UNIVERSITÉ DE ROUEN

Compilateur LLVM

Langage jouet Kawa

Architecture globale 0.1

14 avril 2015

Auteur(s): Kheireddine Berkane,Alexendre Petre et Nasser Adjibi

Version	Date	Changelog
0.1	02/04/2015	Version initiale pour l'introduction à l'architecture global.
0.2	14/04/2015	Apporter des détails concernant les différents modules de l'architecture ainsi que l'objectif initial du projet kawa.

Table des matières

1 Architecture globale

L'objectif de notre projet est de réaliser un compilateur à l'aide des outils LLVM (Low Level Virtual Machine) permettant de convertir un langage orienté objet ressemblant à Java (qu'on appellera **Kawa**) en langage machine.

Pour ce faire il faudra passer par différentes étapes d'analyse et de constructions afin d'obtenir le résultat attendu. Voici la schématisation (très simplifiée) de ces étapes.

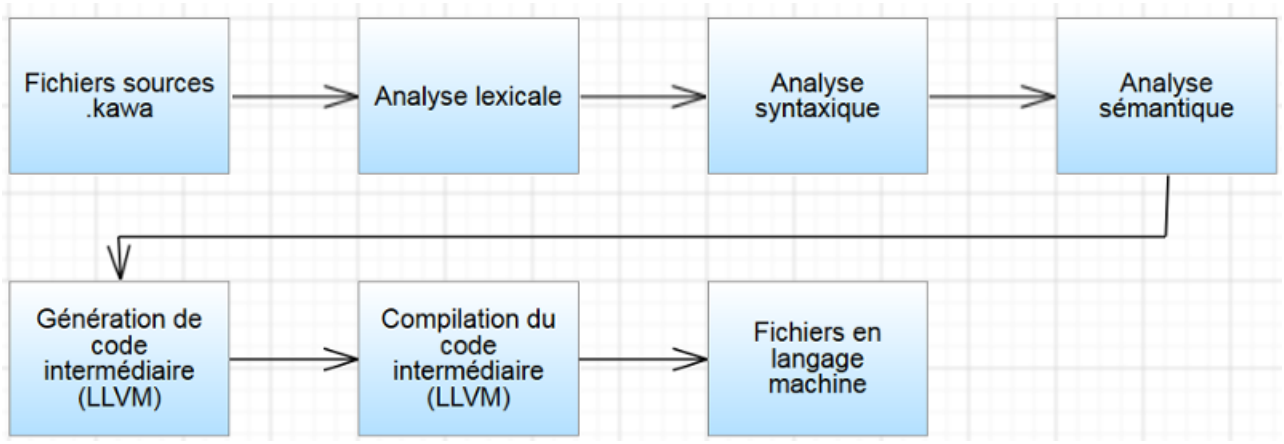

FIGURE 1 – Architecture globale.

Afin de comprendre et d'apréhender le travail à réaliser pour chacunes de ces étapes, nous avons mené des recherches sur le fonctionnement du compilateur ainsi que sur la plateforme LLVM. Ainsi nous proposons une architecture modulaire (reprenant les idées du schéma précédent) qui nous permet de découpler les traitements à réaliser en différentes phases d'analyses. Cela permettra également de rendre le développement du compilateur plus simple à mettre en place et la maintenance plus facile à gérer et à faire évoluer.

Les différents modules communiquent entre eux via des interfaces de connexion, un module orchestre les appels et l'ordre d'exécution des autres modules afin d'avoir toute la chaîne de compilation.

Notre architecture est donc ainsi découpée :

— Une structure (collection de classes) qui représente l'arbre en mémoire : le KawaTree
— Module d'analyse syntaxique (parseur)
— Module d'analyse sémantique
— Module back end

2 Descriptions des modules

2.1 Le Module KawaTree

Le KawaTree est le module au centre de la compilation car il agit comme un point de communication pour tout les autres modules. Le KawaTree est un arbre dont chacun des noeuds est une représentation des éléments du programme. Il est crée et approvisionné par le module syntaxique, lu et décoré par le module sémantique et enfin consulté par le module back end pour déterminer le code à produire.

2.2 Module syntaxique

C'est le module syntaxique qui débute la chaine de compilation.
L'objectif de cette partie est de produire un arbre syntaxique représentant le programme fourni au compilateur par le biais d'un ou de plusieurs fichiers sources Kawa.
C'est donc ici que la syntaxe des fichiers sources est vérifiée et validée. En cas d'erreur le compilateur doit s'arrêter et indiquer une erreur.

