Chapitre 2

Définition et mise en place de AsmL

Nous avons expliqué la notion informelle de *langage algorithmiquement complet* dans le chapitre un, en indiquant que le langage ASM était un tel langage (le seul d'ailleurs à ce jour). Avant de donner une définition formelle de ce langage, nous allons introduire, petit à petit, les primitives de celui-ci sur des exemples. Pour l'instant nous allons présenter les ASM comme un nouveau langage de programmation, un de plus pourrait-on dire. Nous verrons dans les chapitres ultérieurs en quoi il est révolutionnaire.

Le langage ASM est un langage théorique poursuivant des buts théoriques. Il existe cependant plusieurs implémentations partielles des ASM, "partiels" car évidemment toutes restreintes aux fonctions calculables et non aux fonctions calculables avec oracles. Une de ces implémentations a été mise au point sous la direction de Yuri Gurevich lui-même aux laboratoires Microsoft, sous le nom de AsmL (pour *ASM Language*). D'abord écrite en C++, il s'agit maintenant d'un des langages de la technologie .NET de Microsoft.

Pour permettre au lecteur de s'affronter à un vrai langage de programmation, nous allons illustrer notre introduction aux ASM par ce langage. Commençons, dans ce chapitre, à voir comment le mettre en place.

2.1 Définition de AsmL

2.1.1 Les implémentations des ASM

Il existe un page Web officielle sur les ASM de Yuri Gurevich, ses implémentations et ses applications :

`http ://www.eecs.umich.edu/gasm/`

En particulier, la rubrique *tools* renvoie aux implémentations des ASM. Nous ne nous intéresserons qu'à l'une d'entre elles : AsmL.

2.1.2 Le cas de AsmL

En 1991, Bill Gates, président de Microsoft, crée les laboratoires Microsoft, situés à Redmond près de Seattle, à côté du siège de Microsoft. Il s'agit d'un organisme de recherche fondamentale, indépendant de la recherche-développement (à l'instar des fameux feu Bell Labs pour la société ATT). Le responsable de ces laboratoires prend contact avec Yuri Gurevich, alors professeur à l'université Ann Arbor dans le Michigan, pour que ce dernier développe ses résultats en complexité algorithmique. Gurevich accepte de venir aux laboratoires Microsoft, mais pour développer ses ASM, ce qui ne posa pas de problème.

Il constitue donc une petite équipe qui s'attelle à sa propre implémentation des ASM, d'abord en C++ puis dans la technologie .NET pour AsmL 2.0 :

`http ://www.research.microsoft.com/foundations/AsmL/`

Cette équipe développera ensuite un outil de méthodes formelles appelé `SpecExplorer` dont AsmL sera un constituant. Seul AsmL nous intéressera ici.

2.1.3 La technologie .NET

SUN a réussi une formidable percée en 1996 avec sa sortie du langage Java : il s'agit du premier langage multi-plate-forme qui connaisse un succès, en particulier grâce à l'adoption par Netscape, le navigateur Web phare de l'époque, de la prise en compte des applets.

Microsoft essaie de se positionner sur le créneau Java avec *Visual J++* mais se retire rapidement puis propose en 2001 la technologie .NET (lire *dot net* en anglais, *point net* en français). À proprement parler, .NET est une nouvelle API (*Application Programmer Interface*) amenée à remplacer l'API Windows mais également les autres bibliothèques proposées par Microsoft : elle est orientée objet, distribuée en utilisant XML ; elle utilise un pseudo-code intermédiaire, comme le Pascal UCSD ou Java, ce qui permet d'utiliser une variété de langages de haut niveau (Visual C++, Visual Basic ou le nouveau langage C# (lire *si sharp* en anglais, *c dièze* en français), proche de Java).

Pour plus de renseignements sur .NET, on consultera le site :

`http ://www.microsoft.com/net/`

2.2 Mise en place

Nous allons voir comment mettre en place le compilateur de AsmL. Puisque celui-ci est lié à la technologie .NET, il faut d'abord mettre en place la bibliothèque associée, appelée **Framework .NET**.

2.2.1 Mise en place de Windows :)

La technologie .NET est due à Microsoft et donc fortement orientée Windows. Nous supposons donc dans la suite que vous êtes sur une plate-forme Windows. Nous n'allons pas vraiment expliquer comment mettre en place Windows, cependant.

Ceci n'est pas réellement indispensable en fait. Pour pouvoir concurrencer Sun (et Java), Microsoft a proposé .NET comme un standard ECMA (ainsi que le langage C#). Il existe des implémentations pour Linux, voir par exemple le projet Mono :

`http ://www.go-mono.com/`

2.2.2 Mise en place de Framework .NET

Récupération.- Le Framework .NET est distribué (gratuitement) par Internet par chargement sur le site Microsoft :

`http ://msdn2.microsoft.com/fr-fr/netframework/default.aspx`

en cliquant sur `téléchargement` puis sur la dernière version du paquetage redistribuable, .Net Framework 3.0 en décembre 2007.

Attention .NET ne fonctionne que sur Windows à partir de Windows NT 4.0 (certaines parties telles que ASP .NET ne fonctionnent pas sur XP familial, mais cela a peu d'importance pour nous).

