

Composabilité et interfaces riches

Jacques Malenfant et Olena Rogovchenko

¹Université Pierre et Marie Curie-Paris 6, CNRS, UMR 7606 LIP6
contact : Jacques.Malenfant@lip6.fr

Journée « Ingénierie logiciel pour les systèmes hétérogènes 2011 »
25 novembre 2011

Introduction

Les systèmes hétérogènes sont confrontés à de multiples défis, dont les deux suivants sont parmi les plus importants :

- Comment construire une application qui **respecte** l'ensemble des contraintes posées tout en **s'abstrayant** le plus possible du substrat système/matériel/environnement sous-jacent ?
- Comment construire des applications de manière **compositionnelle** compte tenu des interactions multiples qui peuvent intervenir entre les composés ?

Introduction

Les systèmes hétérogènes sont confrontés à de multiples défis, dont les deux suivants sont parmi les plus importants :

- Comment construire une application qui **respecte** l'ensemble des contraintes posées tout en **s'abstrayant** le plus possible du substrat système/matériel/environnement sous-jacent ?
- Comment construire des applications de manière **compositionnelle** compte tenu des interactions multiples qui peuvent intervenir entre les composés ?
⇒ *feature interaction...*

Thèses

Dans le cadre des travaux d'Olena Rogovchenko, nous avons tenté de répondre à ces questions en mettant en avant deux thèses :

Thèses

Dans le cadre des travaux d'Olena Rogovchenko, nous avons tenté de répondre à ces questions en mettant en avant deux thèses :

- 1 Il est possible de s'assurer du respect des contraintes en maintenant une abstraction boîte noire des **détails opérationnels** tout en offrant une boîte de verre **déclarative** sur les propriétés de composants et des ressources matérielles.

Thèses

Dans le cadre des travaux d'Olena Rogovchenko, nous avons tenté de répondre à ces questions en mettant en avant deux thèses :

- 1 Il est possible de s'assurer du respect des contraintes en maintenant une abstraction boîte noire des **détails opérationnels** tout en offrant une boîte de verre **déclarative** sur les propriétés de composants et des ressources matérielles.
⇒ *Interfaces riches à la Henzinger*

Thèses

Dans le cadre des travaux d'Olena Rogovchenko, nous avons tenté de répondre à ces questions en mettant en avant deux thèses :

- 1 Il est possible de s'assurer du respect des contraintes en maintenant une abstraction boîte noire des **détails opérationnels** tout en offrant une boîte de verre **déclarative** sur les propriétés de composants et des ressources matérielles.
⇒ *Interfaces riches à la Henzinger*
- 2 La compositionnalité doit se fonder sur des **algorithmes de composition intrusifs**, c'est-à-dire qui ne s'arrêtent pas au frontières des composants, mais sont autorisés à « voir » (une partie de) l'intérieur des composants lors de la composition.

Thèses

Dans le cadre des travaux d'Olena Rogovchenko, nous avons tenté de répondre à ces questions en mettant en avant deux thèses :

- 1 Il est possible de s'assurer du respect des contraintes en maintenant une abstraction boîte noire des **détails opérationnels** tout en offrant une boîte de verre **déclarative** sur les propriétés de composants et des ressources matérielles.
⇒ *Interfaces riches à la Henzinger*
- 2 La compositionnalité doit se fonder sur des **algorithmes de composition intrusifs**, c'est-à-dire qui ne s'arrêtent pas aux frontières des composants, mais sont autorisés à « voir » (une partie de) l'intérieur des composants lors de la composition.
⇒ *Modèle interne + interface de composition.*

Plan

- 1 Introduction
- 2 Interfaces riches**
- 3 Composition
- 4 Conclusion

Fondements

- Concept introduit par Henzinger et de Alvaro pour capturer la richesse des interactions entre les composants.
- Deux principes fondamentaux de l'approche par composants :
 - ① la conception incrémentale, et
 - ② l'implantation indépendante.
- Ces objectifs sont atteints en faisant en sorte que les interfaces riches exposent toute l'information **nécessaire** et mais uniquement celle qui est **suffisante**.
⇒ *contrainte de conception des interfaces riches !*

