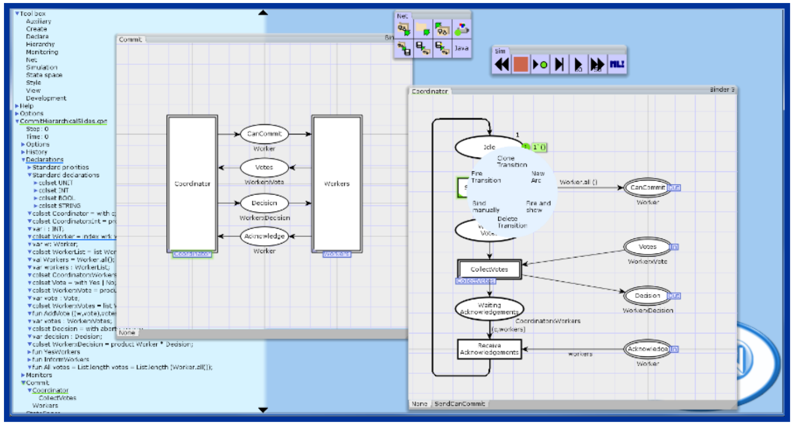
Lecture 4

Hierarchical Coloured Petri Nets with Modules



Lars M. Kristensen
Department of Computing, Mathematics, and Physics
Western Norway University of Applied Sciences
Email: lmkr@hvl.no / WWW: home.hib.no/ansatte/lmkr

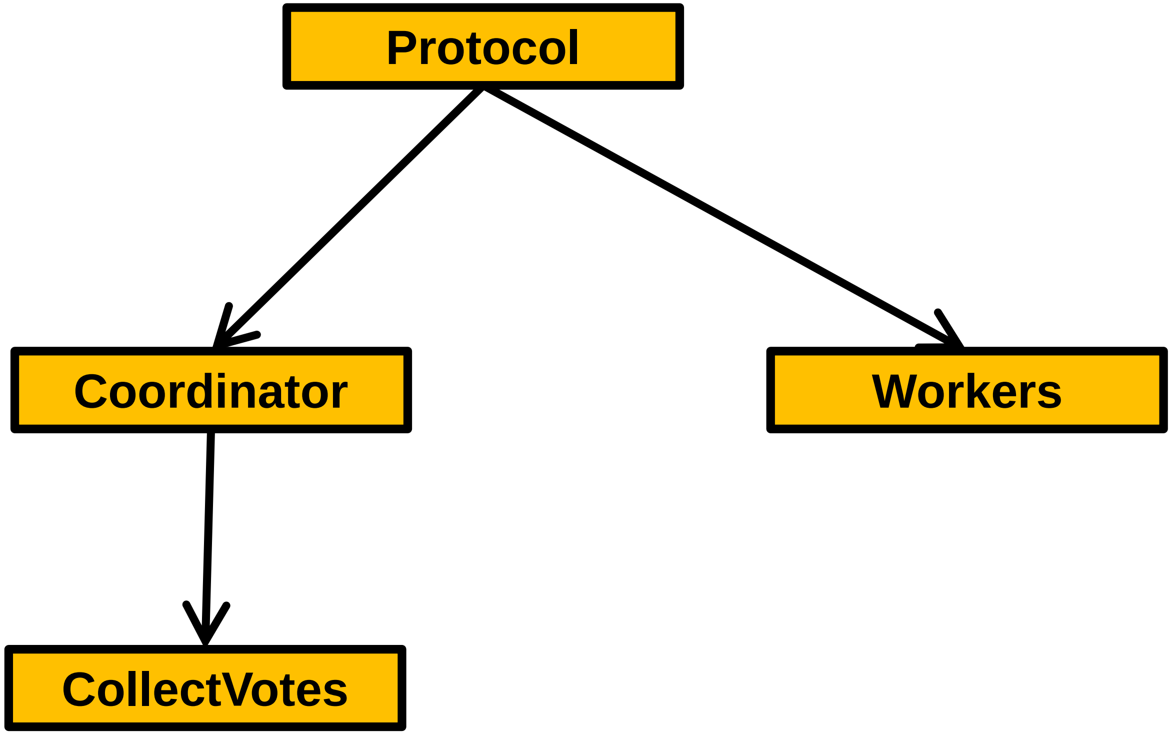
Introduction

- **Important to be able to split a large CPN models into a set of modules with interfaces**
  - To support construction of large CPN models
  - To support reuse of modules and maintainability
  - To support abstraction and management of details