Installation.- La téléchargement nous fournit un fichier `dotnetfx3setup.exe` de 28 Mo. Il suffit de double-cliquer sur son icône, comme d'habitude, pour commencer son installation. Ceci crée un nouveau répertoire :

`C :\WINDOWS\Microsoft .NET\Framework\v3.0`

Il faut ensuite redémarrer l'ordinateur pour que cette nouvelle API soit prise en compte.

Visual Studio .NET.- Comme nous venons de le voir, Microsoft met Framework .NET gratuitement à la disposition de quiconque veut l'utiliser. Il commercialise par contre l'éditeur de développement intégré `Visual studio .NET`. Celui-ci n'est pas indispensable, nous ne l'utiliserons pas dans la suite.

Notons cependant pour ceux que ça intéresse que les accords passés en les universités et Microsoft permet aux étudiants de celles-ci, moyennant une légère rémunération annuelle de la part de l'université, d'obtenir gratuitement tous les outils de développement Microsoft (y compris le système d'exploitation Windows).

2.2.3 Mise en place du compilateur AsmL

Récupération.- Le compilateur AsmL est récupérable uniquement par téléchargement, à l'adresse :

`http ://research.microsoft.com/specexplorer/#Download`

en cliquant sur `here` dans la section *Download* puis sur `download`.

Installation.- Le fichier téléchargé :

`SpecExplorer.9520.msi`

de 12,1 MiO est un fichier d'installation Microsoft. Il suffit de double-cliquer dessus pour démarrer l'installation, en tant que super utilisateur. Choisir par exemple le répertoire :

`c :\Programme Files\Spec Explorer`

et *Everyone* si vous voulez que tous les utilisateurs de votre système puissent l'utiliser.

Remarque.- Cet exécutable est lié à Windows. Il ne fonctionne pas avec mono. Les sources ne sont pas publiques.

Variable d'environnement.- Le chemin des exécutables de AsmL n'est pas placé automatiquement. Il faut donc le placer à la main.

Comme d'habitude avec Windows, cliquez avec le bouton gauche de la souris sur le bouton "démarrer" qui se trouve en bas à gauche de l'écran. Un menu déroulant apparaît dans lequel il faut choisir "Panneau de configuration" avec le bouton gauche de la souris, puis "Performance et maintenance", puis "Système". Une fenètre, appelée "Propriétés système", apparaît alors avec sept onglets. Choisir l'onglet intitulé "Avancé" puis appuyez sur le bouton "variables d'environnement".

Dans "*variables systèmes*", choisir "*Path*". Ajouter à la fin (cela dépend du répertoire que vous avez choisi) :

`;C :\Program Files\Spec Explorer\bin`

en étant très soigneux sur l'orthographe, les espaces et la casse des caractères (d'autant plus qu'il faut bien dire que la mini-fenêtre déroulante est loin d'être très lisible).

Vérification.- On va vérifier que l'installation s'est bien déroulée (sinon il faut recommencer en vérifiant le *path*). Pour cela, faites apparaître une fenêtre de commande : *comme d'habitude avec Windows, appuyer avec le bouton gauche de la souris sur le bouton "démarrer" en bas à gauche de l'écran ; choisir "exécuter" vers le bas du menu déroulant qui apparaît ; tapez "Cmd.exe" dans la fenêtre qui apparaît.*

Une fenêtre à fond noir surgit dans laquelle on va pouvoir faire exécuter ce que l'on veut en ligne de commande. Tapez "asmlc" (pour *AsmL Compiler*). Si tout est bien installé, on devrait voir apparaître :

```
E:\>asmlc
Microsoft (R) AsmL for Microsoft .NET Compiler version 2.2.9520
running on Microsoft (R) .NET Framework v2.0.50727
Copyright (C) Microsoft Corporation 2001-2004. All rights reserved.
error: compilation errors: C# compilation: errors in generated C#:
       error CS5001: Le programme 'e:\output.exe' ne contient pas une
methode 'Main' statique appropriee pour un point d'entree

E:\>
```

Dans notre cas, tout s'est bien passé puisque le compilateur AsmL a été trouvé. L'erreur indiquée provient de ce qu'on n'a pas spécifié de programme source.

2.3 Premier programme

Première étape : écriture du programme source.- Écrivons le programme suivant :

```
Main()
  WriteLine("Bonjour")
```

avec un éditeur de texte, par exemple *Wordpad*.

Deuxième étape : sauvegarde du programme source.- Sauvegardons ce programme source en texte Unicode sous le nom 'bonjour.asml'.

Troisième étape : compilation.- Compilons notre programme :

```
>asmlc bonjour.asml
```

en ligne de commande. On obtient un nouveau fichier nommé `bonjour.exe`.

Quatrième étape : exécution.- Pour exécuter le programme objet, il suffit d'écrire :

```
>bonjour
```

(ou 'bonjour.exe' au choix) sur la ligne de commande puis d'appuyer sur la touche 'entrée'. Dans ce cas simple, 'Bonjour' est écrit sur la ligne suivante.

```
E:\>asmlc bonjour.asml
Microsoft (R) AsmL for Microsoft .NET Compiler version 2.2.9520
running on Microsoft (R) .NET Framework v.2.0.50727
Copyright (C) Microsoft Corporation 2001-2004. All rights reserved.

E:\>bonjour
Bonjour

E:\>
```