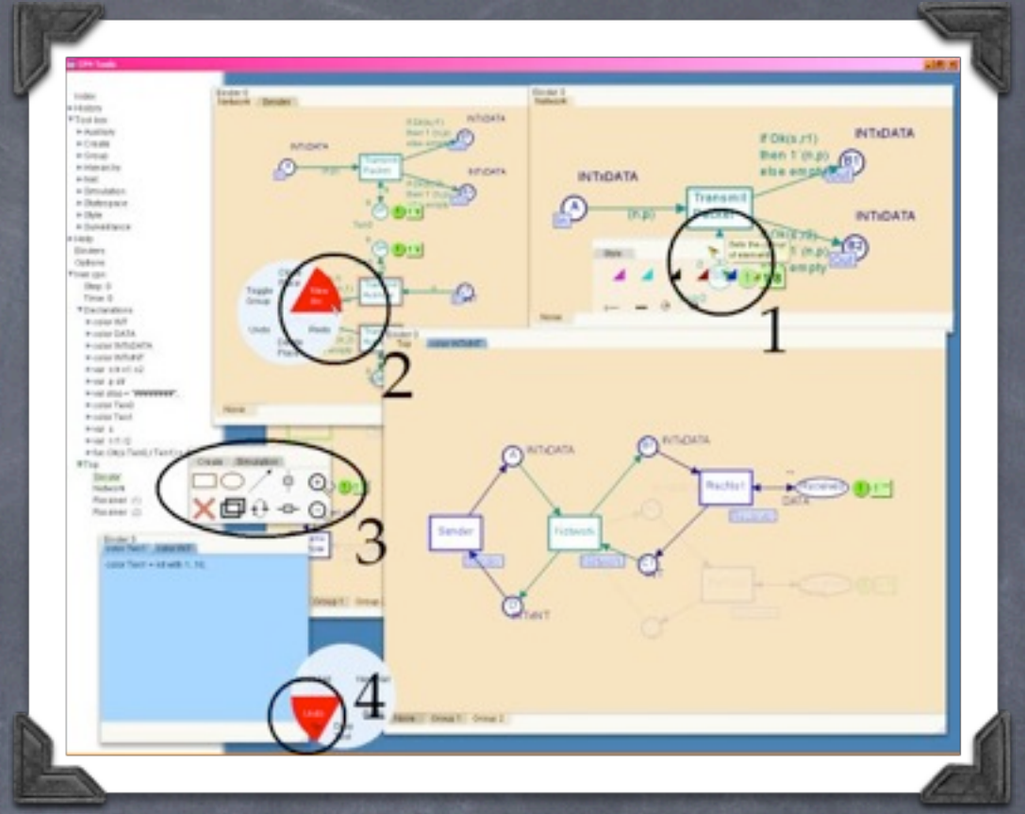
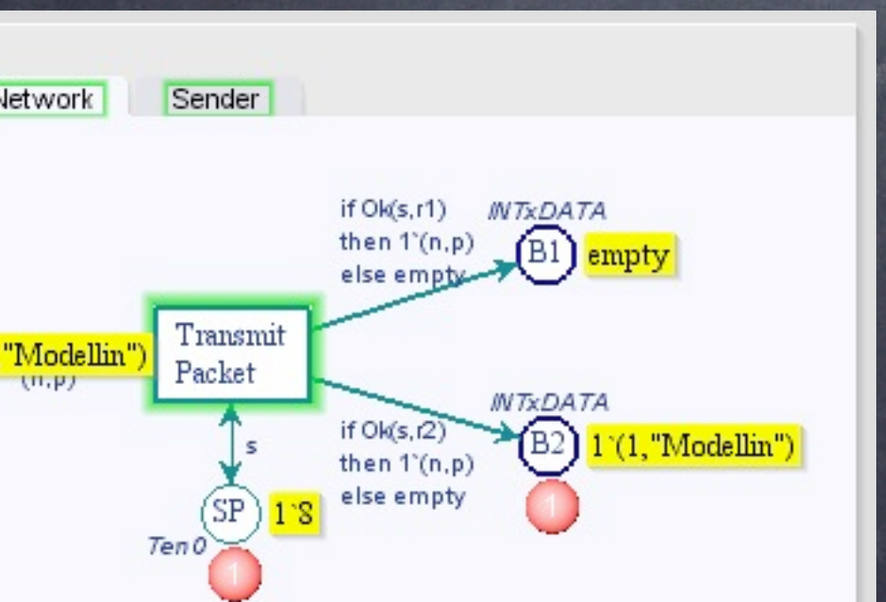
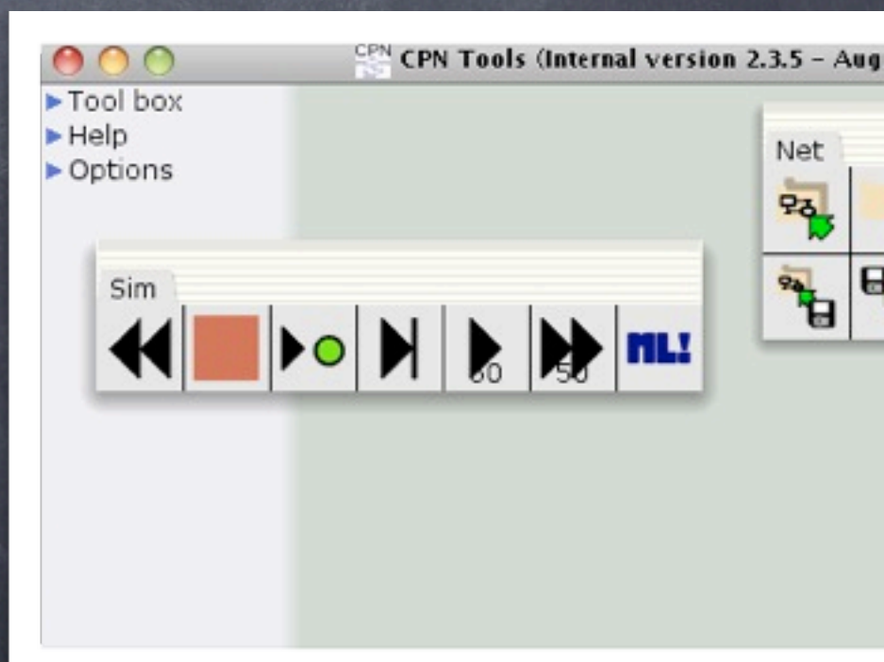


