Concurrency and O-O with Structured Algebraic Nets

Olivier Biberstein
Centre Universitaire d'Informatique
University of Geneva
Switzerland

Slide 1

Plan

- ◇ Introduction of CO-OPN/2
- ◇ Some examples
- ◇ Semantic aspects
- ◇ Refinement
- ◇ Subtyping
- ◇ Conclusion

Slide 2

CO-OPN/2

A concurrent object-oriented specification language

Features	Means
◇ data structures	← algebraic specifications
◇ control structures	} ← algebraic nets
◇ concurrency	
◇ class	← algebraic net module
◇ inheritance	← syntactic mechanism
◇ sub-typing	← order-sorted algebras
◇ dynamic creation	← instantiation mechanism

Slide 3

O-O Concepts of CO-OPN/2

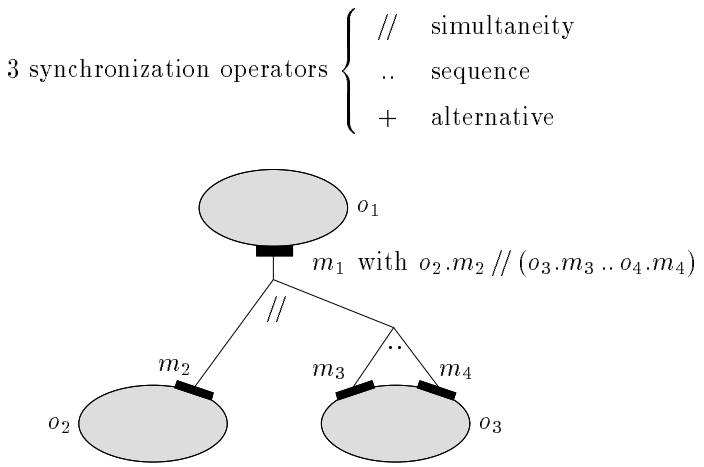
◇ CO-OPN/2 Specification: a collection of classes & ADT's

◇ Class:
- Header: Genericity, Inheritance relationship
- Interface: Methods, Type, Sub-type relationship
- Body: Places, Transitions, Behavioral Axioms

◇ Object identity: algebraic value

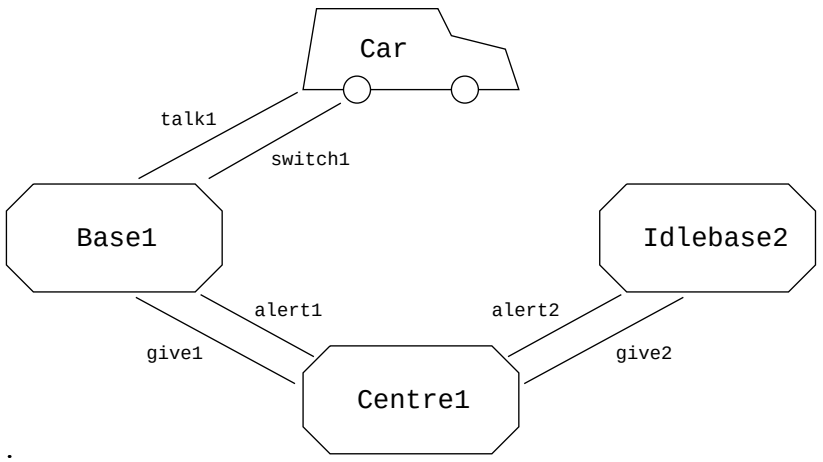
Slide 4

◇ Object interaction: synchronization expressions



Slide 5

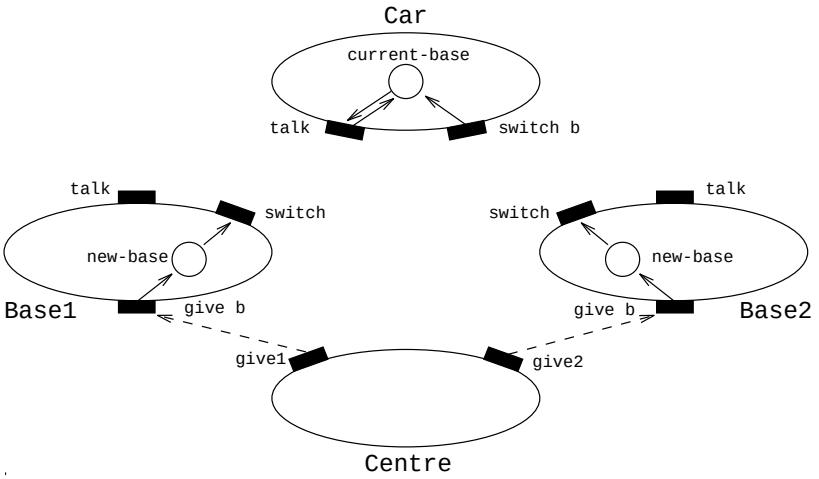
Mobile Telephone



.

