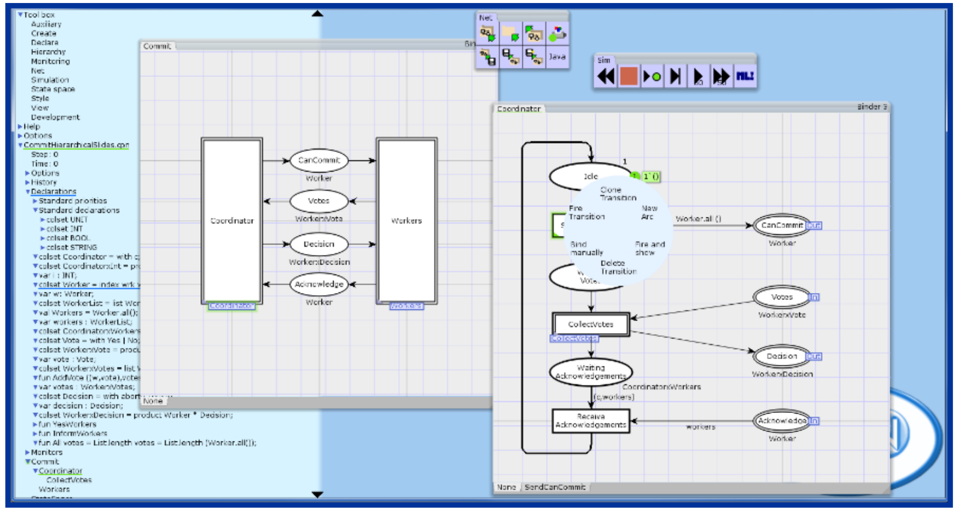
Theory-Tool | Part 1

Motivation and Overview of Coloured Petri Nets and CPN Tools

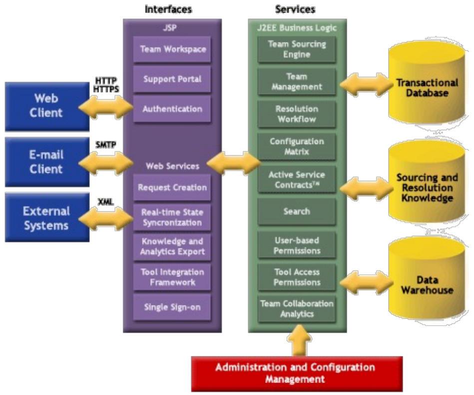


Lars Michael Kristensen
Department of Computer Science, Electrical Engineering, and Mathematical Sciences
Western Norway University of Applied Sciences
Email: lmkr@hvl.no | https://www.hvl.no/person/?user=Lars.Michael.Kristensen

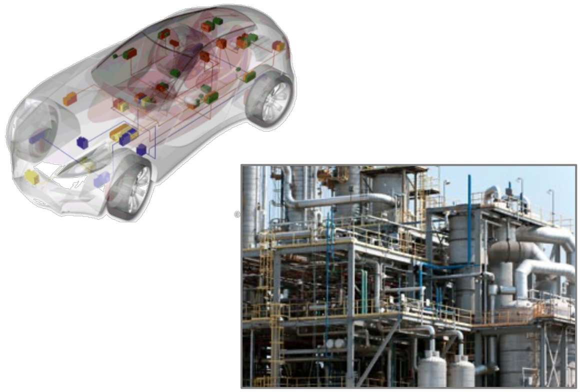
Concurrent and distributed systems

- **The vast majority of system development projects today are concerned with concurrent and distributed systems**
  - Structured as a collection of concurrently executing software components and applications (parallelism)
  - Operation relies on communication, synchronisation, and resource sharing

