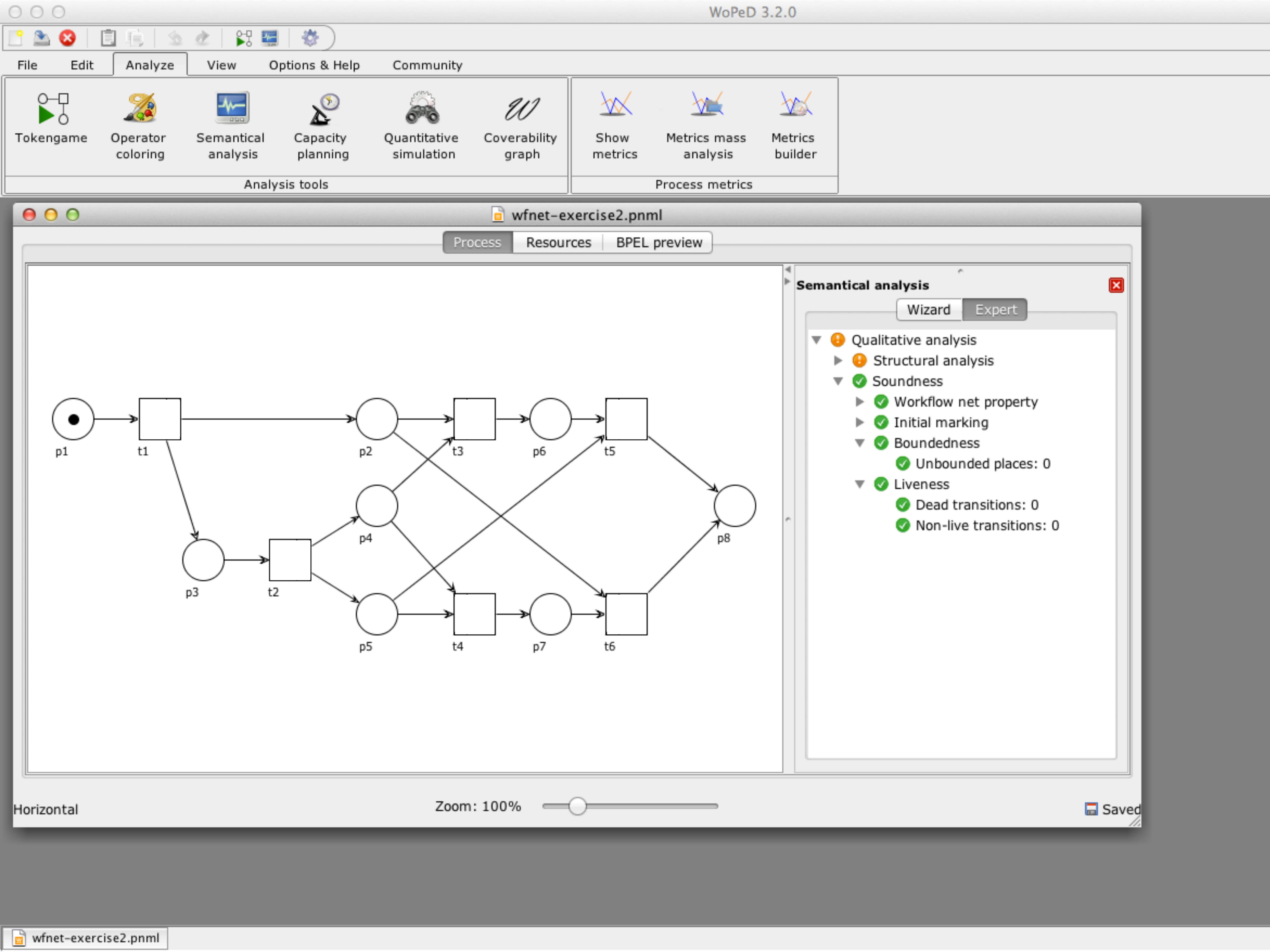


Deadlock

Exercise

Use some tools to check if the net below is a sound workflow net or not

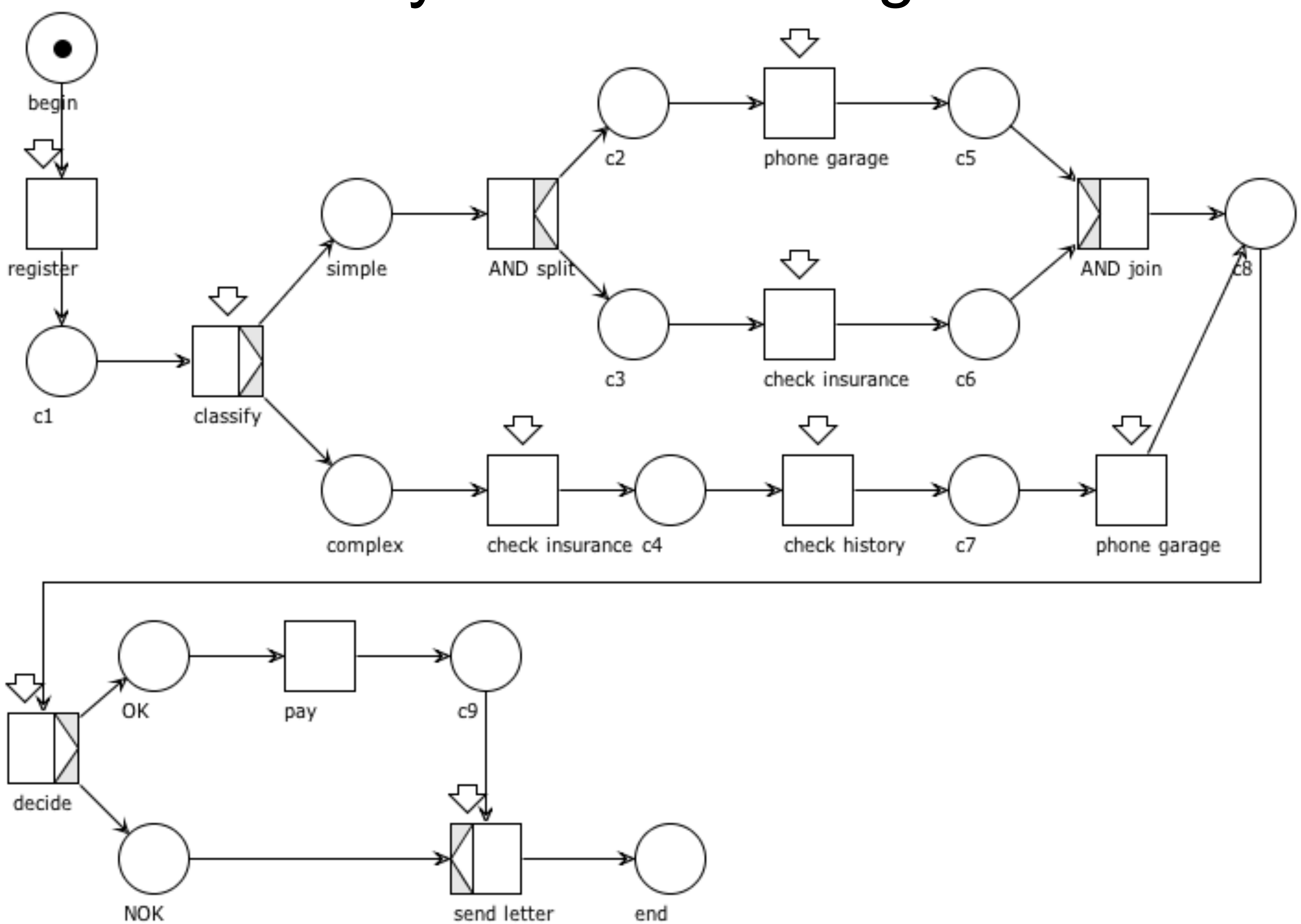


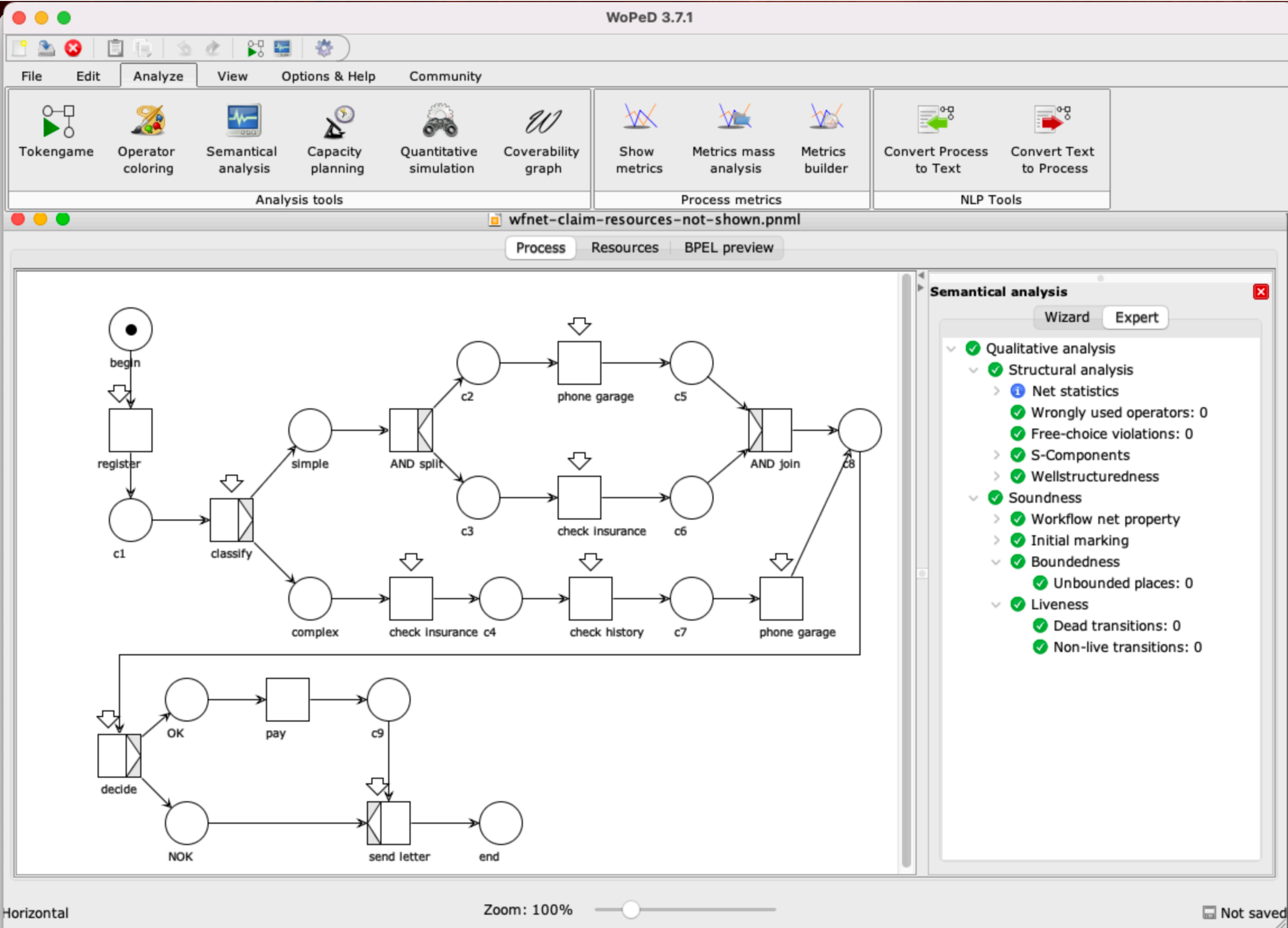


WoPeD 3.2.0
File
Edit
Analyze
View
Options & Help
Community
Tokengame
Operator coloring
Semantical analysis
Capacity planning
Quantitative simulation
Coverability graph
Analysis tools
Show metrics
Metrics mass analysis
Metrics builder
Process metrics
wfnet-exercise2.pnml
Process
Resources
BPEL preview
p1
t1
p2
t3
p6
t5
p4
p3
t2
p8
p5
t4
p7
t6
Semantical analysis
Wizard
Expert
Qualitative analysis
Structural analysis
Soundness
Workflow net property
Initial marking
Boundedness
Unbounded places: 0
Liveness
Dead transitions: 0
Non-live transitions: 0
Horizontal
Zoom: 100%
Saved
wfnet-exercise2.pnml

