

Contrats comportementaux pour composants

Cyril Carrez

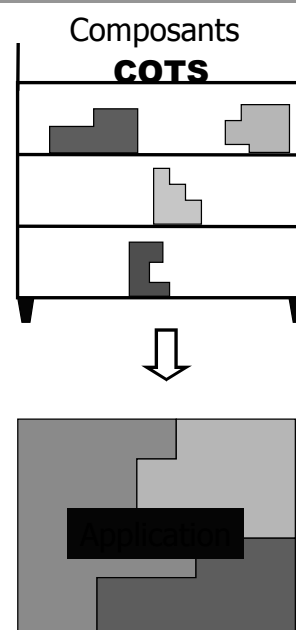
15 Décembre 2003

Ecole Nationale Supérieure des Télécommunications

Approches de développement par assemblage

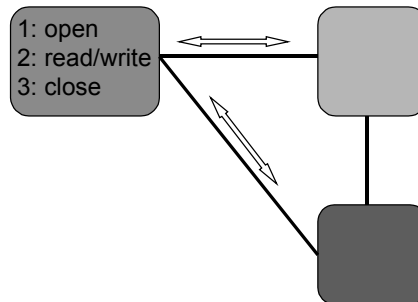
- ADL (90's)
 - composants
 - connecteur
 - configuration
- Catalysis (1998)
 - Modèle abstrait & raffinement
- UML 2.0 (2003)
 - Notation pour composants

Classification
[Medvidovic & Taylor]



Cadre de l'étude

- Composants
 - spécification + code
- *services non uniformes*
- Liens dynamiques



Objectifs

- Propriétés de sûreté : pas d'interblocage externe
- Propriétés de vivacité : les messages seront consommés

Plan

- Positionnement de notre approche
- Langage d'interface
- Sémantique de composant
- Respect de Contrat
- Assemblage sain
- Conclusion & Perspectives

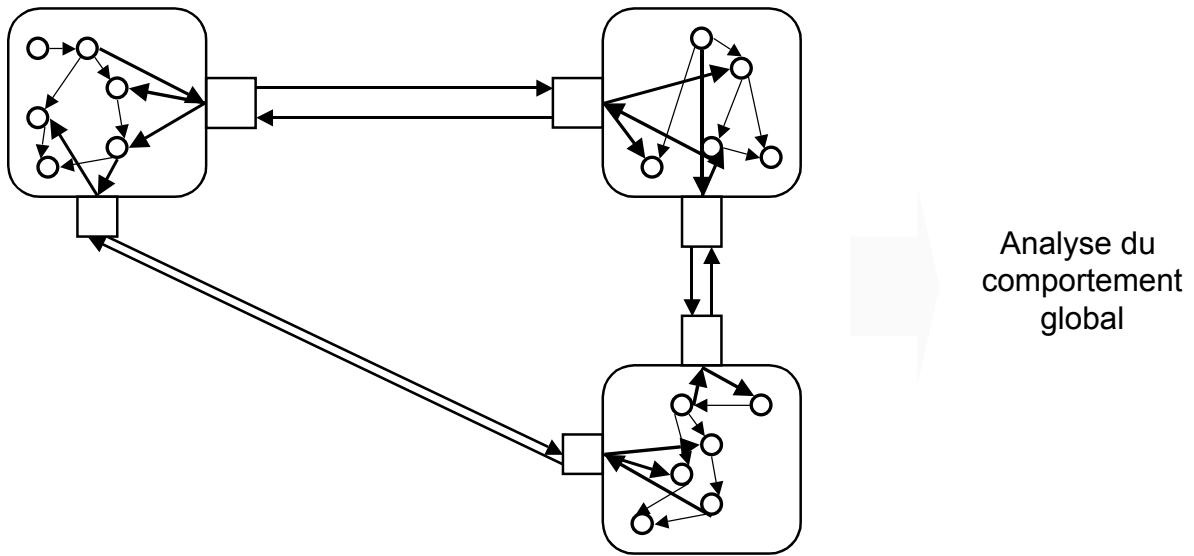
Typage & Contrat

- Typage comportemental [Nierstrasz]
 - Typage implicite
 - π -calcul [Kobayashi]
 - «message orphelin» [Colaço, Colin, Pantel...]
 - Typage explicite
 - Types réguliers [Nierstrasz]
 - «message non compris» [Nimour, Najm, Stefani, Ravara,...]
- Contrats
 - Design by Contract [Meyer]
 - Classification [Beugnard et al.]
 - Syntaxique / **comportement** (pré/post-conditions) / **synchronisation** / QoS

Faiblesses des approches existantes

- Algèbres de Processus vs Objets
- Pas de modalités sur les messages
=> implicitement : modalité = *obligation*
- Pas de prise en compte des liens dynamiques
- Pas de prise en compte des types des références passées en argument
- La vérification est en général non compositionnelle et complexe

L'approche Darwin, Wright,...

