

Composant : spécification et vérification

OBASCO Ecole des Mines de Nantes - INRIA, LINA

IRISA le 22 septembre 2006

- ▶ Niveau spécification formelle, l'aspect plus opérationnel étant principalement la tâche de Sebastian

Modèle de composant

- ▶ Niveau spécification formelle, l'aspect plus opérationnel étant principalement la tâche de Sebastian
- ▶ Un modèle hiérarchique

- ▶ Niveau spécification formelle, l'aspect plus opérationnel étant principalement la tâche de Sebastian
- ▶ Un modèle hiérarchique
- ▶ Composant = interface + protocole + description fonctionnelle

- ▶ Niveau spécification formelle, l'aspect plus opérationnel étant principalement la tâche de Sebastian
- ▶ Un modèle hiérarchique
- ▶ Composant = interface + protocole + description fonctionnelle
- ▶ Des protocoles évolués : système de transition symbolique

- ▶ Niveau spécification formelle, l'aspect plus opérationnel étant principalement la tâche de Sebastian
- ▶ Un modèle hiérarchique
- ▶ Composant = interface + protocole + description fonctionnelle
- ▶ Des protocoles évolués : système de transition symbolique
- ▶ Des communications synchrones et asynchrones

- ▶ Niveau spécification formelle, l'aspect plus opérationnel étant principalement la tâche de Sebastian
- ▶ Un modèle hiérarchique
- ▶ Composant = interface + protocole + description fonctionnelle
- ▶ Des protocoles évolués : système de transition symbolique
- ▶ Des communications synchrones et asynchrones
- ▶ Une colle complexe et expressive

- ▶ Niveau spécification formelle, l'aspect plus opérationnel étant principalement la tâche de Sebastian
- ▶ Un modèle hiérarchique
- ▶ Composant = interface + protocole + description fonctionnelle
- ▶ Des protocoles évolués : système de transition symbolique
- ▶ Des communications synchrones et asynchrones
- ▶ Une colle complexe et expressive
- ▶ DOA 2003, papier de revue en cours

- ▶ Description fonctionnelle sous forme algébrique

Spécification formelle

- ▶ Description fonctionnelle sous forme algébrique
- ▶ Cohérence avec le STS

Spécification formelle

- ▶ Description fonctionnelle sous forme algébrique
- ▶ Cohérence avec le STS
- ▶ Aide à l'extraction de la spécification

Spécification formelle

- ▶ Description fonctionnelle sous forme algébrique
- ▶ Cohérence avec le STS
- ▶ Aide à l'extraction de la spécification
- ▶ Technique de preuve (PVS)

- ▶ Description fonctionnelle sous forme algébrique
- ▶ Cohérence avec le STS
- ▶ Aide à l'extraction de la spécification
- ▶ Technique de preuve (PVS)
- ▶ Analyse des communications asynchrones peut aider (simplification, propriétés sur les communications)

- ▶ Description fonctionnelle sous forme algébrique
- ▶ Cohérence avec le STS
- ▶ Aide à l'extraction de la spécification
- ▶ Technique de preuve (PVS)
- ▶ Analyse des communications asynchrones peut aider (simplification, propriétés sur les communications)
- ▶ Lien entre la logique temporelle (CTL*) et une approche plus classique avec des types de données

- ▶ Description fonctionnelle sous forme algébrique
- ▶ Cohérence avec le STS
- ▶ Aide à l'extraction de la spécification
- ▶ Technique de preuve (PVS)
- ▶ Analyse des communications asynchrones peut aider (simplification, propriétés sur les communications)
- ▶ Lien entre la logique temporelle (CTL*) et une approche plus classique avec des types de données
- ▶ $AGEX \text{ true} \equiv \forall self : TI, \exists *, D_{TI}(self) \Rightarrow \bigvee_{op_R} (prec_{op_R}(self, *))$

- ▶ Description fonctionnelle sous forme algébrique
- ▶ Cohérence avec le STS
- ▶ Aide à l'extraction de la spécification
- ▶ Technique de preuve (PVS)
- ▶ Analyse des communications asynchrones peut aider (simplification, propriétés sur les communications)
- ▶ Lien entre la logique temporelle (CTL*) et une approche plus classique avec des types de données
- ▶ $AGEX \text{ true} \equiv \forall self : TI, \exists *, D_{TI}(self) \Rightarrow \bigvee_{op_R} (prec_{op_R}(self, *))$
- ▶ IASSE 2004, présentation à WADT 2004

- ▶ Difficultés dues aux gardes essentiellement

Vérification pour les protocoles

- ▶ Difficultés dues aux gardes essentiellement
- ▶ Analyse des boites à lettre en communication asynchrone

Vérification pour les protocoles

- ▶ Difficultés dues aux gardes essentiellement
- ▶ Analyse des boîtes à lettre en communication asynchrone
- ▶ Une condition suffisante, permet d'avoir une abstraction utile (DOA 2004) pour la vérification

Vérification pour les protocoles

- ▶ Difficultés dues aux gardes essentiellement
- ▶ Analyse des boites à lettre en communication asynchrone
- ▶ Une condition suffisante, permet d'avoir une abstraction utile (DOA 2004) pour la vérification
- ▶ Etude de la boundedness des STS

Vérification pour les protocoles

- ▶ Difficultés dues aux gardes essentiellement
- ▶ Analyse des boites à lettre en communication asynchrone
- ▶ Une condition suffisante, permet d'avoir une abstraction utile (DOA 2004) pour la vérification
- ▶ Etude de la boundedness des STS
- ▶ Algorithme de boundedness et notion de décomposition sont utiles (FMOODS 2006)

Vérification pour les protocoles

- ▶ Difficultés dues aux gardes essentiellement
- ▶ Analyse des boîtes à lettre en communication asynchrone
- ▶ Une condition suffisante, permet d'avoir une abstraction utile (DOA 2004) pour la vérification
- ▶ Etude de la boundedness des STS
- ▶ Algorithme de boundedness et notion de décomposition sont utiles (FMOODS 2006)
- ▶ Modélisation d'exemples avec propriété de disponibilité

Vérification pour les protocoles

- ▶ Difficultés dues aux gardes essentiellement
- ▶ Analyse des boîtes à lettre en communication asynchrone
- ▶ Une condition suffisante, permet d'avoir une abstraction utile (DOA 2004) pour la vérification
- ▶ Etude de la boundedness des STS
- ▶ Algorithme de boundedness et notion de décomposition sont utiles (FMOODS 2006)
- ▶ Modélisation d'exemples avec propriété de disponibilité
- ▶ Prototype support pour les STS en Python, réécriture en cours (Java ou C++)

Vérification pour les protocoles

- ▶ Difficultés dues aux gardes essentiellement
- ▶ Analyse des boîtes à lettre en communication asynchrone
- ▶ Une condition suffisante, permet d'avoir une abstraction utile (DOA 2004) pour la vérification
- ▶ Etude de la boundedness des STS
- ▶ Algorithme de boundedness et notion de décomposition sont utiles (FMOODS 2006)
- ▶ Modélisation d'exemples avec propriété de disponibilité
- ▶ Prototype support pour les STS en Python, réécriture en cours (Java ou C++)
- ▶ Suite dans le projet ECO-NET ?