Ecole Nationale Supérieure des Mines de Rabat

Département Informatique

RAPPORT DU PROJET :

SIMULATION DU CHAVEVHEMENT DES ACTIVITES DES CLIENTS AU SUPERMARCHE

Réalisé par :

ANOUK Oumnia

CHANA Housna

LARJIL Taha

MEZGOUR Yassine

ZEKRI Khadija

Encadré par :

Mme. OUAZZANI Khadija

Année Universitaire 2020-2021

Remerciement

Nous tenons à exprimer nos sincères remerciements à Madame Khadija OUZZANI TOUHAMI, Professeur au département informatique à l'Ecole Nationale des Mines de Rabat, pour cette occasion de mettre en pratique ce qu'elle nous a appris et pour le grand bénéfice des informations fournies pour notre formation.

Aussi, nos vifs remerciements pour son temps, ses efforts et ses qualités humaines et professionnelles.

Enfin, nous tenons à exprimer nos sincères satisfactions à l’égard du travail que notre équipe à entamer. Nous espérons continuer à trouver le même sérieux et professionnalisme lors de nos prochaines collaborations.

Table de matières

Table de figures

Table de tableaux

Introduction

La simulation permet de résoudre des problèmes concrets de façon efficace. Il s'agit d'une méthode d'analyse importante et qui est facile à vérifier, à communiquer et à comprendre. Elle consiste à construire une abstraction de la réalité ou modèle, et de faire évoluer cette abstraction en fonction du temps. Le modèle peut être mathématique ou, et surtout, logique. Il peut être également déterministe ou, et surtout, stochastique.

Le moyen le plus simple serait de tenter l'expérience, c'est-à-dire d'exercer l'action souhaitée sur l'élément en cause pour pouvoir observer ou mesurer le résultat. Dans de nombreux cas l'expérience est irréalisable, trop chère ou contraire à l'éthique. Nous avons alors recours à la simulation : rechercher un élément qui réagit d'une manière semblable à celui que l'on veut étudier et qui permettra de déduire les résultats.

Quels que soient les secteurs et les disciplines, la simulation offre des solutions précieuses en permettant d'avoir une vision claire de systèmes complexes.

1. Mise en contexte :

1.1. Problématique :

L'attente est l'un des enjeux majeurs du commerce. En effet, plusieurs consommateurs seraient prêts à renoncer à leur achat s'ils jugent l'attente trop importante.

L'exigence des gens a été déformée par leurs expériences d'achat dans le monde virtuel, jusqu'au point d'avoir perdu aujourd'hui toute patience pour faire la queue. Les recherches indiquent que un nombre important des visiteurs ont déjà quitté un magasin à cause des files jugées trop longues ; mais il est encore plus inquiétant que plus que la moitié d'entre eux admettent avoir choisi un magasin concurrent peu après pour faire leurs courses.

Dans le cadre d'étude des chevauchements des activités clients au super marché, la direction du super marché a prévu 2 caisses, mais demande aux clients de former une seule queue pour payer, et si les 2 caisses sont libres, un client passe toujours à la caisse 1. Si à son arrivée un client trouve plus d'une personne en attente devant les caisses, il ne rentre pas au super marché et se dirige ailleurs. C'est un client perdu pour le super marché.

1.2. Objectifs de l'étude :

Ce projet vise à mesurer l'impact de l'ajout d'une 3ème caisse sur le nombre de clients perdu en se focalisant sur l'optimisation du taux d'occupation des ressources, la réduction des temps d'attente et l'examination de l'ajout d'un nouvel investissement. Pour cela la simulation sera faite avec 2 caisses et avec 3 caisses. Il s'agit de quantifier le retour attendu pour pouvoir le comparer aux résultats qui seront obtenus.

Nos prévisions porteront sur la simulation d'une journée de 9h du matin à 9h du soir (720 min), puis nous allons essayer à travers la simulation d'étudier un échantillon de taille n = 40 simulations de 40 journées selon les 8 indicateurs suivants :

- ✓ Le nombre de clients entrés dans le super marché (NCE)
- ✓ Le nombre de clients perdus (NCP)
- ✓ Le temps d'attente moyen pour payer à la caisse (TATmoy)
- ✓ Le temps de séjour moyen dans le système (TSmoy)
- ✓ Le taux ou ratio d'utilisation de la caisse 1 (TauC1), par rapport au temps total simulé
- ✓ Le taux ou ratio d'utilisation de la caisse 2 (TauC2)

✓ Le taux ou ratio d'utilisation de la caisse 3 (TauC3), dans le cas de 3 caisses.

✓ Le ratio de temps, par rapport au temps total simulé, pendant lequel on a au plus un client dans la file d'attente, et celui pendant lequel on a plus d'un client dans la file.

1.3. Définition du système :

Nous commençons notre simulation par la représentation du système, alors nous intéressons dans ce projet par l'étude d'un super marché qui se caractérise par les entités suivantes :

- **Client :** il est défini par le temps de son arrivé, le temps de magasinage, le temps passé à la queue et le temps de paiement.
- **Caisse :** elle est décrite par son état (occupée ou vide), et le nombre de clients servis dans chaque caisse.

Puisque les clients arrivent dans le système à des temps différents, il est clair que le comportement de celui-ci sera décrit par une collection de processus, un pour chaque entité, dont certain peuvent chevaucher.

Donc nous se préoccupons par trois types d'événements :

➢ Arrivée ⟶ type « A »

➢ Fin magasinage ⟶ type « FM »

➢ Fin paiement ⟶ type « FP »

A chaque type d'événement nous associons une fonction que nous exécuterons en cas de réalisation de cet événement.

Cette fonction doit spécifier, dans le cas où l'événement en question survient :

• les traitements à faires

• les nouveaux événements à planifier.