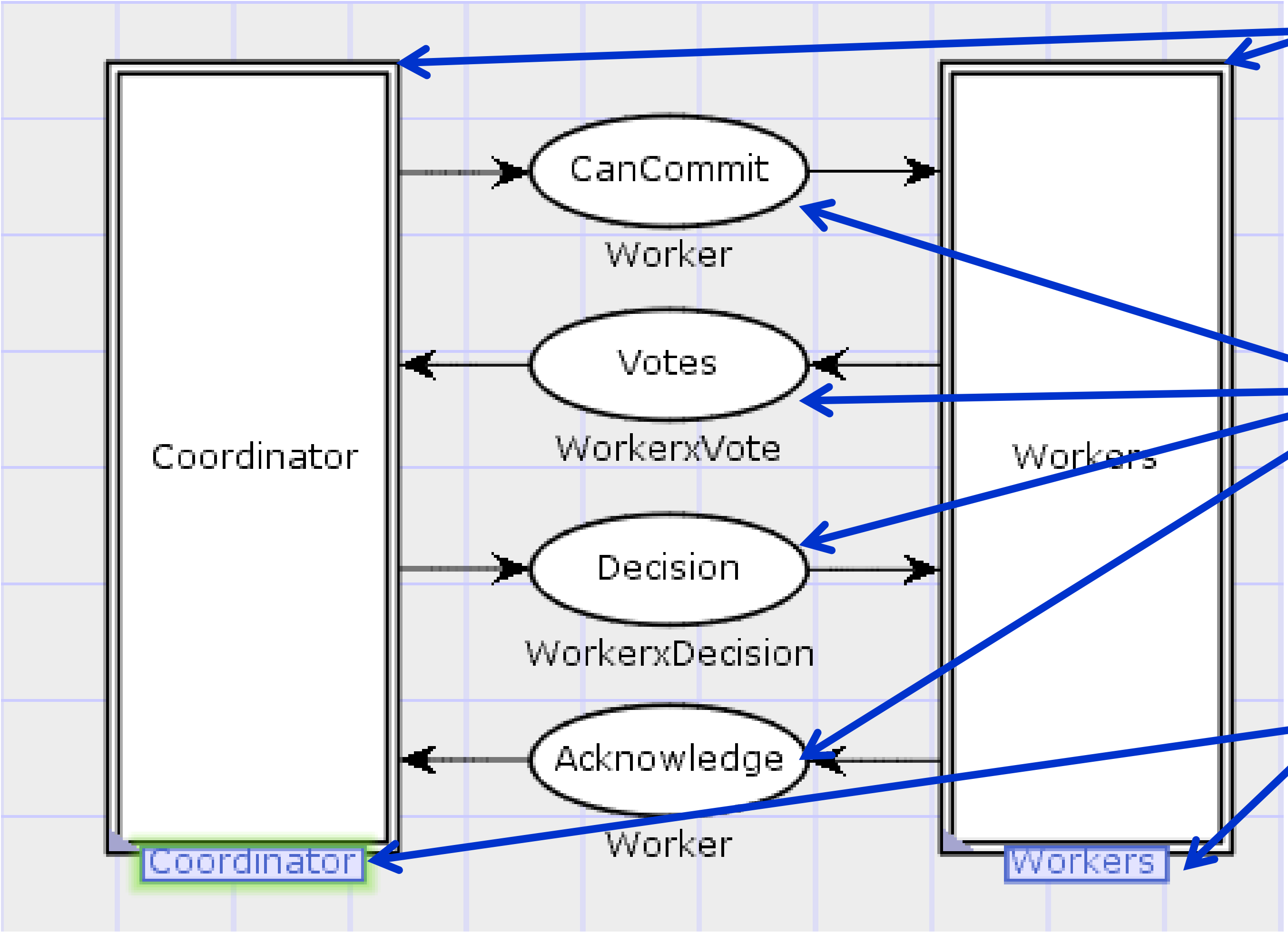
- **Key concepts of hierarchical CPN modules**
  - A module exchange tokens with its environment using input/output port places
  - Substitution transitions have associated submodules
  - Port-socket relation associates socket places of substitution transitions with the port places in the associated submodule

Hierarchical Modules

- **Model is comprised of a collection of modules that are hierarchically organised into levels**
- **Example: the two-phase commit protocol**



