

Subclasses, Reduction Rules, and Process Mining

prof.dr.ir. Wil van der Aalst

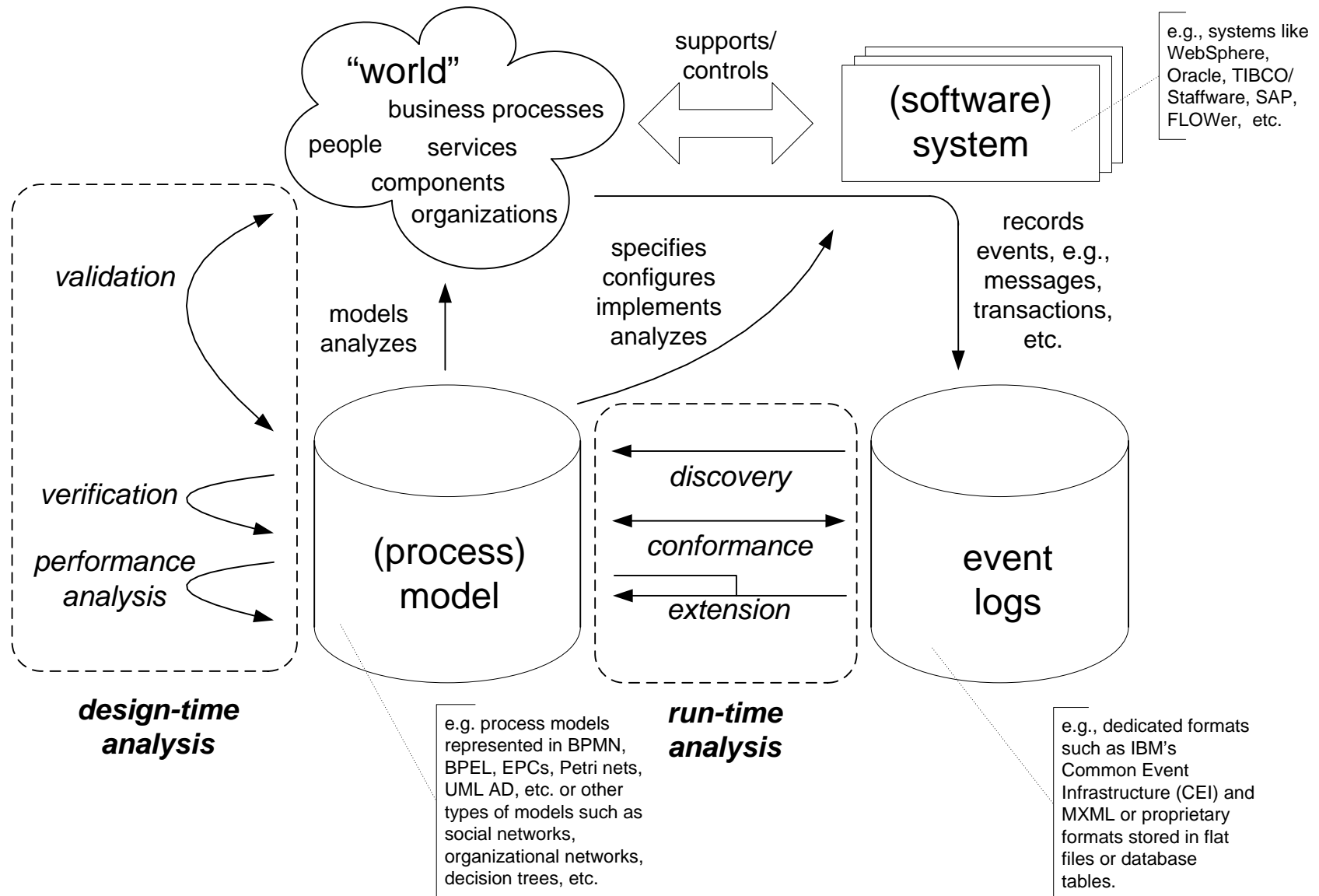


TU/e

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

Design-time analysis vs run-time analysis



Outline

- **Reduction rules**
- **Subclasses**
- **Process mining (introduction)**

Relevant material

1. Jörg Desel, Wolfgang Reisig: Place/Transition Petri Nets. Petri Nets 1996: 122-173. DOI: 10.1007/3-540-65306-6_15
<http://www.springerlink.com/content/x6hn592l35866lu8/fulltext.pdf>
2. Tadao Murata, Petri Nets: Properties, Analysis and Applications, Proceedings of the IEEE. 77(4): 541-580, April, 1989. <http://dx.doi.org/10.1109/5.24143>
<http://ieeexplore.ieee.org/iel1/5/911/00024143.pdf>
3. Wil van der Aalst: Process Mining: Discovery, Conformance and Enhancement of Business Processes, Springer Verlag 2011 (chapters 1 & 5)
 - a) Chapter 1: DOI: 10.1007/978-3-642-19345-3_1
<http://www.springerlink.com/content/p443h219v3u3537l/fulltext.pdf>
 - b) Chapter 5: DOI: 10.1007/978-3-642-19345-3_5
<http://www.springerlink.com/content/u58h17n3167p0x1u/fulltext.pdf>
 - c) Events logs: <http://www.processmining.org/book/>

Reduction rules



TU/e

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

Basic properties revisited ...

terminating

it has only finite occurrence sequences

deadlock-free

each reachable marking enables a transition

live

each reachable marking enables an occurrence sequence containing all transitions

bounded

each place has an upper bound that holds for all reachable markings

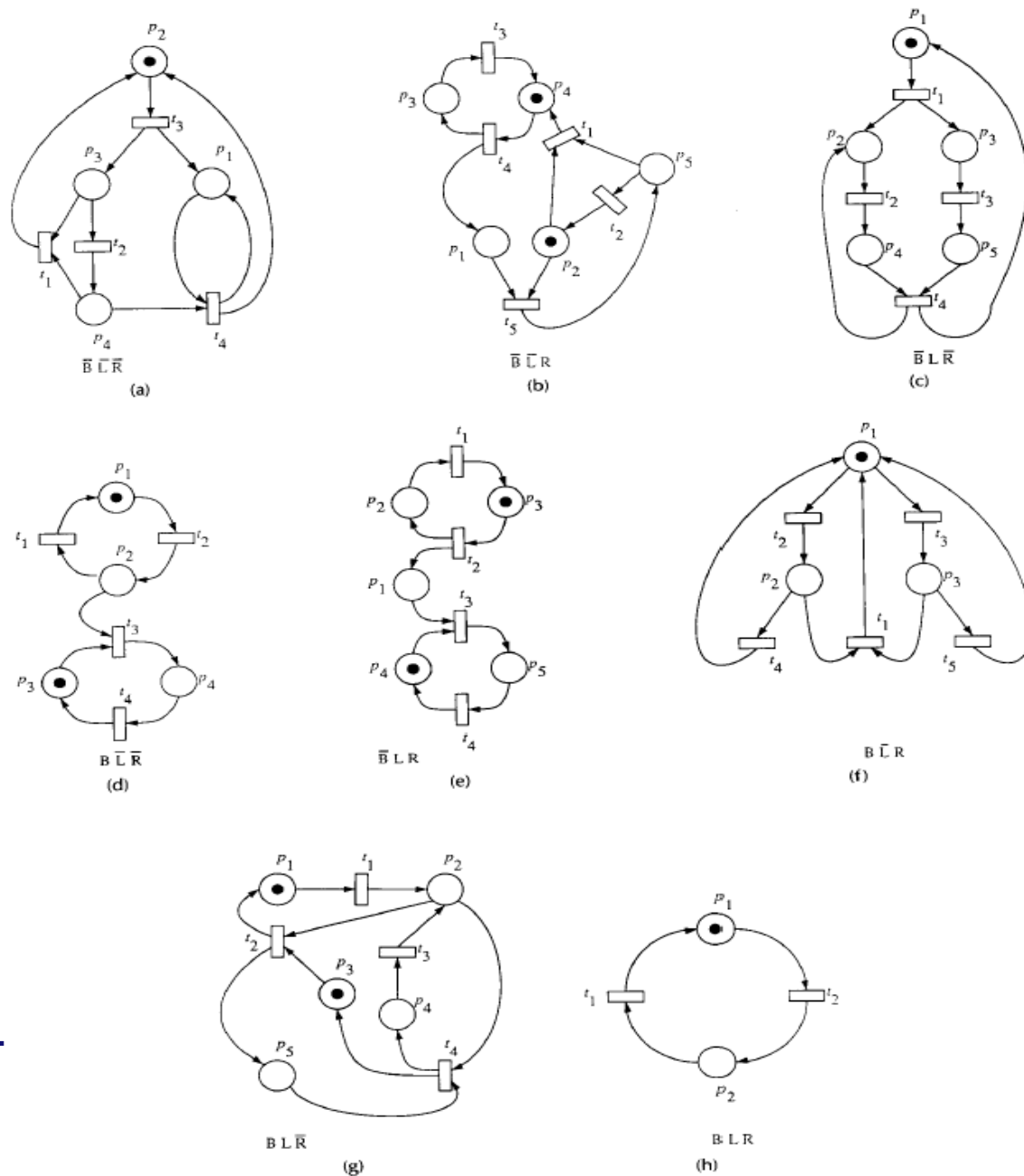
1-safe

1 is a bound for each place s

reversible

m_0 is reachable from each reachable marking, i.e., the initial marking is a so-called **home marking**.

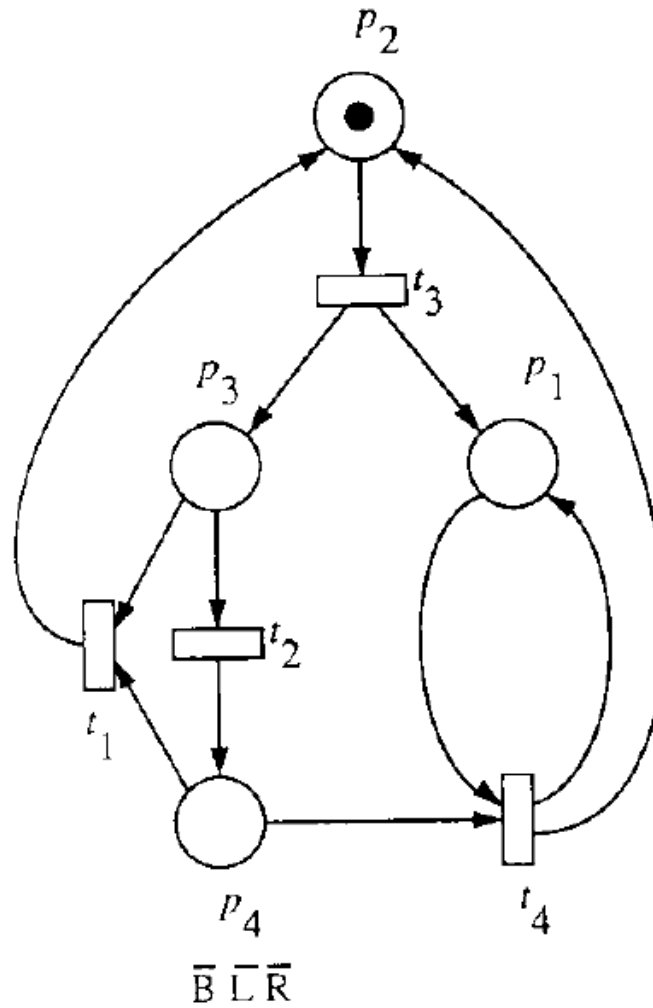
B=Bounded
L=Live
R= Reversible



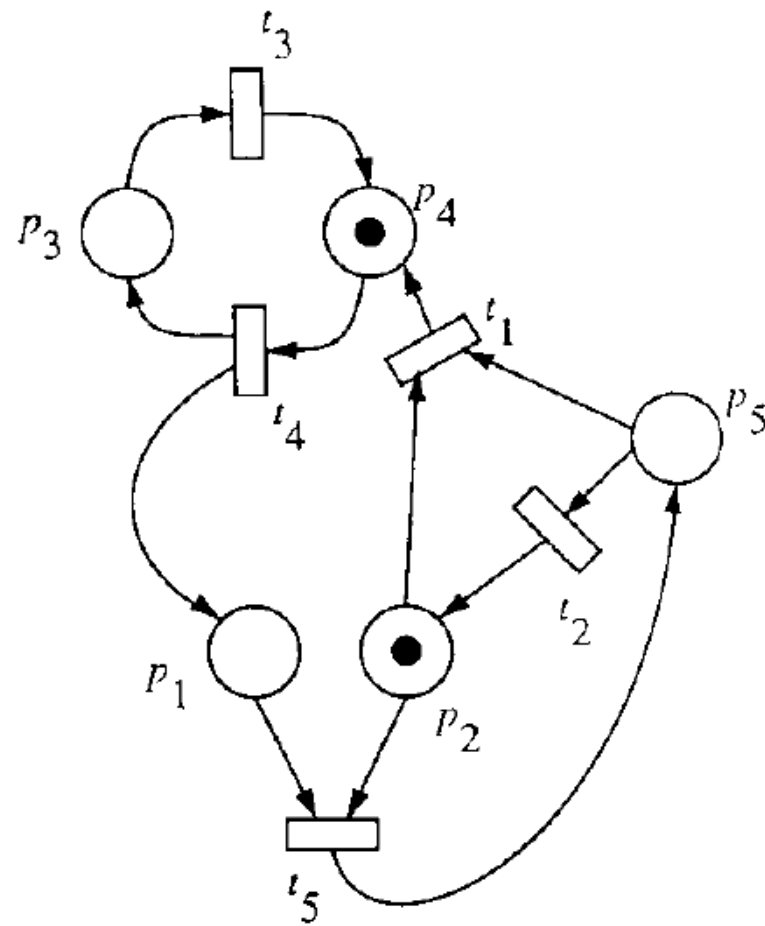
Taken from Murata.

$$2^3=8$$

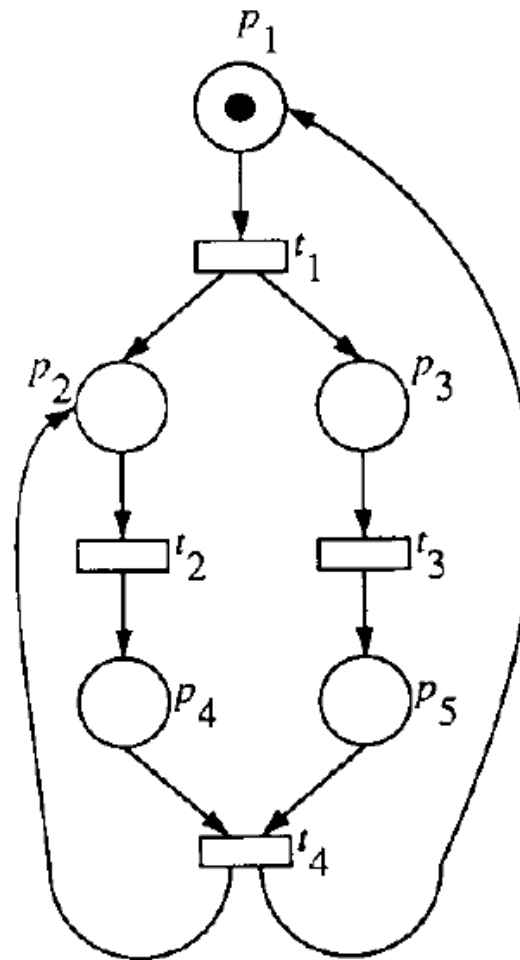
Fig. 17. Examples of Petri nets having all possible combinations of B (bounded), \bar{B} (unbounded), L (live), \bar{L} (nonlive), R (reversible), and \bar{R} (nonreversible) properties. (a) $\bar{B} \bar{L} \bar{R}$ (t_1 dead, p_1 unbounded). (b) $\bar{B} \bar{L} R$ (t_1 dead, p_1 unbounded). (c) $\bar{B} L \bar{R}$ (p_2 unbounded). (d) $B \bar{L} \bar{R}$ (t_1, t_2, t_3, t_4 not L4-live). (e) $\bar{B} L R$ (p_1 unbounded). (f) $B \bar{L} R$ (t_1 dead). (g) $B L \bar{R}$. (h) $B L R$.



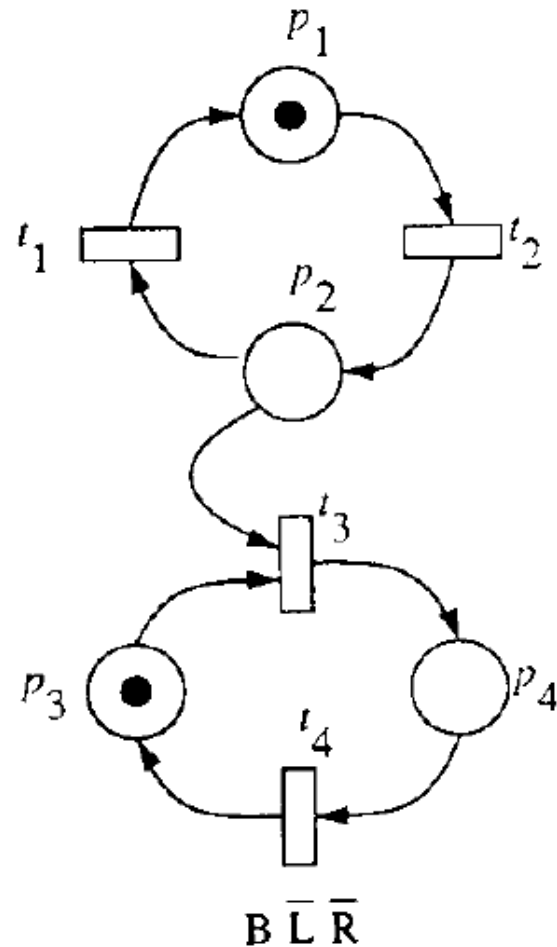
(a)


 $\bar{B} \bar{L} R$

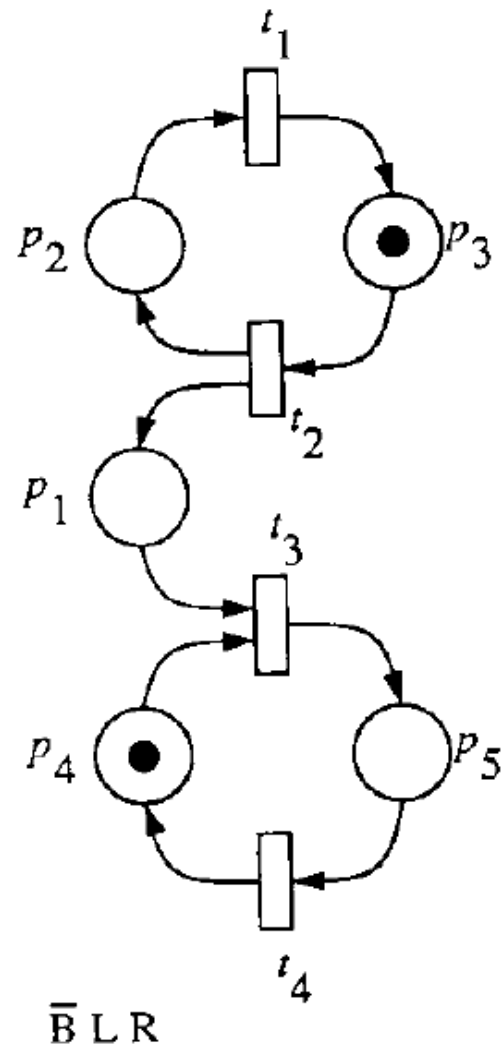
(b)