Avec cette approche nous serons capables de modéliser le processus décrivant la progression d'une entité dans le système en soulignant l'interaction entre les processus.

2. Construction des modèles :

Formuler un modèle qui renferme suffisamment de détails et qui colle suffisamment à la réalité de sorte que les hypothèses faites lors de la construction du modèle peuvent être déduites du système est l'un des étapes les plus importantes lors de la simulation.

Dans la suite de ce travail, seuls les modèles à événements discrets seront utilisés.

2.1. Modèle logique :

Les modèles logiques contiennent les liens logiques entre les éléments du système, et les variables exogènes affectant le système.

2.1.1. Scénario 1 : Deux caisses :

Une modélisation simplifiée de ce problème, avec 2 caisses, peut être la suivante :

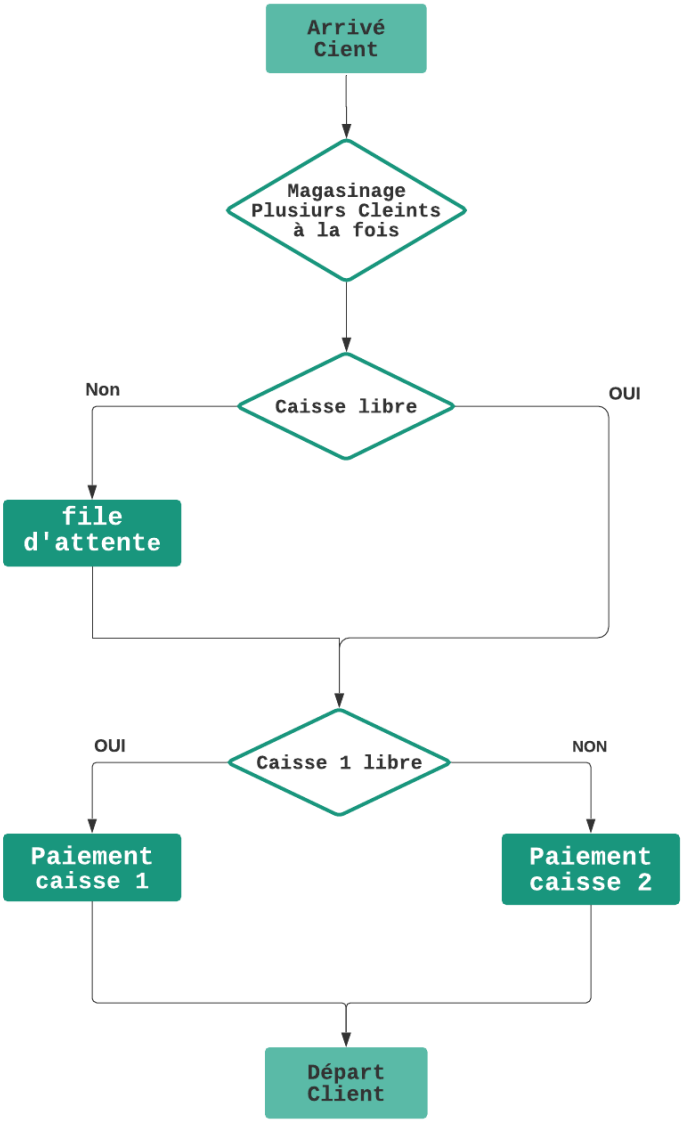

Figure 1 : Modèle logique : le cas de 2 caisses

2.1.2. Scénario 2 : Trois caisses :

Une modélisation simplifiée de ce problème, avec 3 caisses, peut être la suivante :