Stuff About CPN Tools

Michael Westergaard
Department of Computer Science
Aarhus University
mw@cs.au.dk

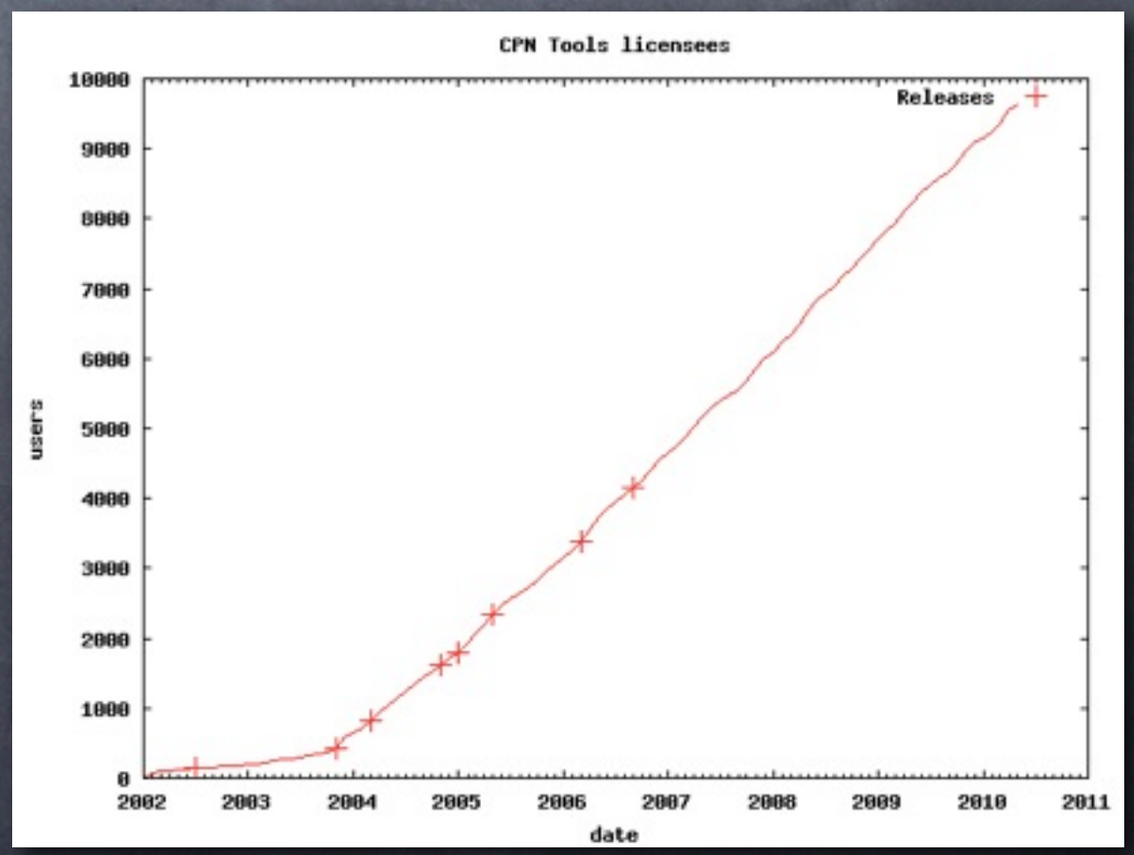




CPN Tools

- Started in 1998 to succeed Design/CPN
- Joint project between HCI, Beta, and CPN group

- Used by ~9500 users/ organizations (~600 commercial) in 142 countries



Me

- Started as student programmer on CPN Tools in January 2001
- Started as PhD student in August 2003, developing the BRITNeY Suite
- Started as PostDog in August 2007, working on ASAP and Access/CPN
- Has recently started working on making CPN Tools run on 64-bit Windows 7

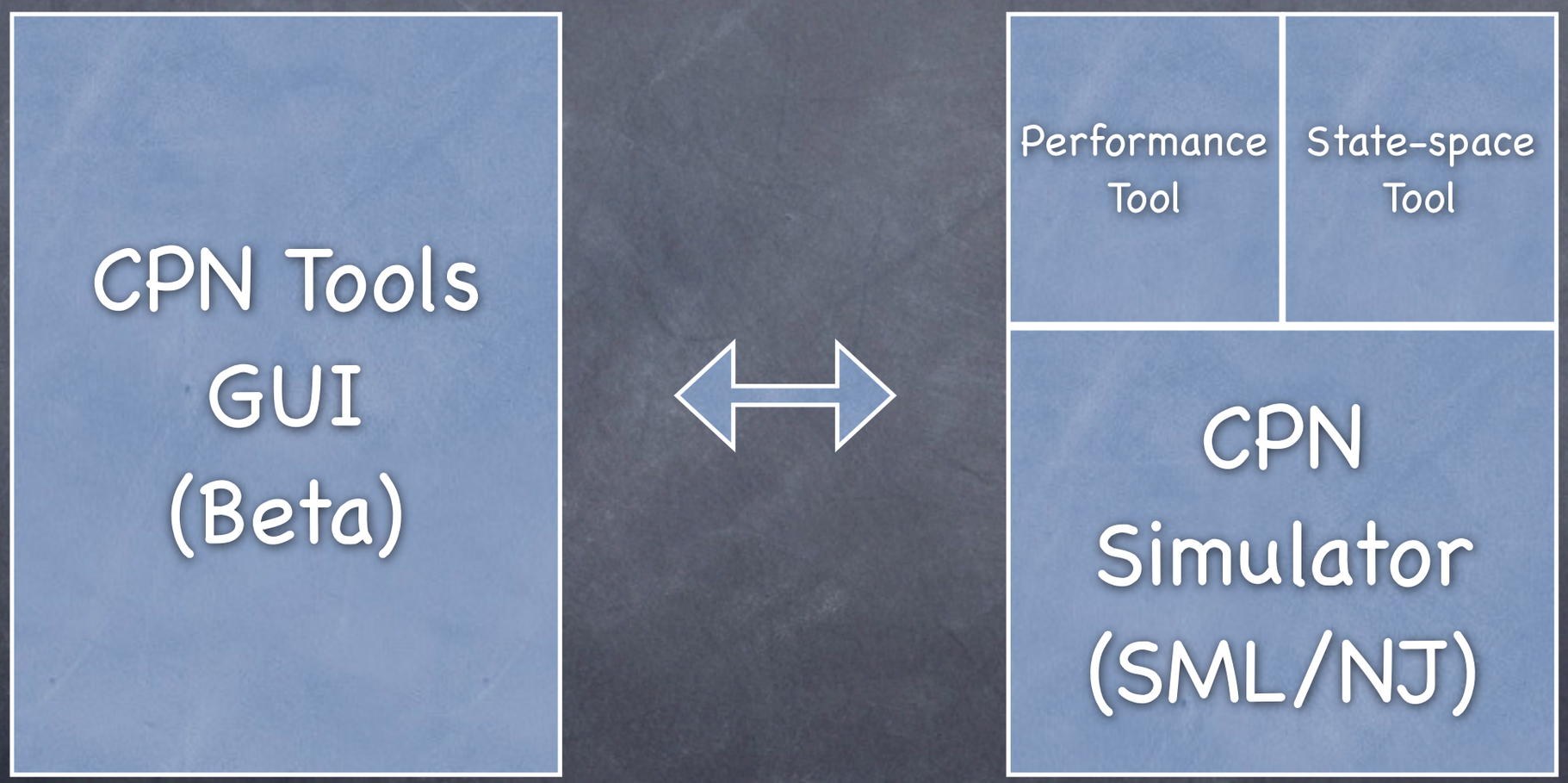
CPN Tools

CPN Tools

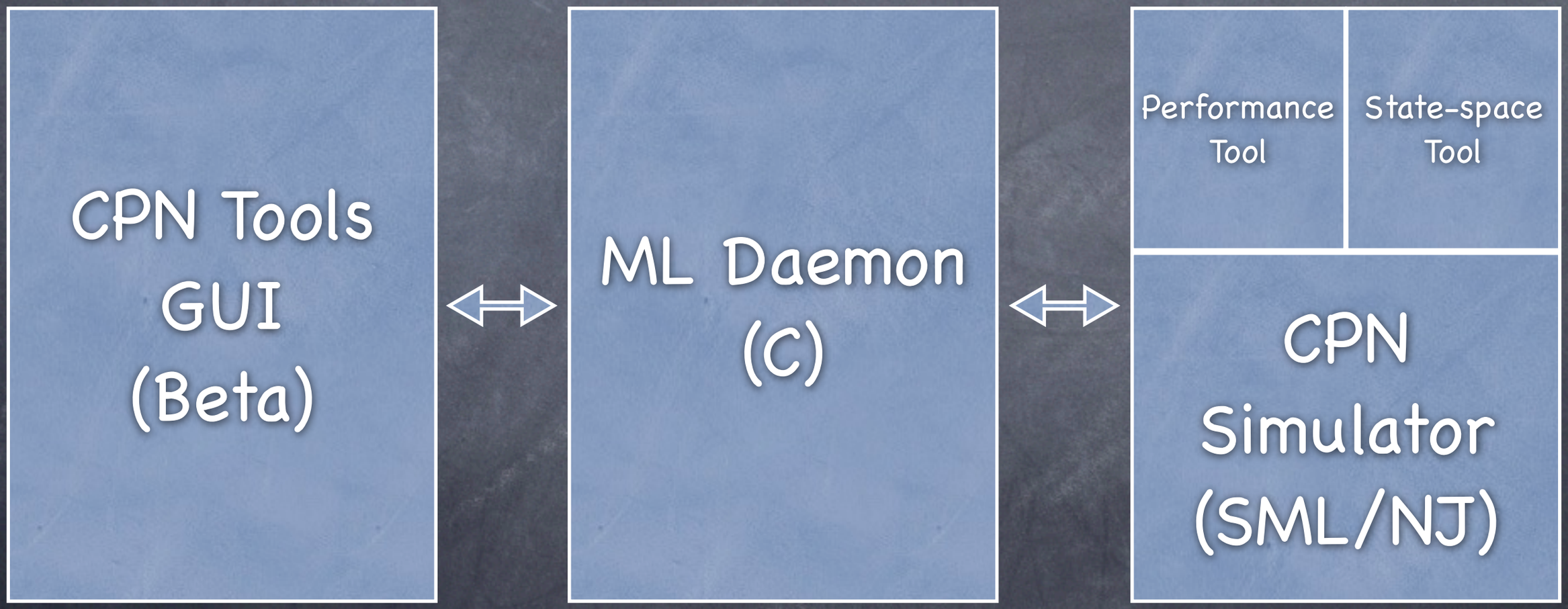
CPN Tools
GUI
(Beta)

