Périphériques et IHM

Capteurs, actionneurs et interface homme-machine

Mise en situation

Un thermostat allume le chauffage quand la pièce est trop froide. Un lampadaire s'éclaire à la tombée de la nuit. Un aspirateur robot change de direction avant de toucher un mur. Dans ces trois objets, un petit ordinateur **mesure** quelque chose dans le monde réel, **décide** quoi faire, puis **agit** sur le monde réel. Dans le modèle de Von Neumann, ces échanges avec l'extérieur sont le rôle des **entrées-sorties**.

Les périphériques d'entrée et de sortie

Définition 1 **(Périphérique)** Un **périphérique** est un appareil relié à l'unité centrale, qui permet à l'ordinateur d'échanger des informations avec l'extérieur. Un **périphérique d'entrée** fait entrer une information dans l'ordinateur (clavier, souris, micro, caméra) ; un **périphérique de sortie** la fait sortir (écran, imprimante, haut-parleur).

Certains périphériques font les deux : un **écran tactile** affiche une image (sortie) et repère la position du doigt (entrée).

Un programme ne pilote pas directement le matériel : c'est le **système d'exploitation** qui gère les entrées-sorties grâce à ses pilotes. Quand un programme Python appelle `input()`, il demande au système le texte tapé au clavier ; quand il appelle `print()`, il lui demande d'afficher un texte à l'écran.

Les capteurs

Définition 2 **(Capteur)** Un **capteur** mesure une **grandeur physique** (température, lumière, distance, pression...) et la transforme en un signal électrique, que le système lit sous la forme d'un **nombre**. C'est un périphérique d'entrée.

Grandeur mesurée	Capteur	Exemple d'utilisation
Température	Thermomètre électronique (sonde)	Thermostat, station météo
Lumière	Capteur de luminosité (photorésistance)	Éclairage public, écran de téléphone
Distance	Capteur à ultrasons ou à infrarouge	Robot, aide au stationnement
Présence, mouvement	Détecteur de mouvement infrarouge	Éclairage d'un couloir, alarme
Contact, appui	Bouton, interrupteur, touche de clavier	Clavier, sonnette, butée d'un robot
Position du doigt	Surface tactile	Téléphone, borne de distributeur

Le clavier et l'écran tactile sont eux aussi des capteurs : chaque touche est un petit interrupteur, et la surface tactile mesure l'endroit où le doigt la touche. Pour le programme, une mesure n'est qu'un nombre : c'est au programmeur de savoir ce que ce nombre représente et dans quelle unité.

Les actionneurs

Définition 3 **(Actionneur)** Un **actionneur** reçoit une **commande** du programme, sous forme de signal électrique, et la transforme en une **action physique** : mouvement, lumière, son, chaleur. C'est un périphérique de sortie.

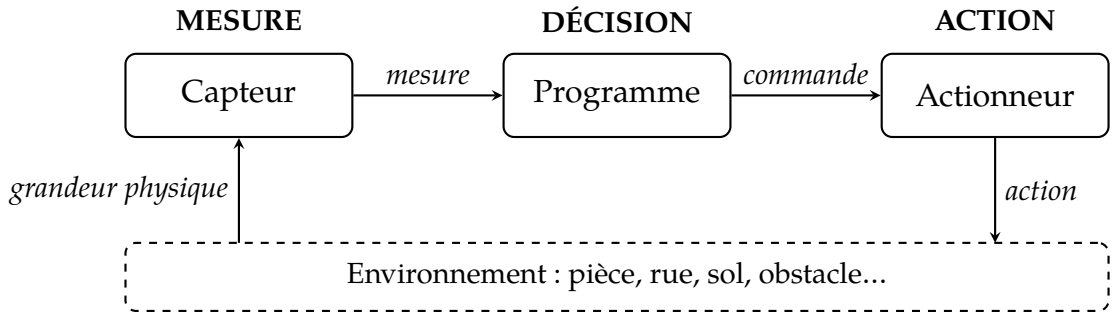
Actionneur	Action produite	Exemple d'utilisation
Moteur	Mouvement de rotation	Roues d'un robot, store électrique
Lampe, diode électroluminescente	Lumière	Éclairage, voyant d'un appareil
Haut-parleur, buzzer	Son	Alarme, signal d'un four
Résistance chauffante	Chaleur	Radiateur, bouilloire
Électrovanne	Ouverture ou fermeture d'un passage d'eau	Arrosage automatique, lave-linge

L'écran est aussi un actionneur particulier : il transforme les données envoyées par le programme en lumière, point par point.