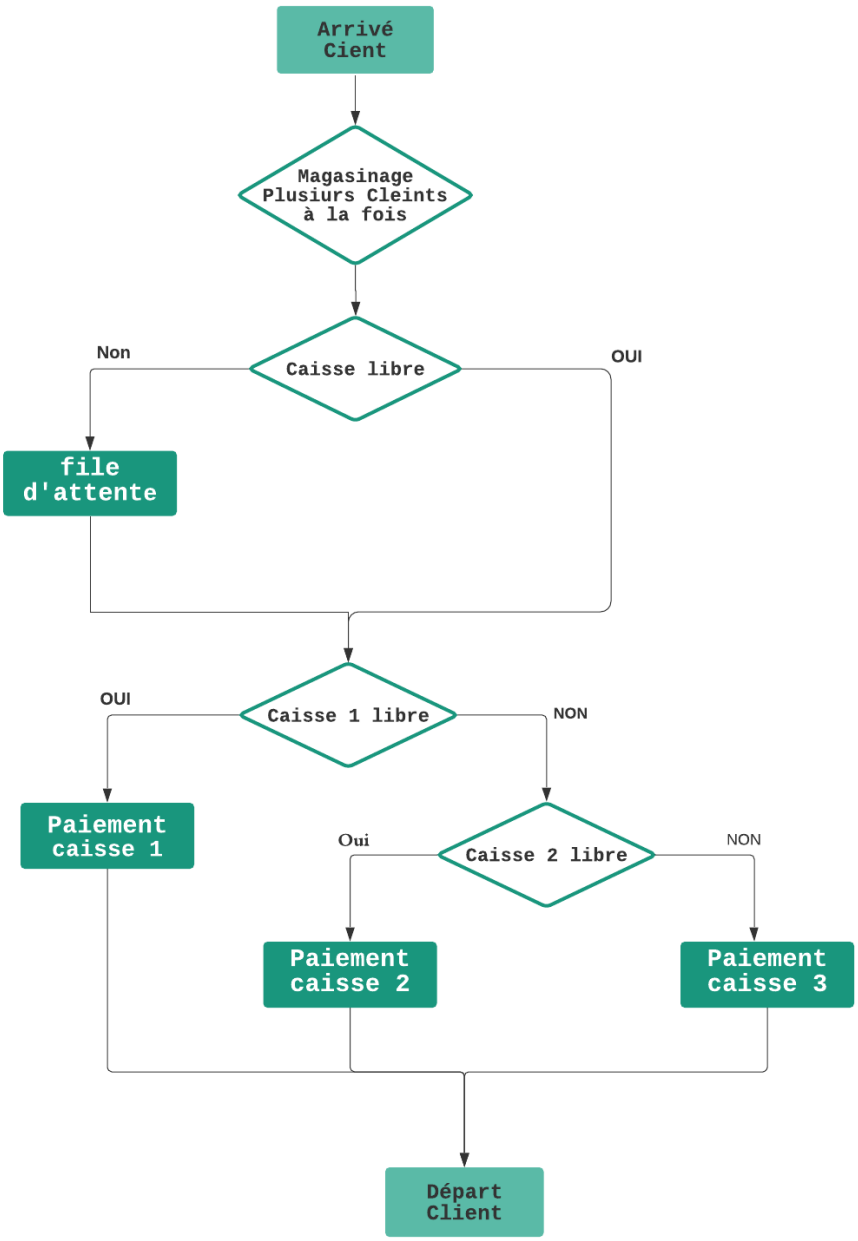

Figure 2 : Modèle logique : le cas de 3 caisses

2.2. Modèle mathématique :

Ce modèle nous permet d'identifier les entités du système et leurs attributs par des variables, et les activités par des fonctions mathématiques. Cependant, tous les facteurs ne sont pas toujours mesurables ni exprimables sous forme de variables.

Dans le système que nous étudions, nous devons disposer d'une liste d'événements, ou calendrier d'événements, contenant à tout instant :

- Un inventaire des événements prévus dans le futur
- La nature et la date de réalisation de chacun de ces événements

Des blocs d'instructions sont associés aux différents types d'événements. Ces instructions doivent :

- Préciser les changements qui peuvent affecter l'état du système au moment de la réalisation d'un événement
- Éventuellement planifier la réalisation de nouveaux événements dans le futur

Avec cette approche, l'horloge centrale de la simulation est avancée jusqu'au moment où l'événement le plus proche se produit, et on exécutera alors les instructions du bloc correspondant à cet événement. L'horloge avance donc par des pas d'inégales amplitudes, ou période variable.

Le temps qui sépare les arrivées successives de clients est distribué suivant le tableau :

Temps qui sépare les arrivées						
Temps (min)	**1**	**2**	**3**	**4**	**5**	**6**
Fréquence (%)	**30**	**50**	**10**	**5**	**3**	**2**
Cumul	**0.3**	**0.8**	**0,9**	**0,95**	**0.98**	**1**
Intervalle	**[0,0.3[**	**[0.3,0.8]**	**]0.8,0.9]**	**]0.9,0.95]**	**]0.95,0.98]**	**]0.98,1]**

Tableau 1 : Temps qui sépare les arrivés

Le temps de magasinage par client suit la distribution :

Temps de magasinage					
Temps (min)	**2**	**4**	**6**	**8**	**10**
Fréquence (%)	**10**	**20**	**40**	**20**	**10**
Cumul	**0.1**	**0.3**	**0,7**	**0.9**	**1**
Intervalle	**[0,0.1[**	**[0.1,0.3[**	**[0.3,0.7]**	**]0.7,0.9]**	**]0.9,1]**

Tableau 2 : Temps de magasinage

Le temps passé à la caisse, temps d'attente exclu, est distribué selon le tableau :

Temps passé à la caisse				
Temps (min)	**1**	**2**	**3**	**4**
Fréquence (%)	**20**	**40**	**25**	**15**
Cumul	**0,2**	**0,6**	**0,85**	**1**
Intervalle	**[0,0.2[**	**[0.2,0.6]**	**]0.6,0.85]**	**]0.85,1]**

Tableau 3 : Temps de paiement

Pour le nombre moyen de clients perdus (NCP), avec le super marché, on prendra un échantillon de taille n= 40, 40 journées simulées par exemple, et on calculera la moyenne X de cet échantillon. La moyenne recherchée μ sera estimée comme étant X ± ε.

Dans le cas où la moyenne μ de toute une population est difficile à calculer, pour une raison ou pour une autre, on peut estimer cette moyenne par un intervalle qui pourrait la contenir avec une certaine probabilité ou confiance. Cet intervalle est appelé l'intervalle de confiance de la moyenne.

Pour estimer le nombre moyen de clients perdus par jour à travers un intervalle de confiance de la moyenne. Pour cela :

- On procédera à l'étude d'un échantillon de 40 jours, afin de pouvoir utiliser le théorème de la limite centrée avec une loi normale.
- Estimer pour chaque journée i la valeur de Xi ou, dans ce cas, NCPi
- Calculer la moyenne X ou, dans ce cas, NCP, de l'échantillon pour ce paramètre, comme étant la somme des NCPi de chaque journée simulée, divisée par n le nombre de journées simulées.
- Calculer l'estimateur S de σ, selon la formule précédente, en utilisant la somme des NCPi 2 et la moyenne X ou NCP des NCPi.
- Estimer l'IC pour un α donné, auquel correspond un Zα/2 donné, en utilisant cette formule : $IC = [X - 1.96 * S \sqrt{n}, X + 1.96 * S \sqrt{n}]$

3. Programmation et validation des modèles :

3.1. Choix de langage :

Notre programme est écrit en langage Python, c'est un langage de programmation interprété, multiparadigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet.