 $\bar{B} \ L \ \bar{R}$

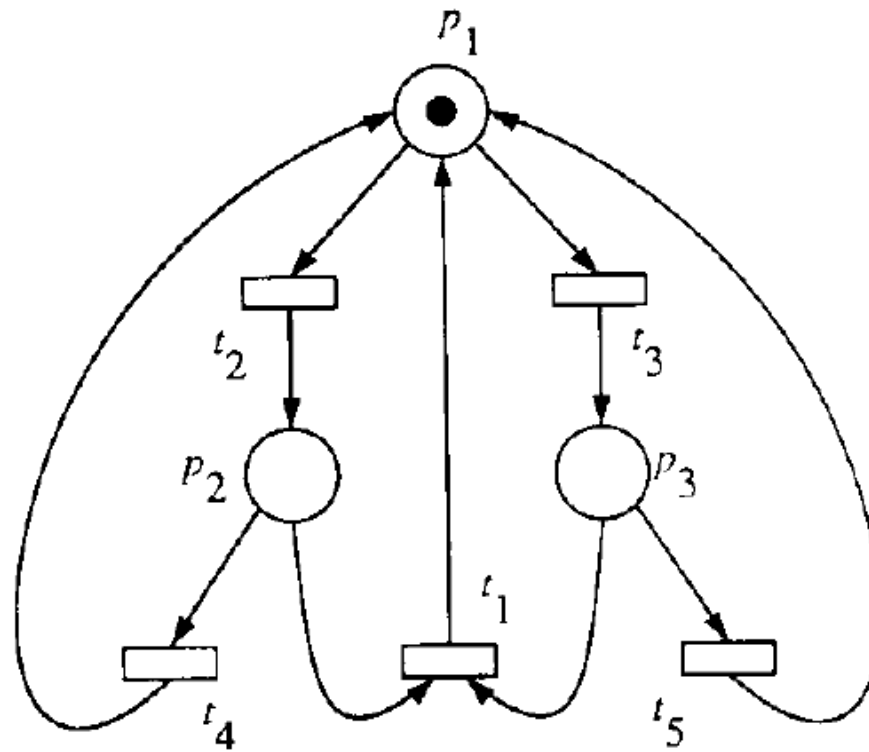
(c)



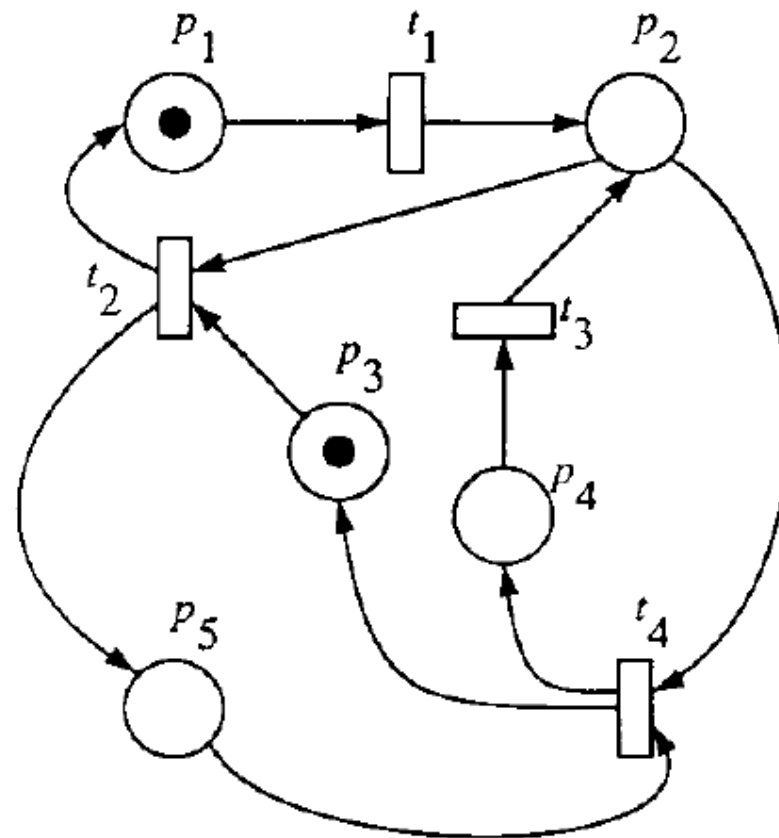
(d)



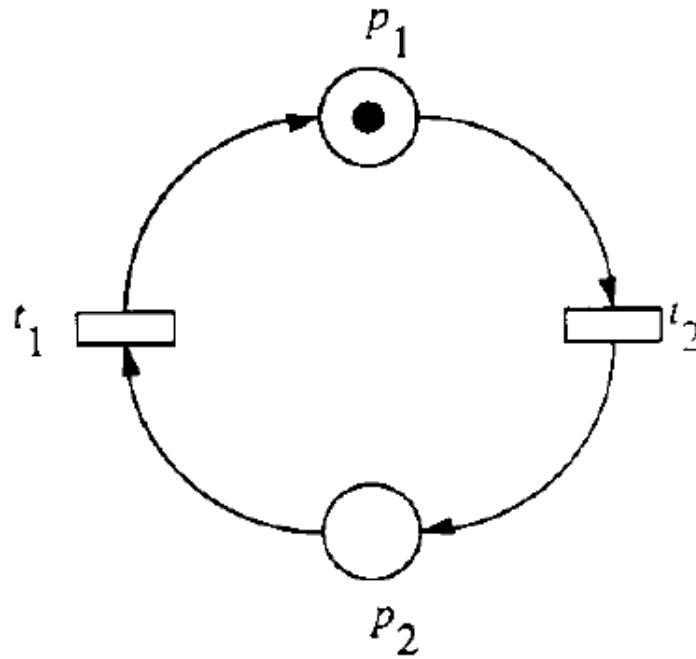
(e)



B \bar{L} R
(f)


 $B \ L \ \bar{R}$

(g)



B L R

(h)

Reduction rules

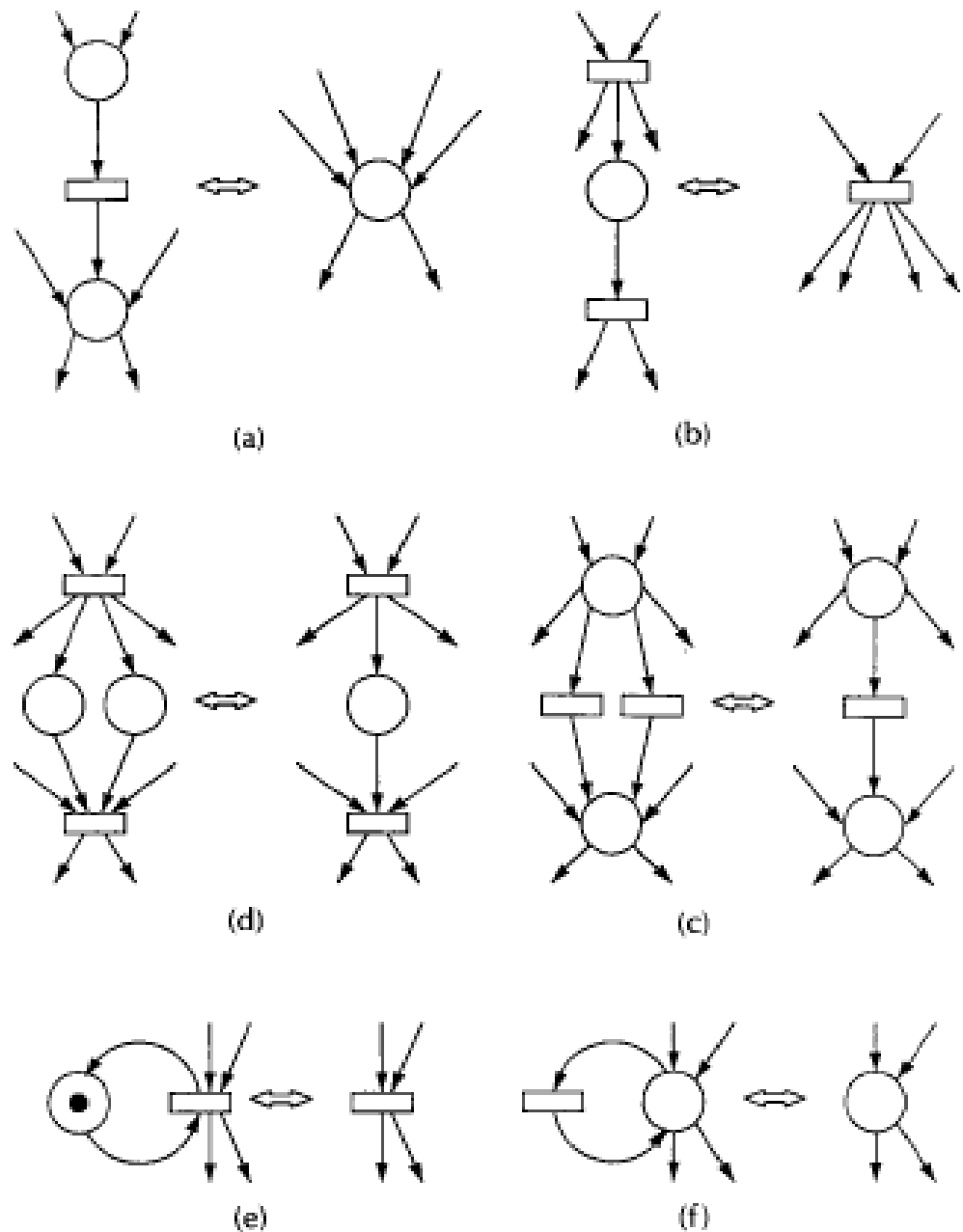
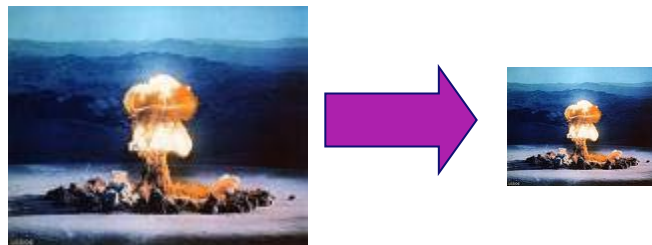
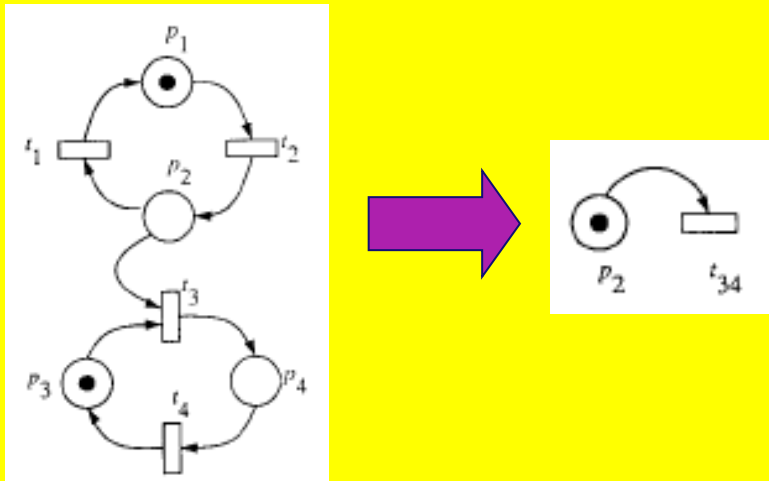
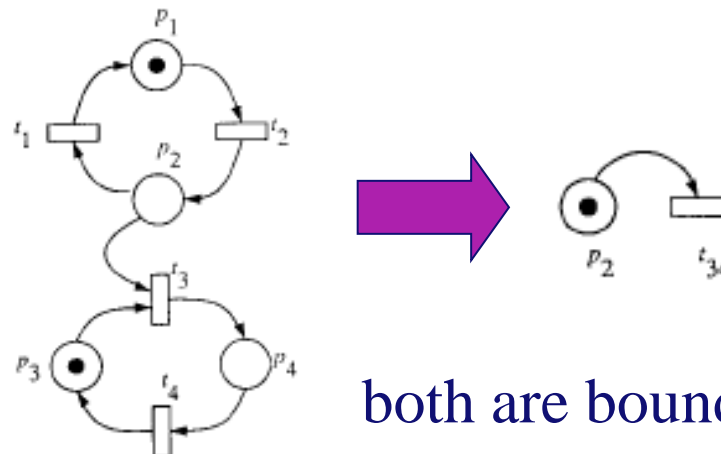


Fig. 22. Six transformations preserving liveness, safeness, and boundedness.

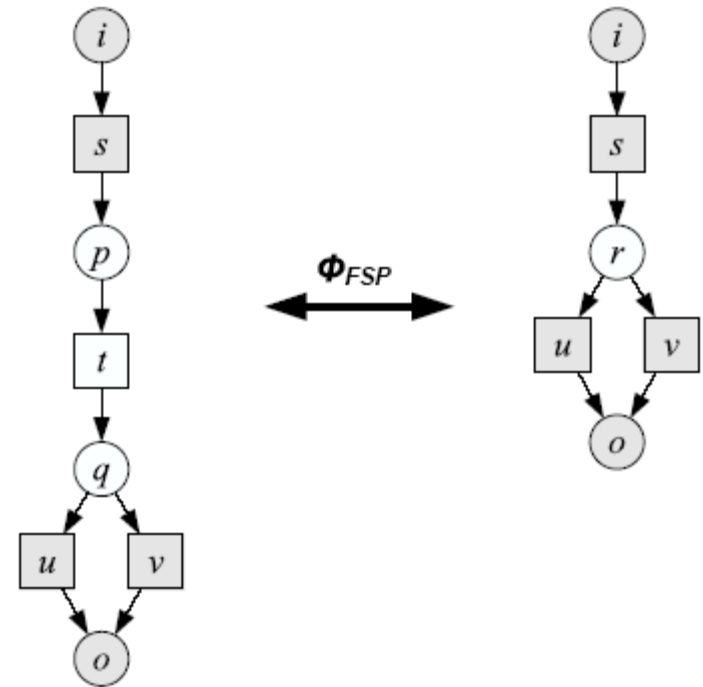
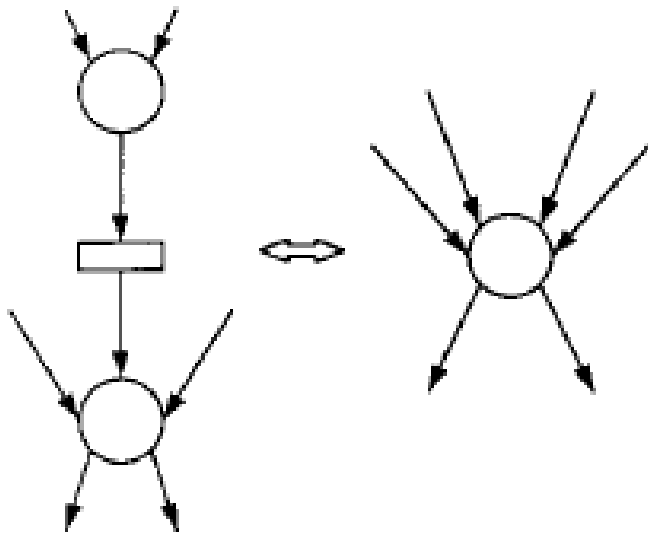
Goal of reduction rules

- Let (N, M) and (N', M') be the Petri nets before and after. Then (N', M') is live, safe, or bounded if and only if (N, M) is live, safe, or bounded, respectively.

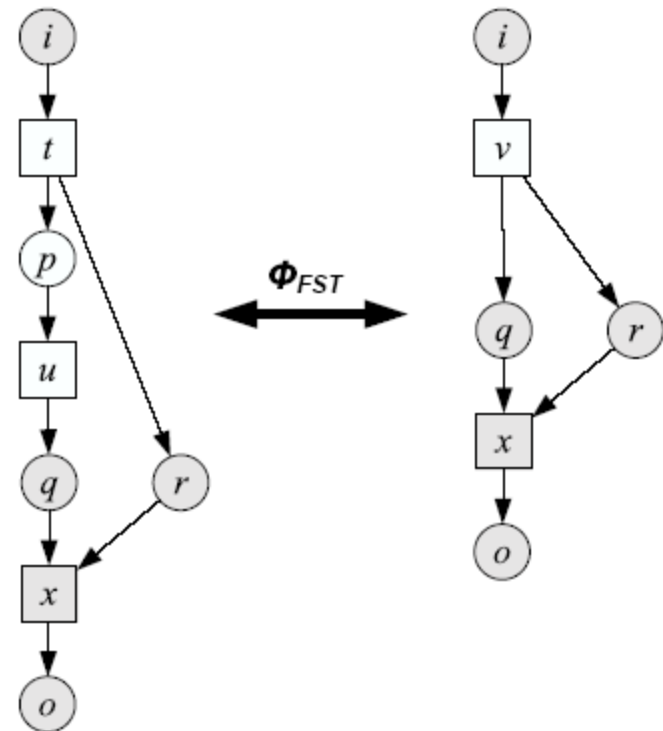
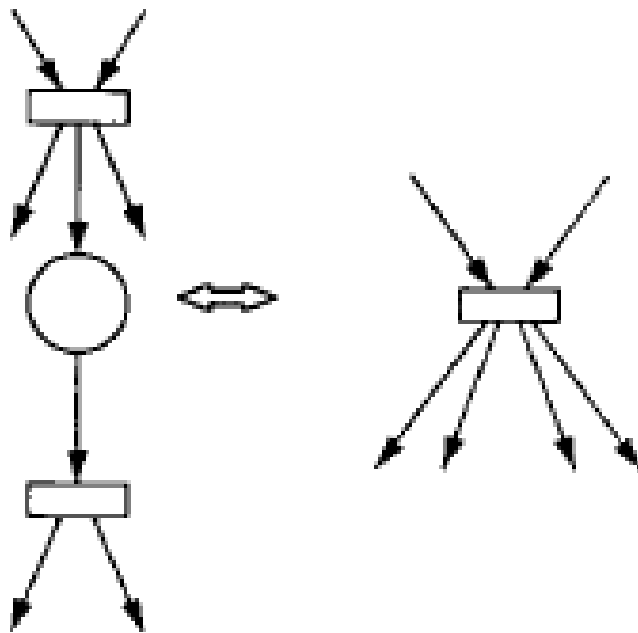


both are bounded and non-live

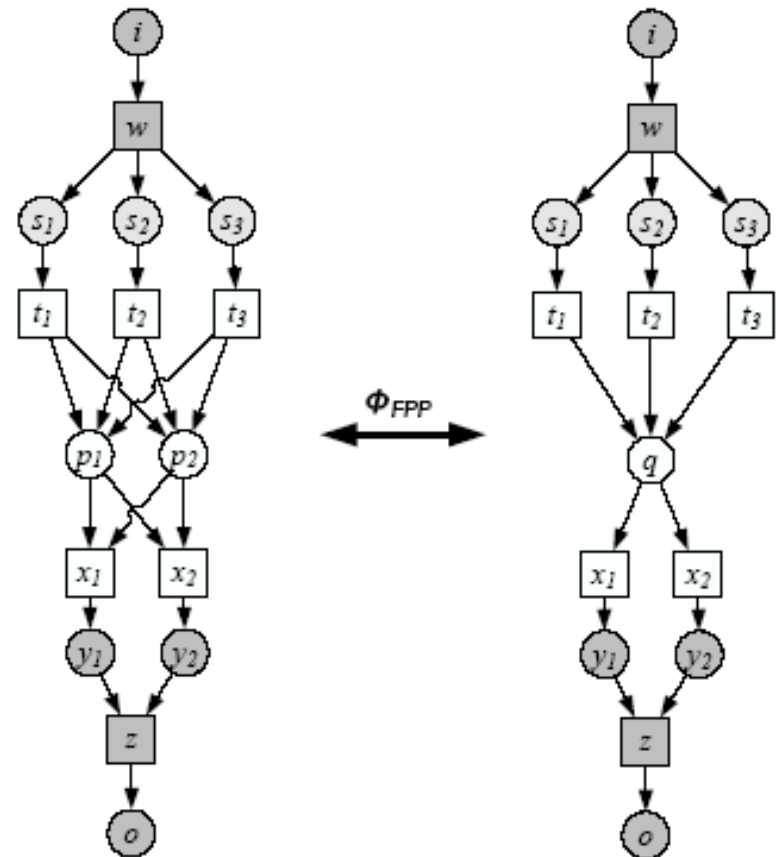
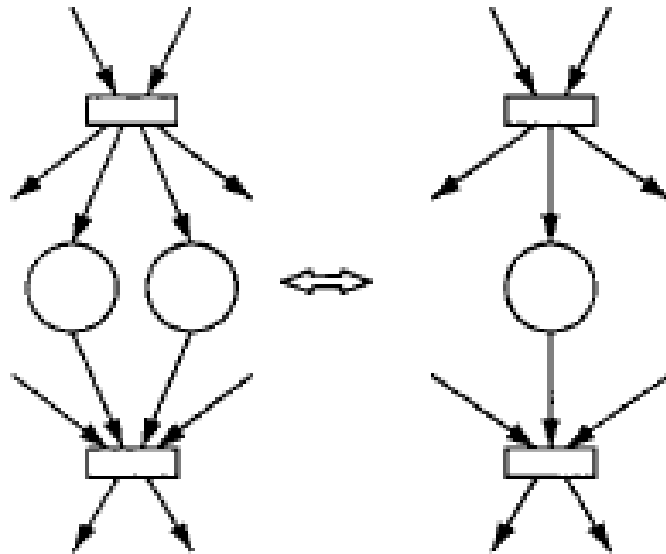
Fusion of Series Places (FSP)



