

# DémoJ

## Rapport final

### Participants

ADAM Arthur, BOUTINAUD Alexandre,  
DAUMAND Anaëlle, LAINÉ ODIC Yann,  
MOULHERAT Hadrien, PALVADEAU Olivier,  
PIAUD Charly, SALOMÉ Cameron, TELLACHE Yasmine

Avec l'encadrement de Olivier RIDOUX



Module projet  
du Master Ingénierie Logicielle et du Master Cloud et Réseau  
ISTIC  
Université de Rennes  
2023-2024

## Résumé

L'état de l'art de l'analyse des impacts énergétiques des systèmes informatiques distingue les impacts de fabrication des impacts d'utilisation, et parmi ceux-ci les impacts se répartissent en trois couches (trois *tiers* en anglais) : les équipements terminaux, réseaux, et serveurs. Les estimations des parts d'impact varient selon les hypothèses (essentiellement selon le périmètre envisagé), mais on convient ici que les impacts de fabrication et utilisation sont du même ordre de grandeur, et que les impacts de l'utilisation ont eux aussi les mêmes ordres de grandeur entre les trois couches. En conséquence, aucun ne peut être considéré négligeable par rapport aux autres. Le résultat est que les utilisateurs non initiés ne perçoivent facilement qu'une petite partie de l'impact énergétique global des systèmes informatiques : la part terminaux de la phase d'utilisation.

Nous proposons un démonstrateur qui facilitera la médiation pour faire comprendre les autres impacts énergétiques de l'utilisation d'un système informatique. Il consiste en un système informatique complet (terminal, réseau et serveur) qui expose de façon très visible l'impact énergétique de chacune de ses couches. L'utilisateur verra trois boîtes, une par couche, chacune équipée de jauge lumineuses pour afficher leur consommation électrique et leur température. Il aura devant lui ce qui est d'habitude réparti sur toute la planète. Ce dispositif permet d'aller au devant du public y compris dans des lieux non académiques, comme des parcs ou des bars ; des tiers-lieux. Le projet DémoJ est de réaliser un tel système.

Ce rapport présente la réalisation des différents blocs fonctionnels du système DémoJ, depuis l'infrastructure électronique commune à toutes les boîtes, dont les capteurs de consommation d'énergie et les afficheurs à base de leds, jusqu'à la programmation des scénarios de démonstration.

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Contexte . . . . .	1
1.2	Un support de médiation simple et dynamique . . . . .	2
1.3	Le projet DémoJ . . . . .	4
<b>2</b>	<b>Les blocs fonctionnels du système DémoJ</b>	<b>4</b>
2.1	Les boîtes . . . . .	5
2.2	L'interface utilisateur . . . . .	5
2.3	Le système électronique-informatique . . . . .	6
<b>3</b>	<b>Réalisation des blocs fonctionnels</b>	<b>8</b>
3.1	Les boîtes . . . . .	8
3.2	Le sous-système électronique . . . . .	11
3.3	Le sous-système informatique . . . . .	18
<b>4</b>	<b>Documentation utilisateur</b>	<b>23</b>
<b>5</b>	<b>Test et validation</b>	<b>23</b>
<b>6</b>	<b>Conclusion</b>	<b>24</b>

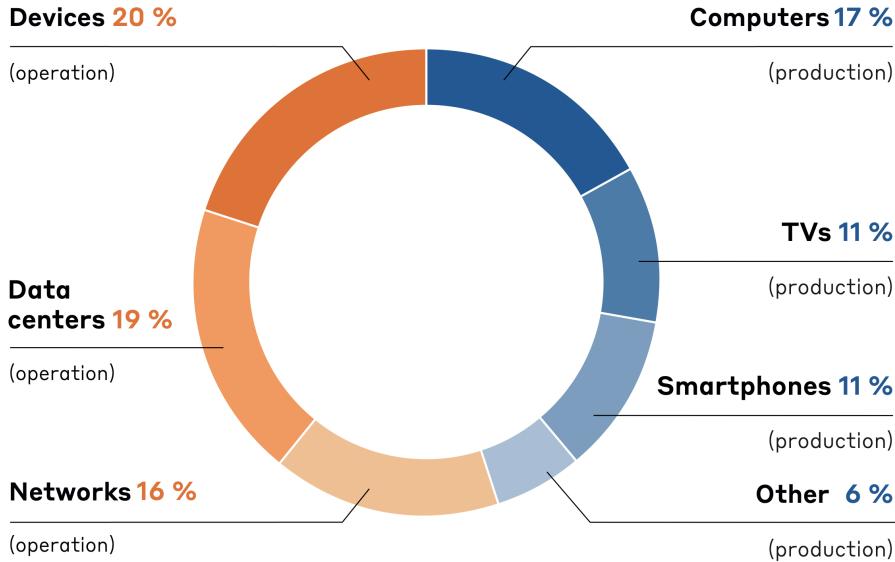


FIGURE 1 – Analyse des impacts énergétiques des systèmes numériques [10]. En bleu, les impacts de la production, en orange les impacts de l'utilisation.

# 1 Introduction

## 1.1 Contexte

L'Atlas du numérique [2] présente une étude de l'ADEME<sup>1</sup> de 2022 selon laquelle la consommation électrique du secteur du numérique en France représente 10 % de la consommation électrique française. Une valeur plus faible, environ 8 %, est retenue au niveau mondial, mais les 10 % sont largement dépassés dans d'autres pays. Par exemple, en Irlande les seuls data centres ont représenté en 2022 18 % de la consommation d'électricité du pays<sup>2</sup>. Il est donc important de comprendre et faire comprendre d'où viennent ces consommations d'électricité dont la plupart ne se font pas devant les utilisateurs finaux.

On distingue d'abord la consommation pour la fabrication et la consommation pour l'utilisation. Différentes sources, ex. [5] et [10], posent que les deux parts se valent globalement, mais en fait cela dépend beaucoup du périmètre des équipements considérés et même de leurs conditions d'utilisation. Par exemple, pour un smartphone en France (c-à-d. un appareil fabriqué en Chine, souvent en veille, et utilisé dans une région du monde où l'électricité est peu carbonnée), la part fabrication domine largement, environ 80 %, mais pour un serveur en Pologne (c-à-d. un appareil fabriqué en Chine, rarement en veille, et utilisé dans une région où l'électricité est très carbonnée) cela sera l'inverse. Le flou de la définition du secteur numérique ajoute un peu de confusion puisque certains équipements à l'utilisation très impactante comme les téléviseurs sont parfois inclus, parfois non.

Côté utilisation, on distingue trois couches (*tiers* en anglais) des systèmes informatiques : les équipements terminaux (appelés parfois *devices*, ex. dans la figure 1 empruntée à un rapport du Shift Project [10]), les équipements réseau et les équipements serveurs (lire par exemple [5, 10]). À nouveau, il y a beaucoup de flou sur les proportions des uns

1. ADEME : Agence de l'environnement et de la maîtrise de l'énergie, maintenant appelée Agence de la transition écologique

2. Central Statistics Office, <https://www.cso.ie/en/releasesandpublications/ep/p-mec/meteredelectricityconsumption2021/>

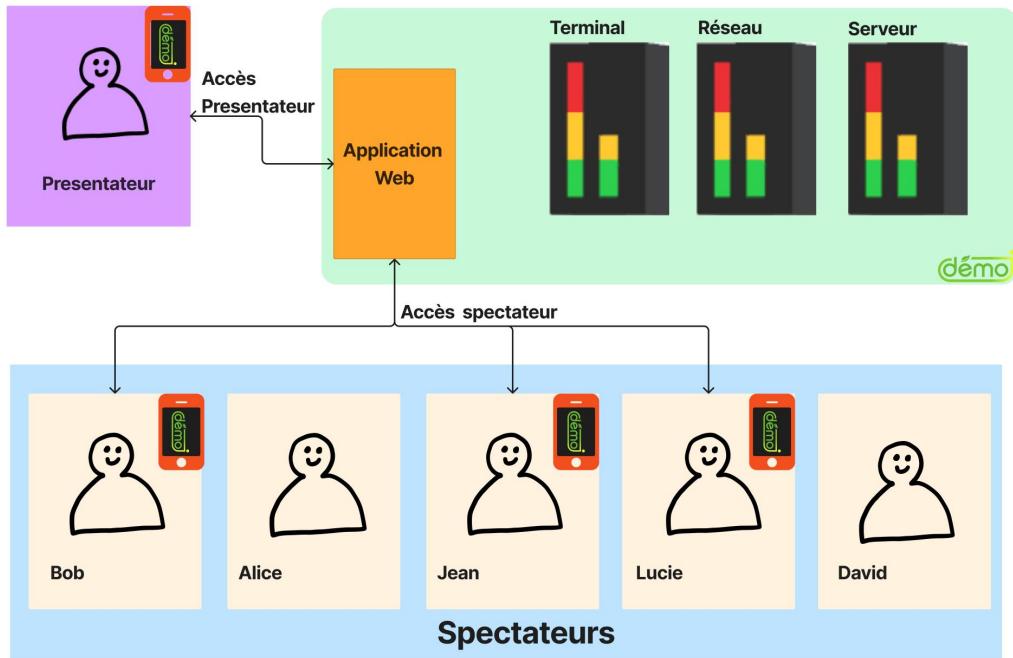


FIGURE 2 – Organisation générale d'une médiation

et des autres, et même sur l’attribution des équipements à une couche ou à une autre (ex. les *box internet* qui sont parfois considérées avec les équipements réseaux, et d’autres fois considérées avec les équipements serveurs), mais on peut dire que globalement les trois couches ont des impacts énergétiques d’ordres de grandeur comparables. La figure 1 représente une telle analyse.

Tout cela montre que l’impact énergétique des systèmes informatiques est à la fois important et difficile à cerner. Le grand public ne voyant que l’utilisation des équipements terminaux n’a une perception que d’un sixième environ de cet impact. Comment peut-il songer diminuer son impact sur les cinq sixièmes qu’il ne connaît pas ? Comment peut-il comprendre une réglementation contraignante qui s’appliquerait sur l’un des cinq sixièmes qu’il ne connaît pas mais qui se répercuterait sur lui ? Nous pensons que pour répondre à cela il faut concevoir des médiations simples mais dynamiques : simple pour éviter les problèmes d’attribution ou les difficultés d’interprétation ; dynamique pour ne pas se limiter à l’affichage d’une feuille de données, ou à un graphique, mais permettre de faire des expériences. De plus, ces médiations devront se garder de produire des chiffres absolus, et devront donc être plus qualitatives que quantitatives.

## 1.2 Un support de médiation simple et dynamique

Nous proposons pour cela de concevoir un démonstrateur des impacts énergétiques de l’utilisation des systèmes informatiques qui suit à la lettre le modèle à trois couches. Il sera lui-même un système électronique à trois couches que le packaging rendra très visibles, voire tangibles. On pourra toucher chacune des couches ! Chaque couche exposera visuellement l’énergie qu’elle consomme à l’aide de jauge lumineuses (voir la figure 2 pour une vue d’ensemble du dispositif, et en haut à droite pour les jauge).

Chaque couche jouera des scénarios typiques de son rôle, ex. servir des fichiers petits ou gros pour la couche serveur. Les répercussions énergétiques de ces scénarios se verront alors sur les autres couches. Un présentateur aura pour rôle de choisir et enchaîner les scénarios et expliquer les phénomènes observés (voir la figure 2 en haut à gauche). Nous espérons ainsi permettre une médiation plus active où les participants pourront jouer des scénarios exploratoires : que se passe-t-il si ... ?

