Petri-netten in Protos: wat moet je ermee?

Dr.ir. Hajo Reijers

Faculteit Technologie Management,
TU Eindhoven

e-mail: h.a.reijers@tm.tue.nl

Agenda

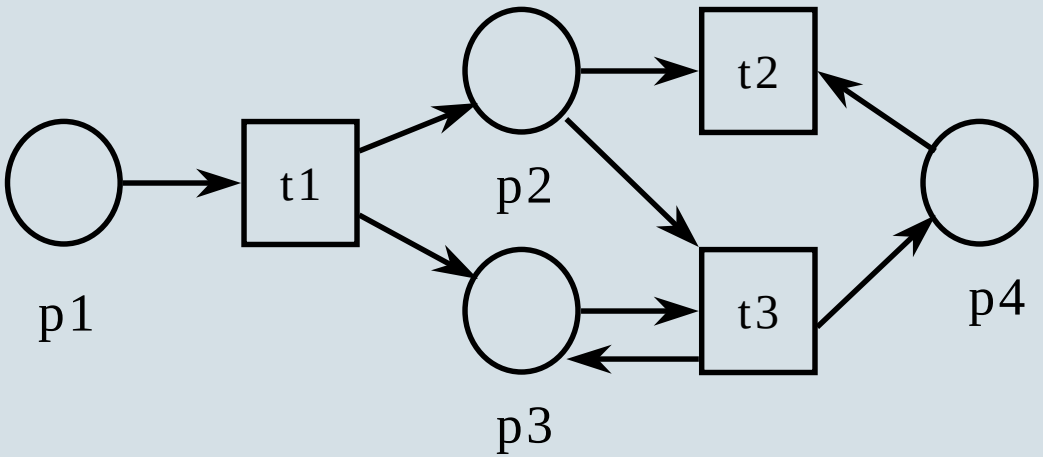
- Petri-netten
  - klein beetje geschiedenis
  - wat is het nou precies?
- De kracht van Petri-netten
  - vergelijking met andere procesmodelleer-technieken
  - specifieke voordelen
- Praktisch gebruik Petri-netten in Protos
  - een paar tips
- Vragen

Petri-netten: een kleine geschiedenis

- Klassiek Petri-net bedacht door Carl Adam Petri in 1962
- Tot 1985 vooral gebruikt binnen de wetenschap
- Sinds de jaren ‘80 is het praktische gebruik toegenomen (hoog-niveau netten en tools)
- Hoog-niveau Petri-netten zijn Petri-netten uitgebreid met:
  - kleur (voor het modelleren van attributen)
  - tijd (voor prestatie analyse)
  - hiërarchie (voor structurering van de modellen)
- Tegenwoordig:
  - meer dan 10.000 wetenschappelijke publicaties
  - gebruik in versch. systemen, modelleergereedschappen, etc.

Wat is een Petri-net?

- “Een bipartiete graaf van transities en plaatsen”
- Kortom:
  - rondjes en vierkantjes met elkaar verbonden door pijlen,
  - pijlen gaan alleen van een rondje naar een vierkant of andersom

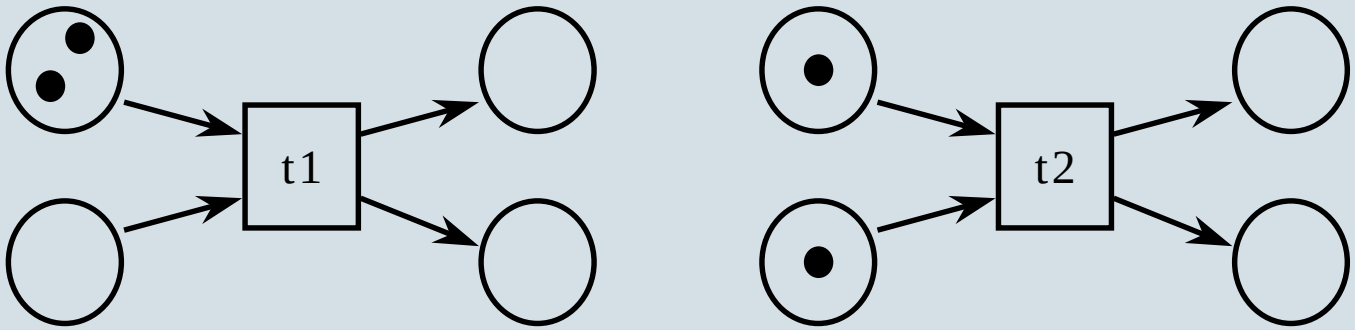


Bij het modelleren van bedrijfsprocessen praten we meestal over:

- *activiteiten* (t1, t2, t3), en
- *toestanden* (p1, p2, p3, p4)

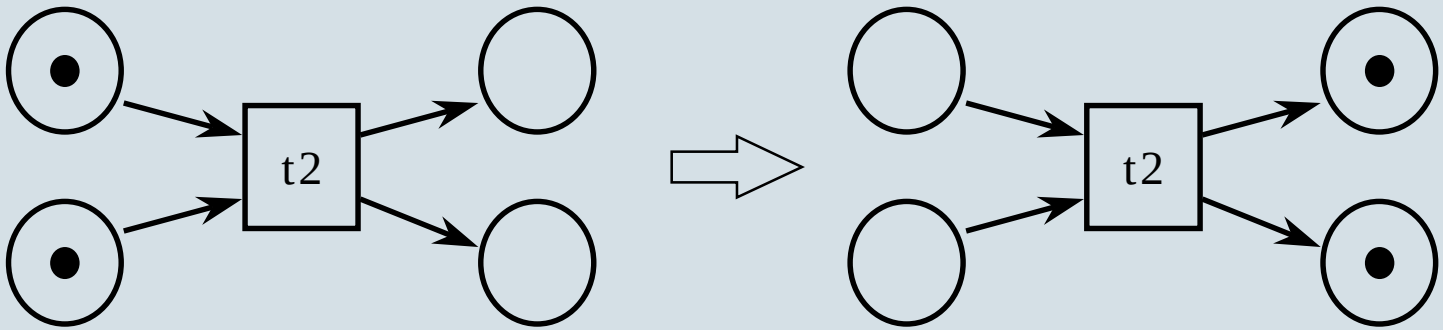
Hoe werkt een Petri-net?