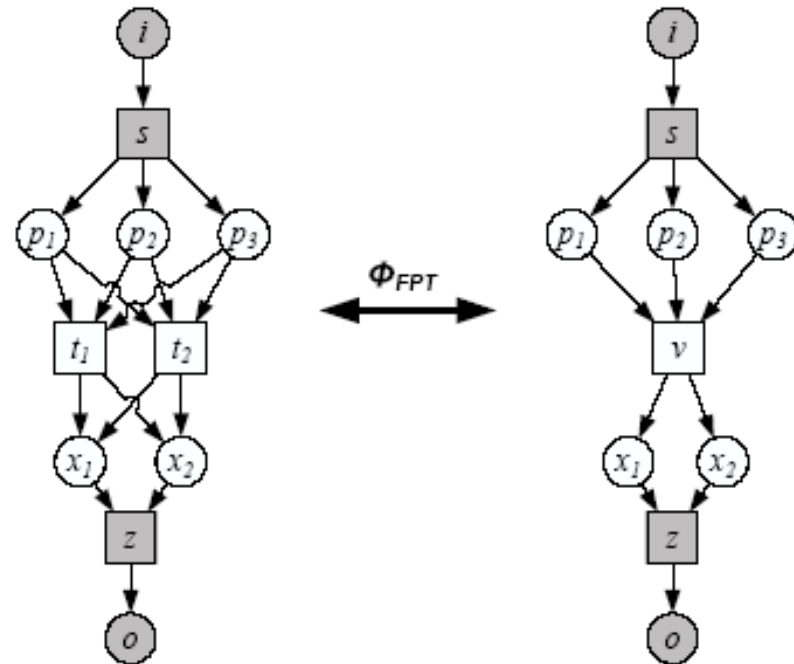
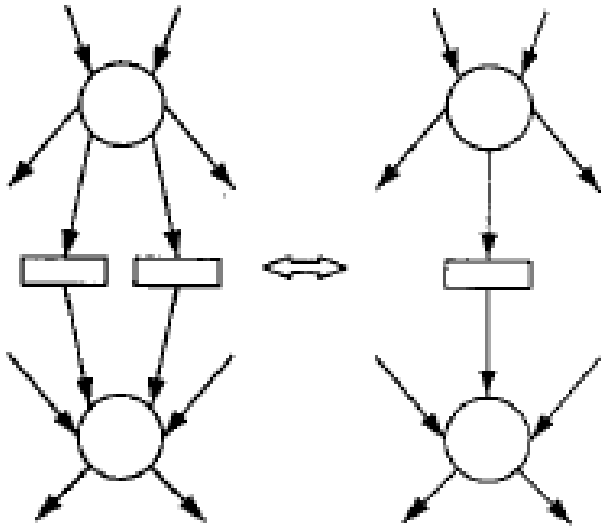
Fusion of Series Transitions (FST)



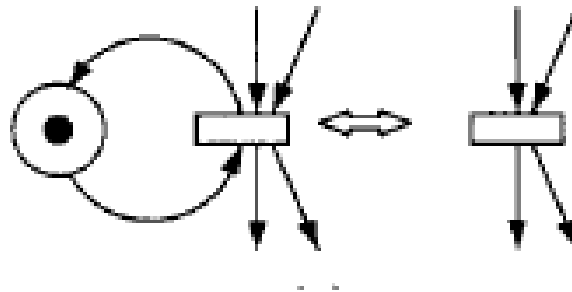
Fusion of Parallel Places (FPP)



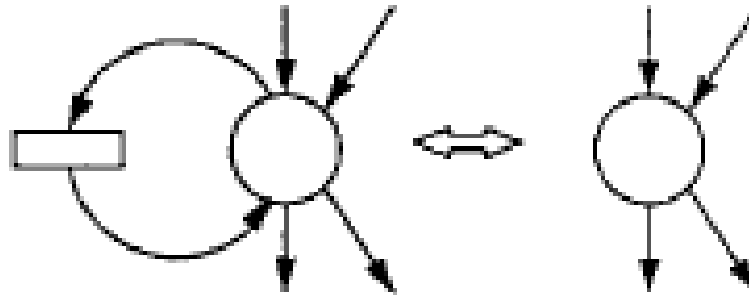
Fusion of Parallel Transitions (FPT)



Elimination of Self-loop Places (ESP)

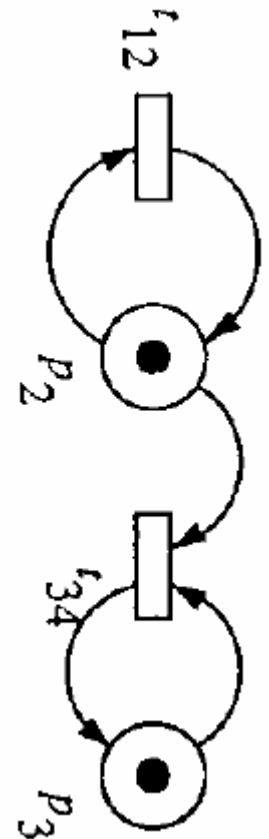
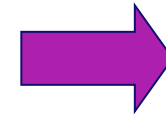
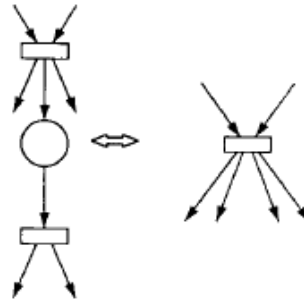
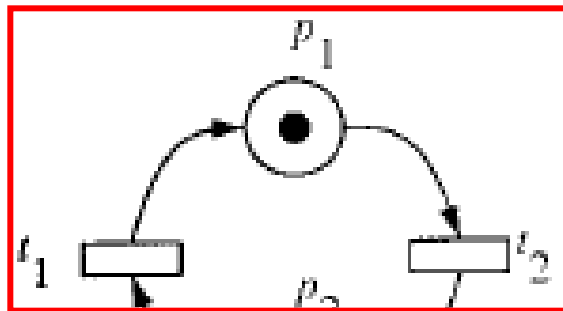


Elimination of Self-loop Transitions (EST)

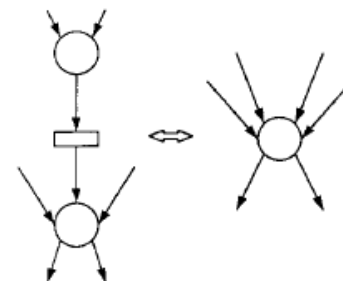
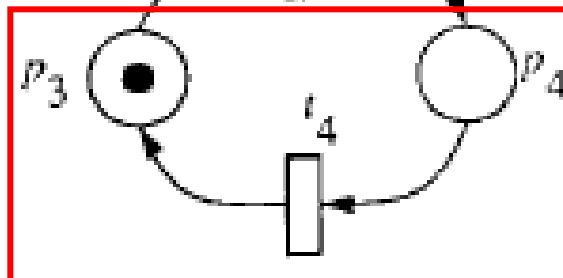


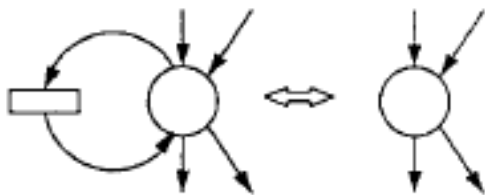
Example

Fusion of Series Transitions (FST)

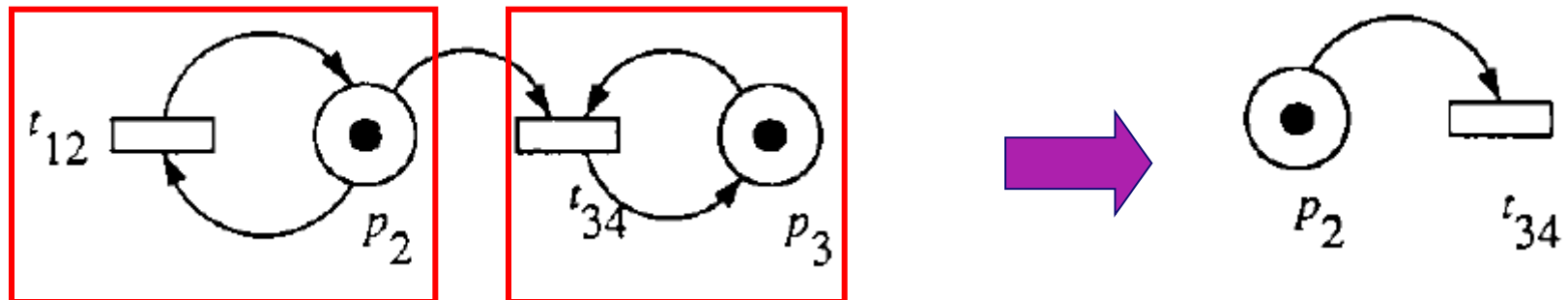


Fusion of Series Places (FSP)

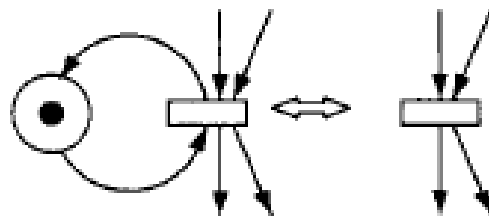


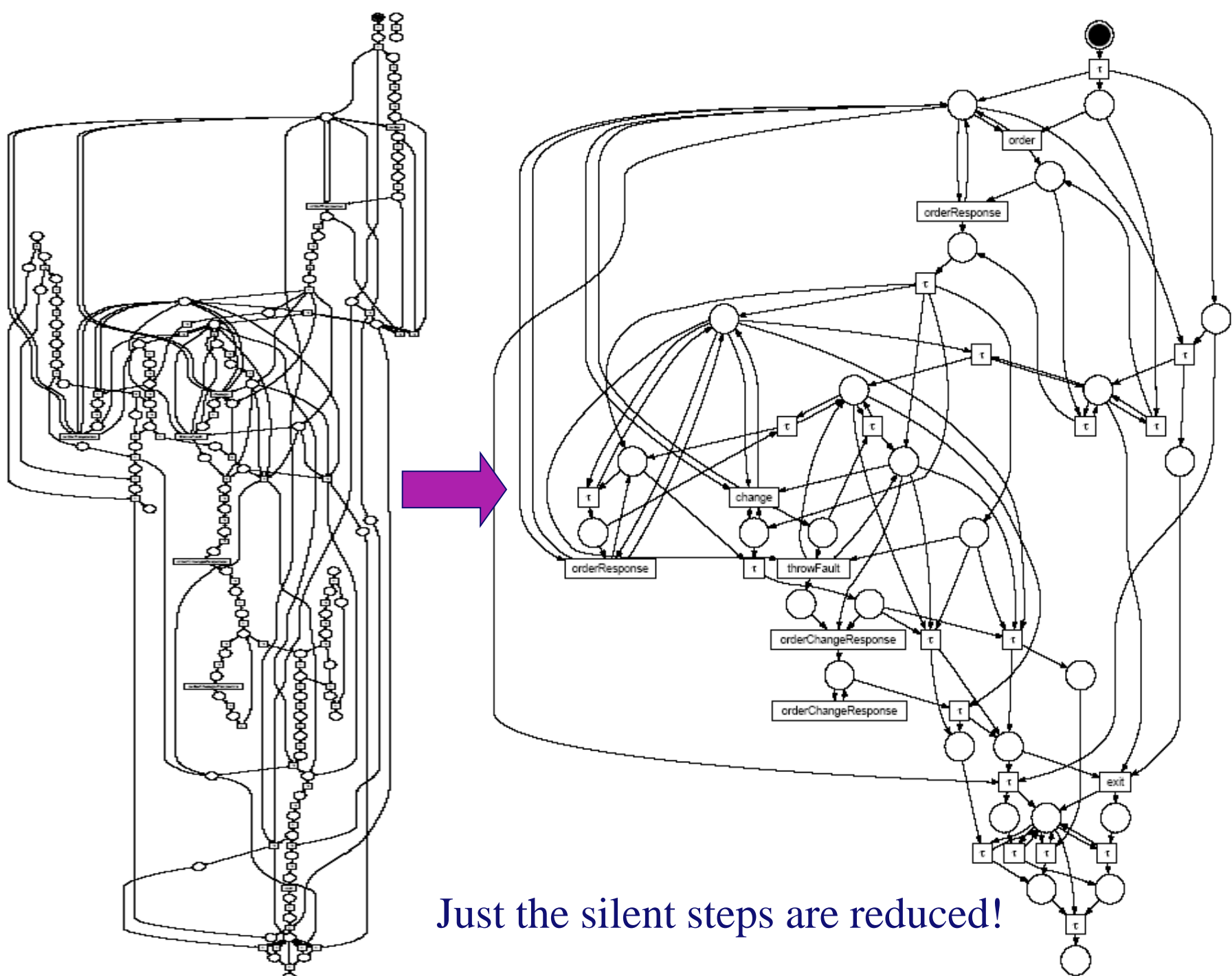


Elimination of Self-loop Transitions (EST)



Elimination of Self-loop Places (ESP)





More information

- Supported by tools like ProM and Woflan.
- G. Berthelot. Checking properties of nets using transformation. In Advances in Petri Nets 1985, pages 19–40, London, UK, 1986. Springer-Verlag.
- G. Berthelot. Transformations and Decompositions of Nets. In W. Brauer, W. Reisig, and G. Rozenberg, editors, Petri Nets: Central Models and Their Properties, Advances in Petri Nets, Proceedings of an Advanced Course, Part 1, volume 254 of Lecture Notes in Computer Science, pages 359–376, Bad Honnef, September 1986. Springer-Verlag.
- J. Desel and J. Esparza. Free Choice Petri Nets, volume 40 of Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, Cambridge, United Kingdom, 1995.
- M.T. Wynn, H.M.W. Verbeek, W.M.P. van der Aalst, A.H.M. ter Hofstede and D. Edmond, Reduction Rules for Reset Workflow Nets, BPM Technical Report, BPM-06-25, BPMcenter.org, 2006.

