




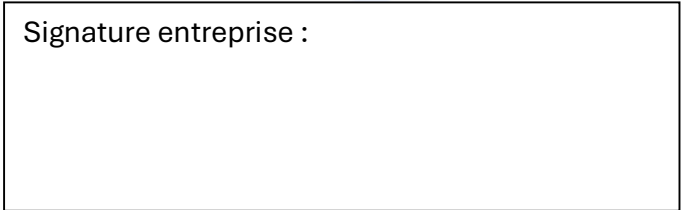
# Rapport d'alternance S6 :

## 5G software engineer



Manager : Pierre SAILLY  
Tuteur entreprise : Laurent ROTAT  
Tuteur pédagogique ING1 : Marwa HARZI

Signature entreprise :



Par Jacques Katunga Mukendi – ING1 – LSI1

## Table des matières

Table des matières .....	2
Remerciements .....	3
Résumé.....	3
Abstract.....	3
L'entreprise.....	4
Le contexte métier.....	4
Le besoin.....	4
Le positionnement de mon équipe dans l'organigramme .....	5
Composition de l'équipe .....	7
La mission.....	8
Les tâches .....	8
Phase principale.....	9
Contexte .....	9
Tâche 1 : Refactoring du code <b>SRunner</b> et Code review.....	9
Tâche 2 : Mise à jour automatique de fichiers de configuration au format JSON.....	12
Conclusion.....	19
Bibliographie .....	20
Annexe.....	21
Annexe 1 : Rack FSP + 3 FCTS .....	21
Annexe 2 : Les couches de la pile du protocole LTE .....	21
Annexe 3 : Les couches de la pile du protocole 5G .....	22
Table des illustrations.....	23

## Remerciements

Je tiens tout d'abord à remercier Monsieur Pierre Saily, team manager et Monsieur Laurent Rotat tech lead **SRunner** et maître d'apprentissage, de me donner la chance de prendre part à cette aventure qu'est l'alternance, dans un cadre aussi excellent. Un cadre qui a su dès mon arrivée bannir mes appréhensions et satisfaire mes attentes vis-à-vis de l'expérience alternante en plus de connaître une extrême montée en compétences, autant techniques que professionnelles. Je tiens également à remercier Monsieur Ahmed Maouche, Product owner, ainsi que l'ensemble de l'équipe pour leur soutien et leurs encouragements tout au long de l'élaboration des différents outils que j'ai développés cette année. J'espère que les années à venir seront tout aussi enrichissantes.

## Résumé

Le 11 octobre 2024, j'ai pris part à un entretien avec M. Saily et M. Rotat dans le siège de Nokia à Massy. Par suite de cet appel j'ai officiellement été invité le 4 novembre pour prendre mon poste. Malgré quelques difficultés techniques dues à mon identité numérique au sein de l'entreprise, j'ai pu rapidement faire connaissance avec les membres du bureau et me mettre à travailler. De plus, en décembre, j'ai pu rencontrer tous les managers et autres alternants autour d'un petit déjeuner de bienvenue qui nous avait été préparé.

Lors de mon arrivée, mon manager m'a fait une présentation de l'ordre hiérarchique de l'entreprise, ainsi que des différents éléments importants de l'environnement technologique dans lequel baignent tous les employés de Nokia, aussi appelés « **Nokians** ».

## Abstract

On October 11, 2024, I took part in an interview with Mr. Saily and Mr. Rotat at Nokia's headquarters located in Massy, France. Following this meeting, Mr. Saily officially called me to start my position on November the 4th. Despite some technical difficulties related to my digital identity within the company, I quickly got to know the office members and began working. Additionally, in December, I had the opportunity to meet all the managers and other apprentices during a welcome breakfast that had been organized by the former for us.

Upon my arrival, my manager introduced me to the hierarchical order of the company, as well as some key aspects of the technological environment surrounding all Nokia's employees, also known as « **Nokians** ».

# L'entreprise

## Le contexte métier



Figure 1 Frederik Idestam

On considère comme étant nouvelle, toute entreprise créée il y a 4 ans ou moins et toute entreprise existant depuis plus de 20 ans est considérée comme ancienne. Nokia du haut de ses 159 ans d'existence est une des entreprises ayant connu le plus de réformes. Frederik Idestam, ingénieur finlandais et fondateur de Nokia l'a créée premièrement pour faire de la papeterie, mais ce qui a réellement fait prendre son essor à Nokia est le tournant qu'elle a pris vers la téléphonie mobile, Nokia est connue pour avoir créé une de ses figures emblématiques, le 3310 en 2000. Aujourd'hui Nokia Corp est une entreprise qui se spécialise dans les réseaux de télécommunication.

En 2015 Nokia fusionne avec Alcatel Lucent et peu après déménage à Massy Palaiseau, Aujourd'hui l'effectif total de Nokia France est estimé à 2700 personnes d'après le quotidien français d'information Les Echos. Aujourd'hui Nokia est un des plus grands équipementiers en télécommunications aux côtés de Huawei et d'Ericsson.

## Le besoin

La station de base<sup>1</sup> 5G de Nokia est constituée d'une baie digitale avec des équipements analogiques, dont les antennes notamment. La baie digitale (ou rack) se compose d'1 carte contrôleur et de 1 à 3 cartes capacité. Ce sont ces racks qui viennent assurer et gérer l'interconnexion entre le flux de l'interface air (ondes radios) et le cœur de réseau d'un opérateur. Ils hébergent différentes fonctions tels que l'interconnexion sécurisée avec le cœur du réseau (fonction transport), la gestion des appels utilisateurs (L3), la gestion des ressources radio (L2) et la gestion de la couche physique digitale et analogique (L1). Le logiciel L2 est testé par les équipes de développement logiciel grâce à un programme propriétaire nommé **SRunner**<sup>2</sup>, qui se charge de l'installer sur différents TestBenchs<sup>3</sup>(racks en configuration complète : 1 carte contrôleur et 3 cartes capacité ( cf. annexe n°1 page 21 ). Je fais partie de l'équipe qui assure l'intégration continue de ce logiciel.