Types d'interfaces riches

- *Interface automatats* (2001)
- *Timed interfaces* (2002)
- *Resource interfaces* (2003)

Types d'interfaces riches

- *Interface automatats* (2001)
- *Timed interfaces* (2002)
- *Resource interfaces* (2003)

- *Théorie sous-jacente permettant de vérifier la compatibilité des composants en ne vérifiant que la compatibilité de leurs interfaces.*

Types d'interfaces riches

- *Interface automatas* (2001)
- *Timed interfaces* (2002)
- *Resource interfaces* (2003)

- *Théorie sous-jacente permettant de vérifier la compatibilité des composants en ne vérifiant que la compatibilité de leurs interfaces.*
- Dans l'approche de l'équipe d'Henzinger, cela reste assez spécifique, avec des propriétés fixées à la conception.

Types d'interfaces riches

- *Interface automatats* (2001)
- *Timed interfaces* (2002)
- *Resource interfaces* (2003)

- *Théorie sous-jacente permettant de vérifier la compatibilité des composants en ne vérifiant que la compatibilité de leurs interfaces.*
- Dans l'approche de l'équipe d'Henzinger, cela reste assez spécifique, avec des propriétés fixées à la conception.
- Notre approche : vision plus « contraintes » et composition comme moyen de rendre compatible des interfaces avec « variables libres ».

Focus : nos interfaces de ressources

Modèles théorique : $I_R(n, \Gamma, \Pi, \beta, csts)$, où

- Γ : description de ressources, $\rho(n, \tau, \epsilon, \mathcal{P})$ où
 - τ : type de ressource (*p.e.*, *processeur*),
 - $\epsilon \subseteq Ide_\epsilon$: identifiants des points d'entrée,
 - \mathcal{P} : propriétés $p(n, u)$ pouvant être contraintes.
- Π : ports et leurs points d'entrée $\pi(n, \epsilon)$.
- $\beta : \Pi \rightarrow \Gamma$ liaisons des ports aux ressources.
- $csts$ contraintes $cst(t, ps, e)$ où
 - $t \in Ide_{cst}$: type de contrainte,
 - ps : identifiants des propriétés contraintes,
 - e : expression de la contrainte.

Indépendance envers la syntaxe de surface (à la Java, IDL, UML/Marte, ...).

Plan

- 1 Introduction
- 2 Interfaces riches
- 3 Composition**
- 4 Conclusion

Composition horizontale

Composition d'interfaces requises pour donner des interfaces requises composites :

- ① Identification des ressources communes
- ② Constructions des ensembles de contraintes maximaux.

$$I_R(n_1, \Gamma_1, \Pi_1, \beta_1, csts_1) \oplus_{(n', csts')} I_R(n_2, \Gamma_2, \Pi_2, \beta_2, csts_2) = \\ I_R(n', \Gamma_1 \cup \Gamma_2, \Pi_1 \cup \Pi_2, \beta_1 \oplus \beta_2, csts_1 \sqcap csts_2 \sqcap csts')$$

où

- $\Gamma_1 \cup \Gamma_2$: union des descriptions de ressources.
- $\Pi_1 \cup \Pi_2$: union des ports et des points d'entrée.
- $\beta_1 \oplus \beta_2$: combinaison des correspondances.
- $csts_1 \sqcap csts_2 \sqcap csts'$: plus grande borne inférieure selon la relation d'implication ($cs \models cs'$).