- Tokens worden gebruikt om de dynamiek van het systeem weer te geven en liggen in toestanden
- Activiteiten zijn de actieve componenten, toestanden en tokens zijn passief
- Een activiteit is uitvoerbaar als *elk* van de inputs een token bevat



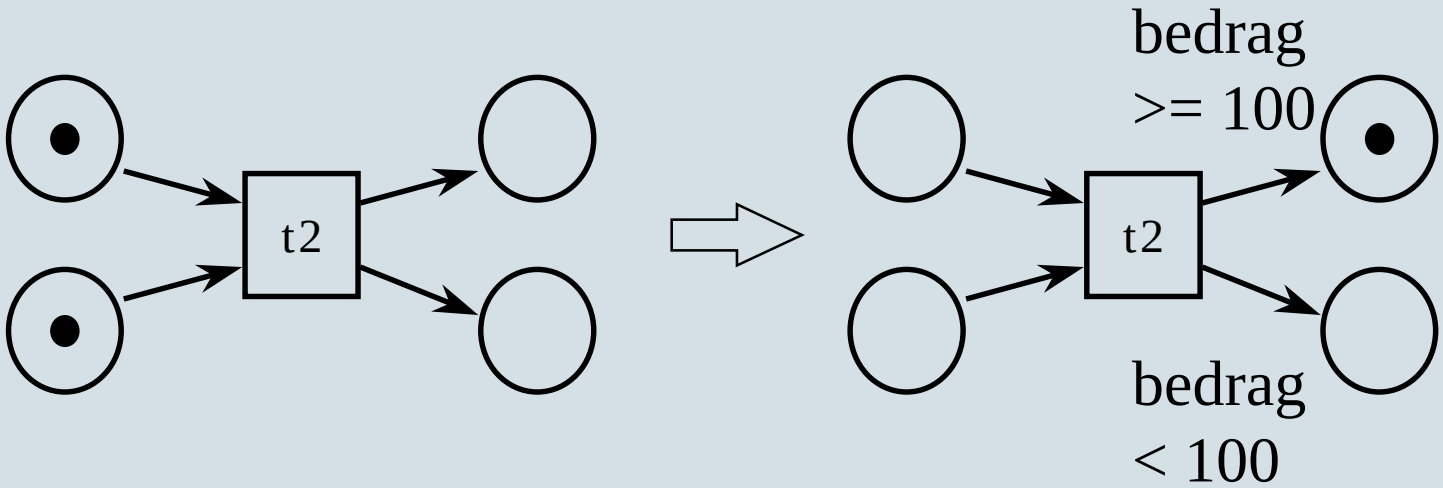
Hoe werkt een Petri-net?

- Een uitvoerbare activiteit kan worden uitgevoerd
- “Uitvoeren” correspondeert met het consumeren van tokens uit de inputs, en het produceren van tokens voor de outputs