Le module syntaxique se découpe en deux phases :

- Analyse Lexicale : identifie et repère les **tokens** au sein des fichiers sources.
- Analyse Syntaxique : identifie les chaines de caractères appartenant à la syntaxe du langage Kawa, grâce aux tokens précédements trouvées par l'analyse lexicale, afin de construire en mémoire un arbre syntaxique (AST) représentant le programme.

Remarque : ces deux phases sont cependant réalisées de manières conjointes.

2.3 Module sémantique

Le module sémantique est la deuxième roue de la chaine de compilation, il ne sera effectué que si le module syntaxique ne rencontre pas d'erreur.

Dans cette partie on vérifie la sémantique du programme et on décore (i.e. : on ajoute des informations) l'arbre fourni par le module syntaxique.

Il faudra donc parcourir l'AST afin de vérifier le respect des nombreuses règles sémantiques mais également afin d'y ajouter des informations nécessaires à la génération du code (module suivant).

2.4 Module back end

C'est la dernière étape de la chaine de compilation. Il est chargé de traduire la structure de l'AST décoré en un language compréhensible par la machine. Son rôle se déroule en deux étapes. La premiere étape consiste à la production du code intermédiare LLVM. La deuxieme étape quant à elle compilera le code intermédiare en code machine au format ELF.

3 Résumé des étapes de compilation

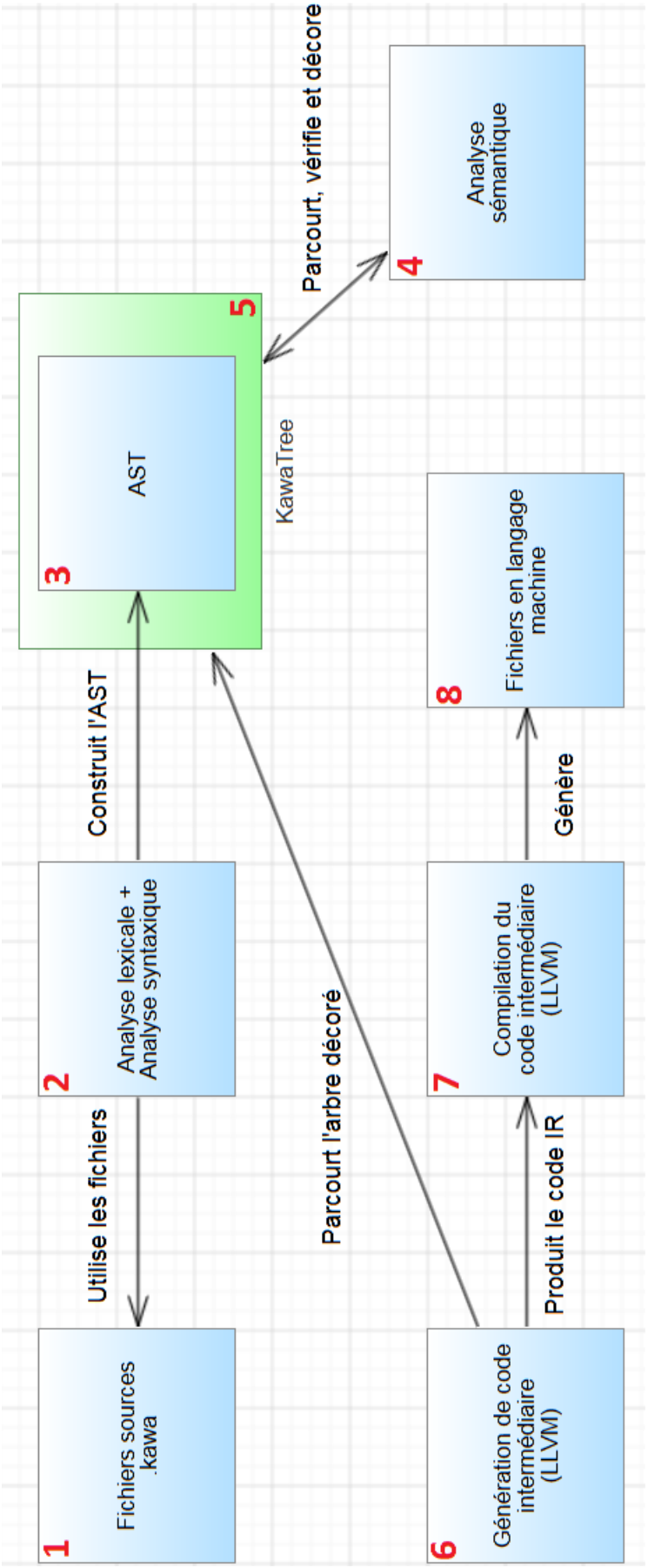

FIGURE 2 – Architecture globale.