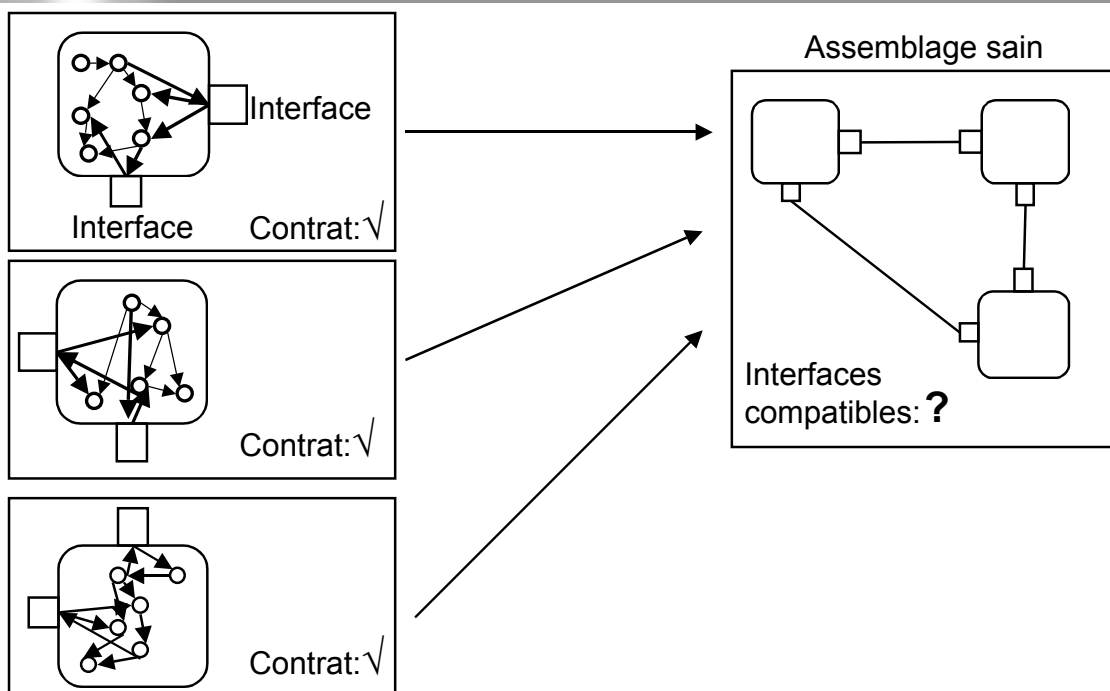


15/12/2003

Contrats Comportementaux pour Composants

7

Notre approche

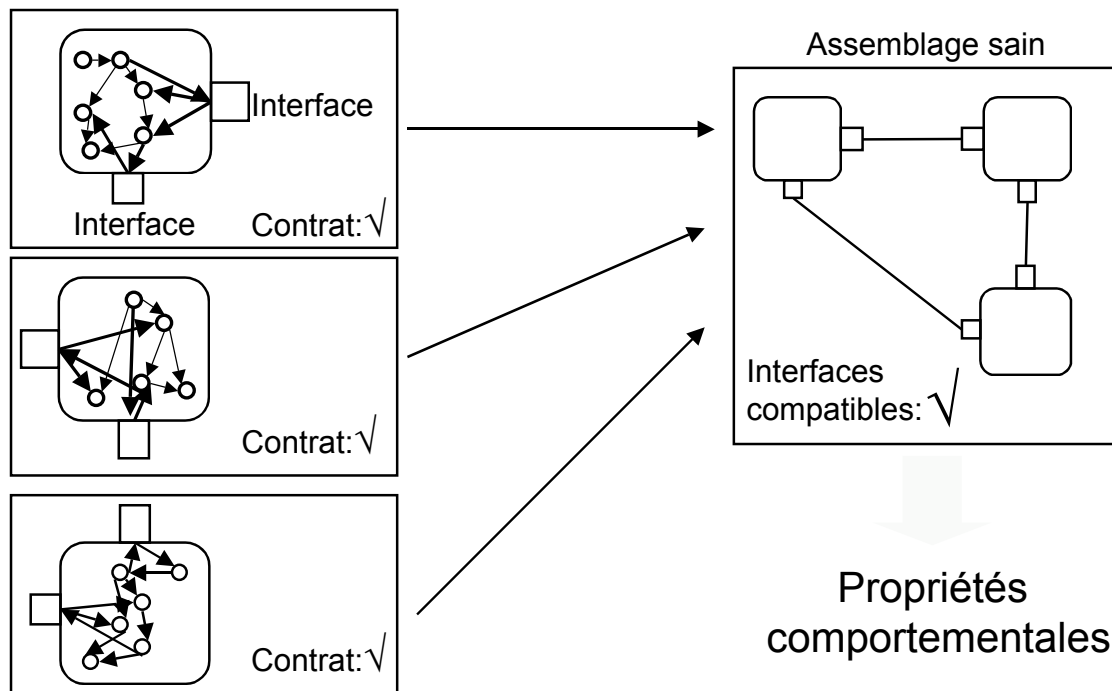


15/12/2003

Contrats Comportementaux pour Composants

8

Notre approche



15/12/2003

Contrats Comportementaux pour Composants

9

Plan

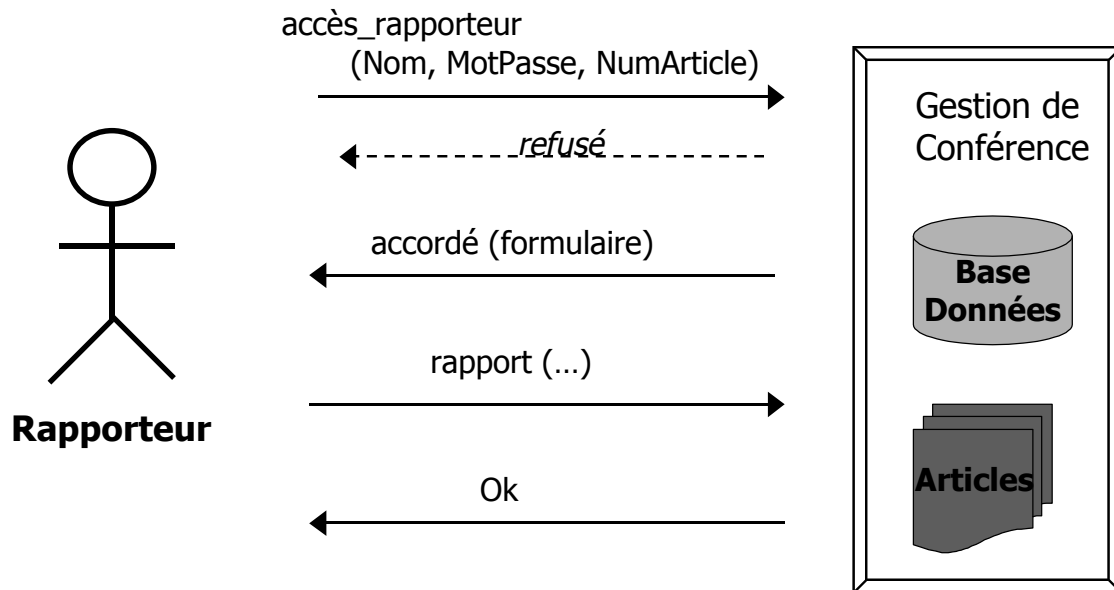
- Langage d'interface
 - modalités **may/must**, envoi de référence
 - sous-typage & compatibilité
- Sémantique de composant
- Respect de Contrat
- Assemblage sain
- Conclusion & Perspectives