Le système permettra aussi de faire participer les auditeurs qui le souhaitent via leurs smartphones dans le rôle d'équipement de la couche terminal (Bob, Jean et Lucie en bas de la figure 2). Cela permettra de démultiplier les équipements terminaux et de rendre visibles des scénarios où les effets cumulés de nombreux utilisateurs affectent les infrastructures. En effet, il y a en réalité beaucoup plus d'équipements terminaux que d'équipements réseaux ou de service. Selon une étude de l'association GreenIT [3], il y avait en 2019 plus de 30 milliards d'équipements terminaux, plus de 6 milliards si on ne compte que les équipements ayant un écran (les autres sont les objets connectés, souvent enfouis dans autre chose : ex. bâtiments, véhicules, appareils domestiques), pour plus de 1 milliard d'équipements réseaux, environ 200 millions si on se limite aux équipements de cœur de réseau (les autres sont les équipements de la périphérie du réseau : ex. *box* internet). Il y aurait des millions de datacentres mais de tailles très variées : du placard à la ferme de serveurs. Il n'y aurait que quelques milliers de gros datacentres qui hébergeraient en tout environ 70 millions de serveurs.

Le système devra pouvoir être utilisé dans des *tiers lieux* [7], y compris des tiers lieux éphémères comme des parcs ou des bars aménagés temporairement pour héberger une médiation. Le système devra être autonome électriquement et informatiquement ; il ne nécessitera donc ni connexion électrique ni connexion réseau. Il n'utilisera pas non plus les supports de médiation usuels qu'on peut trouver dans les espaces professionnels (les *seconds lieux*) ; par exemple, il pourra se passer de vidéo-projecteur. Il sera linguistiquement le plus neutre possible, et s'appuiera sur le moins possible de codification, et, quand il y en a, le plus souvent possible sur des codifications largement utilisées par ailleurs. Par exemple, l'affichage de l'intensité d'un phénomène en vert-orange-rouge, pour intensité faible, intensité moyenne et intensité élevée voire trop élevée, est largement utilisée, depuis l'affichage de l'intensité énergétique des habitations ou des appareils domestiques, jusqu'à l'affichage de l'intensité de sortie d'un amplificateur audio, en passant par les compte-tours des moteurs et le nutriscore.

L'énergie consommée par un système informatique se dissipant en chaleur, on s'attachera à rendre visible l'énergie électrique consommée instantanée (donc essentiellement une puissance), et la température du système. Formellement, ces deux mesures ne sont pas exactement des mesures de l'énergie consommée, mais elles y sont liées et il nous semble qu'elles sont assez concrètes pour l'auditeur non scientifique. Pour faire de la puissance instantanée une mesure de l'énergie, il faudrait l'intégrer par rapport au temps. Et la mesure de température introduit des phénomènes complexes comme son inertie, le fait qu'elle soit très peu uniforme dans le système, et le fait que pour s'en protéger les systèmes informatiques sont organisés pour évacuer la chaleur qu'ils produisent. On essaiera donc de mesurer la température à la source de la dissipation de chaleur et on prendra le risque de concevoir un système qui ne cherche pas trop à évacuer cette chaleur.

On appellera ce système **DémoJ**, J comme joule.

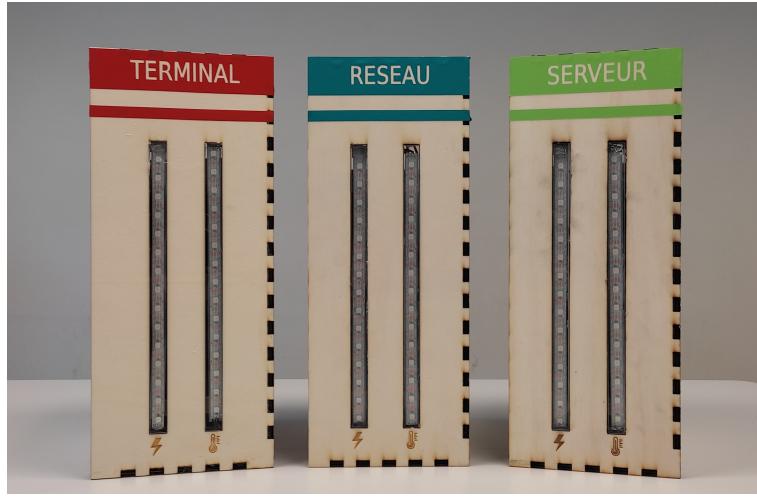


FIGURE 3 – Les trois boîtes du système **DémoJ**

### 1.3 Le projet **DémoJ**

Le projet **DémoJ** a été présenté au Collectif Numérique Responsable du Poool<sup>3 4</sup>, et y a rencontré un accueil favorable avec une invitation à présenter le projet au printemps 2024. Il a aussi été présenté au projet AIR de l'université de Rennes<sup>5</sup> dans la perspective d'une utilisation du système **DémoJ** dans les enseignements liés à la transition environnementale qui sont destinés à monter en puissance dans l'Université de Rennes. Il y a aussi rencontré un accueil favorable, et le projet AIR a pris en charge l'achat des fournitures non-électroniques.

Une première tentative de réaliser le système **DémoJ** a été faite en 2022-23, mais n'a pas abouti. Elle a cependant produit des analyses sur lesquelles nous nous sommes appuyés [8]. En particulier, nous reprenons l'idée de placer le projet **DémoJ** dans le contexte de la (pseudo-)entreprise DémoTech, dont le domaine d'activité est celui de la conception et la fabrication de supports techniques de médiation et d'enseignement. Le projet **DémoJ** est donc un projet de la société DémoTech.

## 2 Les blocs fonctionnels du système **DémoJ**

Cette section décrit la conception du système **DémoJ** sous forme de blocs fonctionnels. Pour bien comprendre l'organisation proposée, il faut se rappeler que le système **DémoJ** a deux sortes d'utilisateurs :

**L'opérateur [de la médiation]** : Il met en route le système **DémoJ**, puis active des scénarios de démonstration de différentes causes de consommation d'énergie ;

**Les auditeurs [de la médiation]** : Ils reçoivent les explications de l'opérateur de

---

3. Poool : successeur de la technopole Rennes Atalante pour le domaine du numérique, <https://lepoool.tech>. Porteur du label La French Tech du ministère de l'Économie, des Finances et de la Souveraineté industrielle et numérique : <https://lafrenchtech.gouv.fr>

4. Collectif Numérique Responsable : groupe de travail du Poool sur le thème du numérique responsable, <https://lepoool.tech/collectif-numerique-responsable/>

5. AIR : Augmenter les interactions à Rennes, <https://www.univ-rennes.fr/actualites/projet-air-experimenter-pour-transformer-grande-echelle-la-pedagogie-numerique>. Ce projet répond à l'appel à projet national « Démonstrateurs numériques dans l'Enseignement Supérieur ».

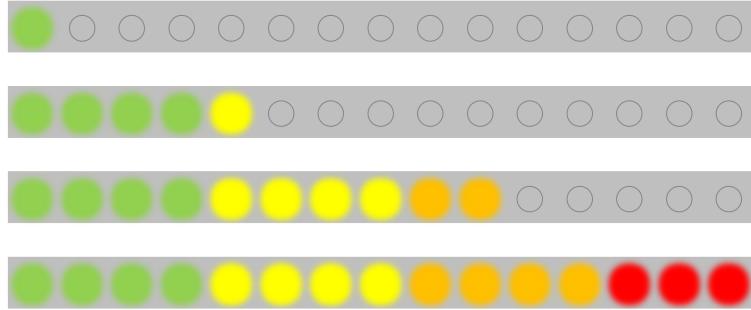


FIGURE 4 – Schéma des jauge **DémoJ**

la médiation. À leur convenance, ils sont passifs ou actif, selon qu'ils interagissent avec les scénarios ou non.

Vu de l'extérieur, le système **DémoJ** prend la forme de trois boîtes matérielles affichant chacune des jauge qui dévoilent leur consommation d'énergie. Rappelons que ces trois boîtes correspondent aux trois couches de l'analyse usuelle qui est faite de la consommation d'énergie due à l'utilisation des systèmes informatiques : les équipements terminaux, les équipements réseaux, et les équipements serveurs. À l'intérieur de chacune de ces boîtes un sous-système électronique-informatique permet de capturer et afficher les données énergétiques de la boîte, et un sous-système informatique permet de former un réseau entre les trois boîtes. Chaque boîte vue de sa dimension informatique est donc un noeud d'un système distribué. Chaque noeud permet aussi d'exécuter des applications spécifiques qui donnent à chacune de ces boîtes son rôle de terminal, réseau ou serveur. Et l'ensemble du système permet d'exécuter des scénarios représentatifs de ce qui se passe en vraie grandeur.

## 2.1 Les boîtes

Les boîtes du système **DémoJ** mesurent  $37,6 \times 14,8 \times 19,2$  cm (H×L×P). Elles sont équipées de deux fenêtres pour insérer les jauge, d'un port de charge, de deux boutons de contrôle et d'une poignée de manutention. Une des faces est amovible pour permettre le montage et la maintenance du sous-système électronique-informatique que chaque boîte contient.

Après quelques tâtonnements, nous avons choisi un format de parallélépipède vertical, inspiré d'une tour, doté de deux ouvertures verticales pour les jauge. Cette disposition offre une grande visibilité, et s'accorde bien avec le principe des jauge lumineuses. Celles-ci donneront l'impression de se remplir plus ou moins selon l'intensité énergétique des opérations exécutées. La figure 3 montre les boîtes prêtes à l'usage.

Ces boîtes hébergent tout l'équipement électronique-informatique de façon à le protéger tout en le laissant accessible, par exemple pour la maintenance, ou visible, dans le cas des jauge.

## 2.2 L'interface utilisateur

L'interface opérateur du système **DémoJ** se limite à ce qui permet de l'allumer et de l'éteindre et au contrôle des scénarios. Ce dernier contrôle se fait à l'aide d'une application web qui agit sur des variables du système **DémoJ**, et qui est contrôlée depuis un terminal de l'opérateur, par exemple son smartphone. L'interface de contrôle doit donc être très

portable, et pour cela nous optons pour une interface web qui s'exécute dans le navigateur du terminal de l'opérateur.

Les auditeurs peuvent aussi interagir avec le système **DémoJ**. Pour cela, une variante de l'interface de contrôle leur est mise à disposition. Cela permet de créer des situations plus réalistes que celle où il y a un unique équipement terminal pour un équipement serveur. Un système **DémoJ** avec une boîte serveur, une boîte réseau, une boîte terminal, plus une dizaine de smartphones des auditeurs constitue un système plus fidèle aux proportions du système mondial.

Le dernier élément d'interface vise les auditeurs. Il s'agit des jauge d'impact énergétique. Elles obéissent au format présenté dans la figure 4. Dans ce format, la longueur du segment allumé augmente avec l'intensité mesurée, et la couleur évolue aussi du vert au rouge avec l'intensité mesurée. Par défaut, une jauge affiche la consommation électrique instantanée, soit la puissance, et une autre affiche la température du processeur de chaque boîte, soit l'énergie dégradée en chaleur. Ces affichages sont sans unité, et doivent seulement être considérés dans leur dynamique. Tout ce qui est montré à l'auditeur est que telle action augmente ou diminue la consommation électrique, et qu'il s'en suit un échauffement plus ou moins critique de telle ou telle couche du système.

