1 Cahier des charges fonctionnel

1.1 Contexte

Lorsqu'un banc de test est réalisé (par exemple pour mesurer des systèmes et/ou effectuer des mesures), des instruments de mesures standards sont utilisés (multimètres, oscilloscopes, générateurs de signaux, etc...). La quasi-totalité de ses instruments sont conçus avec des options permettant de les contrôler depuis un ordinateur et/ou d'autres appareils. Les solutions proposées par les fabricants sont souvent limitées ou très complexes et coûteuses. Des alternatives comme LabView permettent également le contrôle de ces appareils mais le coût est important et les capacités limitées. Ce logiciel est très adapté à des fins didactiques pour des utilisateurs novices (notamment à la programmation).

1.1.1 Opportunité

Les systèmes de communication utilisés pour piloter les instruments de mesure ont été standardisés au fur et à mesure des années. Les instruction qui peuvent être envoyées aux instruments afin de les piloter sont également disponible librement depuis le site internet du fabricant.
En parallèle, des langages de programmation haut-niveau comme Python et Matlab permettent de réaliser des traitement complexes de données complexes de manière simple et efficace.
Un système capable de relier ces deux milieux pourrait servir aux entreprises pour la création de bancs de tests, au privé pour effectuer de l'automatisation de mesure (et de tirer au mieux partis des capacités de leurs instruments) ainsi qu'aux écoles pour simplifier la prise de mesures (lors de laboratoires par exemples).

1.2 Objectif

Le système doit être capable, au moyen de code [1], de configurer un appareil, de lancer des mesures et de récupérer des valeurs.
Les valeurs récupérées doivent pouvoir être traitées facilement puis, si nécessaire, renvoyée au même ou à un autre appareil.
Le système doit également être capable de transférer des données de sources externes vers les appareils et inversement.

1.2.1 Documentation

Cette façon de procéder, qui s'approche de l'"infrastructure as code" [2], permet de décrire un fonctionnement physique par un code informatique, compacte et commentable. Ceci permet d'effectuer le test et sa documentation d'un seul coup. Cette approche permet égaler de recommencer une action à un moment ultérieur sans risquer d'oublier la séquence et/ou de se baser sur une documentation incomplète (le code est répétable).

1.2.2 Critère de performance

Le premier critère permettant de valider la performance du système est sa capacité à contrôler les appareils. A l'exception d'un simple test, il est nécessaire, de part la nature complexe de la communication entre ordinateur et appareils, d'effectuer des tests rigoureux sur une longue durée afin de valider le bon fonctionnement du système (pas de pertes de données, de commandes ratées, etc...).
Le deuxième critère est la facilité de mise en œuvre de nouvelles fonctions et/ou la capacité d'évolution du système. Ce critère devra être évalué lorsque le système est mis en place dans des cas réels et que de nouvelles fonctionnalités sont ajoutées. La réussite est prononcée lorsqu'il est possible de rajouter des fonctionnalités sans altérer des précédents tests (que les codes soient réutilisables malgré l'ajout de nouvelles fonctionnalités).

1. Python, Matlab ou autre
2. `https://en.wikipedia.org/wiki/Infrastructure_as_code`

1.3 Livrables

Les livrables sont sous la forme d'un programme (ou un ensemble de programmes) qui peuvent être installés sur un poste de travail. Une documentation est également livrée afin de permettre à l'utilisateur de débuter.