

# TD 2

## Exercice 1

	 <p>nbs</p>  
---	--

### 1) Liste des évènements

Clic sur Stop / Start / Tic

### 2) Listes des actions

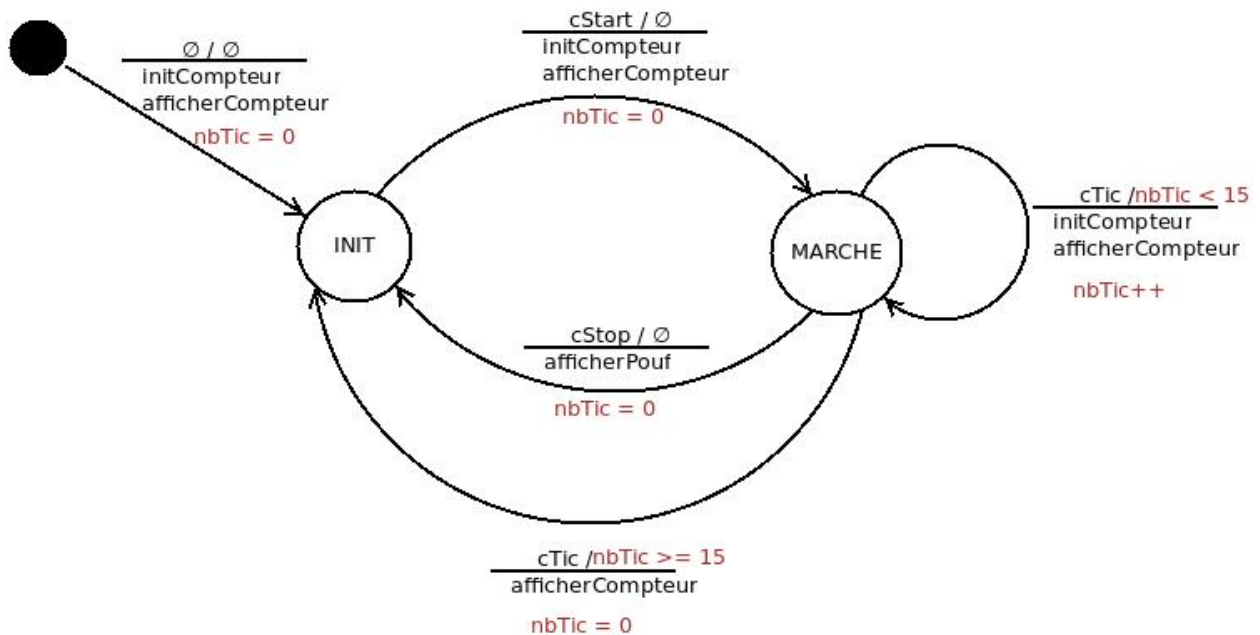
afficherPouf()

afficheCompteur()

activationInit()




activationMarche()

### 3) Automate






	startBtn	stopBtn	clicBtn
<b>INIT</b>	S = MARCHE nbTic = 0 afficherCompteur() activationOn();	⊖	⊖
<b>ON</b>	⊖	S = INIT; nbTic = 0 affichePouf() activationInit()	if nbTic < 15 then S = MARCHE nbTic++ afficherCompteur(); activationOn() else S = INIT nbTic = 0 affichePouf(); activationInit(); end if

Léo :

	startBtn	stopBtn	ticBtn
<b>INIT</b>	<pre>S = MARCHE nbTic = 0 afficherCompteur() activationMarche();</pre>		
<b>ON</b>		<pre>S = INIT; nbTic = 0 affichePouf() activationInit()</pre>	<pre>if nbTic &lt; 15 then S = MARCHE nbTic ++ afficherCompteur(); activationMarche(); else S = INIT nbTic = 0 affichePouf(); activationInit(); end if</pre>

#### 4) Matrice des états/événements

	cStart	cStop	Timer
<b>INIT</b>	S = MARCHE nbTic = 0 initCompteur() afficherCompteur		
<b>MARCHE</b>		S = INIT; nbTic = 0 affichePouf()	si nbTic < 15 alors S = MARCHE nbTic ++ else S = INIT nbTic = 0 afficherCompteur()

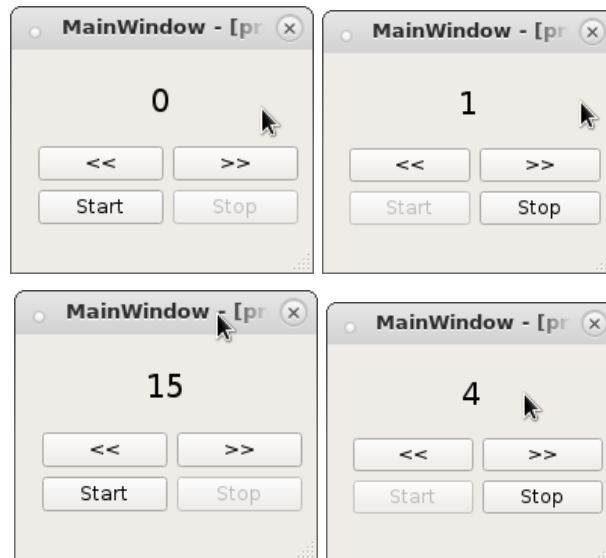
init :

```
s = INIT;
nbTic = 0;
initCompteur;
afficherCompteur();
```