Internet protocols, cloud, IoT, web-based applications



Multi-core platforms and multi-threaded software



Automation systems and networked control systems

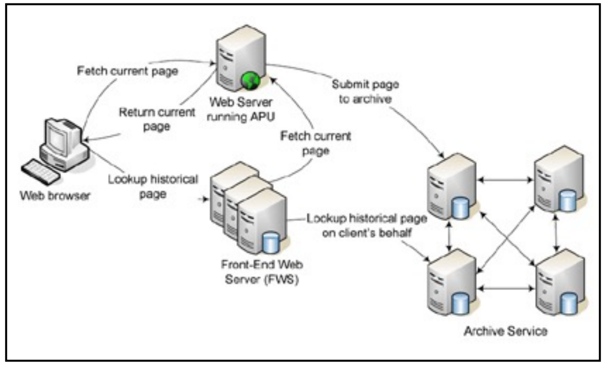
Complex behaviour

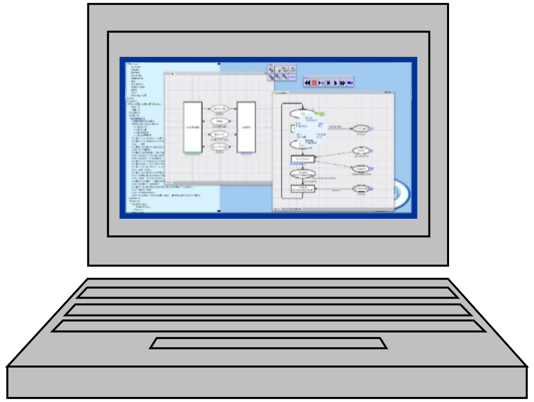
- **The engineering of concurrent and distributed systems is challenging due to their complex behaviour**
  - Concurrently executing and independently scheduled components
  - Non-deterministic and asynchronous behaviour (e.g., timeouts, message loss, external events, …)
  - Almost impossible for developers to have a complete understanding of the system behaviour
  - Testing is challenging and reproducing errors is often difficult

- **Methods to support the engineering of reliable distributed and concurrent systems are highly relevant**

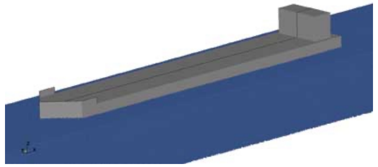
Modelling

- One way to approach these challenges is via the construction of executable models
- Models are abstract representations which can be manipulated by software tools





- Modelling is widely used in most engineering disciplines



Why modelling?

- **Benefits of constructing executable models**
  - Insight into the design and operation of the system
  - Completeness: results in a more complete design
  - Correctness: reveal errors and ambiguities in the design phase
- **Abstraction and communication – validation using high-level and domain-specific concepts in development**
- **Reliability – testing and verification and prior to implementation and deployment**
  - Functional properties (e.g., deadlocks, timing requirements,…)
  - Performance properties (e.g., delay, throughout, scalability,…)

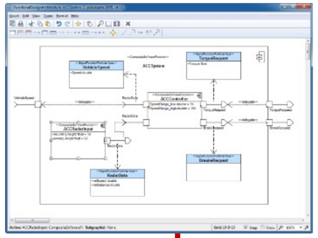
Model-driven engineering

- **Productivity - models may in some cases also be used (directly or indirectly) as a basis for implementation**

Problem domain
(vision and requirements)

Validate

Software engineering
based on models



Model
execution
Code
generation

Technical domain
(implementation)

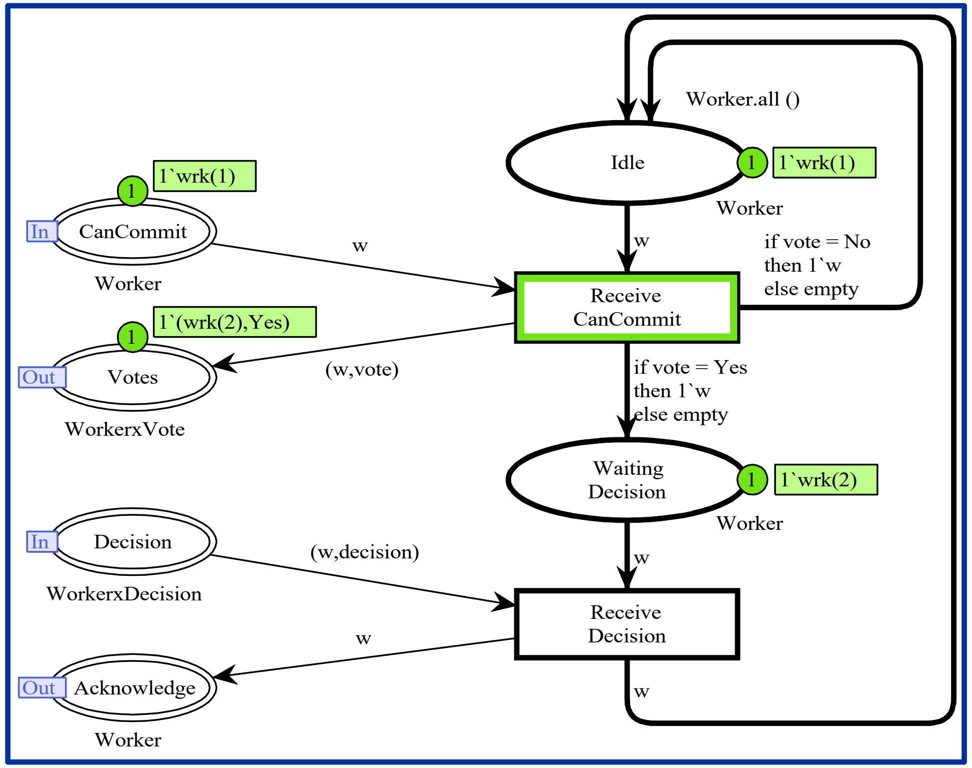
Validation:
Are we building the right software?