---

<sup>1</sup> Base Transceiver Station

<sup>2</sup> Logiciel gérant le lancement des tests du Logiciel L2PS

<sup>3</sup> Traduction = « Bancs de test »

## Le positionnement de mon équipe dans l'organigramme



Figure 2 Justin Hotard  
CEO de Nokia

Au premier avril 2025, il y'a eu un changement au niveau de la direction de Nokia, en Finlande. L'ancien CEO Pekka Lundmark, a légué son poste à Justin Hotard, nouveau Chairman.

Nokia est divisée en plusieurs secteurs. La division la plus importante, "Mobile Networks" (MN), est responsable du développement et de la commercialisation des réseaux d'accès (RAN) de la 2G à la 5G. Au sein de MN, l'unité MN RAN R&D est composée de six unités de développement distinctes, chacune spécialisée dans un domaine fonctionnel particulier. Je travaille dans l'unité qui gère le logiciel L2, responsable, entre autres, de l'attribution des ressources radio aux mobiles connectés au réseau. Cette unité compte environ 1000 ingénieurs répartis dans cinq pays (France, Allemagne, Pologne, Inde et Chine). Je fais partie du département français, situé à Massy (MSY), et plus précisément de l'équipe MN RAN RD L2 SW MSY numéro 3 dont la mission est de développer des outils de test pour l'ensemble de l'unité L2 dont **SRunner**. Il s'agit du logiciel propriétaire auquel je suis affecté pour toute la première année de mon apprentissage. Cette alternance se déroule sous la tutelle de M. Laurent Rotat, mon maître d'apprentissage mais aussi tech lead de notre équipe agile.



Figure 3 Organigramme mis à jour / Position de l'équipe SRunner

## Composition de l'équipe

L'équipe, au format agile est composée de :

Ahmed Maouche : Product owner, Senior 5G software Engineer

Laurent Rotat : Tech lead, Senior 5G software engineer

Przemyslaw Daniel : Senior 5G software engineer

S Abishek : DevOps Engineer

Moi : Apprentice 5G software engineer

Lorsqu'une modification est faite au repository général Nokia, elle ne peut être mergée que si au moins deux autres collègues ou « reviewers » ont validé les modifications sur Gerrit application spécialisée dans la revue de code. Il s'agit là en effet d'une de mes missions en tant qu'alternant à Nokia, prendre part aux code reviews et aux sprints, réunions bi-hebdomadaires. Malheureusement le format d'alternance de l'Efrei m'empêche de prendre part aux deux réunions.

Enfin, j'ai récemment été confronté à une autre réalité du monde professionnel, le départ de collègues. Il y'a un peu plus d'un mois, un des collègues avec qui j'ai eu la chance de travailler sur plusieurs tâches, a quitté ses fonctions récemment, ce qui m'a fait prendre conscience que l'entreprise reste un cadre professionnel avant tout et non pas un lieu tourné vers le social.

## La mission

Prendre part à une équipe de développeurs au format agile pour assurer le bon fonctionnement et l'amélioration continue de **SRunner**.

### Les tâches

#### 1) Phase de mise à niveau

Les tâches principales de cette phase sont :

- Prendre part aux meetings hebdomadaires / Me familiariser avec la nomenclature des équipements et les acronymes Nokia
- Faire du code review + Suivre les training Nokia (Git, Programmation python, valeurs de l'entreprise, sécurité informatique, robustesse et autres.)
- Me familiariser avec le fonctionnement de l'équipe ainsi que les process établis pour la livraison des codes (CI<sup>4</sup> avec Git, Gerrit, Zuul...) et assignation des tickets JIRA<sup>5</sup>.



Lundi 16/09 j'ai eu une visite du laboratoire Nokia situé à Nozay, j'ai pu voir le système permettant aux SW<sup>6</sup> engineers d'effectuer leurs tests de partout dans le monde.

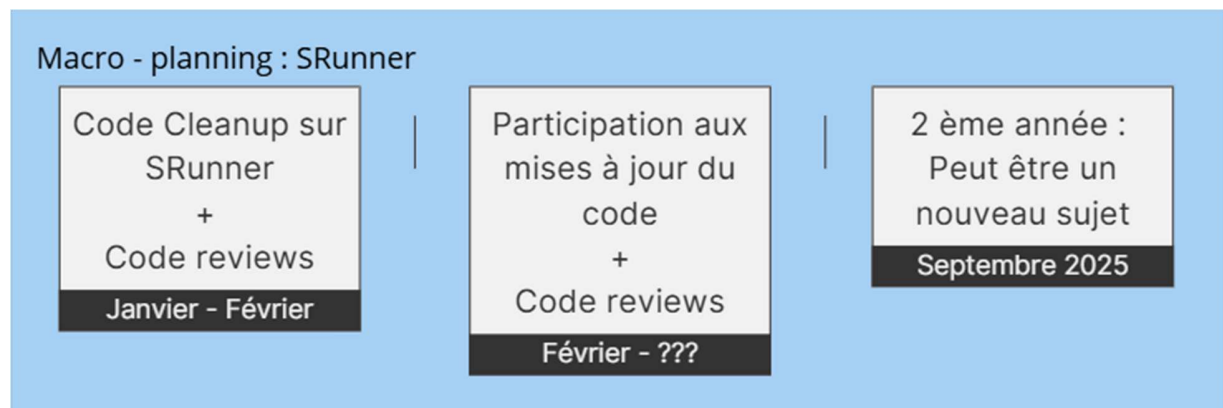


Figure 4 Macro-planning - Prévisionnel

Au cours de cette première phase, j'ai pris part à nombre de trainings proposés sur le portail de Nokia, notamment sur la OOP<sup>7</sup> en python. La validation de ce module m'a permis d'obtenir une certification en développement python.

