# Vérification de Modèles Kmelia

**Utilisation de Kml2B**

Arnaud Lanoix

13 mars 2009

```
COMPONENT
/* an example of component */
          A
INTERFACE
        provides : S_A1
        requires : S_A2
VARIABLES
        var_A
INVARIANT
        inv_A
INITIALIZATION
        init_A
SERVICES
        provided S_A1
                 Pre pre_A1
                 Behavior eLTS_A1
                 Post post_A1
        end ;
        required S_A2
                Context var_A2
                Invariant inv_A2
                Initialization init_A2
                Pre pre_A2
                Post post_A2
        end ;
END_SERVICES
```

FIGURE 1: Composant Kmelia A

## 1 Réécriture en B

Nous supposerons l'existence d'une fonction $\beta$(element_kmelia) qui réécrit en B le concept exprimé en Kmelia.

## 2 Cohérence d'un composant Kmelia

### 2.1 Obligations de preuve B

Les obligations de preuve B suivantes permettraient de s'assurer de la cohérence d'un composant Kmelia. ?

– L'initialisation du service respecte l'invariant du composant A

$$[\ \beta(\text{init\_A})\ ]\ \beta(\text{inv\_A})$$

– pour chaque service fourni SA_1,
  – la post respecte l'invariant du composant A

$$\beta(\text{inv\_A}) \wedge \beta(\text{pre\_A1}) \Rightarrow\ [\ \beta(\text{post\_A1})\ ]\ \beta(\text{inv\_A})$$

  – le eLTS etablit la post

$$\beta(\text{inv\_A}) \wedge \beta(\text{pre\_A1}) \Rightarrow\ [\ \beta(\text{post\_A1})\ ]\ \neg\ [\ \beta(\text{post\_A1})\ ]\ \neg\beta(\text{inv\_A})$$

  De tête, à verifier
– pour chaque service requis SA_2,
  – L'initialisation du contexte de SA_2 respecte l'invariant de SA_2

$$[\ \beta(\text{init\_A2})\ ]\ \beta(\text{inv\_A2})$$

  – le service SA_2 respecte son invariant

$$\beta(\text{inv\_A2}) \wedge \beta(\text{pre\_A2}) \Rightarrow\ [\ \beta(\text{post\_A2})\ ]\ \beta(\text{inv\_A2})$$

### 2.2 Machines B à générer

Pour que ces obligations de preuve soient générées, cela nécessite la construction de
– une machine B abstraite reprenant l'invariant de A, l'initialisation + pour chaque service fourni SA_1, une opération B reprenant la pre et la post de SA_1 (voir Fig 2) ;
– un raffinement de la machine B précédente, avec pour chaque service fourni SA_1, une opération B reprenant la pre et le elts de SA_1 (voir Fig 3) ;
– pour chaque service requis SA_2, une machine B abstraite reprenant l'invariant de SA_2, l'initialisation + une méthode B reprenant la pre et la post de SA_2 (voir Fig 4).

```
MACHINE
        A
VARIABLES
        B(var_A)
INVARIANT
        B(var_A)
        ∧ B(inv_A)
INITIALISATION
        B(init_A)
OPERATIONS
        B(S_A1) =
        PRE
                B(pre_A1)
        THEN
                B(post_A1)
        END
END
```

FIGURE 2: Machine B A à générer

```
REFINEMENT
        A_ref
REFINES
        A
VARIABLES
        B(var_A)
INVARIANT
        B(var_A)
        ∧ B(inv_A)
INITIALISATION
        B(init_A)
OPERATIONS
        B(S_A1) =
        PRE
                B(pre_A1)
        THEN
                B(eLTS_A1)
        END
END
```

FIGURE 3: Raffinement B A_ref à générer

```
MACHINE
        S_A2
VARIABLES
        B(var_A2)
INVARIANT
        B(var_A2)
        ∧ B(inv_A2)
INITIALISATION
        B(init_A2)
OPERATIONS
        B(S_A2) =
        PRE
                B(pre_A2)
        THEN
                B(post_A2)
        END
END
```