Subclasses



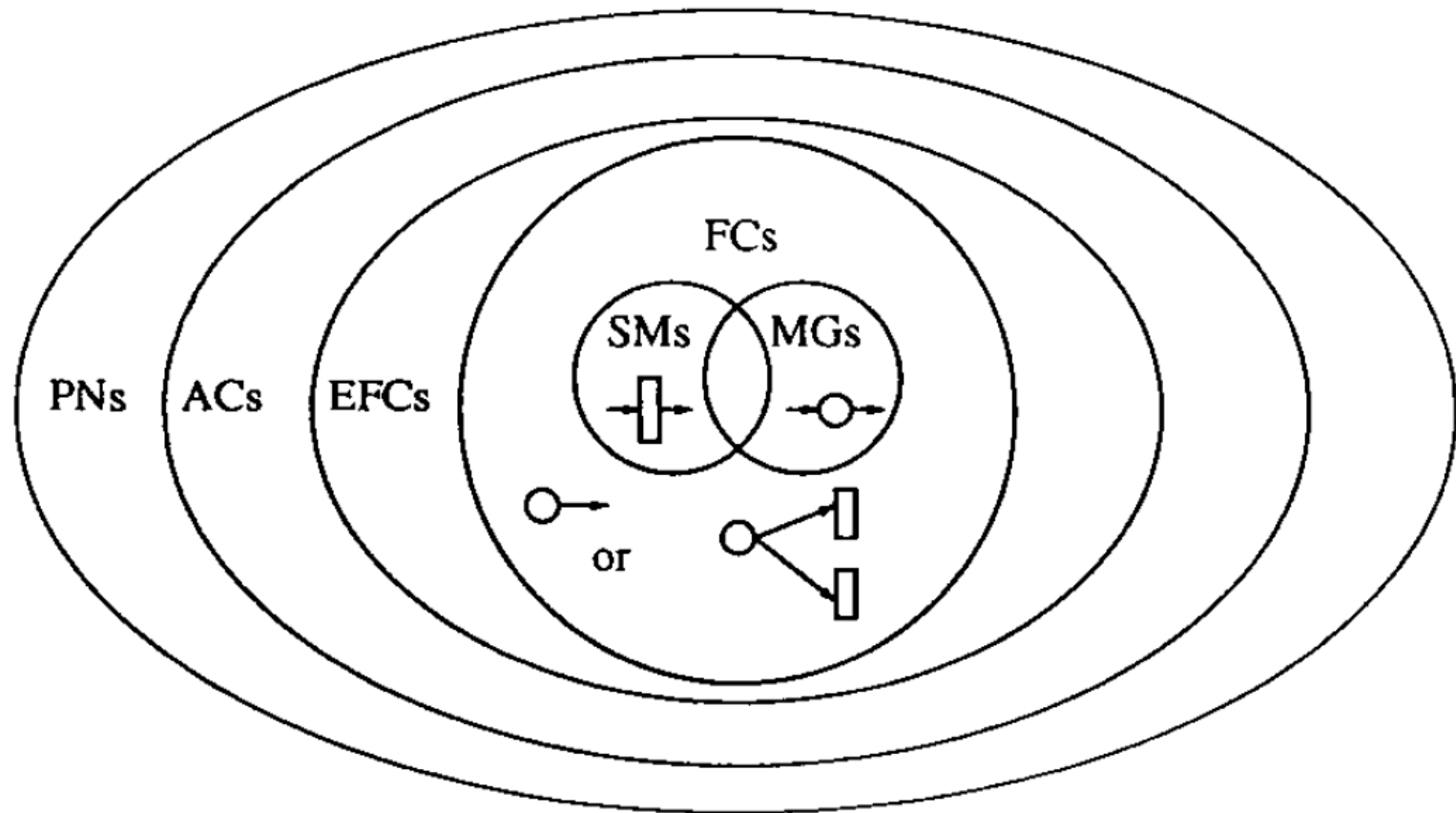
TU/e

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

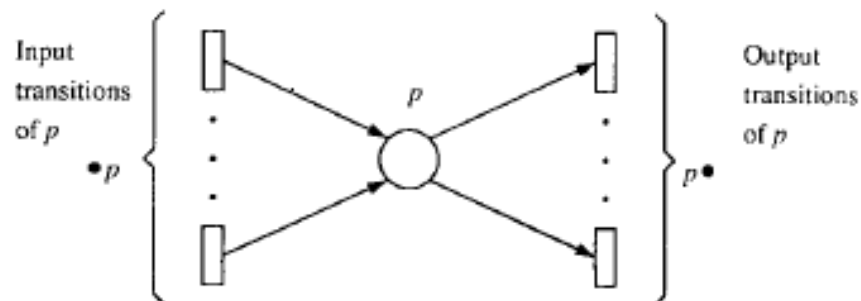
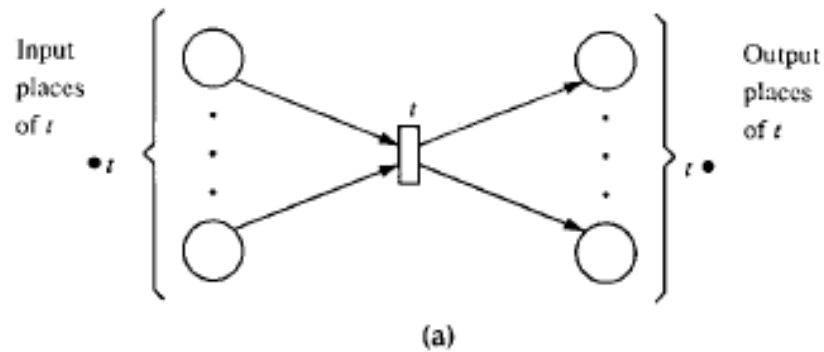
Subclasses of Petri nets

(taken from Murata's paper)



Input and output nodes

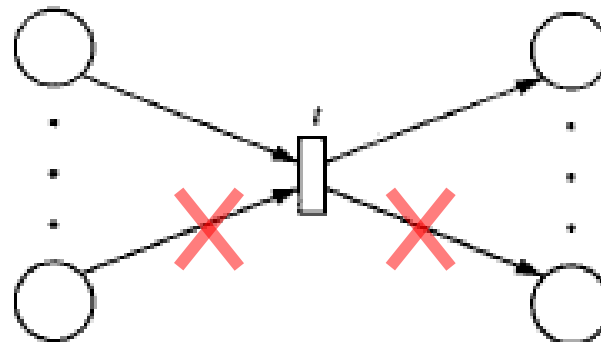
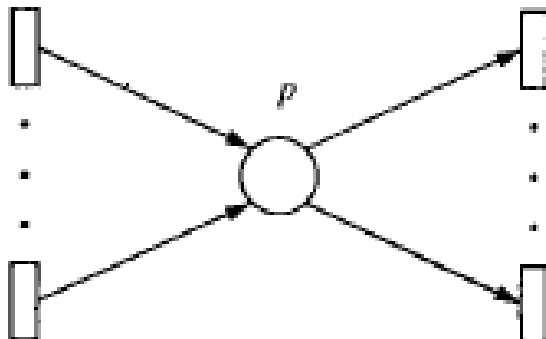
- $t = \{p | (p, t) \in F\}$ = the set of input places of t
- $t^\bullet = \{p | (t, p) \in F\}$ = the set of output places of t
- $p = \{t | (t, p) \in F\}$ = the set of input transitions of p
- $p^\bullet = \{t | (p, t) \in F\}$ = the set of output transitions of p .



State machine

1) A *state machine* (SM) is an ordinary Petri net such that each transition t has exactly one input place and exactly one output place, i.e.,

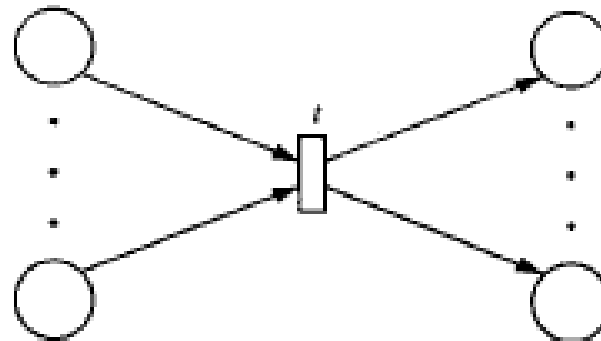
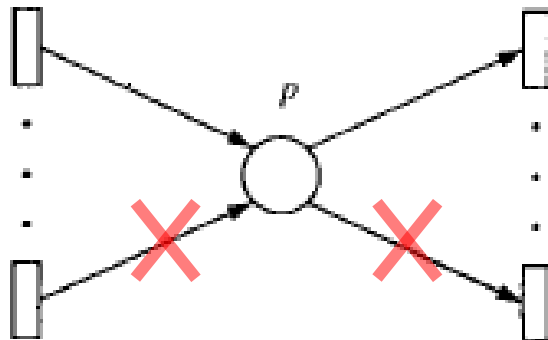
$$|\bullet t| = |t\bullet| = 1 \quad \text{for all } t \in T.$$

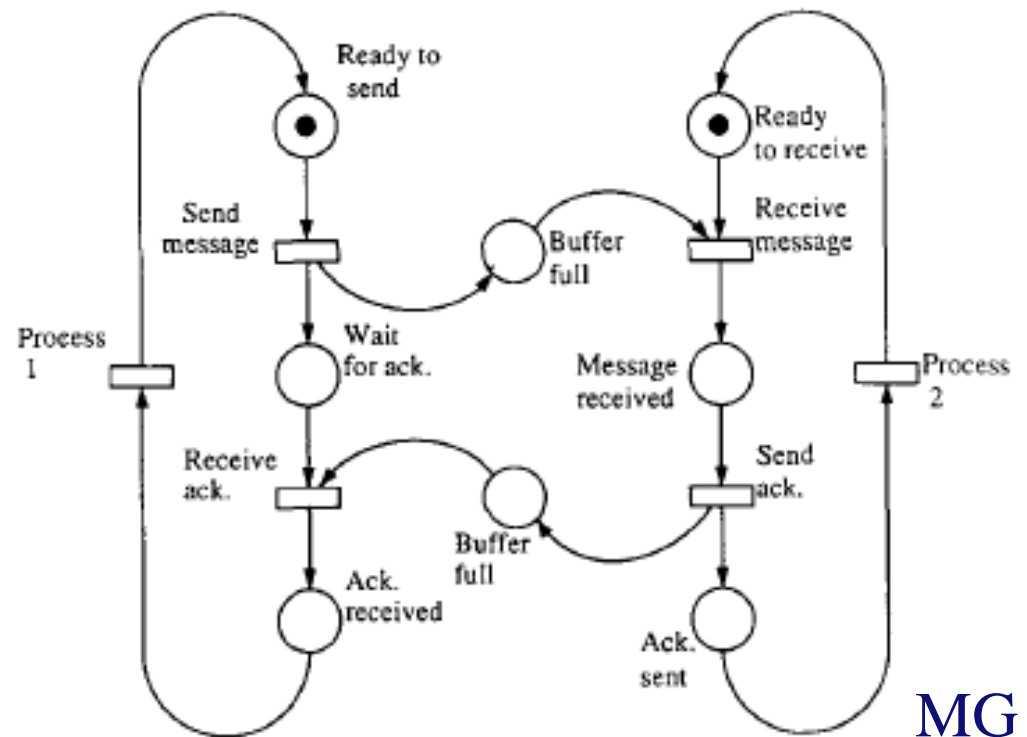
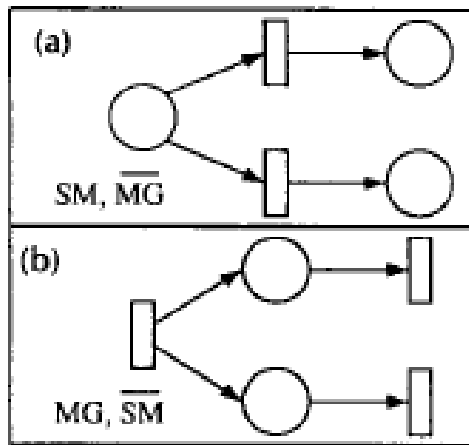


Marked graph

2) A *marked graph* (MG) is an ordinary Petri net such that each place p has exactly one input transition and exactly one output transition, i.e.,

$$|\bullet p| = |p\bullet| = 1 \quad \text{for all } p \in P.$$



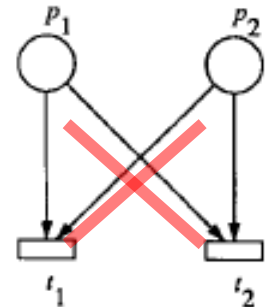
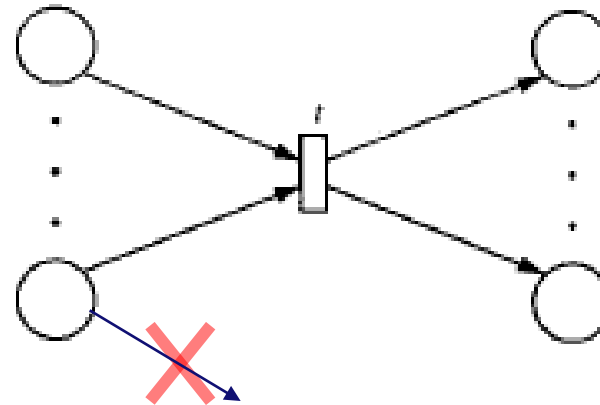
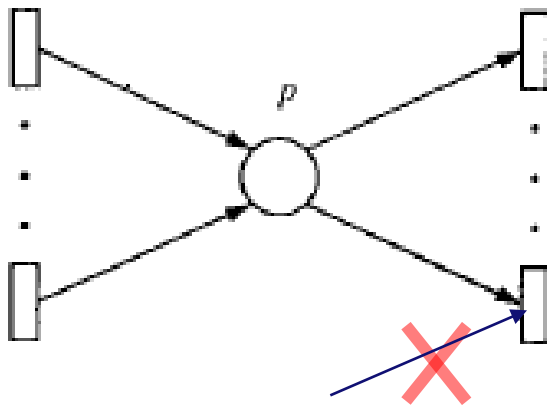


Free-choice net

3) A *free-choice net* (FC) is an ordinary Petri net such that every arc from a place is either a unique outgoing arc or a unique incoming arc to a transition, i.e.,

for all $p \in P$, $|p^\bullet| \leq 1$ or ${}^\bullet(p^\bullet) = \{p\}$; equivalently,

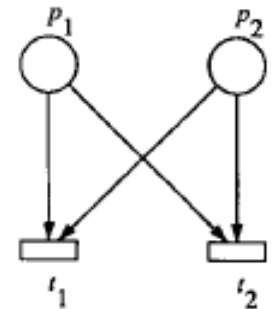
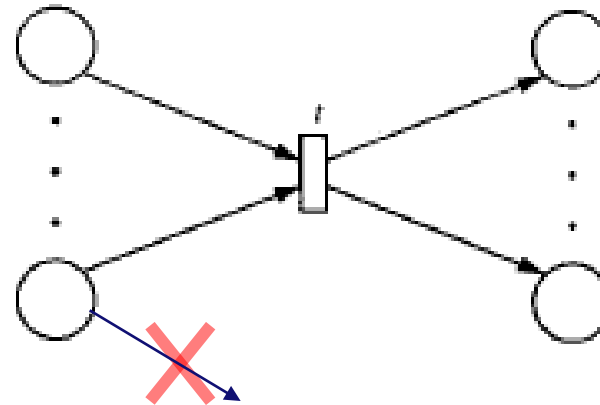
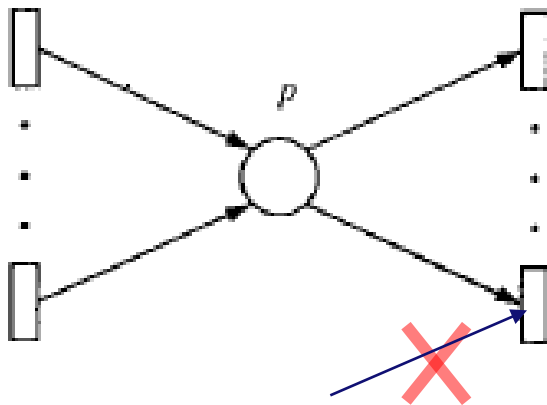
for all $p_1, p_2 \in P$, $p_1^\bullet \cap p_2^\bullet \neq \emptyset \Rightarrow |p_1^\bullet| = |p_2^\bullet| = 1$.



Extended-free-choice net

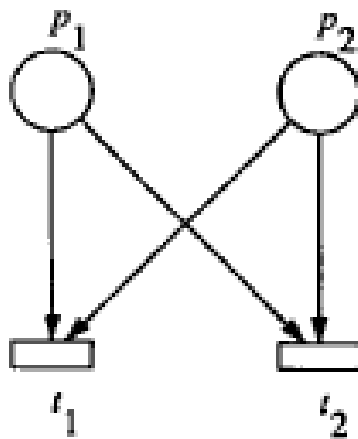
4) An *extended free-choice net* (EFC) is an ordinary Petri net such that

$$p_1^\bullet \cap p_2^\bullet \neq \emptyset \Rightarrow p_1^\bullet = p_2^\bullet \text{ for all } p_1, p_2 \in P.$$

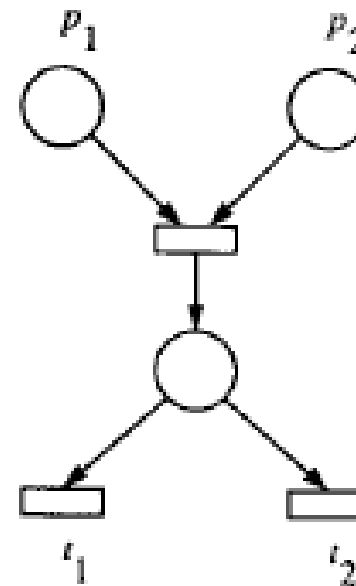


OK

Transformation



EFC



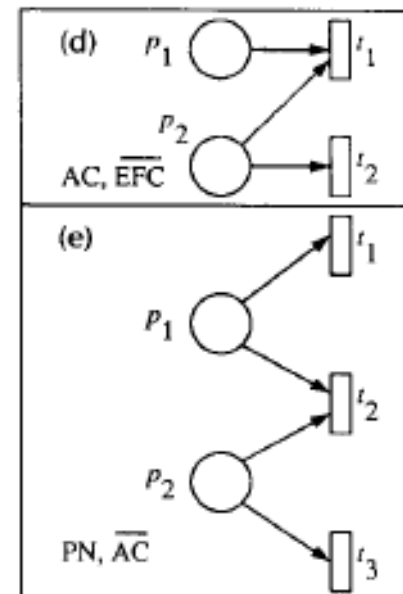
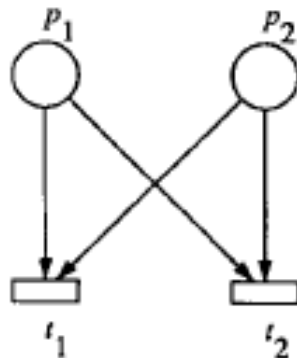
FC

Asymmetric choice net

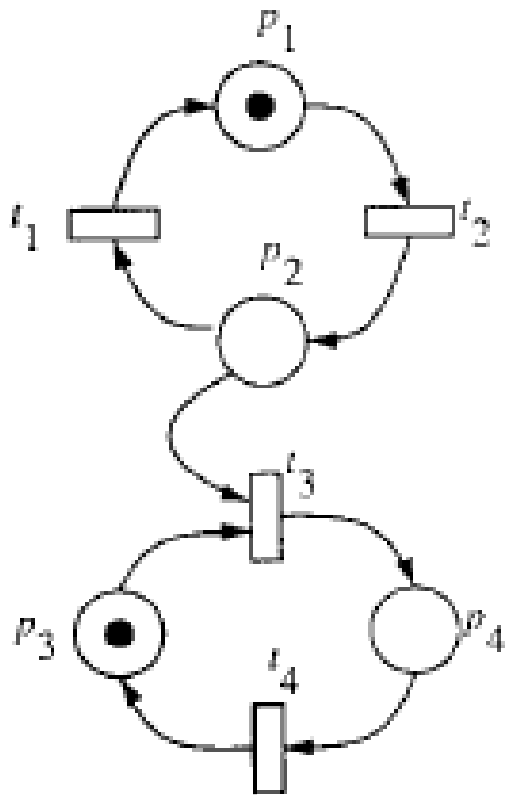
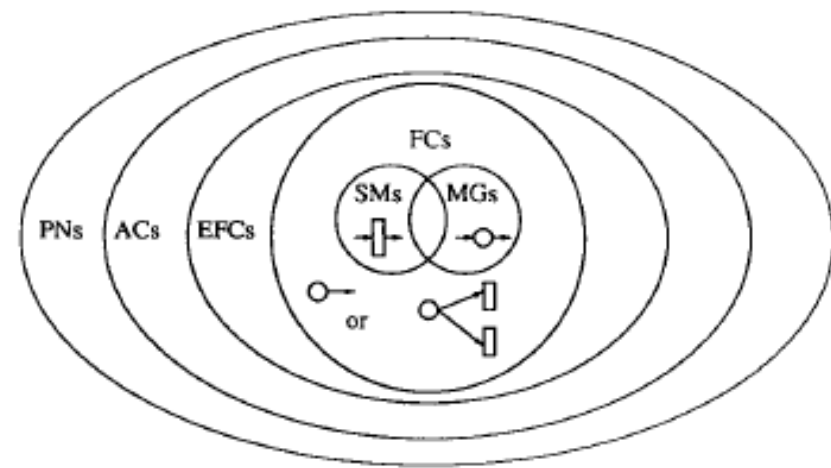
5) An *asymmetric choice net* (AC) (also known as a *simple net*) is an ordinary Petri net such that

$$p_1^\bullet \cap p_2^\bullet \neq \emptyset \Rightarrow p_1^\bullet \subseteq p_2^\bullet \text{ or } p_1^\bullet \supseteq p_2^\bullet$$

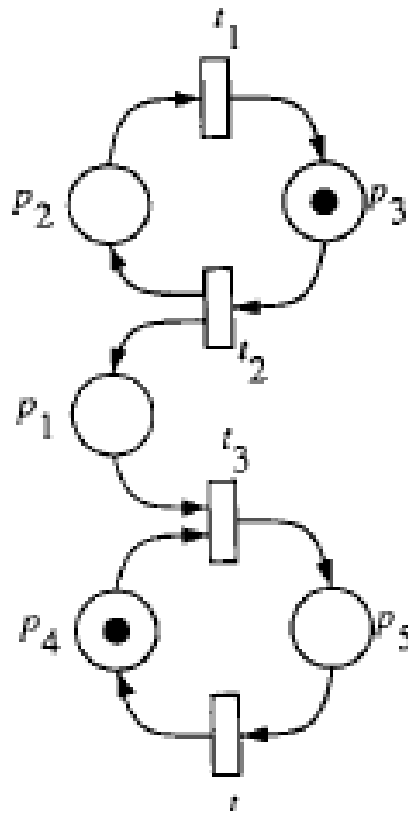
for all $p_1, p_2 \in P$.



