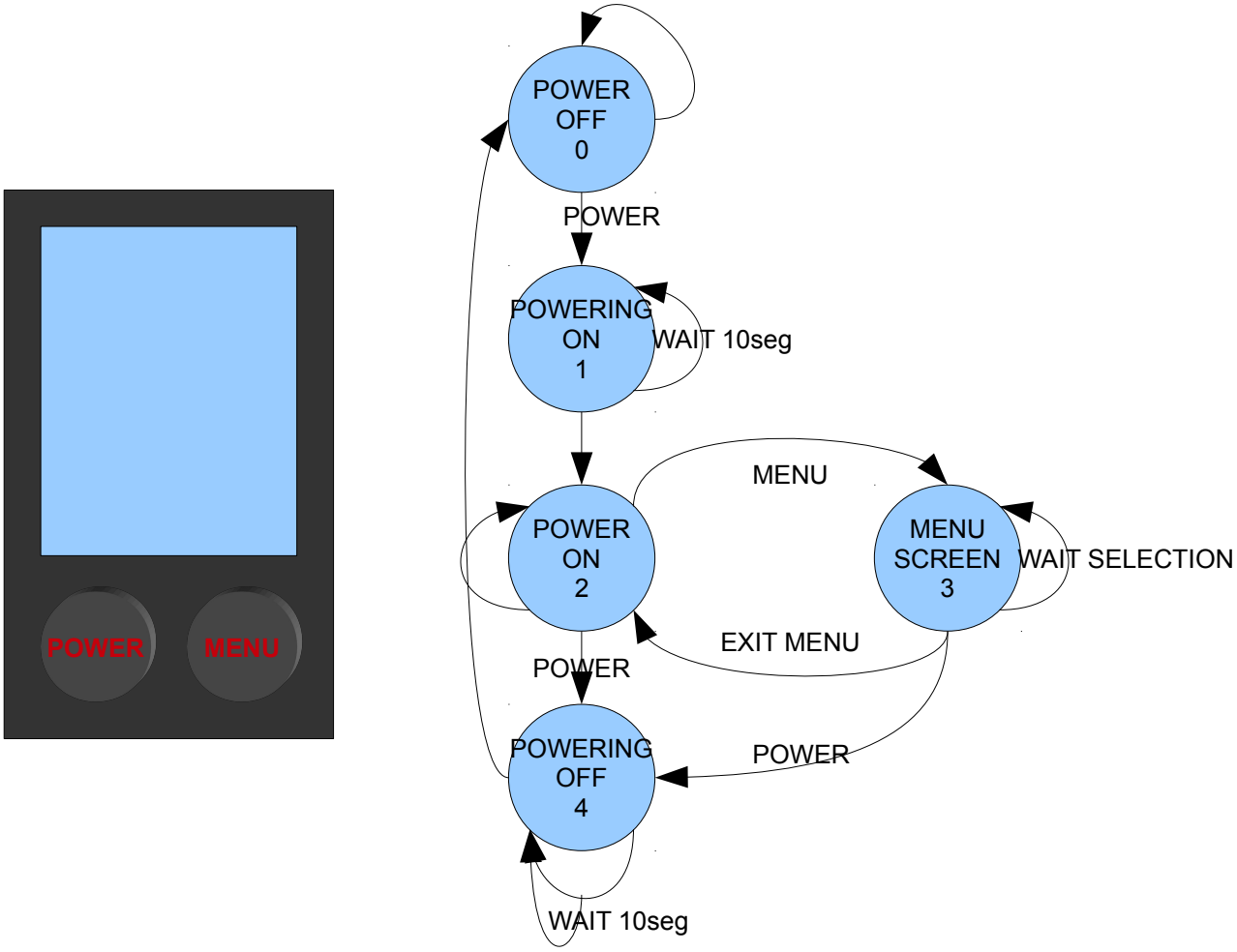
L'objectiu no és explicar tota la teoria de les màquines d'estat o aprofundir-hi, sinó veure'n els elements més bàsics que ens poden servir per simplificar la manera d'afrontar la pràctica.
En aquest sentit ho explicaré de manera molt planera i saltant-me la majoria de normes de nomenclatura i de procediments, si algú vol aprofundir hi ha moltíssima informació disponible.

L'idea de les màquines d'estat és definir el funcionament d'alguna cosa (un programa, una màquina de cafè, un cotxe, ...) a partir de dos elements:

- Estats: són situacions ben definides en les que espot estar. Per exemple el nostre GPS pot estar apagat, o pot estar ensenyant el menú,...
- Transicions: són els elements que defineixen els diferents salts entre estats que podem realitzar. Les transicions tenen asociades les accions o desencadenants que provoquen el canvi d'estat.

