CHAPITRE 4 : Matériel et systèmes d'exploitation 1

CHAPITRE 4 : Matériel et systèmes d'exploitation 1

1 Architecture de réseaux

Un réseau est un ensemble de machines (ordinateurs, tablettes, consoles, smartphones etc.) reliées entre elles par des liaisons filaires (cuivre, fibre optique) ou radio (Wi-Fi, Bluetooth, 4G, 5G, satellite).

1.1 Histoire des réseaux informatiques en bref

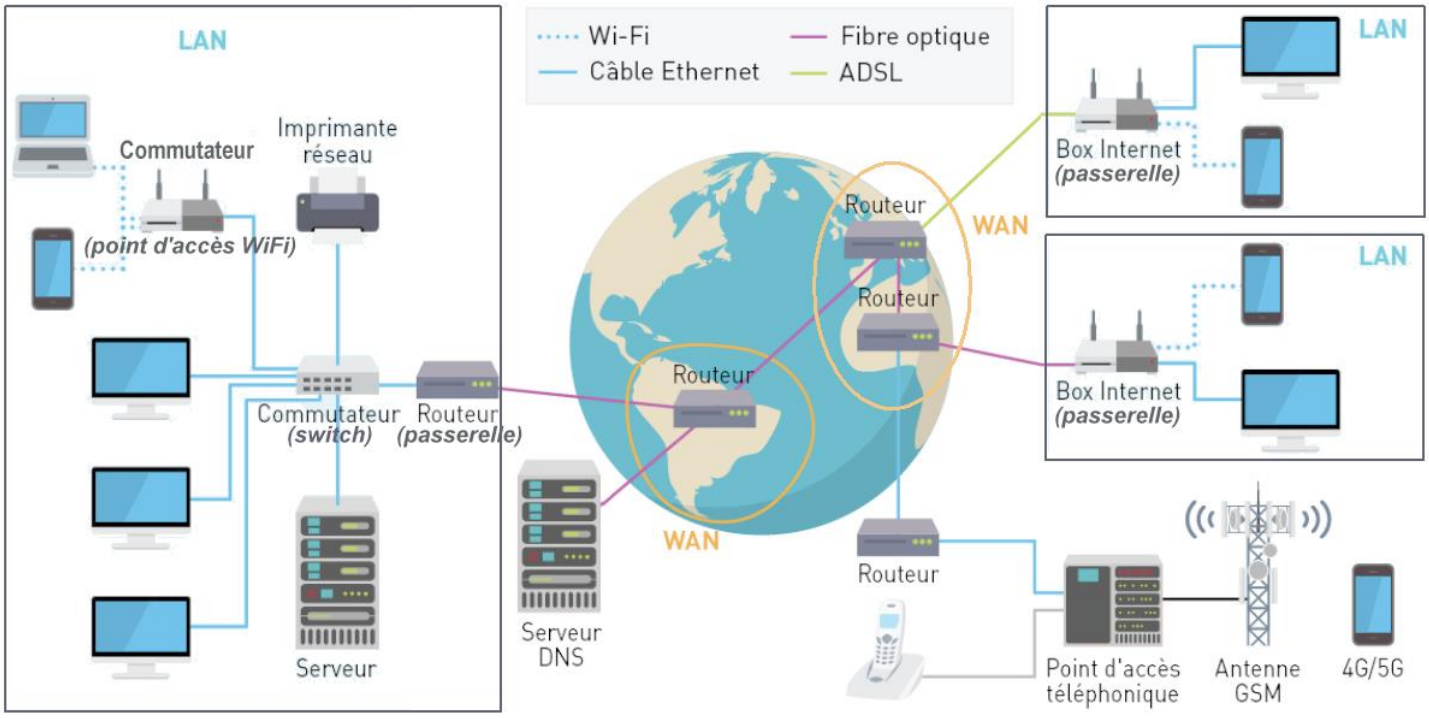
1969 : Création d'**ARPANET** premier réseau où l'information est découpée en "paquets". A ses débuts, ARPANET était composé de 4 ordinateurs et de 8 câbles reliant 4 universités de l'ouest américain.

1983 : Adoption des protocoles ***Transmission Control Protocol*** et ***Internet Protocol*** (TCP IP).

