Equipe: Joseline GOHEP, Cédric SONGO, Jordane TSAFACK

# Langage de programmation HUB

*Hub* est un langage de programmation orienté objet permettant de générer des architectures distribuées optimales pour des besoins spécifiques (calcul, stockage, et sur le long terme d'évaluer les performances d'une architecture de calcul distribué.

Cependant, faute de temps nous nous sommes contentés de concevoir un langage de description d'infrastructure distribuée. Ainsi notre compilateur prend en entrée un code source écrit en *HUB* et génère en sortie un rendu visuel de l'infrastructure distribuée au format svg.

## 1. EBNF

HARDWARE → {Declarations}

NETWORK → {Declarations}

SERVICE→ {Declarations}

CONFIG → {Statements}

PROCESSING → {Statements}

Declarations → {Declaration}

Declaration → Type Identifier =Type ([ Primary] {, Primary})

Statements→ {Statement}

Statement → SelectedObjetCall([Primary] {, Primary})

Primary→ Identifier|Litteral

Identifier→ Letter {Letter | Digit}

Type → Cluster |Network| int | float | double

Litteral → Interger | Float | String

String → Letter {, Letter}

Digit → 0| 1|2 | …. |9

Letter → A| B| B| C | ……. |Z

## 2. DESCRIPTION DU TRAVAIL

Nous avons réalisé une chaîne de compilation en python :

**Lexer, Parser,Visitor,Render, Compiler**

Notre travail se distingue de chaîne de compilation classique par la classe *Render* (dans un souci de modularité) dont les méthodes sont appelées à la visite des différents nœuds de notre AST et qui a pour vocation de créer de manière itérative un visuel final de l'infrastructure.

## 3. UTILISATION

Afin de tester notre langage nous avons écrit des fichiers textes que vous trouverez dans le répertoire tests :

A la sortie de notre compilateur nous n'avons pas du code machine mais un schéma de notre cluster laissant voir la structure générale de l'infrastructure, on observe ainsi les nodes composant les différents clusters. Ainsi que les liaisons entre les différents et réseaux et les réseaux auxquels les clusters sont reliés.

Exemple pour le fichier *test_visiteur.hub* du répertoire tests

Le *fichier test_visiteur.hub* : décrit un architecture de calcul ( cluster , réseaux (débit) )

La commande suivante permet d'obtenir le rendu la visite

***Python 3.5 comipiler.py test_final.hub test_final.html***

Le fichier du rendu final se trouve dans le dossier ***Render*** sous le nom de ***test_final.svg***

Sur la figure 1 : on note le nom de chaque cluster ainsi que les différents nodes qui le constituent. Tout en bas se trouvent les différents réseaux suivis de leur défit. Les traits noirs représentent les liaisons réseaux.

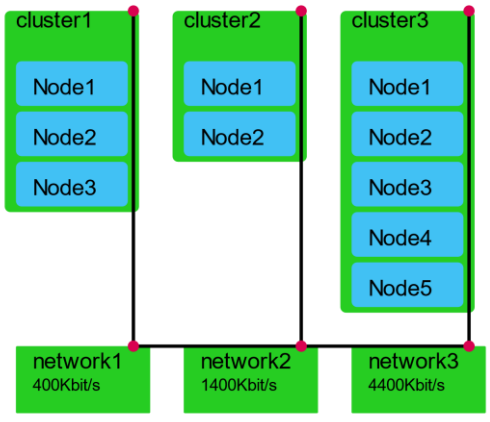


*Figure 1 : rendu pour le fichier test_final*