Verify

Verification:
Are we building the software right ?

Coloured Petri Nets - CPNs

- General-purpose graphical modelling language for the engineering of concurrent and distributed systems
- Combines Petri Nets and a programming language



Petri Nets

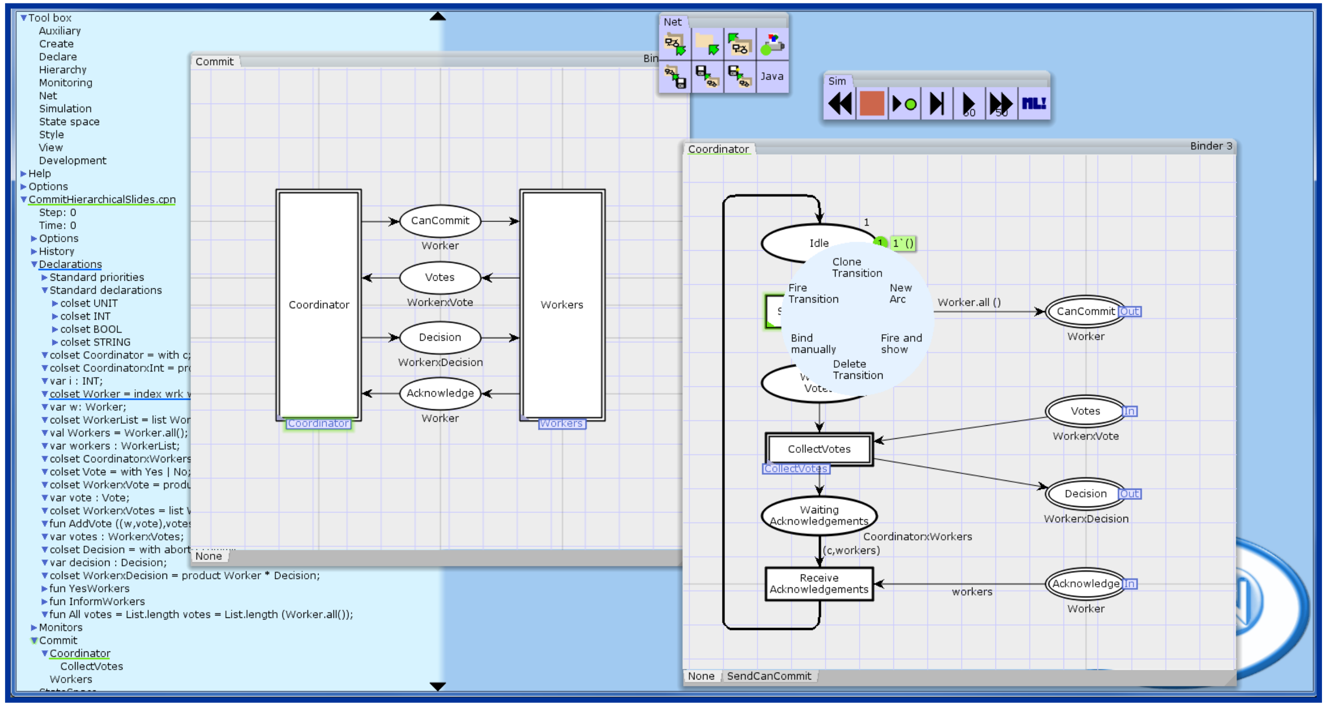
graphical notation
concurrency
communication
synchronisation
resource sharing

CPN ML (Standard ML)

data and data manipulation
compact modelling
parameterisable models

CPN Tools [www.cpntools.org]