CPN
Simulator
(SML/NJ)

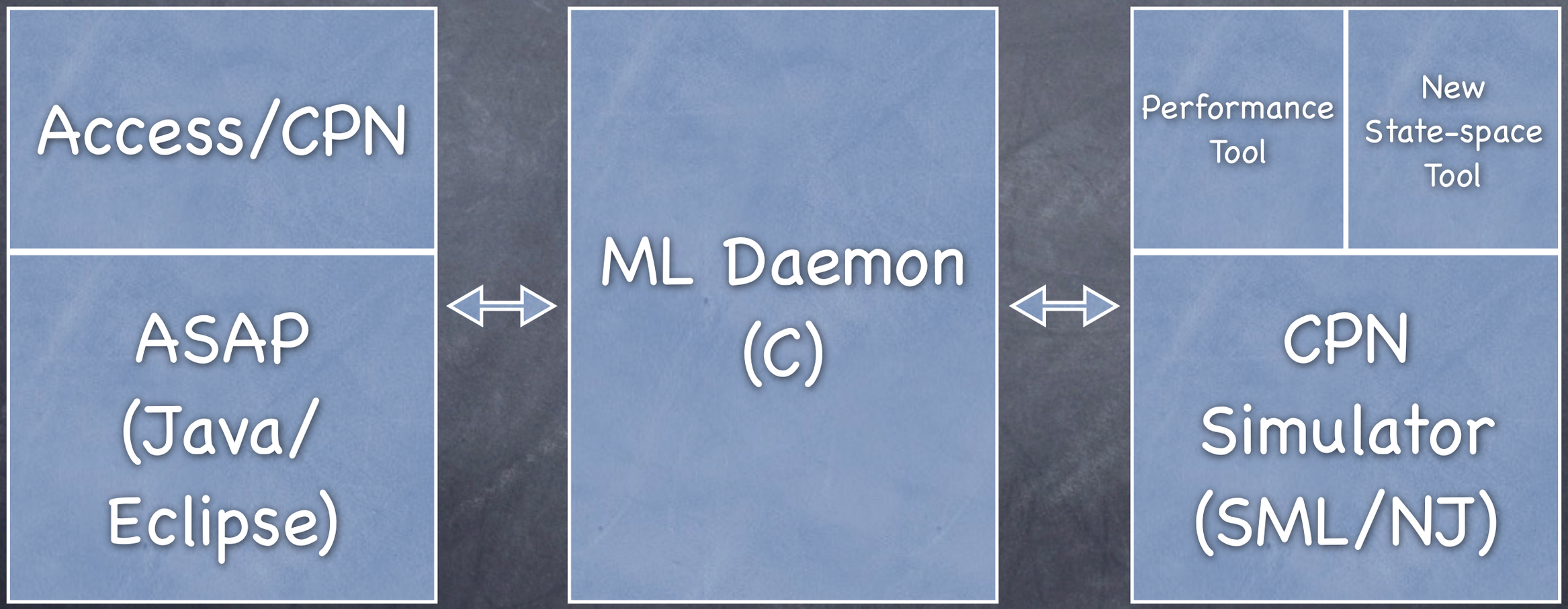
CPN Tools


CPN Tools
GUI
(Beta)
Performance
Tool
State-space
Tool
CPN
Simulator
(SML/NJ)

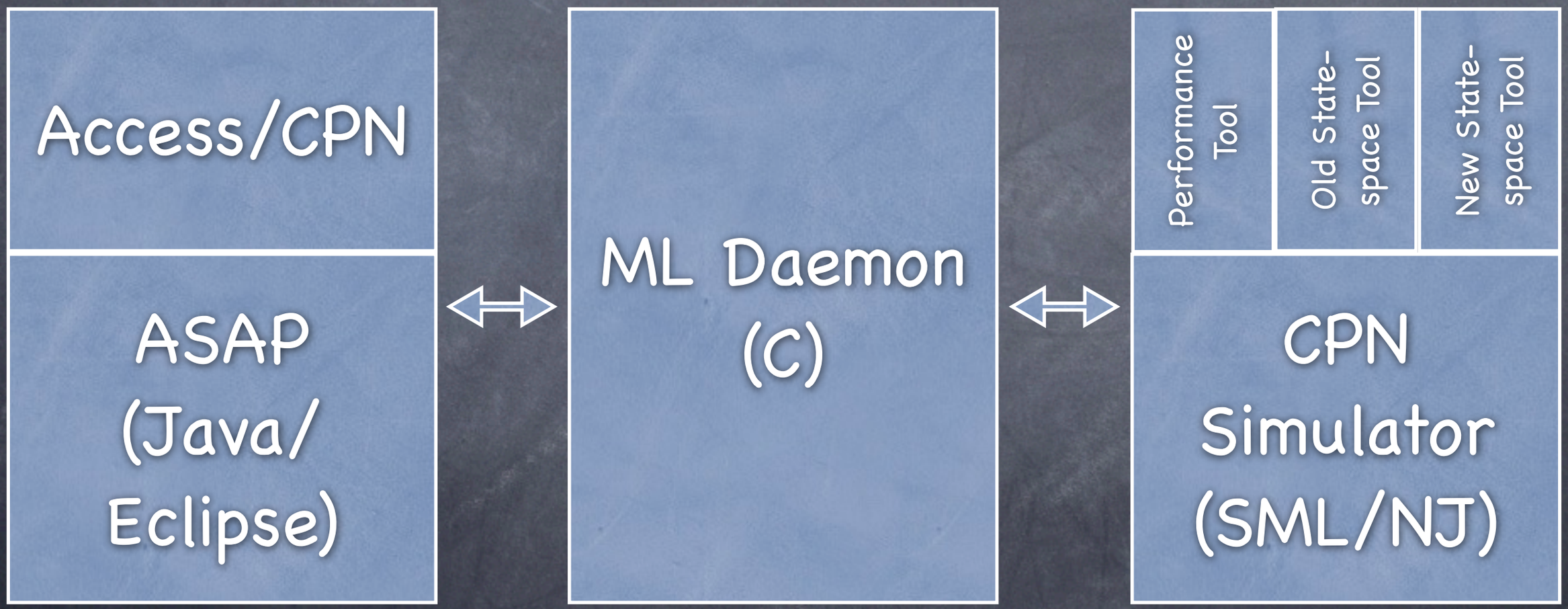
CPN Tools


CPN Tools
GUI
(Beta)
ML Daemon
(C)
Performance
Tool
State-space
Tool
CPN
Simulator
(SML/NJ)