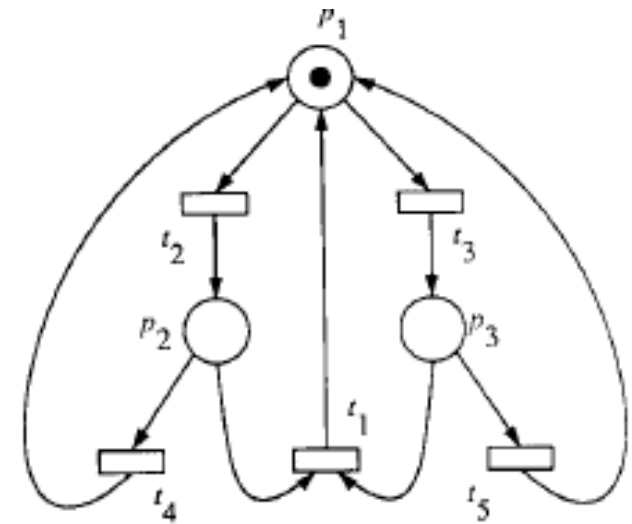
Which class?



AC

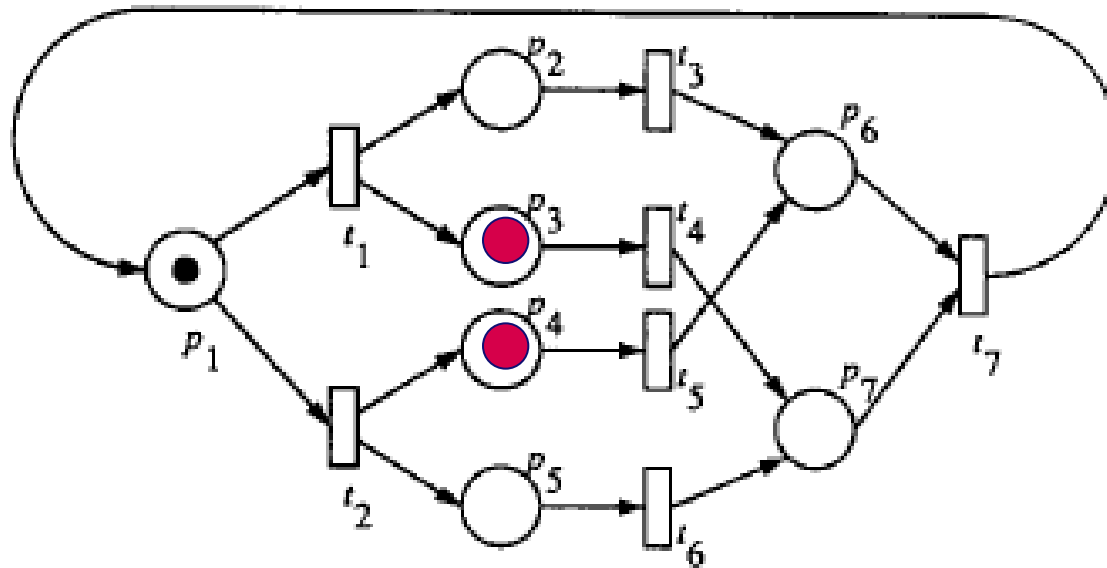


MG



not AC

Which class and which properties?



FC, live, bounded, safe, reversible

Give a live, bounded, safe, and non-reversible initial marking.

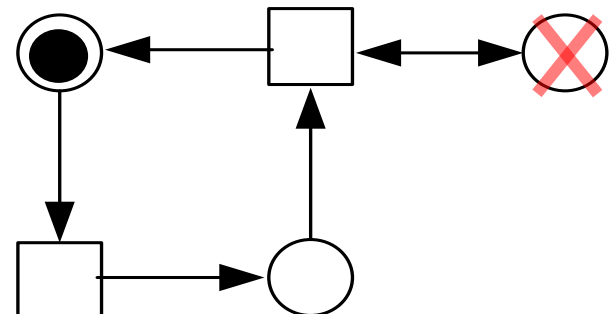
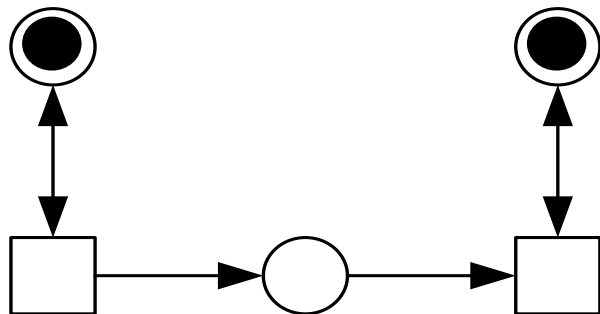
Obvious properties for state machines

Theorem 4: A state machine (N, M_0) is live iff N is strongly connected and M_0 has at least one token. \square

Theorem 5: A state machine (N, M_0) is safe iff M_0 has at most one token. A live state machine (N, M_0) is safe iff M_0 has exactly one token. \square

Marked graphs

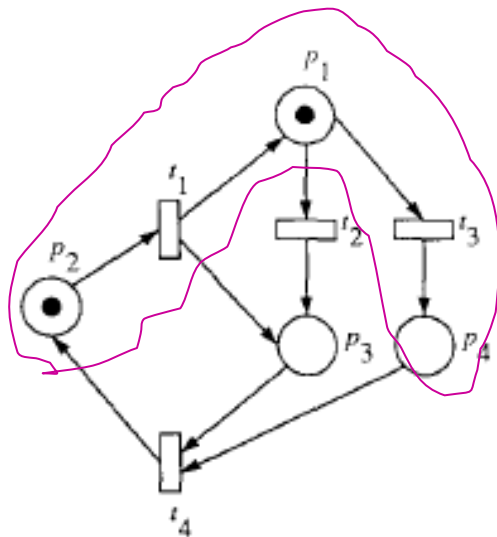
Theorem 6: For a marked graph, the token count in a directed circuit is invariant under any firing, i.e., $M(C) = M_0(C)$ for each directed circuit C and for any M in $R(M_0)$, where $M(C)$ denotes the total number of tokens on C . \square



Theorem 7: A marked graph (G, M_0) is live iff M_0 places at least one token on each directed circuit in G . \square

Free-choice nets

Theorem 12: A free-choice net (N, M_0) is live iff every siphon in N contains a marked trap. \square



Not live since the siphon $\{p_1, p_2, p_4\}$ contains no (marked) traps.

Recall that for any Petri net: If every proper siphon of a system includes an initially marked trap, then the system is deadlock free.

(Extended) free-choice nets

- Liveness *and* boundedness can be decided in polynomial time!!!!
- Uses (outside scope):

Theorem 9 Rank Theorem

An EFC-net is well-formed if and only if

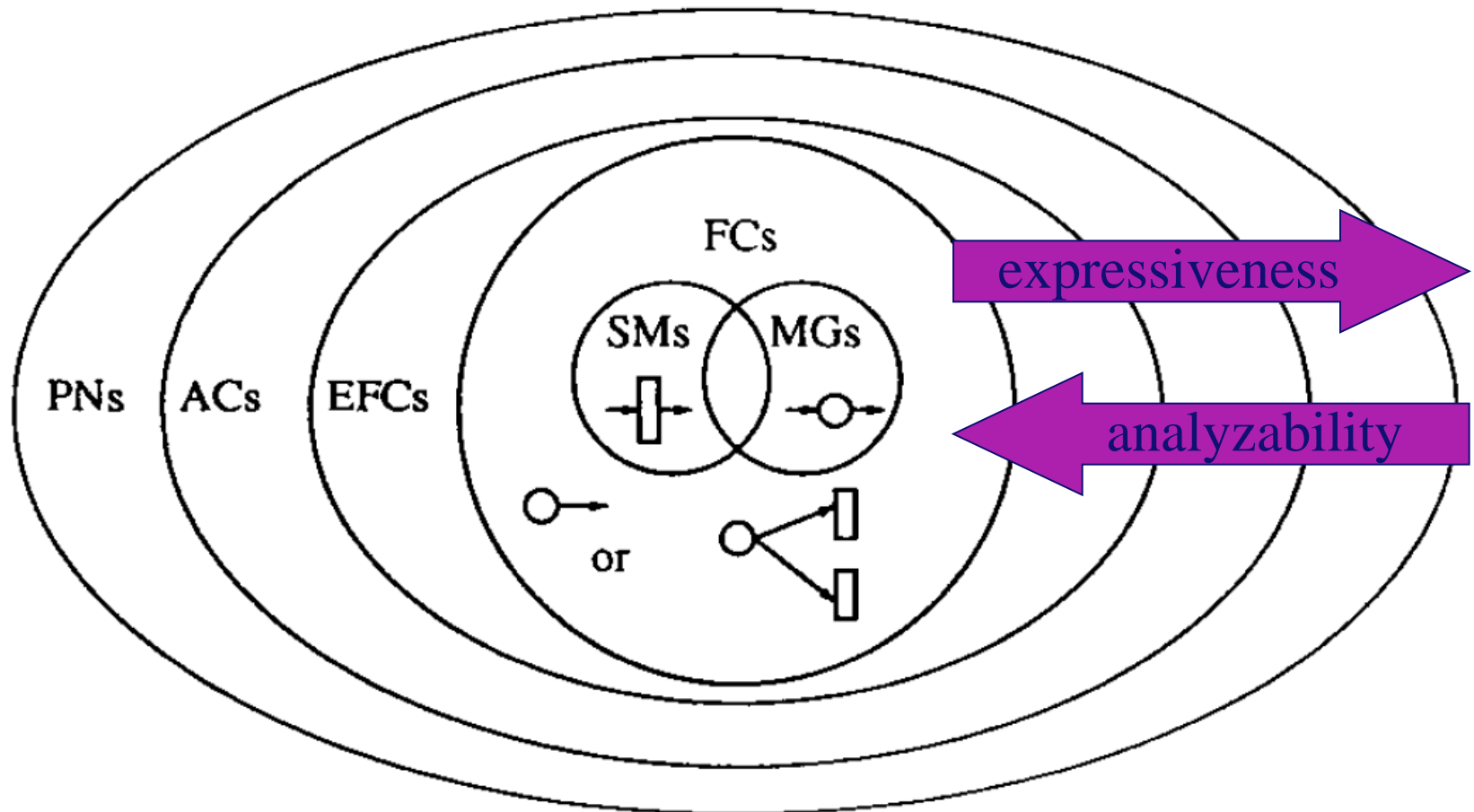
- (1) *it has a positive S -invariant,*
- (2) *it has a positive T -invariant,*
- (3) *the number of its clusters exceeds the rank of its incidence matrix by one.*

Theorem 26

The following problems are polynomially decidable:

- (1) *Given an EFC-net N , to decide if it is well-formed.*
- (2) *Given a marking M of N , to decide if it is live and bounded.*

Conclusion



Introduction to Process Mining



TU/e

Technische Universiteit
Eindhoven
University of Technology

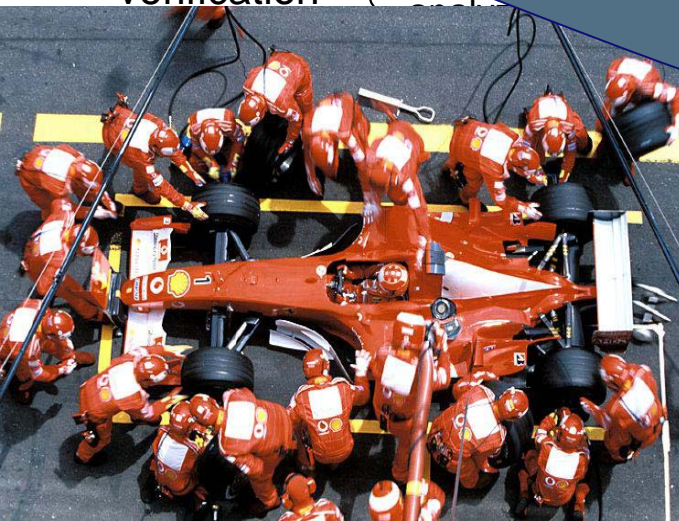
Where innovation starts

Role of models

“world”

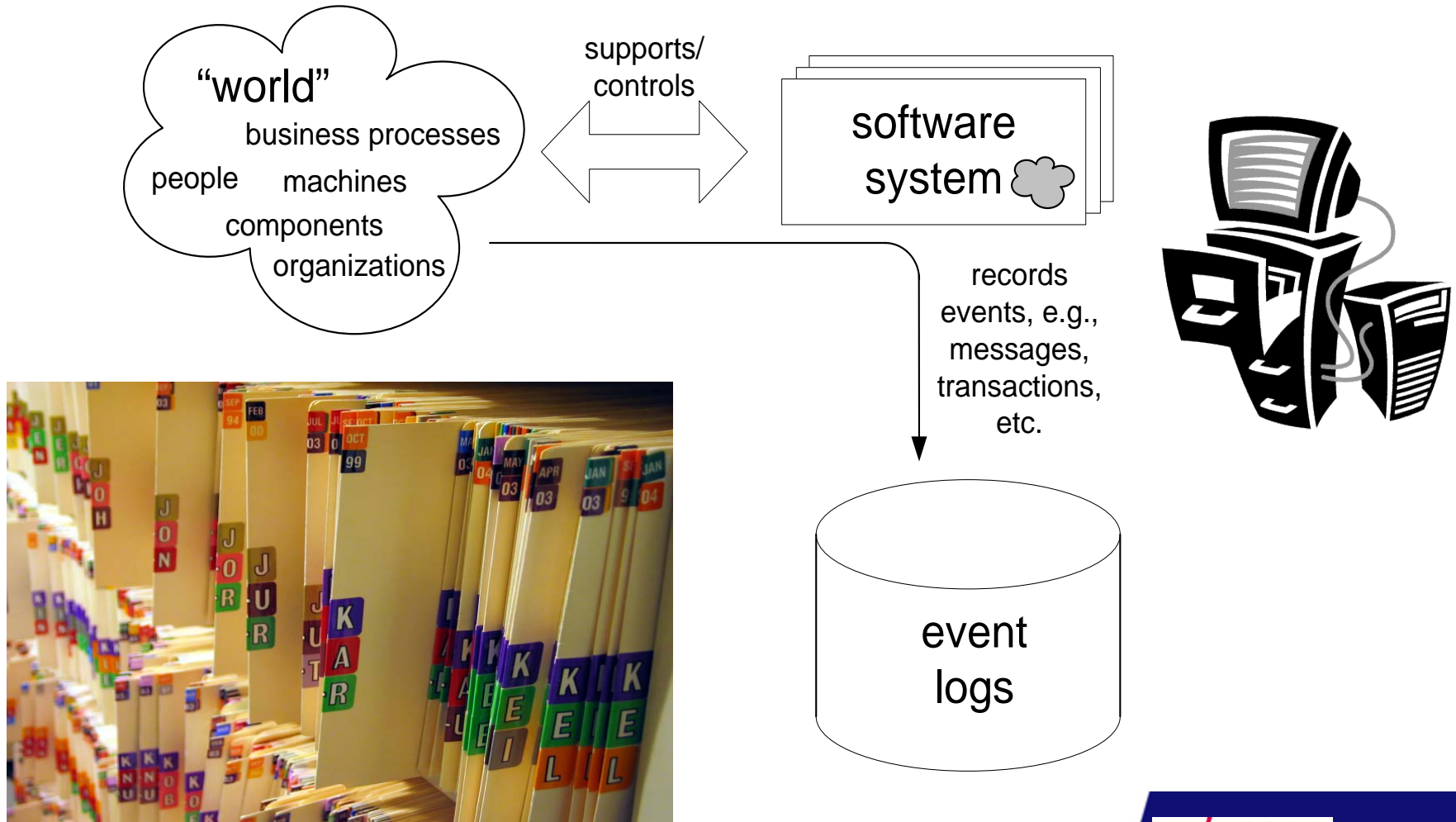
business process
people machines
components
organizations