La chaîne mesure, décision, action

Un système qui réagit à son environnement fonctionne toujours en trois étapes :

1. **mesure** : le capteur mesure une grandeur physique et fournit un nombre au programme ;
2. **décision** : le programme compare ce nombre à une valeur de référence et choisit une commande ;
3. **action** : l'actionneur exécute la commande, ce qui modifie l'environnement.



L'action modifie la grandeur mesurée : le chauffage réchauffe la pièce, donc la température lue au tour suivant change. C'est pourquoi la chaîne se répète **sans fin**, tant que l'appareil est sous tension.

Le programme qui réalise la décision s'exécute sur un petit ordinateur intégré à l'objet :

— un **système embarqué** est un ordinateur intégré à un appareil, qu'il commande : thermostat, lave-linge, voiture, montre ;

— un **objet connecté** est un système embarqué relié à un réseau, souvent Internet : il peut envoyer ses mesures ou recevoir des ordres à distance ;

— un **robot** est un système embarqué qui se déplace ou manipule des objets grâce à ses actionneurs.

Trois exemples courants

Système	Capteur	Décision	Actionneur
Thermostat	Sonde de température	température < consigne ?	Résistance chauffante
Éclairage automatique	Capteur de luminosité, détecteur de mouvement	luminosité < seuil et présence ?	Lampes
Robot aspirateur	Capteur de distance, capteur de contact	obstacle proche ? (avancer ou tourner)	Moteurs des roues

La **consigne** d'un thermostat est la température souhaitée, par exemple 19 °C. Pour tourner, le robot fait tourner ses deux roues à des vitesses différentes.

Programmer la chaîne en Python

Sur une carte programmable ou dans un robot, une bibliothèque fournit des fonctions qui lisent les capteurs et commandent les actionneurs. Sans matériel, on **simule** le capteur et l'actionneur par deux fonctions, ce qui permet d'écrire et de tester la partie la plus importante, la décision.

```
import random


def lire_temperature():
    """Capteur simulé : renvoie la température de la pièce, en degrés Celsius."""
    return random.randint(15, 25)


def commander_chauffage(allumer):
    """Actionneur simulé : allume le chauffage si allumer vaut True, l'éteint sinon."""
```

```
    if allumer:
        print("chauffage allumé")
    else:
        print("chauffage éteint")


def decider(temperature, consigne):
    """Renvoie True s'il faut chauffer : la pièce est plus froide que la consigne."""
    return temperature < consigne
```

La fonction `decider` ne dépend d'aucun matériel : on peut la tester avec des valeurs choisies.

```
assert decider(17, 19) == True     # trop froid : on chauffe
assert decider(21, 19) == False    # assez chaud : on ne chauffe pas
assert decider(19, 19) == False    # exactement la consigne : on ne chauffe pas
```

Le programme principal répète la chaîne mesure, décision, action, ici cinq fois :

```
consigne = 19
for i in range(5):
    temperature = lire_temperature()            # mesure
    chauffer = decider(temperature, consigne)   # décision
    print(f"Mesure {i + 1} : {temperature} °C,", end=" ")
    commander_chauffage(chauffer)               # action
```

Dans un vrai thermostat, la boucle ne s'arrête jamais : on écrit `while True`, et le programme attend un moment entre deux mesures. Chaque fonction a un seul rôle : pour faire tourner le programme sur un vrai thermostat, il suffit de réécrire les deux fonctions qui touchent au matériel, la décision ne change pas.

L'interface homme-machine

Définition 4 **(Interface homme-machine)** L'**interface homme-machine** (IHM) est l'ensemble des moyens par lesquels un utilisateur et une machine échangent des informations : les périphériques (boutons, écran, écran tactile…) et le programme qui affiche des informations et réagit aux actions de l'utilisateur.

Le panneau d'un four, l'écran tactile d'un distributeur de billets, la fenêtre d'une application ou une page web sont des IHM.