15/12/2003

Contrats Comportementaux pour Composants

10

Types d'interfaces: exemple

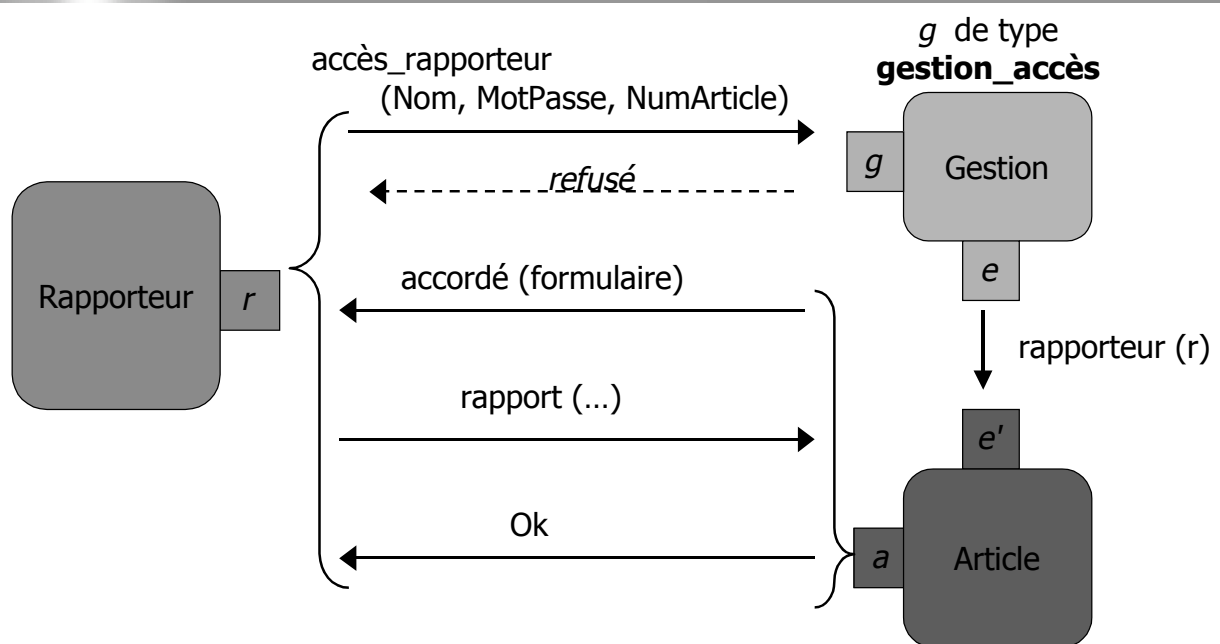


15/12/2003

Contrats Comportementaux pour Composants

11

Types d'interfaces: exemple

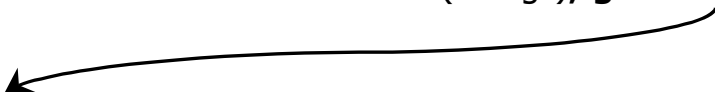


15/12/2003

Contrats Comportementaux pour Composants

12

Exemple: Type gestion_accès

- **gestion_accès =**
 may ? [accès_rapporteur (string,string,integer);
 must ! [refusé; 0
 + accordé (strings); **gestion_rapporteur**]]
 - **gestion_rapporteur =**
 must ? [rapport (strings); **must !** [Ok; **gestion_rapporteur_chg**
 + erreur; **gestion_rapporteur**]]
 - **gestion_rapporteur_chg =**
 may ? [rapport (strings); **must !** [Ok; **gestion_rapporteur_chg**
 + erreur; **gestion_rapporteur_chg**]]
- 

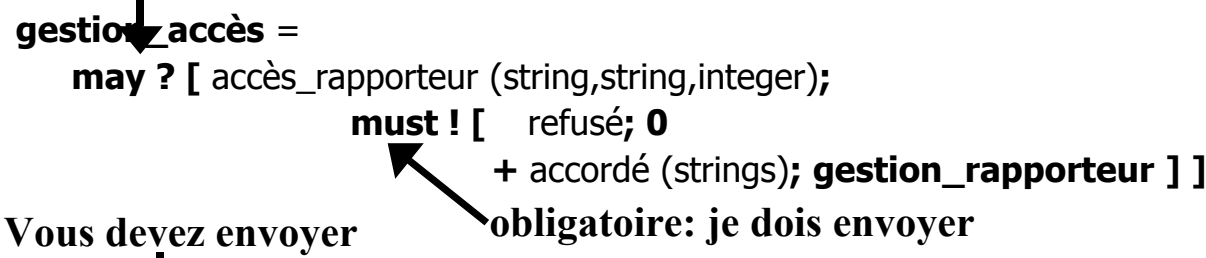
15/12/2003

Contrats Comportementaux pour Composants

13

Exemple: Type gestion_accès

permis: vous pouvez envoyer, je garantis la réception

- **gestion_accès =**
 may ? [accès_rapporteur (string,string,integer);
 must ! [refusé; 0
 + accordé (strings); **gestion_rapporteur**]]
 - **gestion_rapporteur =**
 must ? [rapport (strings); **must !** [Ok; **gestion_rapporteur_chg**
 + erreur; **gestion_rapporteur**]]
 - **gestion_rapporteur_chg =**
 may ? [rapport (strings); **must !** [Ok; **gestion_rapporteur_chg**
 + erreur; **gestion_rapporteur_chg**]]
- 
- Vous devez envoyer
- obligatoire: je dois envoyer