Top-level: Protocol Module

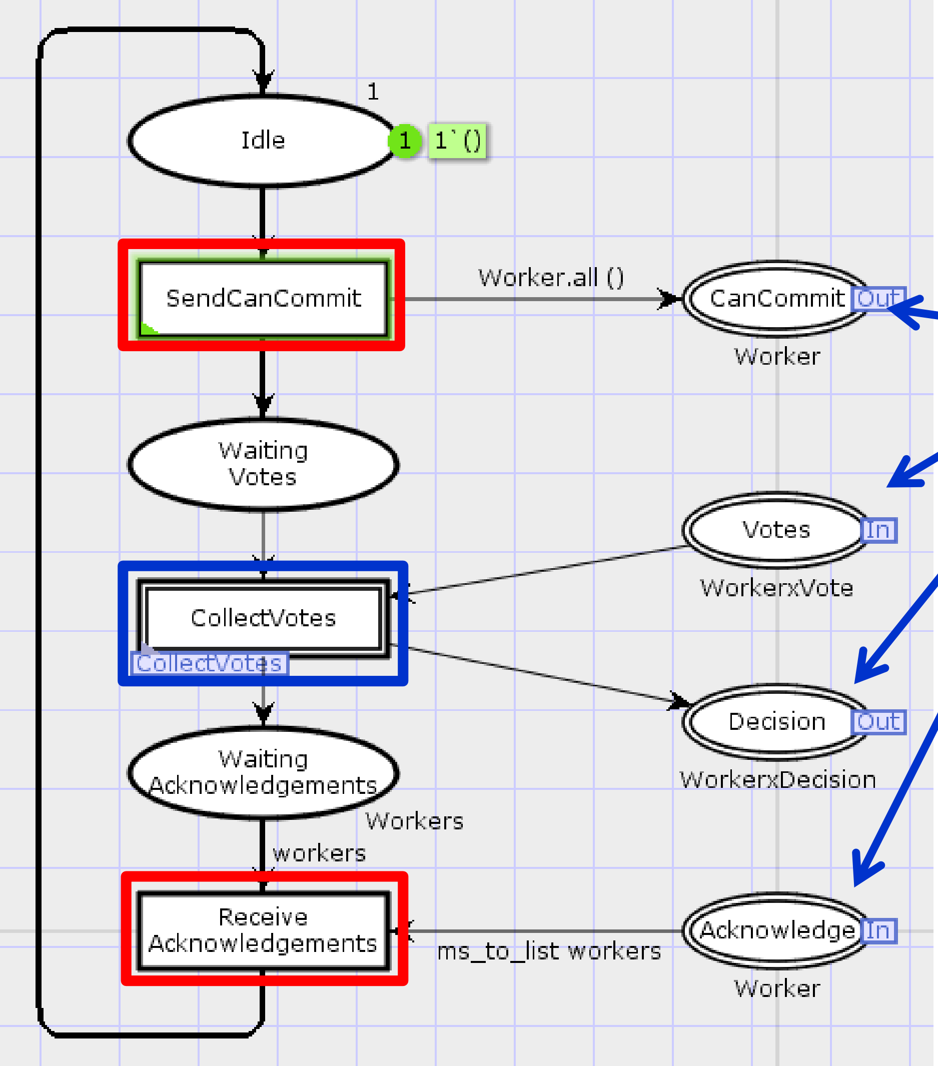


Substitution transitions (double-lined border)

Socket places – places connected to substitution transitions

Substitution tags - name of submodule associated with substitution transition

Coordinator Module



Port place - used for exchanging tokens with the upper-level module (IN,OUT,IN/OUT).

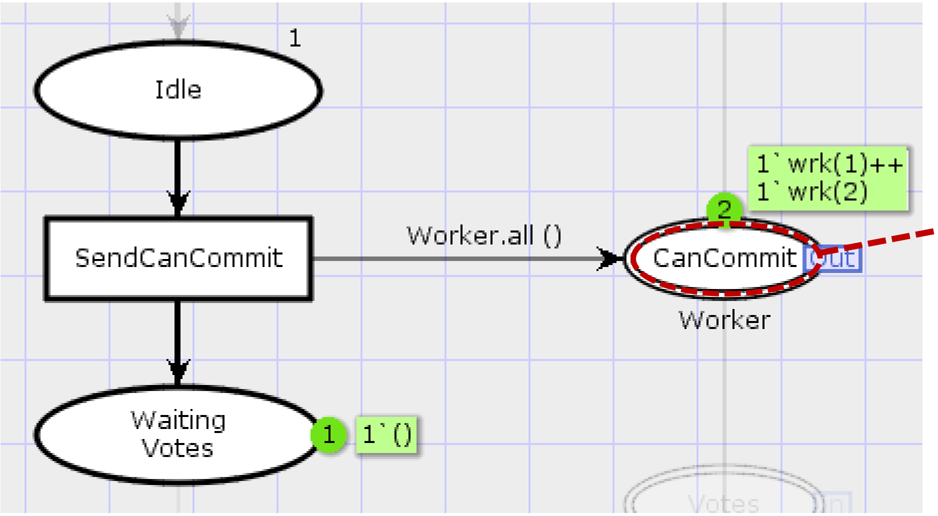
SendCanCommit and ReceiveAcknowledgement are ordinary transitions.

CollectVotes is a substitution transition

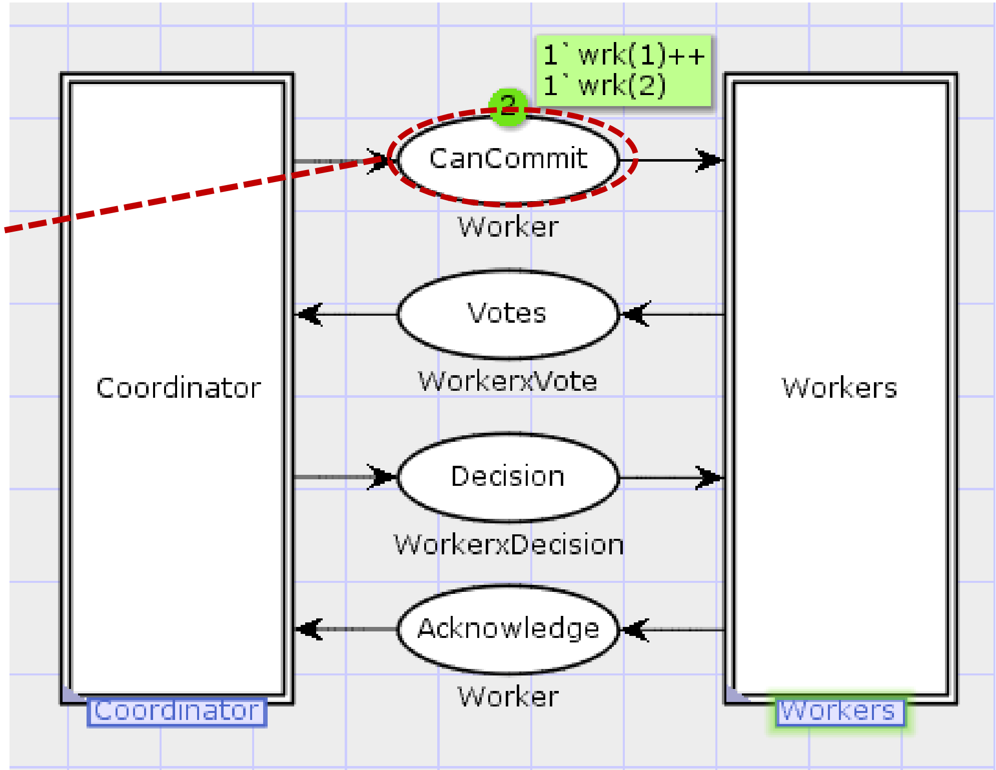
Port-Socket Place Relation

- Tokens added (removed) on a port place are added (removed) on the associated socket place