<sup>4</sup> Continuous Integration

<sup>5</sup> JIRA : Outil de gestion de projets permettant la planification et le suivi des tâches

<sup>6</sup> SW = SoftWare

<sup>7</sup> Programmation orientée objet



## Phase principale

### Contexte

Pour assurer la sécurité des données, l'aisance de déploiement des outils et des dépendances, les développeurs se voient assignés une VM<sup>8</sup> ou machine virtuelle. Dessus ils peuvent copier le répertoire Git commun, nommé « GNB<sup>9</sup> ». Pour chaque modification ou chaque nouveau fichier, il y a un processus de validation scrupuleusement respecté, ce qui a pour but d'éviter que toute régression soit induite dans le répertoire commun. Ce qui souligne l'importance des trainings que j'ai suivis dans la première phase. Une fois les trainings complétés, j'ai pu cloner le répertoire commun sur ma VM pour commencer ma première tâche.

### Tâche 1 : Refactoring du code **SRunner** et Code review

Pour réaliser cette tâche j'ai dû faire face à deux défis qui m'ont permis de monter énormément en compétences.

Premièrement, l'utilisation d'un docker sur une VM, lors de ma formation à l'IUT de Cachan en GEII<sup>10</sup>, je n'ai jamais eu à utiliser de docker. 9 mois après le début de l'alternance, je continue d'apprendre, et je me rends compte d'à quel point savoir utiliser un environnement de type Linux (ou une suite GNU<sup>11</sup>), est une compétence primordiale à avoir pour tout ingénieur dans le secteur de la tech.

Deuxièmement, l'utilisation d'un autre IDE<sup>12</sup>, j'ai gagné en adaptabilité au travers de l'utilisation d'un logiciel différent de Vscode, IDE que j'ai utilisé tout au long de ma formation. Cependant peu importe l'IDE le principe reste le même, j'ai très vite pris mes marques en commençant par le débogage et l'utilisation de l'invite de commande intégrée. Grâce à ça, j'ai pu utiliser **SRunner** localement pour tester le bon fonctionnement de mes modifications.

---

<sup>8</sup> Virtual Machine

<sup>9</sup> GNodeB : Station de base des réseaux cellulaires de nouvelle génération ( 5G )

<sup>10</sup> Génie électrique et informatique industrielle

<sup>11</sup> Système d'exploitation de Unix constitué de logiciels libres

<sup>12</sup> Integrated Development Environment = Regroupement d'outils permettant le développement de logiciels

### Le besoin

Traditionnellement en Python, une fonction est définie et utilisée selon ce schéma →

Une fois définie elle doit être appelée grâce aux parenthèses qui invoquent la méthode

`__call__` de ma fonction. En l'occurrence, on passe grâce aux parenthèses deux arguments 3 et 4 ce qui permet d'avoir le résultat 7.

```
def fonction_addition(x, y):  
    return x+y  
  
fonction_addition(3, 4) # retournera 7
```

Figure 5 Fonction addition

Dans le cadre d'une classe, il est aussi possible de définir des méthodes selon ce schéma, cependant si on souhaite accéder à un attribut de la classe, donc une méthode « accesseur », il est plus judicieux d'utiliser les décorateurs de fonction. Les décorateurs ont été introduits avec Python 2.4. C'est une méthode qui permet de « wrapper » ou envelopper une fonction grâce à une autre. En l'occurrence le décorateur que j'ai utilisé pour cette mission est « `@property` » ainsi que « `@_setter` »

Prenons cette classe « Human » par exemple :

```
class Human:  
    def __init__(self, name: str,  
                  age: int,  
                  sex: str  
                  ):  
        self._name = name  
        self._age = age  
        self._sex = sex  
  
    @property  
    def name(self):  
        return self._name  
  
    @name.setter  
    def name(self, name: str):  
        self.name = name  
  
if __name__ == "__main__":  
    human1 = Human("Jacques", 21, "Male")  
    print(human1.name)  
    exit(0)
```

Figure 6 Classe Python: Human

L'utilisation du décorateur `@property` évite d'appeler explicitement la méthode `__call__` de « name », en autorisant son usage comme un attribut de classe. Cela implique que ce décorateur permet aussi de modifier la valeur de l'attribut via un setter associé, `@name.setter`, qui surcharge (overload) la fonction name.

C'est un outil puissant, car il renforce l'homogénéité du code, en simplifie la lecture et améliore sa compréhension tout en le rendant plus compact.

## Réalisation

Ma mission consistait à refactorer le code de **SRunner** en appliquant un décorateur destiné à transformer toutes les fonctions commençant par `get_` ou `is_`.

Pour ce faire, j'ai d'abord recensé l'ensemble des fonctions concernées à l'aide de l'outil `grep`, en combinant des expressions régulières telles que « `^def get_.*\(\self\):` » et « `^def is_.*\(\self\):` ». Ce filtrage m'a permis d'établir un inventaire des méthodes et de leurs emplacements.