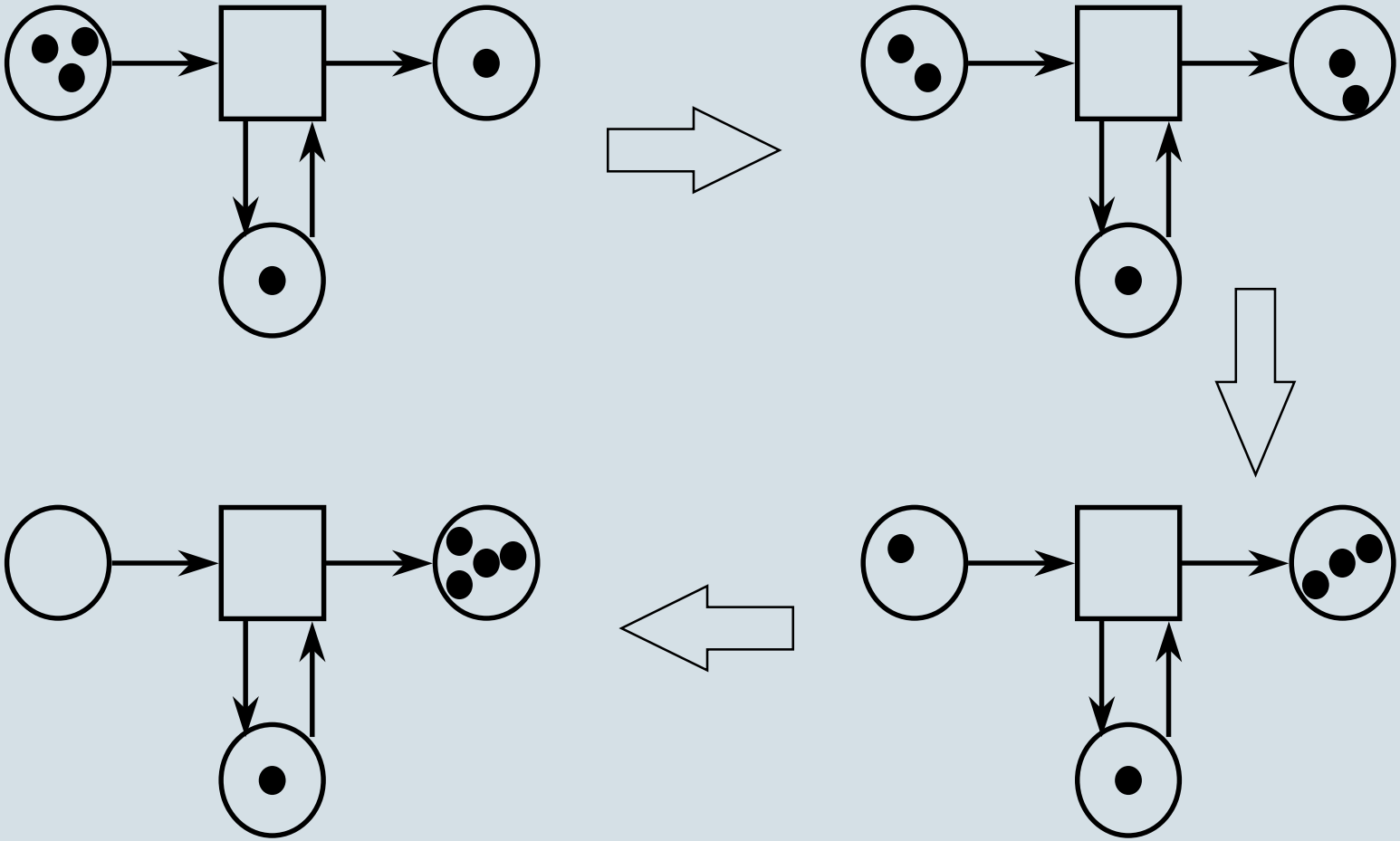


Hoe werkt een Petri-net?

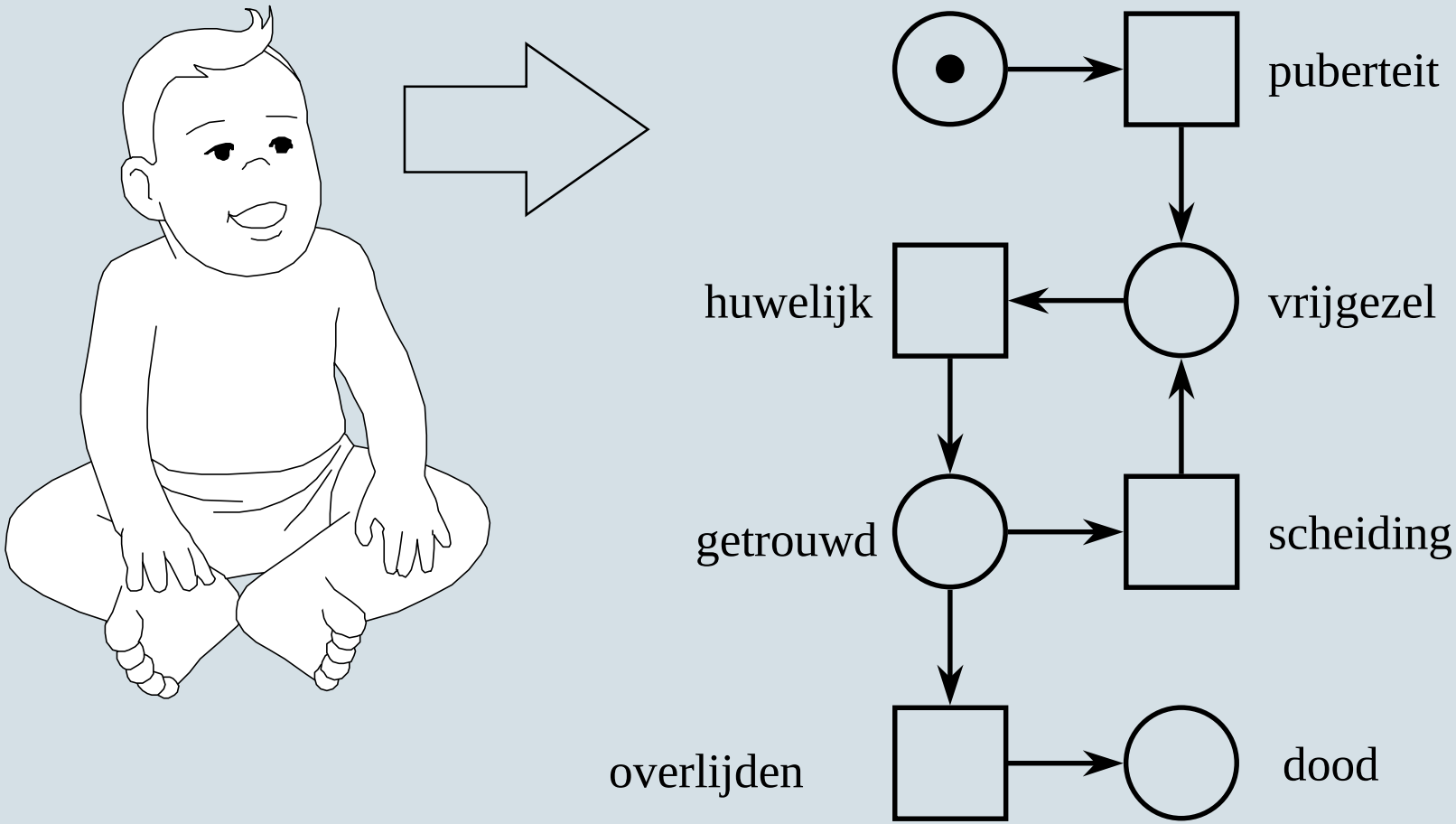
- Een uitvoerbare activiteit kan worden uitgevoerd
- “Uitvoeren” correspondeert met het consumeren van tokens uit de inputs, en het produceren van tokens voor de outputs