verification



real world
powerpoint reality

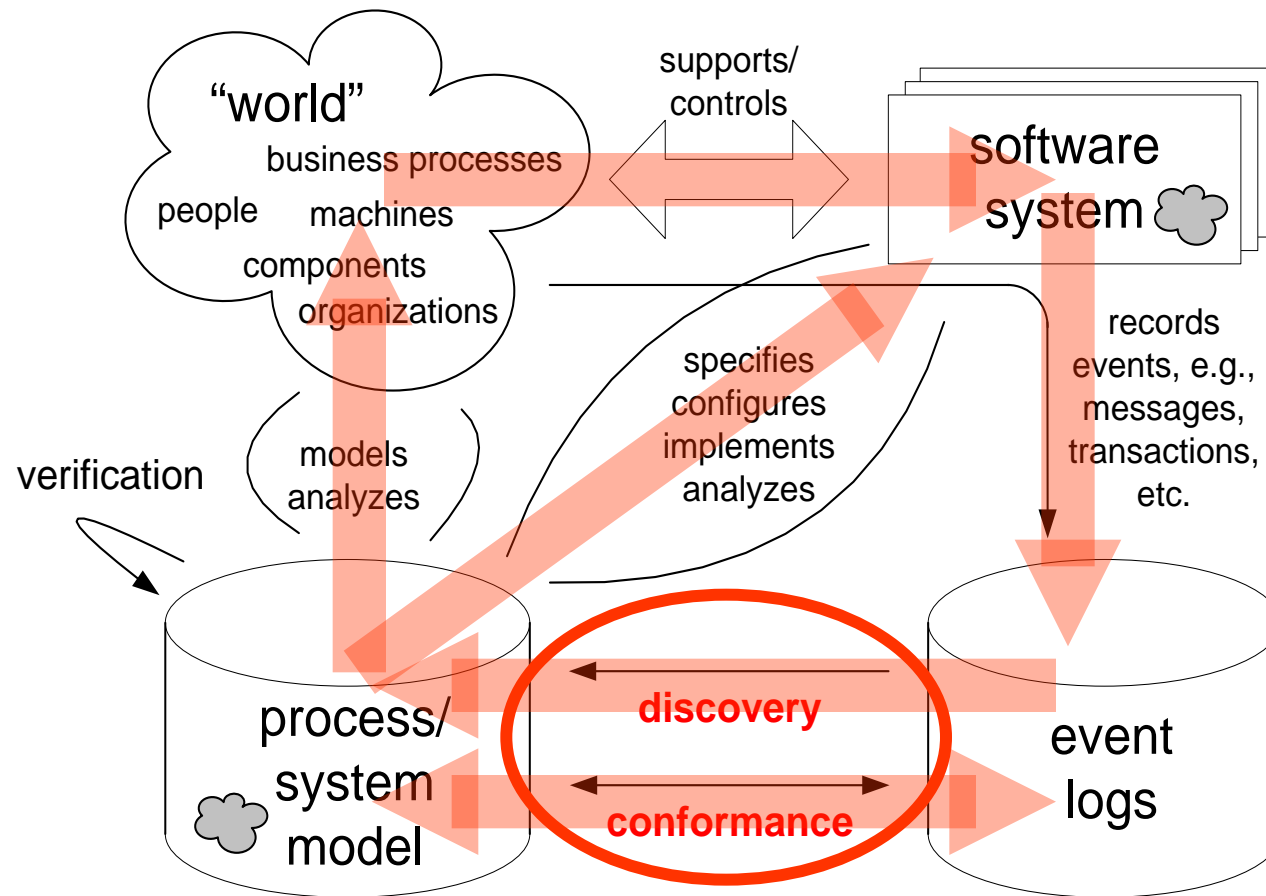
Event logs are a reflection of reality



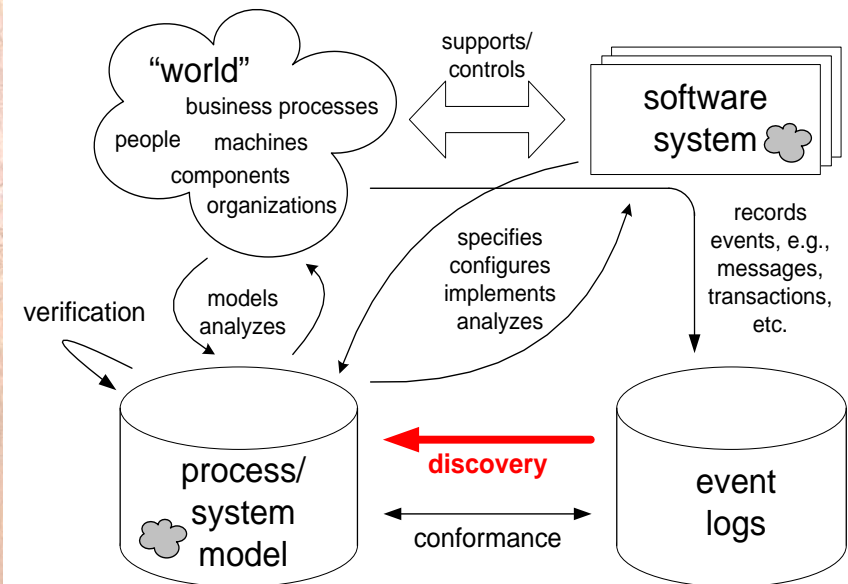
Examples:



Process mining: Linking events to models



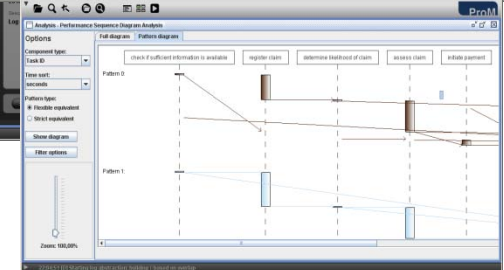
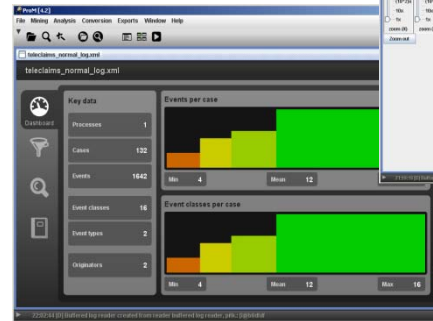
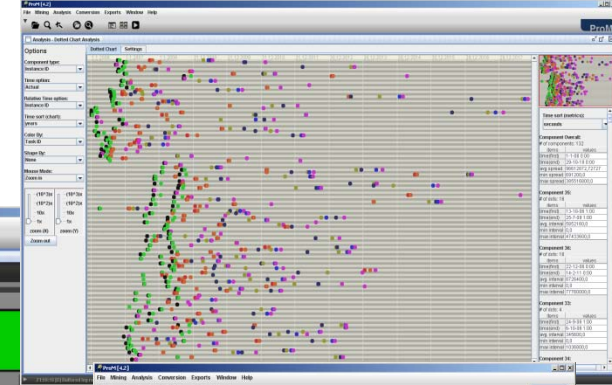
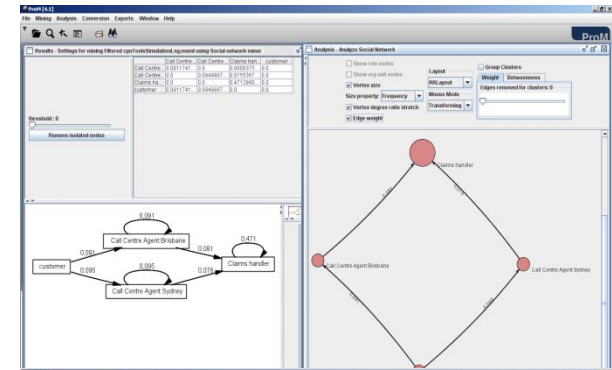
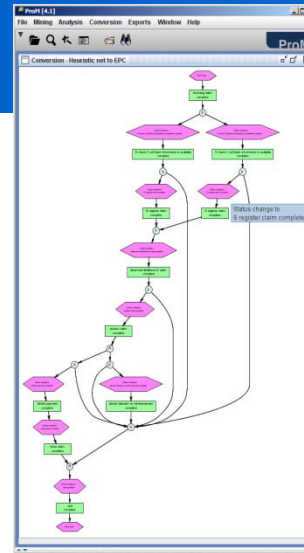
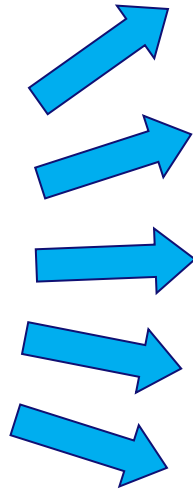
Discovery



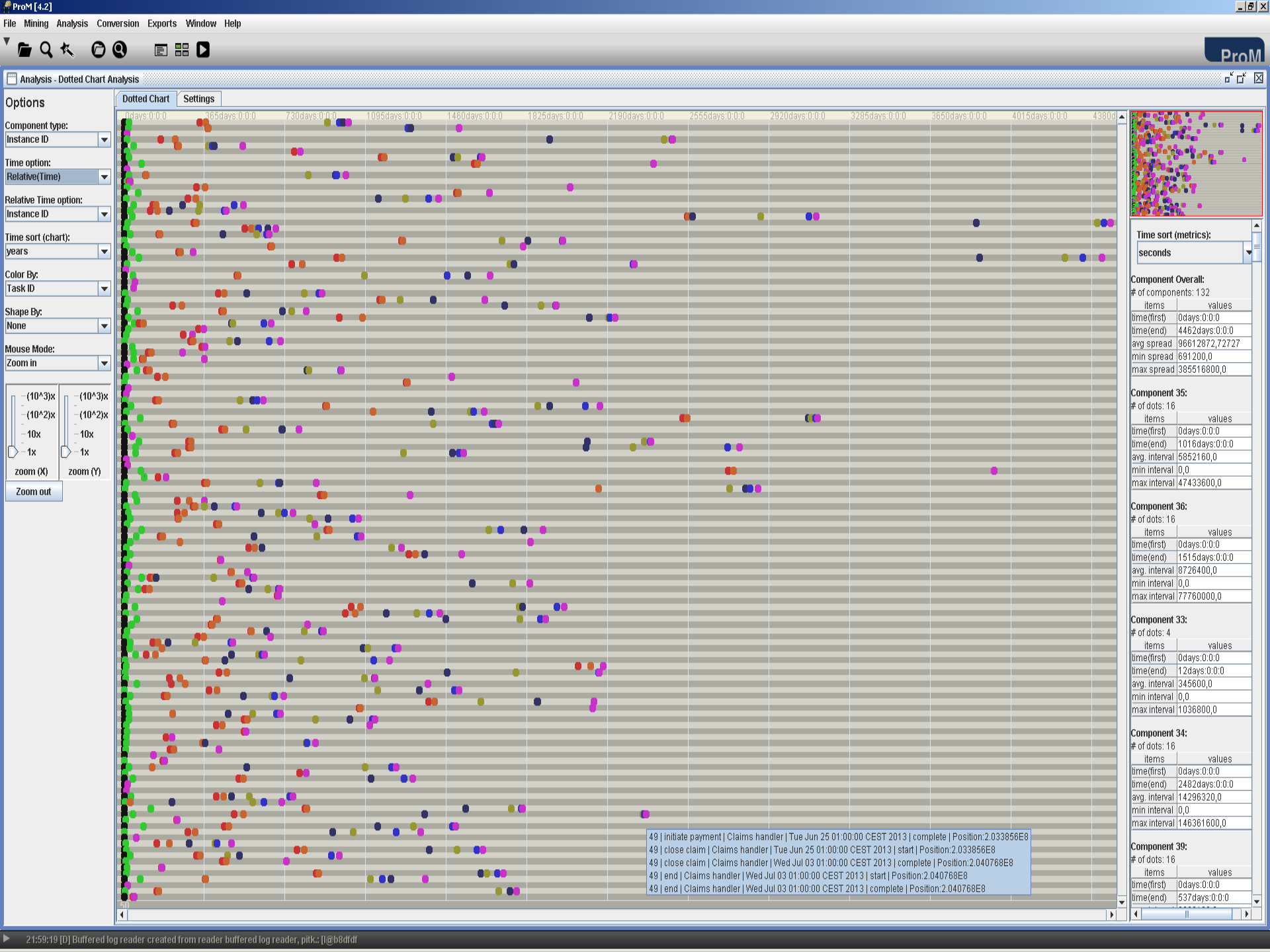


MXML Log

- **instances: 3512**
- **audit trail entries: 46138**



ProM supports +40 types of model discovery!




```

graph LR
    Start(( )) --> A[incoming claim complete]
    A --> B(( ))
    B --> C[B check if sufficient information is available complete]
    B --> D[S check if sufficient information is available complete]
    C --> E(( ))
    D --> F(( ))
    E --> G[B register claim complete]
    F --> H[S register claim complete]
    G --> I(( ))
    H --> I
    I --> J[determine likelihood of claim complete]
    J --> K(( ))
    K --> L[assess claim complete]
    L --> M(( ))
    M --> A
  
```

	3512
--	------

```
graph TD
    Start([Start]) --> Recv[receiving claim complete]
    Recv --> X1((X))
    X1 --> D1{Status change to  
S check if sufficient information is available complete}
    X1 --> D2{Status change to  
S check if sufficient information is available complete}
    D1 --> C1[S check if sufficient information is available complete]
    D2 --> C2[S check if sufficient information is available complete]
    C1 --> X2((X))
    C2 --> X3((X))
    X2 --> D3{Status change to  
S register claim complete}
    X3 --> D4{Status change to  
S register claim complete}
    D3 --> C3[S register claim complete]
    D4 --> C4[S register claim complete]
    C3 --> X4((X))
    C4 --> X4
    X4 --> D5{Status change to  
determine likelihood of claim complete}
    D5 --> End([determine likelihood of claim])
```


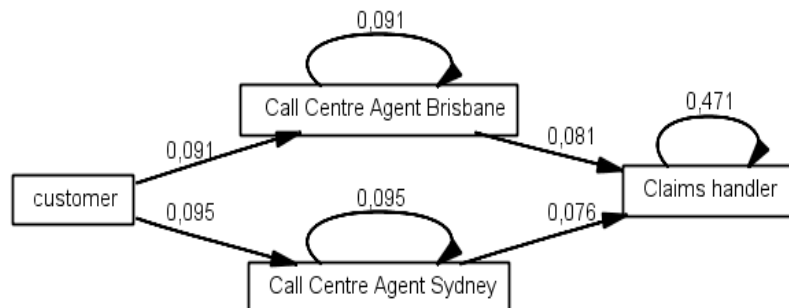
```
graph LR; Start(( )) -- "[default]" --> EndTask[end complete Task]; Start -- "true [0]" --> AdviseTask[advise claimant on reimbursement complete Task]; AdviseTask -- "true [0]" --> EndTask; AdviseTask -- "[default]" --> InitTask[initiate payment complete Task]; InitTask -- "[default]" --> EndTask; InitTask -- "[default]" --> CloseTask[close claim complete Task]; CloseTask -- "[default]" --> EndTask; EndTask -- "[default]" --> Start; EndTask -- "true [0]" --> AdviseTask; EndTask --> Complete[complete]; Complete --> Released{{Released}};
```



Results - Settings for mining Filtered cpnToolsSimulationLog.mxml using Social network miner

	Call Centre...	Call Centre...	Claims han...	customer
Call Centre...	0.0911741...	0.0	0.0808375...	0.0
Call Centre...	0.0	0.0949907...	0.0755367...	0.0
Claims ha...	0.0	0.0	0.4712960...	0.0
customer	0.0911741...	0.0949907...	0.0	0.0

threshold : 0


Remove isolated nodes

Analysis - Analyze Social Network

☐ Show role nodes☐ Show org unit nodes☒ Vertex size

Size property: Frequency

☒ Vertex degree ratio stretch☒ Edge weight

Layout

KKLayout

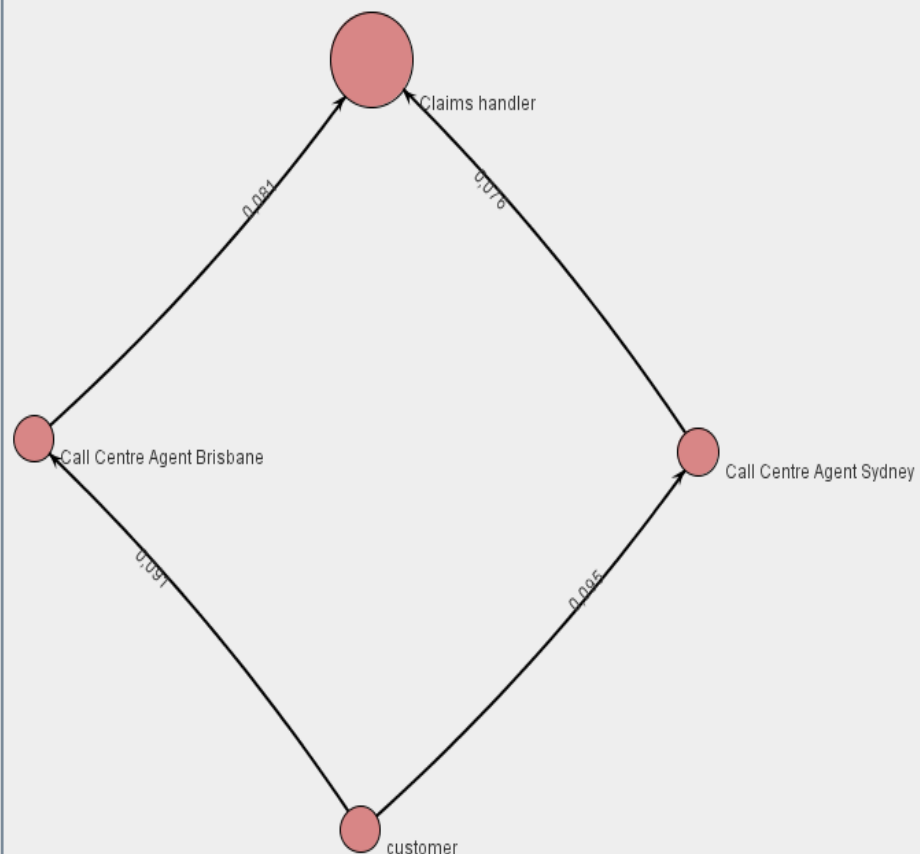
Mouse Mode

Transforming

☐ Group Clusters

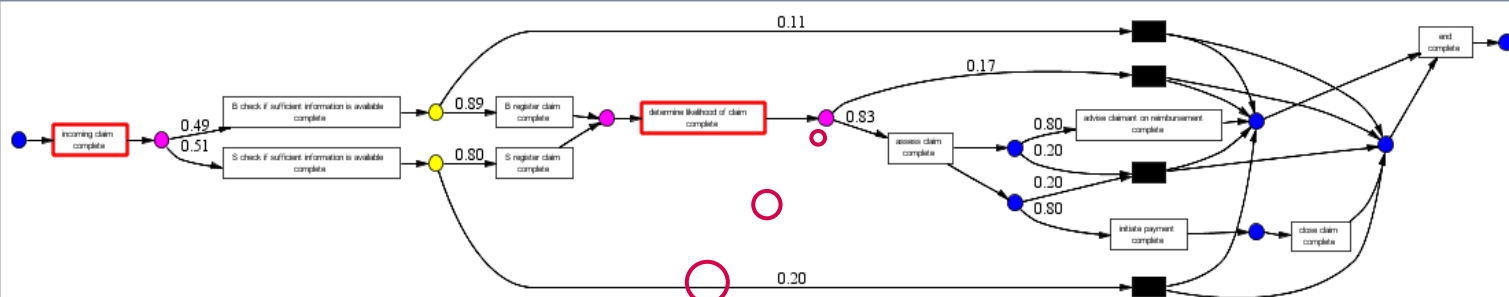
Weight Betweenness

Edges removed for clusters: 0





Analysis - Performance Analysis with Petri net



bottle-necks

flow time from A to B

throughput time

Process information:

Total number selected:

3512 cases

Number fitting:

3512 cases

Arrival rate:

0,12 cases per second

	Throughput time (seconds)
avg	11115,54
min	0,0
max	40704,0
stdev	8906,98
fast 25...	1379,19
slow 2...	23817,24
norma...	9632,87

Change Percentages Export Time-Metrics

Performance information of the selected transitions:

Frequency: 2950 cases

	Time in between (seconds)
avg	12248,87
min	53,0
max	39706,0
stdev	8381,14

Waiting time:

High Medium Low

Settings

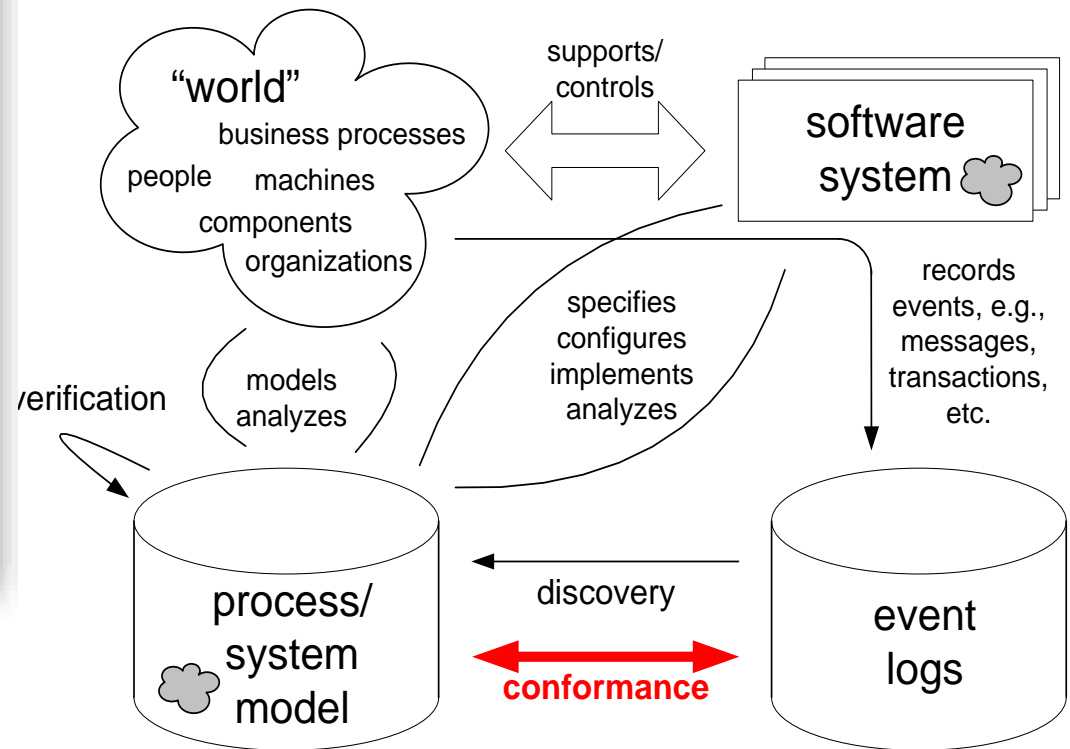
Selected:

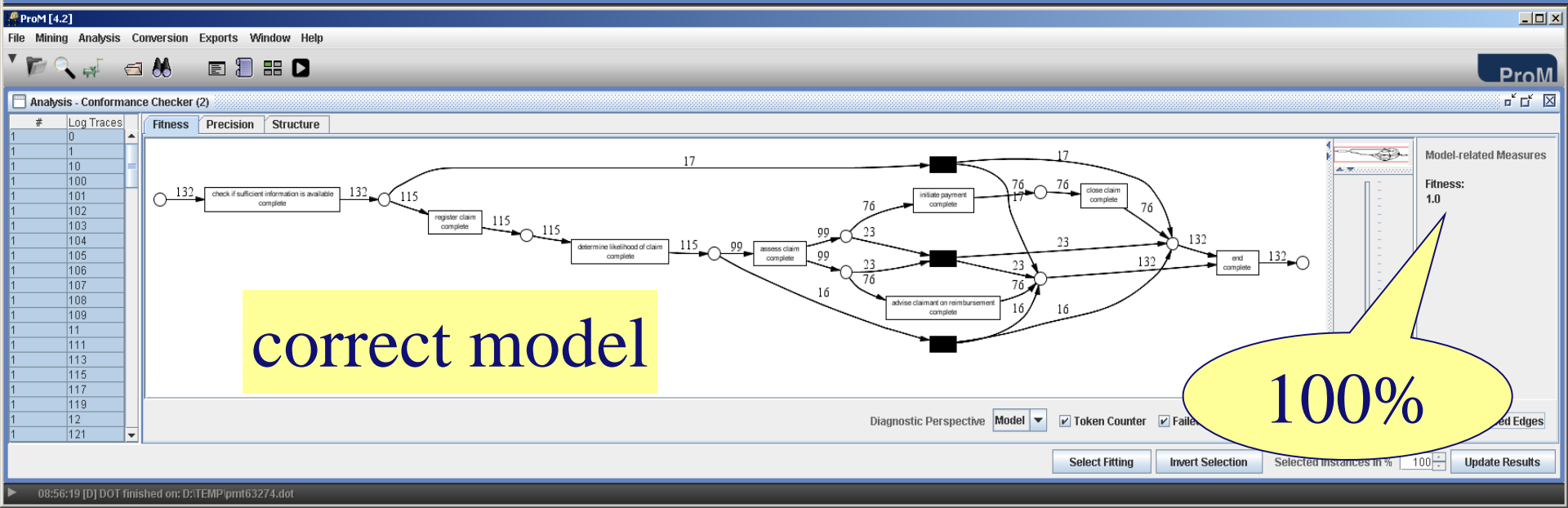
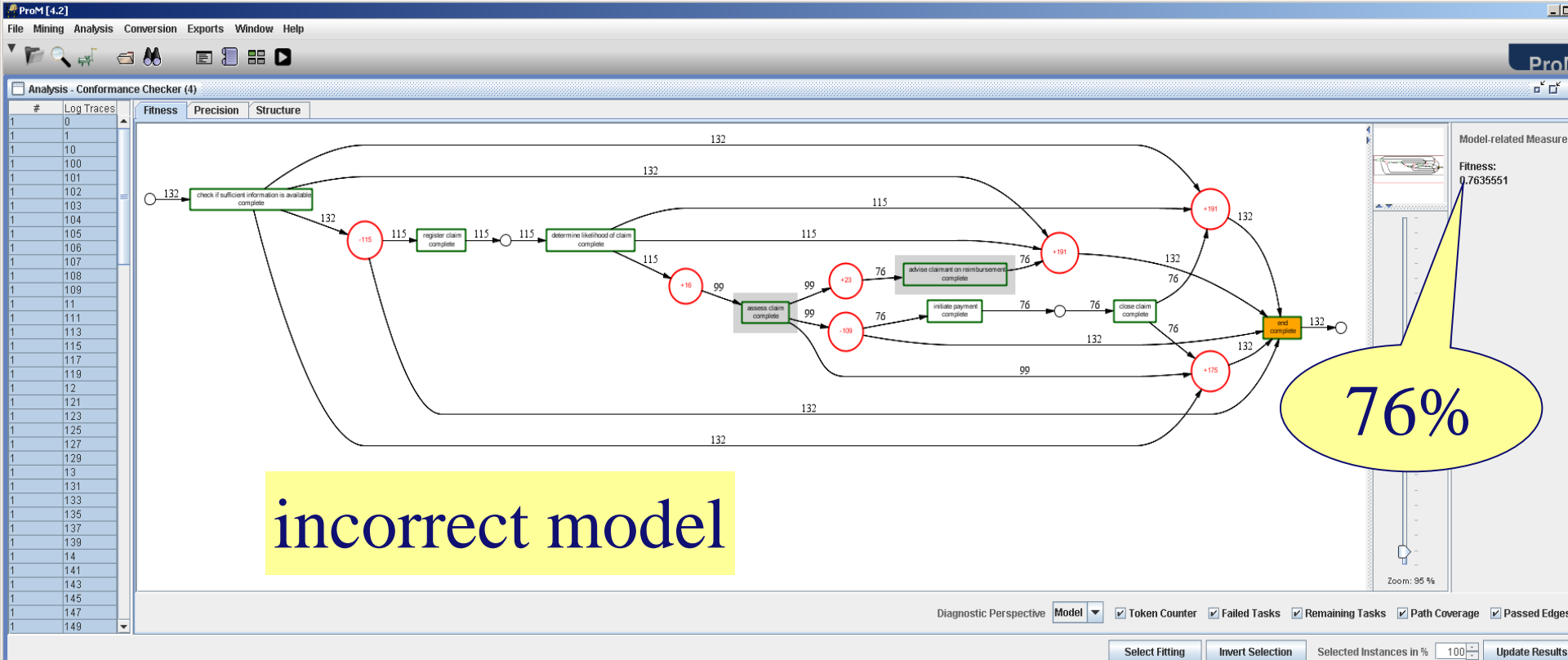
Transition - incoming claim c...

and:

Transition - determine likeliho...

Conformance Checking



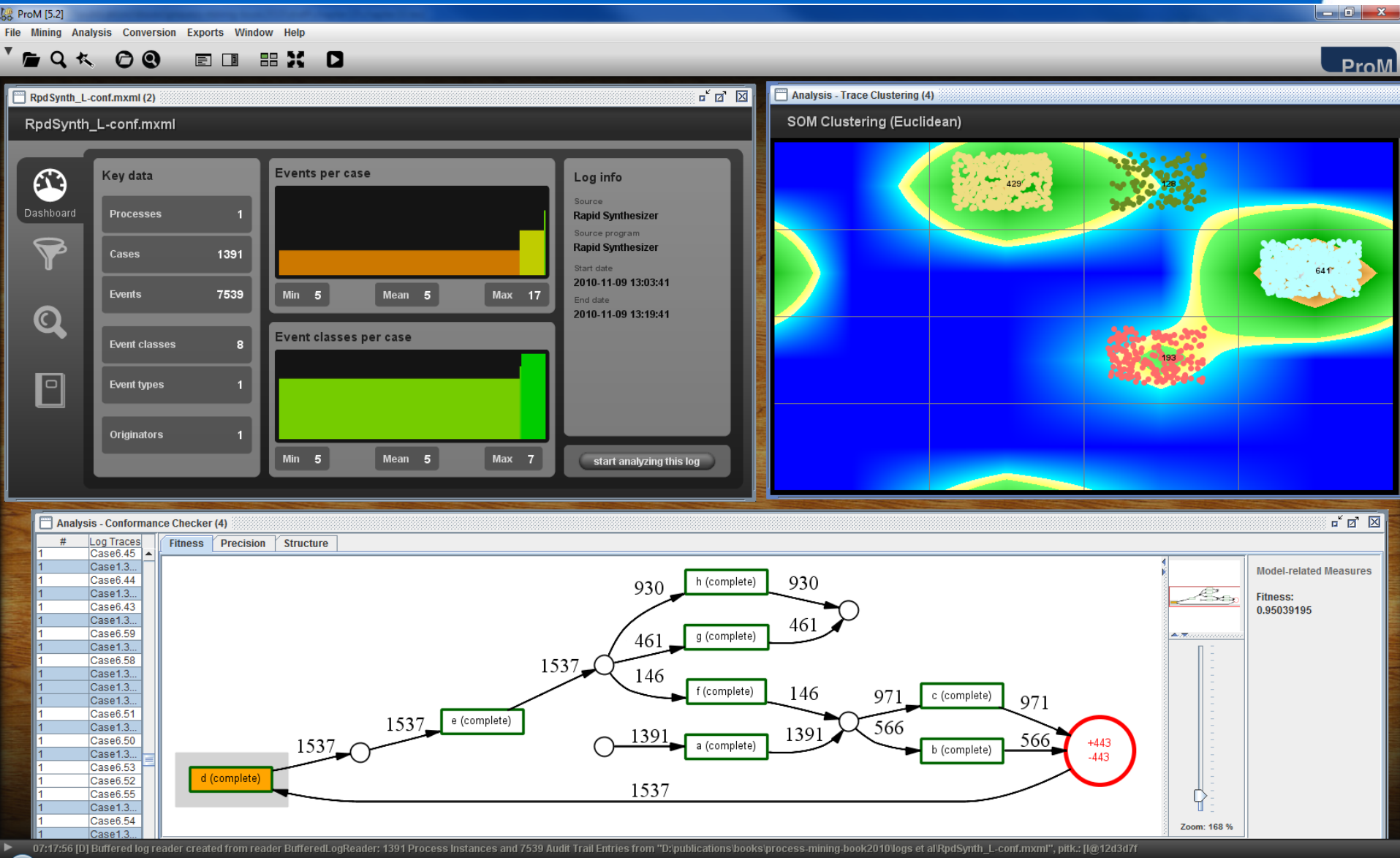


ProM

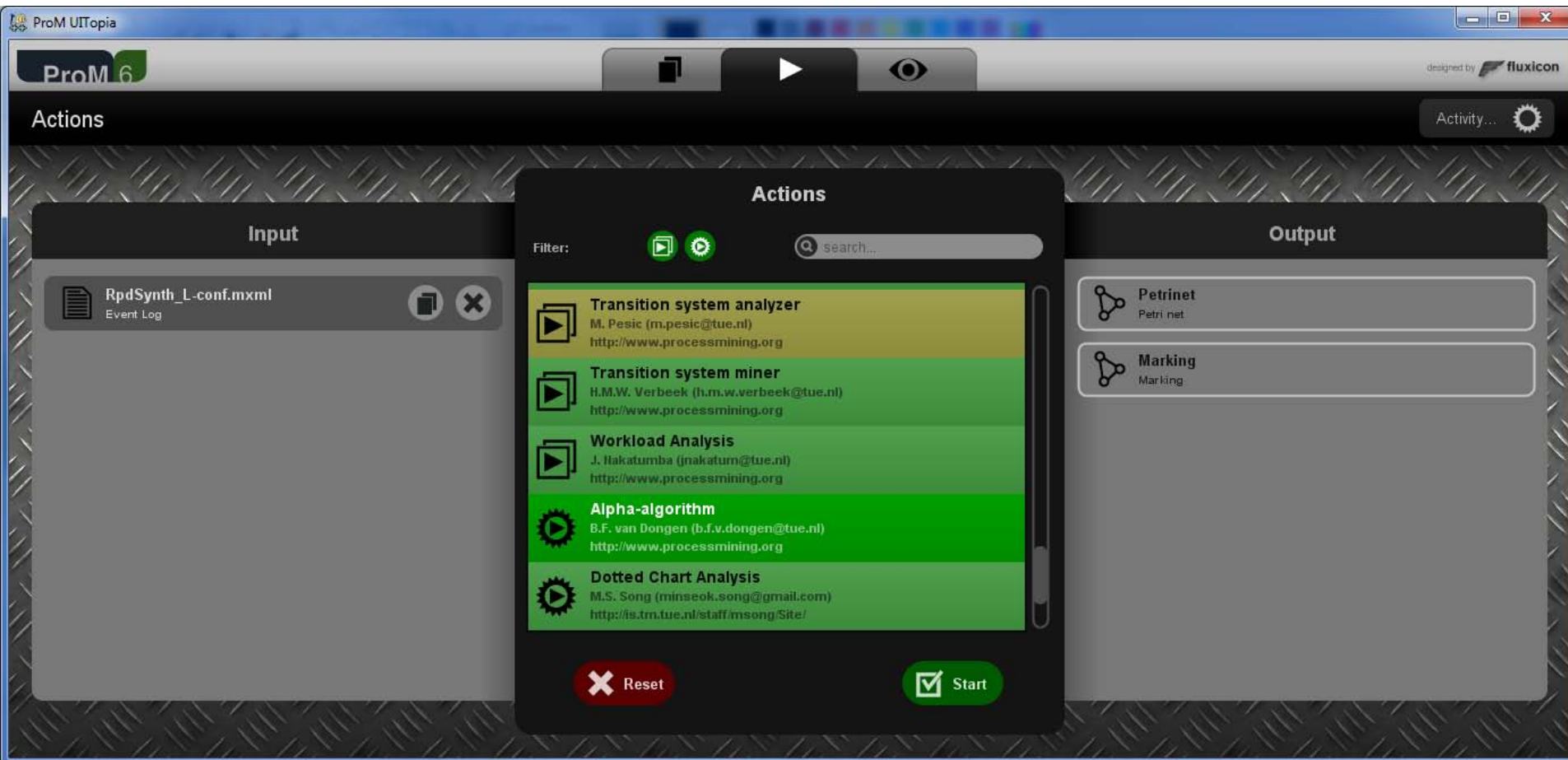
- www.processmining.org
- ProM supports all of the techniques mentioned in book and on slides!
- Pluggable architecture.
- Major differences between ProM 5.2 (and earlier) and ProM 6.



Screenshot of ProM 5.2

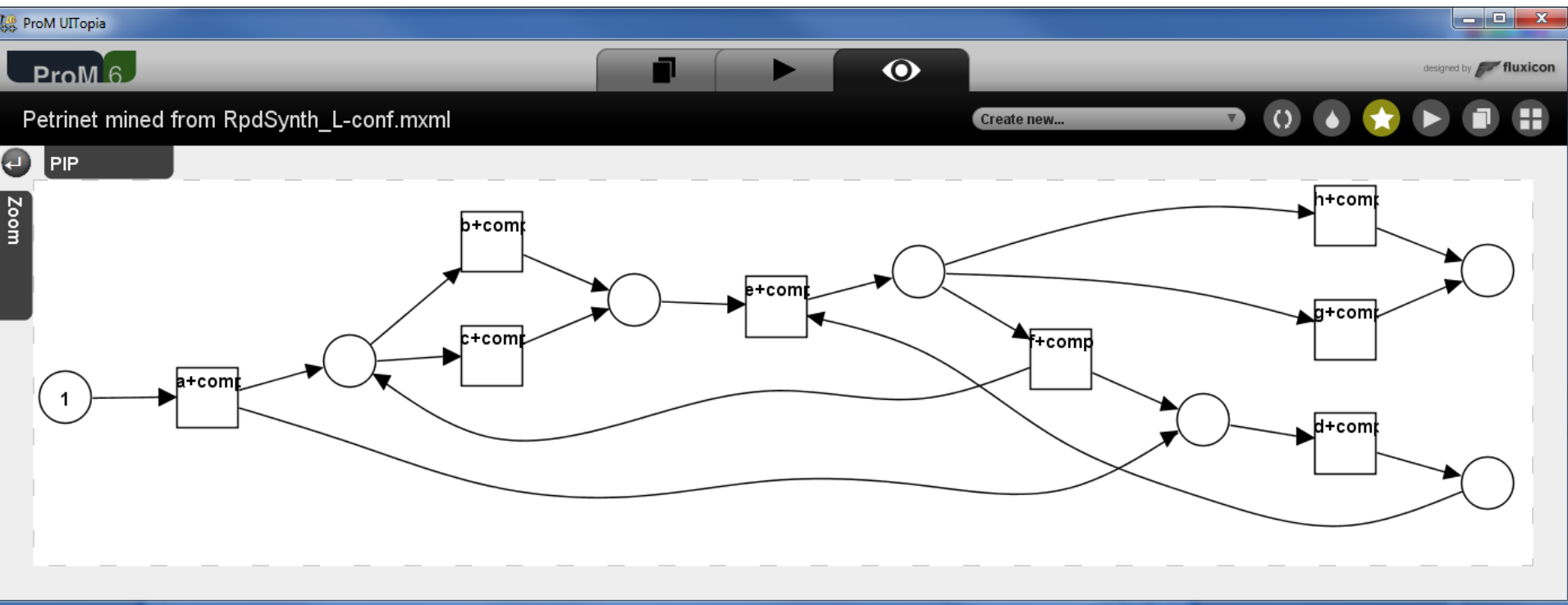


Screenshot of ProM 6

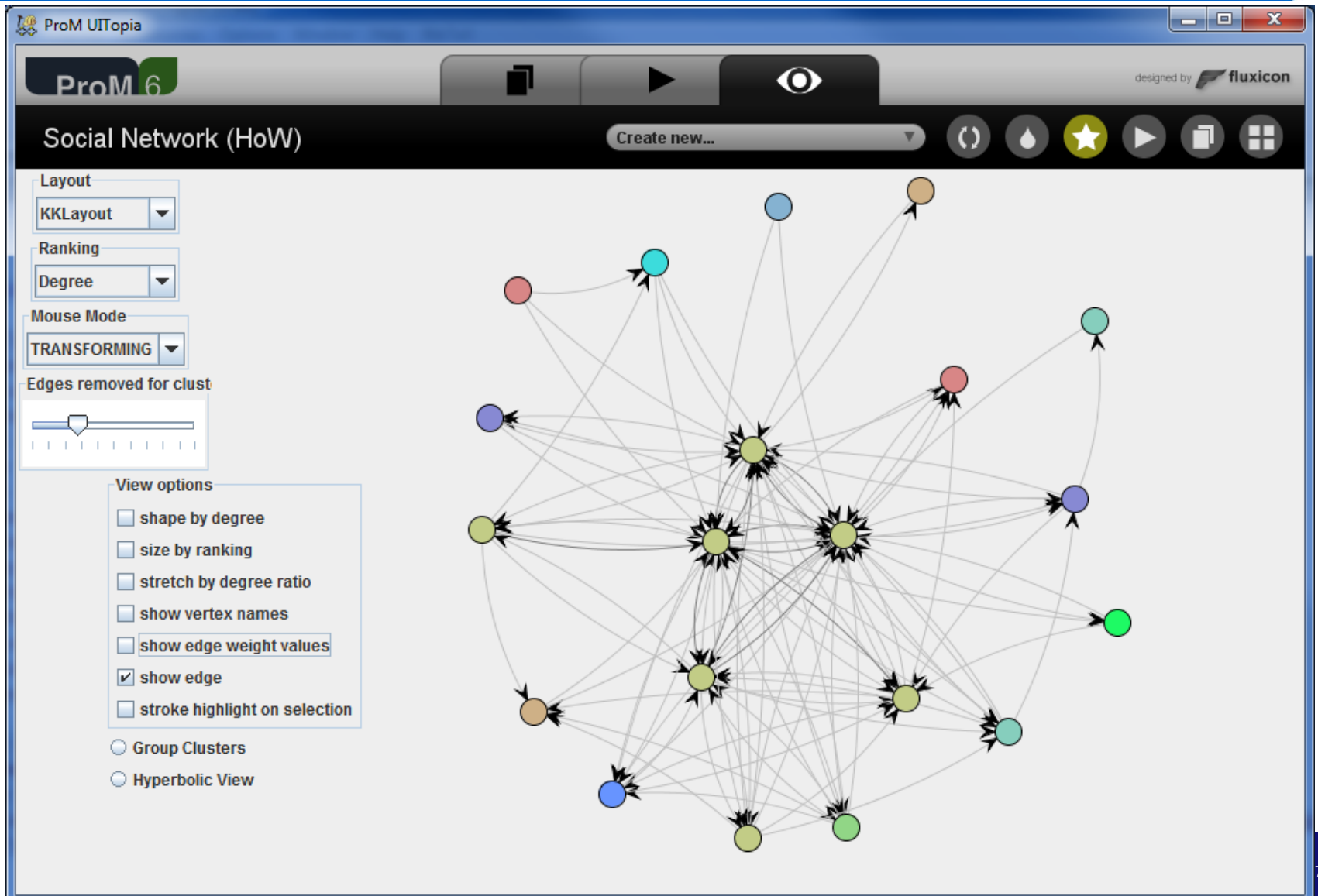


(based on handover of work)