J'ai ensuite procédé aux modifications, en veillant à ce que toutes les occurrences soient bien prises en compte, grâce à une double vérification avec `grep` et l'IDE CLion. L'ensemble des changements a ensuite été soumis à relecture via Git et Gerrit, comme le veut le processus de revue de code en équipe.

Chaque retour d'erreur ou modification demandée a été une opportunité d'apprentissage, en particulier sur l'usage avancé de Git. J'ai ainsi pu mieux maîtriser ses commandes et subtilités, c'est désormais un outil central dans mon environnement de travail chez Nokia.

En annexe, j'ai créé une page confluence<sup>13</sup>, logiciel collaboratif conçu pour faciliter le travail d'équipe. Cette page a pour but d'aider les développeurs de Nokia, à paramétrer leur environnement CLion pour utiliser **SRunner**. Cette initiative a été rendue possible par ma routine consistant à documenter régulièrement mes activités pendant l'alternance. Mon tuteur, m'a alors suggéré de partager ces notes au plus grand nombre. Une fois la page réalisée je l'ai soumise à mon manager. Après avoir pris en compte ses retours et obtenu son approbation, j'ai publié la version finale de mon guide, dans la même démarche, il m'est arrivé de venir en aide aux équipes de développements qui avaient besoin de renseignements ou de précisions sur **SRunner**. Renforçant ainsi ma maîtrise et connaissance de l'outil.

---

<sup>13</sup> Plateforme web de mise en commun

## Tâche 2 : Mise à jour automatique de fichiers de configuration au format JSON

### *Le besoin*

Il y'a plusieurs couches qui constituent la pile protocolaire du réseau 5G. Elles vont de la couche physique au cloud. Dans la couche physique, on trouve les stations de base, pilotées par les racks (cf. annexe 1 page 21 ). Ces racks sont les produits vendus par la division Mobile Networks de Nokia. Ils peuvent être déployés sous plusieurs configurations. En termes de Hardware, mais aussi software. En effet ils sont munis d'un certain nombre de cœurs, qui peuvent être alloués à différentes ressources logicielles selon le besoin, on appelle ces différentes configurations, des déploiements.

Une équipe a pour mission de mettre à jour ces déploiements en fonction des specs<sup>14</sup> stipulés par nos clients. Bien que SRunner permette de tester le logiciel de couche 2, le packet-scheduler<sup>15</sup> appelé « L2RT », il est incapable de simuler l'environnement en situation réelle. Il existe donc une différence entre la configuration utilisée lors des tests et celle en situation réelle pour un même déploiement.

Jusqu'à présent, notre équipe mettait à jour manuellement les données des fichiers de configuration. Chaque champ des fichiers JSON devait être saisi à la main. C'est en réponse à cette problématique que ma tâche a vu le jour. Un ticket JIRA m'a alors été assigné, accompagné des critères de validation suivants :

Revue Validée ( Gerrit )

N'induire aucune régression dans le répertoire commun

Pour organiser la tâche, j'ai créé un diagramme de Gantt prévisionnel ( cf. page 13 ). La tâche devait alors durer un mois avant d'atteindre l'étape finale.

---

<sup>14</sup> Spécifications matérielles

<sup>15</sup> Ordonnanceur de paquets, il décide comment et quand sont transmis les paquets de données de utilisateurs, pour cela il doit tourner à des vitesses extrêmement élevées, de l'ordre de 125µs pour traiter un paquet.

Vous trouverez en annexe un lien vers le fichier Excel correspondant.