15/12/2003

Contrats Comportementaux pour Composants

14

Compatibilité: *Comp (I, J)*

<i>J</i> \ <i>I</i>	must ?	may ?	must !	may !	0
must ?			√		
may ?		√	√	√	√
must !	√	√			
may !		√			
0		√			√

- gestion_rapporteur =
must ? [rapport (strings); **must !** [Ok; gestion_rapporteur_chg
+ erreur; gestion_rapporteur]]
gestion_rapporteur_chg = **may ?** [...]
- entrer_rapport =
must ! [rapport (strings); **must ?** [Ok; 0
+ erreur; entrer_rapport]]

15/12/2003

Contrats Comportementaux pour Composants

15

Compatibilité: *Comp (I, J)*

<i>J</i> \ <i>I</i>	must ?	may ?	must !	may !	0
must ?			√		
may ?		√	√	√	√
must !	√	√			
may !		√			
0		√			√

- gestion_rapporteur =
must ? [rapport (strings); **must !** [Ok; gestion_rapporteur_chg
+ erreur; gestion_rapporteur]]
gestion_rapporteur_chg = **may ?** [...]
- entrer_rapport =
must ! [rapport (strings); **must ?** [Ok; 0
+ erreur; entrer_rapport]]

15/12/2003

Contrats Comportementaux pour Composants

16

Sous-typage: $T \leq S$

- Compatibilité: message envoyé \leq message reçu
- réception:
 - $mod ? M1+M2+M3 \leq mod ? M1+M2$
 - contra-variante
- émission:
 - $mod ! M1 \leq mod ! M1+M2$
 - co-variante
- modalités:
 - **may ? \leq must ?** – **may ? \leq 0** – **may ? \leq may !**
 - **must ! \leq may !** – **0 \leq may !**

Propriétés des sous-types

- \leq est transitive
- Le sous-type peut remplacer le super-type
 - $Comp(I, S) \ \& \ (T \leq S) \ \Rightarrow \ Comp(I, T)$
- Plus grand super-type compatible:
 - dual: $J^D = J$ avec émissions et réceptions inversées
 - $Comp(I, J) \ \Leftrightarrow \ I \leq J^D$
- Démonstrations
 - par induction sur la structure des types

Plan

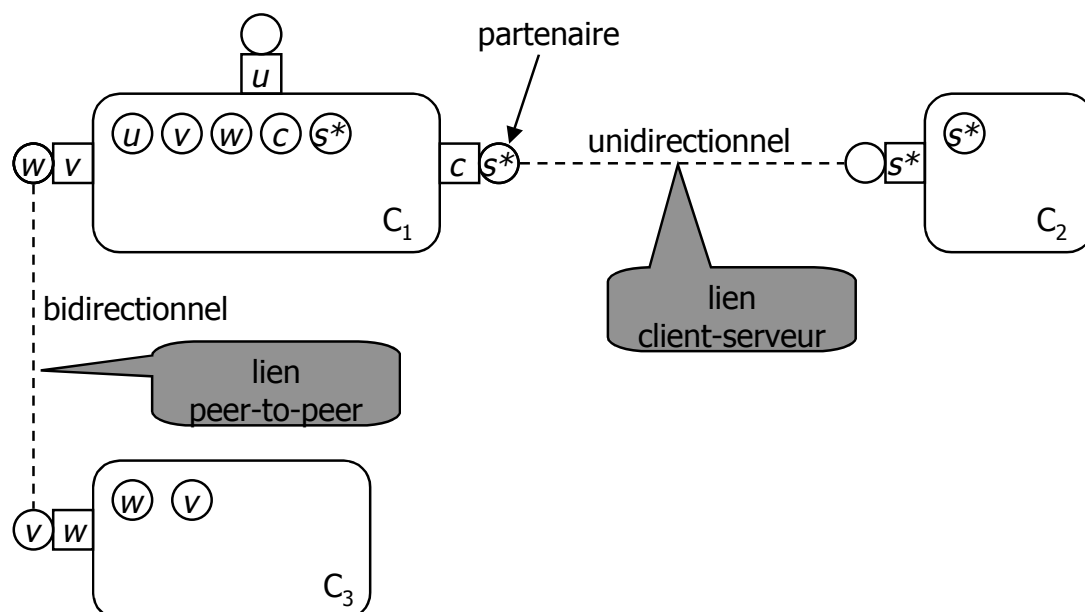
- Positionnement de notre approche
- Langage d'interface
- Sémantique de composant
 - modèle statique & dynamique
 - règles de sémantique opérationnelle
- Respect de Contrat
- Assemblage sain
- Conclusion & Perspectives