ProM 6: α miner



ProM 6: Social network analyzer



Example plug-ins in ProM 6

(see book and website)

plug-in	description
Alpha miner	discovers a Petri net using the α -algorithm, see Section 5.2
Heuristic miner	discovers a C-net using heuristic mining, see Section 6.2
Genetic miner	discovers a C-net using genetic mining, see Section 6.3
Fuzzy miner	discovers a fuzzy model using fuzzy mining, see Section 13.1.3 and [72]
Transition system miner	discovers a transition system based on a state representation function and a log, see Section 6.4.1
Transition system to Petri net	uses state-based regions to create a Petri net based on a transition system, see Section 6.4.2
Declare miner	discovers a Declare model, see Section 7.3
ILP miner	discovers a Petri net using language-based regions, see Section 6.4.3
Simple log filter	filtering a log by answering simple questions, see Fig. 12.6(b)
Dotted chart analysis	creates a dotted chart showing all events at a glance, see Section 8.2
Trace alignment	similar to dotted chart, but now events are aligned based on their context rather than time [37]
Guide tree miner	clusters cases in a tree based on similarities [36]
Social network miner	creates a social network based on a selected criterion, see Fig. 10.6
LTL checker	checks a property expressed in terms of LTL [6]
Fitness	computes fitness of Petri net based on event log
ETConformance	checks conformance by counting “escaping edges” from the state space of the log to the state space of the model [100]
Replay log on flexible model	conformance checker based on A* algorithm [25]; can also be applied to Petri nets, C-nets and YAWL models
PomPom	automatically abstracts from infrequently visited parts of a Petri net, see also Section 13.1.3 showing the same idea using fuzzy models
Transition system analyzer	creates a model to predict the remaining flow time, see Section 9.4 and [17, 21]

Some process mining tools

product name	type	organization
ARIS Process Performance Manager	C	Software AG (www.softwareag.com)
Enterprise Visualization Suite	C	Businesscape (www.businesscape.no)
Disco	C	Fluxicon (www.fluxicon.com)
Genet/Petrify	A	Universitat Politècnica de Catalunya (www.lsi.upc.edu)
Interstage BPME	C	Fujitsu (www.fujitsu.com)
OKT Process Mining suite	O	Exeura s.r.l. (www.exeura.com)
Process Discovery Focus	C	Iontas (Verint Systems) (www.iontas.com)
ProcessAnalyzer	C	QPR (www.qpr.com)
ProM	O	process mining group (managed by the AIS group at TU/e) (www.processmining.org)
Rbminer/Dbminer	A	Universitat Politècnica de Catalunya (www.lsi.upc.edu)
Reflect one	C	Pallas Athena (www.pallas-athena.com)
Reflect	C	Futura Process Intelligence (www.futuratech.nl)
ServiceMosaic	A	University of New South Wales (soc.cse.unsw.edu.au)

Futura Reflect (process view) (also embedded in BPM|one)

Futura Reflect™
Process Intelligence

Home About Help Log out Reflect Demo, Futura

Appeal Process Animate

Switch Save in repository Download

Actions

- Overview
- Mine
- Explore
- Animate
- Charting

Objects

- Models (8)
- Animations (3)
- MFP - Discard 99% (#12)
- Process animation
- Social network animation

Import animation

Charts (8)

Dashboards (0)

Process model - 90%

Number of cases

Case Type	Count
Granted building permit	10
Rejected building permit	2
Human resources	1
Tree cutting permit	1
Demolition permit	1
Townhall registration	1
Job rating	1
Parking place	2
Construction	2
Schoolbus	1
Miscellaneous	1
Traffic regulations	1
Miscellaneous subsidies	1
Monument permit	2
Liquor permit	1
Cultural subsidies	1
Sports subsidies	1
Parking permit	1
Towed car	1
Other	4

Average throughput time

Case Type	Throughput Time
Granted building permit	46d 19:12
Rejected building permit	116d 12:00
Human resources	0d 0:00
Tree cutting permit	0d 0:00
Demolition permit	0d 0:00
Townhall registration	0d 0:00
Job rating	0d 0:00
Parking place	13d 0:00
Construction	20d 0:00
Schoolbus	234d 0:00
Miscellaneous	2d 0:00
Traffic regulations	0d 0:00
Miscellaneous subsidies	0d 0:00
Monument permit	81d 0:00
Liquor permit	0d 0:00
Cultural subsidies	0d 0:00
Sports subsidies	0d 0:00
Parking permit	0d 0:00
Towed car	0d 0:00
Other	0d 0:00

Parameters

Duration: 90

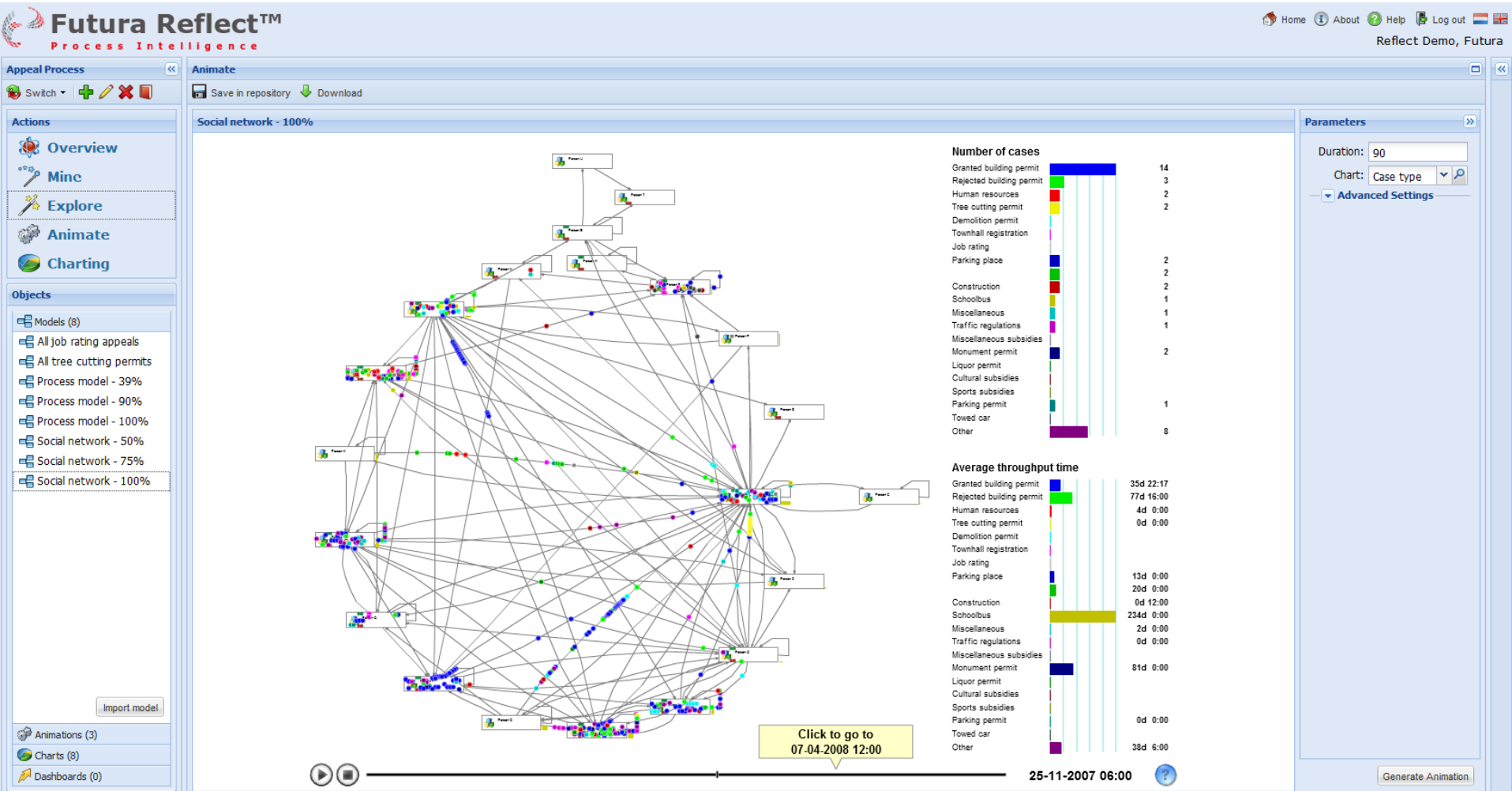
Chart: Case type

Advanced Settings

12-07-2007 02:00

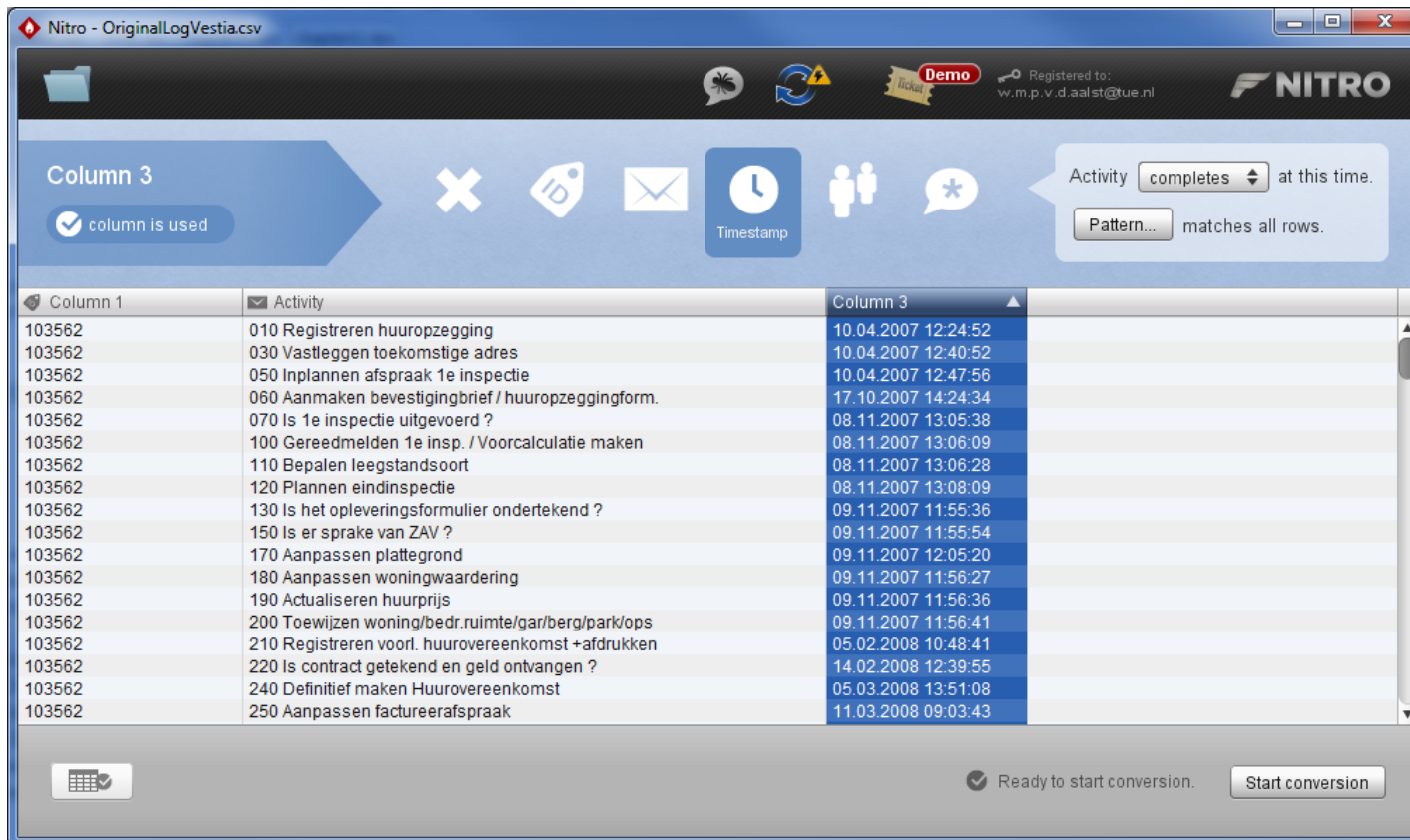
Generate Animation

Futura Reflect (social network)



Loading and converting event logs

- XESame, Nitro, ProMimport



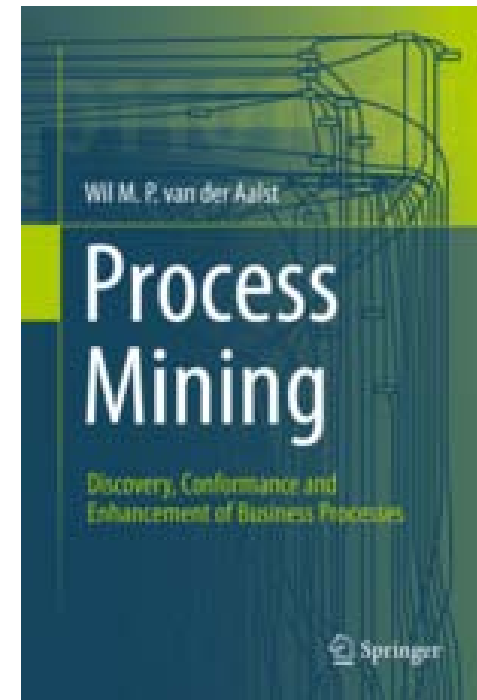
The screenshot shows the Nitro - OriginalLogVestia.csv application window. The interface includes a toolbar with icons for file operations, a 'Demo' label, and a 'Registered to: w.m.p.v.d.aalst@tue.nl' status. Below the toolbar, there are configuration options for 'Column 3' (a dropdown menu showing 'column is used'), a 'Timestamp' button, and an 'Activity' dropdown set to 'completes' with a note 'at this time.' and a 'Pattern...' button with the note 'matches all rows.'.

Column 1	Activity	Column 3
103562	010 Registreren huuropzegging	10.04.2007 12:24:52
103562	030 Vastleggen toekomstige adres	10.04.2007 12:40:52
103562	050 Inplannen afspraak 1e inspectie	10.04.2007 12:47:56
103562	060 Aanmaken bevestigingsbrief / huuropzeggingform.	17.10.2007 14:24:34
103562	070 Is 1e inspectie uitgevoerd ?	08.11.2007 13:05:38
103562	100 Gereedmelden 1e insp. / Voorcalculatie maken	08.11.2007 13:06:09
103562	110 Bepalen leegstandsoort	08.11.2007 13:06:28
103562	120 Plannen eindinspectie	08.11.2007 13:08:09
103562	130 Is het opleveringsformulier ondertekend ?	09.11.2007 11:55:36
103562	150 Is er sprake van ZAV ?	09.11.2007 11:55:54
103562	170 Aanpassen plattegrond	09.11.2007 12:05:20
103562	180 Aanpassen woningwaardering	09.11.2007 11:56:27
103562	190 Actualiseren huurprijs	09.11.2007 11:56:36
103562	200 Toewijzen woning/bedr.ruimte/gar/berg/park/ops	09.11.2007 11:56:41
103562	210 Registreren voorl. huurovereenkomst +afdrukken	05.02.2008 10:48:41
103562	220 Is contract getekend en geld ontvangen ?	14.02.2008 12:39:55
103562	240 Definitief maken Huurovereenkomst	05.03.2008 13:51:08
103562	250 Aanpassen factureringsafpraak	11.03.2008 09:03:43

At the bottom of the window, there is a status bar with a 'Ready to start conversion.' message and a 'Start conversion' button.

More information on Process Mining

Process Mining: Discovery, Conformance and Enhancement of Business Processes
by W.M.P. van der Aalst, Springer Verlag,
2011 (ISBN 978-3-642-19344-6).



- www.processmining.org
- www.processmining.org/book/ (slides supporting book)
- http://www.processmining.org/event_logs_and_models_used_in_book (event logs and models)

Conclusion



TU/e

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

After this lecture you should be able to:

- Apply the boundedness and liveness preserving reduction rules to a concrete marked net.
- Know the following subclasses: state machine (SM), marked graph (MG), free-choice (FC), extended free-choice (EFC), asymmetric choice (AC).
- Construct a Petri net that has a set of desirable properties (terminating, deadlock-free, live, bounded, safe, and/or reversible) *and* that is of a particular subclass (SM, MG, FC, EFC, or AC).
- Know the various properties given for particular subclasses, e.g.:
 - liveness in marked graphs (Theorem 7)
 - liveness in free-choice nets (Theorem 12)
 - (SM and MG covers for free choice nets (Theorems 13 and 14))
- Give an overview of process mining.