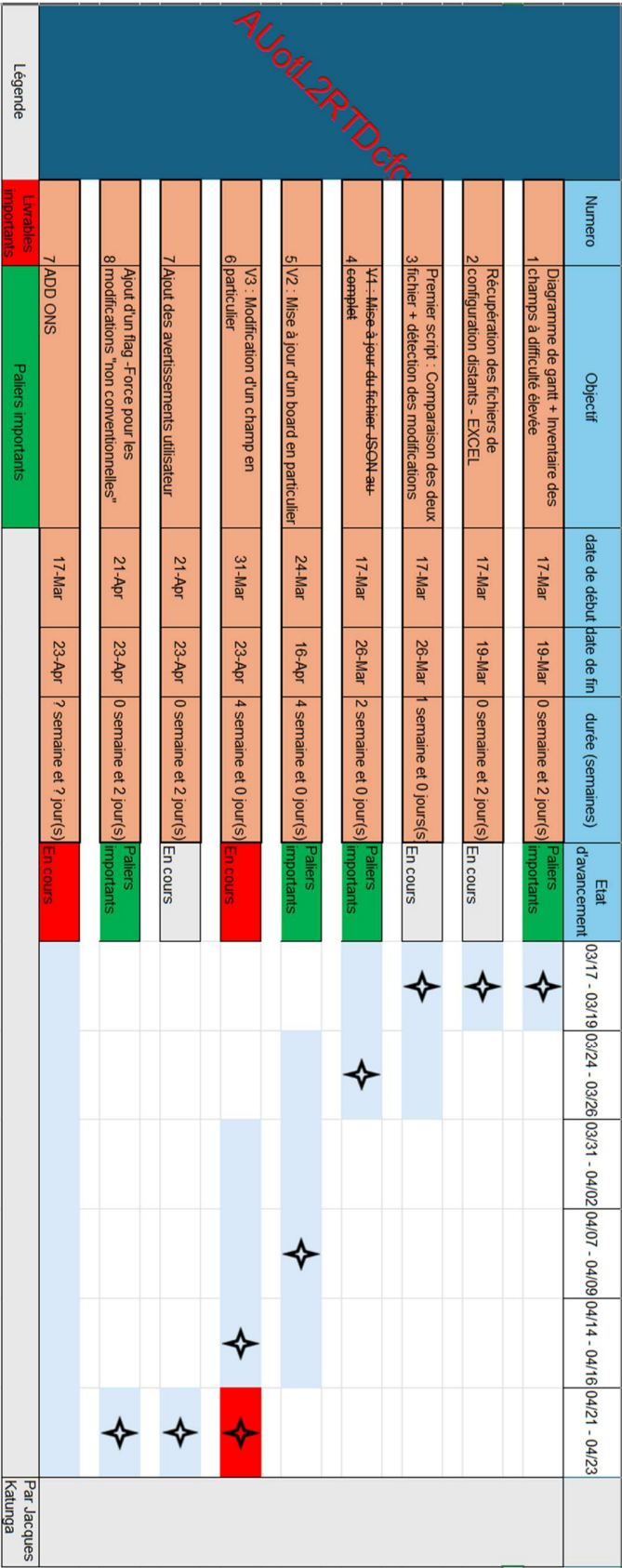


Figure 7 Diagramme de Gantt prévisionnel

## Réalisation

### 1) Etape préliminaire : étude des documents

Dans un premier temps, j'ai effectué une analyse comparative de la structure des fichiers de configuration distants, et des fichiers de configuration pour SRunner. Cela m'a permis d'établir cette table de correspondance :













Les champs situés à gauche correspondent au fichier JSON de SRunner, tandis que ceux de droite sont associés aux fichiers de déploiement client.	SRUNNER Files	Client files
	Runner file	Generated file from the deployment excel script [ABIO]
		
	l2ps	BoardSWDe ployment-...
	Legend :	
	-  = Data that needs pre-processing	
	-  = Data that requires deeper pre-processing	
	- First_type : second_type = Data that needs to be transformed ( probably a cast )	
	- Type : List = A list of "Type" elements	
	- first_Type -> second_Type : List = This is an element of type first_Type that will need to be transformed into a list with elements of the second_Type	
	- First_field : o Second_field = Means the data is located within another structure	
	TOP-LEVEL ENTITY = Boardname [string : LIST]	TOP-LEVEL ENTITY = Module NAME [String]
	 Second level entity = DEPLOYMENTID [str -> List {506,507,XXXX...}]	Second level entity SWDeploymentList [List{??? : ???} ]
	 Third level entity = Container instances [string -> Dict {1 : N}]	Third level entity [BoardDeploymentId]
	////////////////////////////////////	Fourth level entity = HugePageList [List]
	////////////////////////////////////	Fourth level entity = ContainerList [List]
	 boardUpPlaneContainerNumber[int]	BoardUpPlaneContainerNumber[int]
	containerAttributesType[string]	ContainerAttributesType[string]
	containerType[string]	ContainerList : ContainerType[string]
	cpuUnit[string]	DeviceList: -DeviceName[string]
	flavorId[int]	FlavorId[int]    also check FlavorName[string]
	 hugePageDataPlane: -HugePageNumber[int] -HugePageSize[string]	HugePageDataPlane: -HugePageSize[string] -HugePageNumber[int]
	 instanceNumber [int{0;1;N}]	LocalInstance[int : Depends on the container type, do not take the UPUE container types]
	 nrtCoreAffinityList[string -> List]	NRTCoreAffinityList[int : List]
	nrtCpuWeight[int]	NrtCpuWeight[int]
	 privateParameter[string -> Hexa]	ContainerPrivateParameter[string -> Hexa]
	cellTechnology[string : List]	CellTechno [string]
	 rtCoreAffinityList[string -> List]	RTCoreAffinityList[int : List]
	12 fields //////////////////////////////////////	////////////////////////////////////

Figure 8 Table de correspondance entre les fichiers de configuration

Ce fichier est un véritable sommaire ainsi qu'un guide de lecture. Il permet d'obtenir des informations précises sur les données trouvées dans chaque champ, leur nature, nombre mais aussi comment les récupérer et les traiter au sein de mon script. C'est une ressource qui assurera l'évolutivité de mon outil, même si je ne fais plus partie de l'équipe SRunner.

C'est en me basant sur ce fichier que j'ai créé la version 1 de mon outil.

## 2) Première version

Pour la première version de mon outil, le « L2RT\_deployment\_config\_update », codé en python, j'ai opté pour une approche séquentielle avec des fonctions.

### Fonctionnement :

- Les variables globales PROJECT\_DIR, DEPLOYMENT\_DIR et SRUNNER\_DIR permettent de rendre les chemins d'accès aux fichiers accessibles à toutes les fonctions.

```
PROJECT_DIR = Path(__file__).parent
DEPLOYMENT_DIR = input("""If deployment file is in current directory hit enter.
Otherwise paste the Deployment directory path: """)
os.system('cls' if os.name == 'nt' else 'clear')
SRUNNER_VIRTUAL_NODES_DIR = input("""If Srunner Json file is in current directory hit enter.
Otherwise paste the Srunner json config file directory path: """)

DEPLOYMENT_DIR = Path(DEPLOYMENT_DIR.strip()) if DEPLOYMENT_DIR != "" else PROJECT_DIR

if SRUNNER_VIRTUAL_NODES_DIR != "":
    SRUNNER_VIRTUAL_NODES_DIR = Path(SRUNNER_VIRTUAL_NODES_DIR.strip())
else:
    SRUNNER_VIRTUAL_NODES_DIR = PROJECT_DIR
os.system('cls' if os.name == 'nt' else 'clear')
```

Figure 9 Variables Globales de L2RT\_Json\_update

Une fois les chemins des fichiers récupérés, ceux-ci sont ouverts afin d'en créer une sauvegarde datée.

