Une proposition de composant formel

Pascal Poizat[1] et Jean-Claude Royer[2]

[1]LaMI - Université d'Évry Val d'Essonne

[2]IRIN - Université de Nantes

Contexte

- Spécifications formelles mixtes
- Méthodes d’analyse et de conception à objets
- Architecture et composant
  - Codage : EJB, Active-X, COM, ...
  - Conception : UML-RT
  - Spécification d’architecture : Darwin, Olan, Rapide, ...
  - Spécification formelle d’architecture : WRIGHT

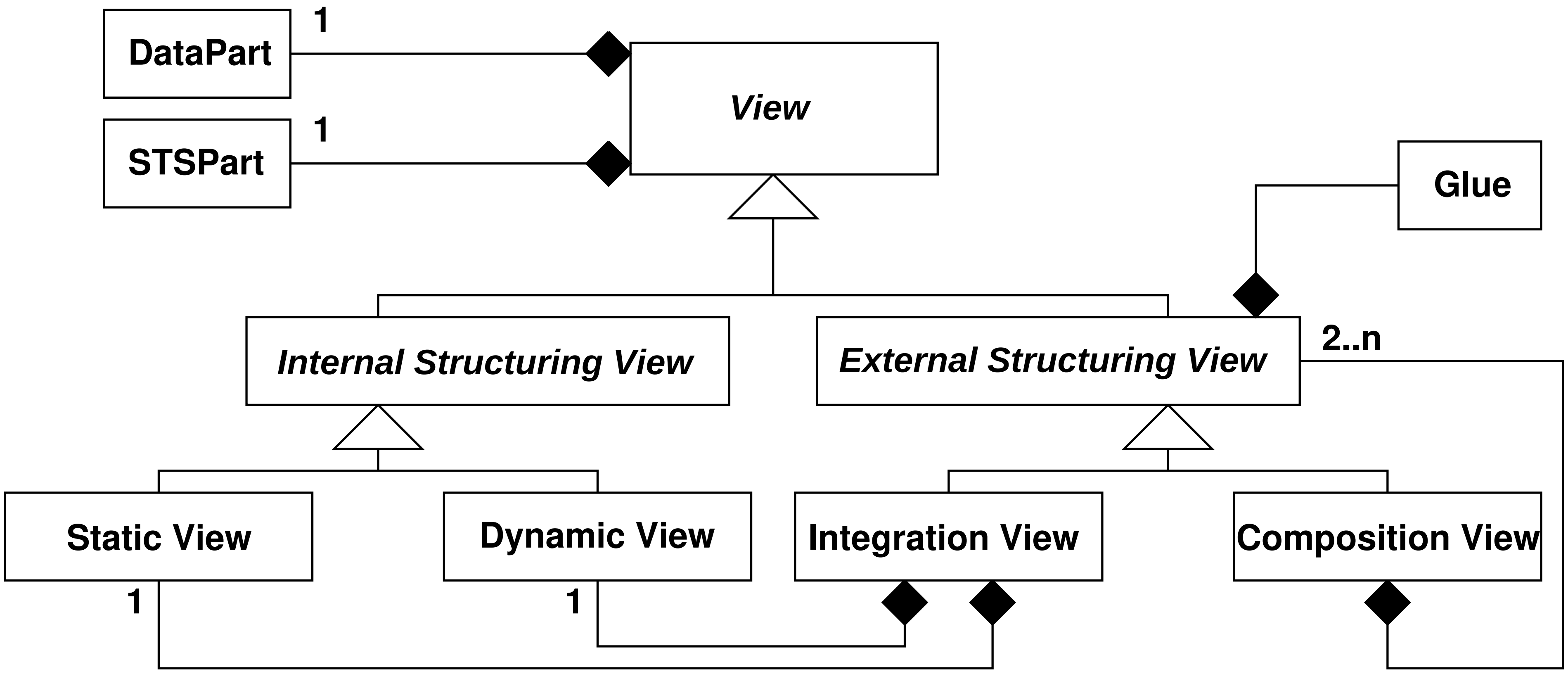
Les besoins

- Interfaces lisibles (abstraites, formelles, standard)
- Moyens de structuration et colle abstraite
- Notations graphiques et textuelles
- Sémantique formelle
- Technique de preuve
- Outils, génération de code pour langages à classes
- Méthode de développement