Le choix de ce langage se pose d'une part du fait qu'il offre de grandes possibilités en analyse et modélisation de données scientifiques avec relativement peu de charge de travail en termes d'apprentissage, d'installation ou de temps de développement et d'autre part parce qu'il nous permet de se focaliser sur ce que nous devons faire plutôt que sur la façon dont nous devons le faire.

3.2. Les fonctions du programme de simulation :

Notre programme s'appuie essentiellement sur plusieurs **méthodes**, nous commençons par définir la fonction « Alea », pour générer les données de la simulation, par rapport aux différentes variables aléatoires du problème.

```
def alea(IX, IY,IZ):
    IX[0] = 171 * (IX[0]%177) - 2 * (IX[0] // 177)
    IY[0] = 172 * (IY[0] % 176) -35*(IY[0] // 176)
    IZ[0]  = 170 * (IZ[0] % 178)- 63*(IZ[0] // 178)
    if(IX[0] < 0):
        IX[0] = IX[0] + 30269
    if (IY[0] < 0):
        IY[0] = IY[0] + 30307
    if (IZ[0] < 0):
        IZ[0] = IZ[0] + 30323
    inter = (IX[0] / 30269) + (IY[0] / 30307) + (IZ[0] / 30323)
    return(inter - int(inter))
```

Figure 3 : Fonction Alea

Puis, nous définissons l'unité de temps appropriée au problème, dans notre cas on simule sur une période de **40 jours,** cette dernière sera accompagnée par une **horloge** qui sera initialisé a 0 et en cours de chaque type de fonctionnement prend un pas qui dépend du résultat de la fonction alea.

```
for j in range(40): #simulation 40 jours
    H=0
    i=[1]
    LQ=[0]
    NCP=[0]
    NCE=[0]
    C1=[0]
    C2=[0]
    QfileIntervall = []

    File = []
    clients = [] #pour calculer TSmoy
    ClientsQ =[] #pour calculer TATmoy
    TC1=[0]
    TC2=[0]
    IX[0]=IX[0]+j*5
    IY[0]=IY[0]+j*5
    IZ[0]=IZ[0]+j*5

    calendrier = [[1,'A',H+selectionnerF1(alea(IX,IY,IZ))]]
```

Figure 4 : Simulation par période fixe

```
def selectionnerF1(alea):
    if alea < 0.3 and alea >=0 :
        return 1
    elif alea <= 0.8 and alea >=0.3:
        return 2
    elif alea <= 0.9 and alea >0.8:
        return 3
    elif alea <= 0.95 and alea >0.9:
        return 4
    elif alea <= 0.98 and alea > 0.95:
        return 5
    else :
        return 6
```

Figure 5 : Le pas associer à l''arrivée d'un nouveau client

```
def selectionnerF2(alea):
    if alea < 0.1 and alea >=0 :
        return 2
    elif alea <0.3 and alea >=0.1:
        return 4
    elif alea <= 0.7 and alea >=0.3:
        return 6
    elif alea <= 0.9 and alea >0.7:
        return 8
    else :
        return 10
```

Figure 6 : le pas associer au fin magasinage

```
def selectionnerF3(alea):
    if alea < 0.2 and alea >=0 :
        return 1
    elif alea <= 0.6 and alea >=0.2:
        return 2
    elif alea <= 0.85 and alea >0.6:
        return 3
    else:
        return 4
```

Figure 7 : Le pas associer à la fin de paiement

Après fixer la période souhaiter nous définissons un calendrier **d'événements** dont nous allons avoir un inventaire des événements prévus dans le futur, la nature et la date de réalisation de chacun de ces événements.

Avec cette approche, l'horloge centrale de la simulation est avancée jusqu'au moment où l'événement le plus proche se produit, et on exécutera alors les instructions du bloc correspondant à cet événement.

```
calendrier = [[1,'A',H+selectionnerF1(alea(IX,IY,IZ))]]
while (calendrier!=[]):
    client = selectionner(calendrier)
    H = client[2] #maj
    if client[1] == 'A':
        arrive(client[0], NCE, NCP,i,LQ,H,calendrier,IX,IY,IZ,clients)
    elif client[1] == 'FM':
        finmagasinage(client[0], File,C1, C2, LQ,H,calendrier,ClientsQ,TC1,TC2,IX,IY,IZ,QfileIntervall)
    else:
        clients = findclient(clients,client)
        finpaiment(client[0], LQ, File, C1, C2,H,calendrier,TC1,TC2,ClientsQ,IX,IY,IZ,QfileIntervall)
```

Figure 8 : Approche d'événement

Après remplissage de calendrier, l'approche suivante permet de modéliser le **processus** décrivant la progression d'une entité dans le système en soulignant l'interaction entre les processus.

```
def Planifier(ref, type,calendrier, F):
    calendrier.append([ref,type,F])
```

Figure 9 : Fonction de planification

```
def selectionner(calendrier):
    mindate=1000000
    index=0
    for i in range(len(calendrier)):
        if(mindate > calendrier[i][2]):
            mindate = calendrier[i][2]
            index = i
    client=[calendrier[index][0],calendrier[index][1],calendrier[index][2]]
    calendrier.remove(calendrier[index])
    return(client)
```

Figure 10 : Fonction sélectionner

```
def findclient(clients,client):
    n = len(clients)
    for i in range(n):
        if (clients[i][0]==client[0]):
            clients[i][2]=client[2]-clients[i][2]  #(Ts) ou (Ta)
    return clients
```

Figure 11 : Fonction de temps de file d'attente

La figure ci-dessus est dédier au calcul du temps de séjours ou d'attente dans une fils d'attente pour chaque client.

```
def avg(clients):
    n = len(clients)
    j=0
    for i in range(n):
        j=j+clients[i][2] #H de FM
    return j/n
```