Coordinator module



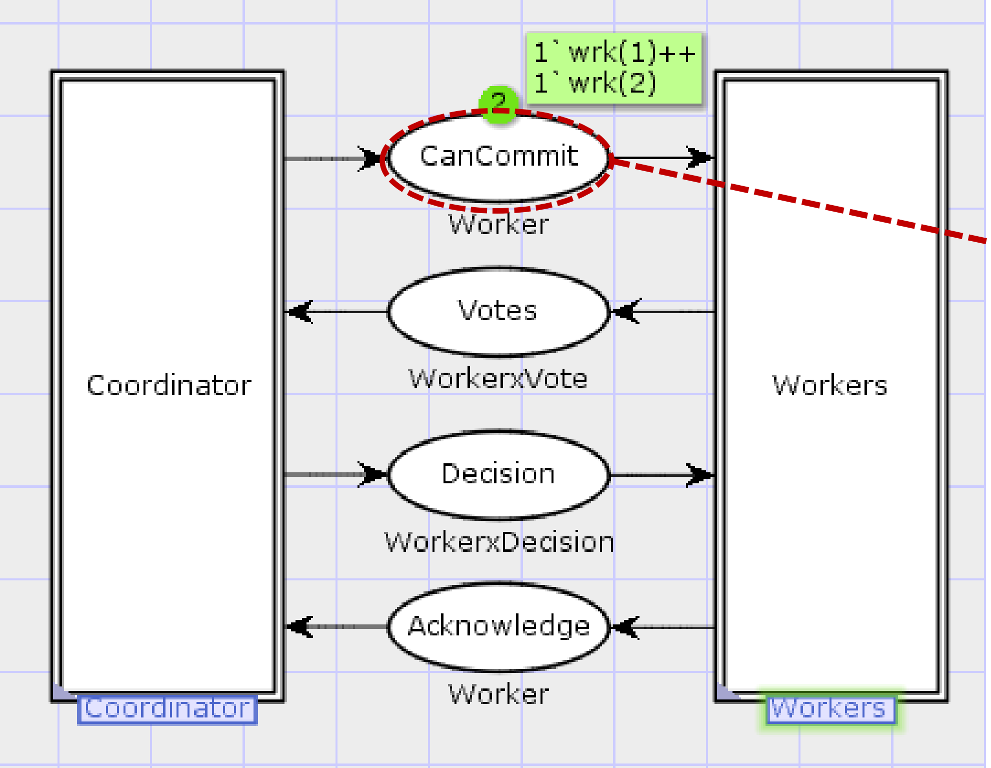
Protocol module

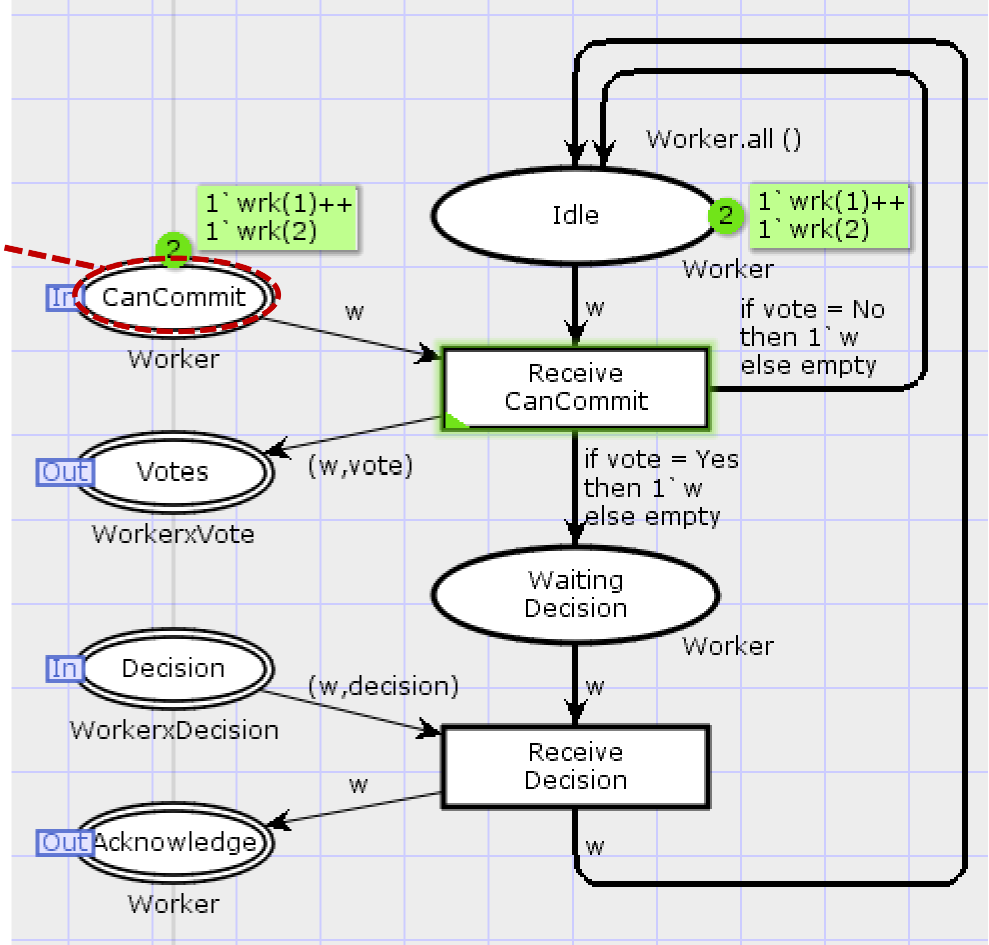


- Associated port and socket places constitute a compound place.

Workers Module

Protocol module | **Workers module**



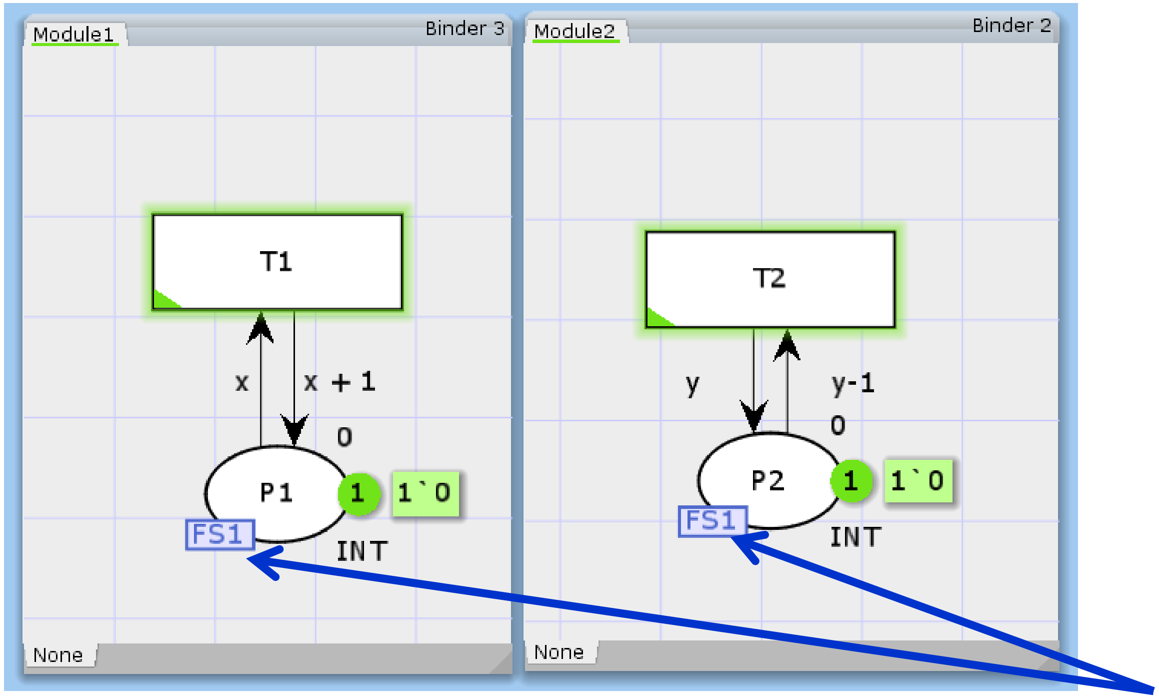


CPN Tools Demo

- **Hierarchical CPN models**
  - Navigating hierarchical models
  - Simulation of hierarchical models
  - Editing of modules: top-down and buttom-up development

Place Fusion Sets

- **Group of places to be treated as one compound (global) place**



Any change in the marking of P1 will be reflected on P2 (and vice versa).