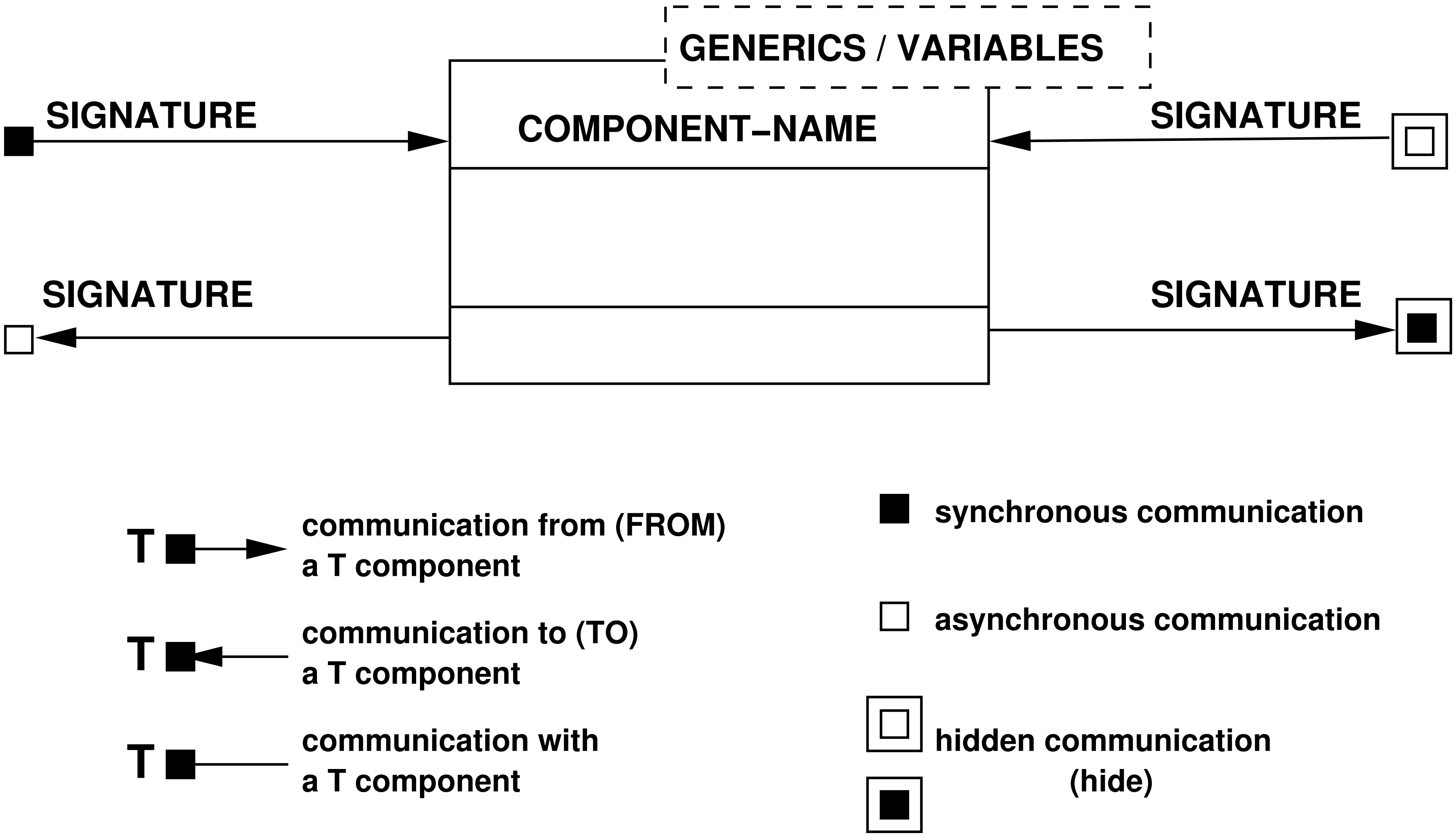
Plan

- Les vues en KORRIGAN
- Système de transitions symboliques
- Schéma d'architecture
- Vue d'intégration
- Une forme limitée d'héritage
- Communications et patrons
- Eléments de comparaisons
- Conclusion et perspectives

