Formation AmbientComp

Stéphane Lavirotte, Jean-Yves Tigli, Gaëtan Rey, Thibaut Gonnin, Gérald Rocher

Equipe Rainbow, Laboratoire I3S, UMR CNRS 7271, Université de Nice Sophia Antipolis,

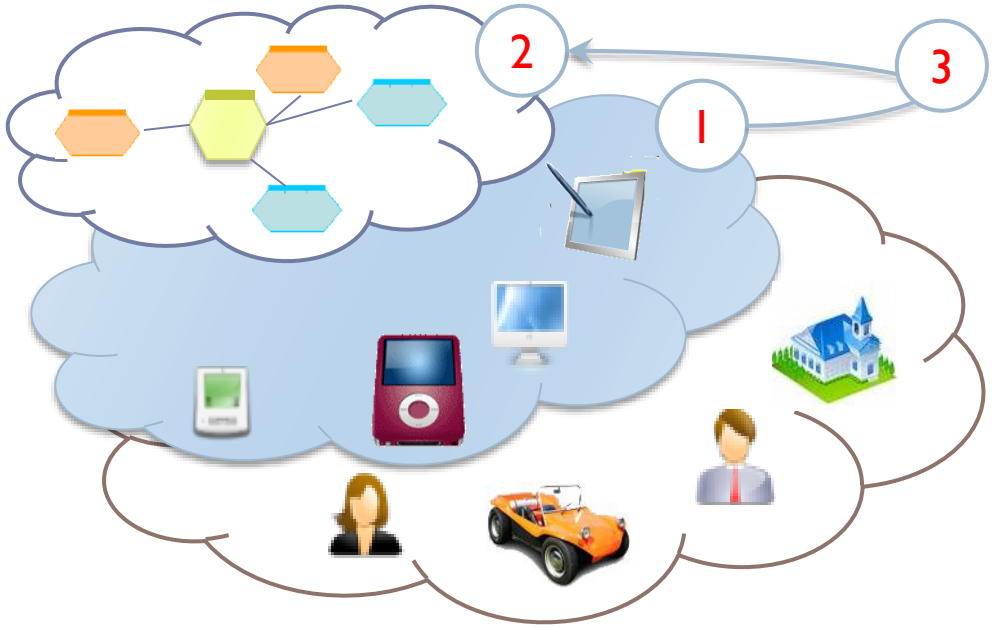
Email : prenom.nom@univ-cotedazur.fr

AmbientComp et Auto-Adaptation

Introduction aux AAs

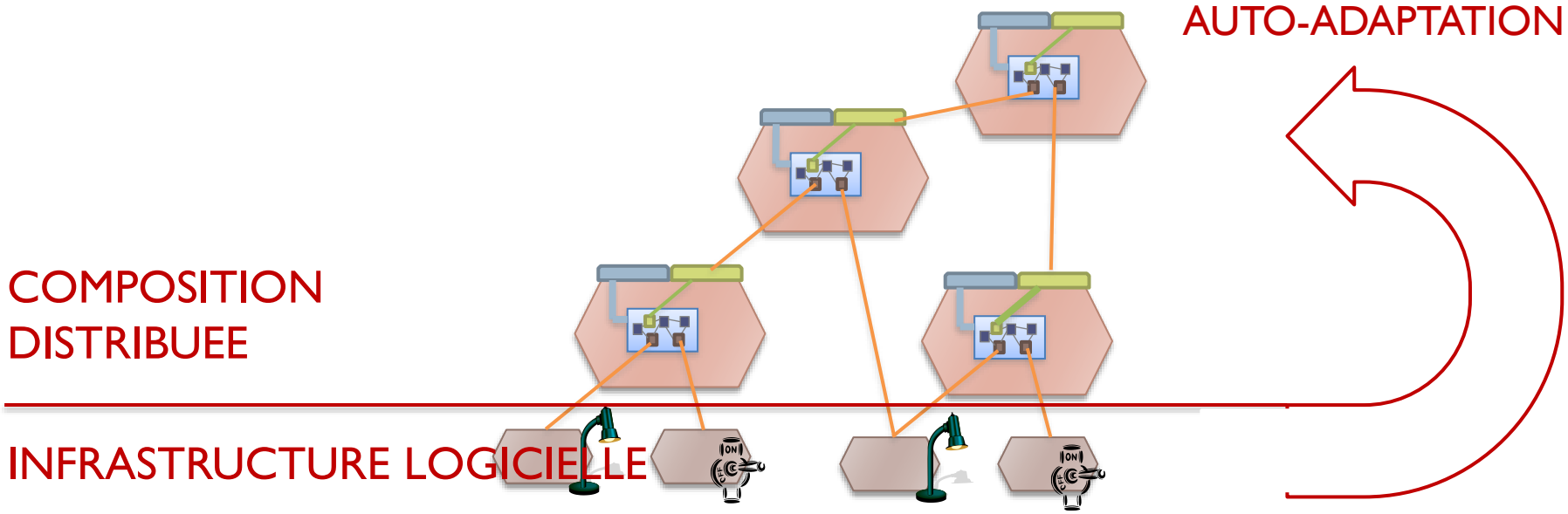
Rappels

- Une infrastructure logicielle de services pour dispositifs dynamique (basée sur UPnP dans AmbientComp.Net)
- Deux niveaux de composition dynamique (LCA et SLCA)
- Un mécanisme d'adaptation dynamique basé sur les AA

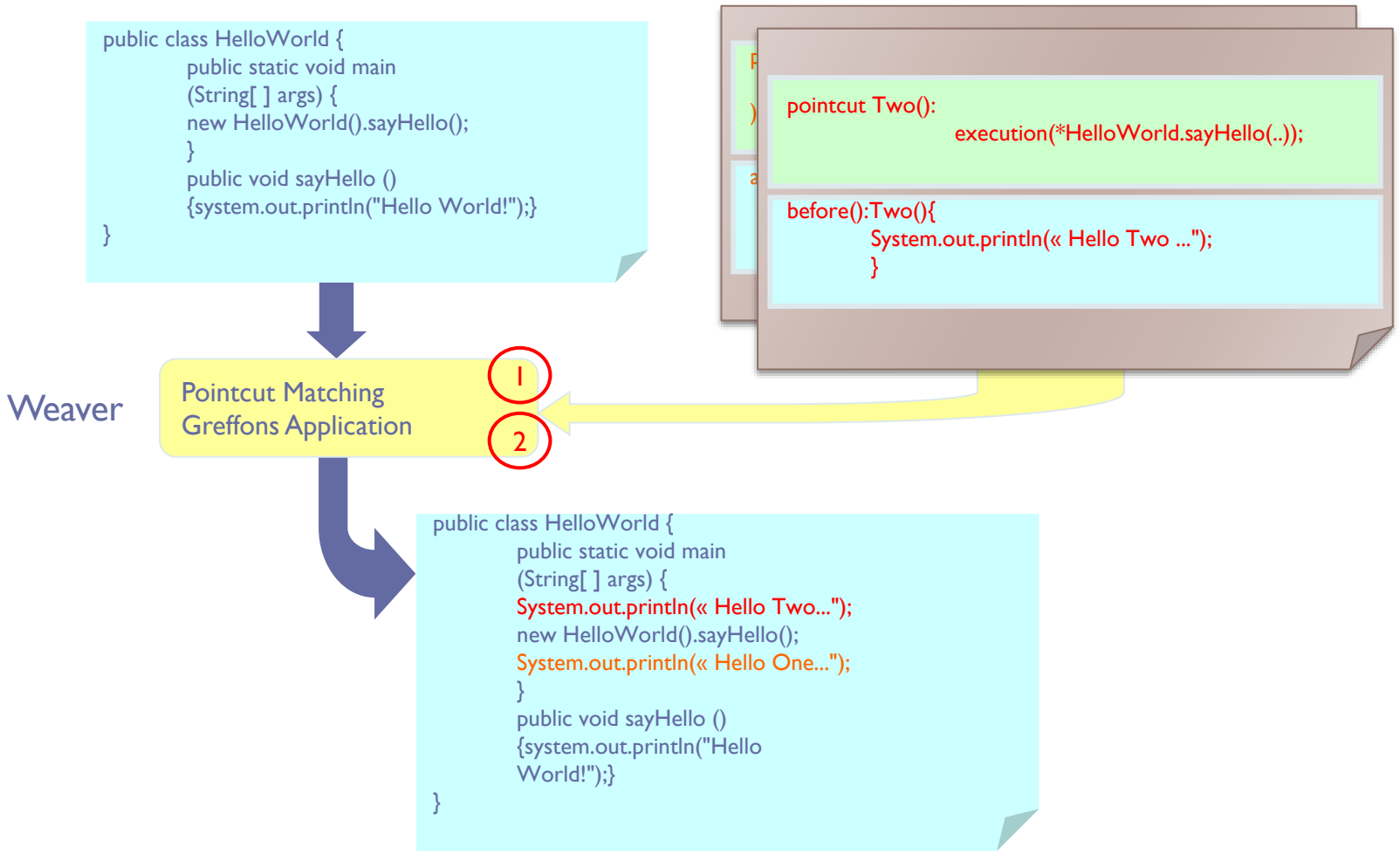


Introduction aux Aspects d'Assemblages (AA)

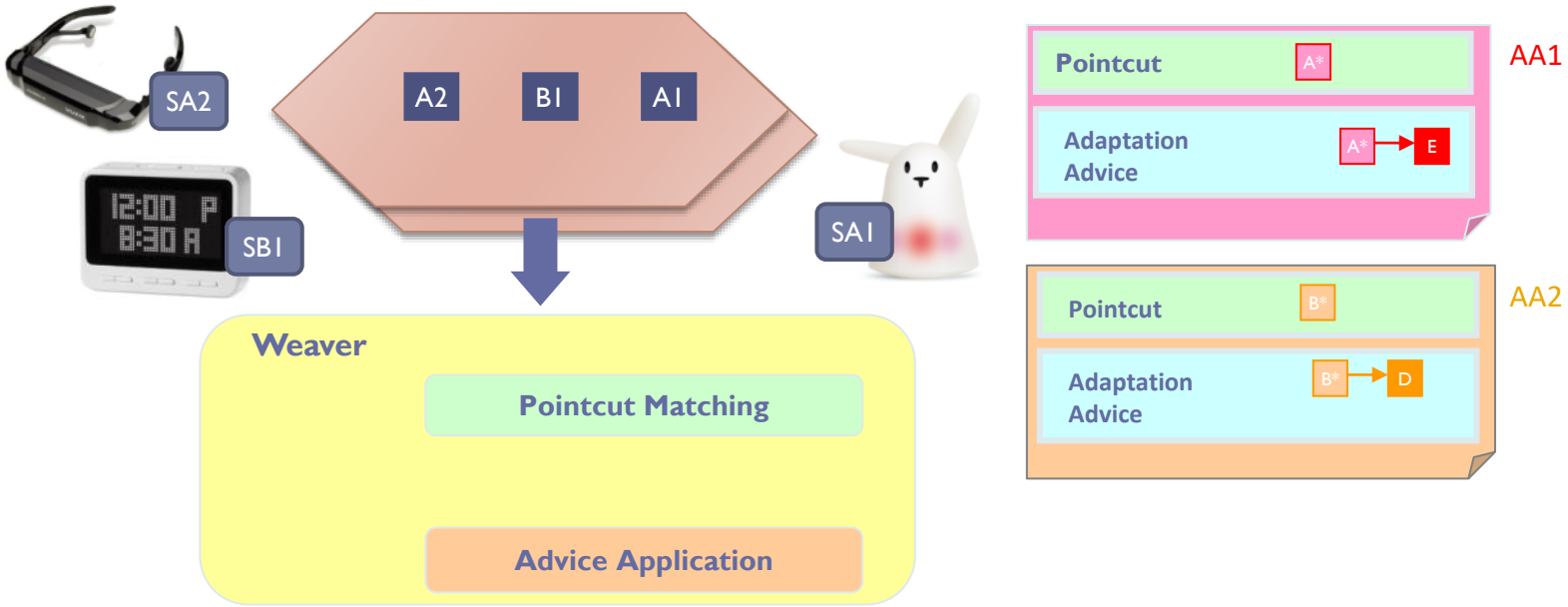
- Problématiques scientifiques :
  - Séparation des préoccupations
  - Maintien de la cohérence de l'application
  - Maîtrise du temps d'adaptation