### 2.3 Le système électronique-informatique

Le système **DémoJ** est fondamentalement hybride. Il est composé d'une couche électronique qui offre capteurs, afficheurs et puissance de calcul, et d'une couche informatique qui contrôle le tout (voir figure 5). Chaque boîte contient un exemplaire d'une plate-forme électronique-informatique commune que nous avons conçue et réalisée, et des composants spécifiques, surtout logiciels, qui correspondent au rôle joué par chaque boîte. La plate-forme commune comprend une couche électronique et une couche système et réseau.

La couche électronique comporte

- une batterie pour l'alimentation (bloc bleu dans la figure 6, connectée au port de charge de la boîte),
- un système mono-carte, ou *single-board computer*, pour rendre chaque boîte programmable (bloc orange dans la figure 6),
- les capteurs, dont un wattmètre (bloc vert dans la figure 6),
- et des jauge d'affichage (bloc rose dans la figure 6).

Il n'y a pas de circuit de mesure de température visible car ce capteur existe déjà dans le système mono-carte utilisé.

Dans le but de disposer d'assez de puissance de calcul et de faciliter l'intégration des blocs fonctionnels électronique et informatique, nous utilisons un système mono-carte de type **Raspberry Pi** [6]. Chaque boîte contient donc une carte Raspberry Pi. L'avantage de ces cartes est qu'il est possible de les utiliser à la fois comme des ordinateurs standards et comme des composants électroniques. En effet, on peut y installer un système d'exploitation tel que **Raspberry OS** qui est une variante de Linux, et elles rendent accessibles des ports de connexion physique, les **ports GPIO**<sup>6</sup>. L'environnement de développement des composants logiciels spécifiques est donc celui d'une machine Linux standard. Et l'environnement de développement de la plate-forme commune est celui de l'électronique numérique. Cela permet de créer des circuits électroniques contrôlables par la carte Raspberry Pi. Dans notre cas, cela permet de réaliser facilement la mesure de l'énergie consommée et de gérer un affichage lumineux dynamique. La carte Raspberry

---

6. GPIO : General Purpose Input/Output

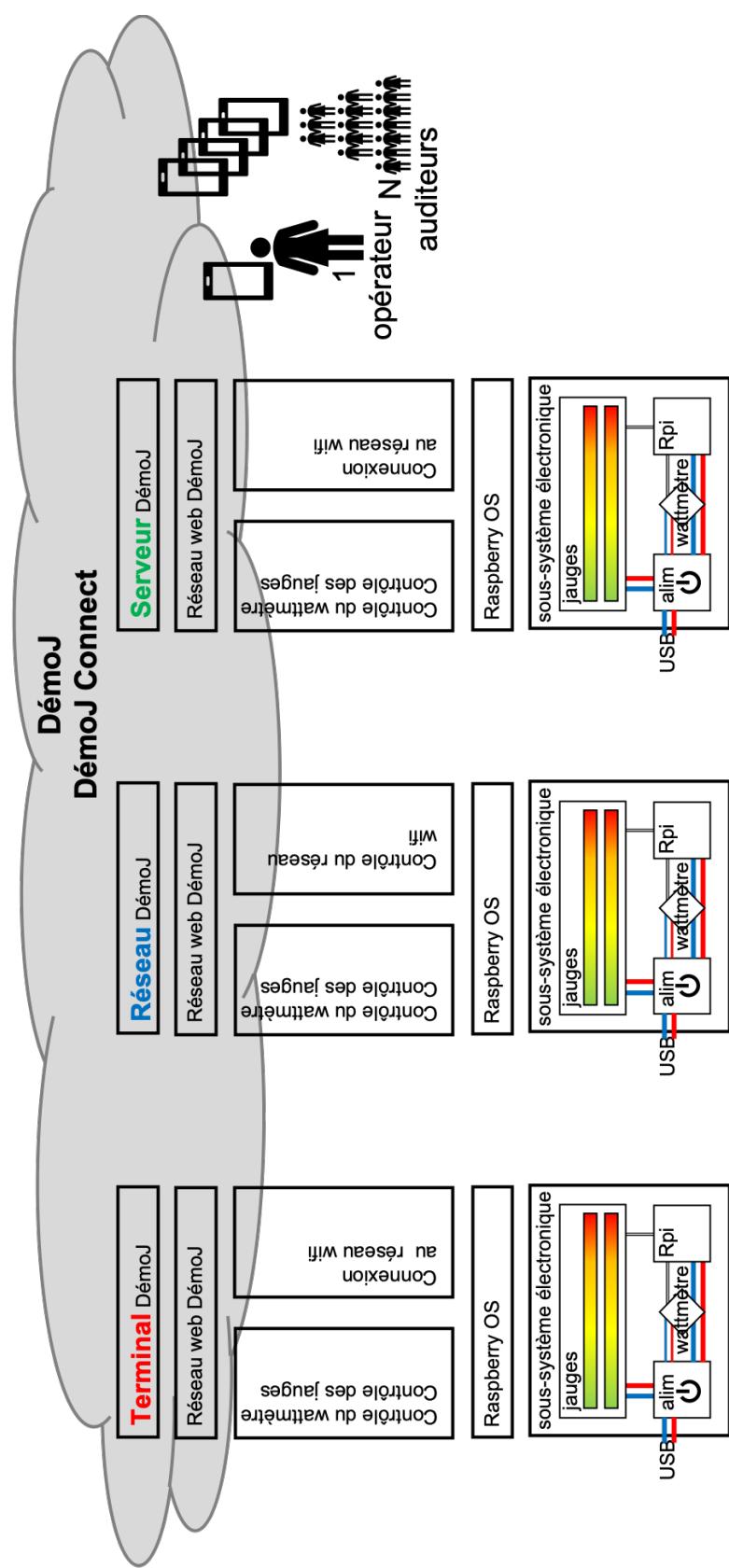


FIGURE 5 – Schéma synthétique du système **DémoJ**

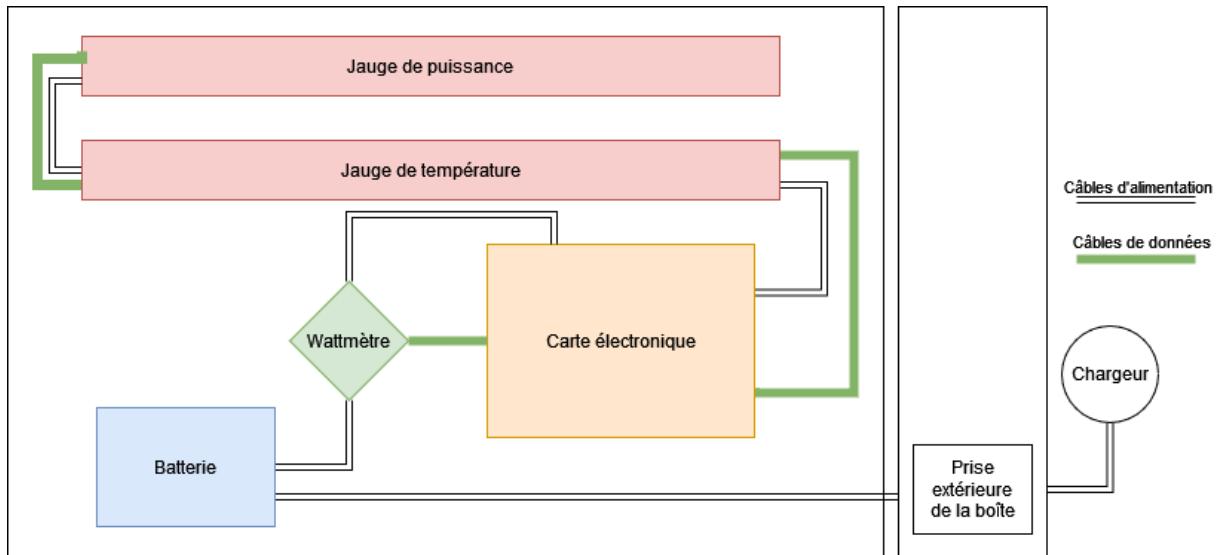


FIGURE 6 – Schéma synthétique de l'électronique des boîtes

Pi est alimentée par la batterie, et contrôle sans les alimenter le wattmètre et les jauge. Ceux-ci sont alimentés par la même batterie sur un circuit distinct.

La couche système et réseau comporte de quoi faire fonctionner un réseau wifi adhoc qui réunit les trois boîtes, plus un terminal pour l'opérateur animateur de la démonstration, et éventuellement des terminaux opérés par les auditeurs.

Les composants spécifiques correspondent aux rôles d'équipement terminal, ex. un navigateur HTTP, d'équipement réseau, ex. acheminer les requêtes et les réponses entre les équipements terminaux et serveurs, et le rôle d'équipement de service, ex. un serveur HTTP. Ces rôles sont encapsulés dans des scénarios qui constituent les briques de base de la médiation. Ils sont contrôlés à distance par une application qui permet à l'opérateur, et aux auditeurs qui le souhaitent, de les piloter sans en connaître les détails de la réalisation. Par exemple, un utilisateur pourra commander un scénario de téléchargement de fichier sans écrire des `curl http://3.127.0.0:152/....`, tout en pouvant spécifier de télécharger une version compressée ou non du document. Nous appelons cette application **DémoJ Connect**.

### 3 Réalisation des blocs fonctionnels

Cette section décrit la réalisation des blocs fonctionnels décrits dans la section 2. Ils ont été développés séparément et ont été composés entre eux progressivement.

#### 3.1 Les boîtes

Elles ont été réalisées en carton puis en bois au Fablab de l'université de Rennes où nous avons reçu l'assistance de Mme Camille Bisson, manager du Fablab. Elles sont constituées de pièces de bois, dont les plans ont été conçus à l'aide du logiciel **Inkscape**<sup>7</sup> et découpées numériquement à l'aide d'une découpeuse laser **Trotec Speedy100R**.

Les marquages génériques, c-à-d. communs à toutes les boîtes, sont réalisés par gravure dans le bois. Cela concerne l'étiquetage des jauge, des interrupteurs, des logos, et du

7. Inkscape : <https://inkscape.org/fr/>

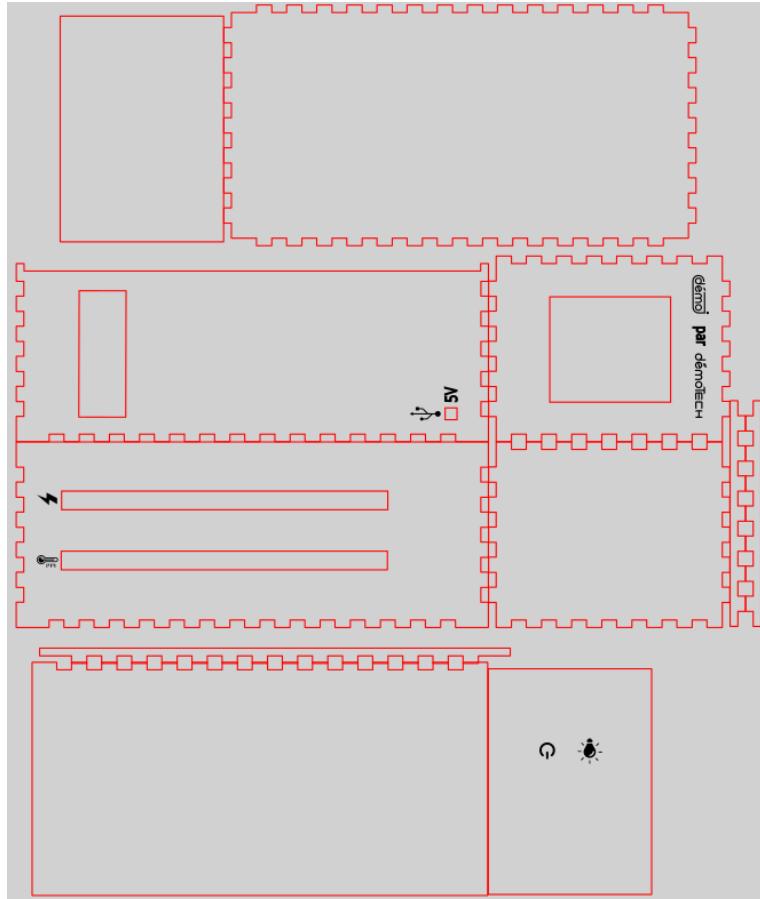


FIGURE 7 – Le plan des boîtes

port de charge. Le rôle spécifique de chaque boîte est identifié par des bandes de sticker, découpées par une découpeuse à stickers **Silhouette Cameo**. Ceux-ci identifient le rôle des boîtes par un nom en toute lettre et par des couleurs distinctives : rouge pour la boîte terminal, bleu pour la boîte réseau et vert pour la boîte serveur. Les stickers sont aussi conçus et découpés numériquement.

