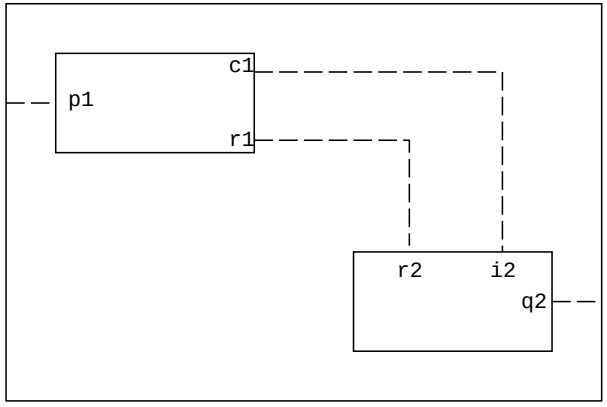
# Styles de composition

FéRIA
ACI SI Fiacre

7-8 mars 2007

# Schéma de base




- Fiacre “séquentiel” offre les boites atomiques.
- Chaque boite est caractérisée par des ports.
  - Ports de synchro non orientés.
  - Ports d’échange de données orientés.

Discussion : alternatives pour assembler les boites atomiques Fiacre.

# Composition à la WRIGHT

**Approche connecteur**

- Chaque composant a un rôle : un comportement.
- Un processus **glue** sérialise les différents rôles définis par le connecteur.

**connector** C-S-connector =
**role** Client = $c1! \rightarrow r1? \rightarrow$ Client
**role** Server = $i2? \rightarrow r2! \rightarrow$ Server
**glue** = Client.$c1? \rightarrow$ Server.$i2! \rightarrow$ Server.$r2? \rightarrow$ Client.$r1! \rightarrow$ **glue**

- sérialisation $\rightsquigarrow$ pas de simultanéité.
- La connexion du composant est statique mais le processus glue peut établir des connexions dynamiques entre les différents composants.

# Composition à la ALTARICA

## Approche vecteur de synchronisation

- Le vecteur de synchronisation identifie, a posteriori, des portes du composant.

**node** main
  **sub**
    comp1 : Type1
    comp2 : Type1
**sync**
  $\langle$comp1.c1, comp2.i2$\rangle$
  $\langle$comp1.r1, comp2.r2$\rangle$
**edon**

- Approche, a priori, plus souple que celle du renommage : pour une même porte, on peut donner des connections alternatives. Mais les différentes connexions sont données statiquement par le vecteur de synchronisation.

# Composition

**Approche par renommage (Cotre)**

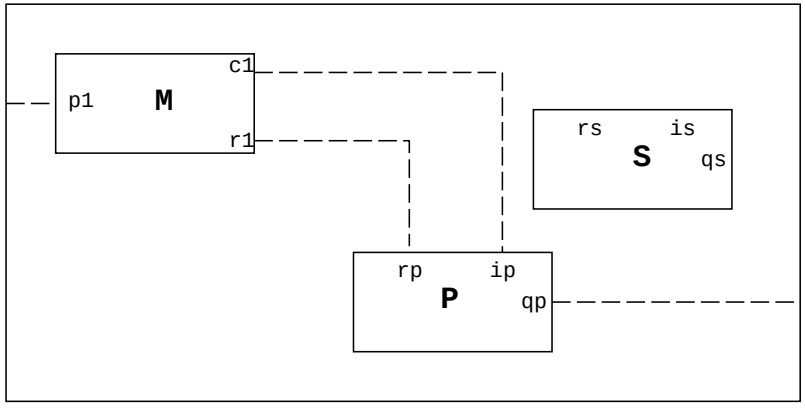
```
component main
  sub
    pa,pb : Port
    comp1 : Type1(pa,pb)
    comp2 : Type1(pa,pb)
end
```

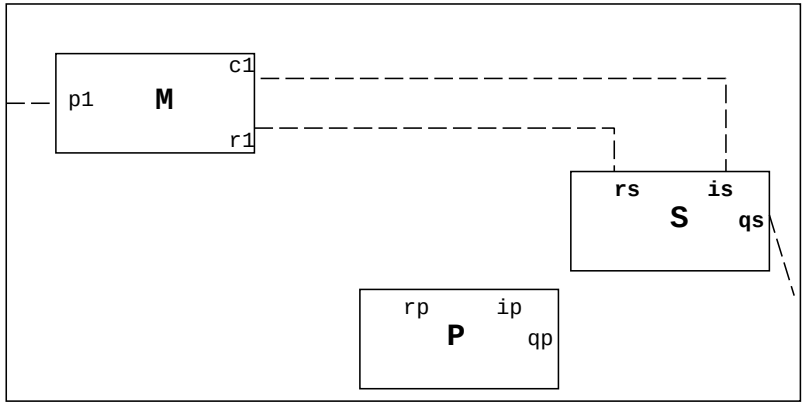
- Les composants sont statiquement connectés.

# Composition à la AADL

```
 1  process H
 2  features
 3    InCmd: in event port
 4    OutE: out event port
 5  end H
 6
 7  process implementation H.basic
 8  subcomponents
 9    M: thread M_ty.basic;
10    P: thread P_ty.basic;
11  connections
12    c_Cmd: event port InCmd -> M.p1;
13    i_c1: event data port M.r1 -> P.rp;
14    o_e: event port P.qp -> OutE;
15  end H.basic
```

# Notion de mode



# Architecture pseudo-dynamique

```
1  process implementation H.basic
2  subcomponents
3    M: thread M_ty.basic in modes (m1,m2);
4    P: thread P_ty.basic in modes (m1);
5    S: thread P_ty.basic in modes (m2);
6  modes
7    m1: initial mode;
8    m2: mode
9    m1 -[ evt ]-> m2;
10 connections
11   c_Cmd: event port InCmd -> M.p1 in modes (m1,m2);
12   i_c1: event data port M.r1 -> P.rp in modes (m1);
13   o_e: event port P.qp -> OutE in modes (m1);
14   i_c_secondaire: event data port M.r1 -> S.rp in m
15   o_e_secondaire: event port S.qp -> OutE in modes
16 end H.basic
```

- possible en WRIGHT : processus CSP.
- possible en BIP.

Remarque : construction éliminable sur un automate non hiérarchique.

# towards modes in FIACRE

```
1  component m(c_Cmd, o_e: port)
2  local ports
3    lp1, lp2: port in modes (m1,m2);
4  subcomponents
5    M: M_ty(lp1,lp2) in modes (m1,m2);
6    P: P_ty(lp1) in modes (m1);
7    S: P_ty(lp2) in modes (m2);
8  states
9    m1, m2 init m1;
10
11   from m1 do evt to m2; ... FIACRE like transition
12
13 end H.basic
```

# Transfert de données

**Signature du transfert :**

- transfert unidirectionnel (WRIGHT, Cotre).
- transfert multiple bidirectionnel (NTIF, BIP, Ada)

**Contrôle du transfert :**

- réalisé par le composant (NTIF, Cotre).
- réalisé par le connecteur (BIP, WRIGHT).

# Discussion

**But de fiacre ?**

- Langage pivot pour formalismes métier (UML, AADL, SYSML, . . . )
- Langage d'assemblage de composants.

- Vecteurs de synchro ou points de synchro ?
- Priorités
- Temps