Figure 12 : Fonction de moyenne d'arrivé des clients

```
def avgAttente1(clients, TFS):
    j=0
    n=len(clients)
    for i in range(n-1):
        j=j+clients[i][2] #H de FM
    j=j/TFS
    return j
```

Figure 13 : fonction de moyenne d'attente

La fonction ci-dessus est faite pour le calcul du taux d'attente par rapport au temps de simulation.

```
def arrive(ref, NCE, NCP,i,LQ,H,calendrier,IX,IY,IZ,clients):
    if (LQ[0]<=1):
        NCE[0] = NCE[0]  + 1
        Planifier(ref,'FM',calendrier,H+selectionnerF2(alea(IX,IY,IZ)))#planifier finmagasinage
        clients.append([i[0],'A',H])
    else: #au plus un client ds la file
        NCP[0]=NCP[0]+1
    i[0]+=1
    DA = H+selectionnerF1(alea(IX,IY,IZ))
    if(DA<=720):
        Planifier(i[0],'A',calendrier,DA)
```

Figure 14 : fonction d'arrivée du client

```
def findLQ(QfileIntervall,interval):
    for j in range(len(interval)):
        for k in range(len(QfileIntervall)):
            if QfileIntervall[k][1][0]<= interval[j][0] and QfileIntervall[k][1][1]>interval[j][0]:
                interval[j][1] = interval[j][1] + 1
```

Figure 15 : fonction findLQ

La fonction findLQ est pour le calcul des rations d'utilisation, le ratio de temps par rapport au temps total simulé, pendant lequel on a au plus un client dans la file d'attente, et celui pendant lequel on a plus d'un client dans la file.

Les intervalles décrites dans la fonction son sous la forme d'une matrice (liste de liste), la premier ligne contient instant t_0 en suite combien de client dans la fils d'attente ect.

[QfileIntervall[k][1][0] ; QfileIntervall[k][1][1[=> [interval[j][0],+=1]

```
def findFileClient(ClientsQinf,client):
    test = 0
    for j in range(len(ClientsQinf)):
        if ClientsQinf[j][0] == client[0]:
            ClientsQinf[j][1].append(client[2])
```

Figure 16 : fonction findFileClient

La fonction findFileClient permet de calculer quand un client arrive et quitte la file d'attente.

3.2.1. Les fonctions du scénario 1 : 2 caisses :

```
def finmagasinage(ref, File,C1, C2, LQ,H,calendrier,ClientsQ,TC1,TC2,IX,IY,IZ,QfileIntervall):
    if(C1[0]==0 or C2[0]==0):
        alea2 = selectionnerF3(alea(IX,IY,IZ))
        if(C1[0]==0):
            C1[0]=ref
            TC1[0] = TC1[0] + alea2
        else:
            C2[0]=ref
            TC2[0] = TC2[0] + alea2 #delta(T)= H + selectionnerF3(alea) - H
        Planifier(ref,'FP',calendrier,H+alea2)
    else:
            LQ[0]+=1
            File.append(ref)
            QfileIntervall.append([ref,[H]])
            ClientsQ.append([ref,'FM',H])
```

Figure 17 : fonction fin magasinage 2 caisses

```
def finpaiment(ref, LQ, File, C1, C2,H,calendrier,TC1,TC2,ClientsQ,IX,IY,IZ,QfileIntervall):
    client2 = [ref,'FP',H]
    if(LQ[0]==0):
        if(C1[0]==ref):
            C1[0]=0
        else:
            C2[0]=0
    else:
        alea2 = selectionnerF3(alea(IX,IY,IZ))
        J=File[0]
        File.remove(J)

        client=[J,'FM',H]
        findFileClient(QfileIntervall,client)
        ClientsQ = findclient(ClientsQ,client)
        LQ[0]-=1
        if(C1[0]==ref):
            C1[0]=J

            TC1[0] = TC1[0] + alea2
        else:
            C2[0]=J

            TC2[0] = TC2[0] + alea2
        Planifier(J,'FP',calendrier,H+alea2)
```

