

Atelier B

MDELTA

Manuel Utilisateur

version 2.0



ATELIER B
MDELTA Manuel Utilisateur
version 2.0

Document établi par CLEARSY.

Ce document est la propriété de CLEARSY et ne doit pas être copié, reproduit, dupliqué
totalement ou partiellement sans autorisation écrite.

Tous les noms des produits cités sont des marques déposées par leurs auteurs respectifs.

CLEARSY

Maintenance ATELIER B
Parc de la Duranne
320 avenue Archimède
Les Pléiades III - Bât. A
13857 Aix-en-Provence Cedex 3
France

Tél 33 (0)4 42 37 12 70

Fax 33 (0)4 42 37 12 71

mail : contact@atelierb.eu

Table des matières

1	Introduction	1
1.1	Objet	1
2	Terminologie	3
2.1	Abréviations	3
2.2	Définition des termes employés	3
3	Présentation du problème	5
3.1	Exemple	5
3.2	Conditions de bonne définition	6
3.3	Exemple de génération de lemmes de bonne définition	6
3.4	Localisation des EDS	7
3.5	Outil mdelta pour la vérification de bonne définition	7
4	Utilisation de l'outil mdelta	9
4.1	Quand l'utiliser	9
4.2	Etape I : procédure de lancement	9
4.2.1	Exemple	9
4.2.2	Résultat de l'étape	10
4.3	Etape II : vérification de l'exécution mdelta	10
4.4	Etape III : Création du projet B <i>delta_NOM_PROJET</i>	10
4.4.1	Création manuelle	10
4.4.2	Création automatique	11
4.4.3	Résultat de l'étape de lancement du script	11
4.5	Etape IV : Preuve des lemmes de bonne définition	11

Chapitre 1

Introduction

1.1 Objet

Un prédicat mathématique peut être :

- vrai
- ou faux

à condition qu'il soit **bien défini**.

La mauvaise définition d'un prédicat¹(ou d'une expression contenue dans ce prédicat) ne permet plus de garantir que la preuve, dans laquelle intervient ce prédicat, est juste.

L'Atelier B V3.6 n'a pas de moyen intrinsèque de détecter ces expressions dépourvues de sens².

Les EDS ne permettent donc pas de garantir pleinement la validité d'un travail de preuve effectué avec l'Atelier B V3.6.

La production de lemmes de bonne définition par l'outil mdelta, et leur preuve automatique ou manuelle, permettent de répondre à ce besoin.

Le présent document définit les expressions dépourvues (et potentiellement dépourvues) de sens et indique la manière de mettre en œuvre l'outil mdelta pour générer les lemmes de bonne définition dont la preuve assurera la validité d'un développement B³ du point de vue de sa bonne définition.

¹La vérification qu'un prédicat ou qu'une expression est bien typé est réalisée par le vérificateur de type. Cette vérification du typage est indépendante de la vérification de bonne définition. La bonne définition n'est étudiée que pour des prédicats et expressions bien typés.

²En abrégé EDS (voir § Terminologie)

³La méthode B est présentée dans [1].

Chapitre 2

Terminologie

2.1 Abréviations

PO : obligation de preuve.

GOP : générateur d'obligation de preuve.

BDR : Base de règles du Prouveur.

BDP : Répertoire Base de Données pour un Projet.

PatchProver : Fichier contenant des règles de l'utilisateur et permettant d'enrichir la BDR, pour l'ensemble d'un projet.

fichier Pmm : Fichier suffixé pmm contenant des règles de l'utilisateur et permettant d'enrichir la BDR, pour un composant d'un projet.

fichier Pmi : Fichier suffixé pmi contenant, en particulier, les commandes interactives saisies par l'utilisateur, pour chaque PO d'un composant.

fichier PO : Fichier suffixé po contenant les PO d'un composant.

Composant : terme désignant indifféremment une machine abstraite, un raffinement ou une implémentation au sens B.