- **IP** indique l'adresse (une suite d'entiers) des machines émettrice et réceptrice.
- **TCP** indique le numéro de port logiciel. Par exemple 80 pour consulter un site web http.

1.2 Commutateurs et routeurs

- Sur un réseau local ou *Local Area Network* (**LAN**) les machines sont reliées à des **commutateurs** (des *switches* par câbles Ethernet ou des *points d'accès Wi-Fi* par radio).
- Le réseau local est connecté à un réseau étendu ou *Wide Area Network* (**WAN**) par une machine appelée **routeur** (fonction de passerelle) elle-même connectée à **d'autres routeurs**.



1.3 Les adresses sur un réseau informatique

1.3.1 Adresse IP

Au démarrage de chaque machine ou de chaque routeur, son **interface** (carte réseau) est en attente d'une adresse IP qui lui permettra d'être identifié puis d'envoyer et de recevoir des paquets.

L'adresse IP peut être **statique** (permanente sauf modification par l'administrateur) ou bien peut être **dynamique** (à chaque nouvelle connexion, un programme sur un serveur distribue une adresse IP).

- Avant 2010, l'adresse IP était composée de **4 octets**, souvent écrits en base décimale (IPv4). Mais le nombre d'adresses différentes possibles $2^{32} \approx 4\ milliards$ est devenu insuffisant. L'adresse IP est passée à **16 octets** souvent écrits en base hexadécimale. Cela permet $2^{128} \approx 3{,}4.10^{29}\ milliards$ adresses différentes. C'est la version IPv6 qui coexiste avec la version IPv4.

Exemple

Version	Composition	Exemple (écriture décimale)	Exemple (écriture hexadécimale)
IPv4	4 octets	216.58.198.195	
IPv6	8 fois 2 octets		2a03:2880:f12d:83:face:b00c:0:25de

`2a03:2880:f12d:83:face:b00c:0:25de` est égal à `2a03:2880:f12d:0083:face:b00c:0000:25de`

1.3.2 Adresse MAC

En plus de l'adresse IP (qui peut varier d'une connexion à l'autre), les cartes réseau (câble Ethernet ou WiFi) possèdent une **adresse permanente fixée** par le constructeur. C'est l'adresse **MAC** *Media Accss Control* sur **6 octets**. Cela permet $2^{48} \approx 2{,}8.10^{5}\ milliards$ adresses différentes.

Exemple

Composition	Exemple (écriture hexadécimale)
6 octets	08:d5:9d:bd:09:cc

1.4 Le routage des paquets

La **technique** qui permet le transport d'un paquet de données d'une machine émettrice à une machine réceptrice s'appelle le **routage**. Chaque paquet contient, en plus des données elles-mêmes, l'adresse IP de l'émetteur et du récepteur.

Chaque routeur possède une **table de routage** qui lui indique, à un moment donné :

- les réseaux qui lui sont directement connectés.
- à quel routeur voisin transmettre le paquet en fonction de l'adresse IP de destination.

1.5 Les commandes réseaux pour Linux

Commande	Rôle de la commande
`ifconfig` `ipconfig` sous Windows	Affiche les informations des interfaces et permet de les configurer.
`ping`	Teste l'accessibilité d'une machine grâce à son adresse IP et le temps d'accès.
`traceroute` `tracert` sous Windows	Donne la liste des routeurs (au maximum 30) intercalés entre l'émetteur et le récepteur.
`nslookup`	Interroge les serveurs DNS afin d'obtenir des informations sur un nom de domaine.

2 Les protocoles

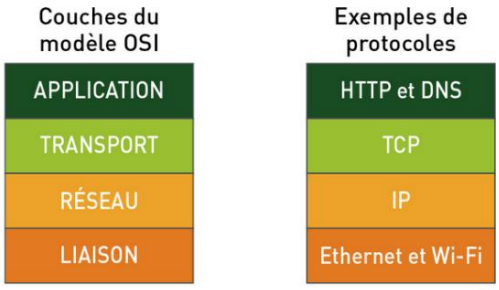
Les protocoles sont des ensembles de règles bien définies.