Figure 18 : fonction fin paiement 2 caisses

Toutes les fonctions présenter auparavant seront utiliser dans notre fonction principale pour le premier scénario :

```
def main(IX1, IY1,IZ1,pt):
    IX=[int(float(IX1))]
    IY=[int(float(IY1))]
    IZ=[int(float(IZ1))]
    x=np.linspace(1,40,num=40)
    y=[]
    z=[]
    KJ = []
    tfs=[]
    tsmoy=[]
    tatmoy=[]
    tauc1moy=[]
    tauc2moy=[]
    ratio11=[]
    ratio22=[]
    MoyJ = [0,0,0,0,0,0,0,0,0]
```

```
    carreNCP = [0]
    for j in range(40): #simulation 40 jours
        H=0
        i=[1]
        LQ=[0]
        NCP=[0]
        NCE=[0]
        C1=[0]
        C2=[0]
        QfileIntervall = []

        File = []
        clients = [] #pour calculer TSmoy
        ClientsQ =[] #pour calculer TATmoy
        TC1=[0]
        TC2=[0]
        IX[0]=IX[0]+j*5
        IY[0]=IY[0]+j*5
        IZ[0]=IZ[0]+j*5

        calendrier = [[1,'A',H+selectionnerF1(alea(IX,IY,IZ))]]
        while (calendrier!=[]):
            client = selectionner(calendrier)
            H = client[2] #maj
            if client[1] == 'A':
                arrive(client[0], NCE, NCP,i,LQ,H,calendrier,IX,IY,IZ,clients)
            elif client[1] == 'FM':
                finmagasinage(client[0], File,C1, C2, LQ,H,calendrier,ClientsQ
,TC1,TC2,IX,IY,IZ,QfileIntervall)
            else:
                clients = findclient(clients,client)
                finpaiment(client[0], LQ, File, C1, C2,H,calendrier,TC1,TC2,Cl
ientsQ,IX,IY,IZ,QfileIntervall)
        TFS = H
        interval = []
```

```
        for k in range(TFS):
            interval.append([k,0])
        findLQ(QfileIntervall,interval)
        ratio1 = 0
        ratio2 = 0
        for k in range(TFS):
            if interval[k][1]>1:
                ratio2 = ratio2 + 1
            else:
                ratio1 = ratio1 + 1
        ratio1 /= TFS
        ratio2 /= TFS
        TSmoy=avg(clients)#tempsSejour
        TATmoy=avgAttente1(ClientsQ,NCE[0])
        TauC1 = (TC1[0]/TFS)*100
        TauC2 = (TC2[0]/TFS)*100
        y.append(NCE[0])
        z.append(NCP[0])
        tfs.append(TFS)
        tsmoy.append(TSmoy)
        tatmoy.append(TATmoy)
        tauc1moy.append(TauC1)
        tauc2moy.append(TauC2)
        ratio11.append(ratio1)
        ratio22.append(ratio2)
        carreNCP[0] = carreNCP[0] + np.square(NCP[0])
        MoyJ=[MoyJ[0]+NCE[0]/40,MoyJ[1]+NCP[0]/40,MoyJ[2]+TFS/40,MoyJ[3]+TSmoy
/40,MoyJ[4]+TATmoy/40,MoyJ[5]+TauC1/40,MoyJ[6]+TauC2/40,MoyJ[7]+ratio1/40,MoyJ
[8]+ratio2/40]
        KJ.append([j+1,NCE[0],NCP[0],TFS,TSmoy,TATmoy,TauC1,TauC2,ratio1,ratio
2])
    NCPbar = MoyJ[1]
    print("la moyenne des NCP :","%.4f" % NCPbar) ##stocker
    S = np.sqrt(1/39*(carreNCP[0]-40*np.square(NCPbar)))
    IC=[NCPbar - 1.96*(S/np.sqrt(40)),NCPbar + 1.96*(S/np.sqrt(40))]
```

```
    print("lintervalle de confiance :",'[',"%.4f" % IC[0],"%.4f" % IC[1],']')
    NCPm=[]
    NCEm=[]
    TFSm=[]
    TSMOy=[]
    TATMOy=[]
    TAUc1=[]
    TAUc2=[]
    RATIo11=[]
    RATIo22=[]
    for j in range(40):
        NCPm.append(MoyJ[1])
        NCEm.append(MoyJ[0])
        TFSm.append(MoyJ[2])
        TSMOy.append(MoyJ[3])
        TATMOy.append(MoyJ[4])
        TAUc1.append(MoyJ[5])
        TAUc2.append(MoyJ[6])
        RATIo11.append(MoyJ[7])
        RATIo22.append(MoyJ[8])
    KJ.append(MoyJ)
    KJ[40].insert(0,'La moyenne')

    fig, axs = plt.subplots(3,3)
#plot NCE
    axs[0][0].plot(x,y)
    axs[0][0].set_title('Evolution du NCE par jour')
    axs[0][0].plot(x,NCEm,'g')
    axs[0][0].set(xlabel='jours', ylabel='NCE')
#plot NCP
    axs[0][1].plot(x,z, 'tab:orange')
    axs[0][1].plot(x,NCPm,'g')
    axs[0][1].set_title('Evolution du NCP par jour')
    axs[0][1].set(xlabel='jours', ylabel='NCP')
```

```
#plot TFS
    axs[0][2].plot(x,tfs, 'tab:purple')
    axs[0][2].plot(x,TFSm,'g')
    axs[0][2].set_title('Evolution du TFS par jour')
    axs[0][2].set(xlabel='jours', ylabel='TFS')
#plot TSMOY
    axs[1][0].plot(x,tsmoy, 'tab:red')
    axs[1][0].plot(x,TSMOy,'g')
    axs[1][0].set_title('Evolution du TSMOY par jour')
    axs[1][0].set(xlabel='jours', ylabel='TSMOY')
#plot TATMOY
    axs[1][1].plot(x,tatmoy, 'tab:pink')
    axs[1][1].plot(x,TATMOy,'g')
    axs[1][1].set_title('Evolution du TATMOY par jour')
    axs[1][1].set(xlabel='jours', ylabel='TATMOY')
#plot TAUC1
    axs[1][2].plot(x,tauc1moy, 'tab:brown')
    axs[1][2].plot(x,TAUc1,'g')
    axs[1][2].set_title('Evolution du TAUC1 par jour')
    axs[1][2].set(xlabel='jours', ylabel='TAUC1')
#plot TAUC2
    axs[2][0].plot(x,tauc2moy, 'tab:cyan')
    axs[2][0].plot(x,TAUc2,'g')
    axs[2][0].set_title('Evolution du TAUC2 par jour')
    axs[2][0].set(xlabel='jours', ylabel='TAUC2')
#plot RATIO1
    axs[2][1].plot(x,ratio11, 'tab:olive')
    axs[2][1].plot(x,RATIo11,'g')
    axs[2][1].set_title('Evolution du RATIO1 par jour')
    axs[2][1].set(xlabel='jours', ylabel='RATIO1')
#plot RATIO2
    axs[2][2].plot(x,ratio22, 'tab:gray')
    axs[2][2].plot(x,RATIo22,'g')
    axs[2][2].set_title('Evolution du RATIO2 par jour')
```

```
    if pt==1:
        plt.show()
    return KJ
```