EDS : Expression Dépourvue de Sens

EPDS : Expression Potentiellement Dépourvue de Sens

2.2 Définition des termes employés

Langage de théorie : langage utilisé pour réaliser certains outils de l'Atelier B V3.6 et le prouveur en particulier.

Obligation de preuve : prédicat logique fabriqué par l'Atelier B V3.6 à partir d'un composant écrit en langage B¹, qu'il faut démontrer pour assurer la cohérence de ce composant.

Expression Dépourvue de Sens : expression du langage B à laquelle on ne peut associer aucune interprétation mathématique comme l'évaluation d'une fonction partielle en un point n'appartenant pas à son domaine ou encore le minimum d'un ensemble vide.

¹Le langage B est présenté dans [2].

Expression Potentiellement Dépourvue de Sens : expression du langage B nécessitant certaines conditions pour ne pas être dépourvue de sens (ces conditions sont listées dans le tableau du chapitre 3).

Lemme de Bonne Définition : prédicat logique qu'il faut démontrer pour assurer la bonne définition d'une expression.

Chapitre 3

Présentation du problème

3.1 Exemple

Soit l'expression

$$y = \frac{x}{\frac{x+8}{c}} \tag{3.1}$$

Cette expression peut être vraie ou fausse, à la condition qu'elle soit bien définie. Si cette expression n'est pas bien définie, il est alors impossible de lui associer une valeur vraie ou fausse. Cette mauvaise définition signifie qu'au moins un opérateur de l'expression a au moins un opérande qui n'appartient pas à son domaine de définition.

L'expression (3.1) est manifestement de nature arithmétique. On considère que l'expression est bien typée (opération réalisée par le vérificateur de types) et que y , x et c sont des entiers relatifs.

Les opérateurs apparaissant dans l'expression (3.1) sont :

- L'égalité
Une égalité $a=b$ est bien définie à condition que a et b soient bien définis
 - l'addition
Une addition $a+b$ est bien définie à condition que a et b soient bien définis
 - la division entière
Une division entière a/b est bien définie si a et b sont bien définis et si b est non nul
- Il va donc falloir vérifier que :
- x et c sont bien définis (ce qui est le cas puisque l'on a supposé que y , x et c sont des entiers relatifs)
 - le dénominateur de $x/(x + 8/c)$ est non nul
 - le dénominateur de $8/c$ est non nul

La bonne définition de l'expression (3.1) passe par la démonstration des prédicats suivants :

$$(x + 8)/c \neq 0 \tag{3.2}$$

$$c \neq 0 \tag{3.3}$$

Le contexte de l'expression doit contenir ces prédicats sous forme d'hypothèses ou doit permettre de le déduire.

Si tel n'est pas le cas, l'expression (3.1) est potentiellement mal définie.

On se reportera au tableau (1) pour la liste des expressions pouvant être mal définies.

3.2 Conditions de bonne définition

Les conditions de bonne définition¹ sont recensées dans le tableau ci-dessous.