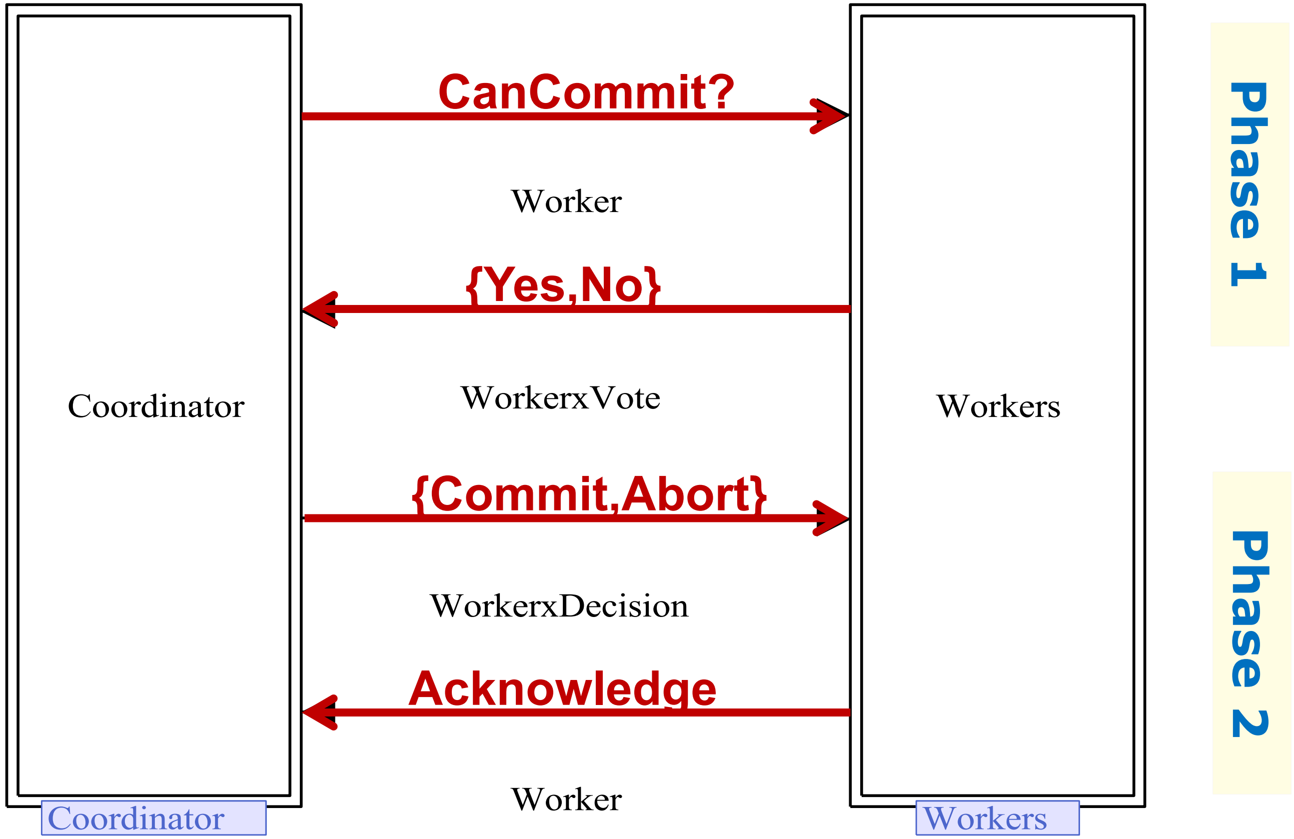
- **Practical use of CPNs is supported by CPN Tools**



- **Editing** and **syntax check**
- **Interactive- and automatic simulation**
- **Verification based on state space exploration**
- **Simulation-based performance analysis**

Example: Two-phase commit transaction protocol

- A concurrent system consisting of a coordinator process and a number of worker processes



CPN Tools demo

`lecture1-introduction.cpn`

- **User-interaction with CPN Tools**
  - **Index and workspace**
  - **Binders and tool palettes - drag-and-drop**
  - **Contextual menus - right click**
  - **No menu-bars or dialog-boxes**

Examples of CPN Tools users

North America

- Boeing
- Hewlett-Packard
- Samsung Information Systems
- National Semiconductor Corp.
- Fujitsu Computer Products
- Honeywell Inc.
- MITRE Corp.,
- Scalable Server Division
- E.I. DuPont de Nemours Inc.
- Federal Reserve System
- Bell Canada
- Nortel Technologies, Canada

Asia

- Mitsubishi Electric Corp., Japan
- Toshiba Corp., Japan
- SHARP Corp., Japan
- Nippon Steel Corp., Japan
- Hongkong Telecom Interactive Multimedia System