Ensuite, grâce au module Json de Python, les fichiers sont convertis en dictionnaires, ce qui facilite grandement l'accès et la modification de leurs données.

Lors de l'exécution du script, deux dictionnaires sont extraits : l'un correspondant au fichier de configuration pour les clients, l'autre au fichier de SRunner. Une fois ces structures obtenues, il devient possible de les comparer.



Chaque déploiement dispose d'un certain nombre d'instances de containers allouées au logiciel L2RT. Mon outil récupère cette valeur. Elle détermine le nombre d'itérations à effectuer dans la boucle pour parcourir chaque instance.

Le principe de modification est le suivant :

Dans le point d'entrée du programme<sup>16</sup>, j'ai développé mon propre parseur d'arguments, une fois les arguments parsés<sup>17</sup>, le module ainsi que le déploiement dont on veut mettre à jour la configuration sont extraits, puis passés à une méthode appelée « module\_and\_deployment(\*args) ».

Dans cette fonction, le champ qu'on souhaite modifier est recherché dans les deux dictionnaires. Les valeurs sont ensuite récupérées puis comparées, si les valeurs sont différentes, on met à jour celles dans le dictionnaire SRunner. Ce processus est répété pour chaque champ des deux dictionnaires, multiplié par le nombre d'instances de containers. Allant de 1 à 3 selon le déploiement.

La fonction « module\_and\_deployment(\*args) » a donc comme responsabilités :

- L'ouverture des fichiers
- La création des sauvegardes datées
- La conversion des fichiers en dictionnaires
- La récupération et la comparaison des données
- Enfin, La gestion des exceptions

#### Mode d'emploi :

./nom\_du\_script [MODULE] [DEPLOYMENT\_ID] [SOURCE\_DIR]

[Module] : un des modules supportés par SRunner ( Abio, Abin, Abil, etc.. )

[Deployment\_id] : Chaque Module possède un certain nombre de déploiements allant de quelques-uns à une dizaine.

[Source\_dir] : Chemin d'accès au répertoire git où se situent les fichiers de configuration

Dans cette version l'utilisation de l'outil se fait avec des paramètres positionnels.

---

<sup>16</sup> L'instruction if « \_\_name\_\_ == "\_\_main\_\_" : », il garantit que le script ne s'exécute pas s'il est importé en tant que module seulement

<sup>17</sup> Analysés, interprétés



### Revue du code :

Durant mes trois années à l'IUT de Cachan, je n'ai que très peu été formé à la programmation en Python. J'ai donc dû apprendre ce langage en grande partie en autodidacte. Cette lacune s'est rapidement fait sentir lors de la revue technique de mon outil. Il ne respectait ni les principes de la programmation "Pythonique", ni les bonnes pratiques en vigueur au sein de l'équipe SRunner.

Grâce aux conseils et commentaires des membres de l'équipe, j'ai pu améliorer significativement la qualité du code, mais aussi progresser en tant que développeur et futur ingénieur. Après avoir échangé avec mon tuteur pour identifier des axes d'amélioration, j'ai pu livrer une version 2 de l'outil, avec deux semaines de retard par rapport à mon diagramme de Gantt, mais avec un niveau de qualité bien supérieur.

### 3) Deuxième version

La première modification apportée par rapport à la version précédente concerne la nomenclature. Le nom du script a été modifié afin de respecter le principe de concision : il s'intitule désormais « L2RT\_Json\_update.py ». J'ai aussi appliqué ce principe à la nomenclature des variables et des méthodes du code.

### Améliorations :

Le principe de base reste le même. Une lecture séquentielle du fichier et la modification des champs après comparaison. Cependant l'implémentation est complètement revisitée. Notamment grâce à l'utilisation de méthodes de programmation plus modernes telles que :

- L'encapsulation (une classe par fichier) → accès aux champs grâce au décorateur @property
- Fonctions mono-responsables et courtes → respect du single responsibility principle
- Utilisation de modules préexistants dans SRunner (classes Parser et Module)
- Et surtout l'utilisation d'un Linter<sup>18</sup> ou pep checker

Ces modifications garantissent l'évolutivité de l'outil, facilitent la compréhension du code et le respect des principes de programmation « Pythonique ». Au vu de sa nouvelle structure, le code est bien plus élégant, et facile à relire mais surtout plus fonctionnel et maintenable.

---

<sup>18</sup> Outil d'analyse de code source, sans exécution, il permet d'assurer le bon respect des bonnes pratiques.