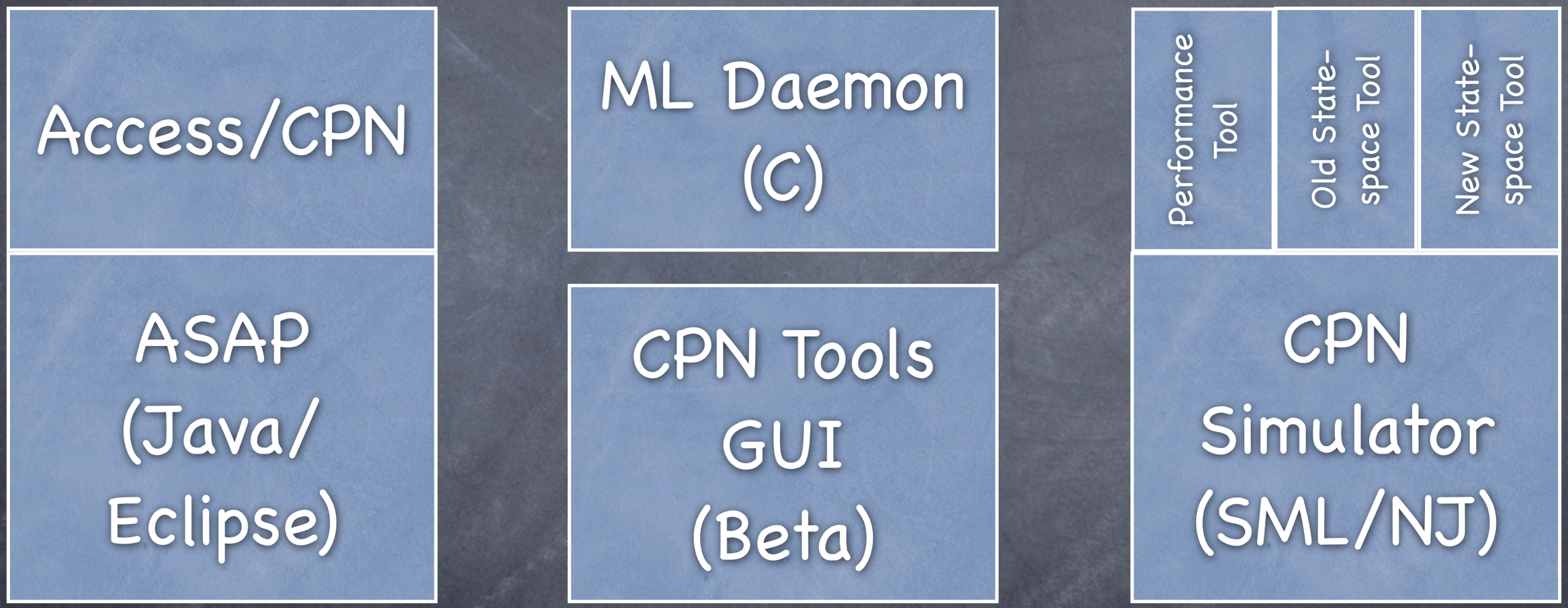
ASAP, Access/CPN



ASAP, Access/CPN



Full Picture



- Plus a few other tools/libraries: Comms/CPN, ASK-CTL, BRITNeY Suite, etc.

Focus Today

ML Daemon
(C)

Performance Tool

Old State-space Tool

New State-space Tool

CPN Tools
GUI
(Beta)

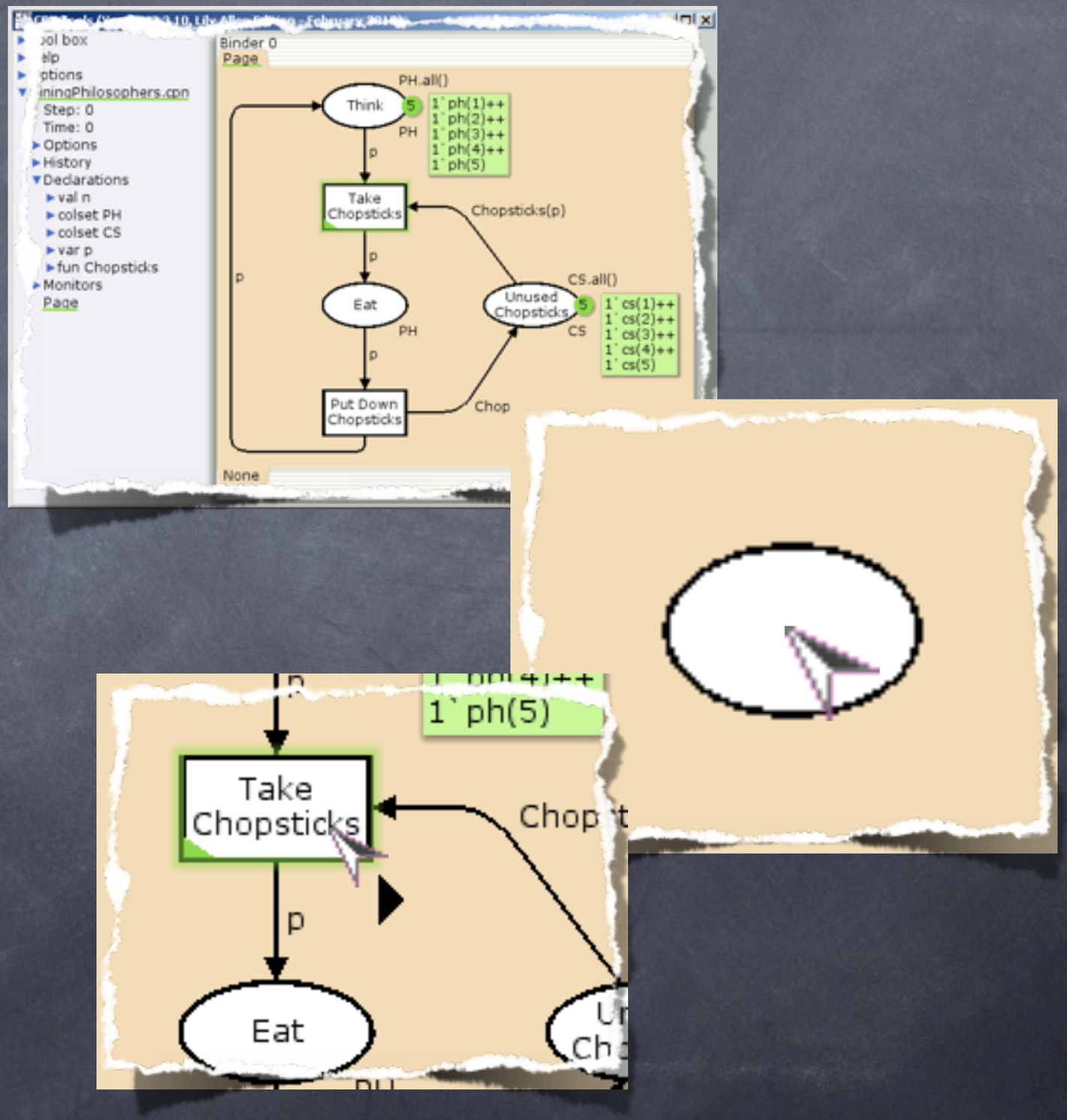
CPN Tools

CPN
Simulator
(SML/NJ)

Simulator

Example

- Load a net
- Add a place
- Simulate model



Beta

- Object-oriented language developed in Aarhus many years ago
- Abstracts objects and methods into patterns
- Has a intricate module system
- Is documented by a book which only tells part of the truth and sometimes also lies
- => not very easy to get started with
  - Also, it's no longer supported by the developer