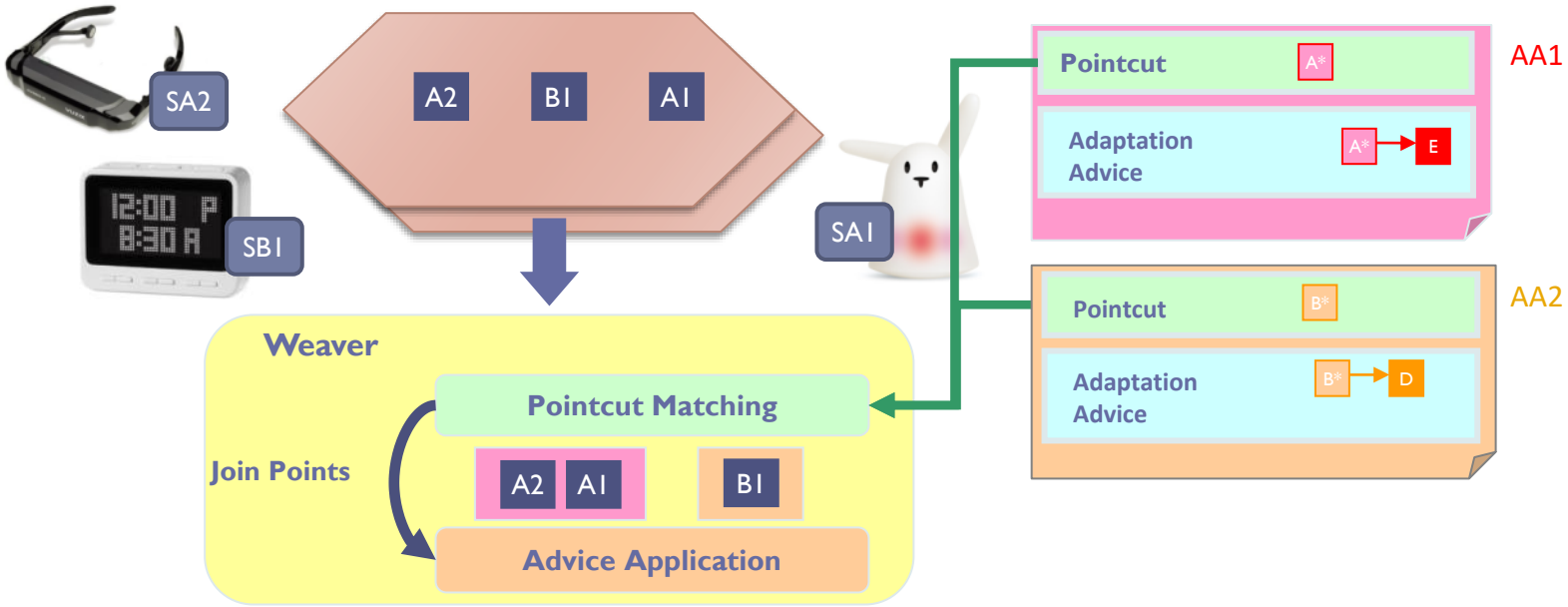
Principe des AA : de l'AOP …



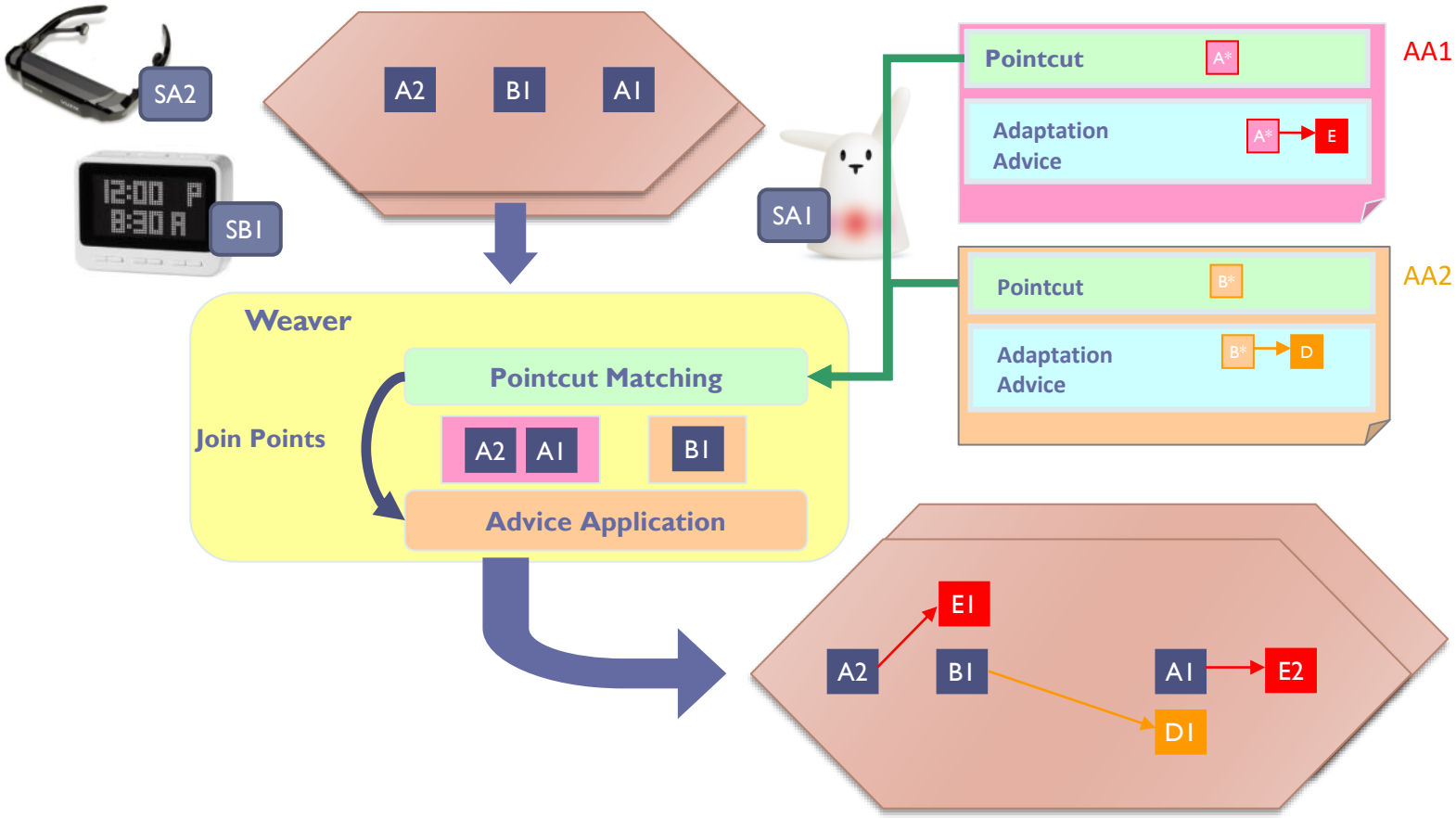
Le principe des AAs



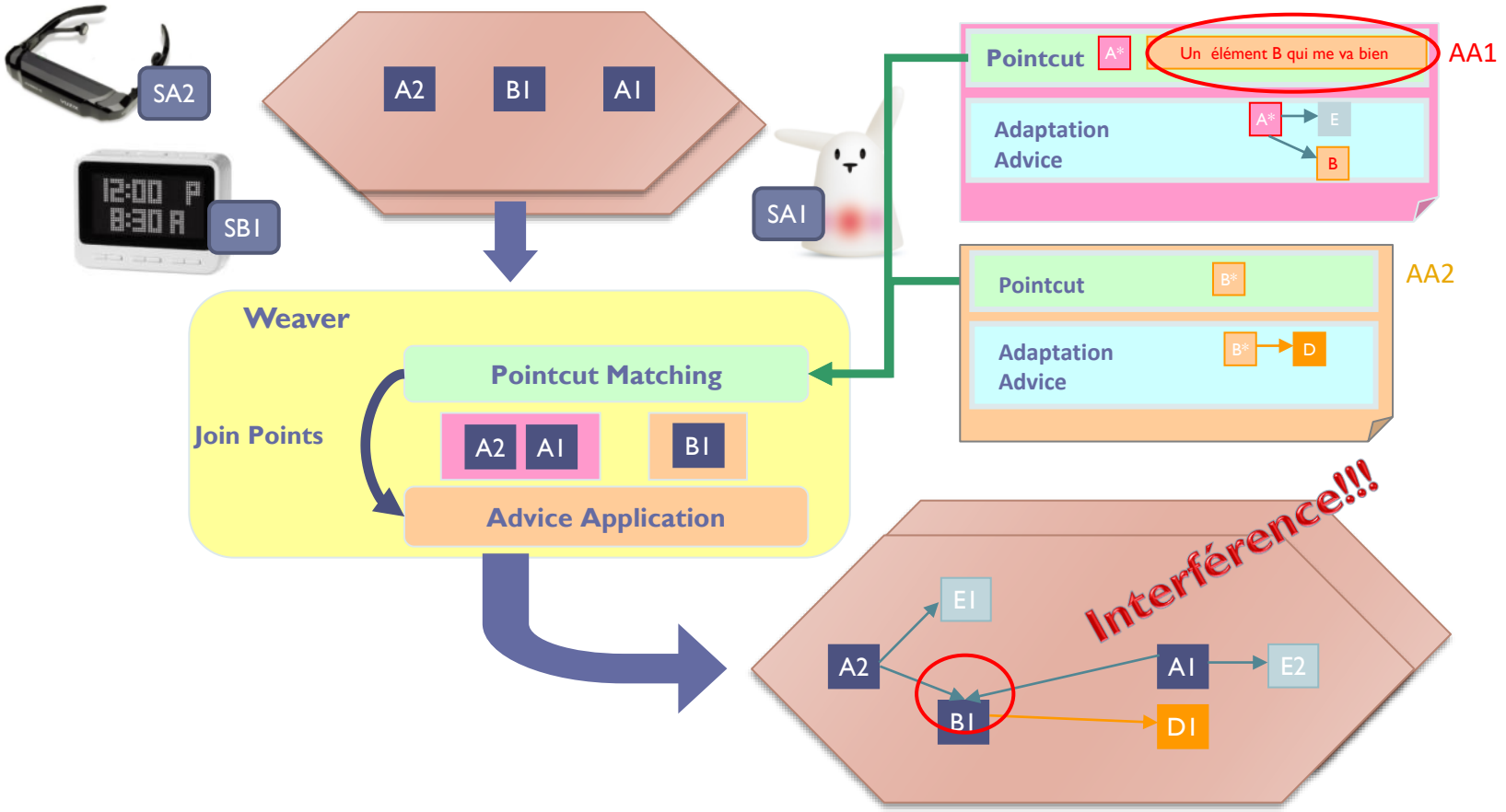
Le principe des AAs



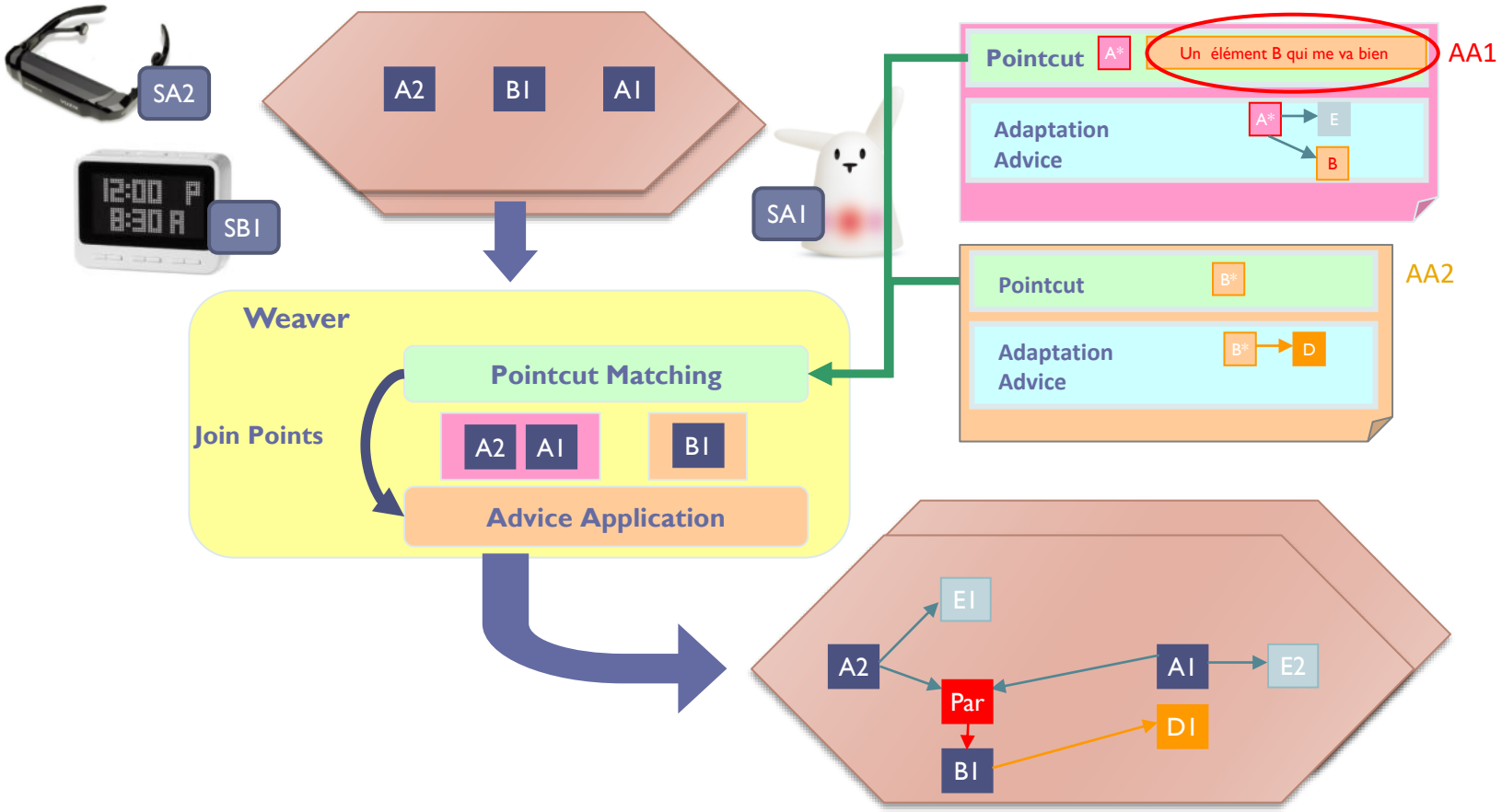
Le principe des AAs



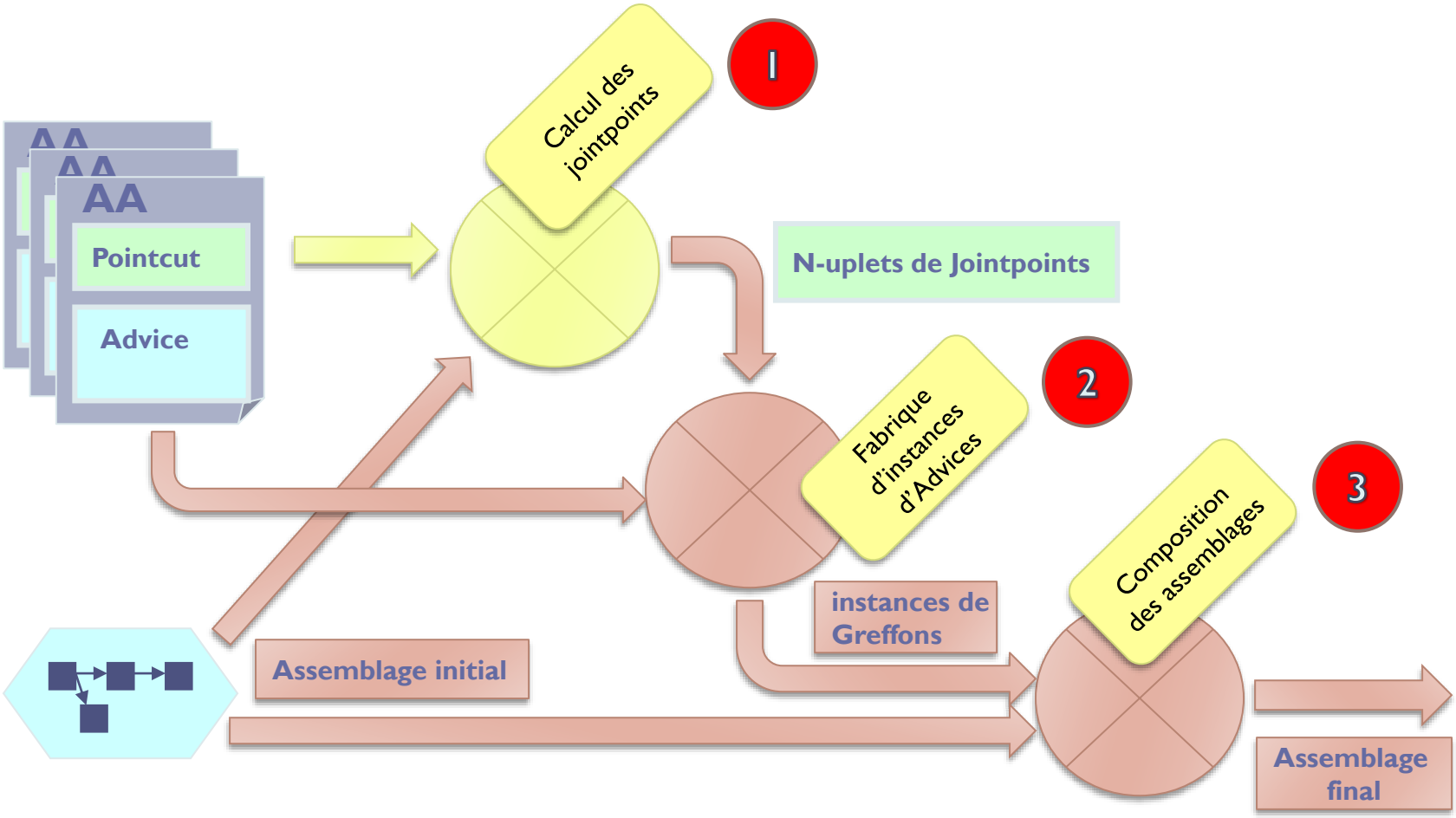
Et quand c'est plus compliqué ?



Résultat après fusion



Détails d'un cycle de tissage d'AA



Poincuts Syntaxiques

- Basés sur des expressions régulières

Pointcut

```
observed := /t*/ ;
timeout :=
    /ct*/  { a[substr($1,3)]=$1 }
    END    { for(i=1;i<=NR;i++){print a[i]} } ;
```

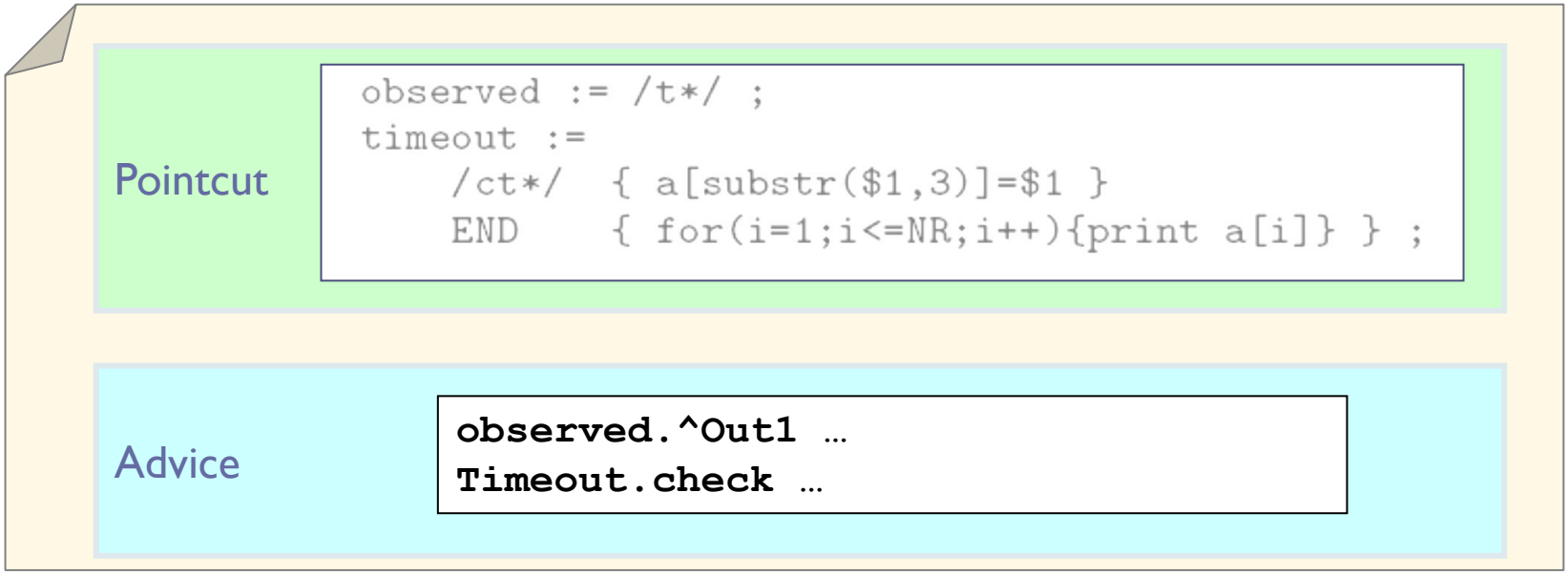
Advice

```
observed.^Out1 …
Timeout.check …
```

- Règles qui introspectent l'assemblage de l'application à la recherche de composants correspondants …