Les boîtes ont d'abord été prototypées (voir figure 8) avec du carton, mais en utilisant les mêmes outils que pour la fabrication finale en bois. Ces prototypes ont permis de se familiariser avec les outils de conception et fabrication numérique du fablab. Ces outils incluent :

- une découpeuse laser (voir figure 10), qui est une machine qui s'interface comme une imprimante mais qui grave ou découpe un matériau en fonction des traits (épaisseur et couleur) du fichier imprimé ;
- une découpeuse à stickers (voir figure 12), qui fonctionne comme la découpeuse laser mais avec des feuilles adaptées ;
- le logiciel gratuit Inkscape, un logiciel graphique permettant de manipuler des images matricielles ou vectorielles pour avoir le maximum de précision sur le plan de découpe.

Ce prototype nous a aussi appris quelques contraintes matérielles : le temps de découpe et le gabarit de la découpeuse laser qui ne permet pas d'envisager des pièces trop grandes.

- Le processus de découpe (ou gravure) est assez long ; la découpe des pièces du prototype en carton a pris au total plus de 30 minutes, tandis que les boîtes finales en bois ont pris environ 80 minutes chacune.

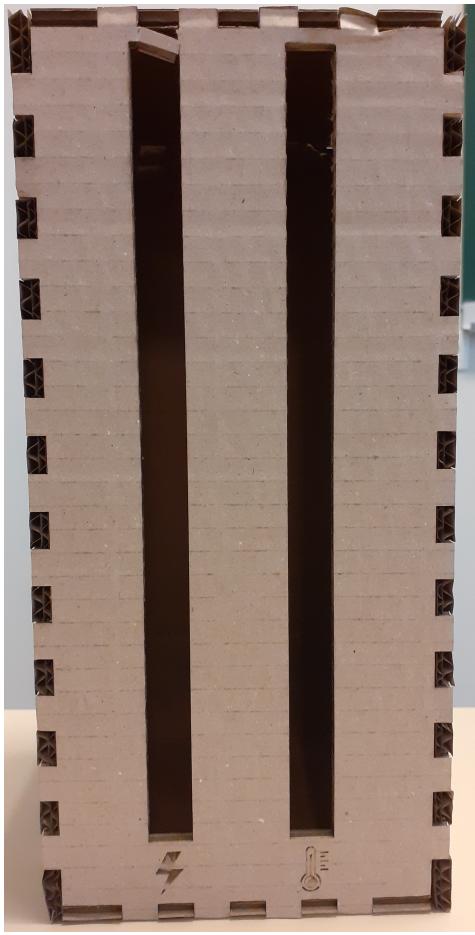


FIGURE 8 – Premier prototype de boîte



FIGURE 9 – Boîte finie

- La machine limite les dimensions des pièces qu'elle peut usiner ; les pièces ne peuvent dépasser 60 cm de longueur, 30 cm de largeur, et l'épaisseur maximale dépend du type du matériau utilisé et de l'action effectuée (gravure ou découpe). Les boîtes finales ont été réalisées en contre-plaquée de 6 mm d'épaisseur.

Les plans de découpe de chaque pièce ont été générés par l'outil de conception en ligne **festi.info**<sup>8</sup>. Cet outil permet de concevoir toutes les pièces, mais pas leur disposition pour la découpe. Pour gérer la disposition les modèles de pièces sont exportés de l'outil festi.info puis composés en utilisant le logiciel Inkscape. C'est à ce moment qu'on se préoccupe de la disposition la plus économique possible des pièces sur une planche (voir figure 11).

Par ailleurs, nous avons mis au point un système de glissière pour ouvrir la boîte, ce qui permet d'accéder facilement à son contenu s'il y en a le besoin. Ce système permet aussi une fermeture assez solide pour empêcher les ouvertures accidentelles, tout en évitant d'avoir des morceaux gênants dépassant de la boîte.

Enfin, les fenêtres des jauge sont fermées à l'aide d'une bande de plexiglas ajustée aux bonnes dimensions. De cette façon, les bandes de leds qui forment les jauge sont visibles de l'extérieur, mais ne sont pas exposées aux agressions externes.

---

8. festi.info : <https://festi.info/boxes.py>



FIGURE 10 – La découpeuse laser

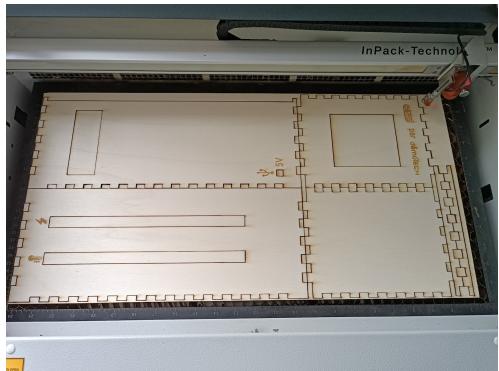


FIGURE 11 – La plaque de bois découpée, dans la découpeuse



FIGURE 12 – La découpeuse à stickers, avec une feuille chargée pour impression

### 3.2 Le sous-système électronique

Le système électronique a été élaboré en utilisant les ressources de l'atelier électronique de l'ISTIC, avec l'assistance de Regis Legave, gestionnaire de l'atelier. Pour ce faire, nous avons dû planifier l'acquisition du matériel nécessaire à sa réalisation. Plusieurs problématiques se sont présentées, notamment le choix des bandeaux de leds, l'acquisition des connectiques nécessaires pour relier les composants, la sélection de matériel compatible avec les dimensions de nos boîtes, et enfin, le choix d'une batterie appropriée.

La question de la batterie s'est avérée être la plus délicate à résoudre. En effet, nous devions nous assurer d'avoir suffisamment de puissance pour alimenter le Raspberry Pi (fonctionnant à 5 volts) ainsi que les bandeaux de leds (requérant au maximum 60 milliampères par led unitaire). Nous avons opté pour une batterie de 10 000 mAh dotée de plusieurs ports de sortie de 5 V. Toutefois, il était essentiel de vérifier que ces ports de 5 V étaient indépendants les uns des autres, plutôt que partagés. Pour répondre à cette question, nous avons réalisé des expériences à l'atelier.

Les résultats de ces expériences confirment que les ports de sorties de la batterie étaient effectivement indépendants les uns des autres. Ainsi, chaque port de 5 V était capable de fournir une alimentation électrique séparée et distincte, répondant ainsi à nos besoins pour alimenter le Raspberry Pi et les bandeaux de leds de manières adéquate.

Finalement, le système électronique est organisé autour des composants suivants :

- batterie externe USB 57976 10000 mAh (bloc bleu dans la figure 13) ;
- carte Raspberry Pi 4 B - 2 GB (bloc orange) ;
- ruban NeoPixel RGB 1 m, 60 leds programmables ADA1138 (bloc rose) ;

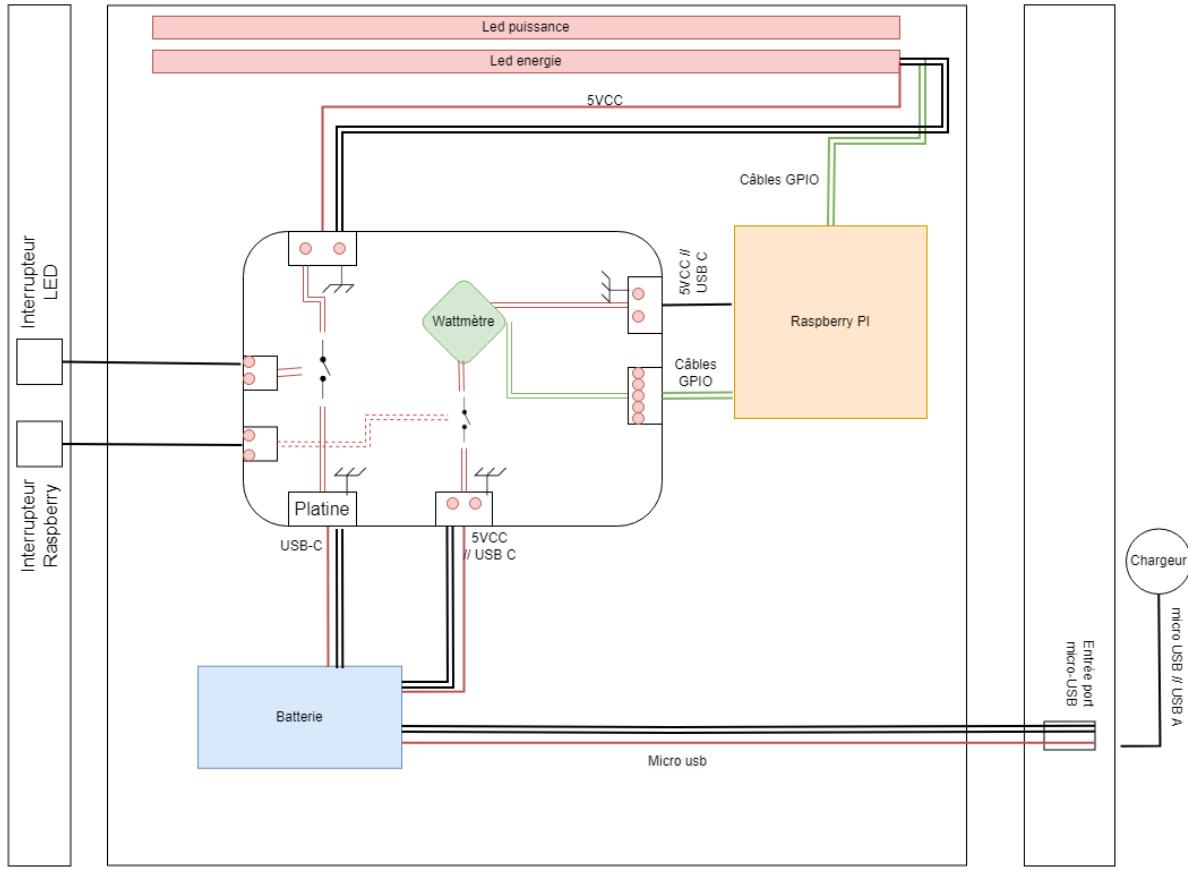


FIGURE 13 – Le circuit électronique

— module wattmètre I<sub>2</sub>C Gravity SEN0291 (bloc vert).