15/12/2003

Contrats Comportementaux pour Composants

19

Modèle de composant

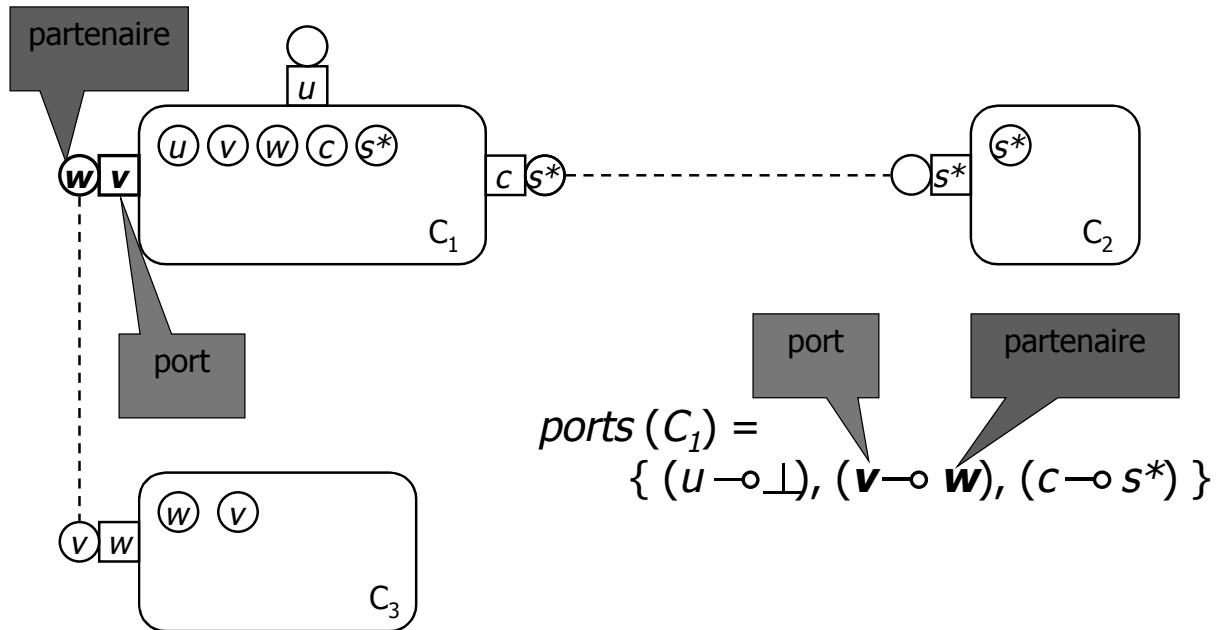


15/12/2003

Contrats Comportementaux pour Composants

20

Modèle de composant

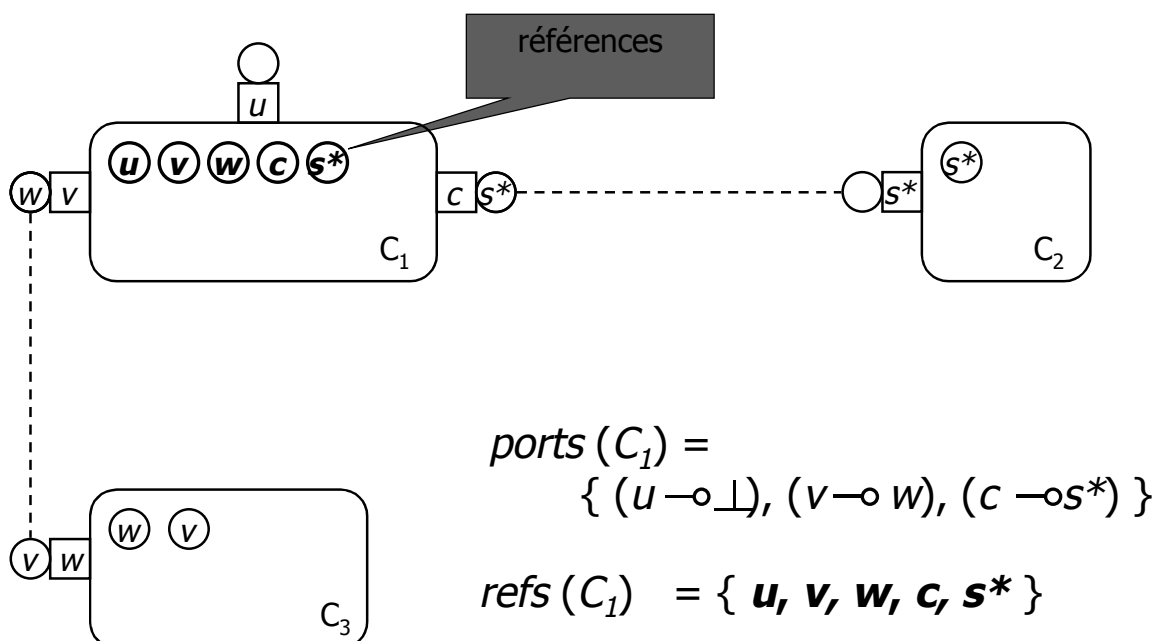


15/12/2003

Contrats Comportementaux pour Composants

21

Modèle de composant



15/12/2003

Contrats Comportementaux pour Composants

22

Modèle de composant: ports

- Modèle basé sur l'observation des ports
- Etat d'un port : $u\rho^\sigma$

$$- \rho = \text{action} = \begin{cases} ! & u \text{ est dans un état d'envoi} \\ ? & u \text{ est dans un état de réception} \\ \mathbf{0} & u \text{ n'a pas d'action} \end{cases}$$

$$- \sigma = \text{activité} = \begin{cases} \mathbf{a} & u \text{ est actif} \\ \mathbf{s} & u \text{ est suspendu} \\ \mathbf{i} & u \text{ est oisif} \end{cases}$$

v Exemple:

– $u ?^{\mathbf{a}}$ = actif en réception $u !^{\mathbf{s}}$ = suspendu en émission