2.1 Les protocoles IP, TCP, HTTP et DNS

- **IP** précise l'adresse de l'émetteur et du récepteur ce qui permet la transmission des paquets par les routeurs. Parfois certains paquets sont perdus.
- **TCP** indique le numéro de port logiciel visé sur le système récepteur. Ce numéro est différent par exemple pour afficher une page web, pour envoyer un e-mail etc. De plus TCP s'assure que les paquets sont bien reçus. Si un paquet est perdu alors un nouveau identique est renvoyé.
- **HTTP** *HyperText Transfer Protocol* permet à un navigateur situé sur la machine cliente (celle de l'utilisateur) de demander à une machine hôte (sur un site) d'envoyer une page web.
- **DNS** *Domain Name System* est un protocole qui permet communiquer avec les serveurs qui transforment un nom de site web (par exemple **google.fr**) en adresse IP (**216.58.198.195**).

2.2 Les couches réseaux

Les protocoles peuvent être vus comme appartenant à des couches allant au plus proche du matériel (niveau Ethernet ou WiFi) jusqu'à l'application manipulée par un utilisateur (navigateur par exemple).



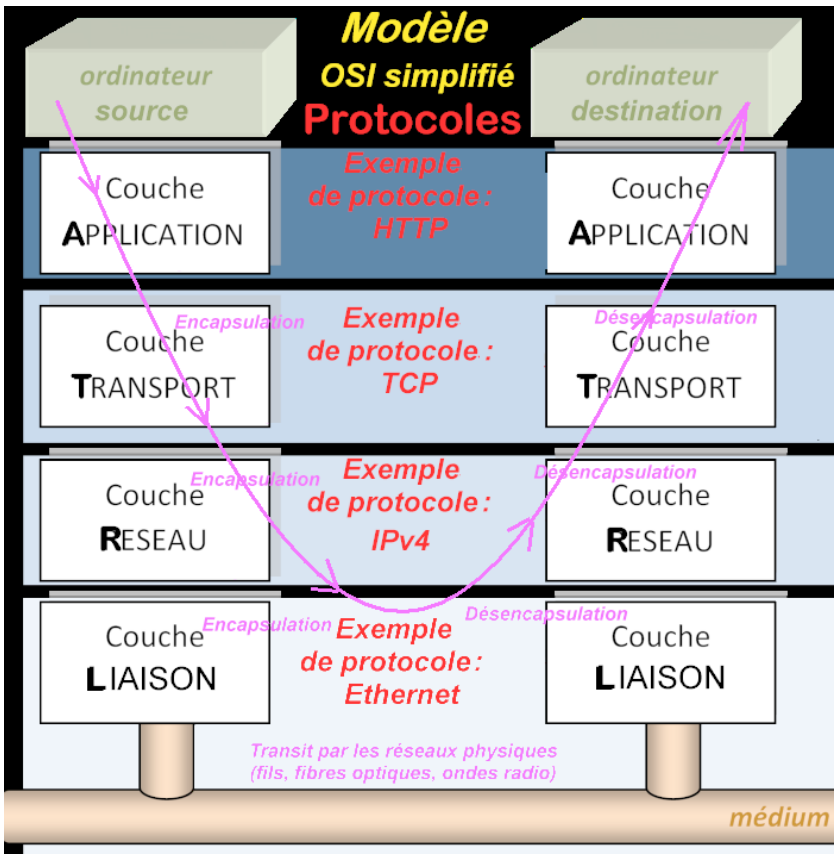
Le **modèle OSI** *Open Systems Interconnexion* est une norme de communication de tous les systèmes informatiques. Le modèle OSI en 7 couches a été présenté en 1978. On peut le simplifier en 4 couches.