### 3.2.1 L’interface électronique

Nous pensions au début connecter l’ensemble des composants sur une carte de prototypage, style *breadboard*, mais Régis Legave nous a convaincu qu’il était préférable de concevoir un circuit imprimé spécifique : l’interface interne du système **DémoJ** (le bloc central blanc dans la figure 13).

**Le concept d’interface** Le système électrique que nous avons élaboré contient plusieurs circuits de natures différentes : l’alimentation des leds par la batterie, la commande des leds par le Raspberry, l’alimentation et la commande du wattmètre par le Raspberry, l’alimentation du Raspberry Pi par la batterie via le wattmètre, le tout contrôlé par des interrupteurs. Toutes ces connexions nécessitent, notamment pour l’alimentation, des adaptateurs pour aller d’un composant à un autre car les branchements ne sont pas les mêmes en sortie de batterie, en entrée et sortie du wattmètre, en entrée du Raspberry Pi, etc. Le circuit d’interface permet de consolider ces connexions en évitant un enchevêtrement des câbles. Nous avions aussi observé lors du prototypage de la partie électrique que des connexions adhoc étaient sensibles aux secousses et au déplacement du système **DémoJ**. Réaliser l’interface interne des composants du système **DémoJ** sous la forme d’un circuit imprimé résout ces difficultés.

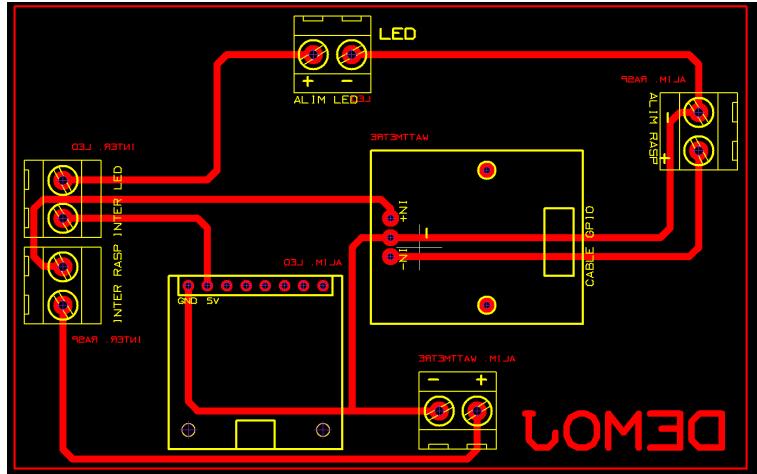


FIGURE 14 – Le schéma de l’interface réalisé avec le logiciel TCI

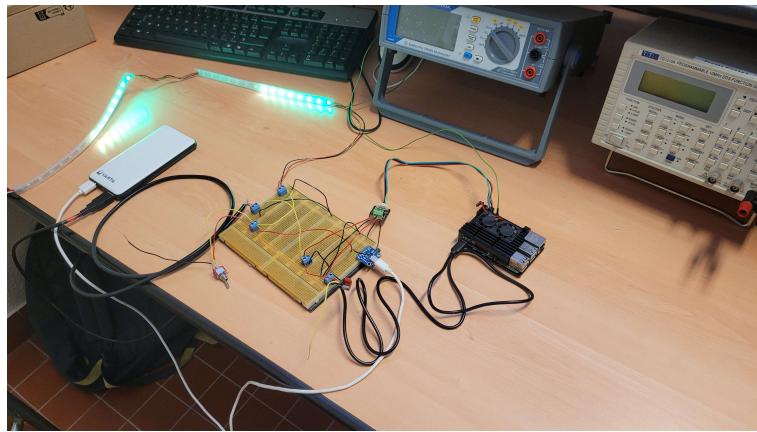


FIGURE 15 – Test du prototype de l’interface avant son impression

**Procédé de fabrication de l’interface** Dans un premier temps, nous avons conçu un schéma de l’interface sur le logiciel TCI<sup>9</sup> (voir la figure 14). Avant de passer à l’impression du circuit, nous en avons fait un prototype sur une *breadboard* comme on peut le voir sur la figure 15. Nous l’avons alors testé et validé, c’est à dire vérifier que celui-ci correspond complètement au schéma que nous avons réalisé et qu’il réalise la fonction attendue.

Nous avons eu la chance de pouvoir participer à l’impression de notre circuit. Le procédé d’impression s’apparente à un tirage photographique :

**Le négatif :** Avant l’impression on a préparé un calque du circuit à l’échelle réelle sur une feuille translucide comme sur la figure 16.

**La surface sensible :** L’impression se fait sur une plaque composée de résine dure et qui a une face recouverte d’un film de cuivre lui-même recouvert d’une résine photosensible, c-à-d. qui change d’état quand elle est exposée à la lumière.

**L’insolation :** On dispose ensuite le calque contre la face cuivrée de la plaque et on place l’ensemble dans une machine à rayons UV avec aspiration d’air pendant 2 minutes. Ceci a pour effet de modifier la structure de la résine photosensible là où elle n’est pas masquée par le calque. L’aspiration d’air permet de bien appliquer le calque contre la plaque.

---

9. TCI : Logiciel français de conception de circuits imprimés édité par Bruno Urbani.

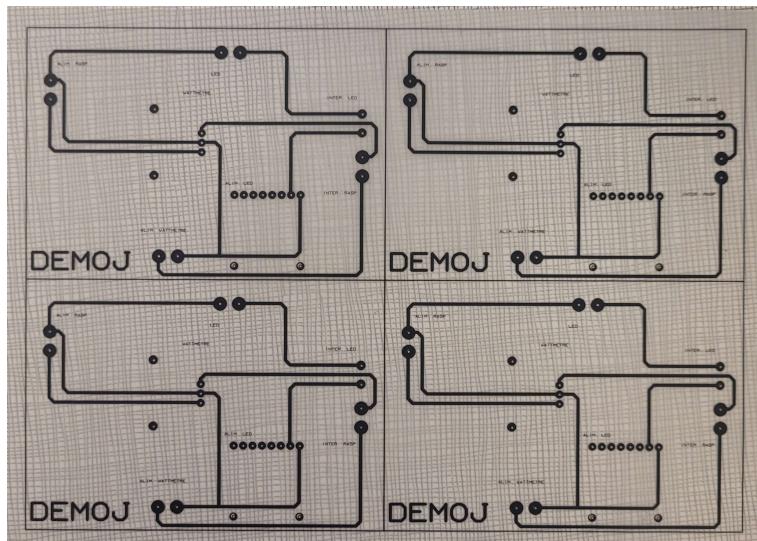


FIGURE 16 – Le calque du circuit pour l'impression

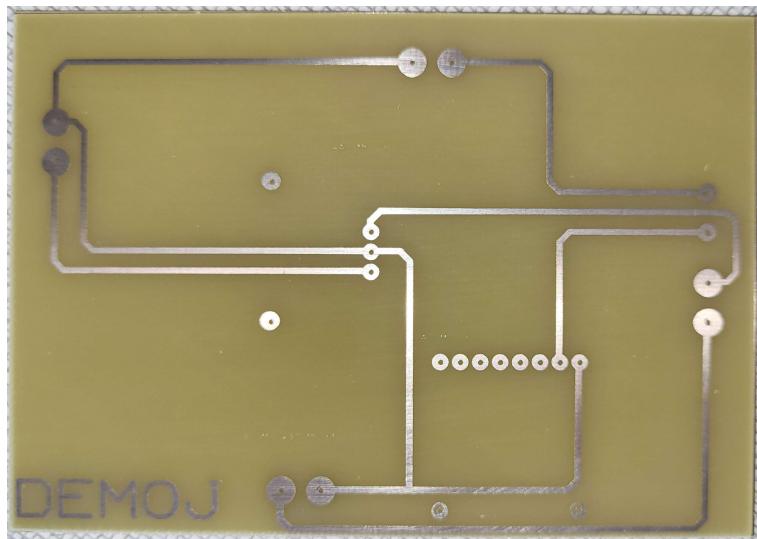


FIGURE 17 – Résultat de l'impression de l'interface

**La révélation :** On trempe la plaque quelques minutes dans un révélateur<sup>10</sup> qui dissout la résine photosensible qui a été exposée aux UV. Le cuivre qui était dessous ces zones est donc mis à nu.

**Élimination du cuivre mis à nu :** La plaque est soumise à un autre solvant, l'acide chlorhydrique, qui ronge le cuivre mis à nu par la révélation, à la suite de quoi il ne reste que la résine dure de la plaque et le cuivre des pistes du circuit sous une couche de résine photosensible intacte.

**Nettoyage :** De l'éthanol est utilisé pour enlever tous les résidus de cuivre, de solvant et de résine photosensible.

**Étamage des pistes :** Les pistes sont ensuite protégées contre l'oxydation en trempant la carte dans de l'étain chimique à froid<sup>11</sup>.

10. Révélateur : hydroxyde de sodium anhydre ( $\text{NaOH}$ ), ou soude caustique. Même produit que pour un débouche-évier.

11. Étain chimique à froid : une solution de sels d'étain dans de l'acide sulfurique ( $\text{H}_2\text{SO}_4$ ). Dans le

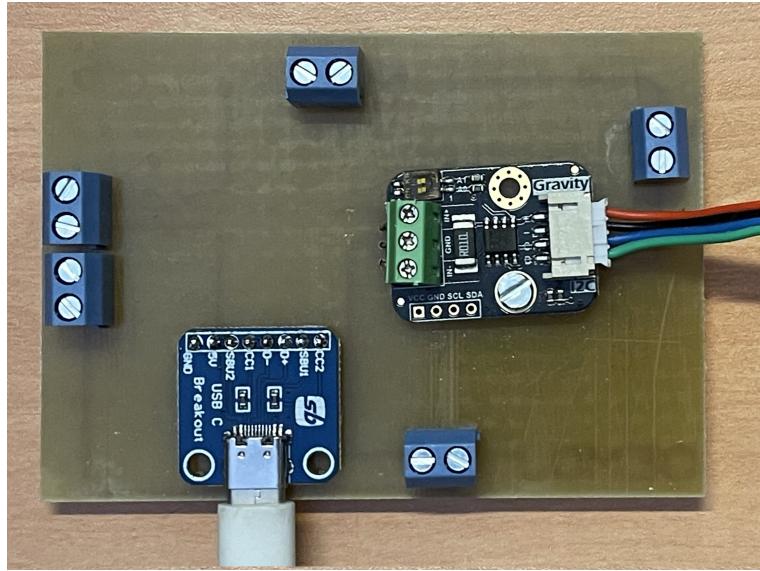


FIGURE 18 – L'interface avec les composants soudés

La figure 17 montre le résultat de l'impression.

Nous avons réalisé le circuit en quatre exemplaires au lieu de trois afin d'avoir un exemplaire libre pour la réalisation de tests. Les quatre exemplaires de l'interface ont été imprimés sur la même plaque pour que l'impression soit plus rapide. Il faut donc à la fin, couper la plaque en quatre pour récupérer les quatre interfaces.

Nous avons ensuite percé les trous qui permettent le passage des pattes des circuits au travers du circuit imprimé : les puces rondes du calque (voir 16). Puis nous avons soudé les composants sur chacune des plaques en suivant attentivement le schéma (voir 14). Nous étions nombreux à l'œuvre, et nous avons pu réaliser l'impression, le perçage et la soudure en moins de 3 heures. Nous étions donc en mesure de tester le fonctionnement du système électronique avec l'interface très rapidement. Finalement ce sont les parties d'analyse, de schématisation, de prototypage et de tests qui prirent le plus de temps. Mais cela est nécessaire car il est préférable d'avoir à imprimer et souder le circuit une seule fois pour économiser du matériel et du temps. Sur les figures 18 et 19 on peut voir l'interface avec ses composants soudés et en fonctionnement.

