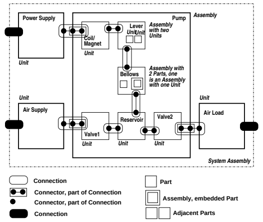
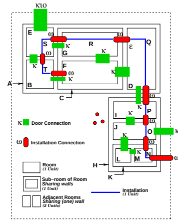
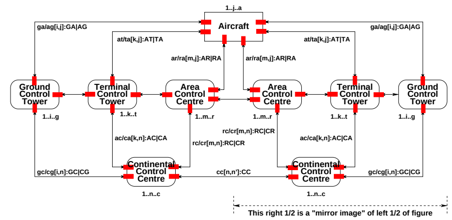
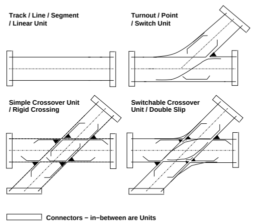
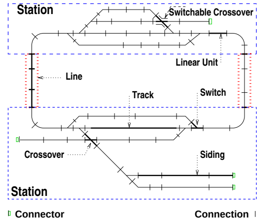
To every Mereology there corresponds a λ-expression

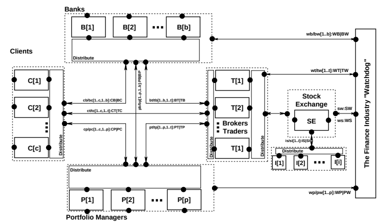
In honour of Prof. Egidio Astesiano

Genoa, March 31, 2012

Dines Bjørner, DTU Informatics, March 28, 2012: 16:21

1. "Concrete" Examples

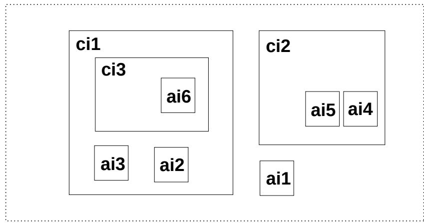






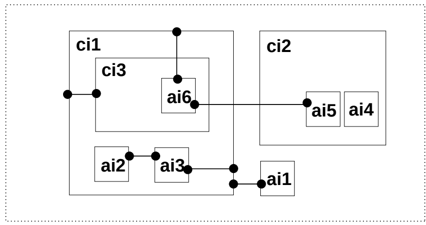


2. Parts and Part Relations: An Axiom System

- **Parts and Attributes**
  - type $\mathcal{P}, \mathcal{A}$
  - $\in: \mathcal{A} \times \mathcal{P} \rightarrow \mathbf{Bool}$.
- **Predicates**

part_of:	$\mathbb{P} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbf{Bool}$	Slide 5
proper_part_of:	$\mathbb{PP} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbf{Bool}$	Slide 5
overlap:	$\mathbb{O} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbf{Bool}$	Slide 5
underlap:	$\mathbb{U} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbf{Bool}$	Slide 5
over_crossing:	$\mathbb{OX} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbf{Bool}$	Slide 6
under_crossing:	$\mathbb{UX} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbf{Bool}$	Slide 6
proper_overlap:	$\mathbb{PO} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbf{Bool}$	Slide 6
proper_underlap:	$\mathbb{PU} : \mathcal{P} \times \mathcal{P} \rightarrow \mathbf{Bool}$	Slide 6





- **Parthood**

$$\forall p_x : \mathcal{P} \bullet \mathbb{P}(p_x, p_x) \quad (1)$$
$$\forall p_x, p_y : \mathcal{P} \bullet (\mathbb{P}(p_x, p_y) \wedge \mathbb{P}(p_y, p_x)) \Rightarrow p_x = p_y \quad (2)$$
$$\forall p_x, p_y, p_z : \mathcal{P} \bullet (\mathbb{P}(p_x, p_y) \wedge \mathbb{P}(p_y, p_z)) \Rightarrow \mathbb{P}(p_z, p_z) \quad (3)$$

- **Proper Part**

$$\mathbb{PP}(p_x, p_y) \stackrel{\triangle}{=} \mathbb{P}(p_x, p_y) \wedge \neg\mathbb{P}(p_y, p_x) \quad (4)$$

- **Overlap**

$$\mathbb{O}(p_x, p_y) \stackrel{\triangle}{=} \exists a : \mathcal{A} \bullet a \in p_x \wedge a \in p_y \quad (5)$$

- **Underlap**

$$\mathbb{U}(p_x, p_y) \stackrel{\triangle}{=} \exists p_z : \mathcal{P} \bullet \mathbb{P}(p_x, p_z) \wedge \mathbb{P}(p_y, p_z) \quad (6)$$

- **Overcross**

$$\mathbb{OX}(p_x, p_y) \stackrel{\triangle}{=} \mathbb{O}(p_x, p_y) \wedge \neg\mathbb{P}(p_x, p_y) \quad (7)$$

- **Undercross**

$$\mathbb{UX}(p_x, p_y) \stackrel{\triangle}{=} \mathbb{U}(p_x, p_z) \wedge \neg\mathbb{P}(p_y, p_x) \quad (8)$$