Advices : des langages … 2

- Basés ou pas, sur des opérateurs/composants de sémantique connue



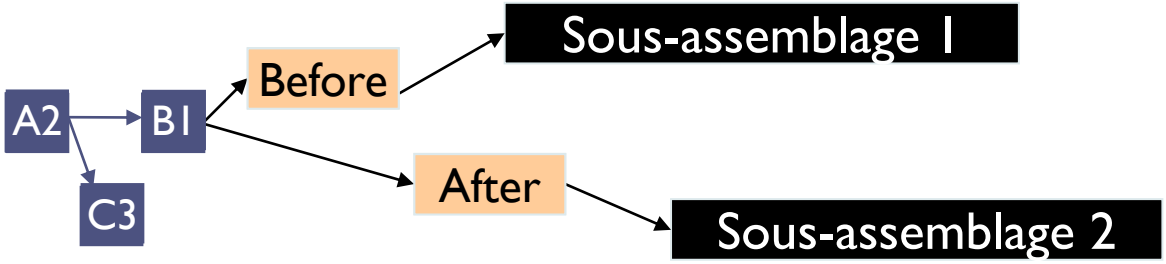
- Langages de descriptions de la fabrique de sous-assemblage
- Exemple : ISL4AmbientComp, BSLAmbientComp, ADLAmbientComp, VVAmbientComp …

Composition des assemblages sans fusion

3

- Composition / superposition de sous-assemblages

```
SCHEMA Ex (observed):
observed.^Out1 -> Bcomp.do;  CALL /* before
observed.^Out2 -> CALL ; Bcomp.do; /* after
```



Composition des assemblages avec fusion

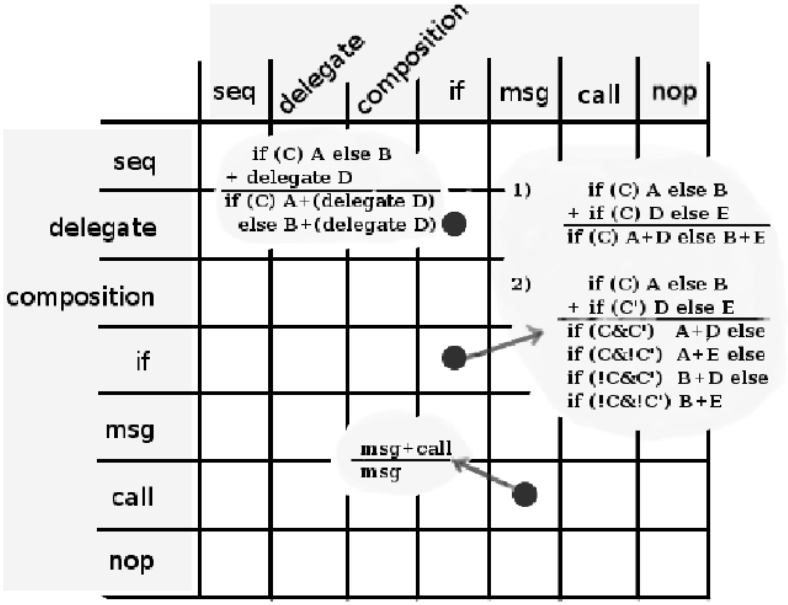
3

- Composition / fusion de sous-assemblages

```
SCHEMA Ex (observed, timeout):
    observed.^Out ->
            (  IF ( timeout.Check ) CALL )
    timeout.Check ->
            (  timeout.Start ; CALL      )
```