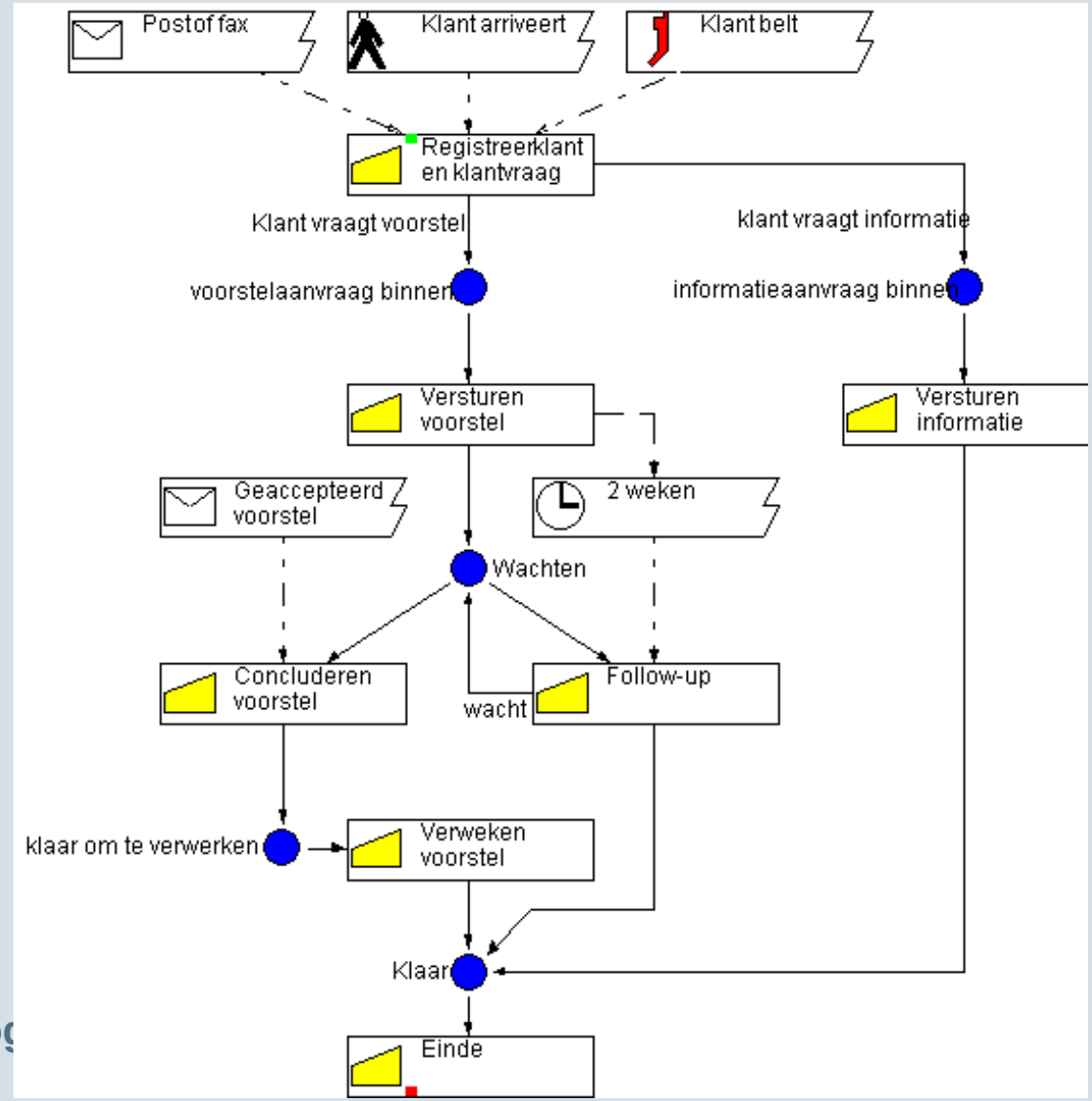
Voorbeeld



Nog een voorbeeld: de levenscyclus



Laatste voorbeeld: een bedrijfsproces



Petri-netten en bedrijfsprocessen

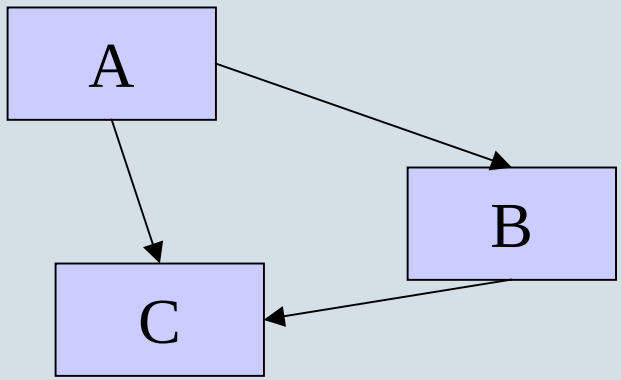
Toestand van het proces wordt gemodelleerd door tokens in toestanden.

Overgangen van de ene naar de andere toestand worden gemodelleerd door activiteiten.

- Tokens representeren objecten (mensen, goederen, machines), informatie, status van objecten.
- Toestanden representeren buffers, kanalen, geografische locaties, condities of toestanden.
- Activiteiten representeren gebeurtenissen, transformaties of transportaties.