Europe

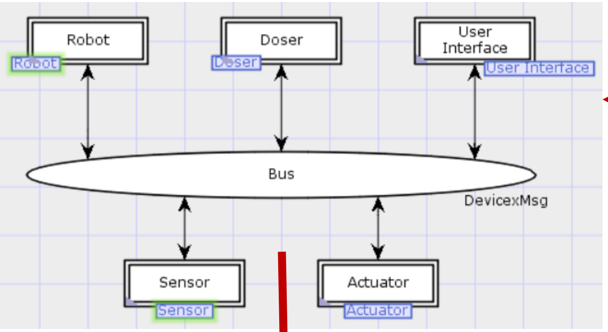
- Alcatel Austria
- Siemens Austria
- Bang & Olufsen, Denmark
- Nokia, Finland
- Alcatel Business Systems, France
- Peugeot-Citroën, France
- Dornier Satellitensysteme, Germany
- SAP AG, Germany
- Volkswagen AG, Germany
- Alcatel Telecom, Netherlands
- Rank Xerox, Netherlands
- Sydkraft Konsult, Sweden
- Central Bank of Russia
- Siemens Switzerland
- Goldman Sachs, UK

http://cs.au.dk/cpnets/industrial-use/

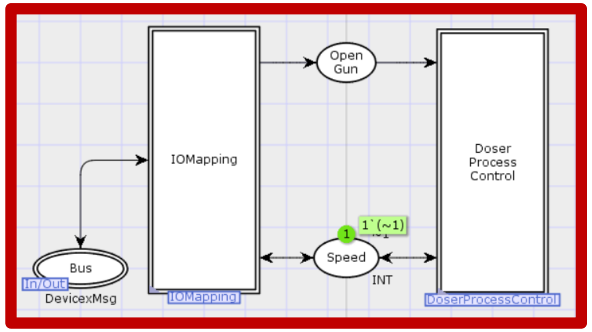
CPN @ Atlas Copco

- **Developing a model-driven software engineering approach and supporting infrastructure**

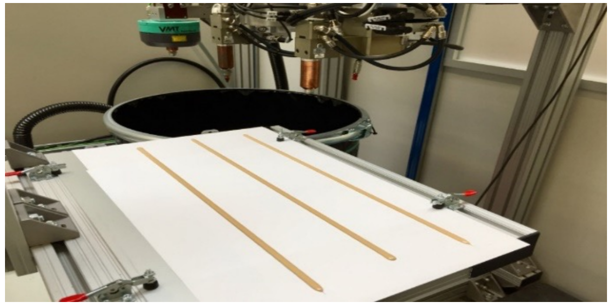
CPN Tools:
editing,
validation, and
verification
(design time)

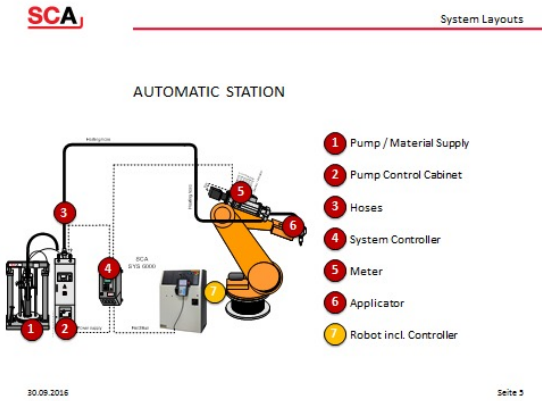


Environment
modelling for
(non-site)
software testing



C++ execution
engine for
deployment and
real-time execution
(run-time)

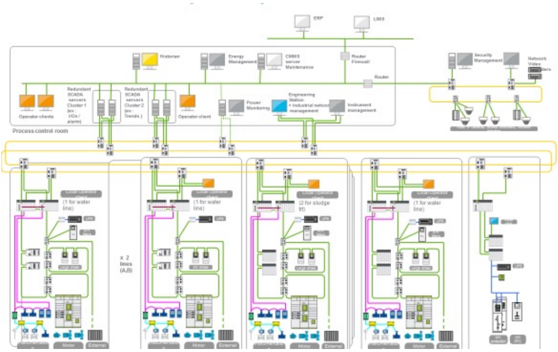




- **The CPN model is directly used as the pump controller software implementation**