Expression	Condition de bonne définition
a^b	$a \in \mathbb{N} \wedge b \in \mathbb{N}$
$a \bmod b$	$b \in \mathbb{N}_1 \wedge a \in \mathbb{N}$
a/b	$b \in \mathbb{Z}_1$
$\Pi(x).(P E)$	$\{x P\} \in \mathbb{F}(\{x P\})$
$\Sigma(x).(P E)$	$\{x P\} \in \mathbb{F}(\{x P\})$
$\max(S)$	$S \cap \mathbb{N} \in \mathbb{F}(\mathbb{N}) \wedge S \neq \emptyset$
$\min(S)$	$S \cap (\mathbb{Z} - \mathbb{N}) \in \mathbb{F}(\mathbb{Z}) \wedge S \neq \emptyset$
$\text{card}(S)$	$S \in \mathbb{F}(S)$
$\text{inter}(U)$	$U \neq \emptyset$
$\bigcap(x).(P E)$	$\{x P\} \neq \emptyset$
r^n	$n \in \mathbb{N}$
$f(x)$	$x \in \text{dom}(f) \wedge f \in \text{dom}(f) \leftrightarrow \text{ran}(f)$
$\text{perm}(S)$	$S \in \mathbb{F}(S)$
$\text{conc}(s)$	$s \in \text{seq}(\text{ran}(s)) \wedge \forall x.(x \in \text{dom}(s) \Rightarrow s(x) \in \text{seq}(\text{ran}(s(x))))$
$s \hat{\ } t$	$s \in \text{seq}(\text{ran}(s)) \wedge t \in \text{seq}(\text{ran}(t))$
$\text{size}(s)$	$s \in \text{seq}(\text{ran}(s))$
$\text{rev}(s)$	$s \in \text{seq}(\text{ran}(s))$
$s \leftarrow e$	$s \in \text{seq}(\text{ran}(s))$
$e \rightarrow s$	$s \in \text{seq}(\text{ran}(s))$
$\text{tail}(s)$	$\text{size}(s) \geq 1 \wedge s \in \text{seq}(\text{ran}(s))$
$\text{first}(s)$	$\text{size}(s) \geq 1 \wedge s \in \text{seq}(\text{ran}(s))$
$\text{front}(s)$	$\text{size}(s) \geq 1 \wedge s \in \text{seq}(\text{ran}(s))$
$\text{last}(s)$	$\text{size}(s) \geq 1 \wedge s \in \text{seq}(\text{ran}(s))$
$s \uparrow n$	$n \in 0 \dots \text{size}(s) \wedge s \in \text{seq}(\text{ran}(s))$
$s \downarrow n$	$n \in 0 \dots \text{size}(s) \wedge s \in \text{seq}(\text{ran}(s))$

Tableau (1) : Expressions potentiellement dépourvues de sens

3.3 Exemple de génération de lemmes de bonne définition

Soit l'expression $x/((x+8)/c)$.

Cette expression est bien définie si

- x est bien défini (vrai car c 'est une variable)
- $(x+8)/c$ est bien défini si
 - $x+8$ est bien défini si
 - x est bien défini (vrai car c 'est une variable)
 - 8 est bien défini (vrai car c 'est un entier)
 - c est bien défini (vrai car c 'est une variable)
 - c est non nul (voir la troisième ligne du tableau (1))

¹Ces conditions sont tirées de [3] et [4].

– $(x + 8)/c$ est non nul (voir la troisième ligne du tableau (1))

Les lemmes de bonne définition de l'expression $x/(x + 8/c)$ sont donc :

$H \vdash x + 8 \neq 0$

$H \vdash c \neq 0$

où H désigne le contexte courant des hypothèses.

3.4 Localisation des EDS

Des EDS peuvent exister, pour un projet B donné, au niveau :

- des composants B
- des règles de l'utilisateur pour le projet entier,
- des règles de l'utilisateur pour un composant donné (du projet)
- des commandes interactives de preuve

3.5 Outil mdelta pour la vérification de bonne définition

L'outil mdelta ne détecte pas, à proprement parlé, les EDS. Lancé sur un projet B donné, il réalise la génération de lemmes de bonne définition pour l'ensemble des EPDS du projet, et cherche à les prouver automatiquement. Le résultat est disponible dans un projet B "fils", créé ou mis à jour automatiquement.

Les fichiers impactés par la recherche d'EPDS sont :

- les fichiers B des composants (forme normale, fichiers suffixés 'nf')
- le fichier Patchprover,
- les fichiers Pmm,
- les fichiers Pmi et les fichiers Po associés (les fichiers Po sont nécessaires pour obtenir les informations de contexte.)

Les lemmes de bonne définition sont regroupés et leur origine identifiable (traçabilité des EPDS).