Similar to global variables - should be used with care

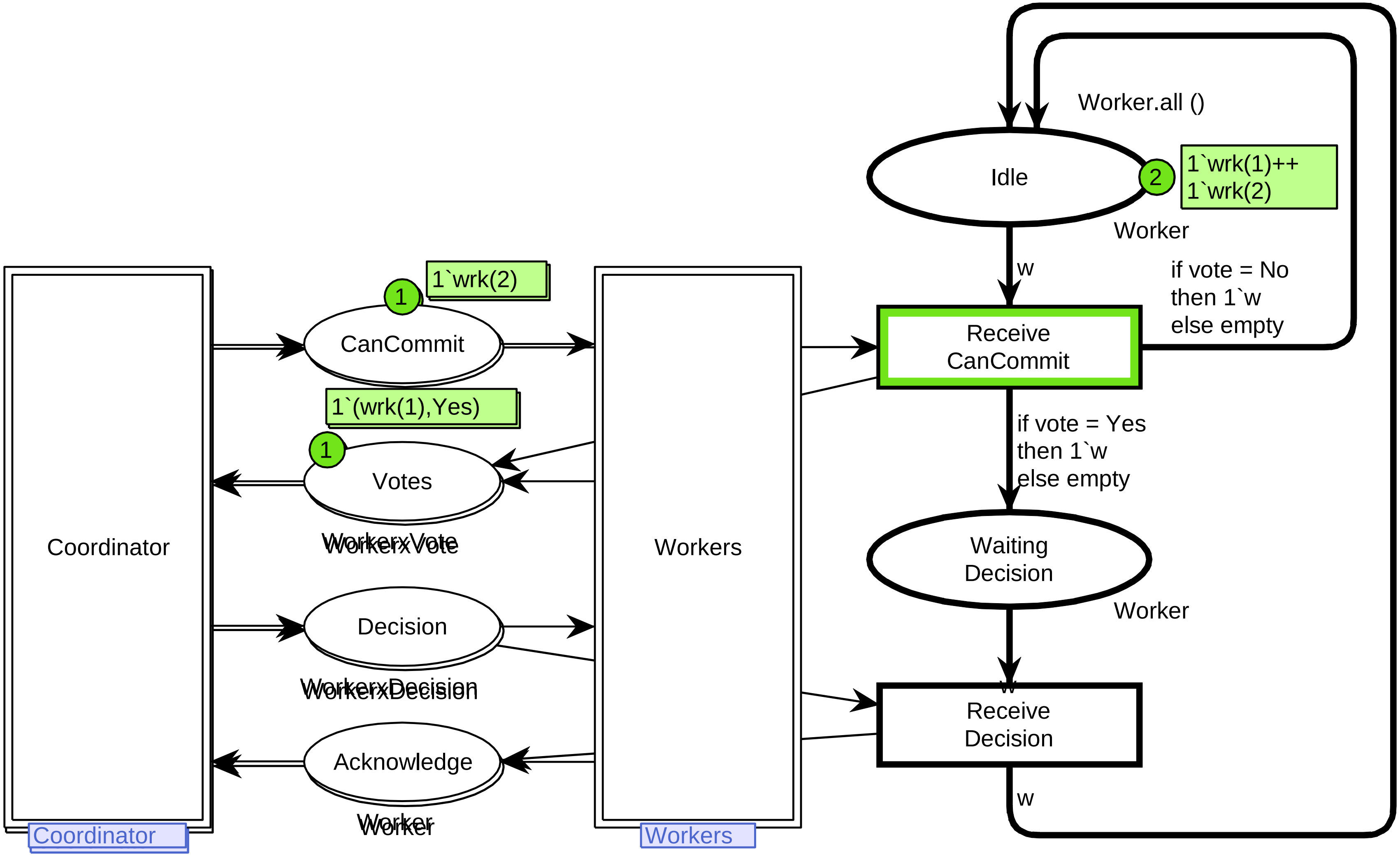
P1 and P2 are fusion places belonging to fusion set FS1.