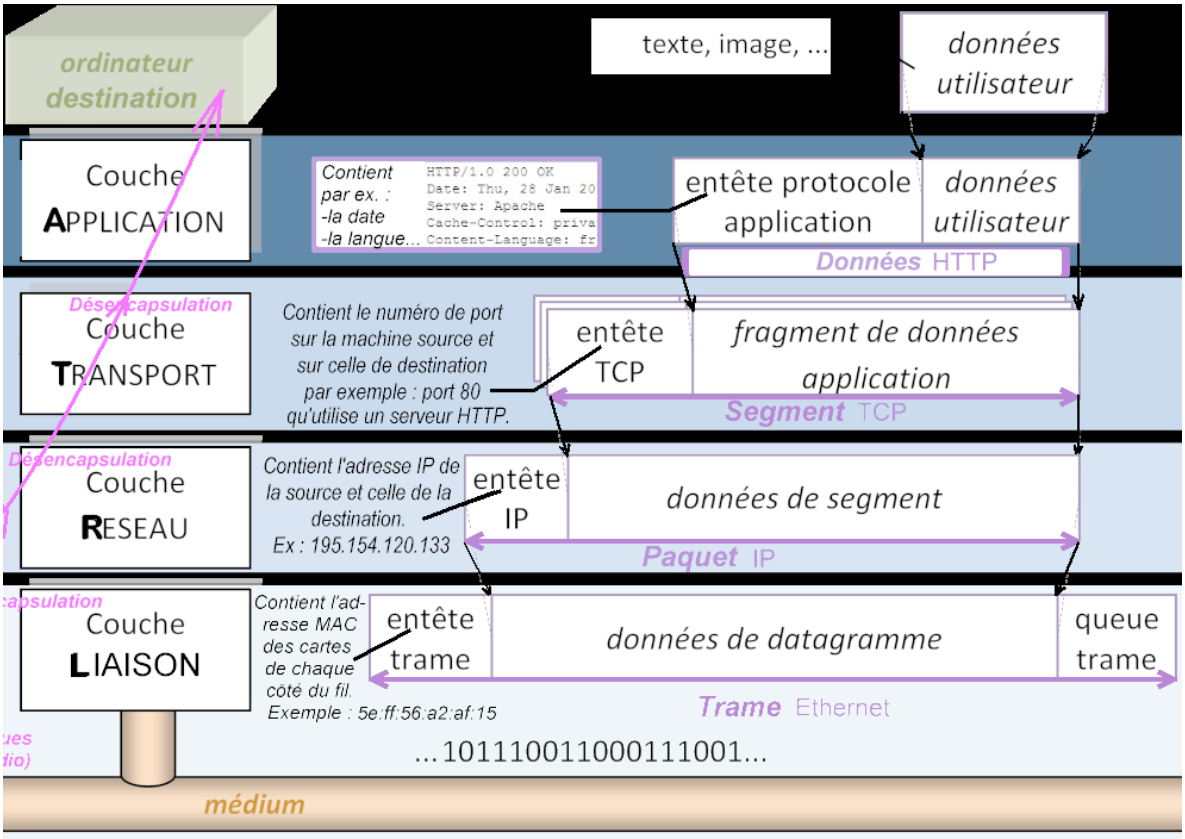
2.3 Intérêt du découpage des données en paquets

- Éviter de devoir tout retransmettre en cas de perte. **Seul le paquet perdu est renvoyé**.
- Routeurs et câbles partagent leur temps à transmettre des paquets provenant de différents émetteurs. **Les routeurs et les câbles ne sont pas monopolisés par un seul émetteur.**

2.4 Fabrication d'un paquet

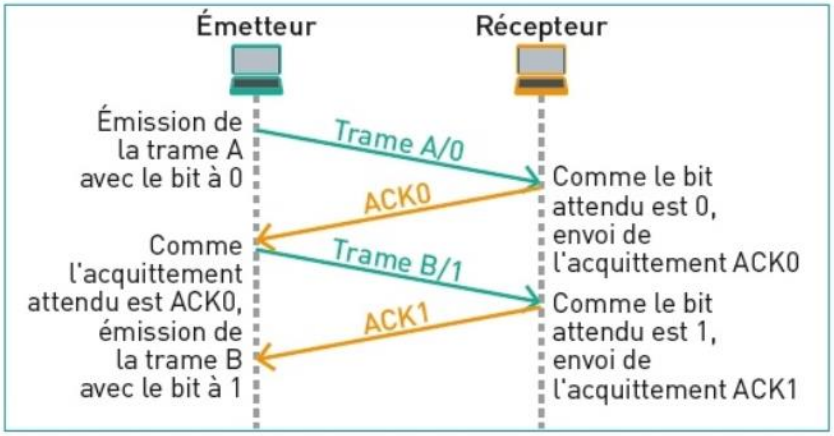
- Les données de départ sont *encapsulées* avec des *données supplémentaires* plusieurs fois de suite en suivant l'ordre descendant du modèle OSI. A l'arrivée, la désencapsulation redonne les données.