Exercise

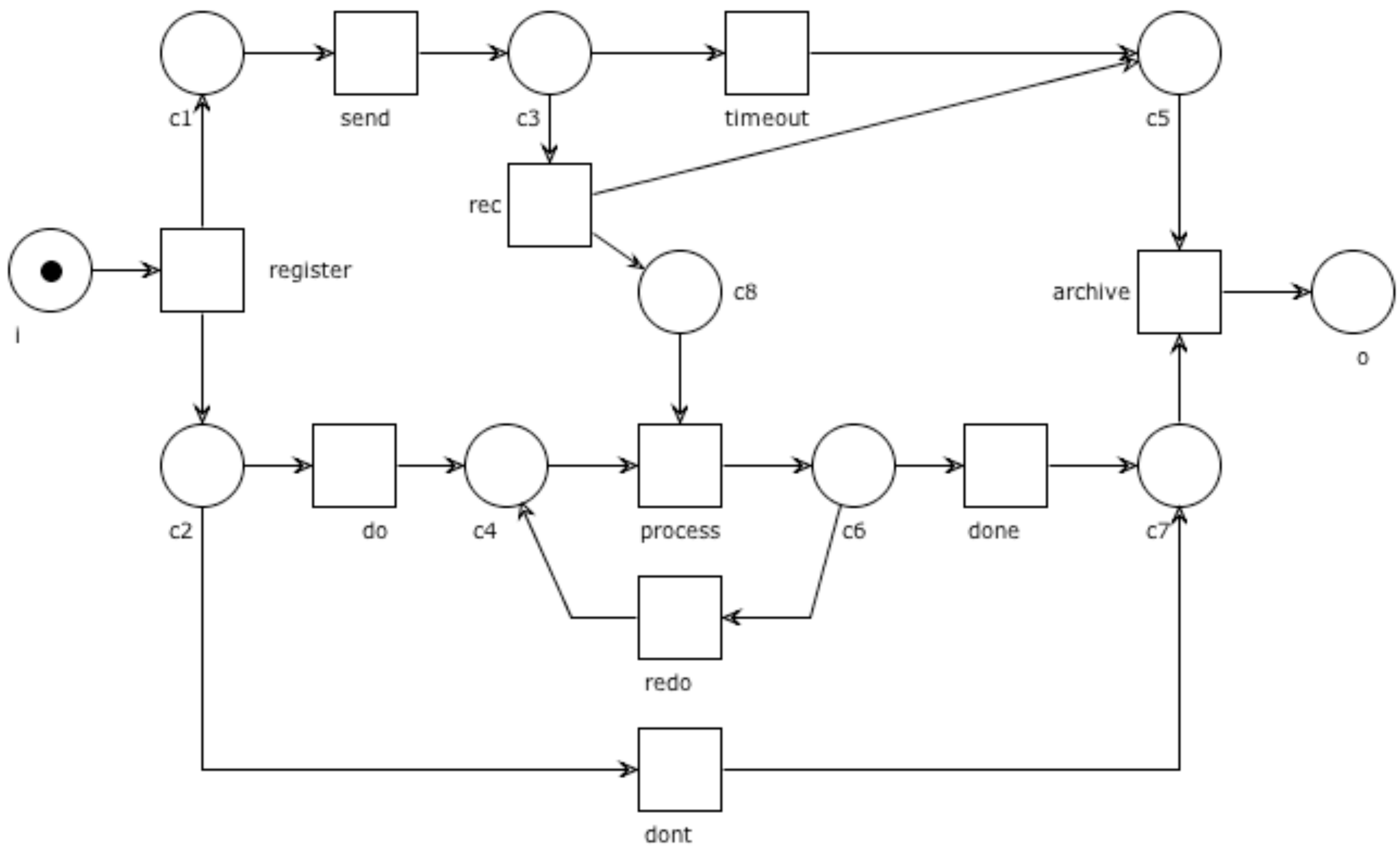
Analyse the following net

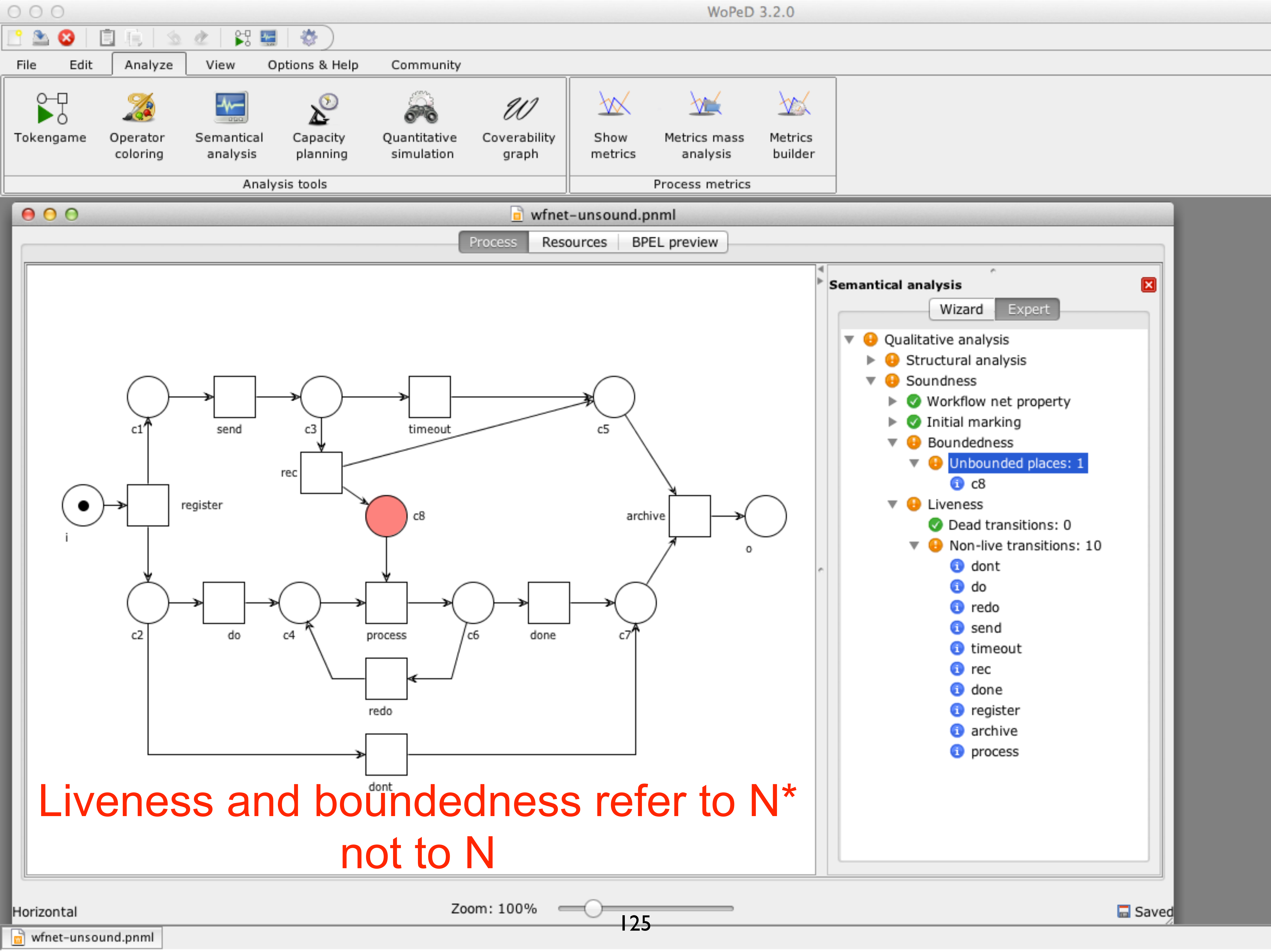



WoPeD 3.7.1
File
Edit
Analyze
View
Options & Help
Community
Tokengame
Operator coloring
Semantical analysis
Capacity planning
Quantitative simulation
Coverability graph
Analysis tools
Show metrics
Metrics mass analysis
Metrics builder
Process metrics
Convert Process to Text
Convert Text to Process
NLP Tools
wfnet-claim-resources-not-shown.pnml
Process
Resources
BPEL preview
begin
register
c1
classify
simple
complex
AND split
c2
c3
phone garage
check insurance
c5
c6
AND join
c8
check insurance
c4
check history
c7
phone garage
decide
OK
pay
c9
NOK
send letter
end
Semantical analysis
Wizard
Expert
Qualitative analysis
Structural analysis
Net statistics
Wrongly used operators: 0
Free-choice violations: 0
S-Components
Wellstructuredness
Soundness
Workflow net property
Initial marking
Boundedness
Unbounded places: 0
Liveness
Dead transitions: 0
Non-live transitions: 0
Horizontal
Zoom: 100%
Not saved