### Mode d'emploi :

L'utilisation du module `argparse`, via la classe `Parser`, a permis d'améliorer l'expérience utilisateur en remplaçant les arguments positionnels par des flags explicites. Cela offre une meilleure lisibilité en ligne de commande, et permet l'implémentation du flag `[-h]`, qui fournit automatiquement une aide contextuelle à l'utilisateur.

```
PS C:\Users\mukendi\OneDrive - Nokia\Python-course\Automatic update of the L2RT config\Jsonparsing> python .\L2RT_Json_update.py -h
usage: L2RT_Json_update.py [-h] -f FILE -m MODULE -i DEPLOYMENT_ID -s SOURCE_REPOSITORY [-g GNB_DIR]

Retrieve fields for L2RT containers from deployment JSON files.
The fields(i.e. container type, HugePage fields and container
disposition lists [L2RT, L2LO]) are updated from the file.
Color.boldINFO: Backup and Log files are created.

options:
  -h, --help                show this help message and exit
  -f FILE, --file FILE      l2ps_api5virtualnodes or l2lo_api5virtualnodes.
  -m MODULE, --module MODULE
                           Updated Board. Default is ABIO
  -i DEPLOYMENT_ID, --deployment-id DEPLOYMENT_ID
                           Numeric value of the deployment
  -s SOURCE_REPOSITORY, --source-repository SOURCE_REPOSITORY
                           Path to deployment repo directory
  -g GNB_DIR, --gnb-dir GNB_DIR
                           Path to gnb directory

Examples:
+ Update a specific board and board id:
  L2RT_Json_update.py -f FILENAME -b BOARD -i DEPLOYMENT_ID -s DEPLOYMENT_DIR
+ [UPCOMING] Update Every specified Board in Srunner
  L2RT_Json_update.py -B ALL -i ALL -s DEPLOYMENT_DIR
```

Figure 10 Aide contextuelle de `L2RT_Json_update`

De plus cette deuxième version supporte tous les fichiers de configuration contenu dans le répertoire `SRunner`. Ce qui évite les répétitions, en développant plusieurs outils. Une fois le script exécuté un fichier `log`<sup>19</sup> est généré. Il permet une traçabilité des éléments qui ont été modifiés lors de son exécution. Un garde-fou en cas d'erreur ou de modification non voulue.

### Revue du code et possibilités d'avancement :

En l'état, j'ai livré la Version 2 de l'outil, qui attend désormais l'approbation des autres membres de l'équipe pour être mergée dans le répertoire commun.

En annexe, j'ai travaillé sur un module qui a terme, sera intégré à `L2RT_Json_update`. Le module accèdera au dossier spécifié dans la variable globale `DEPLOYMENT_DIR` (cf. figure 9 page 15), vérifiera s'il s'agit d'un répertoire git où se trouvent les fichiers de configuration pour les clients. Le cas échéant, le module clonera automatiquement le répertoire pour tout de même poursuivre l'exécution de la mise à jour.

---

<sup>19</sup> Fichier contenant un historique des actions opérées par un système

## Conclusion

Cette première année d'alternance a été une véritable montée en compétences, aussi bien sur le plan technique que professionnel. Entre les défis rencontrés, les outils développés, et les échanges avec les différentes équipes, j'ai pu progressivement consolider ma place au sein de l'entreprise. Ma participation active aux revues de code et aux réunions d'équipe m'a permis de mieux appréhender l'environnement de travail et les protocoles internes, le tout dans un cadre accueillant où l'innovation et l'écoute sont mis à l'avant.

Sur le plan technique, j'ai énormément développé mes compétences en Python et en Bash, mais aussi dans la maîtrise d'outils à usage professionnel comme Gerrit et Jira. Quant à mon aptitude à m'adapter aux attentes dans un cadre professionnel, elle s'est aussi accrue. En effet j'ai pris en confiance dans l'utilisation des différentes infrastructures mises en place. La réservation de salles pour les réunions ou encore l'utilisation des outils mis à disposition de tous. La page de garde ainsi que la mise en page de ce rapport en sont de parfaits exemples.

Je souhaitais à tout prix prendre part à une alternance qui me permettrait de concilier ma formation actuelle en ingénierie informatique avec mes compétences en GEII. Le fait d'avoir suivi un parcours similaire à celui de mon tuteur est un atout. Il a su faire confiance en ma capacité à comprendre les aspects techniques liés au matériel utilisé dans le cadre de notre travail. Cela se reflète dans la liberté qui m'est accordée dans le développement des outils. J'espère pouvoir fournir un travail de qualité sur la totalité des années qu'il me reste au sein de Nokia.

Merci,

Jacques Katunga Mukendi, Apprentice 5G SW engineer, Nokia.

## Bibliographie

Diagramme de Gantt prévisionnel : [Outil L2RT Json update](#)

Nokia France : [Nokia France](#) | [Nokia.com](#)

Communications LTE : [LTE Protocol Stack Layers](#)

Nombre d'employés à Nokia France : [Les salariés de Nokia France](#)

Nomenclature en télécoms : [Terminologie de la 5G](#)

Jira : [Outil de gestion de projets](#)

Gerrit : [Outil de revue de code](#)

OS de type Linux : [Des précisions sur GNU](#)

Python : [Le décorateur Python @property](#)

Grep : [Global Regular Expression Print](#)

Point d'entrée du programme : [Python main Guard](#)

Linter : [Outil d'analyse du code source](#)

Fichiers log : [Historique des opérations](#)

## Annexe

### Annexe 1 : Rack FSP + 3 FCTS

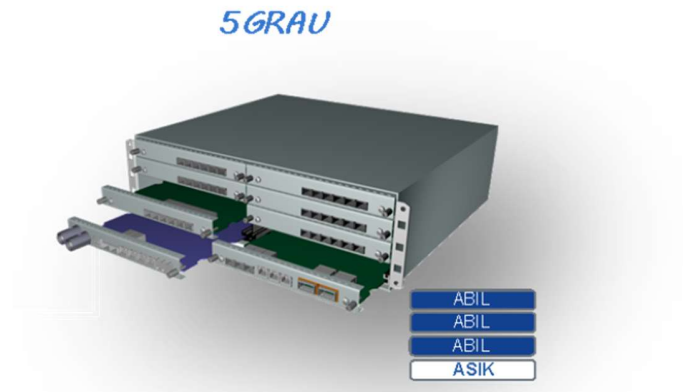


Figure 11 2 Racks (droite et gauche) le droit étant complet (1 FCT et 3 FSPs)