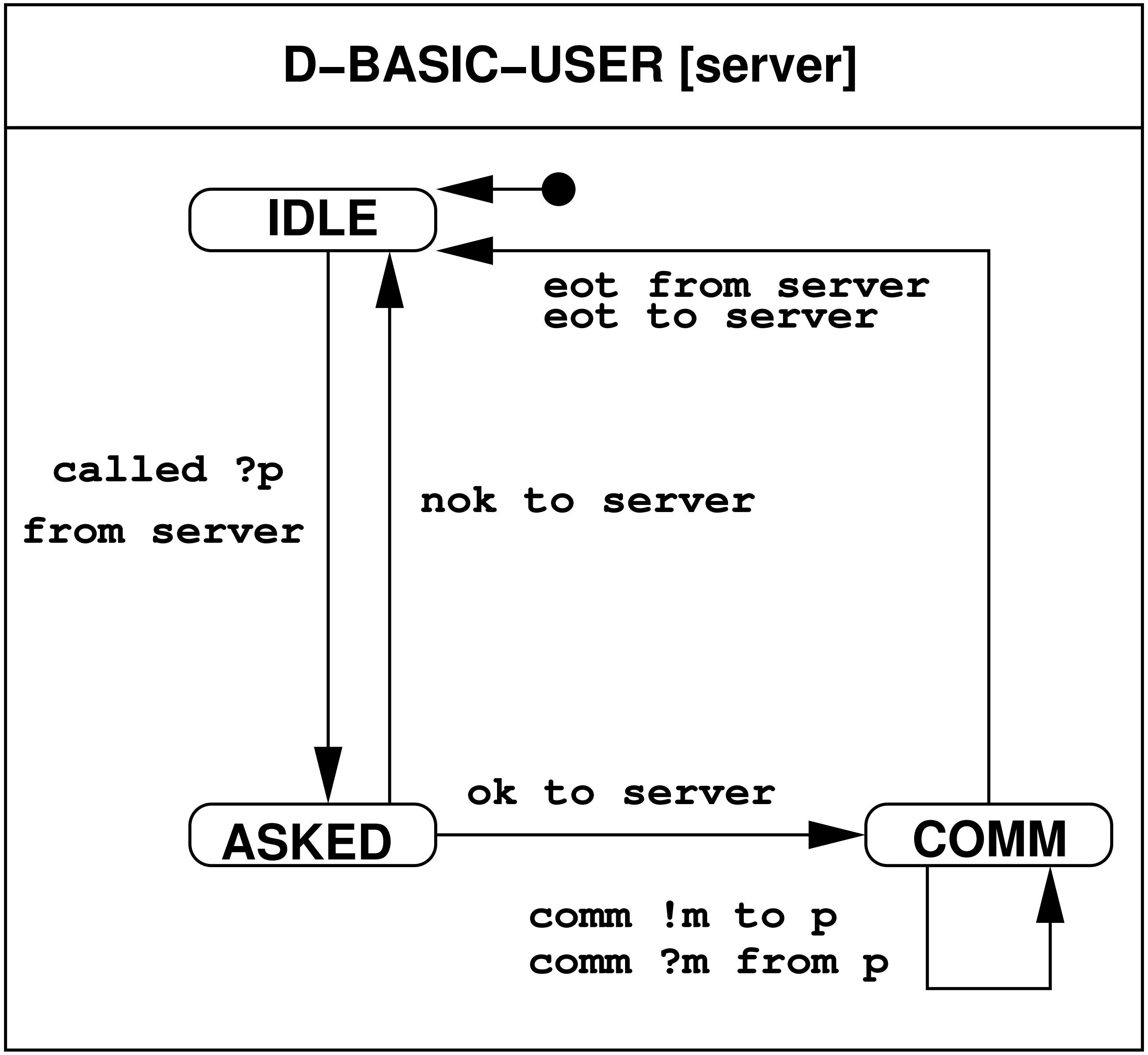
Les vues de KORRIGAN



Interface d'un composant



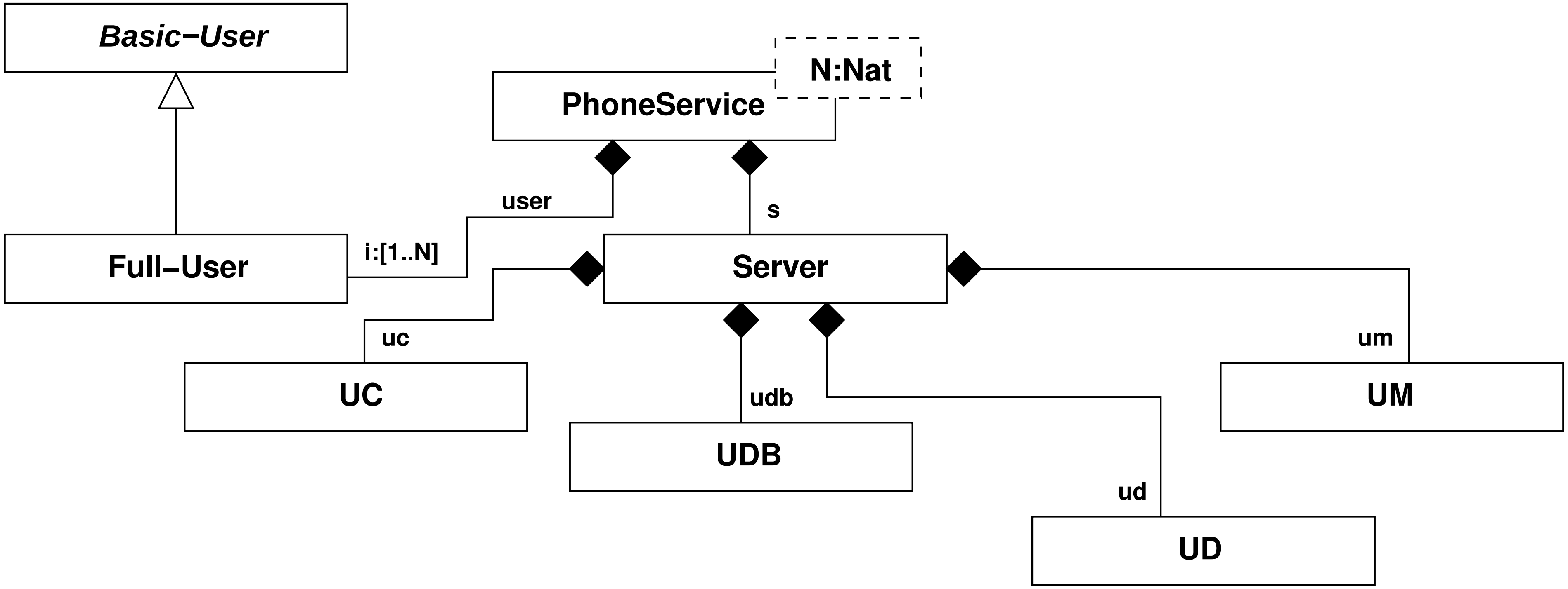
Système de transitions symboliques



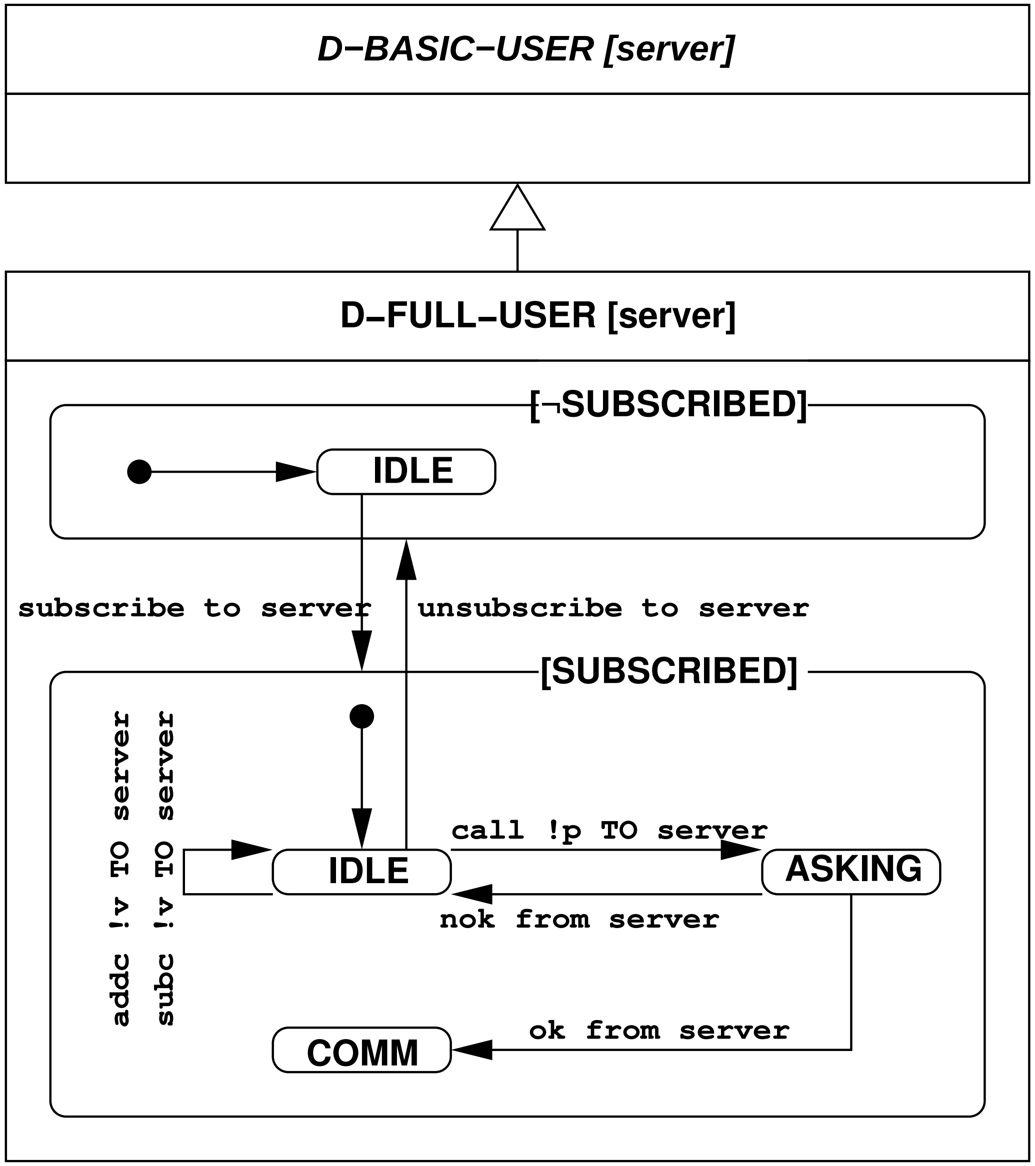
Vue textuelle

DYNAMIC VIEW D-BASIC-USER	
SPECIFICATION	**STS**
imports PId, MSG **generic on** server:PIdServer **ops** called ?p:PIdUser **from** server ok **to** server nok **to** server comm ?m:Msg **from** PIdUser comm !m:Msg **to** PIdUser eot **to** server eot **from** server	• → IDLE IDLE – called ?p:PIdUser **from** server → ASKED ASKED – ok **to** server → COMM ASKED – nok **to** server → IDLE COMM – comm ?m:Msg **from** p → COMM COMM – comm !m:Msg **to** p → COMM COMM – eot **to** server → IDLE COMM – eot **from** server → IDLE