CPN @ Schneider Electric

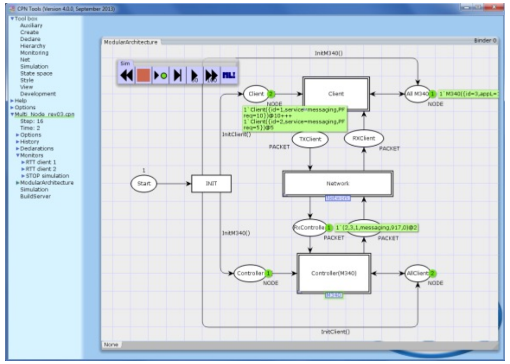
- **Dependability evaluation and capacity planning of large industrial automation architectures**



Dependability analysis software tools

Performance - Reliability
Availability - Safety

Modelling

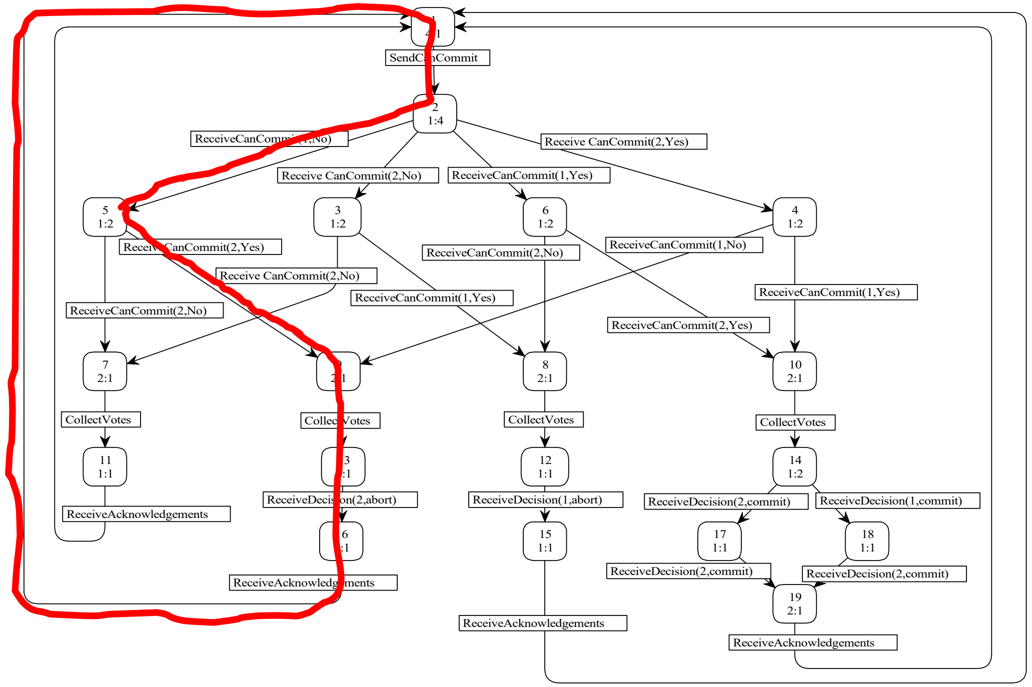


Automated code generation

Tools for modelling driven engineering

Verification and model checking

- **Formal verification** of CPN models can be conducted using **explicit state space exploration**

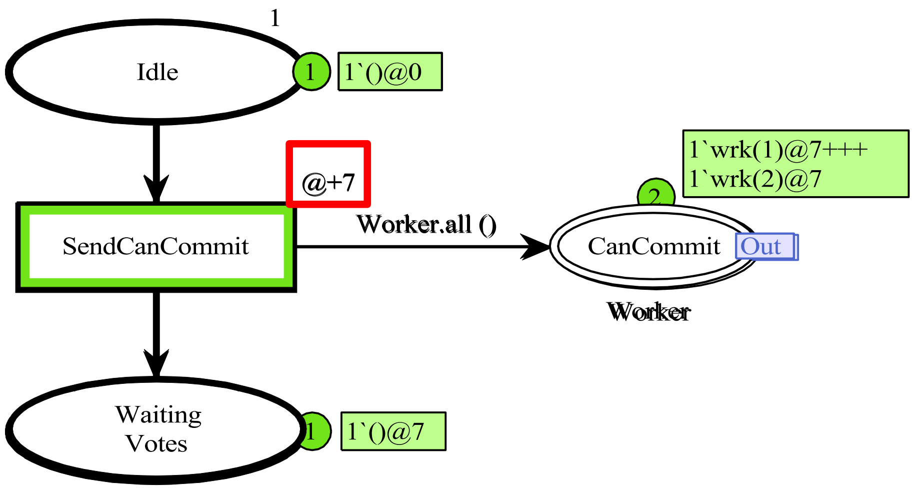


- A state space represents all possible **executions** of the CPN model
- **Standard behavioural properties** can be investigated using the state space report
- **Model-specific properties** can be verified using queries and **temporal logic model checking**