### Annexe 2 : Les couches de la pile du protocole LTE

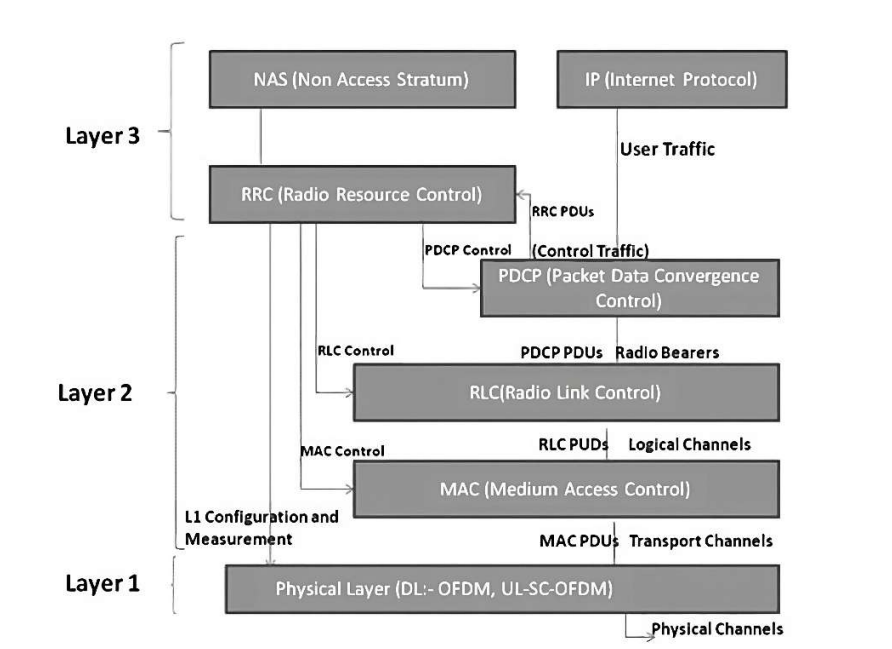


Figure 12 Source : [LTE Protocol Stack Layers](#)

## Annexe 3 : Les couches de la pile du protocole 5G

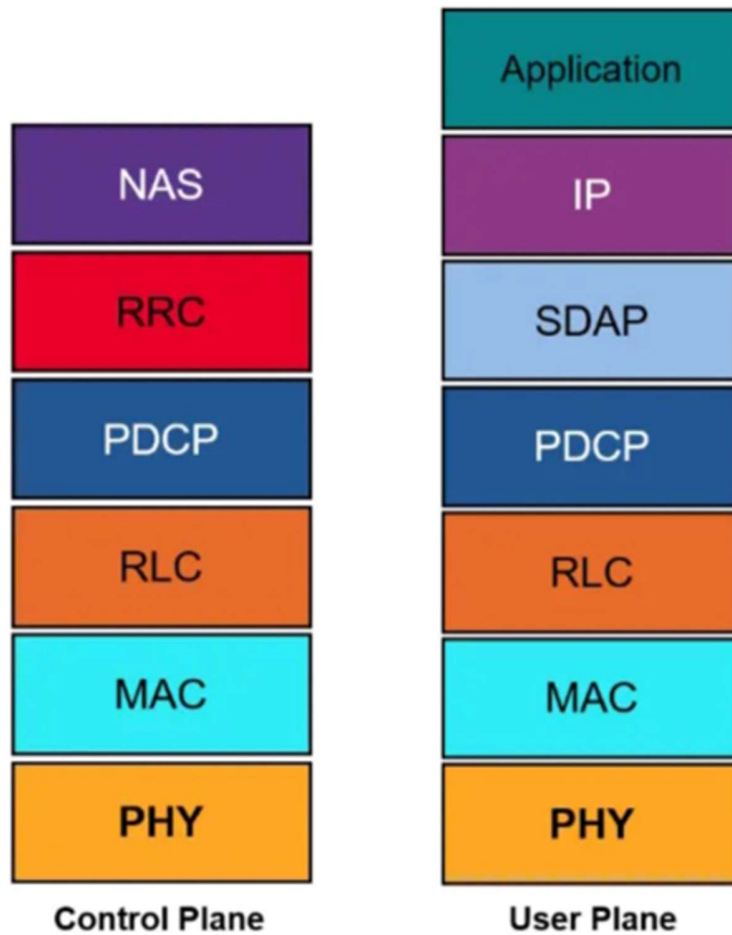


Figure 13 Source : [5G protocol stack Layers](#)

## Table des illustrations

Figure 1 Frederik Idestam .....	4
Figure 2 Justin Hotard, CEO de Nokia .....	5
Figure 3 Organigramme mis à jour / Position de l'équipe SRunner .....	6
Figure 4 Macro-planning - Prévisionnel .....	8
Figure 5 Fonction addition .....	10
Figure 6 Classe Python: Human .....	10
Figure 7 Diagramme de Gantt prévisionnel .....	13
Figure 8 Table de correspondance entre les fichiers de configuration .....	14
Figure 9 Variables Globales de L2RT_Json_update .....	15
Figure 10 Aide contextuelle de L2RT_Json_update .....	18
Figure 11 2 Racks (droite et gauche) le droit étant complet (1 FCT et 3 FSPs) .....	21
Figure 12 Source : LTE Protocol Stack Layers .....	21
Figure 13 Source : 5G protocol stack Layers .....	22

NOKIA