### 3.2.2 Les jauge

**Électronique des jauge** La librairie d'interface fournie avec le ruban NeoPixel ne peut contrôler qu'un unique ruban de leds physique sur un port GPIO série. Les deux jauge sont donc réalisées par deux rubans de leds virtuels implémentés sur un unique ruban de leds physique. En réalité, les deux rubans virtuels sont bien distincts, mais connectés entre eux en série pour ne paraître plus qu'un ruban pour le port GPIO. L'interface électronique ne voit donc qu'un circuit de contrôle et un circuit d'alimentation pour l'ensemble des deux jauge. La virtualisation des jauge consiste essentiellement en une translation d'adresse des leds unitaires logiques vers des leds unitaires physiques.

---

milieu acide, les atomes de cuivre en surface des pistes échangent leur place avec les atomes d'étain de la solution.

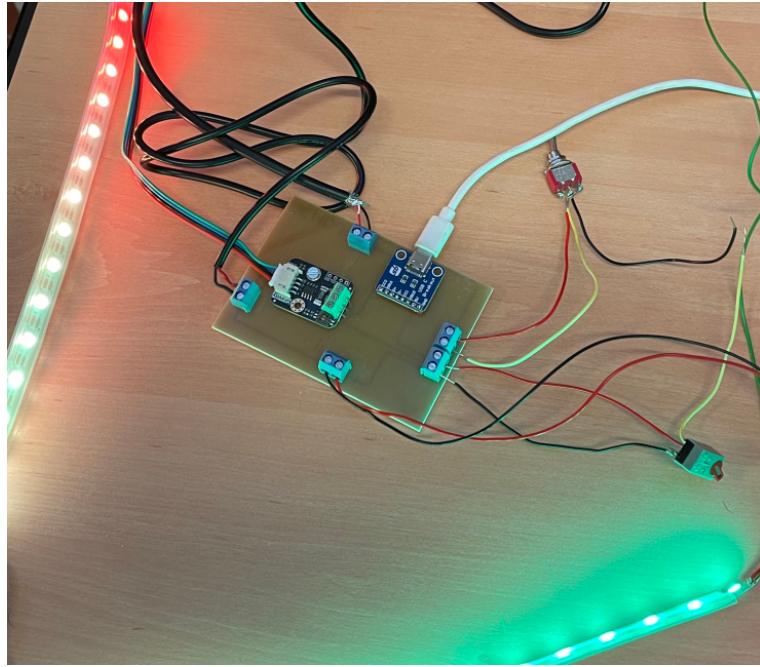


FIGURE 19 – L’interface terminée en fonctionnement

**Logiciel des jauge** Les jauge sont la seule interface directement visible par les utilisateurs (opérateur et auditeurs) du système **DémoJ**. Il est donc important que celles-ci soit lisibles par les deux types d’utilisateurs. Dans un premier rôle les jauge permettent l’affichage de la température et de la puissance à destination des auditeurs, et dans un second rôle elles donnent des indications visuelles à destination de l’opérateur lors de la mise en route des boîtes : le *bootstrap* du système **DémoJ**. Cela concerne essentiellement l’établissement du réseau wifi adhoc du système **DémoJ**.

Le logiciel de l’affichage a été décomposé en plusieurs parties pour favoriser la modularité. Cela nous permet de définir aisément des nouvelles animations. L’interface de contrôle que nous avons programmée permet à l’application principale de contrôler en parallèle les animations qui s’affichent sur les jauge sans se soucier de problèmes de synchronisations.

Toutes les animations de l’affichage du système **DémoJ** s’exécutent sur un fil d’exécution différent de celui de l’application principale. Nous aurions pu lancer les scripts Python séparément dans deux processus différents, mais pour faciliter la communication inter-processus, nous sommes passé par la bibliothèque Python **multiprocessing**<sup>12</sup>. Cette librairie nous permet de contrôler l’affichage du système **DémoJ** à l’aide d’une classe coordinatrice. Chaque animation est donc un processus distinct qui peut être exécuté sans interrompre le reste du programme. L’animation peut être arrêtée et relancée à tout moment. Cela permet par exemple de lancer une animation de chargement lors de la phase d’établissement du réseau **DémoJ**.

Pour l’affichage des jauge, nous avons inter-connecté tous les modules créés précédemment permettant le contrôle du wattmètre et de la température. Nous suivons le schéma de la figure 4 avec quatre plages de couleurs unies : vert, jaune, orange, rouge. Le nombre de leds allouées à chaque couleur est programmable, ce qui nous permet de

---

12. **multiprocessing** : librairie standard de parallélisation inter-processus de Python, <https://docs.python.org/3/library/multiprocessing.html>

réajuster si on le souhaite.

Il reste à définir le contrôle fin de l'affichage des jauge.

**Stabilisation des jauge** Sans contrôle spécifique les jauge ont tendance à clignoter ou à afficher des pics soudains qui sont difficiles à expliquer. Nous avons utilisé pour éviter cela une procédure de lissage par moyenne flottante exponentielle (**EMA**<sup>13</sup>). Celle-ci est très facile à implémenter car elle est régie par une formule très simple :

$$\sigma = \alpha * \phi + (1 - \alpha) * \sigma$$

avec  $\alpha \in [0, 1]$ ,  $\sigma$  la valeur moyenne actuelle et  $\phi$  la dernière valeur lue.

Il s'agit de calculer la moyenne entre toutes les valeurs en donnant plus de poids aux plus récentes. On peut réguler la priorité des nouvelles valeurs sur les anciennes en faisant varier  $\alpha$ . Plus  $\alpha$  est proche de 1 est plus le présent pèse, et plus  $\alpha$  est proche de 0 est plus c'est le passé qui pèse. Le calcul de cette moyenne flottante ajoute un peu de latence à l'affichage, mais moins que d'autres méthodes, et c'est la conséquence du lissage, qui évite les clignotements et les pics soudains.

**Étalonnage des jauge** L'étape suivante est de calibrer la hauteur de la jauge en fonction des valeurs mesurées. C'est sans doute ce qu'il y a de plus fastidieux et long à réaliser, mais probablement l'un des éléments les plus important pour la lisibilité du système **DémoJ** par les auditeurs.

La difficulté est que la relation entre l'effort de calcul et la quantité d'énergie consommée n'est pas simple à formaliser et c'est encore pire pour la température. Malgré tout, il est important que les minimum et maximum affichés correspondent aux minimum et maximum observés.

Ici plusieurs facteurs entrent en compte même si parfois minimes, notamment la température extérieure, le confinement thermique par les boîtes, le refroidissement du système (ventilateurs et dissipateurs de chaleur), etc.

**Jauge d'énergie** L'énergie instantanée consommée par chaque processeur du système **DémoJ** est mesurée par le wattmètre qui est installé sur chaque carte d'interface (voir section 3.2.1). La connexion physique du wattmètre avec le Raspberry Pi utilise le protocole **I2C**<sup>14</sup> sur les broches 2 (**SDA**<sup>15</sup>) et 3 (**SCL**<sup>16</sup>) du port GPIO. L'énergie mesurée transmise au programme par le wattmètre consiste en l'énergie de base consommée par le système en attente, essentiellement Raspberry OS, plus l'énergie consommée par les scénarios. Le minimum affiché ne doit donc pas être zéro, mais l'énergie de base. Le maximum affiché a été mesuré empiriquement et augmenté d'une marge de sécurité. Au total, l'intervalle de consommation électrique effective est de 10-17.5 VA.

Nous avons profité de ces expériences pour tester des procédés de mesure de la charge de la batterie, mais nous y avons renoncé, car les calculs possibles à partir des données fournies par le wattmètre donnent des résultats très imprécis.

---

13. EMA : *Exponential Moving Average*, ou moyenne des mesures pondérées exponentiellement ou  $\sum_{i \in [0..t]} \alpha^i \times m_{t-i}$

14. I2C : *Inter-Integrated Circuit*, <https://les-electroniciens.com/sites/default/files/cours/coursi2c.pdf>

15. SDA : *Serial Data Line*

16. SCL : *Serial Clock Line*

**Jauge de température** L'affichage de la température ajoute une difficulté que ne connaît pas l'affichage de l'énergie instantanée : l'inertie. La température monte avec un délai par rapport au stress d'un scénario et diminue aussi assez lentement après le stress. Nous avons observé que sans radiateur ni ventilateur, la température descend beaucoup plus lentement qu'elle ne monte, au point de gêner l'enchaînement des scénarios ; un scénario démarrerait avec une dette thermique léguée par le scénario d'avant.

De plus, les installations standard font tout pour dissiper la température des processeurs dans l'air ambiant. Pour les Raspberry Pi que nous utilisons, la solution standard est d'installer un radiateur et un ventilateur. Ces installations sont assez efficaces et mettent à mal la lisibilité des scénarios **DémoJ**. En effet, ce qui devrait beaucoup chauffer ne chauffe plus beaucoup. La température n'est plus un proxy de l'énergie consommée. Nous avons donc décidé de supprimer tout ou partie des équipements de dissipation de chaleur pour ne pas masquer les effets thermiques, mais en en laissant juste assez pour que la baisse de température après stress se fasse assez rapidement. Noter que ce n'est pas une tricherie par rapport au message d'une médiation **DémoJ** puisque celui-ci ne porte pas sur la température, mais sur la consommation d'énergie qui se manifeste dans l'élévation de la température.

Pour étalonner l'affichage de la température nous avons donc fait des expériences avec et sans dissipateurs de chaleur, et en utilisant les applications **stress**<sup>17</sup> et **stress-ng**<sup>18</sup> car les scénarios **DémoJ** n'étaient pas disponibles. Nous en avons retenu que le mieux est de conserver le radiateur mais pas le ventilateur, et que dans cette configuration là, l'intervalle de température effectif est de 36-45 °C. Il faut noter que le système Raspberry OS est lui-même muni d'une protection logicielle qui lui fait ralentir les opérations quand la température du processeur atteint 80 °C.

**Les animations indicatrices** Pour que l'opérateur puisse avoir un témoin du démarrage des boîtes, le système **DémoJ** affiche une animation inspirée de la série K2000<sup>19</sup> pendant toute la durée du démarrage avec une couleur rappelant la couleur de la boîte du module correspondant. Il s'agit d'un chenillard avec un dégradé de lumière donnant l'illusion d'une vague de lumière qui se déplace. Une fois la connexion réalisée, les leds s'affichent en vert puis lancent l'animation des jauge.

### 3.3 Le sous-système informatique

Le sous-système informatique est essentiellement un système distribué déployés sur trois nœuds constitués chacun d'un exemplaire du système électronique. On appellera ces noeuds **Réseau**, **Terminal** et **Serveur**.

Chaque nœud est constitué d'une couche logicielle commune à tous les nœuds, plus des logiciels spécifiques au rôle de chacun des nœuds. La couche commune à tous les nœuds constitue le réseau **DémoJ**, c'est-à-dire l'infrastructure partagée par tous les nœuds pour le système puisse fonctionner (voir la section 3.3.1).

Les logiciels spécifiques à chaque nœud sont un serveur web pour le nœud **Serveur**, et des scripts d'application pour le nœud **Terminal**.