Vergelijking met andere technieken

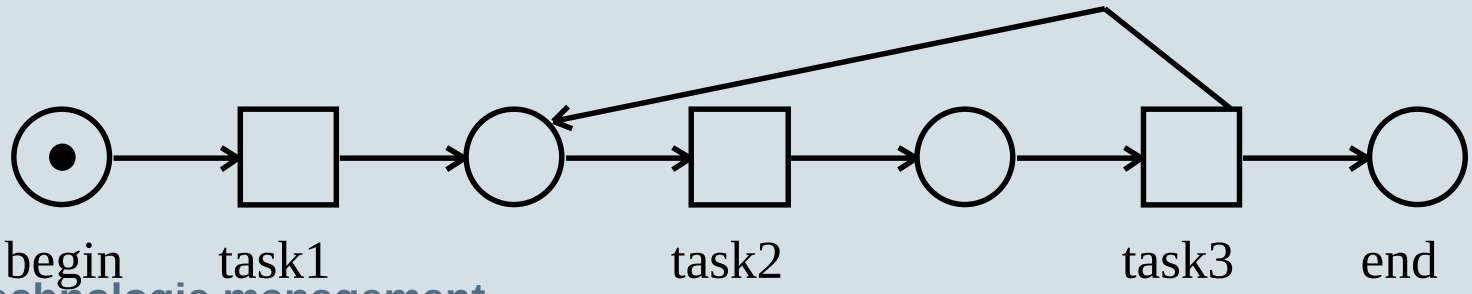
- Veel modelleertechnieken hebben alleen syntax, geen semantiek (b.v. dataflow-diagrammen)



- Sommige technieken hebben duidelijke syntax en semantiek, maar kunnen minder (b.v. flowcharts)
- Sommige technieken *zijn* Petri-netten, maar worden niet zo genoemd (b.v. UML activity diagrams)

Specifieke voordelen Petri-netten: semantiek

- Zeer precieze betekenis:
  - *echte* afstemming mogelijk tussen partijen
  - uitvoerbaar door computers (Workflow management systemen, simulatie-pakketten, etc.)
  - verifieerbaar (b.v. tool als Woflan: http://tmitwww.tm.tue.nl/research/workflow/woflan/)
  - andere analyse-technieken (b.v. *state-space-analysis*)



Specifieke voordelen Petri-netten: toestanden

- “Waarom die toestanden? Deze doen toch eigenlijk niets?”
- Eerst de “zachte” argumenten:
  - Output-toestanden maken de toegevoegde waarde van een activiteit duidelijk.
  - Input-toestanden maken noodzakelijk welke deelproducten gereed moeten zijn.
  - In veel processen is de toestand direct gerelateerd aan belangrijke mijlpalen (b.v. juridische toestanden)
  - Een casus bevindt zich *het merendeel van de tijd in een toestand*!

Process	Mean waiting-time as percentage of throughput-time	Source
Treatment of pension claims	98%	Covee 1990 (p. 32)
Treatment of insurance claims	> 99%	Dur 1992 (pp. 150-151)
Treatment of insurance claims	> 99%	Cox 1992 (p. 6)
Treatment of insurance claims	> 95%	Schijndel 1991 (p. 25)
Treatment of tax returns	> 99%	V. Egten/Platier 1993

Source: E.A.H. Platier, proportion of waiting-time in the total throughput-time in business processes in financial service organisations, PhD Thesis, 1996.

Specifieke voordelen Petri-netten: toestanden

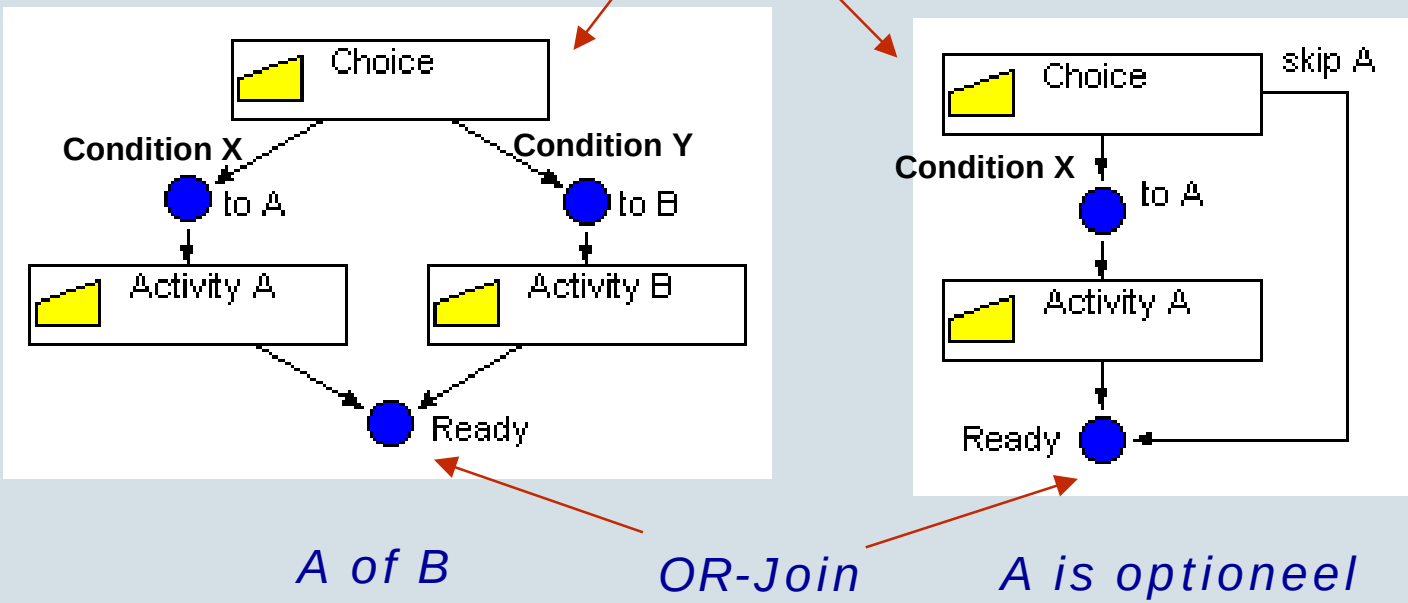
Dan het “harde argument”:

- Zonder toestanden zijn bepaalde procesconstructies niet (natuurlijk) te modelleren of te onderscheiden van elkaar:
  - expliciete keus versus impliciete keus
  - mijlpaalconstructies
  - andere toestandgebaseerde patronen (zie http://www.tm.tue.nl/it/research/patterns)