Mirem-ho en un exemple simplificat de la pràctica, on només hi ha l'opció d'encendre/apagar i anar al menú:



Cada cercle representa un estat en el que es pot trobar el GPS, i conté el nom de l'estat i el número d'estat (l'utilitzarem en el codi).
Les flexes representen les transicions, els camins entre estats que podem realitzar. A part indiquen quan es seguirà aquest camí (al apretar un botó, al realitzr-se una acció concreta,...). Si no hi ha res indicat vol dir que es saltarà automàticament al següent estat indicat tan aviat com sigui possible.

Amb aquesta màquina d'estats podem veure que:

- Per defecte estem a l'estat 0 (apagat) i que ens quedarem aquí (fletxa sense res) fins que

apretem el botó POWER. Al apretar POWER anem a l'estat 1.

- Estat 1 (encenent l'equip): Aquí farem apareixer el logo a la pantalla i ens hi estarem 10 segons. Al finalitzar aquest temps saltarem automàticament a l'estat 2.
- Estat 2 (equip encès): ens quedarem en aquest estat fins que:
  - Apretem el botó MENÚ: saltem a l'estat 3
  - Apretem el botó POWER: saltem a l'estat 4
- Estat 3 (menú): pintem el menú en pantalla i esperem que seleccionin l'opció a executar (en aquest cas el menú només té l'opció de sortir del menú). Podem triar:
  - Sortir del menú: saltem a l'estat 2
  - Apagar el GPS: saltem a l'estat 4.
- Estat 4 (apagant): es pot fer apareixer un missatge o una latra vegada el logo i passats 10 segons salta directament a l'estat 0

Per tant aquí tenim l'interpretació d'aquest esquema, es veu clar que amb un parell de dibuixets ens estalviem molta lletra, oi?

El següent pas es transformar això en codi, en el nostre cas en Processing. Els elements mínims que utilitzarem serà una variable **state** que ens indicarà l'estat actual, i una estructura de control vasada en l'estrucutra condicional switch (http://processing.org/reference/switch.html). Escric un codi orientatiu (amb molts errors):

```
int state = 0; //Inicialitzem la variable a l'estat per defecte que volguem.

/*
 * Dins de draw comprovem en quin estat estem i pintem tots els elements associats a aquest estat.
*/
draw (){
switch (state){
  case 0:
    dibuixem el GPS apagat
    break;
  case 1:
    afegim el logotip
    si el cronometre >= 10 seg, fem state = 2
    break;
  case 2:
    pintem el GPS encès
    break;
  case 3:
    pintem el menú a la pantalla (o hi fiquem els botons,...)
    break;
  case 4:
    dibuixem missatge de comiat
    si el cronometre >= 10 seg, fem state = 0
    break;
}
}
```

```
/*
 * A part de la funció draw() tenim totes les funcions associades a controls
*/
funció associada al botó POWER(){
//mirem on podem deixar que s'apreti el botó de power i actuem en conseqüència
// en el nostre cas el podem apretar si estem als estats 0, 2 i 3, en la resta de casos no fem res.
switch (state){
  case 0: state = 1 i inicio el cronòmetre.
  case 2: state = 4 i inicio el cronòmetre.
  case 3: state = 4 i inicio el cronòmetre.
}
}

funció associada al MENU(){
// només la podem utilitzar si estem a l'estat 2
if (state == 2){
 state = 3;
}
}
```

Afegim la resta de funcions que depenen de controls (power, menu, exit menu)

El problema d'aquest sistema és que a la funció draw() hi ha moltíssim codi, però té una solució molt senzilla: en comptes de ficar tot el codi dins de cada "case x:", crearem una funció associada a cada estat i la cridarem. Quedaria una cosa d'aquest estil:

```
draw(){
  case (...){
    case 0: funcioEstatPowerOff();
    case 1: funcioEstatPoweringOn();
    ....
  }
}

I afegim aquestes funcions fora de draw()
funcioEstatPowerOff(){
 dibuixem el GPS apagat
}

funcioEstatPoweringOn(){
 afegim el logotip
 si el cronometre >= 10 seg, fem state = 2
}
```

Per tant ens quedarien 3 blocs ben diferenciats:

- draw(): hi ha el llistat de tots els estats que podem tindre, i dins de cada estat crida la funció que s'encarrega de pintar el que calgui
- bloc de les funcions de pintat que es criden des de dins de draw()
- bloc de les funcions associades als controls que ens fan saltar entre estats.