Figure 19 : La fonction Main le cas de 2 caisses

3.2.2. Les fonctions du scénario 2 : 3 caisses :

```
def finmagasinage(ref, File,C1, C2, LQ,H,calendrier,ClientsQ,TC1,TC2,IX,IY,IZ,QfileIntervall,TC3,C3):
    if(C1[0]==0 or C2[0]==0 or  C3[0]==0):
        alea2 = selectionnerF3(alea(IX,IY,IZ))
        if(C1[0]==0):
            C1[0]=ref
            TC1[0] = TC1[0] + alea2
        elif C2[0]==0 :
            C2[0]=ref
            TC2[0] = TC2[0] + alea2 #delta(T)= H + selectionnerF3(alea) - H
        else:
            C3[0]=ref
            TC3[0] = TC3[0] + alea2
        Planifier(ref,'FP',calendrier,H+alea2)
    else:
            LQ[0]+=1
            File.append(ref)
            QfileIntervall.append([ref,[H]])
            ClientsQ.append([ref,'FM',H])
```

Figure 20: fonction fin magasinage 3 caisses

```
def finpaiment(ref, LQ, File, C1, C2,H,calendrier,TC1,TC2,ClientsQ,IX,IY,IZ,QfileIntervall,TC3,C3):
    client2 = [ref,'FP',H]
    if(LQ[0]==0):
        if(C1[0]==ref):
            C1[0]=0

        elif C2[0] == ref:
            C2[0]=0

        else:
            C3[0]=0

    else:
        alea2 = selectionnerF3(alea(IX,IY,IZ))
        J=File[0]

        File.remove(J)

        client=[J,'FM',H]
        findFileClient(QfileIntervall,client)
        ClientsQ = findclient(ClientsQ,client)
        LQ[0]-=1
        if(C1[0]==ref):
            C1[0]=J
            TC1[0] = TC1[0] + alea2

        elif(C2[0]==ref):
            C2[0]=J
            TC2[0] = TC2[0] + alea2

        else:
            C3[0]=J
            TC3[0] = TC3[0] + alea2

        Planifier(J,'FP',calendrier,H+alea2)
```