Load a Net

- Generate data structure
- Instantiate simulator
- Show on screen

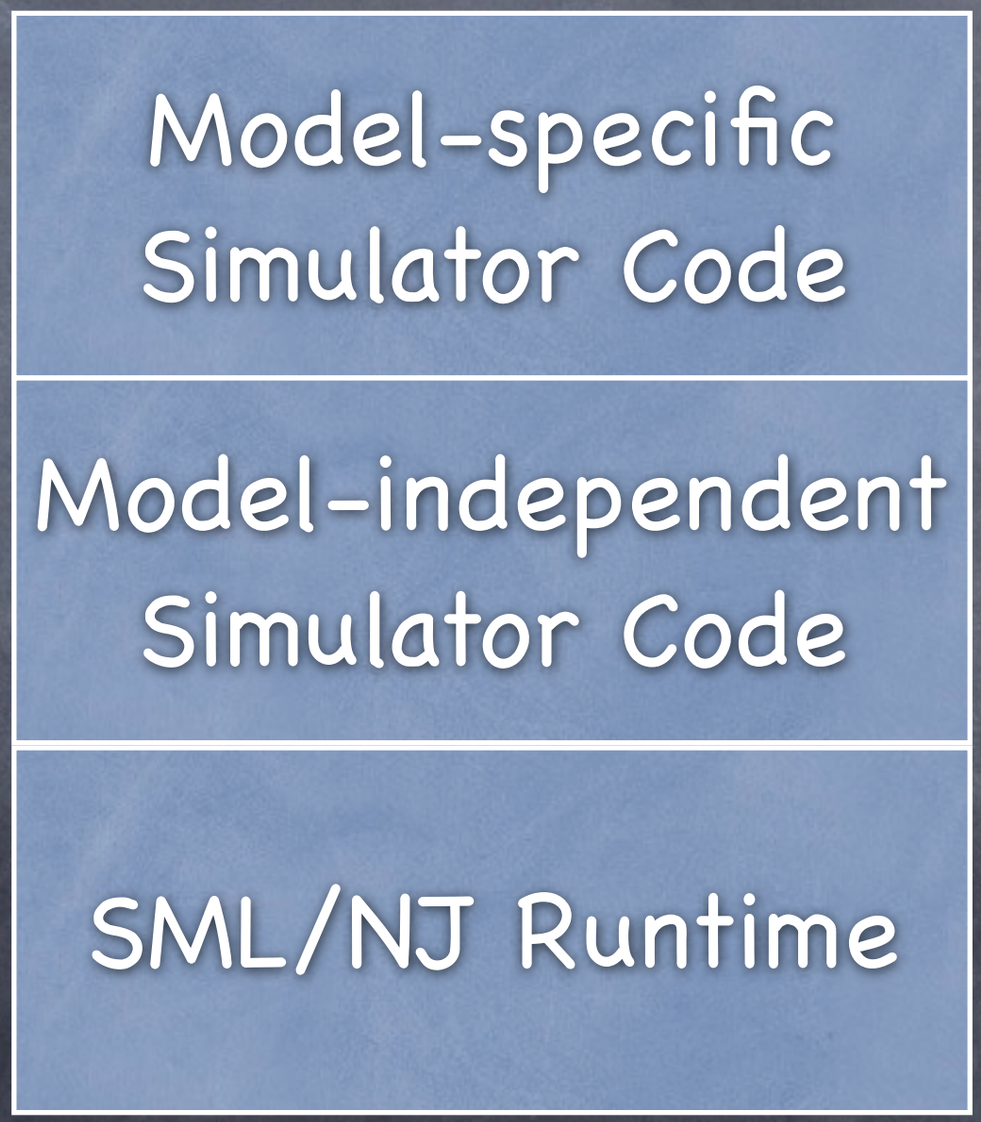
Data Structure

- Represent model using a data structure
- Structure (called cpnet or APN) representing places, transitions, arcs, declarations, ...
- Structured is mirrored for instances
- Implements observer, composite, and prototype design patterns

Data Structure

- Very tall hierarchy (no multiple inheritance/ interfaces/add-ins)
- Data structure also contains (nearly) all graphical properties
- The structure of .cpn files is hidden in and closely reflects this datatype as well

Simulator


Model-specific Simulator Code
Model-independent Simulator Code
SML/NJ Runtime

SML/NJ

- You probably know CPN-ML and basic SML/NJ
- SML/NJ also contains signatures, structures, and functors
  - These mostly correspond to interfaces, objects, and classes

Model-independent Simulator Code

- Main entry points: CPN'Sim, CPN'PageTable, CPN'PlaceTable, CPN'TransitionTable, CPN'InstTable
- Functors for colour-set types (CPN'UnitCS, CPN'IntCS, CPN'RealCS), time types (CPN'UnitTime, CPN'IntTime, CPN'IntInfTime, CPN'RealTime), etc.
- Functors for creating places, transitions, and declarations
- Loading/renaming library functions

Model-dependent Simulator Code

- For each colour-set, a structure and a type is created
- For each place and transition a structure is created
  - I.e., pages and arcs are not represented explicitly

Sho Scre

Node

NetNode

Binder

MarkingMenu (CPNMenu)