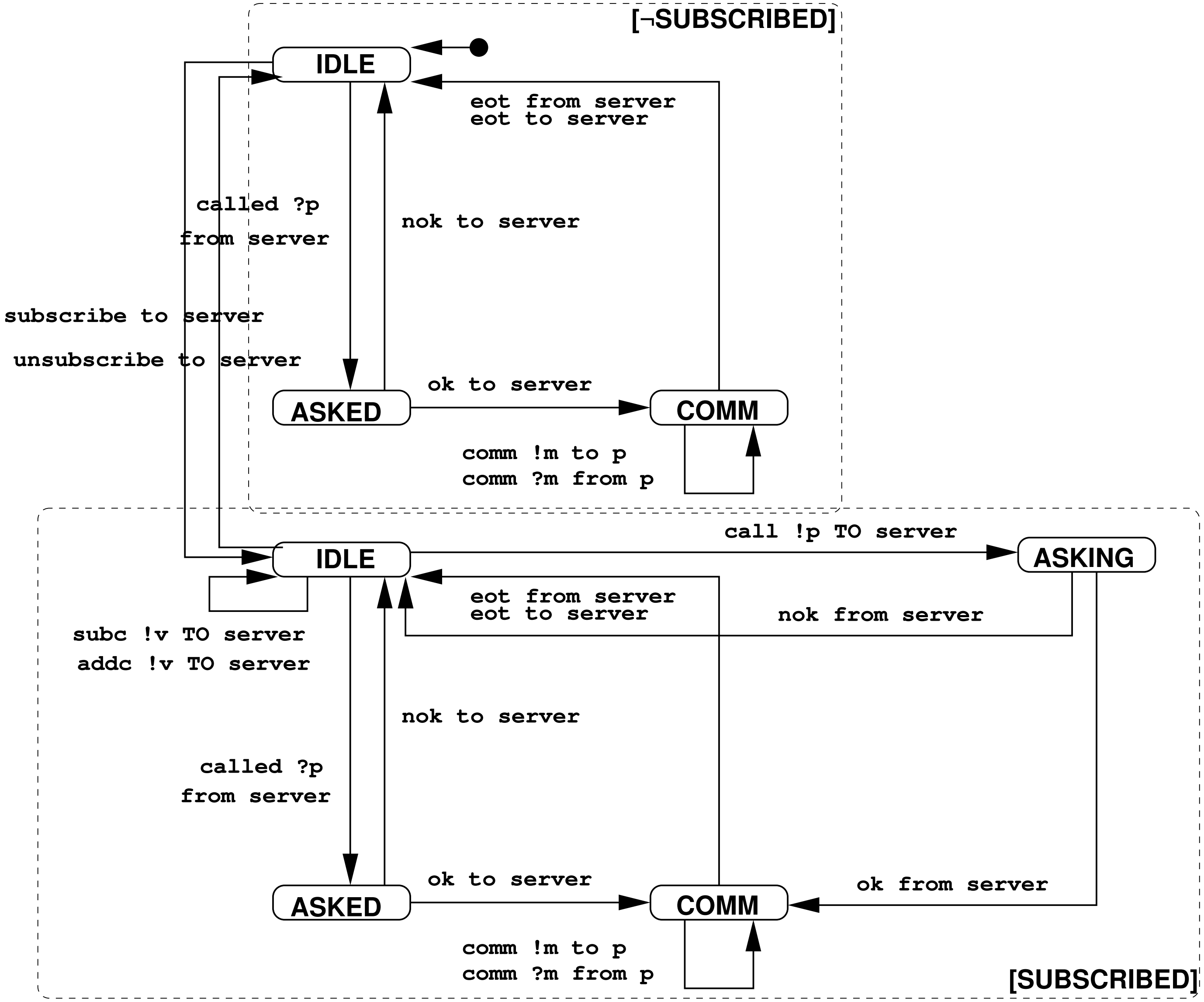
Architecture concurrente



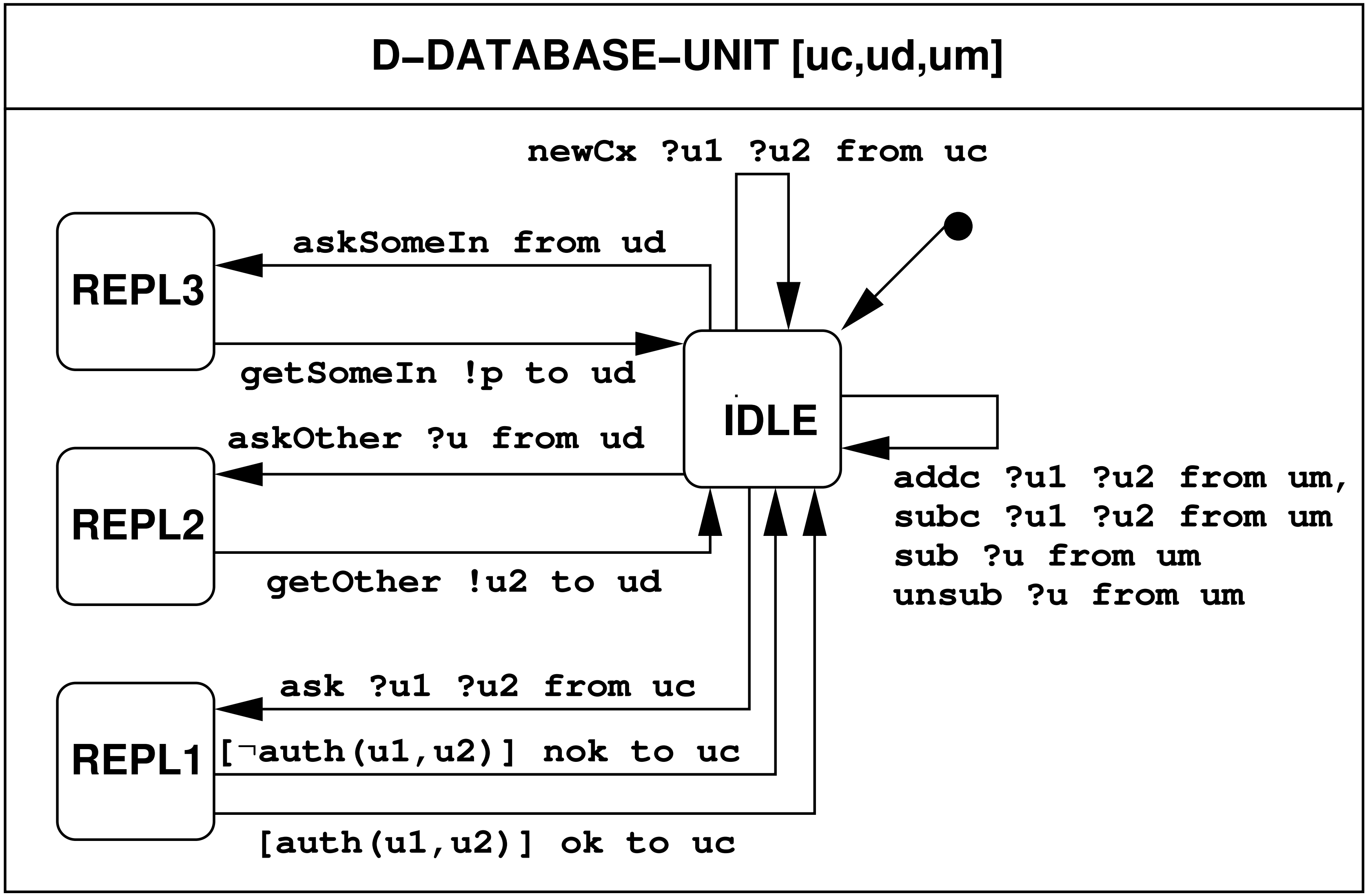
Forme limitée d'héritage



Héritage complet



L’unité de base de données



Vue d'intégration

INTEGRATION VIEW DATABASE-UNIT (UDB)

SPECIFICATION

generic on

uc:PIdUC, ud:PIdUD, um:PIdUM

COMPOSITION

is

sl : SET [PIdFullUser]
al : SET [Couple [PIdUser]]
cl : SET [Couple [PIdUser]]
d : D-DATABASE-UNIT [uc,ud,um]

axioms {al.member((u1,u2)) == d.auth(u1,u2)}

...

Vue d'intégration (suite)

INTEGRATION VIEW suite

TEMPORAL GLUE

with true, {
(sl.add(u), d.sub ?u **from** um)
(sl.remove(u), d.unsub ?u **from** um)
(al.add((u1,u2)), d.addc ?u1 ?u2 **from** um)
(al.remove((u1,u2)), d.subc ?u1 ?u2 **from** um)
(cl.add((u1,u2)), d.newCx ?u1 ?u2 **from** uc)
(cl.member(u, u2), d.getSomeIn !u **to** ud $\vee$ d.getSomeIn !u2 **to** ud)
(cl.member((u,u2)) $\vee$ cl.member((u2,u)),
d.($-$ askOther ?u **from** ud $>$ $-$ getOther !u2 **to** ud $>$))}
initially true