Logique de fusion & Propriété

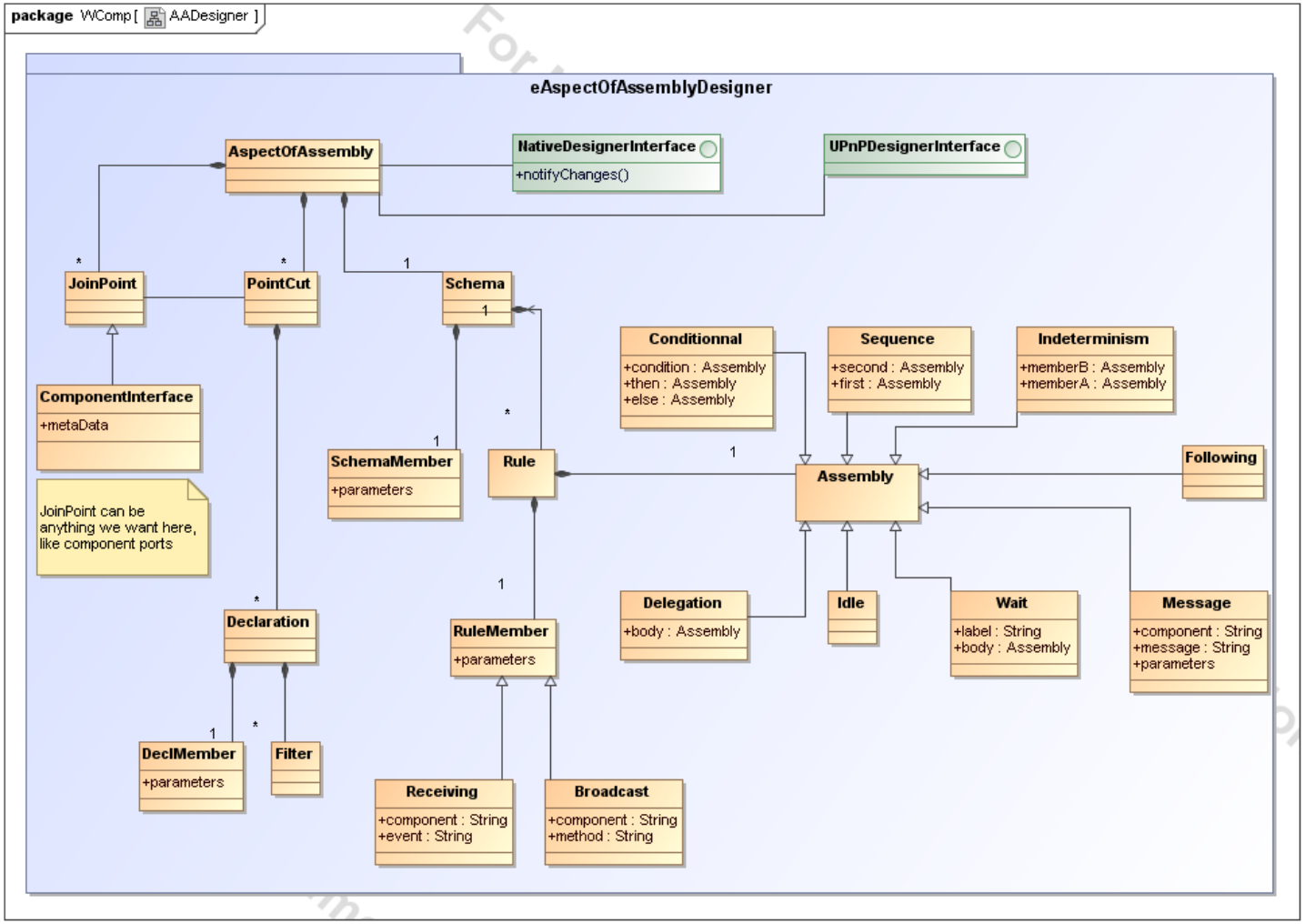
- Ex. Logique de Fusion de ISL4AmbientComp



- Avec propriété de symétrie

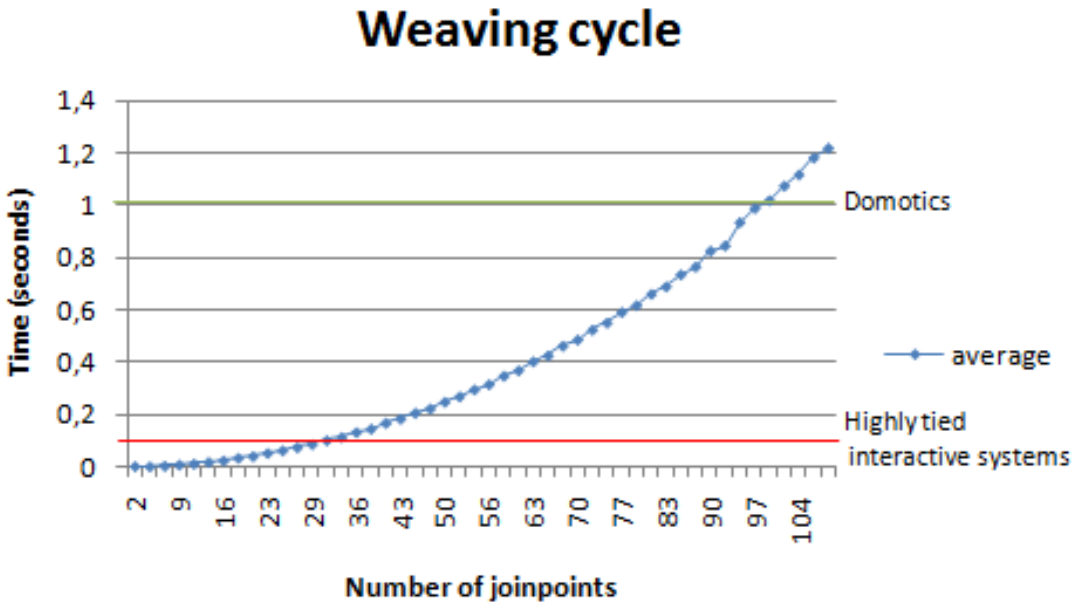
Commutativité : $AA0 \otimes AA1 = AA0 \otimes AA1$
Associativité : $(AA0 \otimes AA1) \otimes AA2 = AA0 \otimes (AA1 \otimes AA2)$
Idempotence : $AA0 \otimes AA0 = AA0$

Modèle de l'AA Designer



Maitrise du temps d'adaptation (Cycle de tissage d'AA)

- Méthode :
  - étude de la complexité des différentes étapes algorithmiques
  - modèle et identification de paramètres dans l'analyse des performances



Mise en œuvre des AA sur AmbientComp

- Installation
  - AA Designer (sélection des Aspects d'Assemblages)
- Réalisations:
  - Apparition et Disparition de Services pour Dispositifs
  - Ecriture d'Aspects d'Assemblages
  - Adaptation dynamique de l'assemblage en fonction des services pour dispotififs disponibles

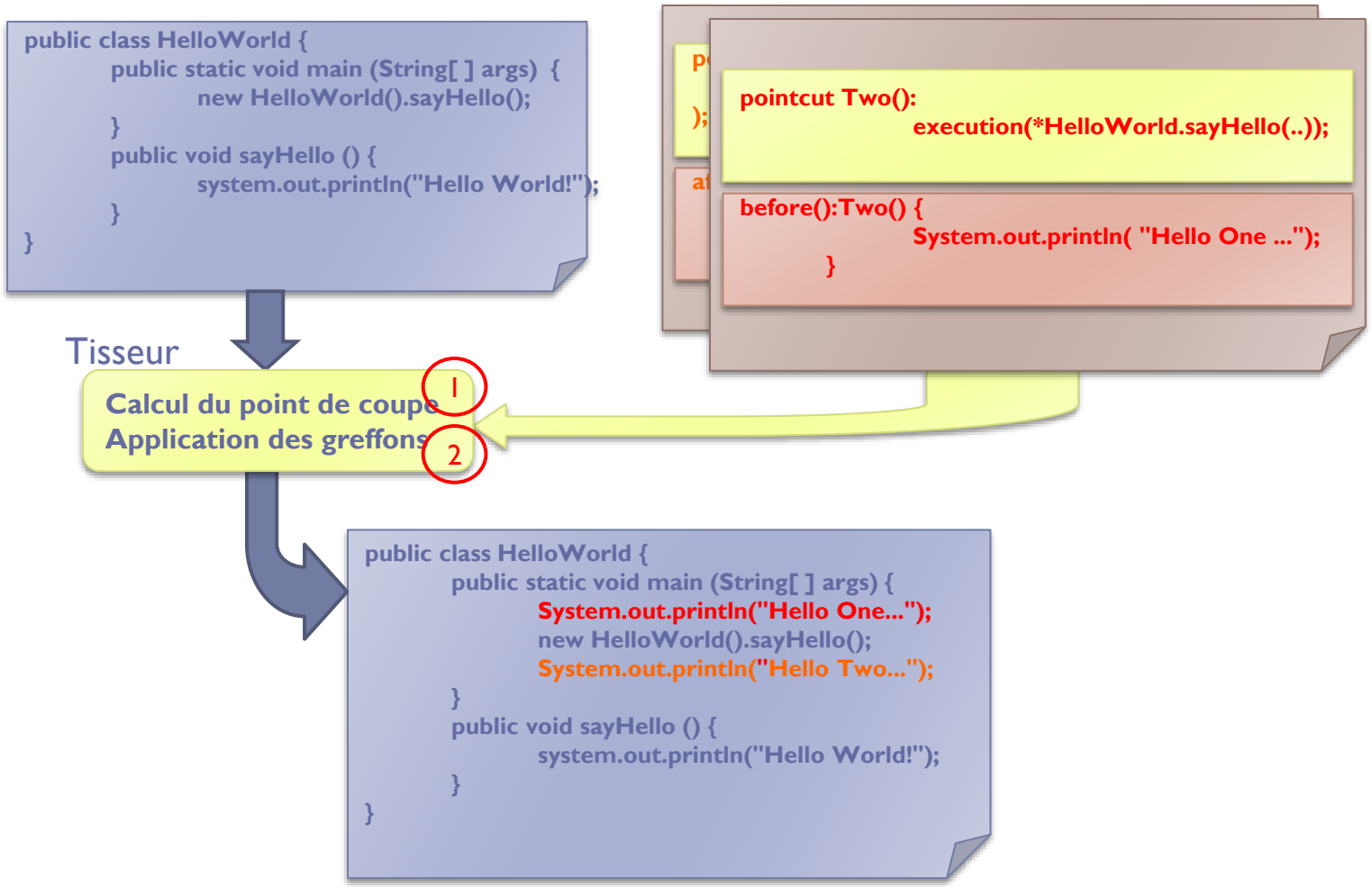
Bilan des apports

- Infrastructure logicielle de services pour dispositifs
  - Gère de l'Hétérogénéité et Découverte Dynamique de dispositifs
  - Permet l'interaction événementielle avec les Dispositifs
  - Gère l'apparition/disparition dynamique des Dispositifs dans l'infrastructure

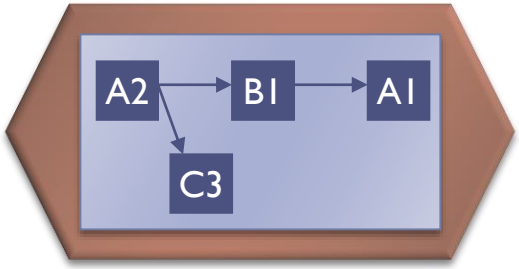
- Composition de services pour dispositifs
  - Composition modifiable dynamiquement
  - Local / Distribué
    - Des compositions de services pour dispositifs locaux
    - Des compositions de services composites pour application globale