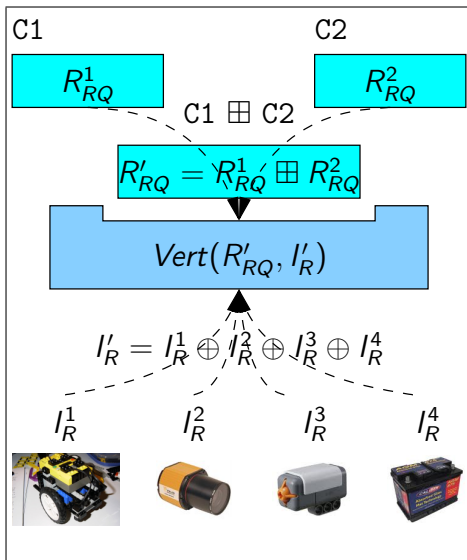
Composition verticale

- 1 Médiation des niveaux d'abstraction dans le but global.
- 2 Identification des ports et points d'entrée à connecter.
- 3 Résolution conjointe de l'ensemble des contraintes.
⇒ *vérification de compatibilité!*
- 4 Allocation des ressources (aspect quantitatif).
⇒ *récupération des liaisons de la résolution de contraintes!*
- 5 Calcul de l'ordonnancement (pour le cas temps-réel).
⇒ *idem!*

Processus général de composition

En résumé :

- Ressources requises
 R_{RQ}^1, R_{RQ}^2 + buts globaux.
- Composition des interfaces requises : R'_{RQ} .
- Ressources offertes
 $I_R^1, I_R^2, I_R^3, I_R^4$.
- Composition des ressources offertes : I'_R .
- Composition verticale :
conteneur $Vert(R'_{RQ}, I'_R)$.



Exemple complet : composition horizontale

LineFollower :

```
RRQ({r(m1, Motor, ...), r(m2, Motor, ...), r(c1, Camera, ...)},
    {rct(m1, speed, (≥, 0.25, Hz)), rct(m2, speed, (≥, 0.25, Hz)),
     rct(c1, resolution, (≥, 600 × 600))},
    {goal(autonomy, ·, (>, 240, min))})
```

ObstacleAvoider :

```
RRQ({r(mleft, Motor, ...), r(mright, Motor, ...), r(cs1, ContactSensor, ...)},
    {rct(m1, speed, (≥, 0.40, Hz)), rct(m2, speed, (≥, 0.40, Hz)),
     rct(cs1, frequency, (≥, 5, Hz))},
    {goal(autonomy, ·, (>, 300, min))})
```

LineFollower ⊞ ObstacleAvoider :

```
RRQ({r(ml, Motor, ...), r(mr, Motor, ...), r(c1, Camera, ...), r(cs1, ContactSensor, ...)},
    {rct(ml, speed, (≥, 0.40, Hz)), rct(mr, speed, (≥, 0.40, Hz)),
     rct(c1, resolution, (≥, 600 × 600, ·))},
    {rct(cs1, frequency, (≥, 5, Hz))},
    {goal(autonomy, ·, (>, 300, min))})
```

Exemple complet : interfaces des ressources

$$I_{\mathcal{R}}(\text{twoWheell},$$

$$\{ \rho(\text{leftM}, \text{Motor}, \{\text{go}, \text{stop}, \text{speed}\}, \{p(\text{speed}, 0.5, \text{Hz}), p(\text{resp_delay}, 50, \text{ms})\}),$$

$$\rho(\text{rightM}, \text{Motor}, \{\text{go}, \text{stop}, \text{speed}\}, \{p(\text{speed}, 0.5, \text{Hz}), p(\text{resp_delay}, 50, \text{ms})\}) \},$$

$$\{ \pi(\text{Imp}, \{\text{start}, \text{stop}, \text{set_speed}\}), \pi(\text{rmp}, \{\text{start}, \text{stop}, \text{set_speed}\}) \},$$

$$\{ \text{Imp.start} \mapsto \text{leftM.go}, \dots \},$$

$$\{ \text{cst}(\text{energy_cons}, \{t\}, 2 \times (t < 10 ? (0.004 - 0.00025 \times t : 0.0015) \times t)) \})$$