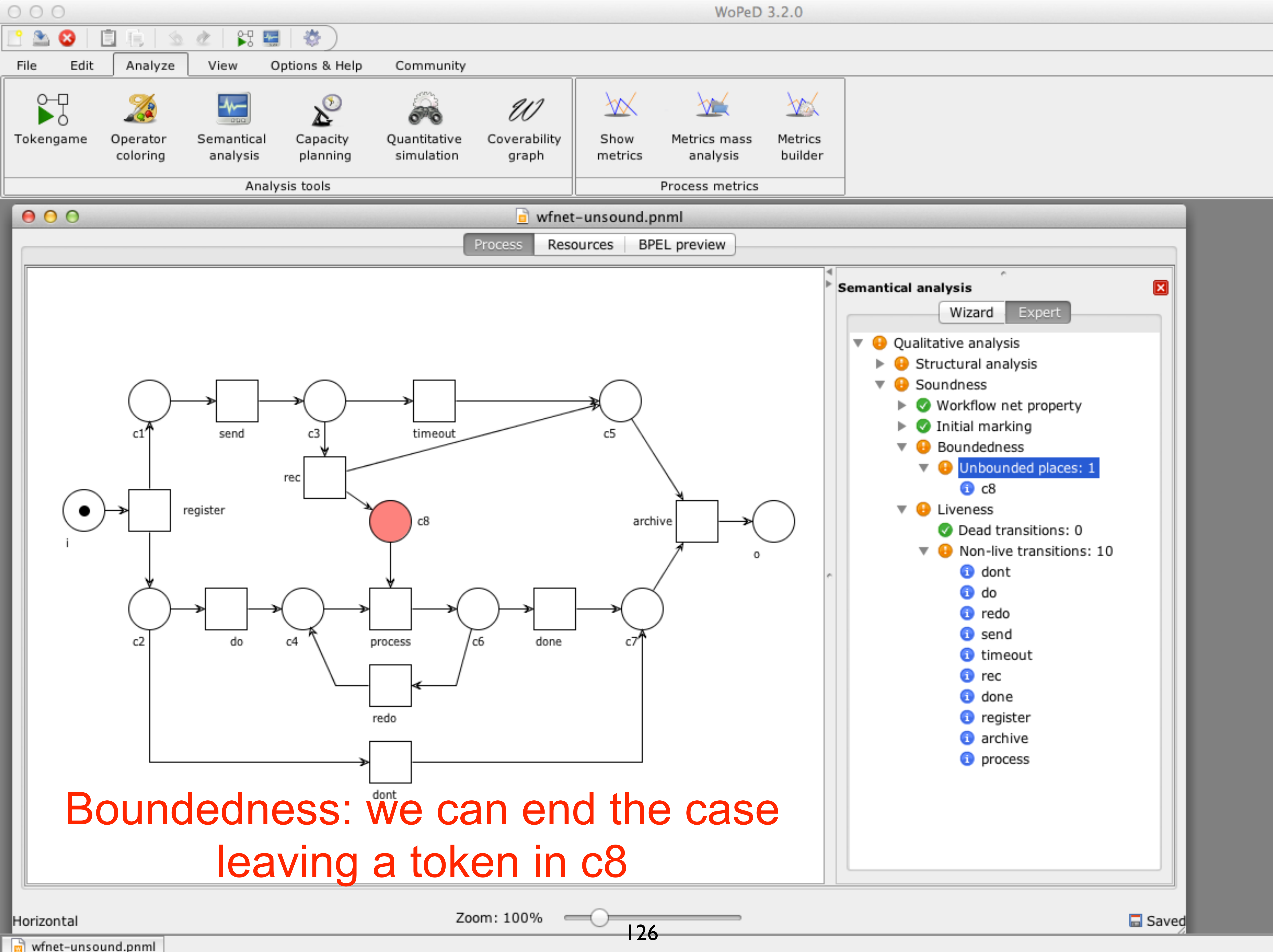
Exercise

Analyse the following net





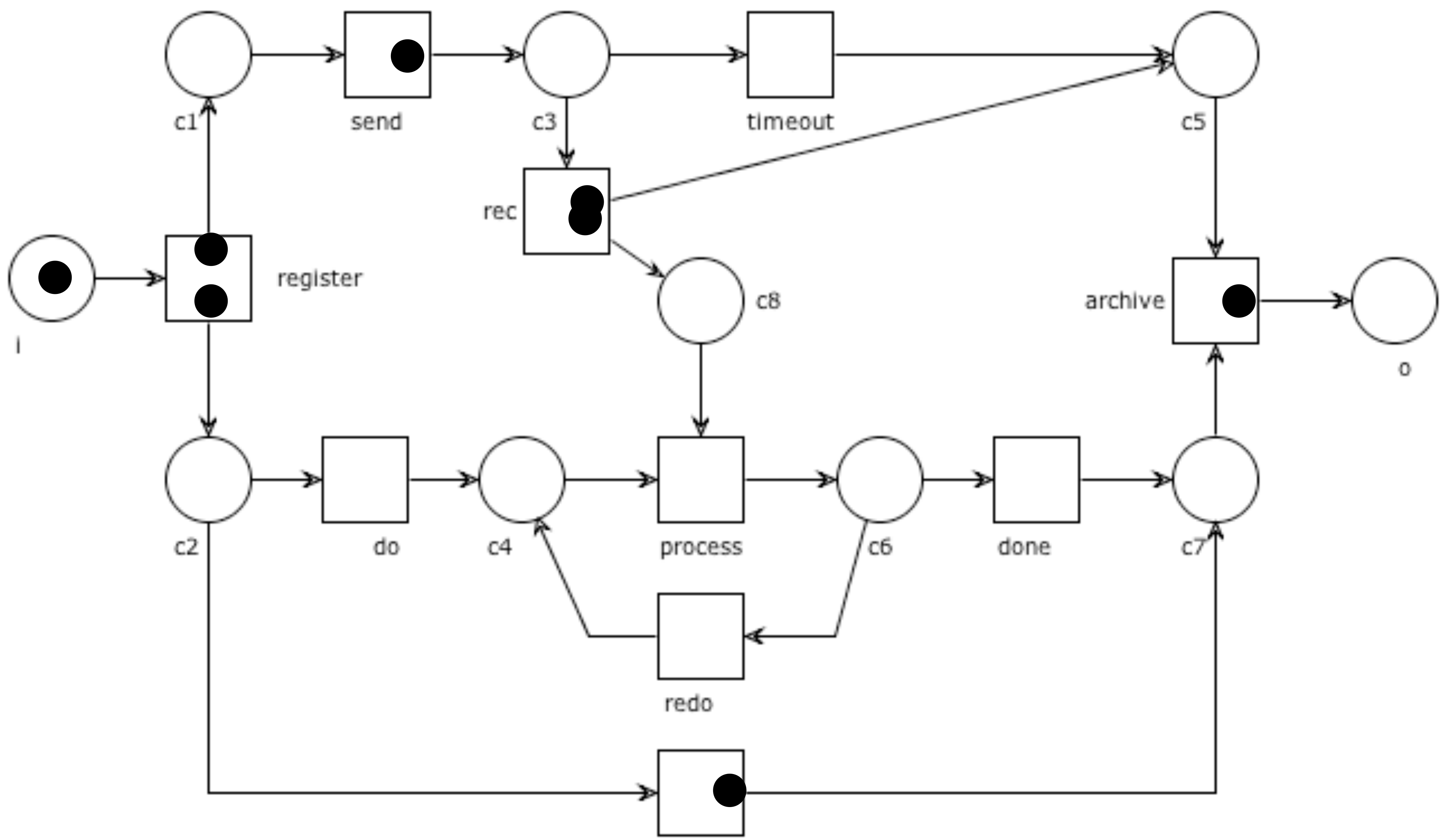
