One minutes description of COOPN

COOPN (Concurrent Object-Oriented Petri Nets) is an **object-oriented specification language** based on synchronized **algebraic Petri nets**. CoopnBuilder is an integrated development environment designated to the support of concurrent software development based on the COOPN language.

Briefly,

```
COOPN defines Petri nets and coordination between Petri nets using object-oriented approach.
```

The main characteristics of COOPN are:

- **Declarative language**: using prolog-like syntax, behavior of Petri nets are defined by rules. Transitions are specified with pre- and post- conditions.
- **Algebraic Petri net based**: the essential elements of a COOPN defined system are **place**, **transition(event)** and **token**. They are extended notion of classical place/transition nets because of their richness and complicity. It's fairly easy to represent classical P/T nets and colored Petri nets using COOPN, the inverse is often not possible.
- **Object-oriented**: a Petri net is encapsulated by means of COOPN class, only defined methods/events are visible from exterior.
- **Concurrency**: the semantic of COOPN guarantees the synchronization of events of defined Petri nets, three primitive operators are: ***alternative***, ***parallel*** and ***sequential***.
- **Coordination**: COOPN context is a higher level of encapsulation, it's an environment model and specifies the connections of internal components: objects (instances of classes) and sub-contexts. COOPN contexts do not define Petri nets but their compositions.

The execution of COOPN uses prolog-like interference engine.

COOPN can be used to model large, concurrent, transactional, distributed systems and construct prototypes for these systems. CoopnBuilder provides the facilities to edit, verify COOPN modules, compile them into java prototype and interpret, simulate the prototype.

There are three types of COOPN module: *Abstract(Algebraic) Data Type(ADT), Object and Context*.

- ADT represents data with no states and they are immutable. ADTs have operations.
- Class has places, methods(visible transitions), non-visible transitions and gates(outgoing events). Places may contain ADT data or objects.
- Context has methods and gates.

Methods are input events, **gates** are output events. Both of them can have parameters, the type of parameters can be ADT or COOPN class.

We define tokens using ADT with algebraic specifications, thus the type of tokens are richer than colored Petri nets, in addition, tokens support operations.

Ang Chen [http://cui.unige.ch/%7Echena/] @ SMV [http://smv.unige.ch/] , 6 April, 2005