Tab

Marking

OptionNode

Workspace

Sheet/ Page

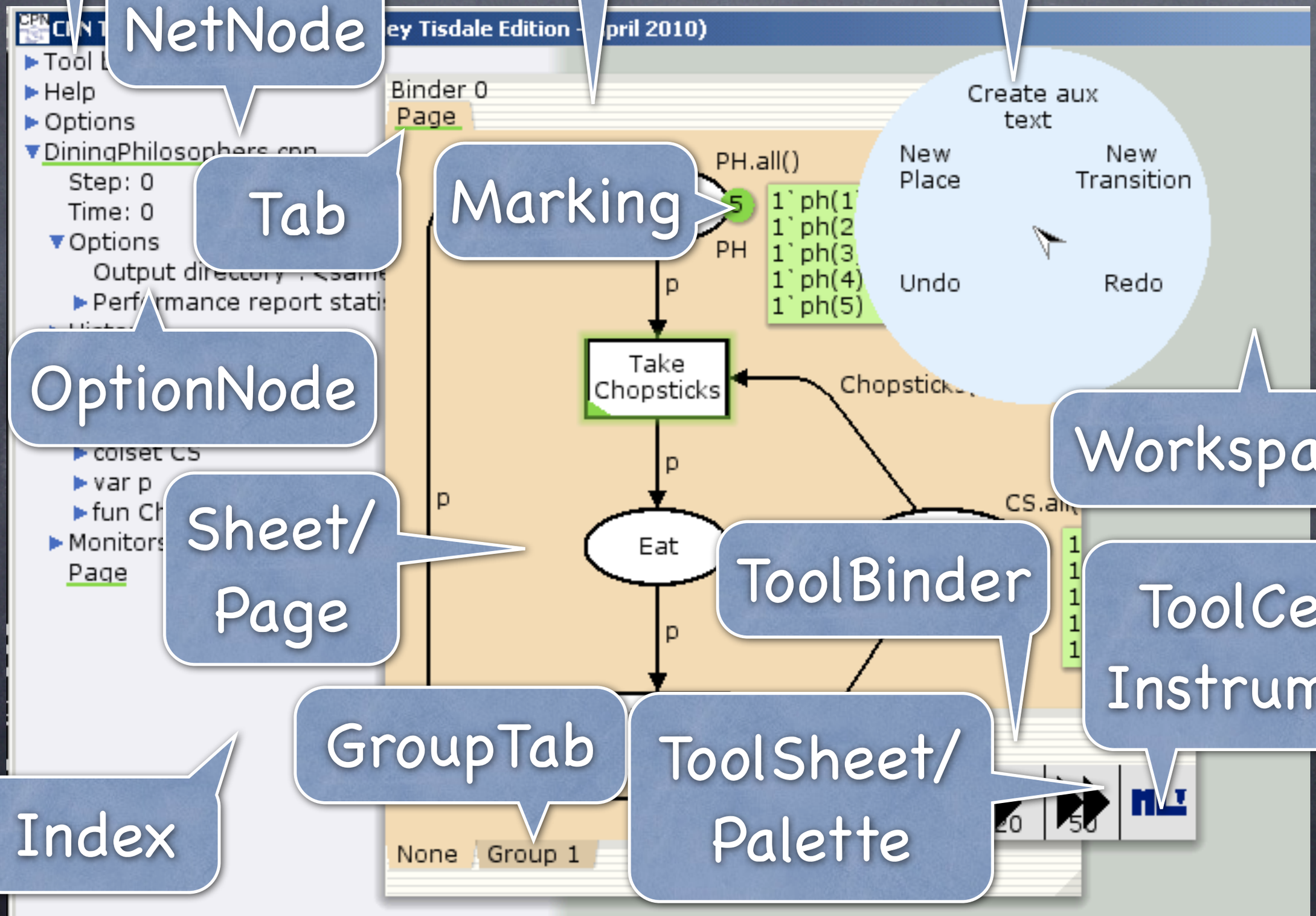
ToolBinder

ToolCell/ Instrument

Index

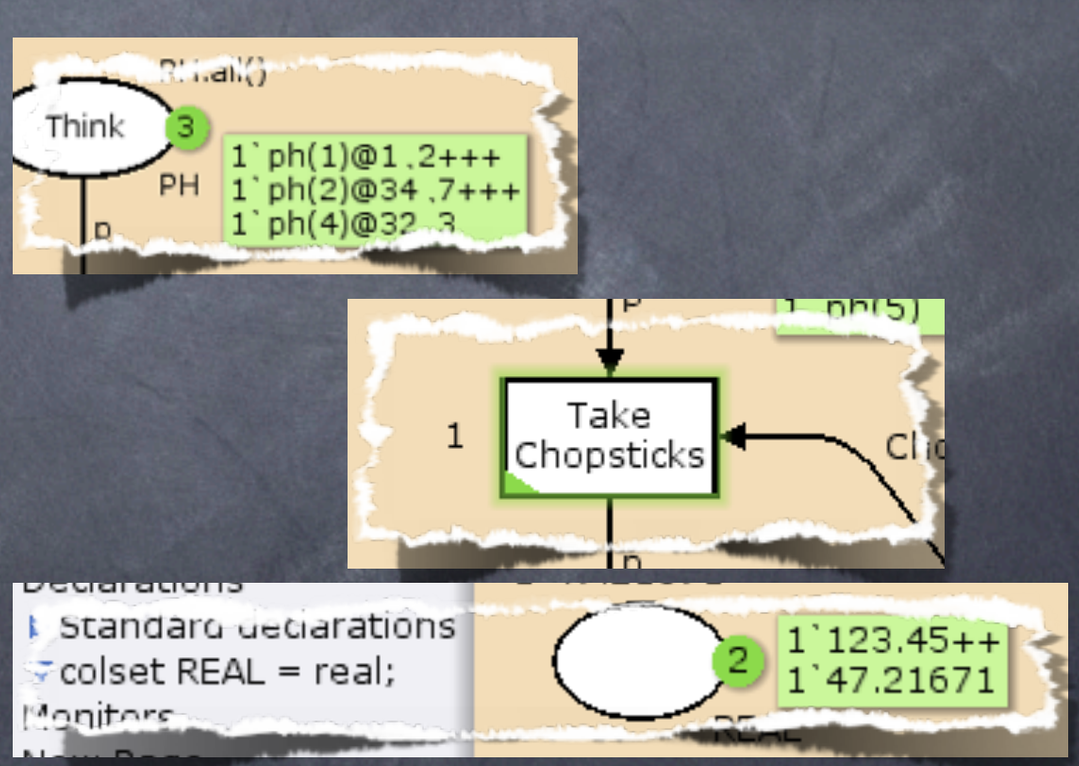
GroupTab

ToolSheet/ Palette



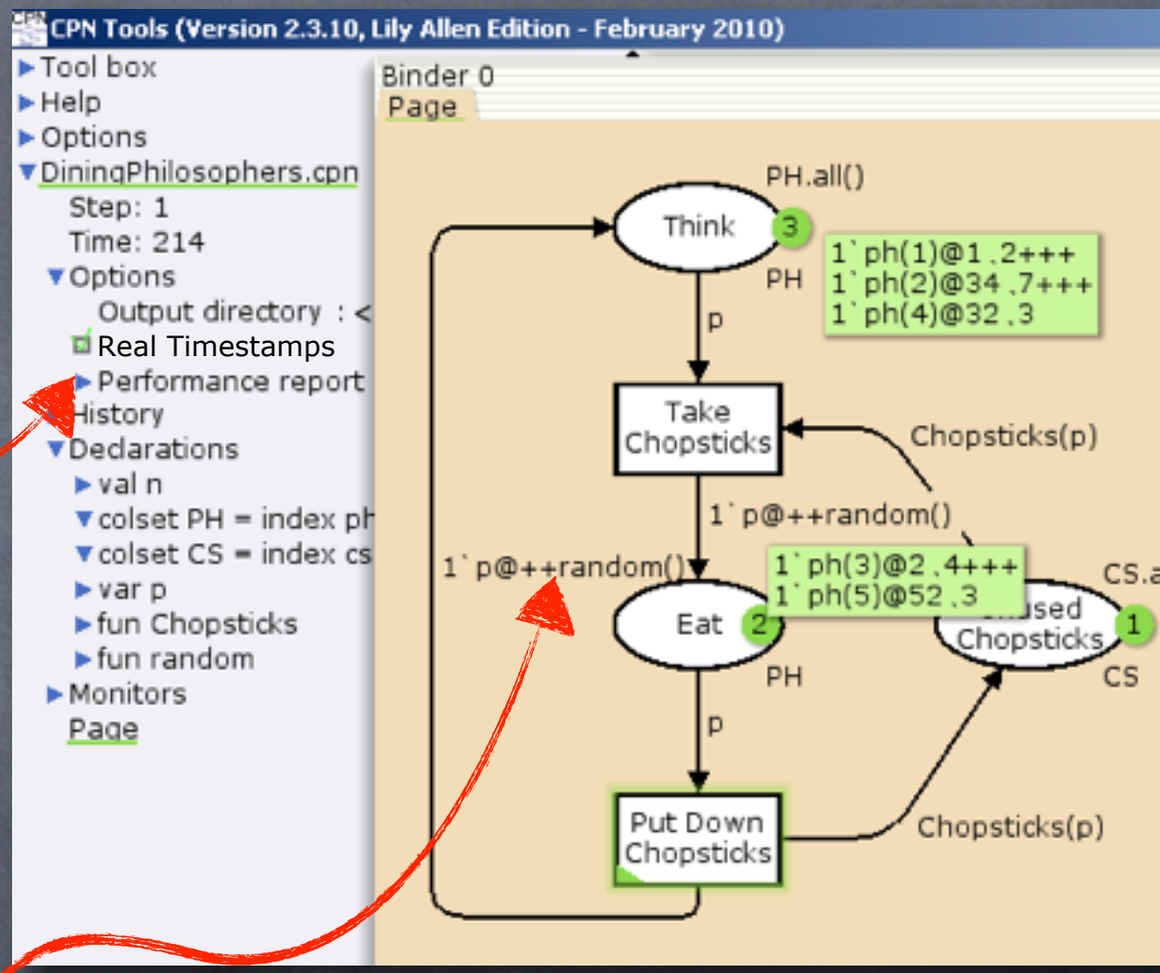
New Features

- This week I've worked with Eric Verbeek to implement
  - Real time-stamps
  - Prioritized transitions
  - Real colour-sets



Real Time-stamps

- Add option to models (including making it show up in the index)
- Use option to call simulator functions correctly
- Add @++ annotation
- Modify how transition code is generated



Add a Place

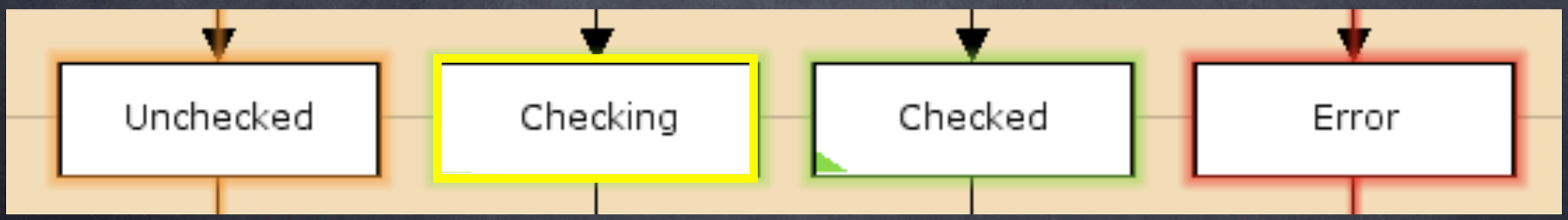
- Graphically adding the place
- Adding new place to the simulator

Tools in CPN Tools

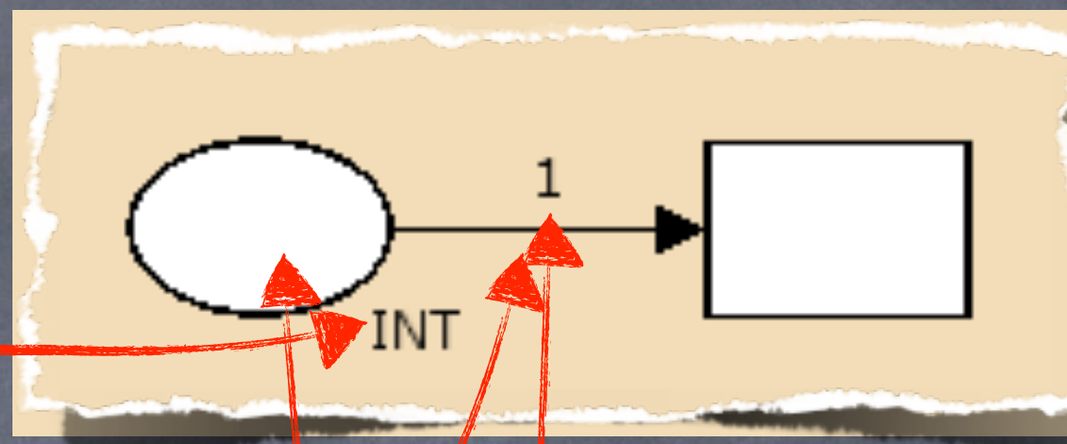
- Instrument: decides if an action is enabled and if so executes it
  - Examples: create place, open marking menu, select tool
- Command: actually performing actions
  - Create place, color red
  - Implements command design pattern
- Instruments can be added to tool palettes and marking menus

Adding to Simulator

- CPN Tools uses incremental syntax check
- It basically has a checker process whose task is to discover new elements and when appropriate send them to the simulator
- Elements can have one of the statuses: unchecked, checking, checked, or error