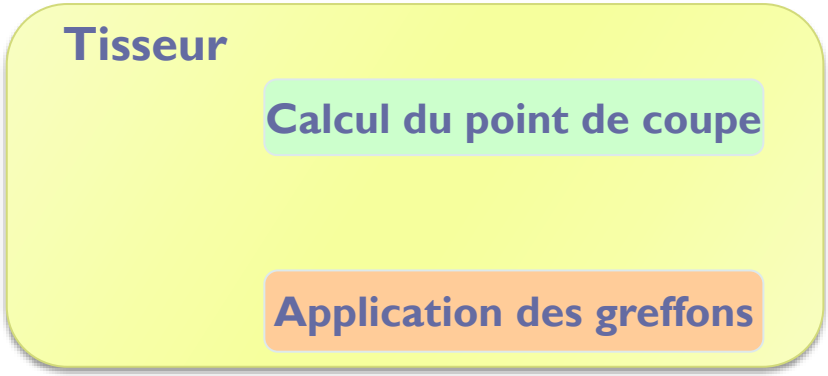
Les AAs en détails

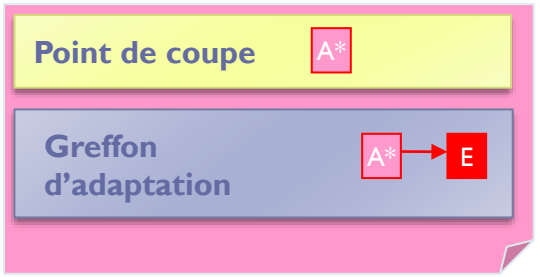
Principes de l'AOP

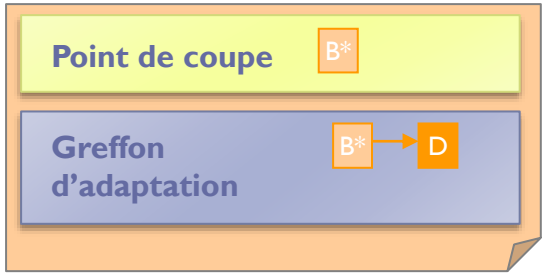


Aspect d'Assemblage (AA) Principes

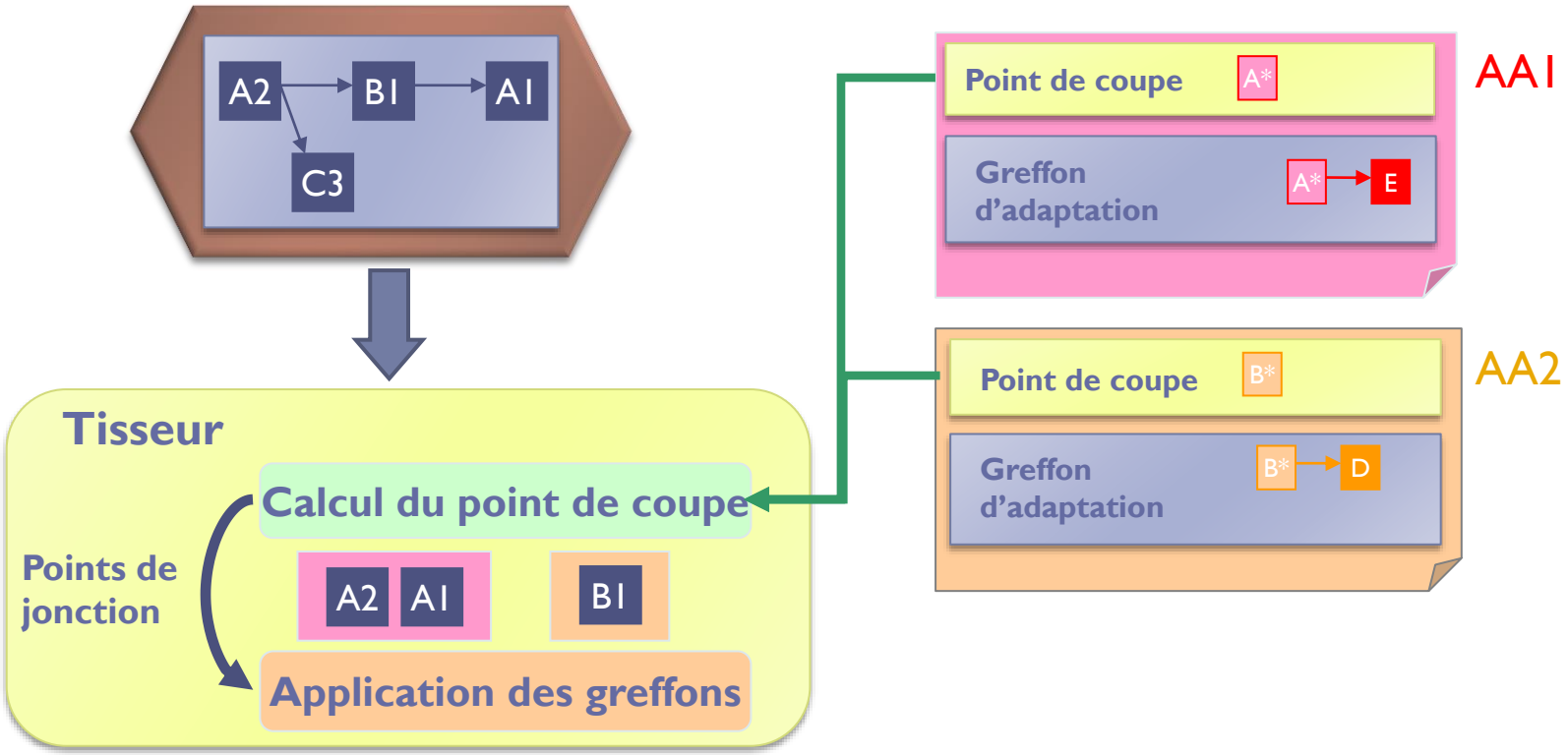




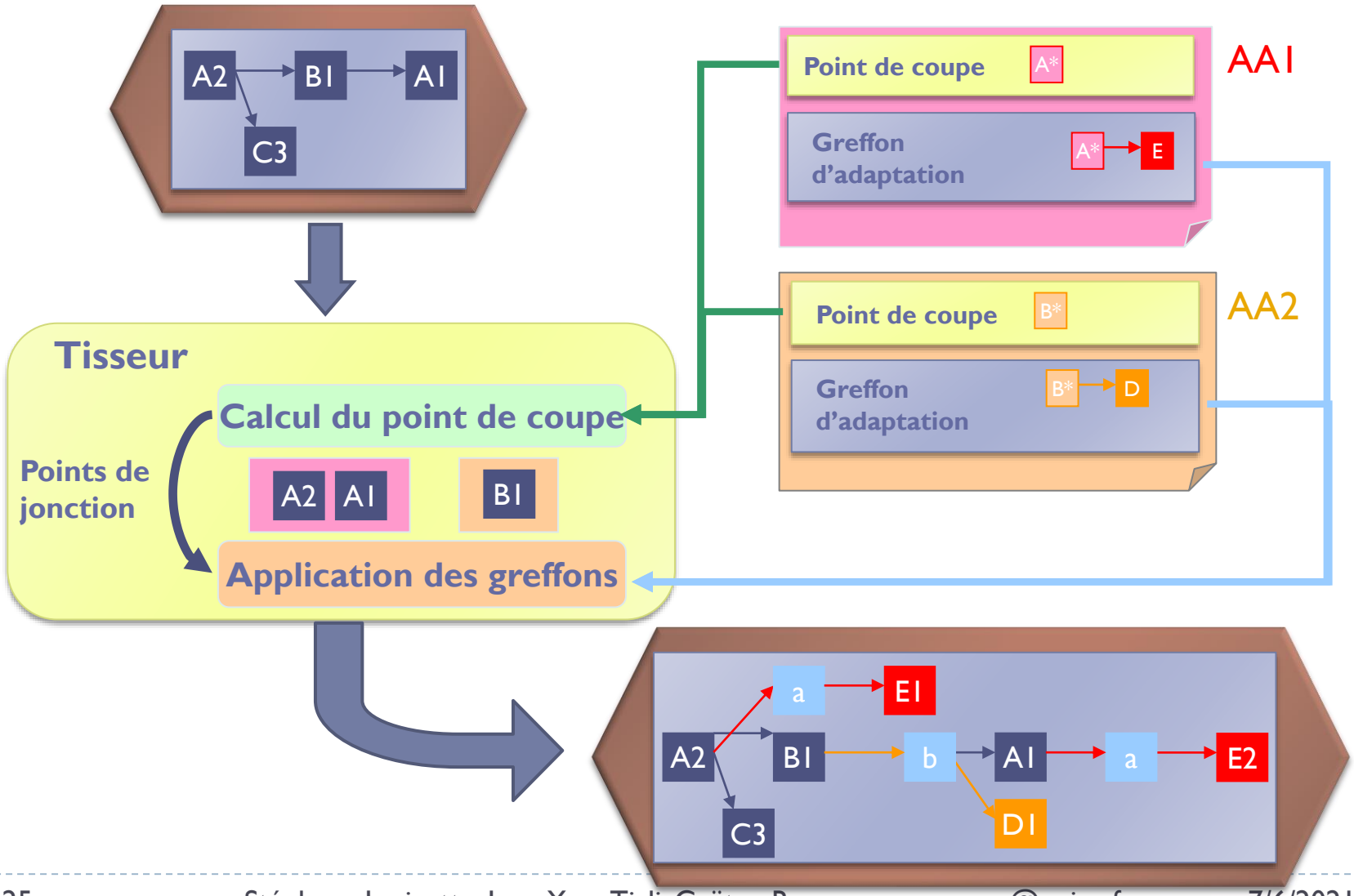




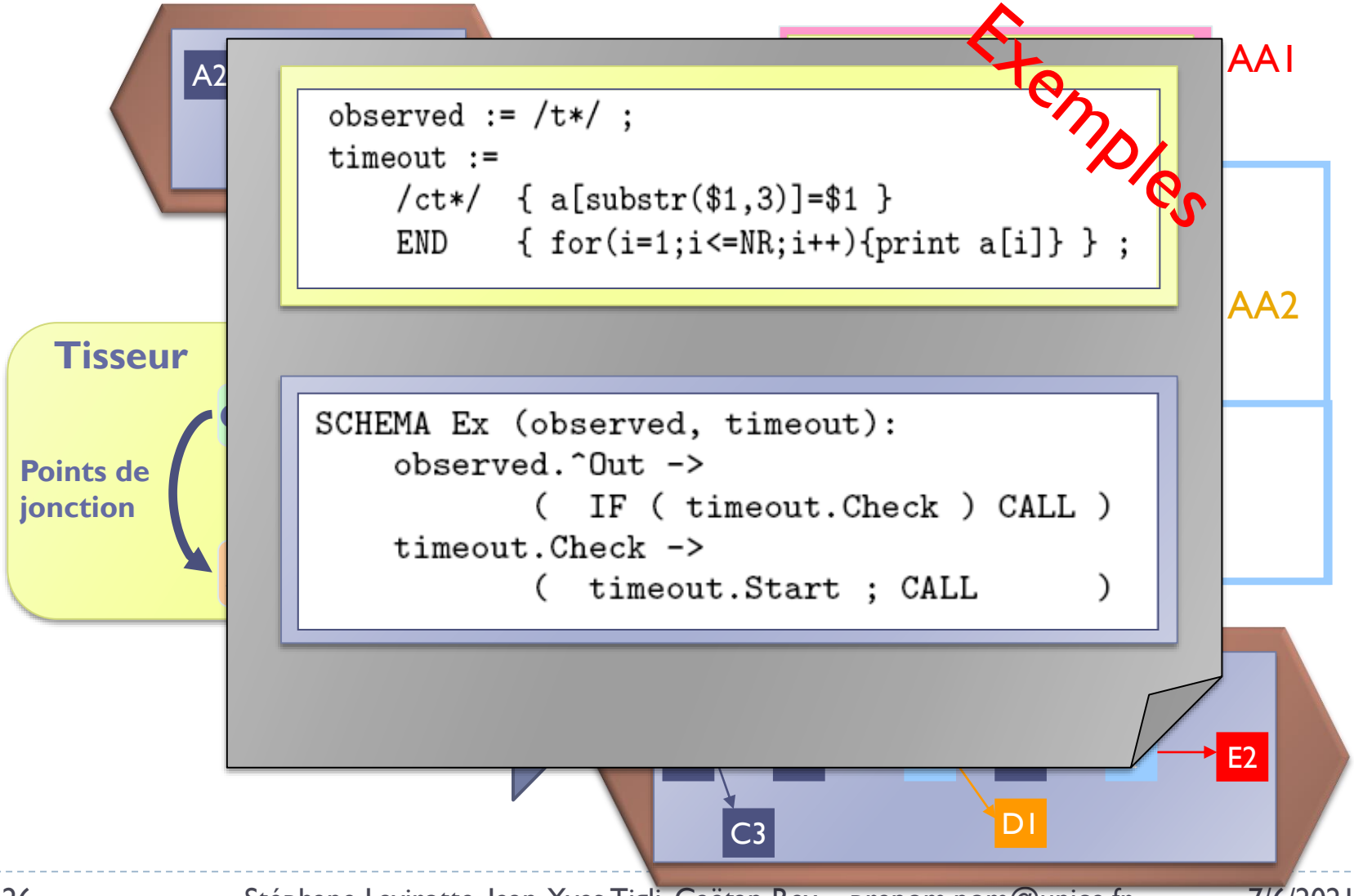
Aspect d'Assemblage (AA) Principes



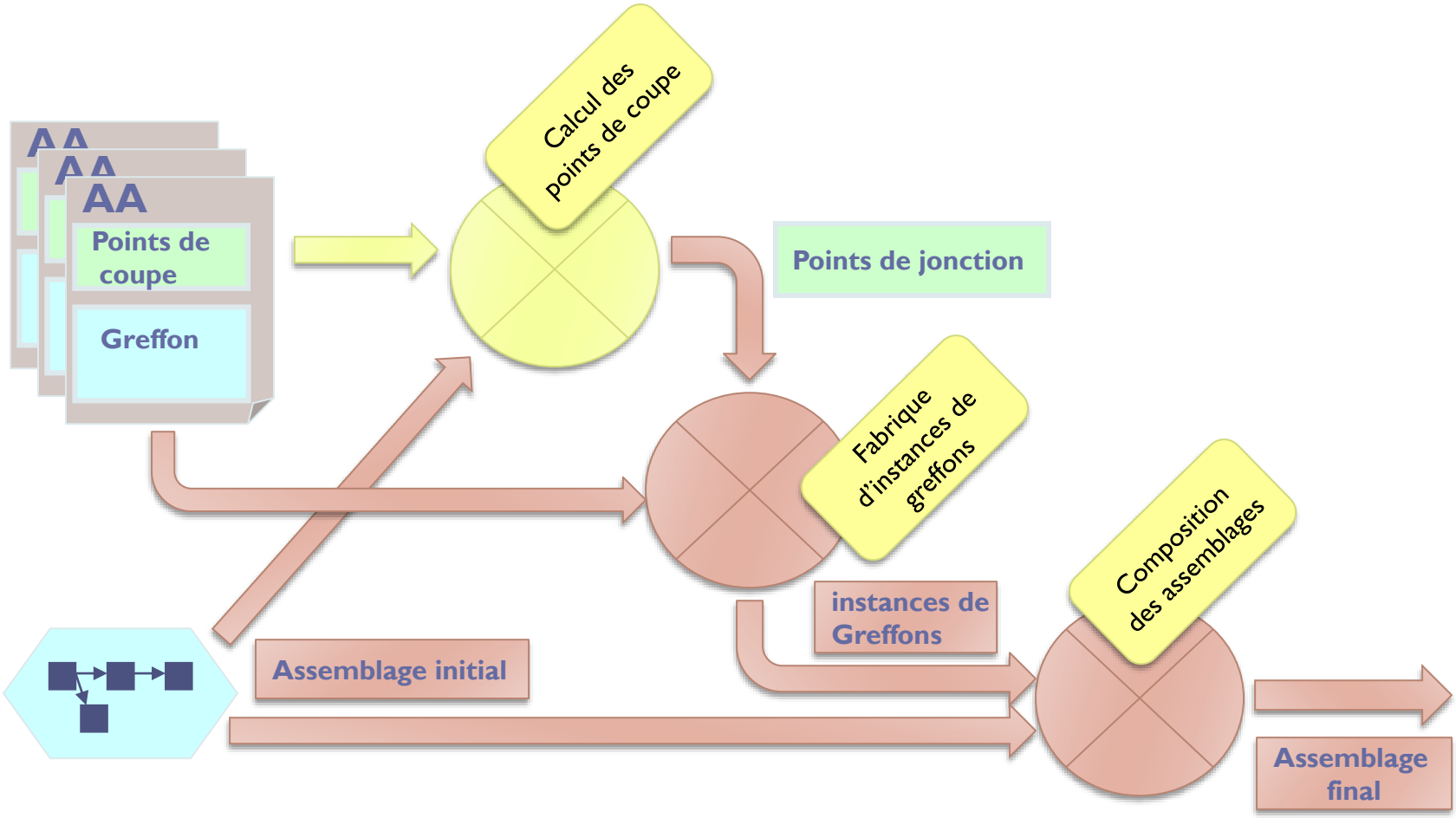
Aspect d'Assemblage (AA)
Principes



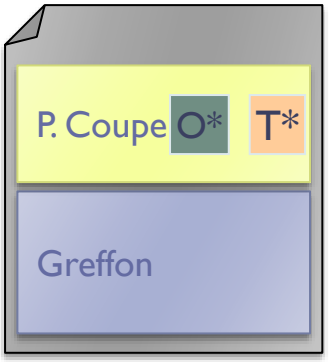
Aspect d'Assemblage (AA) Principes



Architecture fonctionnelle du Tisseur d'AA

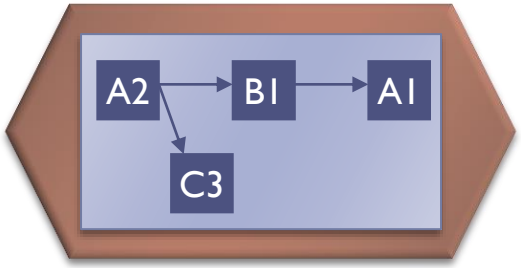


AA: Langage et Calcul du point de coupe




```
observed := /t*/ ;
timeout :=
    /ct*/  { a[substr($1,3)]=$1 }
    END    { for(i=1;i<=NR;i++){print a[i]} } ;
```

ct3, ct2, ct1, …



variable

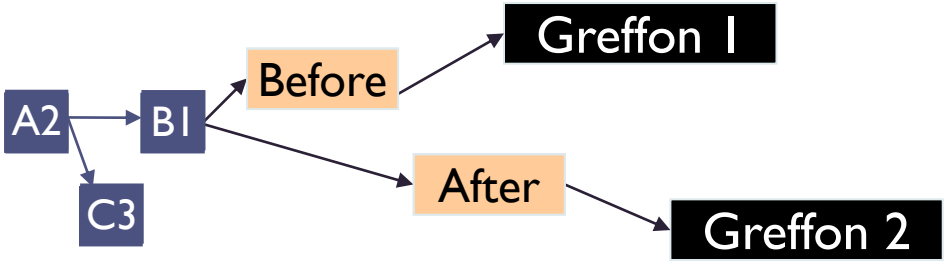
Après avoir parcouru la liste des composants de l'assemblage

Pattern matching + Action

Perspective : d'autres langages de matching

AA: Fabrique de greffon et gestion des conflits

- A l'instar des approches classique AOP :
  - Composition externe entre les greffons



- L'approche AA intègre une logique de fusion (inspirée des travaux antérieurs de Rainbow [30]) :
  - On connait partiellement la sémantique du greffon
  - Greffons sont fusionnés
  - Logique de Fusion dotée de propriétés

AA : Une logique de Fusion et ses propriétés