2.5 Protocole du bit alterné

- Le protocole du bit alterné **ABP** *Alternating Bit Protocol* est un protocole au niveau de la couche de liaison (la plus basse du modèle OSI). ABP s'assure que les trames sont bien reçues. Si la trame a été perdue, elle n'est pas acquittée et une nouvelle trame identique est émise.



- **Émetteur :** "J'envoie la trame A/**0** avec le bit "drapeau" ("flag" en anglais) **0** ."
- Par exemple cette trame est perdue. Donc Récepteur ne peut pas envoyer de trame d'acquittement.
- Émetteur n'a pas reçu d'acquittement. Au bout d'un certain délai :
- **Émetteur :** "J'envoie la trame A/**0** avec le bit drapeau **0** ."

- Cette trame est reçue par Récepteur.
- **Récepteur :** "Trame d'acquittement ACK0 : bien reçu une trame avec le drapeau 0 ."
- Cette trame d'acquittement est reçue par Émetteur.
- Émetteur en déduit qu'il peut envoyer la trame suivante B avec un drapeau 1 .

- **Émetteur :** "J'envoie la trame B avec le drapeau 1 ."
- Cette trame est reçue par Récepteur.
- **Récepteur :** "Trame d'acquittement ACK1 : bien reçu une trame avec le drapeau 1 ."
- Par exemple cette trame est perdue.

- Émetteur n'a pas reçu d'acquittement. Au bout d'un certain délai :
- **Émetteur :** "J'envoie la trame B avec le drapeau 1 ."
- Cette trame est reçue par Récepteur.
- **Récepteur :** "Trame d'acquittement ACK1 : bien reçu une trame avec le drapeau 1 ."

- Cette trame d'acquittement est reçue par Émetteur.
- Émetteur en déduit qu'il peut envoyer la trame suivante avec un drapeau 0 .
- **Émetteur :** "J'envoie la trame C/**0** avec le drapeau **0** " et ainsi de suite.

- Ce protocole **numérote les trames et les acquittements** avec seulement **deux valeurs** 0 ou 1.
- Il gère aussi bien les **pertes de trames** que les **pertes d'acquittements**.
- Il gère bien les **petits retards de transmission** des trames et des acquittements. Mais l'émetteur retransmettra de multiples trames identiques **en cas de gros retards** de transmission d'une trame.

3 Interface homme – machine (IHM)

3.1 Définitions

- **Capteur** : c'est qui **dispositif de mesure dans le monde physique** et qui envoie une information à une entrée d'une carte informatique.
- **Actionneur** : c'est un dispositif qui reçoit une information d'une sortie d'une carte informatique et qui **agit sur le monde physique**.

Exemples de capteurs

Le capteur	mesure
Bouton poussoir	État (non appuyé ou appuyé)
Capteur de température	Température (°C)
Capteur de pression	Pression (N/m²)
Capteur d'accélération	Accélération (m/s²)

Exemples d'actionneurs

L'actionneur	agit sur
Écran	Lumière
Moteur	Force mécanique
LED	Lumière
Afficheur	Lumière

- **Cahier des charges** : c'est un document rédigé par un utilisateur. Il regroupe les descriptions des fonctionnalités à développer. Un certain nombre d'aller et retours (erreur – correction - erreur – correction etc.) sont nécessaires entre l'équipe de développement et l'utilisateur. Tous les types de manipulations doivent être envisagées, y compris les coupures de courant.