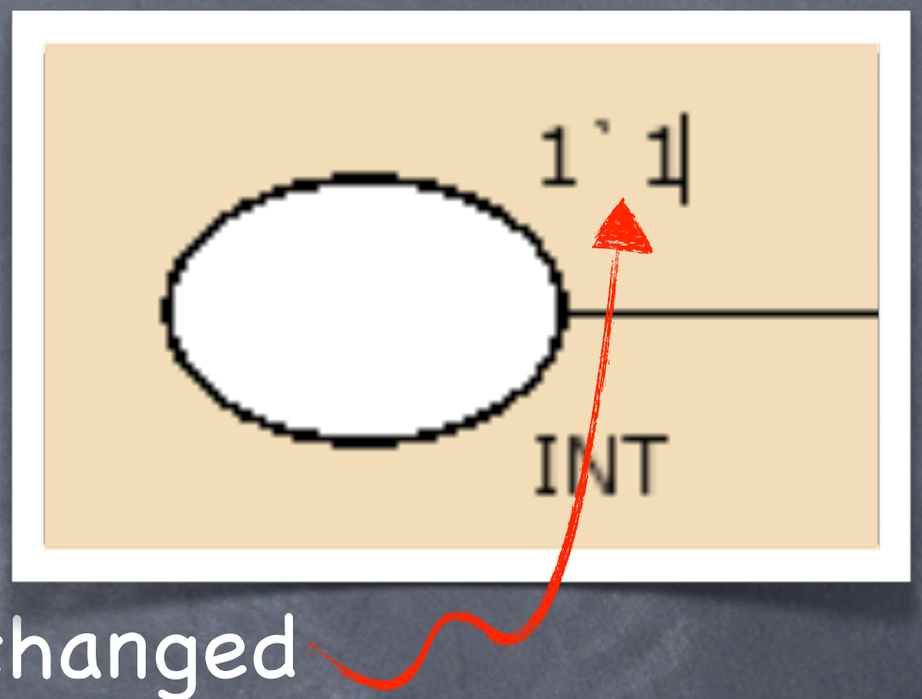


When to Check



- A place can be checked when
  - It has a colour-set
- A transition can be checked when
  - It has at least one arc
  - All surrounding arcs have inscriptions
  - All surrounding places have been checked
- For performance, neither are checked while being edited
- This check is performed by CPN Tools (GUI)

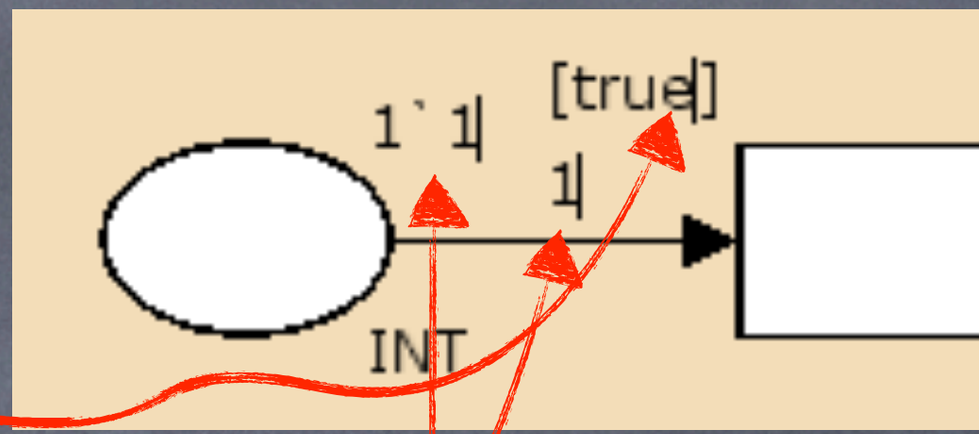
When to Re-check



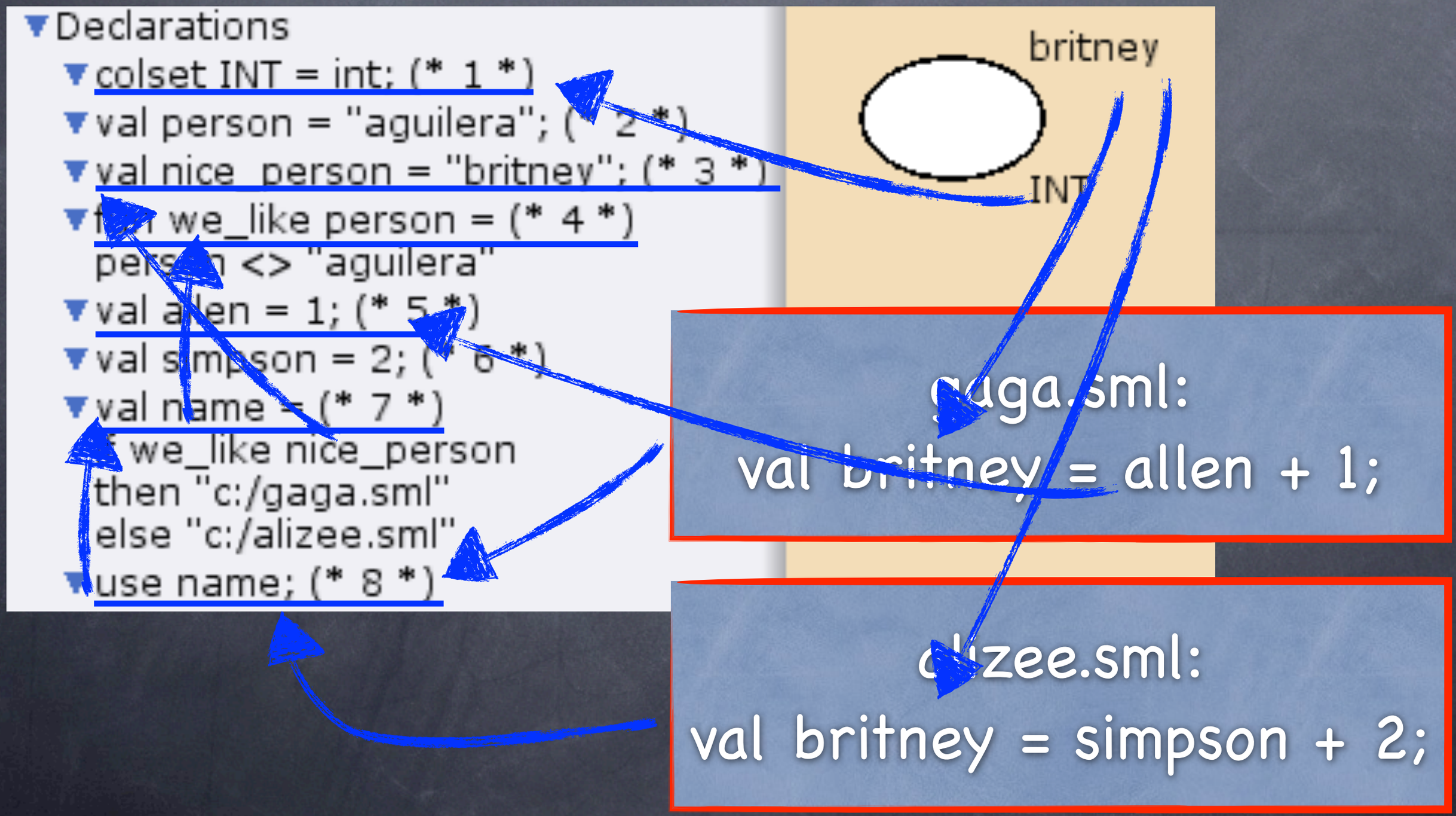
- A place is re-checked when
  - Any inscription has been changed
  - Any declaration we use has been changed
- The first is easily checked in CPN Tools

When to Re-check

- A transition is re-checked when
  - Any inscription is changed
  - Any surrounding arc has been added/removed/changed
  - Any surrounding place has been changed
  - Any declaration we use has been changed
- The first three are easily checked in CPN Tools



Declaration We Use?



Declarations We Use!