Liveness and boundedness refer to N* not to N



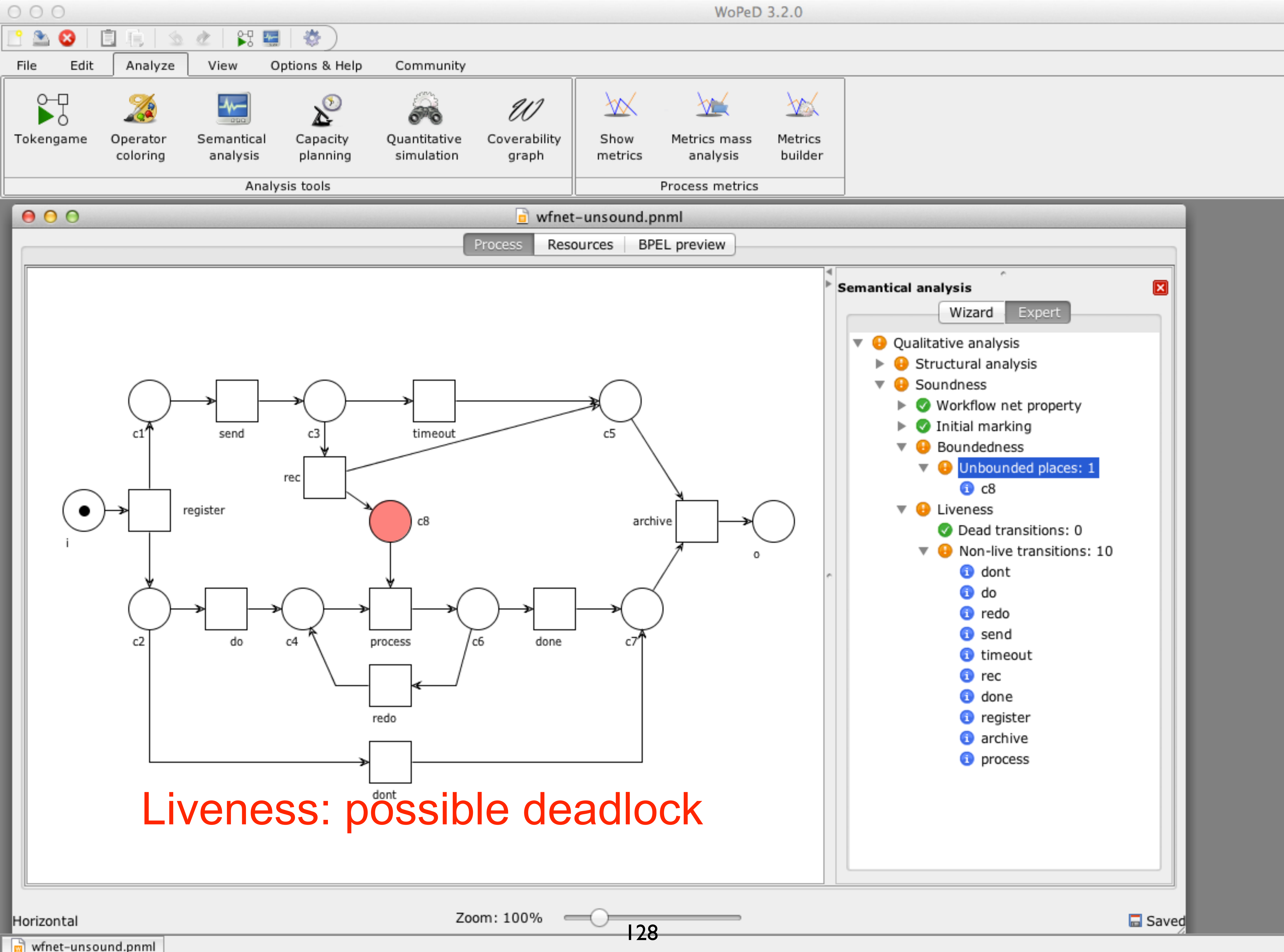
Boundedness: we can end the case leaving a token in c8