3.2 Exemple de réalisation d'une IHM en Python

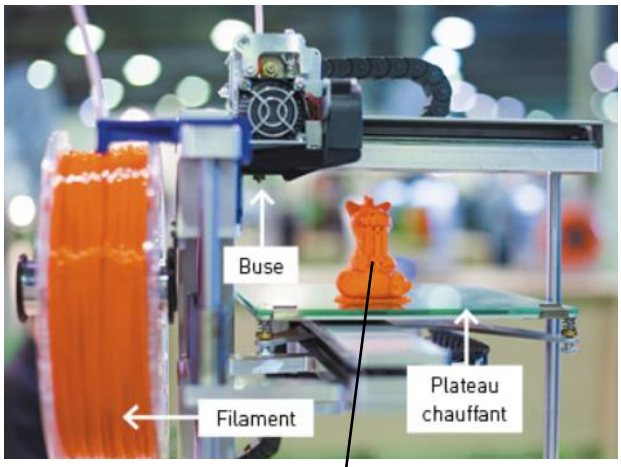
Cahier des charges

Voir l'aperçu dans cette vidéo astrovirtuel.fr/nsi/p116.mp4

Description de l'imprimante 3D

L'imprimante 3D utilise un filament pastique qu'elle fait fondre à travers une buse à haute température (180°C ou 230°C). La buse située sur un bras mobile dépose sur un plateau mobile le filament fondu.

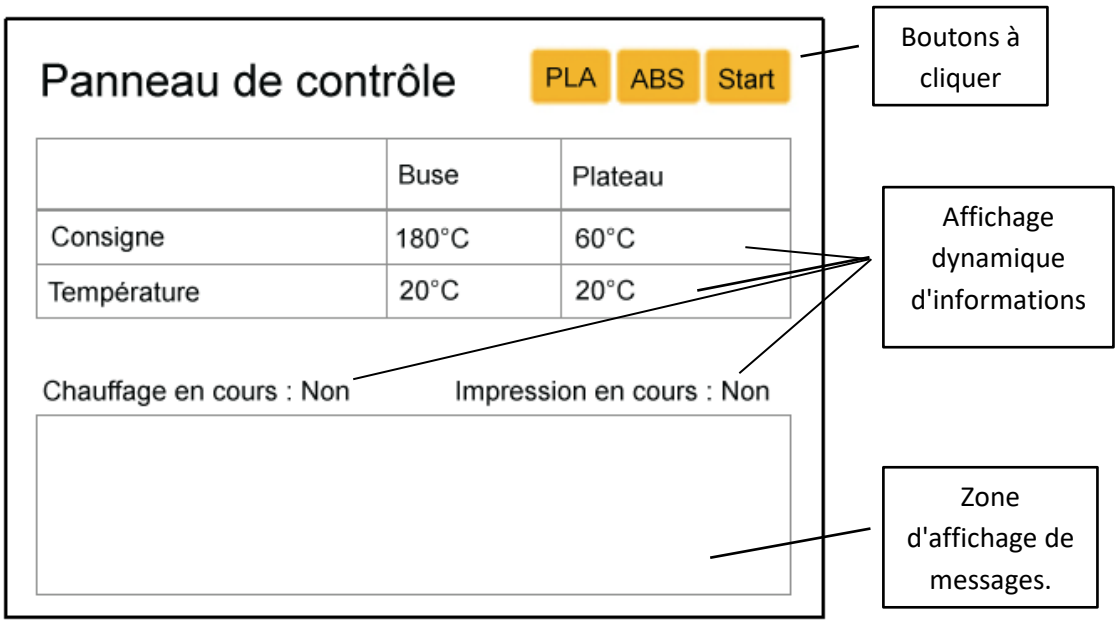
Le filament se solidifie en refroidissant. Mais le plateau chauffe lui aussi à une température moyenne (60°C ou 100 °C) pour maintenir une certaine souplesse du plastique déjà déposé, dans l'attente des futures couches.



L'objet à imprimer se forme peu à peu.

Description du panneau de contrôle

L'utilisateur souhaite une interface qui se présente sous la forme de ce panneau de contrôle :