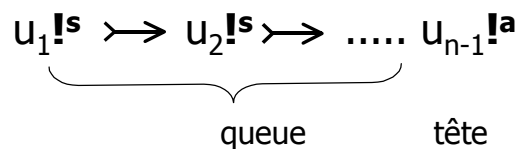
Modèle de composant: threads

- Composants multi-tâches
- Dépendances entre les ports: $x \succrightarrow y$
 - l'activité de x est suspendue jusqu'à ce que y termine ou devienne oisif
- Une tâche est une chaîne (*tête*, *queue*)
 - *tête*: port actif courant,
 - *queue*: séquence ordonnée de ports suspendus
 - peut augmenter/diminuer dynamiquement

$$\underbrace{u_1 !^{\mathbf{s}} \succrightarrow u_2 !^{\mathbf{s}} \succrightarrow \dots \succrightarrow u_{n-1} !^{\mathbf{s}}}_{\text{queue}} \succrightarrow u_n ?^{\mathbf{a}} \quad \text{tête}$$

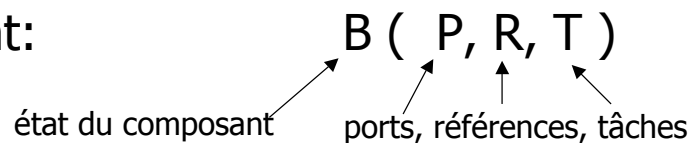
Modèle de composant: threads

- Composants multi-tâches
- Dépendances entre les ports: $x \succrightarrow y$
 - l'activité de x est suspendue jusqu'à ce que y termine ou devienne oisif
- Une tâche est une chaîne (*tête*, *queue*)
 - *tête*: port actif courant,
 - *queue*: séquence ordonnée de ports suspendus
 - peut augmenter/diminuer dynamiquement



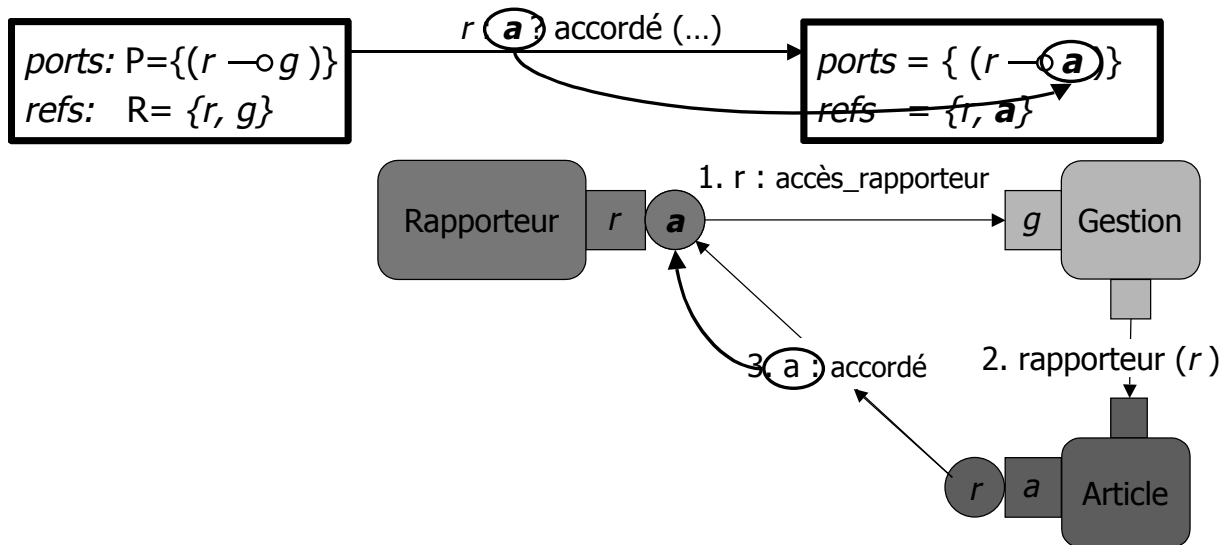
Sémantique du composant

- Un composant:



- Sémantique opérationnelle
 - $B(P, R, T), Com \longrightarrow B'(P', R', T'), Com'$
- 11 Règles:
 - création / suppression des ports
 - attachement
 - (dés)activation des ports (oisif, actif, suspendu)
 - envoi / réception de messages

Exemple: règle RECV pour le composant Rapporteur



$$T' = T[u\rho/u?] \quad R' = R \cup \{\text{refs}(\tilde{v}), u''\} - \{u' \mid (u -o u') \wedge \text{peer}(u')\}$$

$$Com' = Com[u] \quad P' = P[u -o u''] \text{ si } \text{peer}(u)$$

$$B(P, R, T), Com \xrightarrow{u:u''?M(\tilde{v})} B'(P', R', T'), Com'$$

Plan

- Positionnement de notre approche
- Langage d'interface
- **Sémantique de composant**
- Respect de Contrat
- Assemblage sain
- Conclusion & Perspectives

Composant et son contrat

■ Composant contractuel: $B(\dots), \tilde{C}$

– comportement correct

$$\frac{\tilde{C} \xrightarrow{\alpha} \tilde{C}' \quad B(\dots) \xrightarrow{a} B'(\dots) \quad a : \alpha}{B(\dots), \tilde{C} \xrightarrow{a:\alpha} B(\dots), \tilde{C}'}$$

– transition non autorisée

$$\frac{\tilde{C} \not\xrightarrow{\alpha} \tilde{C}' \quad B(\dots) \xrightarrow{a} B'(\dots) \quad a : \alpha}{B(\dots), \tilde{C} \xrightarrow{a:\alpha} Error}$$