Expliciete keus

Conditionele Routering

OR-Split (expliciet)



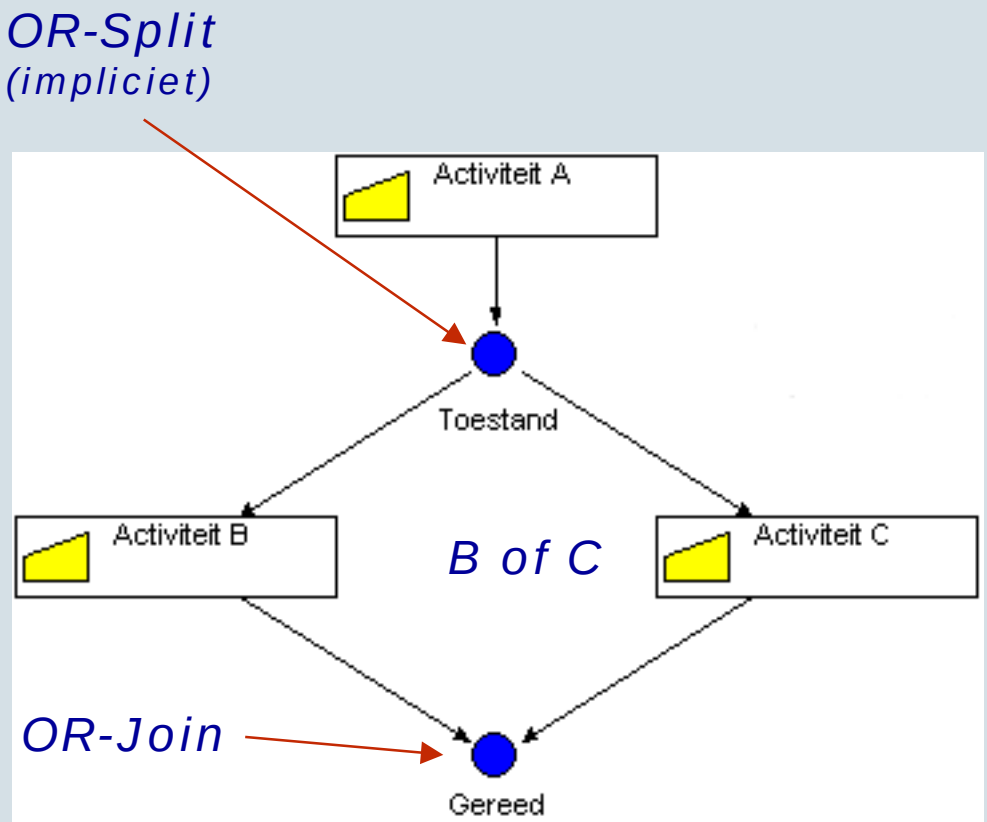
A of B

OR-Join

A is optioneel

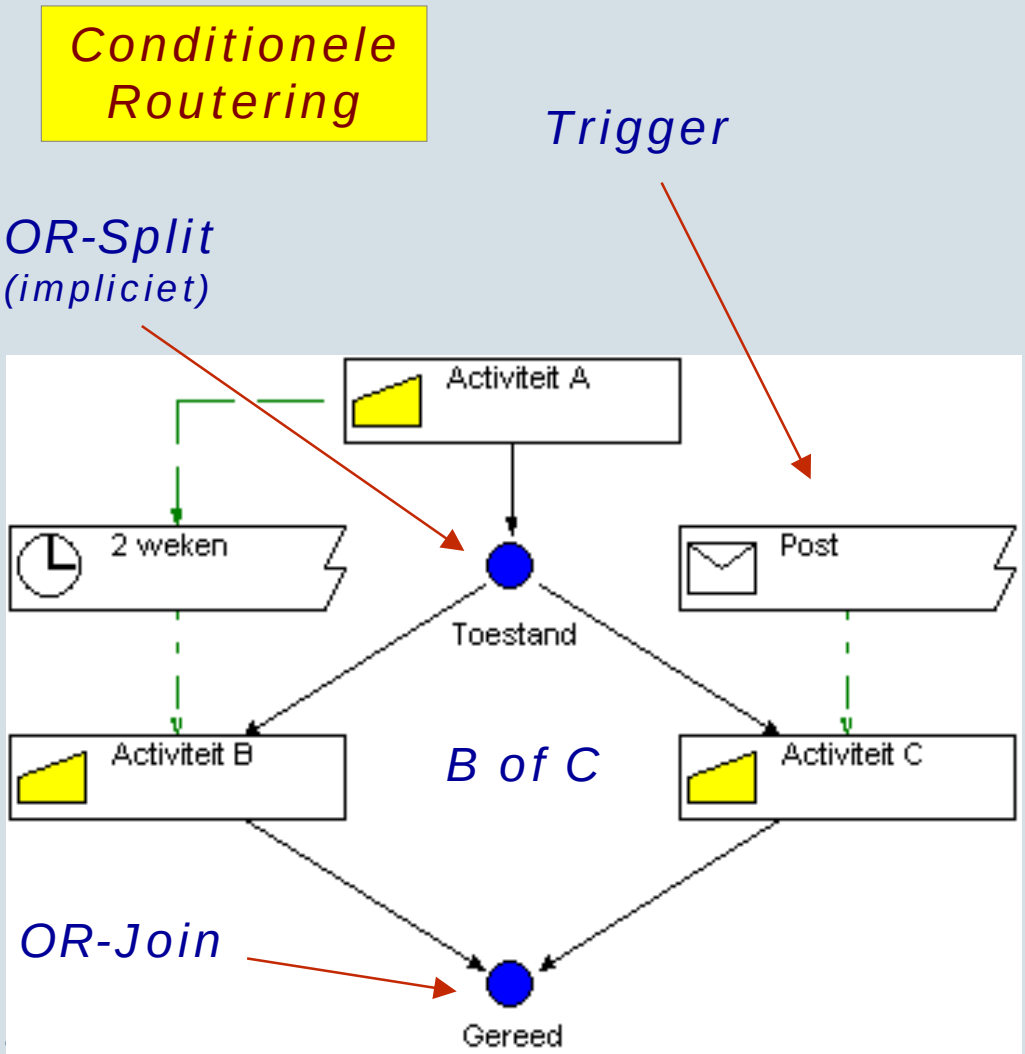
Impliciete keus

Conditionele Routering



- De keuze wordt NIET in Activiteit A gemaakt.
- Keuze wordt gemaakt DOORDAT B of C gestart wordt.
- Terwijl A in gang is, is nog niet bekend welke volgende activiteit uitgevoerd moet worden. Dit hangt af van …