- **Proper Overcross**

$$\mathbb{PO}(p_x, p_y) \stackrel{\triangle}{=} \mathbb{OX}(p_x, p_y) \wedge \mathbb{OX}(p_y, p_x) \quad (9)$$

- **Proper Undercross**

$$\mathbb{PU}(p_x, p_y) \stackrel{\triangle}{=} \mathbb{UX}(p_x, p_y) \wedge \mathbb{UX}(p_y, p_x) \quad (10)$$

3. Parts and Part Relations: A Syntax Model

type
  W = P=set
  P = A | C
value
  is_A: P → **Bool**, is_C: P → **Bool**
axiom
  ∀ a:A,c:C•a≠c, i.e., A∩C={||} ∧ is_A(a)≡∼is_C(a)∧is_C(c)≡∼is_A(c)

value
  obs_Ps: C → P-**set**   **axiom** ∀ c:C · obs_Ps(c)≠{}
  xtr_Ps: C → P-**set**
  xtr_Ps(c) ≡ **let** ps = obs_Ps(c) **in** ps ∪ ∪ {obs_Ps(c′)|c′:C · c′ ∈ ps} **end**

3.1. 'Within' Relations

value
  imm_within: P × P $\stackrel{\sim}{\to}$ **Bool**
  imm_within(p,p′) ≡ is_C(p′) ∧ p ∈ obs_Ps(p′)

  within: P × P $\stackrel{\sim}{\to}$ **Bool**
  within(p,p′) ≡ imm_within(p,p′) ∨ ∃ p″:C · p″ ∈ obs_Ps(p′) ∧ within(p,p″)

3.2. 'Adjacency' Relations

value
  imm_adjacent: P × P → C $\stackrel{\sim}{\to}$ **Bool**,
  imm_adjacent(p,p′)(c) ≡ p≠p′ ∧ {p,p′}⊆obs_Ps(c)

  adjacent: P × P → C $\stackrel{\sim}{\to}$ **Bool**
  adjacent(p,p′)(c) ≡
   imm_adjacent(p,p′)(c) ∨
   ∃ p″,p‴:P · imm_adjacent(p″,p‴)(c) ∧
    ((p=p″)∨within(p,p″)(c)) ∧ ((p′=p‴)∨within(p′,p‴)(c))

3.3. Unique Identifications

type
  Π
value
  uid_Π: P → Π
axiom
  ∀ p,p′:P • p≠p′ ⇒ uid_Π(p)≠uid_Π(p′)
value
  xtr_Πs: C → Π-**set**
  xtr_Πs(c) ≡ {uid_Π(c)} ∪ {uid_Π(p)|p:P•p ∈ xtr_Πs(c)}

3.4. Attributes

type
  ATR
value
  atr_ATRs: P → ATR-**set**
  share: P×P → **Bool**
  share(p,p′) ≡ p≠p′∧∃ atr:ATR•atr∈atr_ATRs(p)∧atr∈atr_ATRs(p′)
  ∈: ATR × ATR-**set** → **Bool**

3.5. Connections

type
  K = {| k:Π-**set** • **card** k = 2 |}
value
  mereo_Ks: P → K-**set**
  xtr_Ks: P → K-**set**
  xtr_Ks(p) ≡ {{uid_Π(p),π}|π:Π•π ∈ mereo_Πs(p)}

3.6. Connector and Attribute Sharing Axioms

axiom
  ∀ w:W•
    let ps = xtr_Ps(w), ks = xtr_Ks(w) **in**
    ∀ p,p′:P • p≠p′ ∧ {p,p′}⊆ps ∧ share(p,p′) ⇒ {uid_Π(p),uid_Π(p′)} ∈ ks ∧
    ∀ {uid,uid′} ∈ ks ⇒
      ∃ p,p′:P • {p,p′}⊆ps ∧ {uid,uid′}={uid_Π(p),uid_Π(p′)} ⇒ share(p,p′) **end**
value
  xtr_Ks: W → K-**set**
  xtr_Ks(w) ≡ ∪{xtr_Ks(p)|p:P•p ∈ obs_Ps(p)}

  sharing: P × P → **Bool**
  sharing(p,p′) ≡ p≠p′∧share(p,p′)

4. Satisfaction

- The *model* of the previous section
- is a model for the *axioms*.

We assign

1. P as the meaning of $\mathcal{P}$
2. ATR as the meaning of $\mathcal{A}$,
3. imm_within as the meaning of $\mathbb{P}$,
4. within as the meaning of $\mathbb{PP}$,
5. $\in_{(\text{of type:}ATR\times ATR-\mathbf{set}\rightarrow\mathbf{Bool})}$ as the meaning of $\in_{(\text{of type:}\mathcal{A}\times\mathcal{P}\rightarrow\mathbf{Bool})}$ and
6. sharing as the meaning of $\mathbb{O}$.