– transition obligatoire manquante

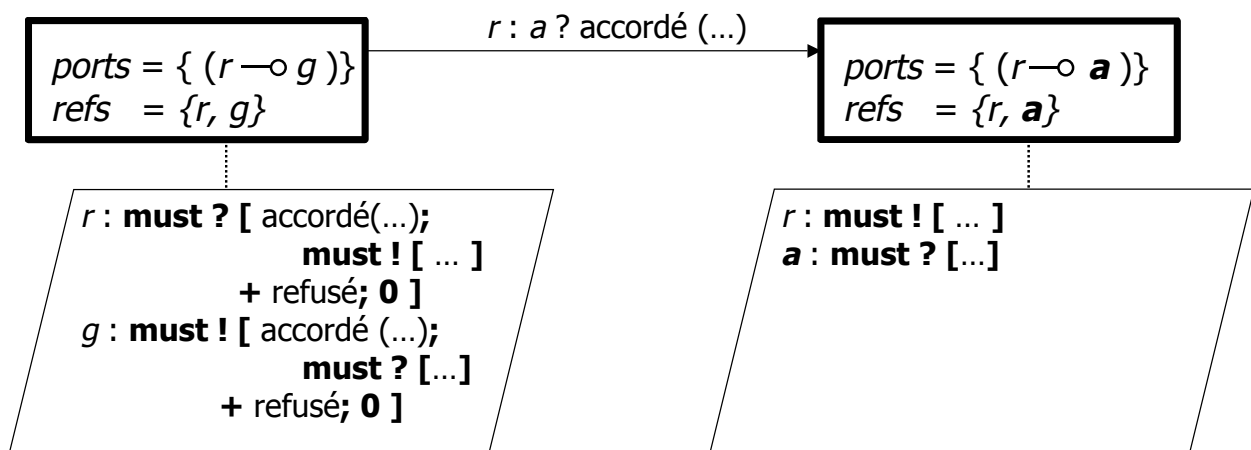
$$\frac{\tilde{C} \xrightarrow{\alpha} \tilde{C}' \quad B(\dots) \not\xrightarrow{a} B'(\dots) \quad a : \alpha \text{ mod } (\alpha) = \mathbf{must}}{B(\dots), \tilde{C} \xrightarrow{a:\alpha} Error}$$

15/12/2003

Contrats Comportementaux pour Composants

29

Exemple: règle RECV pour le composant Rapporteur



$$u : T \equiv \text{mod } ? M_{\Sigma}$$

$$u' : T' \equiv \text{mod}' ! M'_{\Sigma} \quad B(P, R, T) \xrightarrow{u, u' ? m_k} B'(P', R', T')$$

$$(B(P, R, T), \tilde{C}) \xrightarrow{u, u' ? m_k} (B'(P', R', T'), \tilde{C}[u : T_k / T, u' : T'_k / u' : T'] \Leftarrow \tilde{v}' : \tilde{U}'_k) \blacktriangle$$

Plan

- Positionnement de notre approche
- Langage d'interface
- **Sémantique de composant**
- Respect de Contrat
- Assemblage sain
- Conclusion & Perspectives

Assemblage sain de composants

- Composant honorant un contrat
 - B est bien typé: $B(P,R,T),\tilde{C}$ ne mène jamais à *Error*
- Assemblage de composants:

$$\mathcal{A} = \{ (B_1(P_1,R_1,T_1),\tilde{C}_1), \dots, (B_n(P_n,R_n,T_n),\tilde{C}_n), Com \}$$

- clos du point de vue des références
- uniquement des attachements client/serveur et peer-to-peer
- tous les ports sont actifs et indépendants
- Assemblage sain:
 - tous les composants respectent leur contrat
 - les ports attachés entre eux sont compatibles

Propriétés

Sûreté maintenue au cours de l'évolution

– une configuration saine de composants ne mène jamais à *Error*

$$\forall C : \mathcal{A} \longrightarrow * C, C \not\rightarrow Error$$

Tous les messages seront consommés

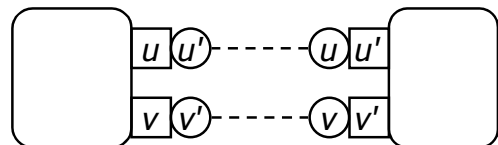
$$\forall u, v, i, M : (u \circ v) \in P_i, C \xrightarrow{u:v!M} C'$$

$$\Rightarrow \exists C'', C''' \text{ tel que } C' \longrightarrow * C'' \xrightarrow{v:u?M} C'''$$

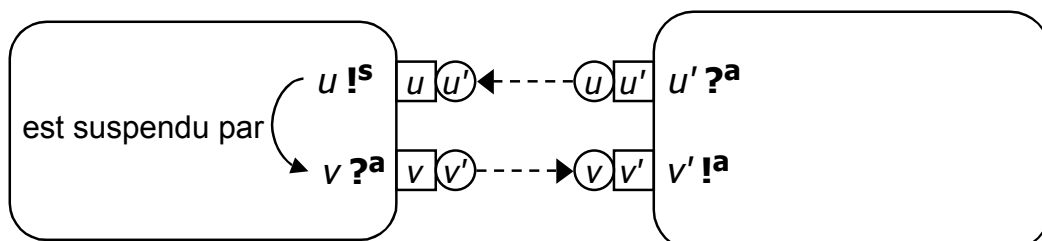
Interblocage externe

- A l'assemblage : pas de vérification du comportement global

- types de u et u' compatibles
- types de v et v' compatibles



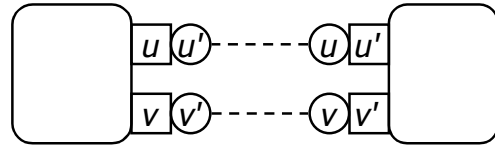
- En cours d'exécution :



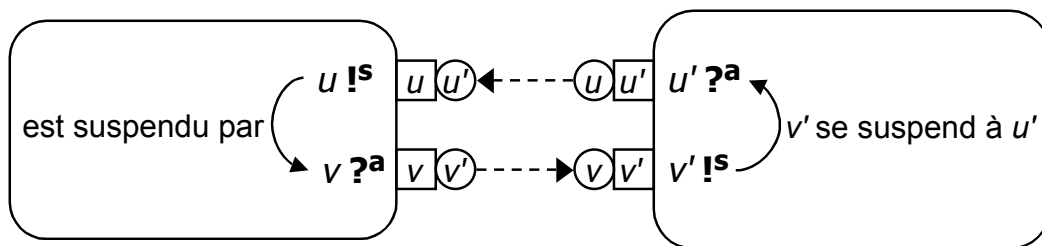
Interblocage externe

- A l'assemblage : pas de vérification du comportement global

- types de u et u' compatibles
- types de v et v' compatibles



- En cours d'exécution :