FIGURE 4: Machine B S_A2 à générer

```
COMPONENT  /* another example */
           B
INTERFACE
        provides : S_B
VARIABLES
        var_B
INVARIANT
        inv_B
INITIALIZATION
        init_B
SERVICES
        provided S_B
                 Pre pre_B
                 Behavior eLTS_B
                 Post post_B
        end ;
END_SERVICES
```

FIGURE 5: Composant Kmelia B

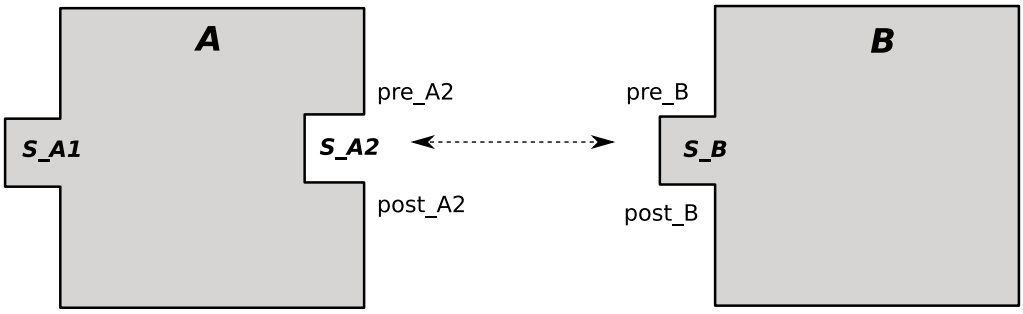


FIGURE 6: Assemblage A–B

# 3 Assemblage Kmelia

Nous supposons qu'un assemblage est établit entre l'interface requise S_A2 du composant A et l'interface fournie S_B du composant B. En d'autre terme, il s'agit de montrer que S_B est une implantation "correcte" de S_A2 (un raffinement, dans le langage de B).
Il est nécessaire de pouvoir exprimer dans l'assemblage le lien (renommage, transtypage, etc.) qu'il y a entre les variables de contexte var_A2 de S_A2 et les variables var_B du composant B.
Supposons que ce lien soit exprimé par un prédicat $link(\text{var\_A2}, \text{var\_B})$.

## 3.1 Obligations de preuve B

Les OP suivantes garantiraient la correction de l'assemblage Kmelia.
– L'initialisation de l'implantation S_B respecte l'initialisation de S_A2

$$[\ \beta(\text{init\_B})\ ]\ \neg\ [\ \beta(\text{init\_A2})\ ]\ \neg(\beta(\text{inv\_B}) \wedge link(\text{var\_A2}, \text{var\_B}))$$

de tête, à vérifier...
– Le service S_B est une implantation "correcte" de S_A2

$$\beta(\text{inv\_A1}) \wedge \beta(\text{inv\_B}) \wedge link(\text{var\_A2}, \text{var\_B}) \wedge \beta(\text{pre\_A}) \Rightarrow$$

$$\beta(\text{pre\_B}) \wedge\ [\ \beta(\text{post\_B})\ ]\ \neg\ [\ \beta(\text{post\_A1})\ ]\ \neg(\beta(\text{inv\_B}) \wedge link(\text{var\_A2}, \text{var\_B}))$$

idem...

## 3.2 Machines B à générer

Pour que les obligations de preuve précédentes soient générées il va s'agir de construire deux modèles B :
– la première machine correspond à la ré-expression du service S_A2. Cette machine a déjà été exprimée (voir figure 4) ;
– La seconde manchine correspond à un raffiment. Il s'agit ici d'exprimer le service S_B implantant S_A2 (voir figure 7).
**Note :** attention si S_B propose des paramètres différents de S_A2.

```
REFINEMENT
        S_B
REFINES
        S_A2
VARIABLES
        B(var_B)
INVARIANT
        B(var_B)
        ∧ B(inv_B)
        ∧ link(var_A2, varB)
INITIALISATION
        B(init_B)
OPERATIONS
        B(S_A2) =
        PRE
                B(pre_B)
        THEN
                B(post_B)
        END
END
```

FIGURE 7: Raffinement B de S_B à générer