Exercise

Analyse the following net



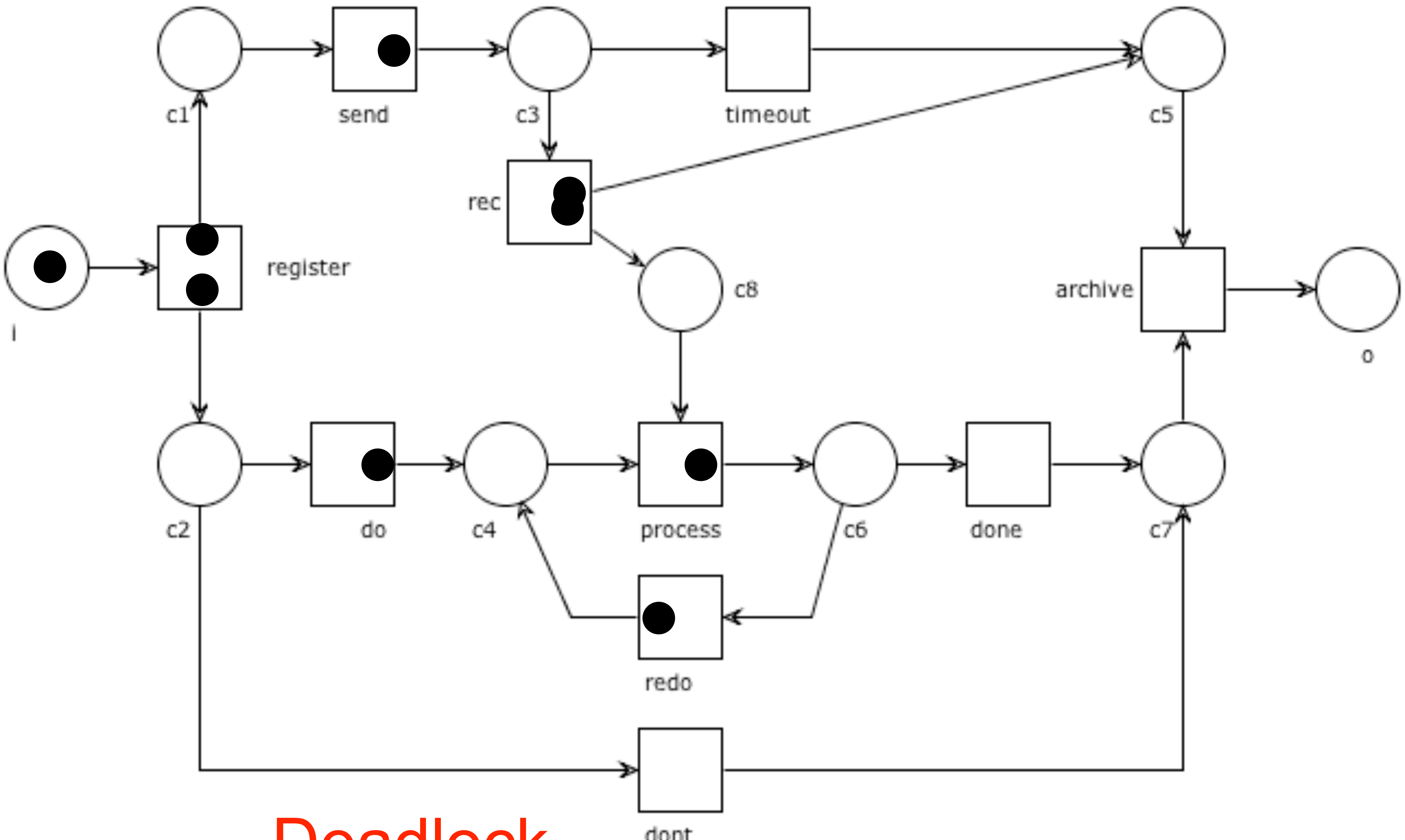
Some tokens left in the net after case completion



Liveness: possible deadlock

Exercise

Analyse the following net



Deadlock

Design and analysis of WF-nets

The workflow of a computer repair service (CRS) can be described as follows.
A customer brings in a defective computer and the CRS checks the defect and hands out a repair cost calculation back.
If the customer decides that the costs are acceptable, the process continues, otherwise she takes her computer home unrepaired.
The ongoing repair consists of two activities, which are executed sequentially but in an arbitrary order.
One activity is to check and repair the hardware,
whereas the other activity is to check and configure the software.
After both activities are completed, the proper system functionality is tested.
If an error is detected the repair procedure is repeated,
otherwise the repair is finished and the computer is returned.

Model the described workflow as a sound workflow net.

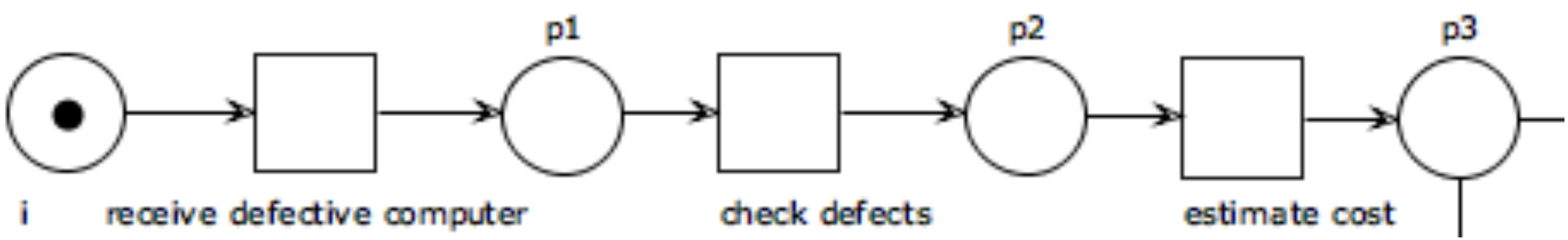
Design and analysis of WF-nets

The workflow of a computer repair service (CRS) can be described as follows.
A customer **brings in a defective computer** and the CRS **checks the defect** and **hands out a repair cost** calculation back.
If the customer decides that the **costs are acceptable**, the process continues, **otherwise** she **takes her computer home** unrepaired.
The ongoing repair consists of two activities, which are executed **sequentially but in an arbitrary order**.
One activity is to **check** and **repair** the **hardware**,
whereas the other activity is to **check** and **configure** the **software**.
After both activities are completed, the proper system **functionality is tested**.
If an **error is detected** the repair **procedure is repeated**,
otherwise the repair is finished and the **computer is returned**.

Model the described workflow as a sound workflow net.