- Fusion à partir de règles logiques et modifié selon les langages de greffons
- Exemple de Propriétés de la composition / fusion inspiré d'ISL [30]

Commutativité : $AA0 \otimes AA1 = AA0 \otimes AA1$
Associativité : $(AA0 \otimes AA1) \otimes AA2 = AA0 \otimes (AA1 \otimes AA2)$

- Auquel on rajoute

Idempotence : $AA0 \otimes AA0 = AA0$

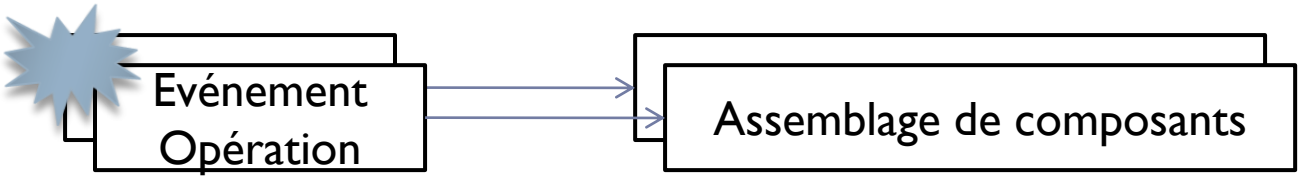
- Permet au concepteur de ne pas se soucier de l'ordre d'application des Aspects d'Assemblage

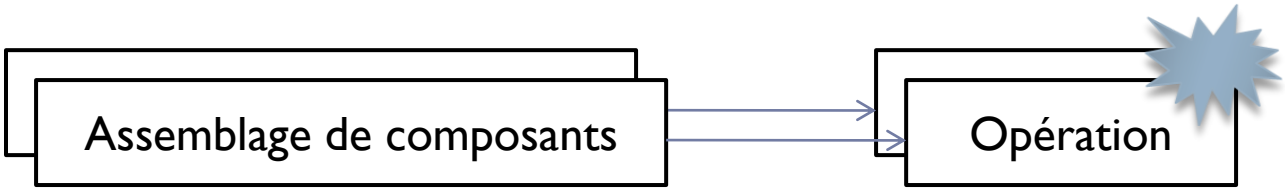
Perspective:
Autres propriétés
pour la fusion

AA : Deux langages de greffons

- ISL4AmbientComp pour gérer la concurrence au niveau de la diffusion d'événements et réécriture d'opération
- Principe syntaxique :

- BSL pour gérer la concurrence au niveau des invocations d'opérations
-



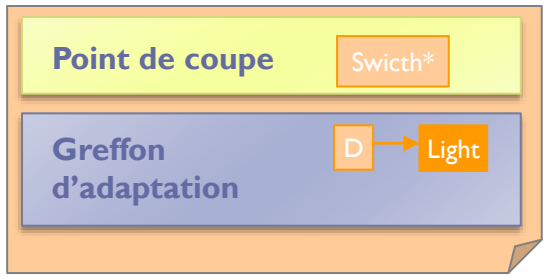


AA: le premier langage pour greffon ISL4AmbientComp

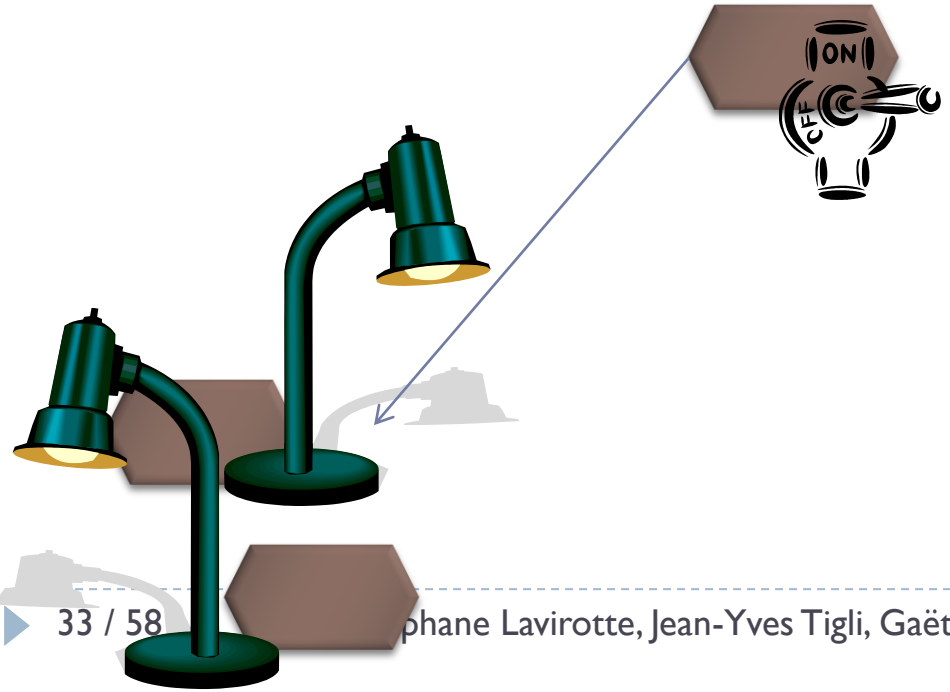
- Spécificités
  - Opérations / Evénements au membre gauche
  - Instance de greffons sous forme d'assemblage de composants correspondant
    - Aux opérateurs de base du langage
    - Pas de code généré à la volée dans un unique composant
  - Différentes sémantiques aux opérateurs
    - delegate (absorbant)
    - But: supprimer les non résolutions dans la fusion.
  - Propriété supplémentaire pour la fusion :
    - Idempotence
    - But : ne pas se soucier de l'application multiple d'un même AA