$$I_{\mathcal{R}}(\text{cameral},$$

$$\{ \rho(\text{cam}, \text{Camera}, \{\text{start}, \text{stop}, \text{turn}\}, \{p(\text{resolution}, \{360 \times 360, 800 \times 800\}, \cdot)\}) \},$$

$$\{ \pi(\text{cp}, \{\text{start}, \text{stop}, \text{turn}\}) \},$$

$$\{ \text{cp.start} \mapsto \text{cam.start}, \dots \},$$

$$\{ \text{cst}(\text{energy_cons}, \{t, (\text{camera}, \text{resolution})\},$$

$$\text{resolution} = 800 \times 800 ?$$

$$((t < 10 ? 0.002 - 0.0001 \times t : 0.001) \times t)$$

$$: \text{resolution} = 360 \times 360 ?$$

$$((t < 10 ? 0.0015 - 0.0007 \times t : 0.0008) \times t)) \})$$

$$I_{\mathcal{R}}(\text{contactl},$$

$$\{ \rho(\text{cont}, \text{ContactSensor}, \{\text{touch}\}, \{p(\text{frequency}, 10, \text{Hz})\}) \},$$

$$\{ \pi(\text{csp}, \{\text{touch}\}) \},$$

$$\{ \text{csp.touch} \mapsto \text{cont.touch} \},$$

$$\{ \text{cst}(\text{energy_cons}, \{t\}, 0.0005 \times t) \})$$

$$I_{\mathcal{R}}(\text{batteryI}, \{ \rho(b, \text{Battery}, \emptyset, \{p(\text{capacity}, 100000, \text{mAh})\}) \}, \emptyset, \emptyset, \emptyset)$$

Exemple complet : composition horizontale des ressources

$$\oplus\{\text{compresl}\}(I_R(\text{twoWheel}, \dots), I_R(\text{cameral}, \dots), I_R(\text{contactl}, \dots), I_R(\text{battery}, \dots)) =$$

$$I_R(\text{compresl}, \{\rho(\text{leftM}, \text{Motor}, \{\text{go}, \text{stop}, \text{speed}\}, \{p(\text{speed}, 0.5, \text{Hz}), p(\text{resp_delay}, 50, \text{ms})\}),$$

$$\rho(\text{rightM}, \text{Motor}, \{\text{go}, \text{stop}, \text{speed}\}, \{p(\text{speed}, 0.5, \text{Hz}), p(\text{resp_delay}, 50, \text{ms})\}),$$

$$\rho(\text{cam}, \text{Camera}, \{\text{start}, \text{stop}, \text{turn}\}, \{p(\text{resolution}, \{360 \times 360, 800 \times 800\}, \cdot)\}),$$

$$\rho(\text{cont}, \text{ContactSensor}, \{\text{touch}\}, \{p(\text{frequency}, 10, \text{Hz})\}),$$

$$\rho(\text{b}, \text{Battery}, \emptyset, \{p(\text{capacity}, 100000, \text{mAh})\}),$$

$$\{\pi(\text{Imp}, \{\text{start}, \text{stop}, \text{set_speed}\}), \pi(\text{rmp}, \{\text{start}, \text{stop}, \text{set_speed}\}),$$

$$\pi(\text{cp}, \{\text{start}, \text{stop}, \text{turn}\}), \pi(\text{csp}, \{\text{touch}\})\},$$

$$\{\text{Imp.start} \mapsto \text{leftM.go}, \dots, \text{cp.start} \mapsto \text{cam.start}, \dots, \text{csp.touch} \mapsto \text{cont.touch}\},$$

$$\{\text{cst}(\text{energy_cons}, \{t, (\text{camera}, \text{resolution})\},$$

$$\text{resolution} = 800 \times 800 ?$$

$$((t < 10 ? (0.002 - 0.0001 \times t) + 2 \times (0.004 - 0.00025 \times t) + 0.0005$$

$$: (0.001 + 2 \times 0.0015 + 0.0005) \times t)$$

$$: \text{resolution} = 360 \times 360 ?$$

$$((t < 10 ? (0.0015 - 0.0007 \times t) + 2 \times (0.004 - 0.00025 \times t) + 0.0005$$

$$: (0.0008 + 2 \times 0.0015 + 0.0005) \times t)\})\}$$

Exemple complet : élaboration du but global

Objectif : autonomie de 5 heures (300 minutes)