15/12/2003

Contrats Comportementaux pour Composants

35

Propriétés: pas d'interblocage externe

- Interdiction qu'un port se suspende à un port en réception

Interblocage externe:

- $u \mathcal{S} v \triangleq u \succrightarrow v \ \forall u \dashrightarrow v \quad (\dashrightarrow \text{ dépendance externe})$

- $\text{Ext_deadlock}(C) \triangleq$

$\exists (u_i)_{1..n} \in C \text{ telle que } \forall k < n : u_k \mathcal{S} u_{k+1} \wedge u_n \mathcal{S} u_1$

- Démonstration (absence d'interblocage):

- par induction & raisonnement par l'absurde

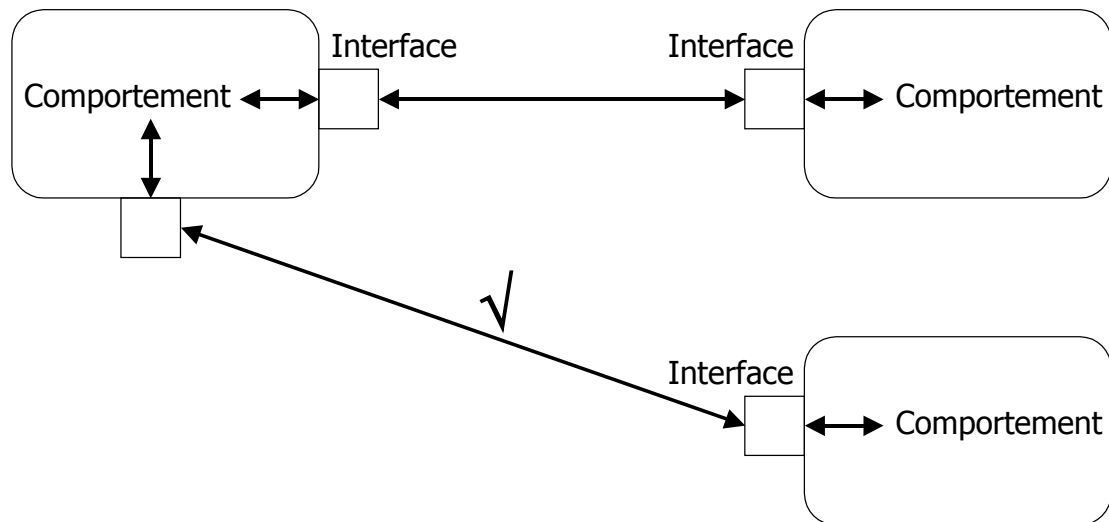
15/12/2003

Contrats Comportementaux pour Composants

36

Application

■ Extension saine en cours d'exécution

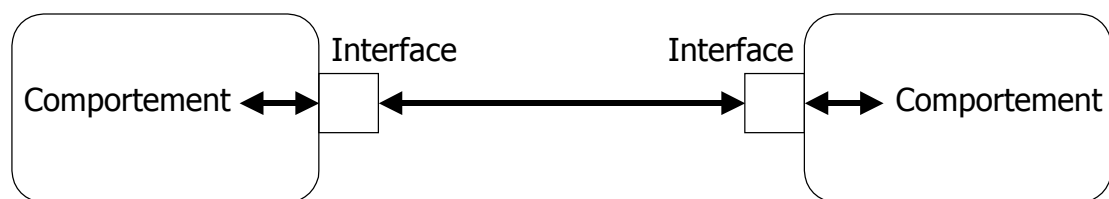


15/12/2003

Contrats Comportementaux pour Composants

37

Conclusion



- Conformance au contrat: \longleftrightarrow vérification à la compilation
- Compatibilité des interfaces: \longleftrightarrow vérification au déploiement

■ Propriétés de l'assemblage sain

- sûreté: une configuration ne mène jamais à *Error*
- sûreté: absence d'interblocage externe
- vivacité: tout message envoyé sera consommé

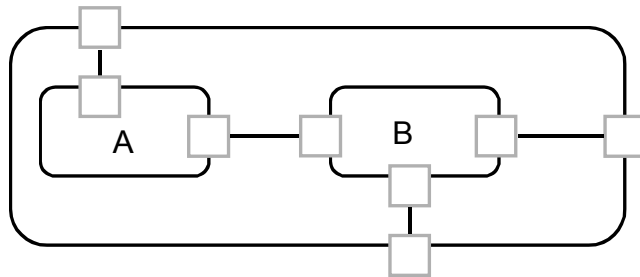
15/12/2003

Contrats Comportementaux pour Composants

38

Perspectives

- Interfaces: automates infinis
- Intégration aux plates-formes de composant
- Profil UML
- Composants composite & délégation:



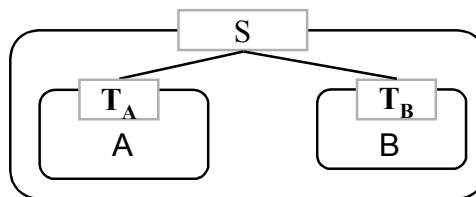
15/12/2003

Contrats Comportementaux pour Composants

39

Travaux futurs

- Application à UML2.0: délégation multiple



- Application à un langage
- Des contrats d'interface vers les contrats de composants
- Extension aux interfaces temporisées

15/12/2003

Contrats Comportementaux pour Composants

40