Slide 6

Mobile Telephone

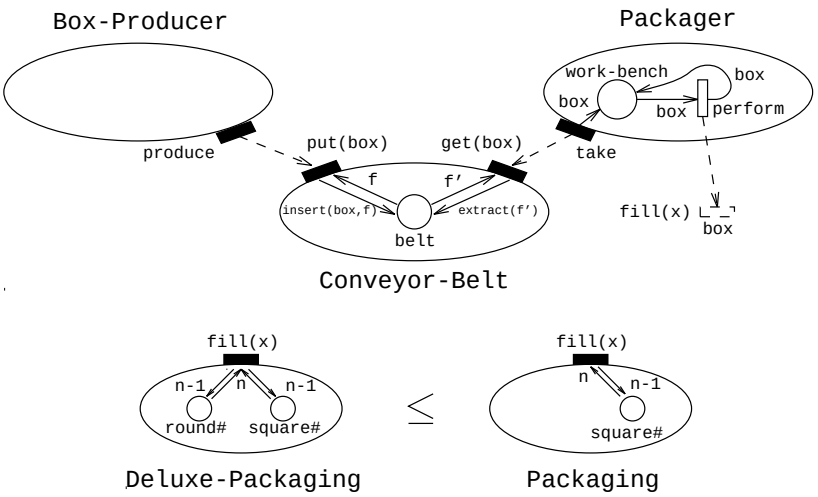


Slide 7

```
Object Car;
Interface
  Use Base;
  Methods
    talk;
    switch _ : base;
Body
  Place
    current-base _ : base;
  Axioms
    switch b : current-base b' -> current-base b;
    talk With b.talk : current-base b -> current-base b;
  Initial
    current-base Base1;
  Where b, b' : base;
End Car;
```

Slide 8

Swiss Chocolate Factory



Slide 9

```
Class Packaging;
Interface
  Use Nat, Miscellaneous;
  Type packaging;
  Methods
    fill _ : kind;
Body
  Place
    square# _  : nat;
  Axioms
    fill praline : square# (n+1) -> square# n;
  Initial
    square# nb-sqr;
  Where n : nat;
End Packaging;
```

Slide 10

```
Class Deluxe-Packaging;
Inherit Packaging;
Renaming packaging Into deluxe-packaging;
Interface
  Subtype deluxe-packaging < packaging;
Body
  Place
    round# _  : nat;
  Axioms
    fill rum-truffle : round# (n+1) -> round# n;
  Initial
    round# nb-rnd;
  Where
    n : nat;
End Deluxe-Packaging;
```

Slide 11

Semantics

◇ Concurrent compositional semantics

◇ Based on transition system (step semantics)

◇ Expressed by means of inference rules

Semantics of a class C : $Sem(C)$

Semantics of a specification $Spec$: $Sem(Spec)$

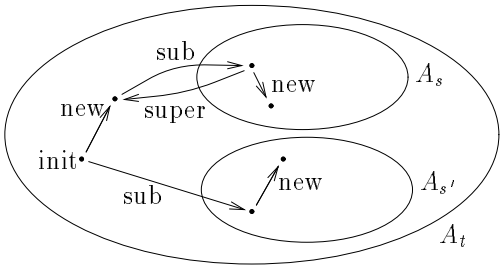
Slide 12

Semantics

◇ An object identifier algebra which captures sub-typing

$s \leq t$

$s' \leq t$



init, new, sub, super

Slide 13

Refinement

Refinement = Substitution of a part of specification by another one (more concrete) which "preserves" the semantics.

Let Cl, Cl' be two sets of classes such that $Cl \subseteq Spec$.

Cl' is a refinement of Cl ($Cl' \preceq Cl$) if $\exists$ a relation $\preceq_{\text{Sem}}$ such that

$$Sem(Cl') \preceq_{\text{Sem}} Sem(Cl) \Rightarrow Sem((Spec \setminus Cl) \cup Cl') \preceq_{\text{Sem}} Sem(Spec)$$

Slide 14

Behavioural Sub-typing

Sub-typing : Based on the strong form of substitutability principle

Let C_s and C_t be two classes and their associated types s, t.

s is a sub-type of t ($s \leq t$) if $\exists$ a relation $\leq_{\text{Sem}}$ such that

$$Sem(C_s) \leq_{\text{Sem}} Sem(C_t) \Rightarrow Sem((Spec \setminus C_t) \cup C_s) \leq_{\text{Sem}} Sem(Spec)$$

Slide 15

Conclusion

◇ CO-OPN/2 formal O-O concurrent specification language

- Inheritance $\neq$ Sub-typing
- Synchronization expressions
- Each object is referred by its identity (oid algebra)

◇ Introduction of an abstract view of refinement and sub-typing

Future Works

○ Illustrate our abstract view of refinement and sub-typing

○ Study more deeply the notion of object equivalence

○ Introduce the notion of observer in our view of sub-typing

○ Distributed operational semantics

Slide 16