- Parsing SML is notoriously difficult
- But we already use the SML runtime...
- A bit of tinkering allows us access to the parser in SML/NJ
- Use the built-in parser to collect all symbols defined and used
  - We depend on declarations defining symbols we use

Simulate Model

- Sending commands to simulator
- Performing simulation
- Adding new commands

Communication With Simulator

- Communication is handled by a simple RPC protocol (called APN or DMO)
- We basically send packets in a format called BIS (boolean, integer, string)
- We encode the command and subcommand as integers and parameters depending on the command as booleans, integers, and strings

Communication with Si

Command: Simulation

- Example: Start a run:
  - B = nil
  - I = 500, 11
  - S = nil

Subcommand: Start run

- Result:
  - B = nil
  - I = 1
  - S = sim-step, sim-time, why-stopped-msg
- Source: tinyurl.com/dmo-protocol

Performing Simulation

- Two (three) methods
  - Automatic (fast forward)
  - Interactive (play, single step)
- The latter is done in CPN Tools (GUI) as
  - Find all enabled transition instances
  - Find a random of those
  - Execute it in a random binding
  - For single step the search area may be limited

Automatic Simulation

- Partition transition instances into
  - unknown: enabling unknown
  - disabled: known to be disabled
  - maybe_ready: token-enabled but not at the current time
- Pick a random transition from unknown and try executing it

Automatic Simulation

- If unknown is empty
  - Pick earliest available from maybe_ready
  - Increase time to time this is ready
  - Move this and all from maybe_ready that are ready at this time to unknown

Automatic Simulation

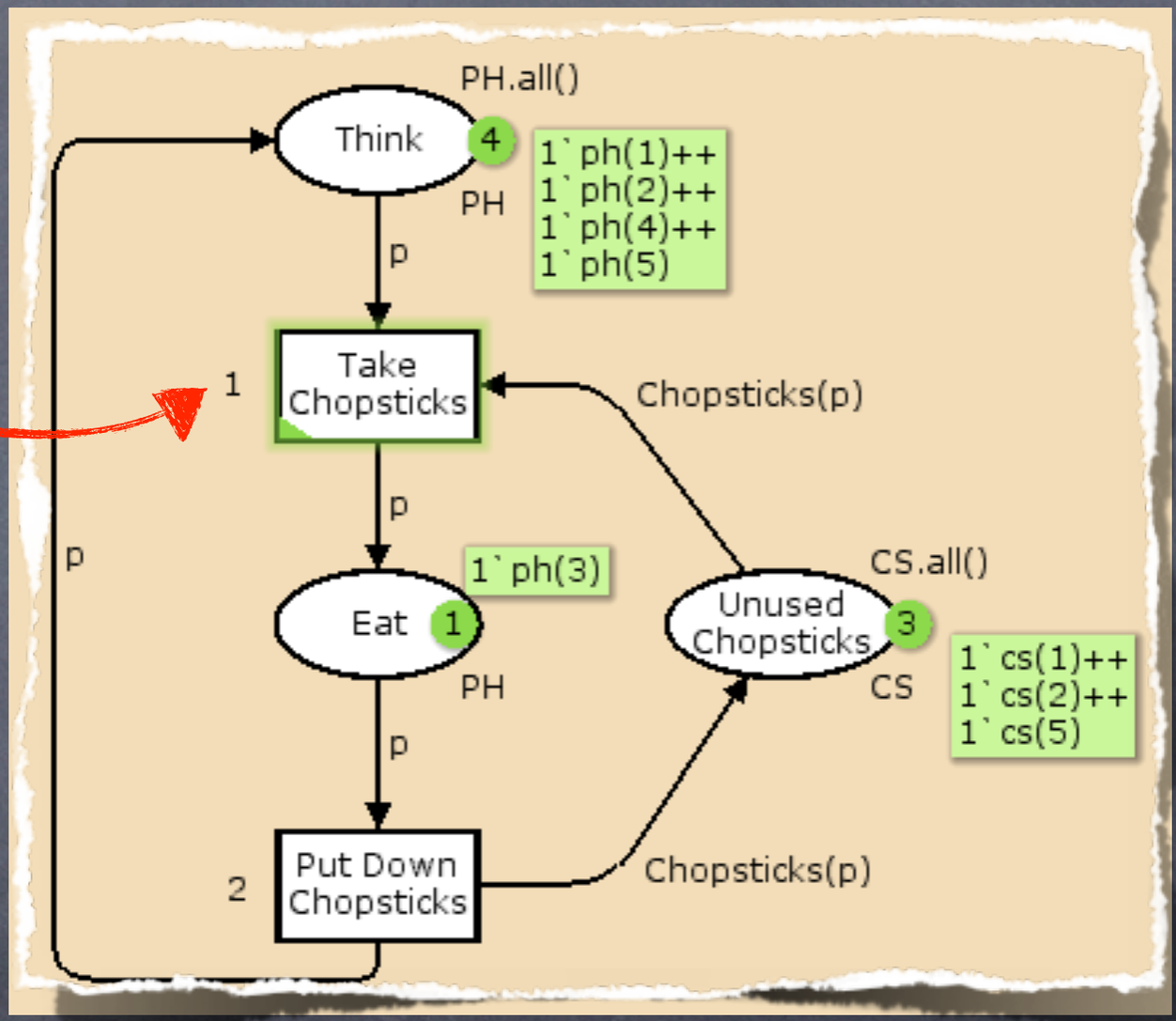
- If picked transition instance is disabled, move it to disabled; if it is token-enabled but not now, move to maybe_ready
- When executing a transition instance
  - Move all transitions connected to places, we produce tokens on, to unknown
- That is: enabling is only recalculated depending on tokens!

Adding New Commands

- To add a new command, we mush create
  - a stub in CPN Tools (mlcommands)
  - an entry in the dispatcher (simglue)
- The dispatcher is an excellent entry-point to find out how things are done in the simulator

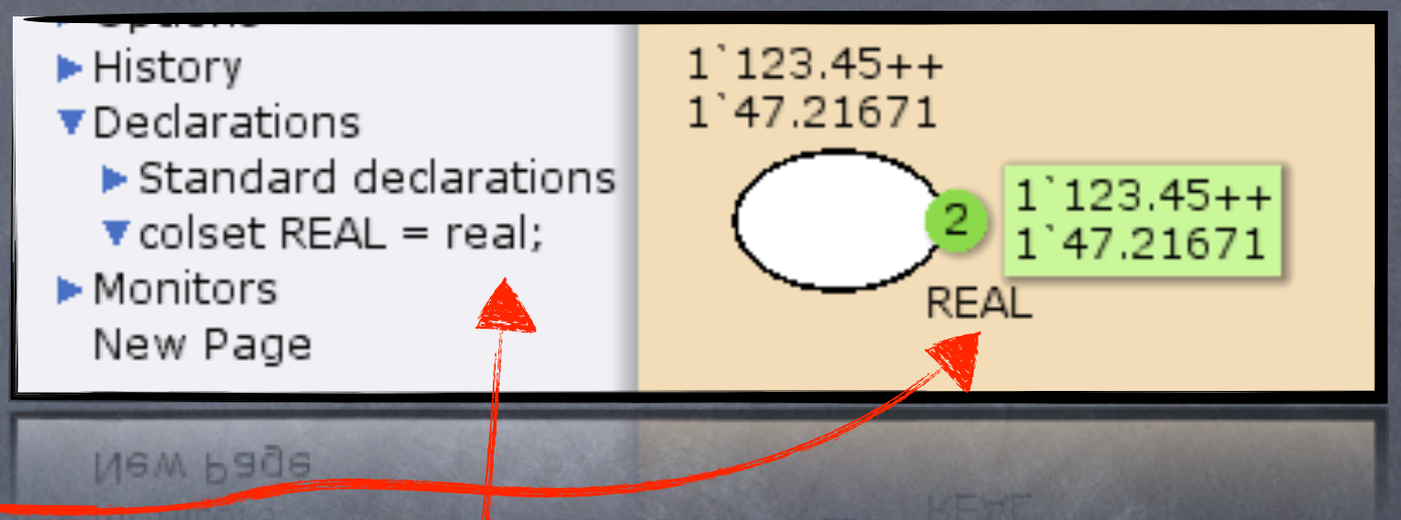
Prioritized Transitions

- Add new inscription to transition data structure (including saving it)
- Modify schedulers
  - Interactive simulation
  - Automatic simulation
- Todo: state-space analysis?



Real Colour-sets

- Add support for the type (mostly there, "just" need to make real an equality type in the simulator)



- Add type to protocol (both in the simulator and in CPN Tools)
- Add type to declaration parser in CPN Tools (really remove inhibitor)

CPN Tools GUI

Data-structure

Main program, logging, etc.

Resources

cpnet
cpntools
cpntools/cursors, fonts, images, Language, templates
cpntools/external
cpntools/instruments
cpntools/menues
cpntools/palettes
cpntools/resources
cpntools/resources/figures
cpntools/resources/text
cpntools/wselements
document/ml

Instruments, commands, marking menus, tool palettes

Specialized GUI elements

Abstraction of APN protocol

CPN Tools Simulator

APN protocol implementation

Comms/CPN

com
comms
patch

Patch for SML/NJ required for incremental syntax check

rmi

BRITNeY interface

sim

Main simulator

sim/perf

sim/ReportStuff

Performance tool and (parts of) old SS tool

Thank You for Your Attention

- Online version of slides: westergaard.eu/cv/other/