Triggers



- Vaak is een prikkel ***(Trigger)*** aanleiding om een activiteit te starten. Vaak niet bekend wanneer en welke prikkel plaats zal vinden.
- Een trigger kan b.v. een externe gebeurtenis of het bereiken van een bepaald tijdstip zijn.

Einde?

- Iedereen kan nu Petri-netten modelleren met Protos!

Einde?

- Iedereen kan nu Petri-netten modelleren met Protos!

- Zo min mogelijk toestanden? Accoord, maar dan de volgende tips…

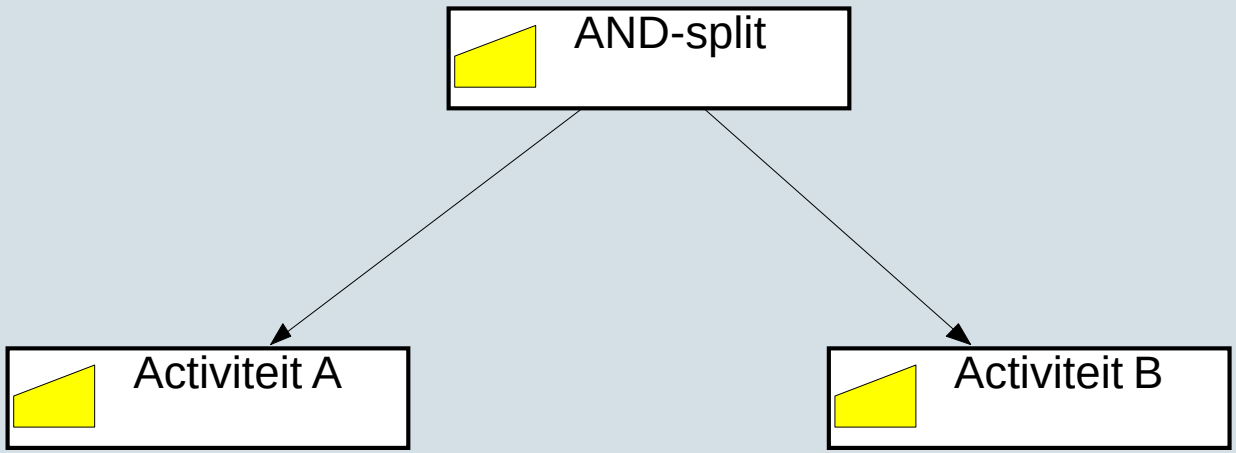
Praktisch gebruik

Geen gevaar voor dubbelzinnigheden, dus toestanden kunnen weggelaten worden

Petri-net

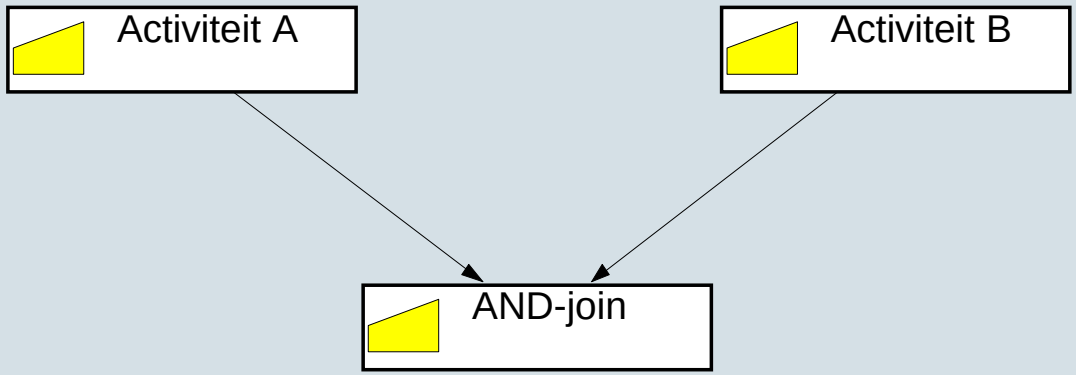
Protos

AND-Split



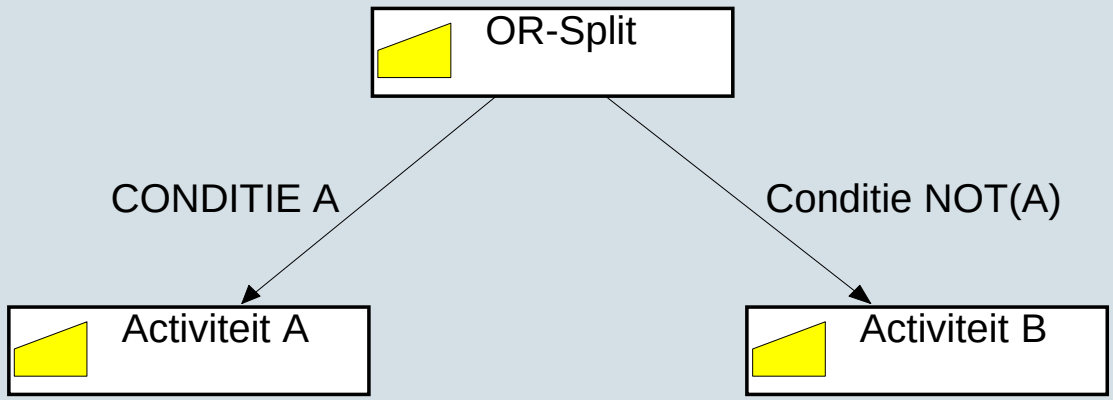
Zowel Activiteit A als Activiteit B worden (al dan niet gelijktijdig) uitgevoerd. Er zijn ***geen toestanden*** *nodig* ***EN*** *er staan* ***geen condities op de pijlen****.*