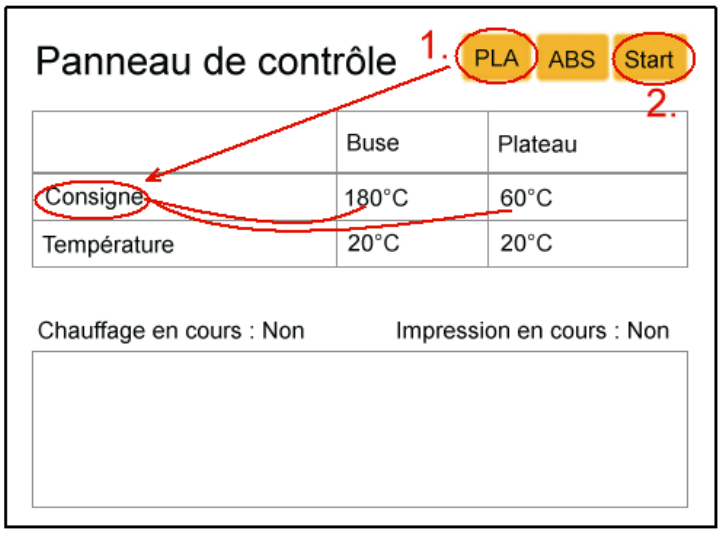
Description détaillée du fonctionnement

- **Sélectionner** (donc c'est une partie qui fonctionne dans le sens humain vers machine)

1. Sélectionner le matériau **PLA** ou **ABS** (deux types de plastique possibles). Chaque type a sa propre température de fusion au niveau de la buse et sa propre température de maintien au niveau du plateau. L'appui sur ces boutons règle les consignes de température de buse et de plateau à atteindre par l'imprimante une fois le départ lancé.

Si PLA a été choisi, les consignes à atteindre sont 180°C pour la buse et 60°C pour le plateau.

Si ABS a été choisi, les consignes à atteindre sont 230°C pour la buse et 100°C pour le plateau.



Les températures actuelles de la buse et du plateau sont affichées en dernière ligne du tableau.

- **Démarrer** (donc c'est une partie qui fonctionne dans le sens humain vers machine)

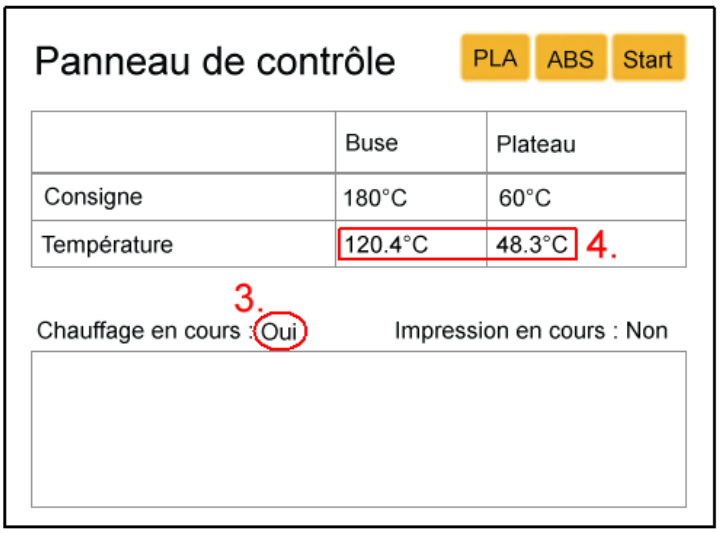
2. Cliquer sur le bouton **Start** provoque départ du chauffage de la buse et du plateau.

- **Être informé** (donc c'est une partie qui fonctionne dans le sens machine vers humain)

3. Dès qu'on a appuyé sur le bouton Start le processus démarre. Tout d'abord l'indication "Chauffage en cours" passe à Oui, l'indication "Impression en cours" reste à Non.

4. Les Consignes restent inchangées mais les températures actuelles sont fréquemment actualisées.

On peut suivre ainsi "en temps réel" l'augmentation de température du plateau et de la buse.

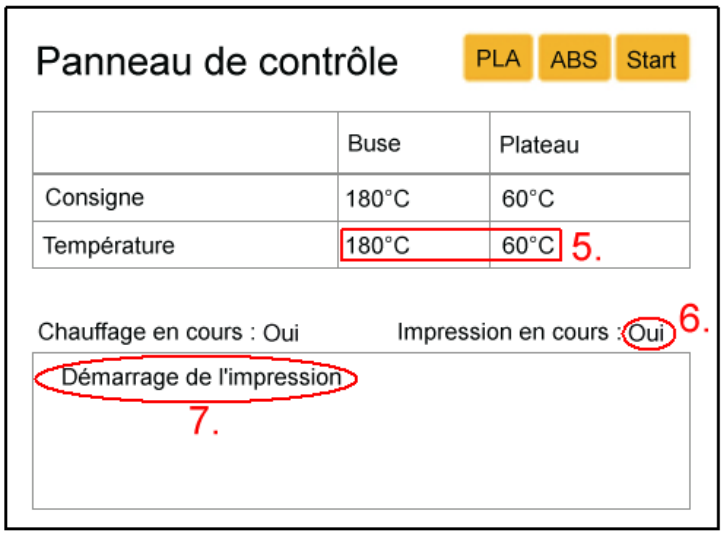


- **Être informé (**machine vers humain)