Chapitre 4

Utilisation de l'outil mdelta

4.1 Quand l'utiliser

Cet outil peut être lancé sur n'importe quel projet B, à n'importe quel moment de son développement, sans que le travail de preuve soit terminé. Il faut néanmoins que les composants soient passés avec succès au typechecker.

Il est fortement recommandé d'utiliser mdelta à deux moments particuliers d'un projet B :

- Lorsque le projet est validé de manière informelle (toutes les po semblent justes, sans travail de preuve formelle)
- et lorsque le travail de preuve a validé le projet de manière formelle (des EPDS peuvent avoir été introduites pendant le travail de preuve interactive, par l'intermédiaire de commandes `ah`, `ph` ou `se` par exemple ou de règles utilisateurs)

4.2 Etape I : procédure de lancement

La commande à taper est la suivante :

```
mdelta < CHOIX_DELTA >< ATB_HOME >< ATB_RESSOURCE >< NOM_PROJET >
```

avec :

- CHOIX_DELTA : directives de compilation :
 - `-r` : pour lancer l'outil sur les règles utilisateurs (fichiers Pmm et PatchProver),
 - `-p` : pour lancer l'outil sur les commandes interactives (fichiers `.pmi` et `.po`),
 - `-b` : pour lancer l'outil sur les composants B (fichiers `.nf`),
 - n'importe quelle combinaison des trois options précédentes,
 - `-a` : pour lancer l'outil complet (équivalent à `-r -p -b`).
- ATB_HOME : répertoire racine de l'Atelier B V3.6 à utiliser
- ATB_RESSOURCE : le chemin complet du fichier de ressources de l'Atelier B V3.6 à utiliser
- NOM_PROJET : le projet déclaré dans l'Atelier B V3.6 sur lequel on veut lancer l'outil

4.2.1 Exemple

```
mdelta -a /beta/3.6/AB /beta/3.6/AB/AtelierB DeltaTest
```

4.2.2 Résultat de l'étape

L'ensemble des résultats de cette étape est inscrit dans le répertoire BDP du projet `NOM_PROJET`, sous la forme de fichiers suivants :

- Pour les EPDS d'un composant B 'X' (lorsque l'option `-a` ou `-b` a été choisie) :
 - `X_wd.po` : fichier contenant les lemmes de bonne définition
 - `X_wd.pmi` : fichier contenant le travail de preuve interactive à réaliser
- Pour les EPDS de la preuve interactive d'un composant B 'X' (lorsque l'option `-a` ou `-p` a été choisie) :
 - `X_pmi.po`
 - `X_pmi.pmi`
- Pour les EPDS du fichier PatchProver (lorsque l'option `-a` ou `-r` a été choisie) :
 - `PatchProver.lemmas`
 - `PatchProver.unresolved` : contient les expressions qui n'ont pas pu être analysées car la syntaxe est trop imprécise
 - `PatchProver.res` : contient les lemmes du PatchProver.lemmas ainsi que leur état après utilisation du prouveur de prédicat (prouvé ou non prouvé).
- Pour les EPDS de chaque fichier Pmm (lorsque l'option `-a` ou `-r` a été choisie) :
 - `X.lemmas`
 - `X.unresolved`
 - `X.res`
- Un fichier script nommé `createDelta.sh` (cf Etape III) lorsque l'option `-a` a été choisie.

4.3 Etape II : vérification de l'exécution mdelta

Il faut vérifier le bon déroulement de l'étape I, en étudiant les traces générées à l'écran. Si certains messages indiquent un problème :

- analyser et corriger l'origine du problème
- Relancer l'étape I

4.4 Etape III : Création du projet B *delta_NOM_PROJET*

4.4.1 Création manuelle

Pour utiliser l'AtelierB sur les lemmes de bonne définition générés à l'étape précédente, il faut disposer d'un projet B "factice" dont les obligations de preuve correspondent aux lemmes de bonne définition.