- Several **advanced techniques** available to alleviate the inherent **state explosion problem**

Performance analysis

- **CPNs include a concept of time that can be used to model the timed taken by activities**

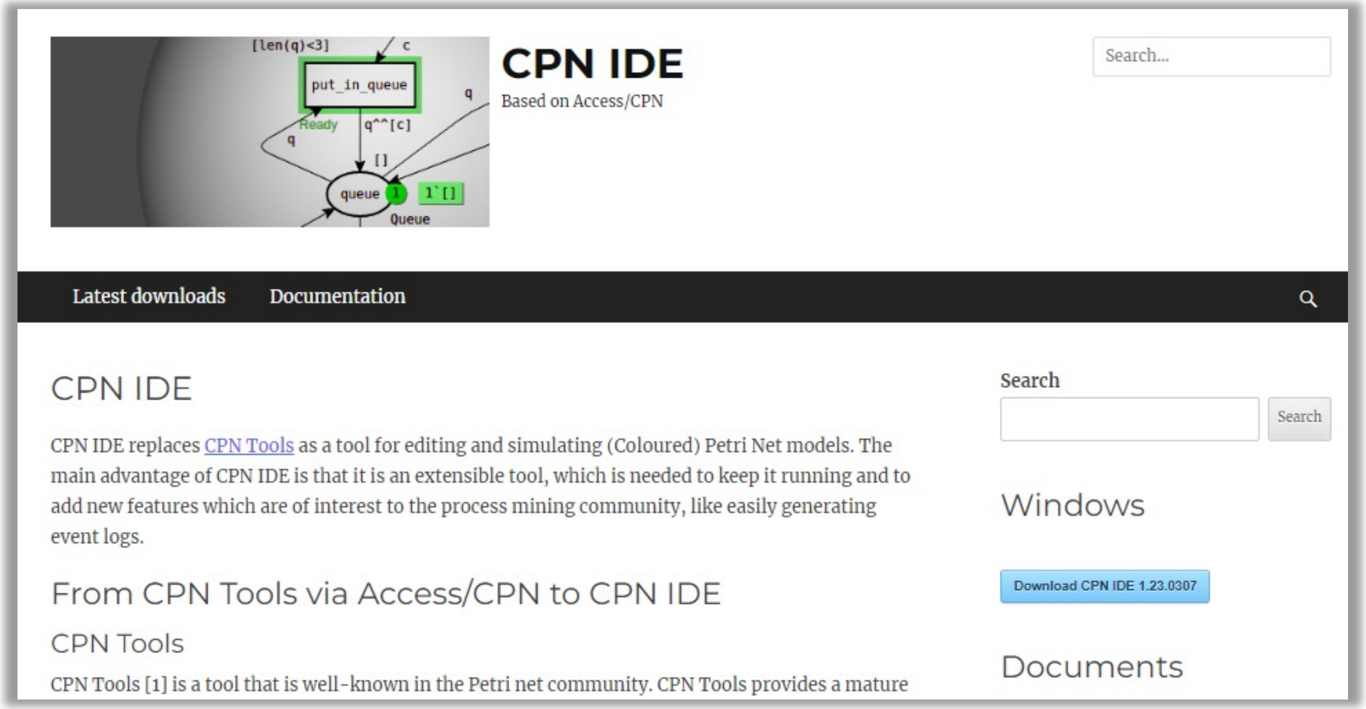


- **A global clock representing the current model time**
- **Tokens carry time stamps describing the earliest possible model time at which they can be removed**
- **Time inscriptions on transitions and arcs are used to give time stamps to the tokens produced on output places**