- With the above assignments is is now easy to prove that
  - the other axiom-operators
  - $\mathbb{U}$, $\mathbb{PO}$, $\mathbb{PU}$, $\mathbb{OX}$ and $\mathbb{UX}$
  - can be modelled by means of
  - imm_within, within, $\in_{(\text{of type:}ATR\times ATR-\mathbf{set}\rightarrow\mathbf{Bool})}$ and sharing.

5. Parts and Part Relations: A Semantics Model

- The model is expressed in CSP.
- **Parts ≡ Processes**
- **Connectors ≡ Channels**

value
  w:C
  ps:P-**set** = ∪{xtr_Ps(c)|c:C•c ∈ w} ∪ {a|a:A•a ∈ w}
  ks:K-**set** = xtr_Ks(w)
type
  K = Π-**set axiom** ∀ k:K•**card** k=2
  ChMap = Π $\overrightarrow{m}$ K-**set**
value
  cm:ChMap = [uid_Π(p)↦xtr_Ks(p)|p:P•p ∈ ps]
channel
  ch[k|k:K•k ∈ ks] MSG

5.1. Process Definitions

value

system: W → **process**
system(w) ≡
  ‖{comp_process(uid_Π(c))(c)|c:C•c ∈ w} ‖ ‖{atom_process(uid_Π(a),a)|a:A•a ∈ w}

comp_process: π:Π → c:C→ **in**,**out** {ch(k)|k:K•k ∈ cm(π)} **process**
comp_process(π)(c) ≡ [**assert:** π = uid_Π(c)]
  $\mathcal{M}_\mathcal{C}$(π)(c)(atr_ATR(c)) ‖
  ‖ {comp_process(uid_Π(c′))(c′)|c′:C•c′ ∈ obs_Ps(c)} ‖
  ‖ {atom_process(uid_Π(a))(a)|a:A•a ∈ obs_Ps(c)}

$\mathcal{M}_\mathcal{C}$: π:Π → C → ATR-**set** → **in**,**out** {ch(k)|k:K•k ∈ cm(pi)} **process**
$\mathcal{M}_\mathcal{C}$(π)(c)(c_attrs) ≡ $\mathcal{M}_\mathcal{C}$(c)($C\mathcal{F}$(c)(c_attrs)) **assert:** atr_ATR(c) ≡ c_attrs

$C\mathcal{F}$: c:C → ATR-**set** → **in**,**out** {ch[em(i)]|i:KI•i ∈ cm(uid_Π(c))} ATR-**set**

atom_process: a:A → **in**,**out** {ch[cm(k)]|:K•k ∈ cm(uid_Π(a))} **process**
atom_process(a) ≡ $\mathcal{M}_\mathcal{A}$(a)(atr_ATR(a))

$\mathcal{M}_\mathcal{A}$: a:A → ATR-**set** → **in**,**out** {ch[cm(k)]|k:K•k ∈ cm(uid_Π(a))} **process**
$\mathcal{M}_\mathcal{A}$(a)(a_attrs) ≡ $\mathcal{M}_\mathcal{A}$(a)($A\mathcal{F}$(a)(a_attrs)) **assert:** atr_ATR(a) ≡ a_attrs

$A\mathcal{F}$: a:A → ATR-**set** → **in**,**out** {ch[em(k)]|k:K · k ∈ cm(uid_Π(a))} ATR-**set**

value
  $\mathcal{F}$: p:(C|A) → ATR-**set** → **in**,**out** {ch[em(k)]|k:K · k ∈ cm(uid_Π(p))} ATR-**set**
  $\mathcal{F}$(p)(π,πs,atrs) ≡
    ⌈⌉ {**let** av = ch[em({π,j})] ? **in**
      ... ; [optional] ch[em({π,j})] ! in_reply(atrs)(av);
      (π,πs,in_update_ATR(atrs)(j,av)) **end**
      | {π,j}:K•{π,j} ∈ πs}
   ⌈⌉ ⌈⌉ { ... ;
      ch[em({π,j})] ! out_reply(atrs);
      (π,πs,out_update_ATRs(atrs)(j))
      | {π,j}:K•{π,j} ∈ πs}
    ⌈⌉ (π,πs,own_work(atrs))
  assert: π = uid_Π(p)

  in_reply: ATR-**set** → Π × VAL → VAL
  in_update_ATRs: ATR-**set** → Π × VAL → ATR-**set**
  out_reply: ATR-**set** → VAL
  out_update_ATRs: ATR-**set** →Π → ATR-**set**
  own_work: ATR-**set** → ATR-**set**

6. Laudatio

- Dear Egidio,
  - ⋆ **Today we honour you ! It is about time.**
  - ⋆ I am glad, and honoured, to be here.
  - ⋆ You were the saving grace of the **Ada Formal Definition.**
  - ⋆ You and your team bore the brunt of the work — **Gianna Reggio** and **Elena Zucca** amongst them.
  - ⋆ I have often related the story about how you devised the **SMoLCS** structuring that, for example, allowed a variety of interpretations of Ada Concurrency. **Brilliant !**
  - ⋆ You have built up a group here at Genoa – a group which has made ever-lasting and deep contributions to science – one of the finest groups in Europe
- Thanks for the time you spent showing Genoa to our son Nikolaj !