UNIVERSITÉ D'ÉVRY – VAL D'ESSONNE


# Réalisation en python d'un outils de gestion de services distribués

à l'aide de l'API python clustershell

**Guillaume Dubroeucq**
**Théo Poccard**
**Nicolas Chapron**

le 18 octobre 2016

*Professeur*
Patrice LUCAS
*adresse mail*
*patrice.lucas@cea.fr*

Année universitaire 2016-2017

# Table des matières

# 1 Introduction

## 1.1 Objectif

Développée et utilisée au CEA, l'API Clusterhell est une bibliothèque en Python qui permet d'exécuter en parallèle des commandes local et distantes sur des nœuds d'un cluster. Elle fournit également 3 outils en ligne de commande (script utilitaires basés dessus) qui nous permettent de bénéficier des fonctionnalités de la bibliothèque : clush,nodeset et clubak.
Ce projet nous demande de réalisé et de développer un outils en ligne de commande de gestion distribué des service de systèmes permettant d'administrer ces services sur plusieurs nœuds, et cela en utilisant l'API Python ClusterShell.
Nous allons donc dans un premier temps implémenter une version basique de gestion de services avec des fonctionnalités simple comme : start, stop, restart , etc.. sur un ensemble de nœuds distant. Puis une fois cette base réalisé, nous allons mettre en place une configuration statique de la répartition des services grâce à des fichiers. Et pour finir nous développerons une IHM à partir des éléments déjà crée afin de parfaire l'outil de gestion des services distribué.

## 1.2 Présentation des outils

Commençons tout d'abord par définir les 3 fonctionnalités de la bibliothèques de ClusterShell définit plus haut :crush,nodeset et clubak.

- Nodeset : Permet la création et la manipulation de liste de nœuds . En effet on peut créer des listes machines ainsi que des ranges de nœuds, on peut effectuer plusieurs opérations sur ces listes ( union, exclusion, intersection , etc...)
- Clush : Permet l'exécution des commandes en parallèle sur des machines distantes, prends également en charge les groupes.
- Clubak : Regroupement de sorties standards qui permet de présenter de manière synthétique un résultat d'exécution un peu trop verbeux.

# 2 Gestion des Services

# 3 Configuration des Services

# 4 Création d'une IHM

Pour la création d'une interface graphique, nous nous sommes tournés vers l'environnement de développement Qt. Qt est basé sur le langage C++ pour créer ses IHM. Cependant il existe le module PyQt permettant de programmer en python une interface graphique aisément.

## 4.1 Installation de Qt Creator et PyQt

## 4.2 Utilisation de Qt Creator et PyQt

Au départ, on utilise Qt Creator pour pouvoir créer les fenêtres avec tous les composants nécessaires. Lorsque l'on créé une fenêtre, Qt nous génère un fichier **.ui**. A l'aide de l'utilitaire **pyuic**, on peut convertir ce fichier en python.

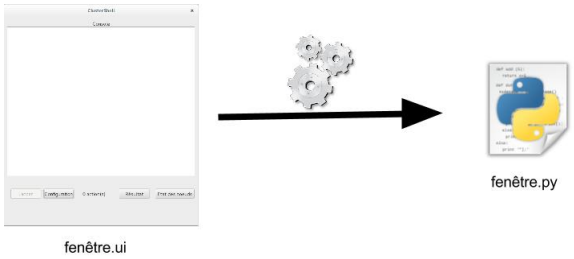



FIGURE 1 – Conversion ui - py

Pour convertir on utilise la commande suivante : **pyuic4 fenêtre.ui > fenêtre.py**

### 4.2.1 Les signaux et les slots

Pour de la programmation événementielle, on utilise deux moyens qui sont propres à Qt : les signaux et les slots. Chaque composant graphique (comme un bouton) possède des signaux et des slots qui vont permettre d'intéragir avec d'autres composants et fonctions(exemple : ouvrir une fenête via un bouton).

**Un signal :** Un signal est un message envoyé par une classe lors du déclenchement d'un événement comme le clic sur un bouton
**Les slots :** Les slots sont tout simplement des fonctions qui seront déclenchés par les signaux. Les fonctions peuvent être créés par nous même ou cela peut être des fonctions propres à une classe de Qt( exemple : la fonction **quit** de **QApplication** qui quitte le programme.
Voici un petit exemple pour mieux comprendre :

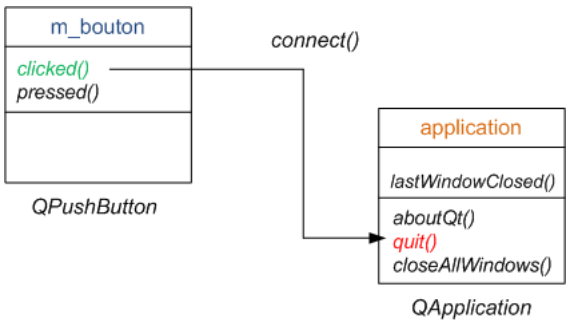
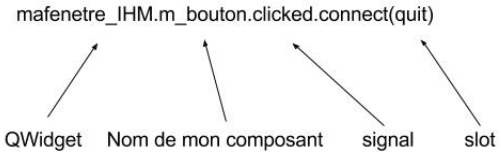



FIGURE 2 – Les signaux et slots

Pour pouvoir assigner un slot à un signal on doit utilise la fonction **connect** que l'on définit dans la classe de notre fenêtre :




### 4.3 Mise en place des résultats d'exécution dans des logs

## 5 Sources

**Nodeset :** `http://clustershell.readthedocs.io/en/latest/api/NodeSet.html`
**Task :** `http://clustershell.readthedocs.io/en/latest/api/Task.html`
**Qt GUI :** `http://doc.qt.io/qt-4.8/qtgui-module.html`
`https://pythonspot.com/en/pyqt4/`
`http://pyqt.sourceforge.net/Docs/PyQt4/qtgui.html`