Unfolding Coloured Petri Nets to Place/Transition Nets

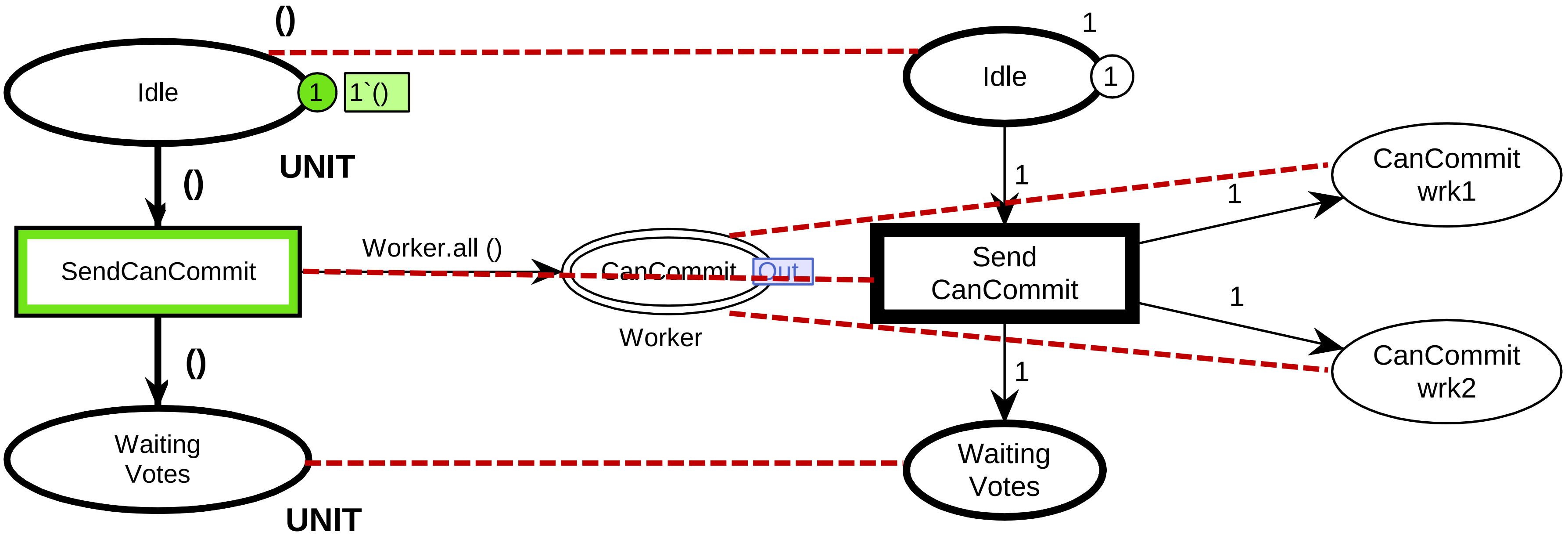
Unfolding Coloured Petri Nets

- **A hierarchical CPN model can be unfolded to a non-hierarchical Coloured Petri Net**
  - Recursively replace each substitution transition with its associated submodule
  - Associated port and socket places are merged into a single place
- **A non-hierarchical Coloured Petri Net can be unfolded into a Place/Transition Net (PTN)**
  - Replace each CPN place with one PTN place for each colour in the colour set of the CPN place
  - Replace each CPN transition with one PTN transition for each possible binding of the CPN transition