Pour cela, il suffit de créer un projet avec ses répertoires `bdp`, `src` et `lang`, d'y ajouter les machines vides¹ `X_wd.mch` et `X_pmi.mch` (dont les fichiers `X_wd.pmi`, `X_wd.po`, `X_pmi.pmi` et `X_pmi.po` ont été générés à l'étape précédente), de les "type-checker" et de lancer le générateur d'obligations de preuves (qui n'en générera pas puisque les machines sont vides) sur chacune d'elles.

¹X.mch est une machine vide si elle est réduite à "MACHINE X END"

Une fois ceci fait, il reste à remplacer les fichiers `.pmi` et `.po` générés par le générateur d'obligations de preuve par ceux créés par le `mdelta`.

Les lemmes de bonne définitions sont maintenant disponibles pour un travail de preuve interactive.

4.4.2 Création automatique

Le script `createDelta.sh`, généré pendant l'étape I avec l'option `-a`, permet de créer un projet B `delta_NOM_PROJET` afin de :

- Visualiser les lemmes de bonne définition à partir de l'atelier B
- Effectuer un travail de preuve interactive sur ces lemmes

Tapez simplement la commande `sh ./createDelta.sh`.

4.4.3 Résultat de l'étape de lancement du script

- Si un projet `delta_NOM_PROJET` existe déjà :
 - les données importantes (fichiers `.po`, `.pmi`, `.lemmas`, `.res` et `.unresolved` du BDP du projet `delta_NOM_PROJET` sont sauvegardées, au même endroit, (fichiers renommés en `*.< date de la sauvegarde >`)
 - il est détruit
- le projet `delta_NOM_PROJET` est recréé avec des composants “factices” mais nécessaires au travail de preuve des lemmes de bonne définition.
- Les fichiers créés à l'étape I sont déplacés vers le répertoire BDP du projet `delta_NOM_PROJET`.

4.5 Etape IV : Preuve des lemmes de bonne définition

Le travail de vérification informelle et/ou de preuve formelle des lemmes de bonne définition peut commencer :

- Pour les lemmes correspondants aux composants ou aux commandes interactives : travail de preuve interactive classique, sous l'atelier B, en ayant ouvert le projet B `delta_NOM_PROJET`.
- Pour les lemmes correspondants aux bases de règles (données des fichiers suffixés ‘lemmas’) : Ces lemmes ne sont pas accessibles depuis l'Ihm de l'atelier. Les fichiers suffixés “res” contiennent le résultat de l'application du prouveur de prédicat sur ces lemmes (effectuée lors de l'étape I), c'est-à-dire leur état de preuve (Proved ou Unproved). Les lemmes non prouvés doivent alors être démontrés manuellement par l'utilisateur qui devra fournir un document de preuve formelle.
- Pour les règles contenues dans les fichier “.unresolved” : Elles doivent aussi être démontrées manuellement mais nécessitent deux étapes supplémentaires préalables à la preuve proprement dite. En premier lieu, l'utilisateur doit traduire ces règles en lemmes mathématiques² susceptibles d'être prouvés. Une fois la traduction effectuée, l'utilisateur doit en déduire les lemmes de bonne définition correspondants (d'après le tableau du chapitre 3 du

²Utiliser [5]

présent document). La preuve de ces lemmes constitue alors la dernière étape du traitement des fichiers “.unresolved”.

Bibliographie

- [1] B-Book, Jean-Raymond ABRIAL (1996).
- [2] Langage B - Manuel de référence, version 1.1, ClearSy.
- [3] Partial logics reconsidered : a conservative approach, Formal aspect of computing, Olaf OWE (1993).
- [4] Traitement des expressions dépourvues de sens de la théorie des ensembles. Application à la méthode B (Thèse de Doctorat), Lilian BURDY.
- [5] Guide de Rédaction des Règles, version 1.0, ClearSy.