Exemple de fusion ISL4AmbientComp

D = Switch*
D.^on → Light.on



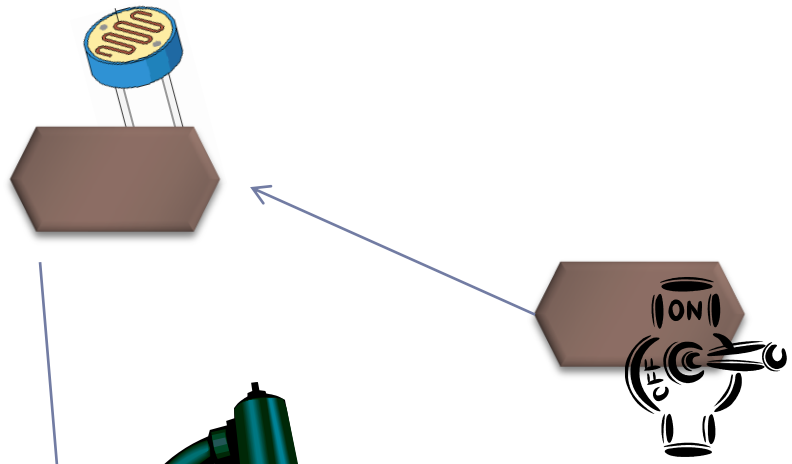
AAI

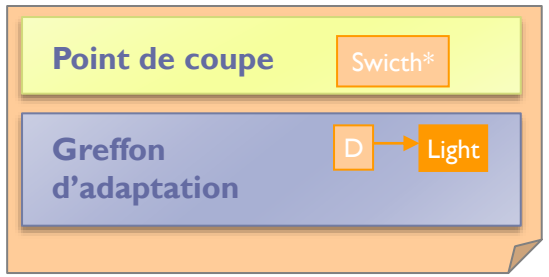


phane Lavirotte, Jean-Yves Tigli, Gaëtan Rey – prenom.nom@unice.fr

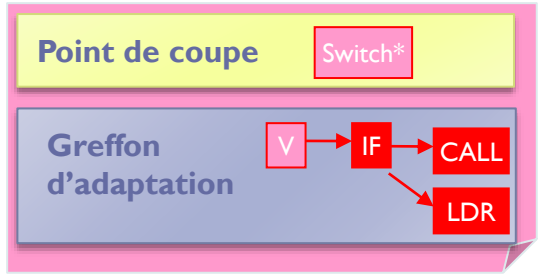
Exemple de fusion ISL4AmbientComp

D = Switch*
D.^on → Light.on





AA1



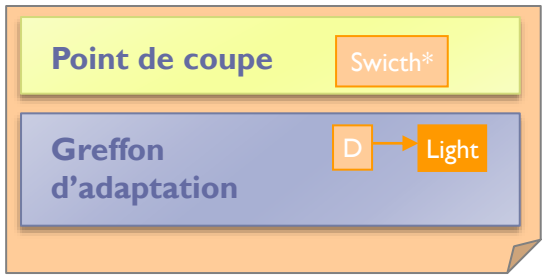
AA2

V = Switch*
V.^on → if(LDR.Bright) nop else call

phane Lavirotte, Jean-Yves Tigli, Gaëtan Rey – prenom.nom@unice.fr

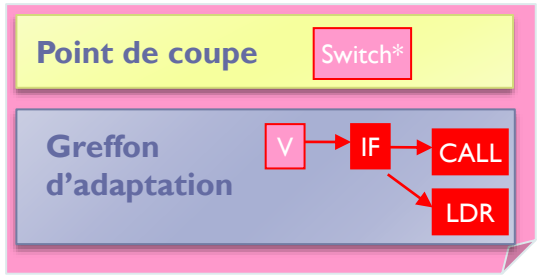
Résultat de la fusion ISL4AmbientComp

D = Switch*
D.^on → Light.on



AA1

V = Switch*
V.^on → if(LDR.Bright) nop else call



AA2

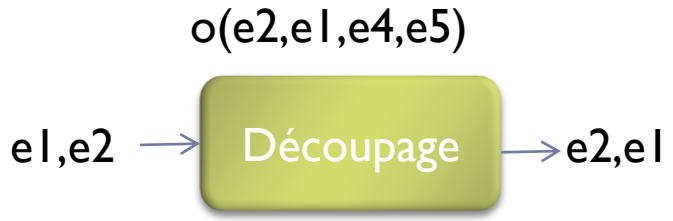
Switch.^on → Light.on + if(LDR.Bright) nop else call
→ if(LDR.Bright) Light.On + nop else Light.On + call [Classement]
→ if(LDR.Bright) nop else Light.On [Règles finales]

AA: le second langage pour greffon BSL

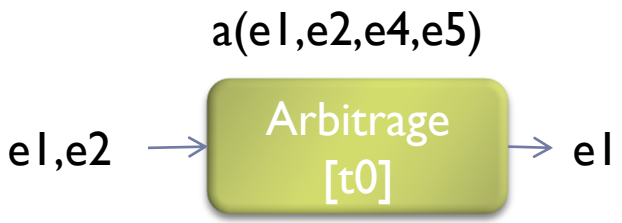
Behavioral Specif. Lang.

- Composants spécifiques
  - Arbitrage, Découpage
- Même propriété pour la Logique de Fusion
  - Associativité, Commutativité, Idempotence

Opérateurs

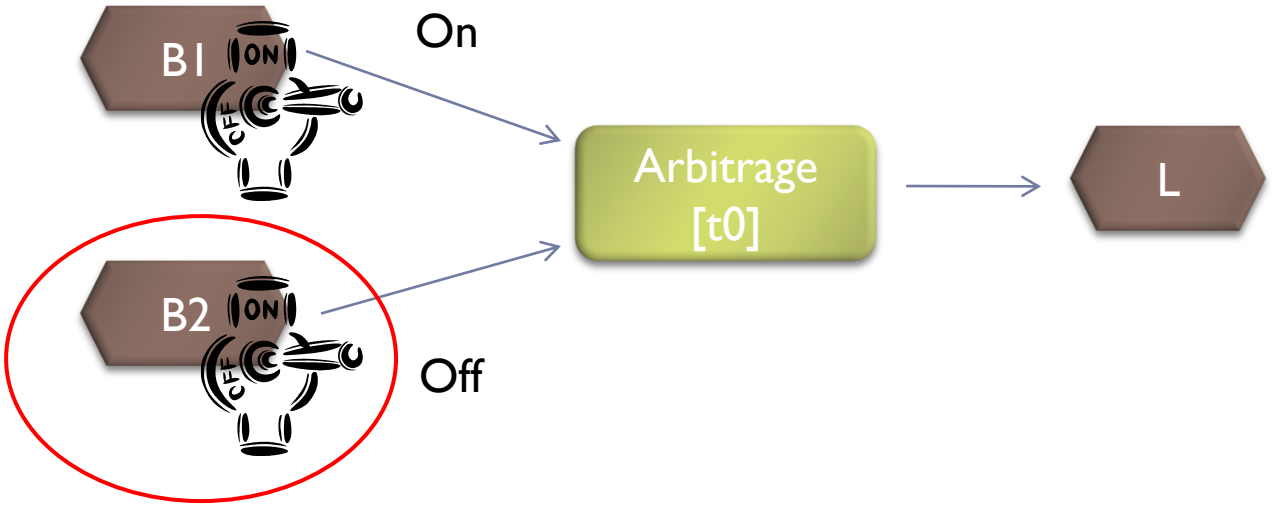


Réordonner l'arrivée d'événements

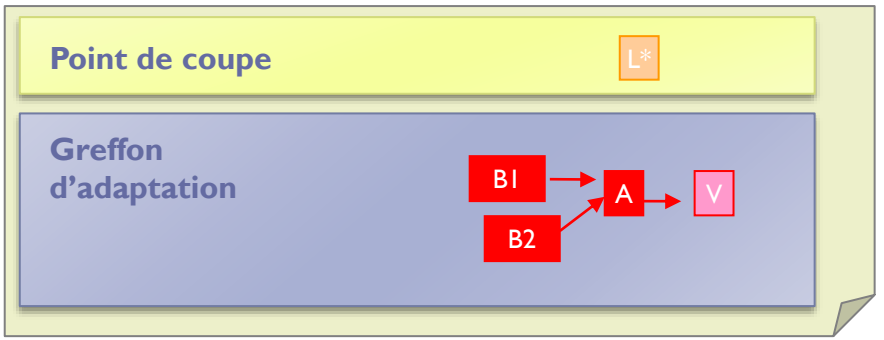


Liste d'événements prioritaires pendant une durée donnée

Exemple de composition BSL



B2 prioritaire pendant un durée donnée



AA

V := /L*/

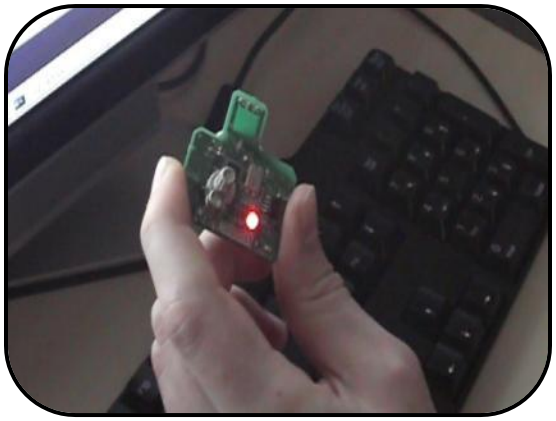
a(B2.^Open, B1.^Close)→ V.Action

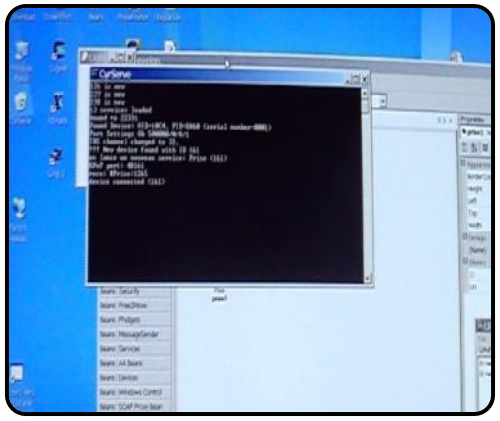
Mise en œuvre ISL4AmbientComp et BSL

- Règles logiques pour la logique de Fusion
  - 20 prédicats pour ISL4AmbientComp
  - 14 prédicats logiques pour BSL
- Programmées en Prolog
  moteur CxProlog léger (Dias)
- Intégration de l'interpréteur AWK du projet Busybox pour les points de coupe

Expérimentation

- Adaptation par post-sélection d'AA (dirigée par l'utilisateur)
- Adaptation par présélection d'AA (réactif aux variations de l'infrastructure)
- Illustration de la fusion