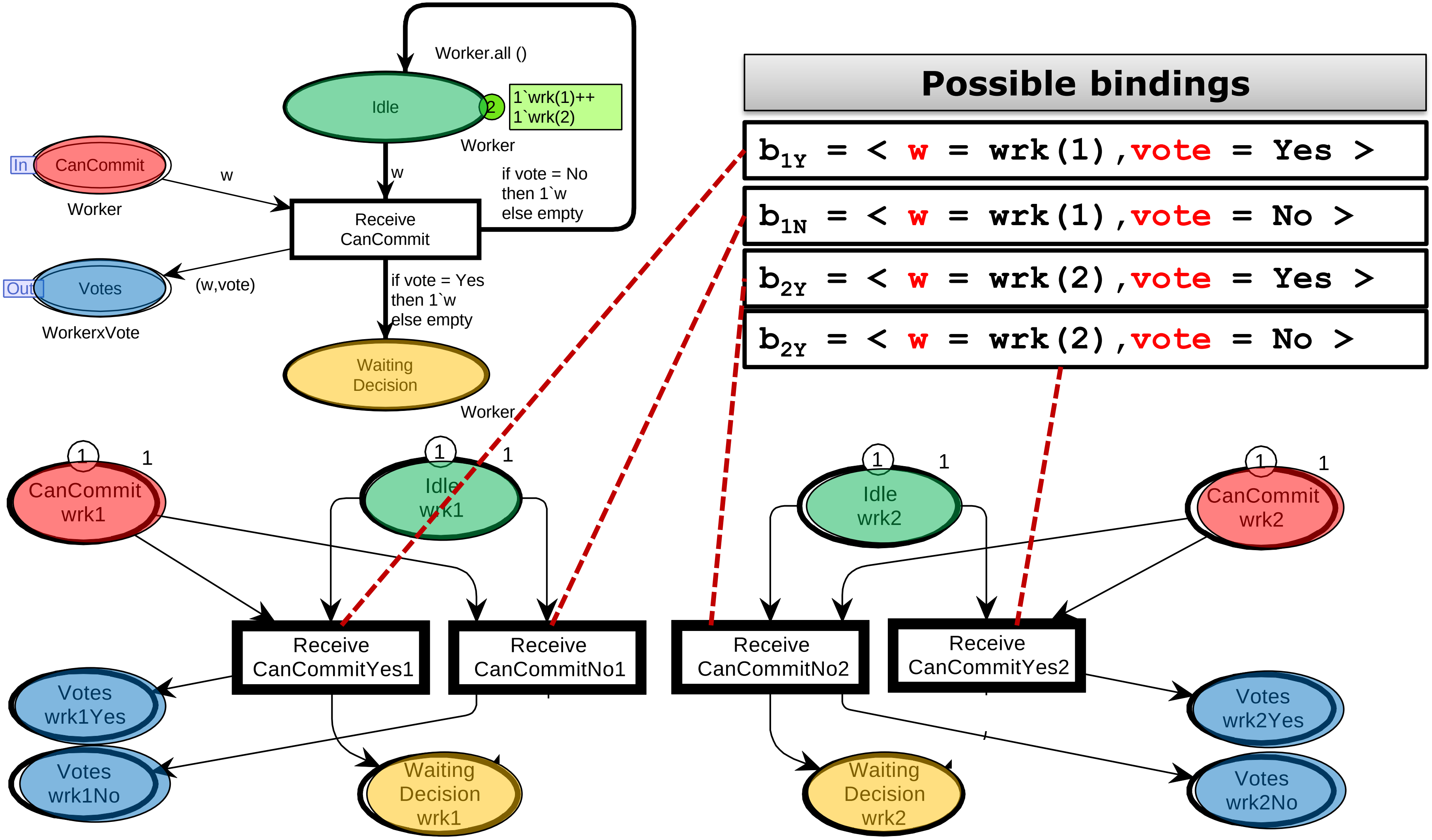
Unfolding hierarchical CPNs



Unfolding CPN Places



Unfolding CPN Transitions



Possible bindings

b_{1Y} = < w = wrk(1),vote = Yes >

b_{1N} = < w = wrk(1),vote = No >

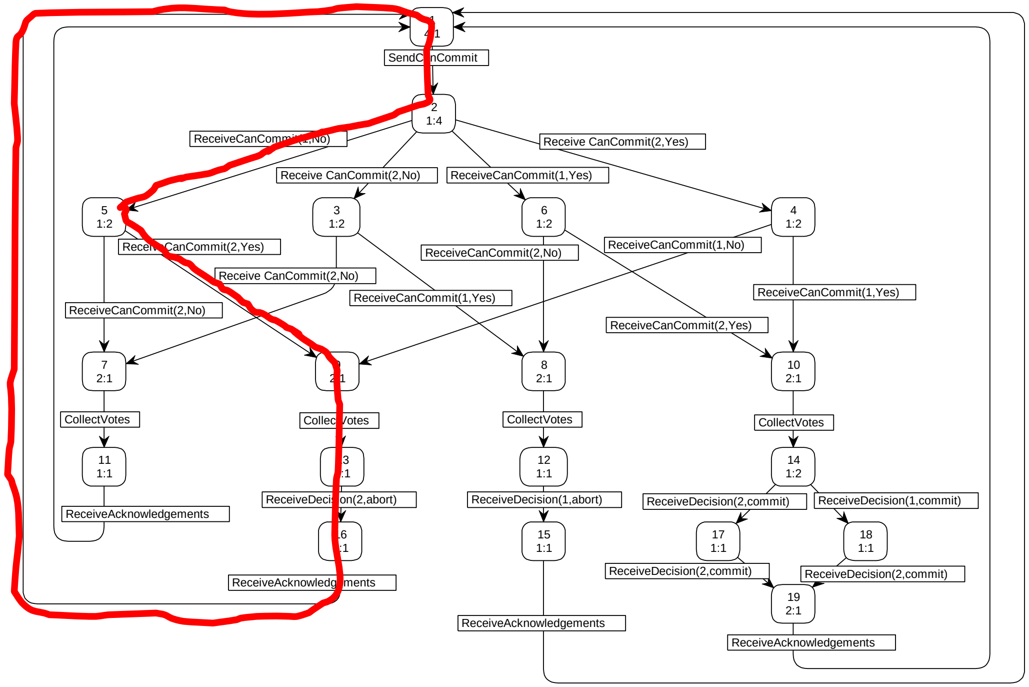
b_{2Y} = < w = wrk(2),vote = Yes >

b_{2Y} = < w = wrk(2),vote = No >

CPN Analysis Methods

Verification and Model Checking

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

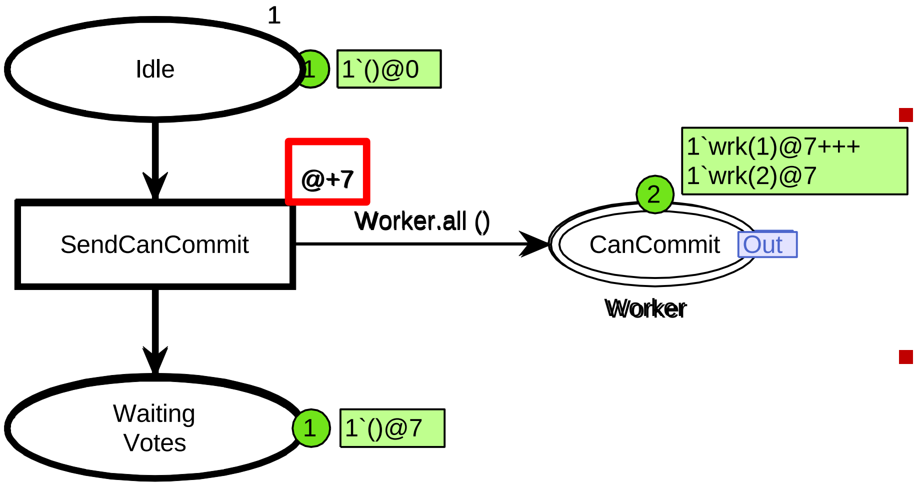


- **A state space represents all possible executions of the CPN model**
- **Standard behavioural properties can be investigated using the state space report**
- **Model-specific properties can be verified using queries and temporal logic model checking**