AND-Join



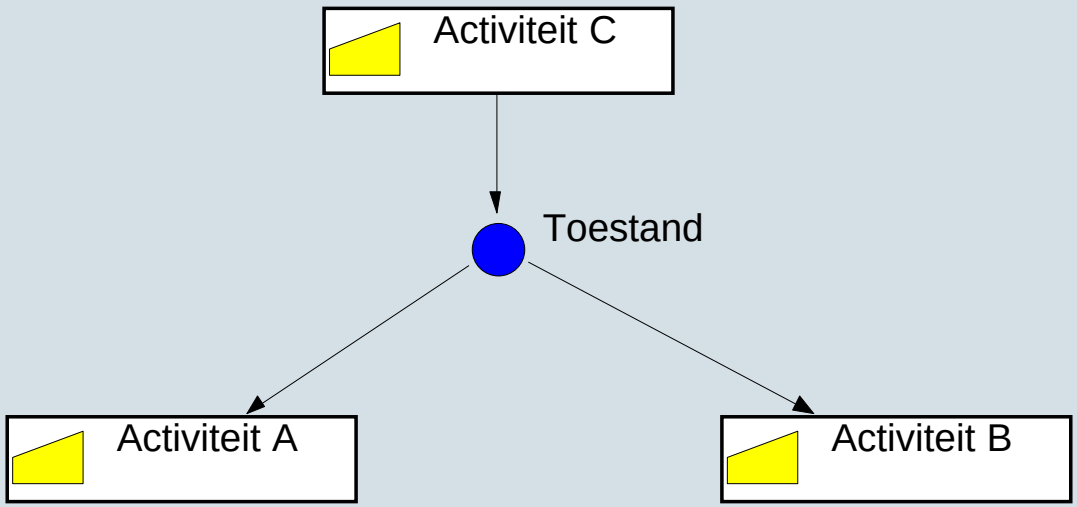
*Als zowel Activiteit A als Activiteit B uitgevoerd zijn dan kan de AND-join plaatsvinden. Er zijn **geen toestanden** nodig.*

Expliciete OR-Split



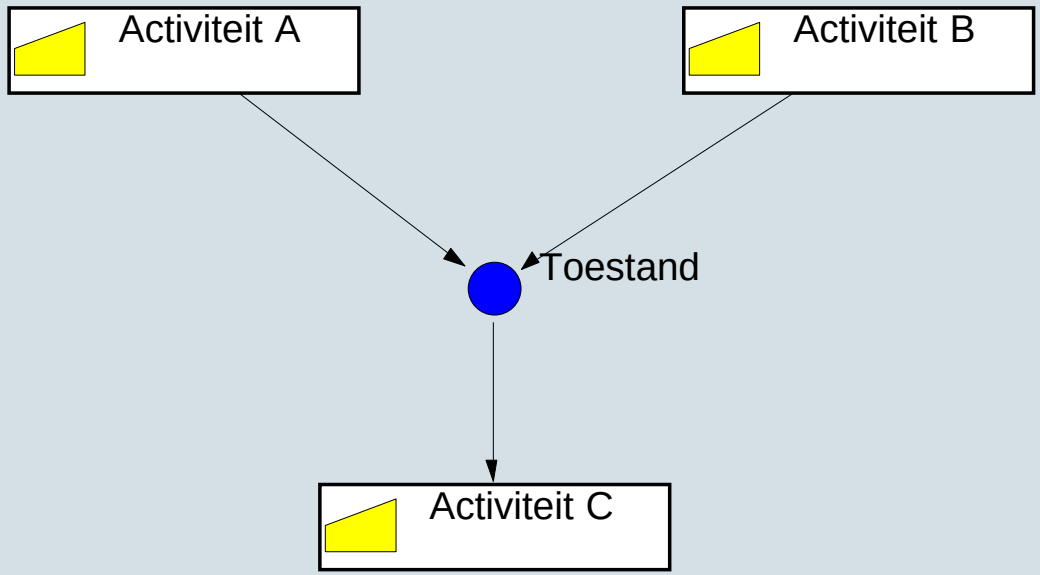
Als aan conditie A voldaan wordt dan wordt Activiteit A uitgevoerd, anders Activiteit B. Het beslismoment ligt in de OR-split activiteit.
*Er zijn **geen toestanden** nodig, maar wel **CONDITIES OP PIJLEN**.*

Impliciete OR-Split



*Activiteit A of Activiteit B wordt uitgevoerd, het beslismoment ligt **NA** uitvoering van Activiteit C. Er is een Toestand noodzakelijk er zijn geen condities op de pijlen*

OR-Join



Twee alternatieve deelprocessen komen samen in een toestand.
*Een toestand is **verplicht**.*

Resumé

- Indien geen verwarring mogelijk mogen plaatsen weg….
- Maar, plaatsen:
  - zijn soms noodzakelijk en vaak handig,
  - geven informatie over toestand proces (bereikte mijlpalen / afgeronde deelproducten),
  - geven grootste deel van levenscyclus zaken weer!
- Petri-netten geven proces precieze betekenis


Vragen?