Le cahier des charges

Avant de programmer une IHM, on écrit son **cahier des charges** : ce que l'interface doit afficher, les actions que l'utilisateur peut faire et ce que le programme doit faire en réponse. Celui d'un thermostat :

1. l'utilisateur règle la consigne entre 15 °C et 25 °C, avec un curseur ; elle vaut 19 °C au démarrage ;
2. un bouton « Mesurer » lit le capteur de température ;
3. l'interface affiche la dernière température mesurée ;
4. l'interface affiche « Chauffage allumé » en rouge si la température est inférieure à la consigne, « Chauffage éteint » en bleu sinon.

La programmation événementielle

Un programme avec une interface graphique crée la fenêtre et ses composants, associe une fonction à chaque action possible de l'utilisateur, puis **attend** : c'est la **boucle d'événements**. Un clic sur un bouton est un **événement** ; le programme répond en appelant la fonction associée. C'est la **programmation événementielle**.

Une IHM avec tkinter

Composant ou méthode	Rôle
`tk.Tk()`	crée la fenêtre principale
`tk.Label(fenetre, text="...")`	affiche un texte (sortie)
`tk.Button(fenetre, text="...", command=f)`	un clic appelle la fonction `f` (entrée)
`tk.Scale(fenetre, from_=a, to=b)`	curseur pour choisir un entier entre `a` et `b` (entrée)
`composant.pack()`	place le composant sous le précédent
`curseur.get()`, `curseur.set(v)`	lit ou fixe la valeur du curseur
`etiquette.configure(text="...", fg="red")`	change le texte ou la couleur d'un `Label`
`etiquette.cget("text")`	renvoie le texte affiché par un `Label`
`fenetre.mainloop()`	lance la boucle d'événements

Le programme suivant répond au cahier des charges du thermostat (à exécuter dans Thonny).

```
import tkinter as tk
import random


def lire_temperature():
    """Capteur simulé : renvoie la température de la pièce, en degrés Celsius."""
    return random.randint(15, 25)


def mesurer():
    """Lit le capteur, compare à la consigne, puis affiche la mesure et l'état."""
    temperature = lire_temperature()
    texte_mesure.configure(text=f"Température mesurée : {temperature} °C")
    if temperature < curseur.get():
        texte_chauffage.configure(text="Chauffage allumé", fg="red")
    else:
        texte_chauffage.configure(text="Chauffage éteint", fg="blue")


fenetre = tk.Tk()
fenetre.title("Thermostat")

# Point 1 : curseur de 15 à 25, réglé sur 19 au démarrage
curseur = tk.Scale(fenetre, label="Consigne (°C)", from_=15, to=25,
                   orient="horizontal", length=250)
curseur.set(19)
curseur.pack()

```

```
29 # Point 2 : le bouton appelle la fonction mesurer
30 bouton = tk.Button(fenetre, text="Mesurer", command=mesurer)
31 bouton.pack()
32
33 # Points 3 et 4 : deux textes, modifiés par la fonction mesurer
34 texte_mesure = tk.Label(fenetre, text="Température mesurée : aucune mesure")
35 texte_mesure.pack()
36 texte_chauffage = tk.Label(fenetre, text="Chauffage éteint", fg="blue")
37 texte_chauffage.pack()
38
39 fenetre.mainloop()
```

Le clic sur « Mesurer » est un événement, qui appelle `mesurer` ; cette fonction lit le capteur (mesure), compare la température à la valeur du curseur (décision), puis modifie les deux textes (action sur l'interface). Dans `command=mesurer`, on écrit le nom de la fonction **sans parenthèses** : avec `command=mesurer()`, la fonction serait appelée une seule fois, à la création du bouton.

Que retenir ?

A minima…

— Un **périphérique d'entrée** fait entrer une information dans l'ordinateur, un **périphérique de sortie** la fait sortir.

— Un **capteur** mesure une grandeur physique et la transmet au programme sous forme de nombre ; un **actionneur** transforme une commande du programme en action physique.

— Un système qui réagit à son environnement répète sans fin la chaîne **mesure, décision, action**.

— Une **IHM** est l'ensemble des moyens par lesquels un utilisateur et une machine échangent des informations. On la conçoit à partir d'un **cahier des charges**.

Au mieux…

— Identifier, dans un objet courant, les capteurs, les actionneurs et la décision prise par le programme.

— Écrire en Python la décision sous forme d'une fonction qui reçoit les mesures et renvoie la commande, puis la tester avec `assert`.

— Programmer avec `tkinter` une IHM qui répond à un cahier des charges : composants d'entrée, composants de sortie, fonctions appelées lors des événements, boucle d'événements.