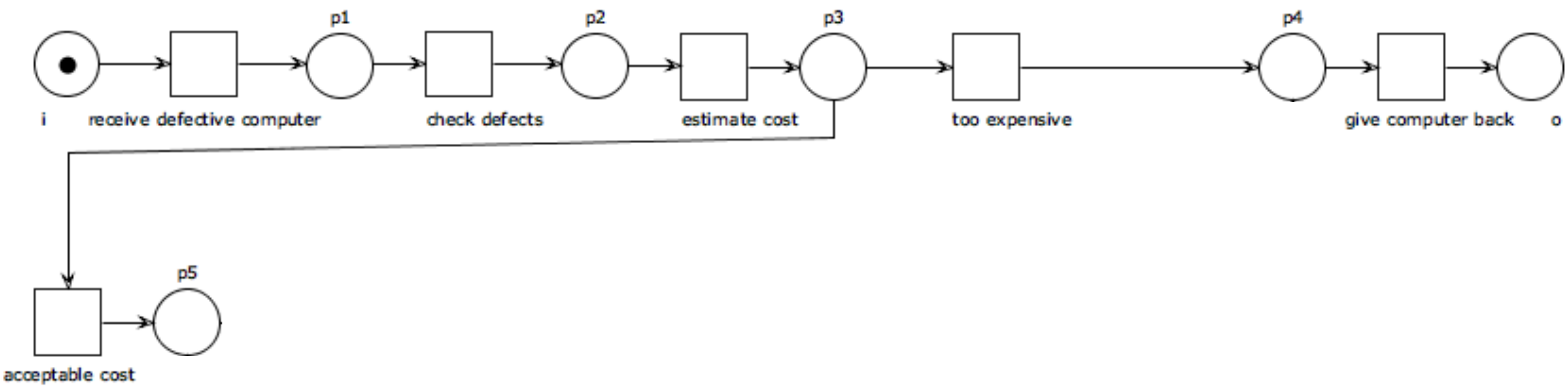
Design and analysis of WF-nets

The workflow of a computer repair service (CRS) can be described as follows.
A customer **brings in a defective computer** and the CRS **checks the defect** and **hands out a repair cost** calculation back.



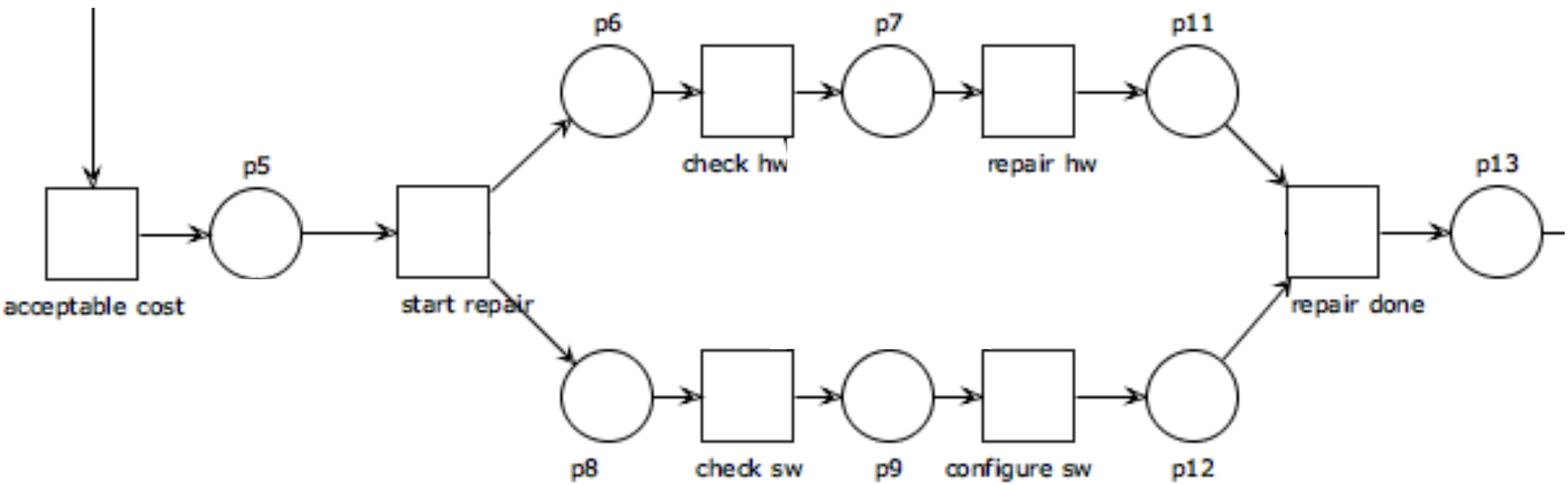
Design and analysis of WF-nets

If the customer decides that the **costs are acceptable**, the process continues, **otherwise** she **takes her computer home** unrepaired.



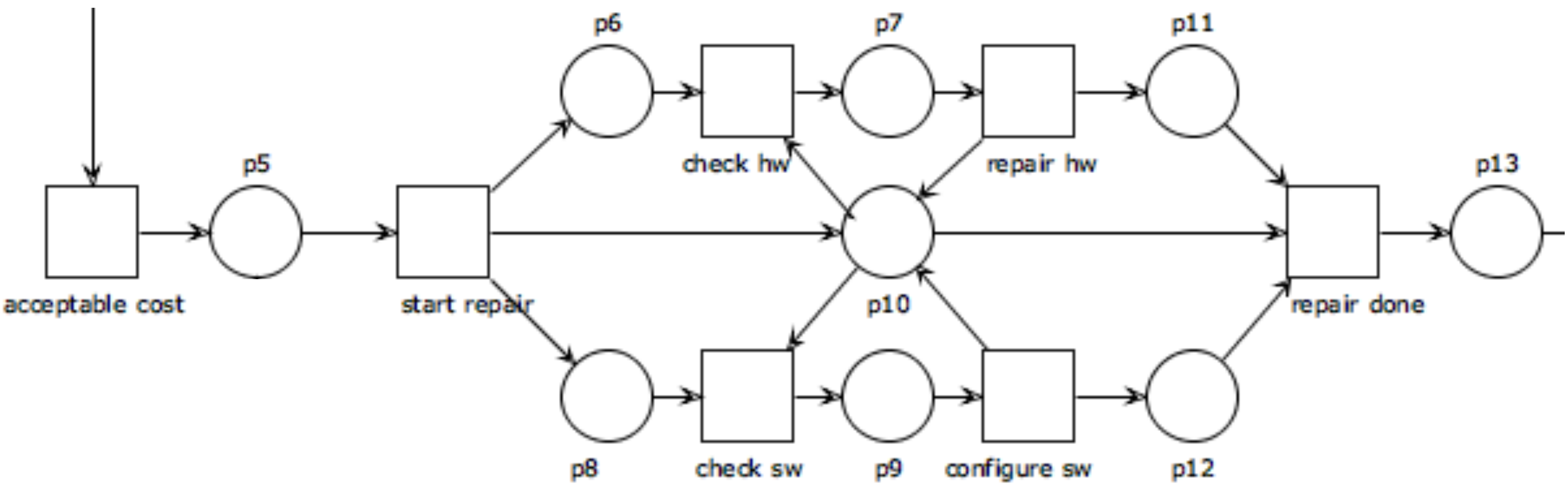
Design and analysis of WF-nets

The ongoing repair consists of two activities, which are executed **sequentially but in an arbitrary order**.
One activity is to **check** and **repair** the **hardware**,
whereas the other activity is to **check** and **configure** the **software**.