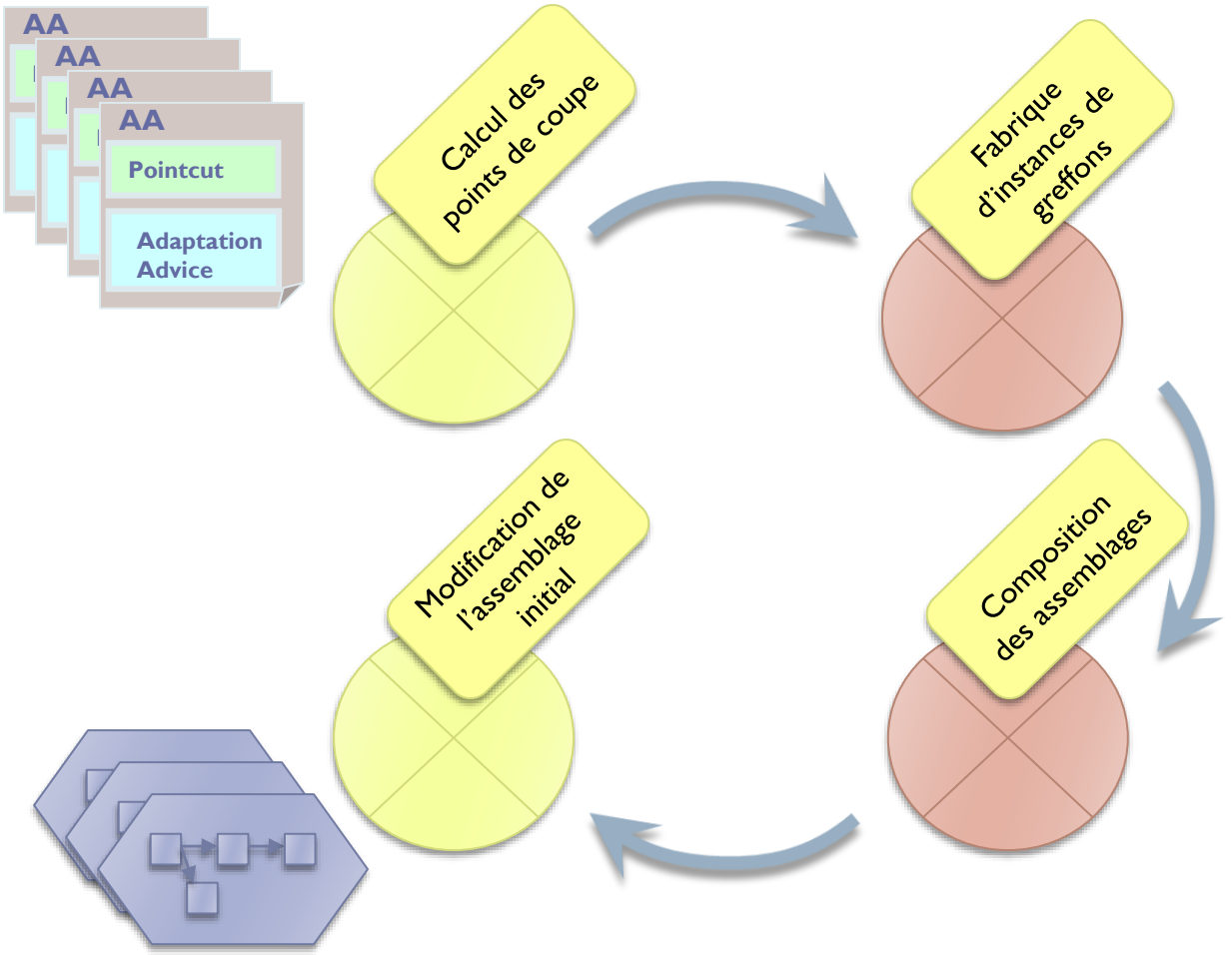


Evaluation des performances dans la mise en œuvre des AA

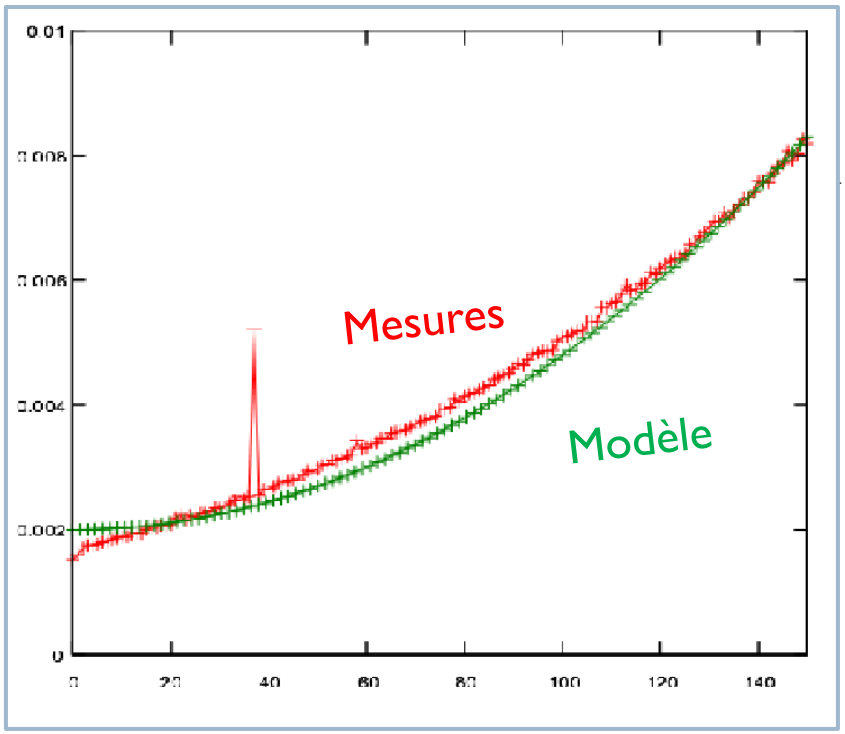
- Variables expérimentales
  - Paramétrage du nombre d'AA appliqués :
    - performance du calcul des points de coupe
  - Paramétrage du nombre de composants :
    - performance de la composition

- Conditions expérimentales
  - PC de bureau 1GHz
  - Point de coupe sans application multiple d'AA
  - Greffon contenant une connexion

Illustration d'un cycle d'évolution de services composites



Evaluation du coût du calcul des points de coupe



$1)c^2 + a_2$

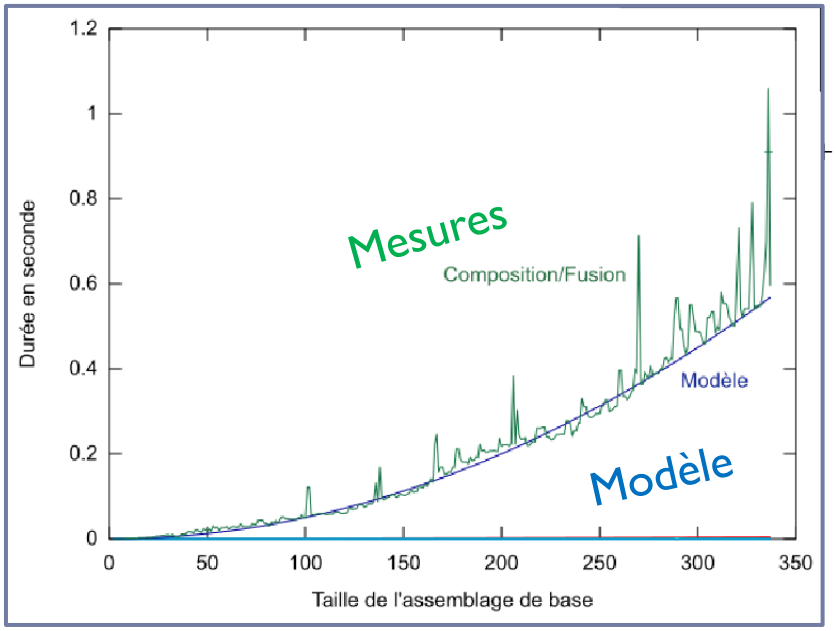
- Identification des paramètres
  - δ_i: nb d'application du greffon *i*
  - *c* : nb de composants

$$a_1 = 280.10^{-9}$$

$$a_2 = 2.10^{-3}$$

- **Dépend de la taille de l'assemblage de base et du nombre d'AA**

Evaluation du coût de la composition / modification



$- p_i.C(g_0, g_i))$

- Identification des paramètres

p_i : proba

C : cout de la fusion

n : nombre de règles

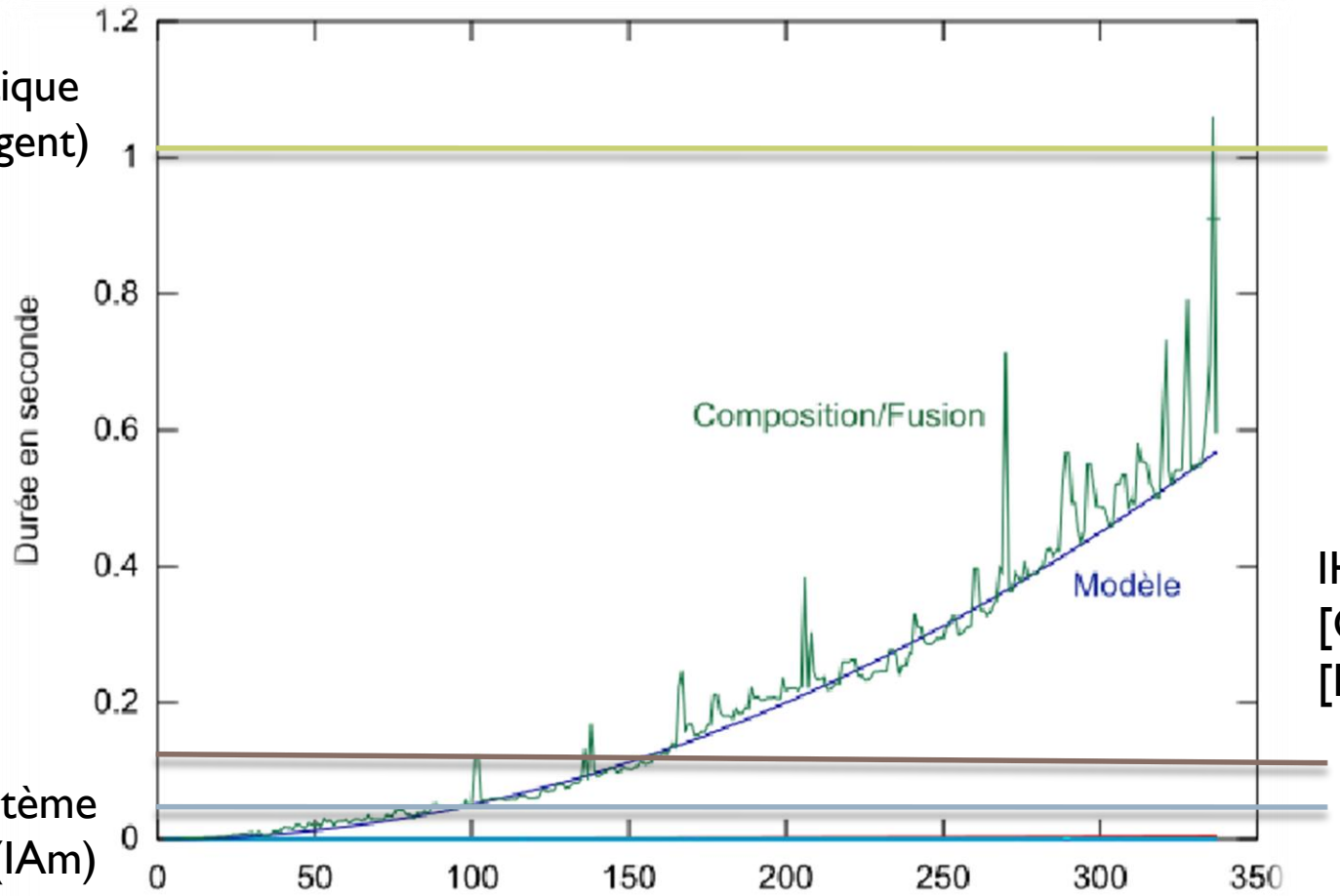
$$b = 2{,}6.10^{-6}$$

- **Dépend seulement de la taille de l'assemblage**

Validation

Gestion énergétique
(Bâtiment Intelligent)

Adaptation par AA



Système
robotisé (IAm)

IHM
[Card]
[Berard]