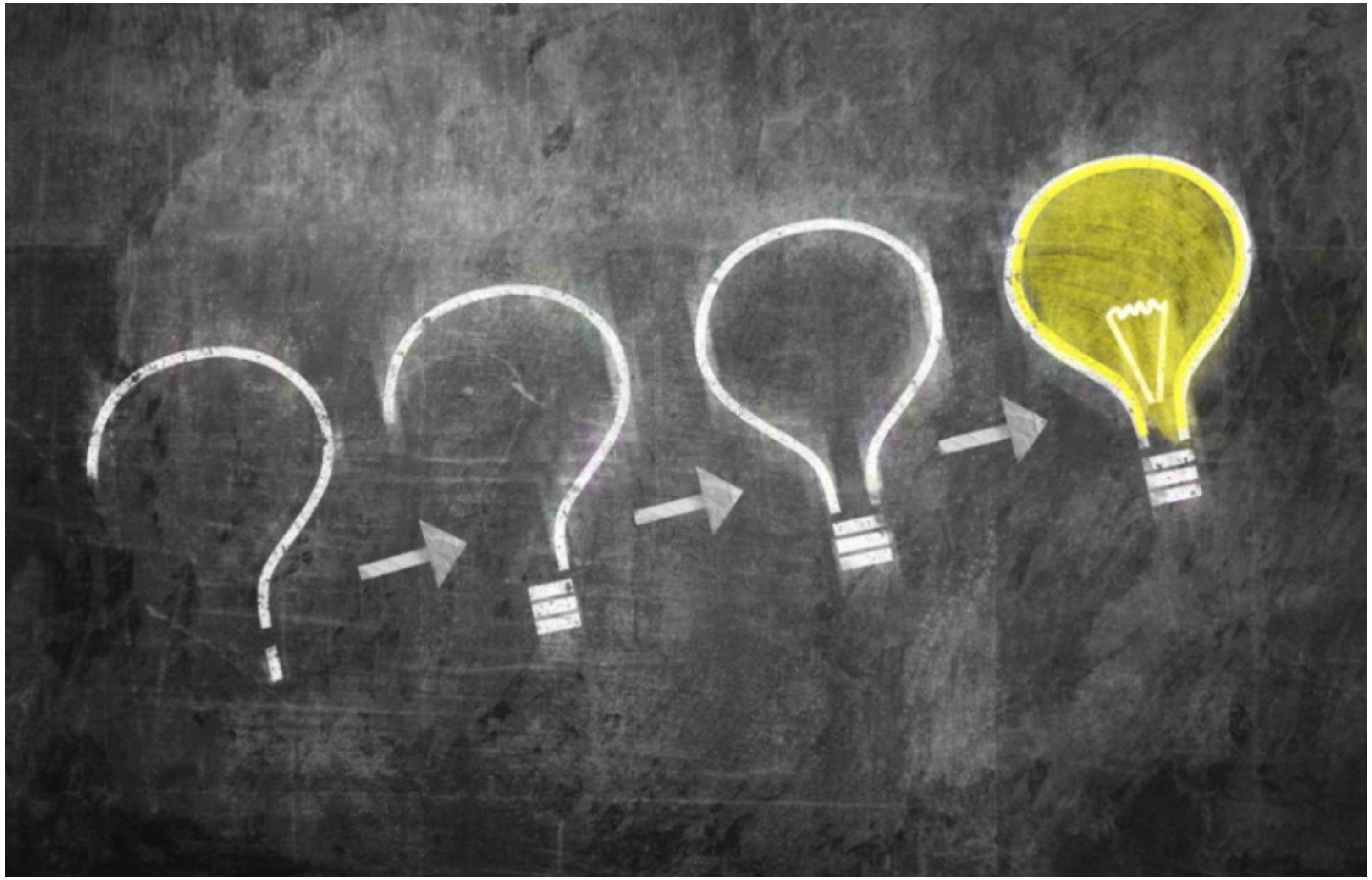
- **Random distribution functions can be used in arc expressions (variable delays, packet loss probabilities, ...)**
- **Data collection monitors and batch simulations can be used for performance analysis**

CPN IDE [cpnide.org]

- **Web-based front-end replacing the CPN Tools GUI**

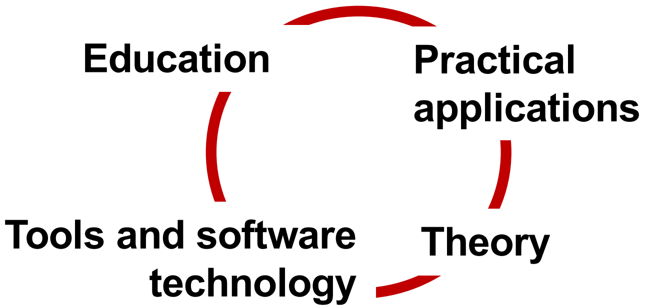


- **Relies on the same underlying simulator as CPN Tools**



Perspectives on CPNs

- **Modelling language combining Petri Nets with a programming language**
- **The development has been driven by an application-oriented research agenda**



- **Key characteristics**
  - **Few but still powerful and expressive modelling constructs**
  - **Implicit concurrency inherited from Petri nets**
    - everything is concurrent unless explicitly synchronised
  - **Verification and performance analysis supported by the same modelling language**