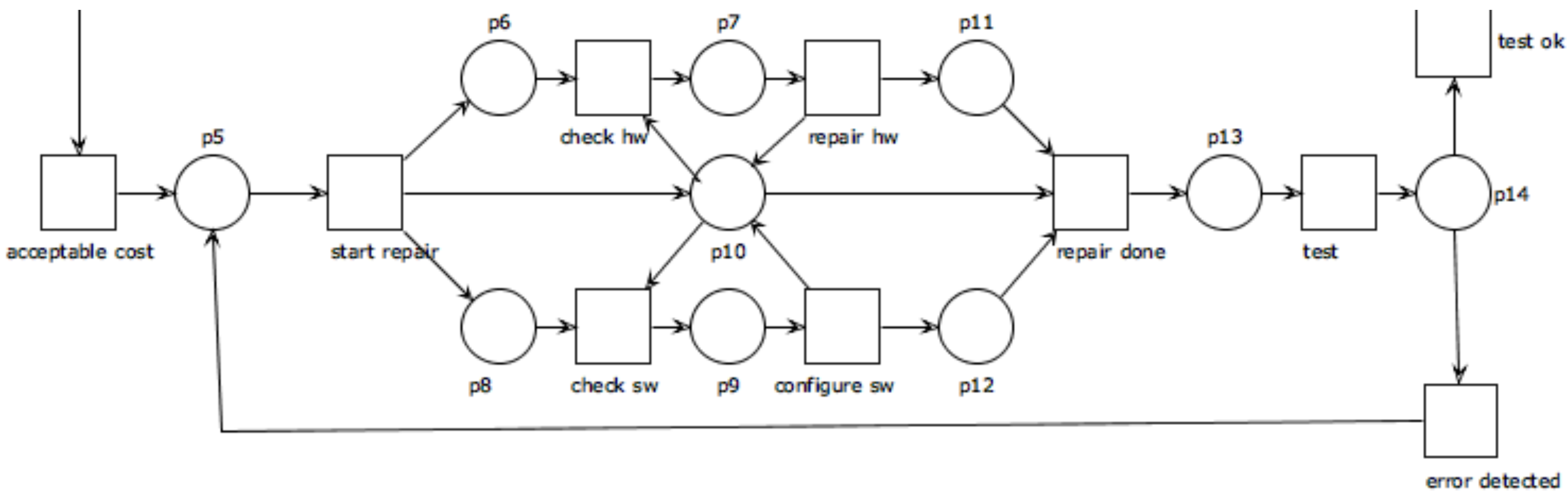
Design and analysis of WF-nets

The ongoing repair consists of two activities, which are executed **sequentially but in an arbitrary order**.
One activity is to **check** and **repair** the **hardware**,
whereas the other activity is to **check** and **configure** the **software**.



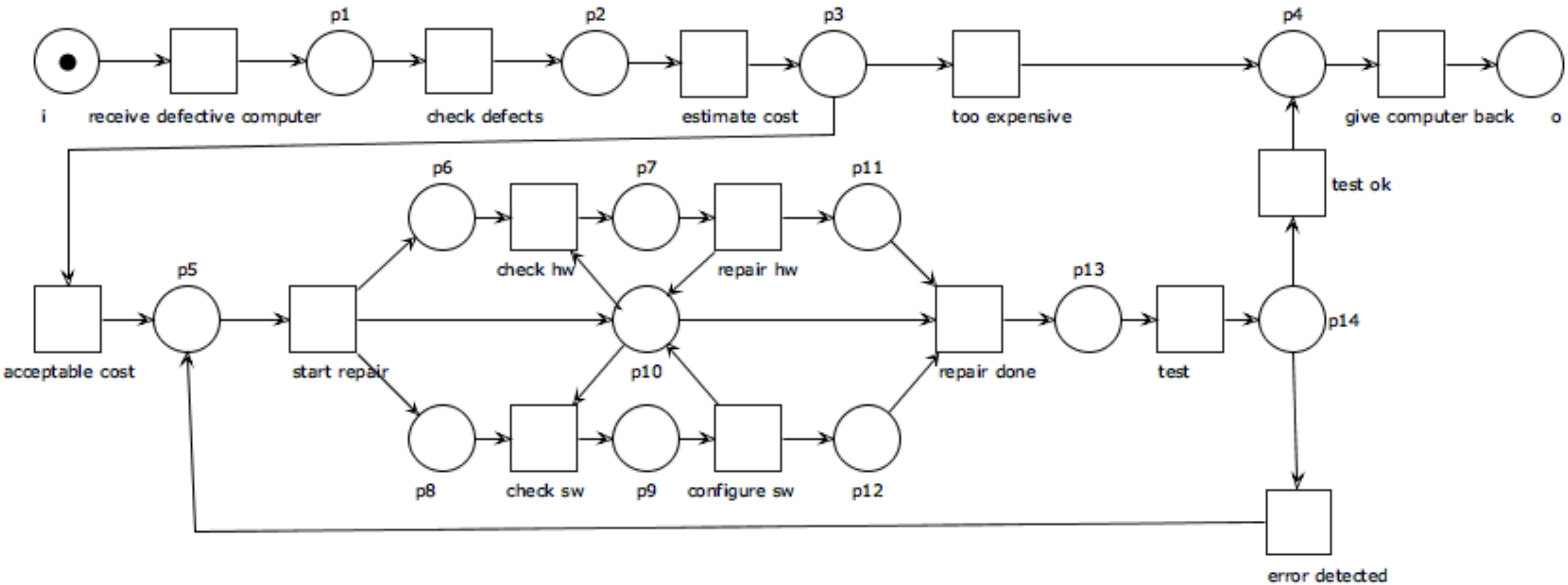
Design and analysis of WF-nets

After both activities are completed, the proper system **functionality is tested**.
If an **error is detected** the repair **procedure is repeated**,



Design and analysis of WF-nets

otherwise the repair is finished and the **computer is returned**.



Design and analysis of WF-nets