5. A un moment donné les deux consignes de températures Buse **et** Plateau sont atteintes.

6. L'imprimante *passe automatiquement* à l'impression 3D . L'indication "Chauffage en cours" reste à Oui, l'indication "Impression en cours" passe à Oui.

7. Au même moment, le message "Démarrage de l'impression" apparait dans la zone de texte.



Mise au point de l'Interface Homme-Machine

Vous avez à votre disposition p10 un canevas de programme Python. En particulier, sont déjà réalisées:

- Une **bibliothèque** `imprimante` avec **8 fonctions déjà existantes**. Elles sont donc directement utilisables dans les fonctions IHM que vous écrirez sous la forme `imprimante.fonction()`
- 5 **fonctions qui gèrent l'IHM** :
  - 3 des 5 fonctions sont déjà connectées aux 3 boutons de l'IHM (PLA, ABS, Start). Par exemple la fonction `fixer_temperatures_pla()` est appelée lorsque l'utilisateur appuie sur le bouton "PLA" à la condition que le chauffage soit éteint.
  - 1 fonction d'affichage d'un message dans la zone de texte du panneau de contrôle.
  - La fonction `loop()` ("boucle" en anglais) qui, lorsqu'elle est lancée, commence par lire les températures de consigne et les températures actuelles. Puis lorsque les températures actuelles atteignent ou dépassent les consignes, démarre l'impression et affiche un message. C'est elle qui assure la lecture de température "en temps réel".

Votre travail consiste à remplacer l'instruction pass pour **compléter successivement les trois fonctions IHM** :

1) fixer_temperatures_abs() 2) afficher_message() 3) demarrer_impression()

```
import imprimante

"""
Le module "imprimante" définit les fonctions suivantes :
 - reset() : Réinitialise l'imprimante.
 - fixer_temperature_cible(buse, plateau) : Assigne les températures cibles.
 - demarrer_chauffage() : Démarre le chauffage.
 - demarrer_impression() : Démarre l'impression.
 - lire_temperature() : Retourne un 2 uplet contenant les températures courantes
 de la buse et du plateau.
 - lire_temperature_cible() : Retourne un 2 uplet contenant les températures cibles
 de la buse et du plateau.
 - chauffage_en_cours() :  Retourne True si l'imprimante est en train de chauffer, False sinon.
 - impression_en_cours() :  Retourne True si l'imprimante est en train d'imprimer, False sinon.
"""

# Cette fonction est appelée quand l'utilisateur appuie sur le bouton "Start"
def demarrer_impression():
    pass


# Cette fonction est appelée une fois au démarrage de l'impression
def afficher_message():
    pass

# Cette fonction est appelée quand l'utilisateur appuie sur le bouton "ABS"
def fixer_temperatures_abs():
    pass


# Cette fonction est appelée quand l'utilisateur appuie sur le bouton "PLA"
def fixer_temperatures_pla():
    if not imprimante.chauffage_en_cours():
        imprimante.fixer_temperature_cible(180, 60)

message_affiche = False

# Cette fonction est appelée cycliquement, à la suite de la fonction setup
def loop():
    global message_affiche
    t_b_cible, t_p_cible = imprimante.lire_temperature_cible()
    t_b, t_p = imprimante.lire_temperature()
    if round(t_b, 1) >= t_b_cible and round(t_p, 1) >= t_p_cible and\
    imprimante.chauffage_en_cours() and not imprimante.impression_en_cours():
        imprimante.demarrer_impression()
    if imprimante.impression_en_cours() and not message_affiche:
        afficher_message()
        message_affiche = True
```

5 Fonctions de l'IHM

`global` permet de modifier dans la fonction une variable globale définie en dehors.

L'antislash "\" permet de couper une ligne trop longue.