- **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

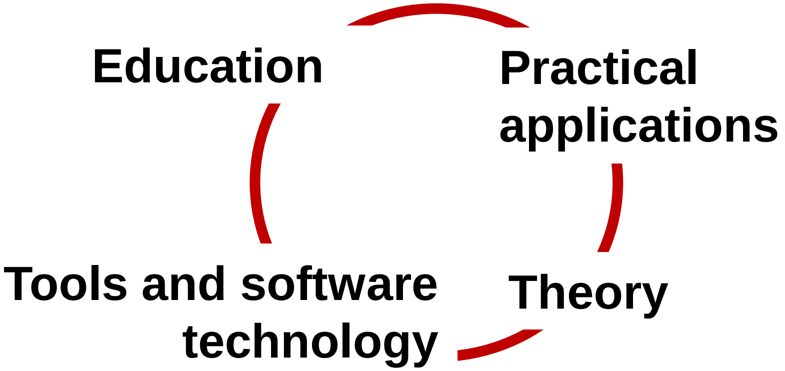


  - A global clock representing the current model time
  - Tokens carry time stamps describing the earliest possible model time at which they can be removed
  - Time inscriptions on transitions and arcs are used to give time stamps to the tokens produced on output places

- Random distribution functions can be used in arc expressions (delays, packet loss, …).
- Data collection monitors and batch simulations can be used to compute performance figures.

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 explicit synchronised
  - Verification and performance analysis supported by the same modelling language