Prévoyant des mises à jour fréquentes pendant la phase de développement, et des mises à jours plus rares mais réalisées par un opérateur non expert pendant la phase

---

17. stress : <https://github.com/mattixtech/stress>

18. stress-ng : <https://github.com/ColinIanKing/stress-ng>

19. K2000 : série télévisée des années 80

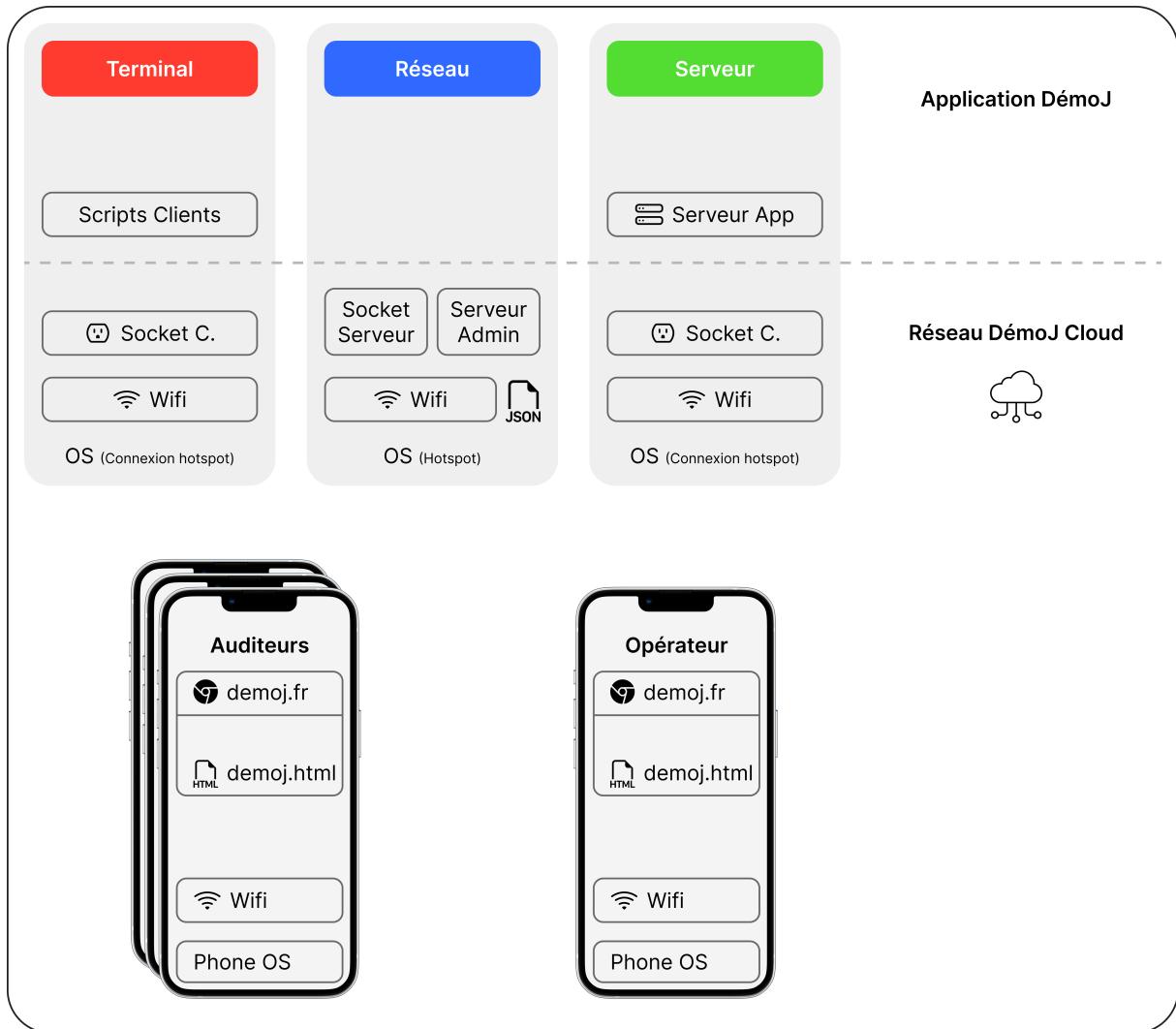


FIGURE 20 – Schéma du réseau DémoJ

d'utilisation, nous avons réfléchi à une manière simple de réinstaller le logiciel du système **DémoJ**. Ici, simple signifie que cela devrait être possible sans connaissance approfondie et sans risque d'erreur de manipulation. Nous avons donc développé un script `bash` qui effectue toutes les actions nécessaires pour créer le disque amorçable de chaque boîte.

### 3.3.1 Le réseau DémoJ

Le réseau **DémoJ** est constitué d'une pile qui comporte une couche wifi adhoc (réseau wifi DemoJ), dont le nœud **Réseau** est le *hotspot*, et d'une couche de sockets qui permet au nœud de communiquer entre eux de façon bi-directionnelle (voir la figure 20). Le nœud **Réseau** héberge aussi un serveur d'administration qui permet de partager les variables d'état du système **DémoJ** entre tous les nœuds, et qui met à disposition de l'opérateur et des auditeurs l'application **DémoJ Connect**. Les variables d'état sont stockées dans un fichier **JSON**<sup>20</sup> (`demoj.json`), et l'application **DémoJ Connect** se présente comme un fichier **HTML**<sup>21</sup> (`demoj.html`).

20. JSON : *Javascript Object Notation*

21. HTML : *Hypertext Markup Language*

Sur les smartphones une pile similaire est construite à l'aide du système d'exploitation des smartphones (Phone OS dans la figure 20) et de leur navigateur. L'utilisateur du smartphone qui voudrait l'utiliser comme un terminal **DémoJ** devra se connecter au réseau wifi **DemoJ**, puis lancer son navigateur pour requêter l'URL `demoj.fr`. En réponse, le navigateur recevra la page HTML de l'application **DémoJ Connect** (`html1.fr`). Cette page HTML utilise des web sockets qui correspondent avec les sockets des noeuds **DémoJ**.

Pour mettre en place le réseau **DémoJ**, chaque noeud démarre son système d'exploitation. Ensuite, le noeud **Réseau** initialise son point d'accès (*hotspot*) en créant un réseau wifi ouvert (*DemoJ*) à l'aide de l'application **RaspAp**<sup>22</sup>. Ce réseau wifi est utilisé par les noeud **Terminal** et **Serveur**, qui sont configurés pour s'y connecter automatiquement avec des adresses IP statiques respectives : 10.3.141.2 pour le terminal et 10.3.141.3 pour le serveur (et 10.3.141.1 pour le réseau). Les smartphones des auditeurs peuvent aussi se connecter au réseau wifi **DemoJ**.

Le serveur d'administration du réseau **DémoJ** est programmé en **Flask**<sup>23</sup> [4]. Il comporte essentiellement deux routes : l'une sert le fichier `demoj.html`, l'autre répond aux requêtes programmées dans la page `demoj.html`.

Nous utilisons des adresses IP statiques pour faciliter la communication entre les noeuds dans les scripts Python.

### 3.3.2 Application de contrôle **DemoJ Connect**

L'application de contrôle **DémoJ Connect** est constituée essentiellement d'une application de gestion des variables d'état du système **DémoJ** et d'application du rôle terminal qu'on appelle les **scénarios**.

**Architecture de l'application DémoJ Connect** Pour garantir une expérience utilisateur fluide, nous avons opté pour une architecture de type **SPA**<sup>24</sup> plutôt que de développer une application mobile dédiée. Cette décision s'est appuyée sur la plus grande accessibilité d'une application web depuis n'importe quel dispositif via un navigateur, évitant ainsi les contraintes liées au développement mobile et permettant une accessibilité quasi universelle.

L'application **DémoJ Connect** contrôle le système **DémoJ** via des variables d'état représentées dans un fichier JSON. La représentation de ces données en JSON a été choisie par commodité : sa grande modularité et son adéquation avec Javascript, le langage principal utilisé dans l'application **DemoJ Connect**.

Le fichier JSON des variables d'état du système **DémoJ** contient des informations telles que le statut de connexion, les paramètres des modules, et les détails des scénarios. Il est stocké dans le noeud **Réseau**. Ce dernier est chargé de le mettre à jour via les requêtes HTTP provenant des autres modules et de l'application **DemoJ Connect**.

**Les scénarios** Les scénarios sont des applications dont on peut facilement expliquer la fonction au public et qui stressent tout ou partie du système de façon suffisamment éloquente et contrôlée. La difficulté est de combiner ces deux dimensions. Les applications **stress** en sont un contre-exemple car si elles permettent bien de stresser de façon contrôlée un processeur (au cœur près), elles ne réalisent aucune fonction facilement explicable.

---

22. RaspAp : <https://raspap.com>

23. Flask : <https://flask.palletsprojects.com/en/3.0.x/>

24. SPA : *Single Page Application*

## Calculatrice



## Streaming

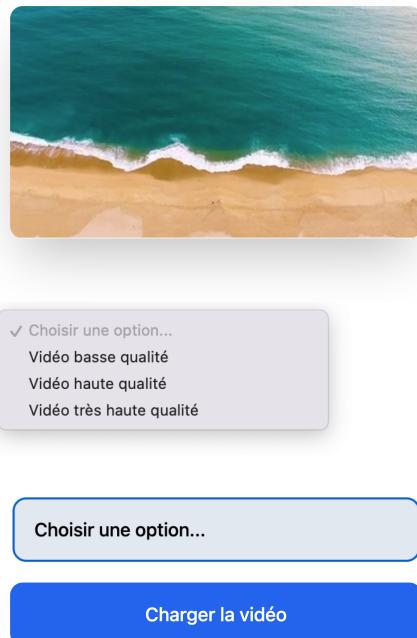


FIGURE 21 – Écran de contrôle du scénario Calculatrice

FIGURE 22 – Écran de contrôle du scénario Streaming

Dans le contexte d'un outil de médiation, nous avons préféré développer des scénarios à la fonction facilement explicable, même si ils ne stressent pas le système de façon aussi prévisible.

Afin de mettre en place l'exécution des scénarios depuis l'application **DemoJ Connect**, il nous fallait décider de la façon dont les modules et l'application communiqueront. Ainsi, l'application **DemoJ Connect** a la responsabilité de gérer le fichier de base de données puisque l'application et ce dernier partagent le même format (JSON). De ce fait, les routes disponibles sur le serveur correspondent à la récupération du fichier de base de données JSON, à l'exécution d'un scénario ainsi qu'à la modification des paramètres des modules. Le fichier étant très léger (173 octets), il n'y a pas de problème à le transmettre via le réseau.

Dans cette première version du système **DémoJ**, les scénarios sont au nombre de trois. Ils permettent de mettre en évidence différents phénomènes qui pourront ainsi être médiatisés à l'intention du public.