#### 5) Pseudo-Code

<pre> Eventhandler CStart switch S     case INIT         S = MARCHE;         nbTic = 0;         activationMarche()         initCompteur();         afficheCompteur();     case MARCHE :         throw new RuntimeException(); </pre>	<pre> Eventhandler CStop switch S     case INIT         throw new RuntimeException();     case MARCHE :         s = INIT;         nbTic = 0;         affichierPouf();         activationINIT(); </pre>
--	--

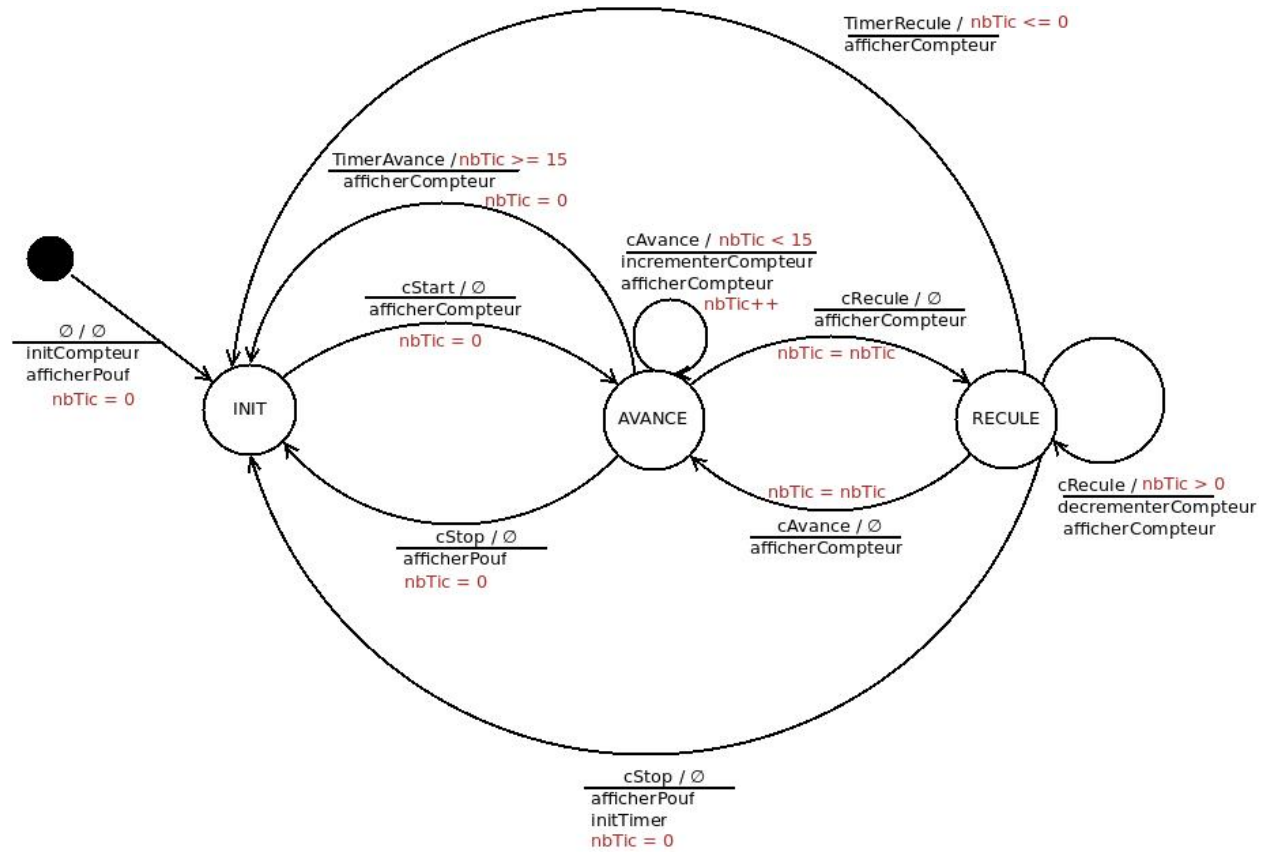
## Exercice 2 :



1/ ClicStart / ClicStop / ClicAvance / ClicRecul / TimerAvance, TimerRecul

2/ **Actions** : IncrementerCompteur, decrementerCompteur, initCompteur, afficherPouf, afficherCompteur

3/ **Automate**














#### 4) Matrice Etat/Evt

Etat INIT :

s = INIT

nbTic = 0

afficheCompteur

	cStart	cStop	cAvance r	cReculer	TimerInc	TimerDec
INIT	s = AVANCE activerIncr nbTic = 0 initCompteur afficheCompteur avanceActivation();					
AVANCE		s = INIT activerINIT afficheCompteur nbTic = 0; initActivation();		S = RECULE activerDecr afficheCompteur nbTic = nbTic reculerActivation	if nbTic < 15 nbTic++ incrCompteur afficheCompteur avanceActivation else afficheCompteur s = INIT activerINIT nbTic = 0 initActivation end if	 t
RECULE		s = INIT activerINIT afficheCompteur nbTic = 0 initActivation();	S = AVANCE activerIncr afficheCompteur nbTic = nbTic avanceActivation			if nbTic > 0 nbTic-- decrCompteur afficheCompteur reculerActivation else S = INIT activerINIT afficheCompteur nbTic = 0 initActivation

```
cStartActionPerformed
switch (s) {
    case INIT:
        s = AVANCE;
        nbTic = 0;
        break;
    case AVANCE:
    case RECULE:
}
```

```
activation
cStart.setEnabled(false);
cStop.setEnabled(true);
cAvance.setEnabled(false);
cReculer.setEnabled(true);
timerAvance.start();
```

```
timerReculé.stop();
```