Compilation

Une fois un programme écrit, les instructions qu'il contient doivent être exécutées par la machine choisie. Chaque machine possède sa propre architecture (composants, mémoire, processeurs, périphériques, etc ...). Ces différents éléments électroniques possédent en général deux états stables, activé ou chargé et désactivé ou déchargé. Ceci impose une représentation utilisant des 0 et des 1, qui est dite « *binaire* ». Chacune de ces architectures est régie par son propre langage appelé « *langage machine* » ou « *assembleur* ». Par exemple le programme de la Figure 1, écrit dans le langage assembleur du processeur Intel 80386, calcule le minumum de trois entiers. Expliquer le fonctionnement d'un tel programme dépasse largement le cadre de cette fiche.

```
min:
    pushl   %ebp
    movl    %esp, %ebp
    movl    8(%ebp), %edx
    movl    12(%ebp), %ecx
    movl    16(%ebp), %eax
    cmpl    %edx, %ecx
    leave
    cmovbe  %ecx, %edx
    cmpl    %eax, %edx
    cmovbe  %edx, %eax
    ret
```

FIGURE 1 – Un exemple de programme assembleur

Cependant, *in fine*, la machine n'exécute que des opérations binaires sur des composants électroniques. Il faut donc produire un fichier exécutable, aussi appelé « *fichier binaire* », à partir du programme en assembleur pour que la machine puisse accomplir les instructions du programme.

Ce fichier binaire est généré par un « traducteur » (lui-même un programme), appelé *compilateur*, entre le langage de programmation et le langage propre de la machine constitué d'instructions en binaire. Plus généralement, un compilateur est un programme spécifique qui transforme un programme écrit dans un langage de programmation (le langage source) en un autre (le langage cible). Au cours de ce processus, le compilateur vérifie la syntaxe et effectue souvent des optimisations.

Les compilateurs et les langages machines offrent un moyen plus simple que le binaire pour contrôler une machine. Cependant, ces langages restent fastideux à utiliser. Ainsi sont nés les langages dit de *haut niveau* comme le C, le Java ou le Python. Ces langages, plus compréhensibles par l'humain, permettent une écriture plus facile des programmes. Lors de la production d'un fichier binaire, il y a finalement en quelque sorte deux compilations successives, une du langage de haut niveau vers le langage machine et une du langage machine vers le binaire.

La Figure 2 montre les principales étapes de la programmation, partant de la conception d'un algorithme par un être humain jusqu'à la réalisation d'actions par une machine. Chaque langage de programmation est traduit vers les langages assembleurs d'un certain nombre de machines cibles à l'aide de compilateurs spécifiques. Par exemple, il n'exsite pas encore de compilateur pour le langage Prolog vers les langages machines des consoles de jeu.

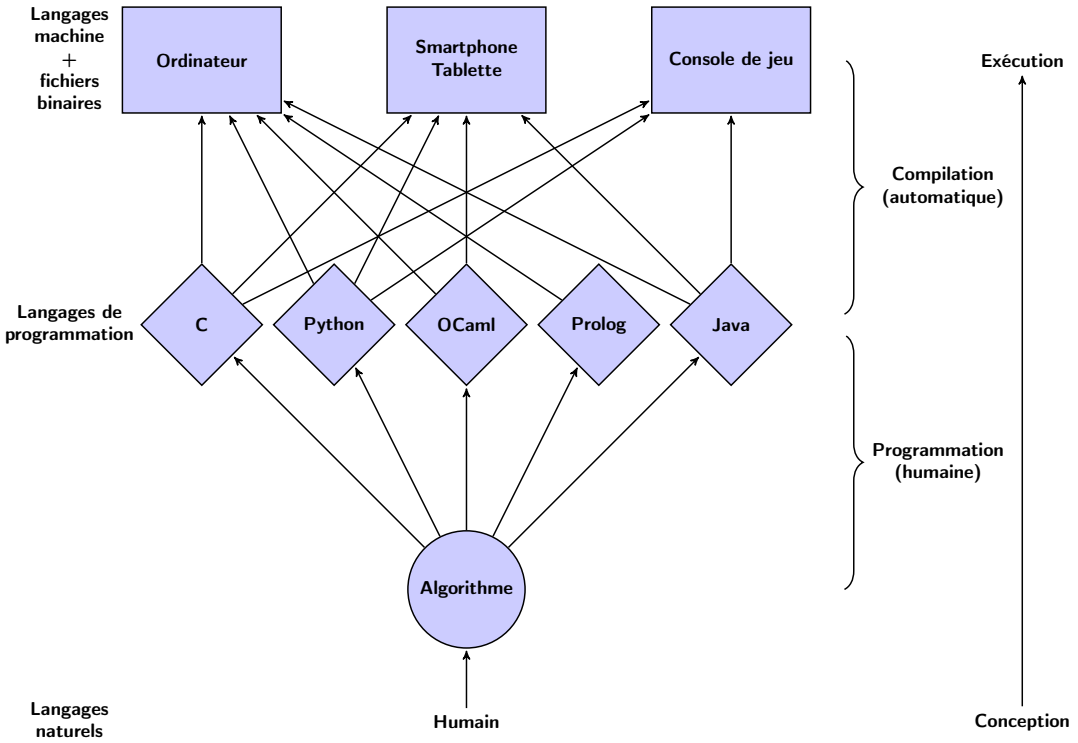

FIGURE 2 – Étapes de la programmation : de la conception à l'exécution.

Selon les langages de programmation l'étape de compilation peut s'effectuer à la volée pendant l'exécution (on parle alors d'interpréteur au lieu de compilateur) ou bien préalablement. Par exemple le C est un langage compilé tandis que le langage Prolog possède un interpréteur, et le langage Python peut être interprété ou compilé selon le choix de l'utilisateur.