**Scénario 1 : Une calculatrice de précision arbitraire** Il s'agit d'un scénario de stress du processeur, soit celui du nœud **Serveur**, soit celui du nœud **Terminal**. Il prend la forme d'une calculatrice qui opère sur des nombres de tailles non bornées, et qui en

plus des quatre opérations classiques, propose le calcul de la factorielle, le calcul du  $i$ -ème élément de la suite de Fibonacci, et le test de primalité d'un nombre entier (voir figure 21). On peut ainsi réaliser des calculs qui stressent les processeurs. Ce scénario est commandable via l'application **DémoJ Connect** pour faire que les calculs aient lieu soit sur le nœud **Serveur**, soit sur le nœud **Terminal**. Même si les nombres manipulés peuvent paraître très grands, ils sont limités par la taille de la mémoire allouée à un processus. Ils ne sont donc pas si grands que cela et les échanger entre le nœud **Serveur** et le nœud **Terminal** ne stresse pas le nœud **Réseau**. Si les calculs sont exécutés côté serveur et son commandés par plusieurs terminaux, celui de l'opérateur de la démonstration et ceux des auditeurs, ce scénario permet aussi de stresser plusieurs coeurs du processeur du nœud **Serveur**. Enfin, les opérations les plus complexes existent en une version naïve et en une version de l'état de l'art. Par exemple, le test de la primalité d'un nombre  $n$  est réalisé naïvement par test de divisibilité par les nombres premiers entre 1 et  $\sqrt{n}$  ou en employant l'algorithme AKS [1]. Ces variantes permettent de sensibiliser à l'impact du logiciel et des algorithmes employés.

**Scénario 2 : Un service de streaming** Le streaming est un des services offerts au grand public par les systèmes informatiques qui est le plus montré du doigt [9]. Ce scénario prend la forme d'un service de streaming qui sert des fichiers de qualités différentes mais de contenus similaires. Il permet de stresser toutes les couches du système et de sensibiliser aux impacts des formats vidéos trop lourds. Lors de nos tests avec ce scénario, nous avons constaté que charger des vidéos de différentes qualités sur un navigateur d'ordinateur ne posait aucun problème. Cependant, sur un smartphone, il était impossible de charger une vidéo en très haute qualité, souvent dépassant la résolution de l'écran, en raison des optimisations du navigateur. Pour y parvenir, des ajustements de paramètres étaient nécessaires.

**Scénario 3 : Une application d'IA générative** Pour répondre aux interrogations du public sur le développement des applications d'IA générative, nous avons conçu un scénario novateur dans ce domaine, mettant en œuvre l'outil Ollama<sup>25</sup>. Inspiré par des technologies telles que ChatGPT d'OpenAI, Gemini (Google Bard) de Google, et Copilot de Microsoft, notre projet offre au public la possibilité d'interagir avec un assistant virtuel doté de capacités similaires. La démocratisation croissante de ces technologies a désormais rendu possible l'exécution de modèles de langage de grande envergure sur des ordinateurs personnels, offrant ainsi la perspective d'avoir un assistant virtuel entièrement autonome et fonctionnel en local, en quelques secondes seulement. Quel en est l'impact ?

Comparé aux deux précédents scénarios, ce dernier présente des caractéristiques différentes, notamment en ce qui concerne le stress sur la RAM du nœud **Serveur**. En utilisant ce scénario, il sera possible d'observer clairement les pics d'utilisation de la RAM du nœud **Serveur** dès qu'un utilisateur soumettra une question à l'assistant virtuel. La mise en place de ce scénario a permis de mettre en lumière plusieurs éléments ayant un impact sur l'efficacité de l'outil Ollama ainsi que sur la pertinence des réponses fournies.

Des tests ont d'abord été réalisés sur un ordinateur personnel, où les modèles ont affiché des temps de réponse imperceptibles. Deux modèles de langage de tailles différentes, 1,6 Go et 4,7 Go respectivement, ont été utilisés lors de ces tests. En ce qui concerne le contenu, le modèle de taille réduite ne fournissait pas de réponse en français correct.

---

25. Ollama : <https://ollama.com/>

Les tests ont alors continué sur un Raspberry Pi équipé d'un ventilateur, et plusieurs observations notables ont pu être faites. Malgré la production de réponses plus pertinentes et en français par le modèle plus volumineux, il s'est avéré impossible de l'utiliser sur le Raspberry Pi, en raison de la nécessité de charger entièrement le modèle en mémoire vive avant leur utilisation. Cette contrainte pose problème étant donné que les Raspberry Pi dont nous disposons ne sont équipés que de 4 Go de RAM. Nous avons donc opté pour le modèle de plus petite taille, qui ne permet que les échanges en anglais. La seconde observation est que à chaque envoi de requête à l'assistant, la température du Raspberry Pi augmente d'environ 10 °C, mettant en évidence le stress thermique induit par cette opération.

Par ailleurs, lors de l'installation de l'outil sur un Raspberry Pi, un avertissement nous a été envoyé, indiquant l'absence de GPU et donc la limitation des performances. Cela souligne que ces assistants virtuels requièrent une quantité significative de ressources pour fonctionner. Dans le contexte actuel où l'intégration de l'intelligence artificielle dans les appareils électroniques est une tendance marquée, une évolution des processeurs est en cours. Cette évolution inclut l'intégration d'accélérateurs d'intelligence artificielle, connus sous le nom de NPU (*Neural Processing Unit*) en anglais, visant à rendre ces appareils fonctionnels de manière optimisée en local.

En outre, la configuration actuelle du scénario permet de recevoir la réponse de l'assistant virtuel mot par mot, reflétant ainsi le comportement des agents conversationnels comme ChatGPT. Chaque mot est donc une réponse que le navigateur reçoit et que le réseau prend en charge. Toutefois, il serait envisageable de modifier ce comportement pour bufferiser la réponse et l'envoyer en un seul bloc. La possibilité de choisir entre ces deux modes pourrait potentiellement mettre en évidence un stress différent sur le réseau.

## 4 Documentation utilisateur

Une documentation utilisateur a été conçue à destination de l'opérateur de la médiation. Elle est composée de deux types de documents : un manuel à lire de préférence avant toute médiation pour se préparer à déployer le système **DémoJ** puis à utiliser l'application de contrôle **DémoJ Connect**, et des fiches-réponses en cas de trous de mémoire pendant la médiation.

Le manuel prend la forme d'un texte rédigé, tandis que les fiches-réponses sont juste des aides-mémoires conçus sur le modèle des fiches de sécurité qu'on peut trouver dans les établissements recevant du public.

## 5 Test et validation

Pour garantir le bon fonctionnement du système **DémoJ**, nous avons mis en place une série de tests visant à évaluer les performances du sous-système des jauge, de l'application **DémoJ Connect** et ses scénarios, ainsi que la communication entre les différents modules.

Dans le cadre des tests des jauge, nous avons développé plusieurs programmes permettant d'afficher la température et la consommation énergétique à chaque étape du développement du système **DémoJ**. Ces programmes nous ont offert un contrôle total sur le fonctionnement des jauge, nous permettant ainsi de vérifier la précision de l'affichage et de détecter d'éventuels dysfonctionnements.

Étant donné que l'application web communique avec les différents modules du démonstrateur énergétique via une API REST et des connexions WebSocket, nous avons effectué des tests de communication pour garantir la fiabilité et la stabilité de ces échanges de données. Nous avons vérifié la capacité de l'application à recevoir et à traiter correctement les données provenant des capteurs, ainsi que sa capacité à envoyer des commandes de contrôle aux rubans de leds.

Les tests ont révélé que DemoJ fonctionne de manière fiable et stable, offrant une expérience utilisateur fluide et réactive. Aucun bug majeur n'a été détecté, et toutes les fonctionnalités principales ont été validées avec succès. Les performances de l'application ont également été satisfaisantes, même dans des conditions de charge élevée.

## 6 Conclusion

Le projet **DémoJ** consiste à produire une maquette dynamique et ludique d'un système informatique dans le but d'être présentée au grand public dans des environnements divers, tels que des salons ou des bars, dans le but de sensibiliser à la consommation électrique des systèmes informatiques.

Nous avons réalisé une première version aboutie et robuste du système **DémoJ**. Robuste, car nous avons apporté un soin particulier au packaging physique et électronique du système. Aboutie, car le système **DémoJ** a été réalisé de bout en bout sous tous ses aspects.

Le résultat est un démonstrateur énergétique capable d'afficher la consommation instantanée et la température d'un système informatique composé des trois couches essentielles suivantes : les équipements terminaux, les équipements réseau et les équipements serveur.

Sachant que notre objectif principal est de sensibiliser le grand public, nous avions comme objectif de rendre DemoJ accessible à tous. C'est pour cela que DemoJ a été pensé pour être transportable et attractif grâce à son apparence soignée. Les utilisateurs seront donc en mesure de réaliser des présentations en s'appuyant sur notre maquette dans des lieux à leur convenance et d'une façon qui sera compréhensible pour leur audience.

Le système **DémoJ** en est à sa première version et il existe donc de nombreuses pistes d'amélioration telles que un meilleur processus de refroidissement du Raspberry Pi, un suivi de la décharge de la batterie et par là la possibilité de suivre la consommation cumulée, ou encore une meilleure façon de déployer l'application.

Dans sa conception actuelle le système **DémoJ** représente chaque couche par un unique équipement. C'est une simplification extrême qui permet essentiellement de présenter l'aspect qualitatif des choses. On pourrait imaginer une variante **DémoJ\*** où chaque couche est représentée par plusieurs équipements. Cela demanderait de refondre toute l'architecture mais cela pourrait peut-être d'offrir une vision plus quantitative.

Tout au long de ce projet nous avons évolué dans un environnement hybride nous permettant d'apprendre énormément dans différents domaines. Notamment sur le plan électronique, web, réseau, pour les domaines techniques, mais aussi sur le plan de packaging et de la documentation pour le domaine de la conception produit. Ce projet a été bénéfique au sein de notre formation, nous avons acquis des nouvelles connaissances et compétences dans de nombreux domaines qui relèvent parfois d'un autre domaine que celui de l'informatique.

**Remerciements :** Nous remercions Mme Camille Bisson, manager du fablab de Beau-lieu, et M. Régis Legave, responsable de l'atelier d'électronique de l'ISTIC, pour leur accueil, et leurs conseils concernant des technologies que nous ne connaissions pas bien en commençant.

## Références

- [1] Manindra AGRAWAL, Neeraj KAYAL et Nitin SAXENA. « PRIMES is in P ». In : *Annals of Mathematics* (2004).
- [2] Dominique CARDON et al. *Atlas du numérique*. Les presses de SciencesPo, 2023.
- [3] GREENIT. *The environmental footprint of the digital world*. 2019. URL : [https://www.greenit.fr/wp-content/uploads/2019/11/GREENIT\\_EENM\\_etude\\_EN\\_accessible.pdf](https://www.greenit.fr/wp-content/uploads/2019/11/GREENIT_EENM_etude_EN_accessible.pdf).
- [4] Miguel GRINBERG. *Flask Web Development: Developing Web Applications with Python*. O'Reilly Media, Inc., 2018.
- [5] Udit GUPTA et al. « Chasing Carbon: The Elusive Environmental Footprint of Computing ». In : *International Symposium on High-Performance Computer Architecture (HPCA 2021)*. 2021.
- [6] Gareth HALFACREE. *LE GUIDE OFFICIEL du débutant Raspberry Pi*. Raspberry Pi Press, 2020. URL : [https://www.framboise314.fr/docs/BeginnersGuide-4thEd-FR\\_v5.pdf](https://www.framboise314.fr/docs/BeginnersGuide-4thEd-FR_v5.pdf).
- [7] Ray OLDENBURG. *The Great Good Place: Cafés Coffee Shops Bookstores Bars Hair Salons and Other Hangouts at the Heart of a Community*. Group West, 1999.
- [8] The DemoJ PROJECT. *Rapport final*. 2023.
- [9] The Shift PROJECT. *Climate Crisis: The Unsustainable Use of Online Video*. 2019. URL : <https://theshiftproject.org/en/article/unsustainable-use-online-video/>.
- [10] The Shift PROJECT. *Expanding digital sufficiency*. 2021. URL : <https://theshiftproject.org/en/article/implementing-digital-sufficiency/>.