- 1 Calcul de la longueur du cycle : 50ms
- 2 Consommation par cycle : 0,195mAh
- 3 5 heures = 360.000 cycles
- 4 Consommation maximale : 70.200 mAh

La capacité de la batterie étant de 100.000mAh, le but est atteint malgré la résolution de 800×800 requise.

Exemple complet : composition verticale

```

Vert{container}(RRQ(...), IR(compresl, ...)) =
  container(..., // component deployment model
    {ρ(leftM, Motor, {go, stop, speed}, {p(speed, 0.5, Hz), p(resp_delay, 50, ms)}),
      ρ(rightM, Motor, {go, stop, speed}, {p(speed, 0.5, Hz), p(resp_delay, 50, ms)}),
      ρ(cam, Camera, {start, stop, turn}, {p(resolution, {800 × 800}, ·)}),
      ρ(cont, ContactSensor, {touch}, {p(frequency, 10, Hz)}),
      ρ(b, Battery, ∅, {p(capacity, 100000, mAh)}),
      {π(lmp, {start, stop, set_speed}), π(rmp, {start, stop, set_speed})},
      π(cp, {start, stop, turn}), π(csp, {touch})},
    {lmp.start ↦ leftM.go, ..., cp.start ↦ cam.start, ..., csp.touch ↦ cont.touch},
    {cst(energy_cons, {t, (camera, resolution)}),
      ((t < 10-7 ? (0.002 - 0.0001 × t) + 2 × (0.004 - 0.00025 × t) + 0.0005
        : (0.001 + 2 × 0.0015 + 0.0005) × t))})
  
```

Plan

- 1 Introduction
- 2 Interfaces riches
- 3 Composition
- 4 Conclusion**

Conclusion

Notre approche nécessite :

- une panoplie d'interfaces riches
- une panoplie d'opérateurs de composition
- un modèle minimal du fonctionnement interne des composants
⇒ partiellement fait pour les composants réactifs

Conclusion

Notre approche nécessite :

- une panoplie d'interfaces riches
- une panoplie d'opérateurs de composition
- un modèle minimal du fonctionnement interne des composants
⇒ partiellement fait pour les composants réactifs

Perspectives :

- axiomatique des opérateurs (associativité, distributivité, ...)
- démonstration empirique de la fermeture de l'ensemble des opérateurs de composition
- et, à terme, la modélisation de l'environnement physique
⇒ vers les « Cyber-Physical Systems »...

Merci! Questions?

Bibliographie de nos travaux



Olena Rogovchenko and Jacques Malenfant.

Composants et composition pour les architectures de contrôle de robots.

Revue des systèmes série JESA, Journal Européen des Systèmes Automatisés, 42(4) :423–438, mai 2008.



Olena Rogovchenko and Jacques Malenfant.

Composition and Compositionality in a Component Model for Autonomous Robots.

In *Proceedings of Software Composition, SC 2010*, number 6144 in Lecture Notes in Computer Science, pages 34–49. Springer-Verlag, July 2010.



Olena Rogovchenko and Jacques Malenfant.

Handling Hardware Heterogeneity through Rich Interfaces in a Component Model for Autonomous Robotics.

In *2nd International Conference on Simulation, Modeling and Programming for Autonomous Robots, SIMPAR 2010*, number 6472 in Lecture Notes in Artificial Intelligence, pages 312–323. Springer-Verlag, 2010.



Olena Rogovchenko and Jacques Malenfant.

Interfaces riches pour des architectures de contrôle de robots compositionnelles.

Revue des Techniques et Science Informatiques, TSI, 30(6) :713–741, juin 2011.