Patron client-serveur

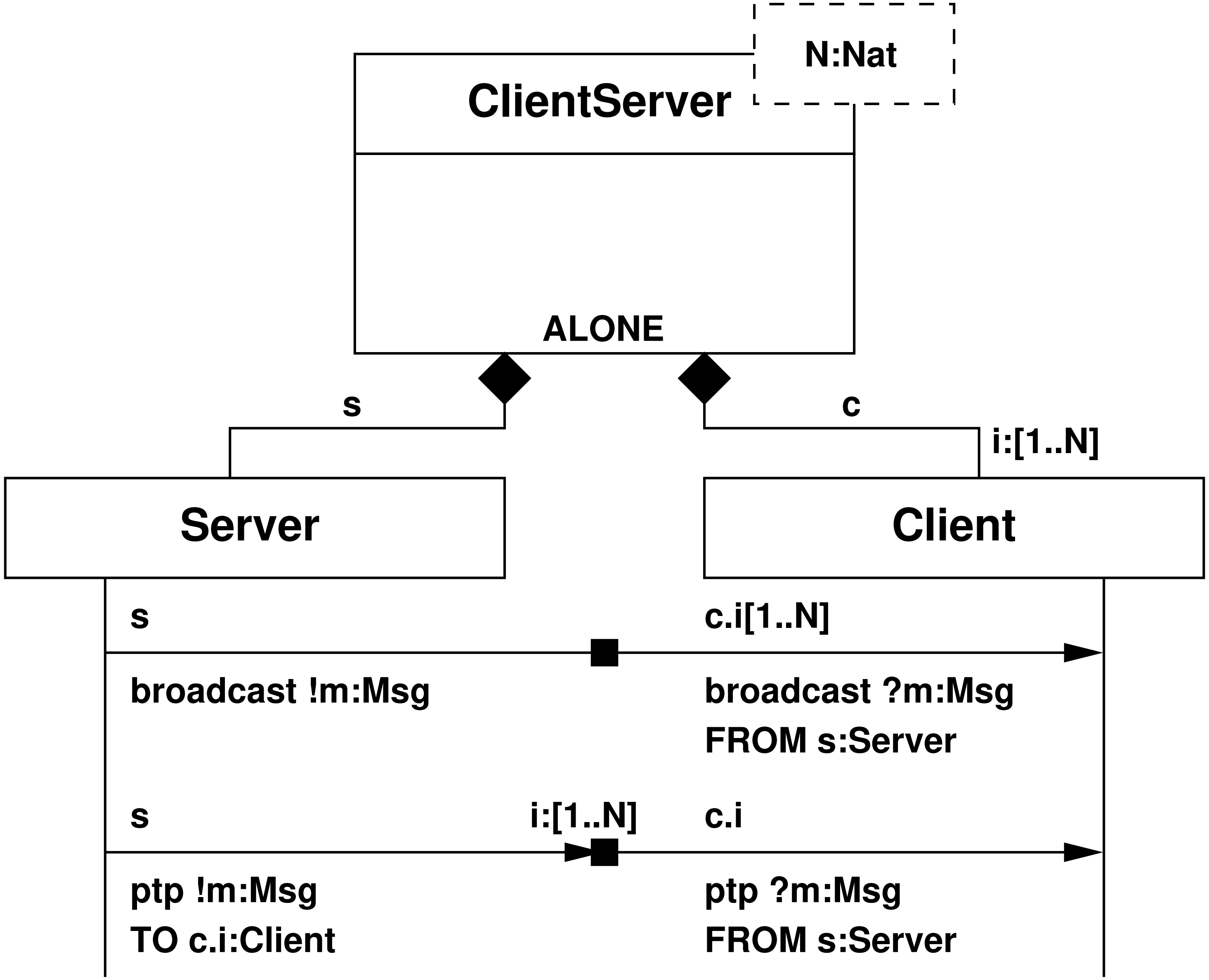


Diagramme de communication

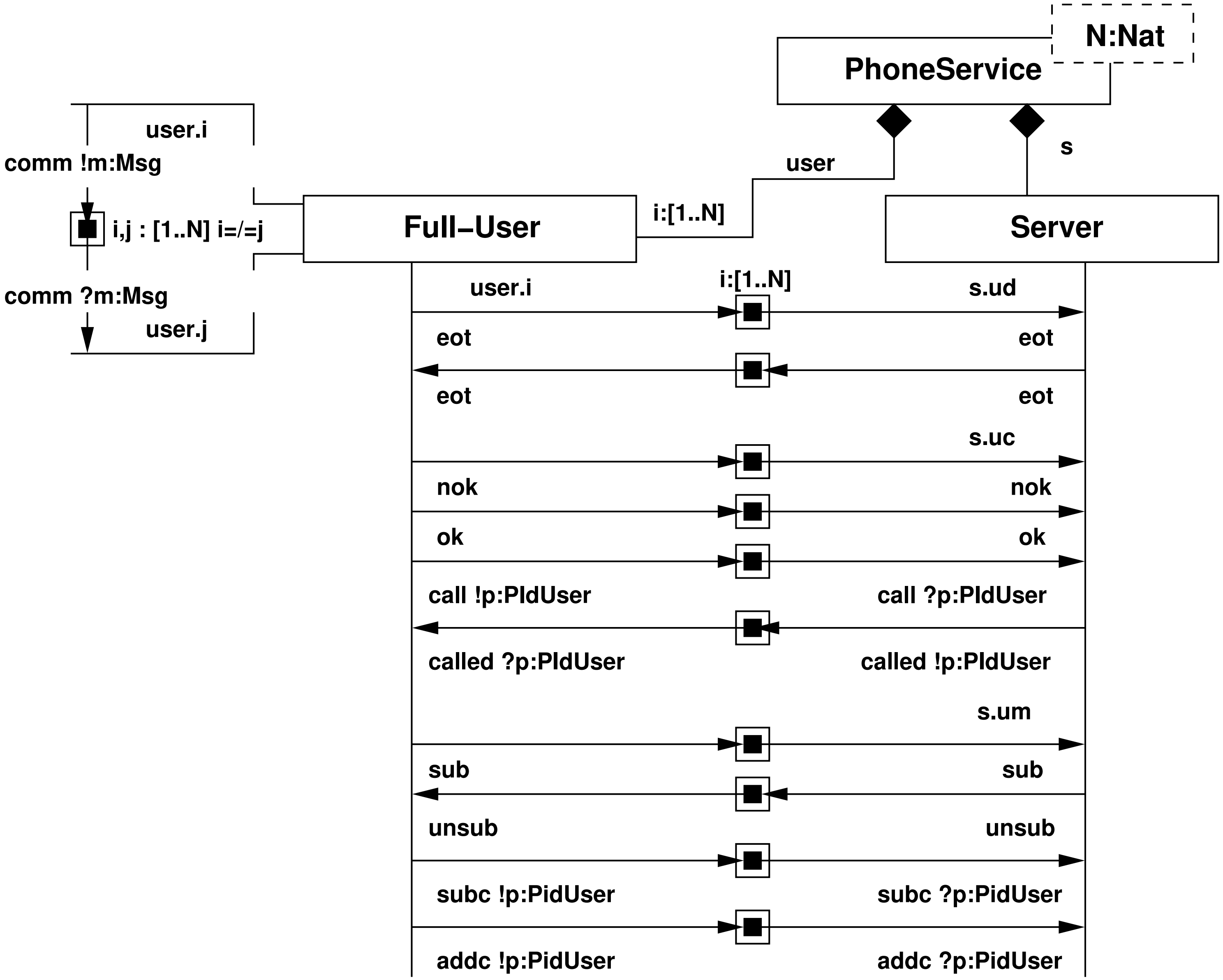
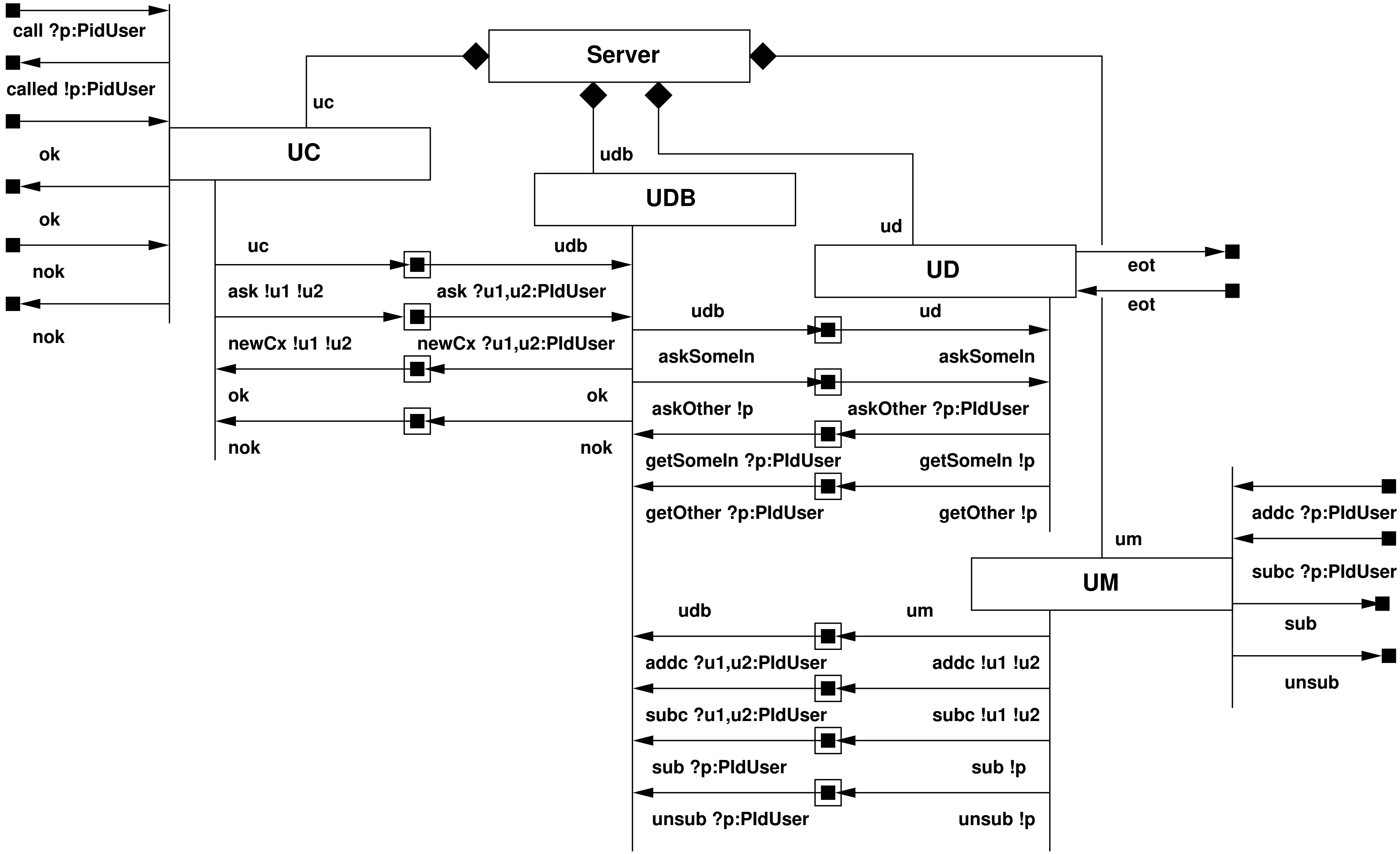


Diagramme de communication (suite)



Comparaisons

- WRIGHT
  - Habillage de CSP, vérifications dynamiques
  - Reconfiguration dynamique
  - Notations graphiques, vérifications dynamiques et fonctionnelles
- UML-RT
  - Niveau conception et applications temps rééls
  - Cohérence données - dynamiques ?
  - Différences au niveau communications
- Darwin, Rapide, Olan, ...

Conclusion et perspectives

- Korrigan : une base pour des composants formels
- Notations graphiques lisibles
- Sémantique formelle opérationnelle
- Génération de code à objets
- Méthode d'écriture des spécifications
- Futur
  - technique de preuve
  - reconfiguration dynamique, produit n-aire
  - génération pour CORBA
  - standard