Figure 21 : fonction fin paiement 3 caisses

La fonction Main du deuxième scénario est programmée comme suit :

```
def main(IX1,IY1,IZ1,pt):
    IX=[int(IX1)]
    IY=[int(IY1)]
    IZ=[int(IZ1)]
    x=np.linspace(1,40,num=40)
    y=[]
    z=[]
    K=[]
    tfs=[]
    tsmoy=[]
    tatmoy=[]
    tauc1moy=[]
    tauc2moy=[]
```

```
    ratio11=[]
    ratio22=[]
    #[NCE[0]/40,NCP[0]/40,TFS/40,TSmoy/40,TATmoy/40,TauC1/40,TauC2/40,TAT_un_c
lient/40]
    MoyJ = [0,0,0,0,0,0,0,0,0,0]
    carreNCP=[0]
    for j in range(40): #simulation 40 jours
        H=0
        i=[1]
        LQ=[0]
        NCP=[0]
        NCE=[0]
        C1=[0]
        C2=[0]
        C3=[0]
        QfileIntervall = []

        File = []
        clients = [] #pour calculer TSmoy
        ClientsQ =[] #pour calculer TATmoy
        TC1=[0]
        TC2=[0]
        TC3=[0]
        IX[0]=IX[0]+j*5
        IY[0]=IY[0]+j*5
        IZ[0]=IZ[0]+j*5
        calendrier = [[1,'A',H+selectionnerF1(alea(IX,IY,IZ))]]

        while (calendrier!=[]):
            client = selectionner(calendrier)
            H = client[2] #maj
            if client[1] == 'A':
                arrive(client[0], NCE, NCP,i,LQ,H,calendrier,IX,IY,IZ,clients)
            elif client[1] == 'FM':
```

```
                finmagasinage(client[0], File,C1, C2, LQ,H,calendrier,ClientsQ
,TC1,TC2,IX,IY,IZ,QfileIntervall,TC3,C3)
            else:
                clients = findclient(clients,client)
                finpaiment(client[0], LQ, File, C1, C2,H,calendrier,TC1,TC2,Cl
ientsQ,IX,IY,IZ,QfileIntervall,TC3,C3)
        TFS = H
        interval = []
        for k in range(TFS):
            interval.append([k,0])
        findLQ(QfileIntervall,interval)
        ratio1 = 0
        ratio2 = 0
        for k in range(TFS):
            if interval[k][1]>1:
                ratio2 = ratio2 + 1
            else:
                ratio1 = ratio1 + 1
        ratio1 /= TFS
        ratio2 /= TFS
        TSmoy=avg(clients)#tempsSejour
        TATmoy=avgAttente1(ClientsQ,NCE[0])
        TauC1 = (TC1[0]/TFS)*100
        TauC2 = (TC2[0]/TFS)*100
        TauC3 = (TC3[0]/TFS)*100
        y.append(NCE[0])
        z.append(NCP[0])
        tfs.append(TFS)
        tsmoy.append(TSmoy)
        tatmoy.append(TATmoy)
        tauc1moy.append(TauC1)
        tauc2moy.append(TauC2)
        ratio11.append(ratio1)
        ratio22.append(ratio2)
        carreNCP[0] = carreNCP[0] + np.square(NCP[0])
```

```
        MoyJ=[MoyJ[0]+NCE[0]/40,MoyJ[1]+NCP[0]/40,MoyJ[2]+TFS/40,MoyJ[3]+TSmoy
/40,MoyJ[4]+TATmoy/40,MoyJ[5]+TauC1/40,MoyJ[6]+TauC2/40,MoyJ[7]+TauC3/40,MoyJ[
8]+ratio1/40,MoyJ[9]+ratio2/40]
        K.append([j+1,NCE[0],NCP[0],TFS,TSmoy,TATmoy,TauC1,TauC2,TauC3,ratio1,
ratio2])
    NCPbar = MoyJ[1]
    print("\nla moyenne des NCP :","%.4f" % NCPbar)
    S = np.sqrt(1/39*(carreNCP[0]-40*np.square(NCPbar)))
    IC=[NCPbar - 1.96*(S/np.sqrt(40)),NCPbar + 1.96*(S/np.sqrt(40))]
    print("lintervalle de confiance :",'[',"%.4f" % IC[0],"%.4f" % IC[1],']')
    NCPm=[]
    NCEm=[]
    for j in range(40):
        NCPm.append(MoyJ[1])
        NCEm.append(MoyJ[0])
    NCPm=[]
    NCEm=[]
    TFSm=[]
    TSMOy=[]
    TATMOy=[]
    TAUc1=[]
    TAUc2=[]
    RATIo11=[]
    RATIo22=[]
    for j in range(40):
        NCPm.append(MoyJ[1])
        NCEm.append(MoyJ[0])
        TFSm.append(MoyJ[2])
        TSMOy.append(MoyJ[3])
        TATMOy.append(MoyJ[4])
        TAUc1.append(MoyJ[5])
        TAUc2.append(MoyJ[6])
        RATIo11.append(MoyJ[8])
        RATIo22.append(MoyJ[9])
    K.append(MoyJ)
```

```
    K[40].insert(0,'La moyenne')
    fig, axs = plt.subplots(3,3)
#plot NCE
    axs[0][0].plot(x,y)
    axs[0][0].set_title('Evolution du NCE par jour')
    axs[0][0].plot(x,NCEm,'g')
    axs[0][0].set(xlabel='jours', ylabel='NCE')
#plot NCP
    axs[0][1].plot(x,z, 'tab:orange')
    axs[0][1].plot(x,NCPm,'g')
    axs[0][1].set_title('Evolution du NCP par jour')
    axs[0][1].set(xlabel='jours', ylabel='NCP')
#plot TFS
    axs[0][2].plot(x,tfs, 'tab:purple')
    axs[0][2].plot(x,TFSm,'g')
    axs[0][2].set_title('Evolution du TFS par jour')
    axs[0][2].set(xlabel='jours', ylabel='TFS')
#plot TSMOY
    axs[1][0].plot(x,tsmoy, 'tab:red')
    axs[1][0].plot(x,TSMOy,'g')
    axs[1][0].set_title('Evolution du TSMOY par jour')
    axs[1][0].set(xlabel='jours', ylabel='TSMOY')
#plot TATMOY
    axs[1][1].plot(x,tatmoy, 'tab:pink')
    axs[1][1].plot(x,TATMOy,'g')
    axs[1][1].set_title('Evolution du TATMOY par jour')
    axs[1][1].set(xlabel='jours', ylabel='TATMOY')
#plot TAUC1
    axs[1][2].plot(x,tauc1moy, 'tab:brown')
    axs[1][2].plot(x,TAUc1,'g')
    axs[1][2].set_title('Evolution du TAUC1 par jour')
    axs[1][2].set(xlabel='jours', ylabel='TAUC1')
#plot TAUC2
    axs[2][0].plot(x,tauc2moy, 'tab:cyan')
```

```
axs[2][0].plot(x,TAUc2,'g')
    axs[2][0].set_title('Evolution du TAUC2 par jour')
    axs[2][0].set(xlabel='jours', ylabel='TAUC2')
#plot RATIO1
    axs[2][1].plot(x,ratio11, 'tab:olive')
    axs[2][1].plot(x,RATIo11,'g')
    axs[2][1].set_title('Evolution du RATIO1 par jour')
    axs[2][1].set(xlabel='jours', ylabel='RATIO1')
#plot RATIO2
    axs[2][2].plot(x,ratio22, 'tab:gray')
    axs[2][2].plot(x,RATIo22,'g')
    axs[2][2].set_title('Evolution du RATIO2 par jour')
    axs[2][2].set(xlabel='jours', ylabel='RATIO2')
    if pt == 1:
        plt.show()
    return K
```

Figure 22 : La fonction Main dans le cas de 3 caisses

3.3. Validation du programme :

Une fois le modèle réalisé, il faut vérifier si les règles logiques qui décrivent le flux sont bien programmées et correspondent à ce qui est demandé. Nous exécutons donc des tests de simulation uniquement pour vérifier le bon fonctionnement du modèle. Si des différences apparaissent, il faut pouvoir "tracer" tous les événements du modèle et vérifier leur cohérence par rapport aux données.

```
-> def main(IX1, IY1,IZ1,pt):
(Pdb) step
--Return--
> <string>(1)<module>()->None
(Pdb) run
Restarting main.py with arguments:
        main.py
> /home/main.py(1)<module>()
-> def main(IX1, IY1,IZ1,pt):
(Pdb) continue
The program finished and will be restarted
> /home/main.py(1)<module>()
-> def main(IX1, IY1,IZ1,pt):
(Pdb) next
--Return--
> /home/main.py(1)<module>()->None
-> def main(IX1, IY1,IZ1,pt):
(Pdb) step
--Return--
> <string>(1)<module>()->None
(Pdb)
```

Figure 23 : Bien debugger

4. Expérimentations et validation du modèle :

Dans cette partie on traitera la validation du modèle et l'expérimentation.

L'exploitation de la simulation est l'étape où l'on utilise le modèle comme support expérimental pour évaluer le comportement dynamique du système.

Pour voir les résultats, les chiffres du tableau, nous exécutons d'abord notre programme, une interface graphique s'affiche d'abord, contenant les modalités de notre programme.

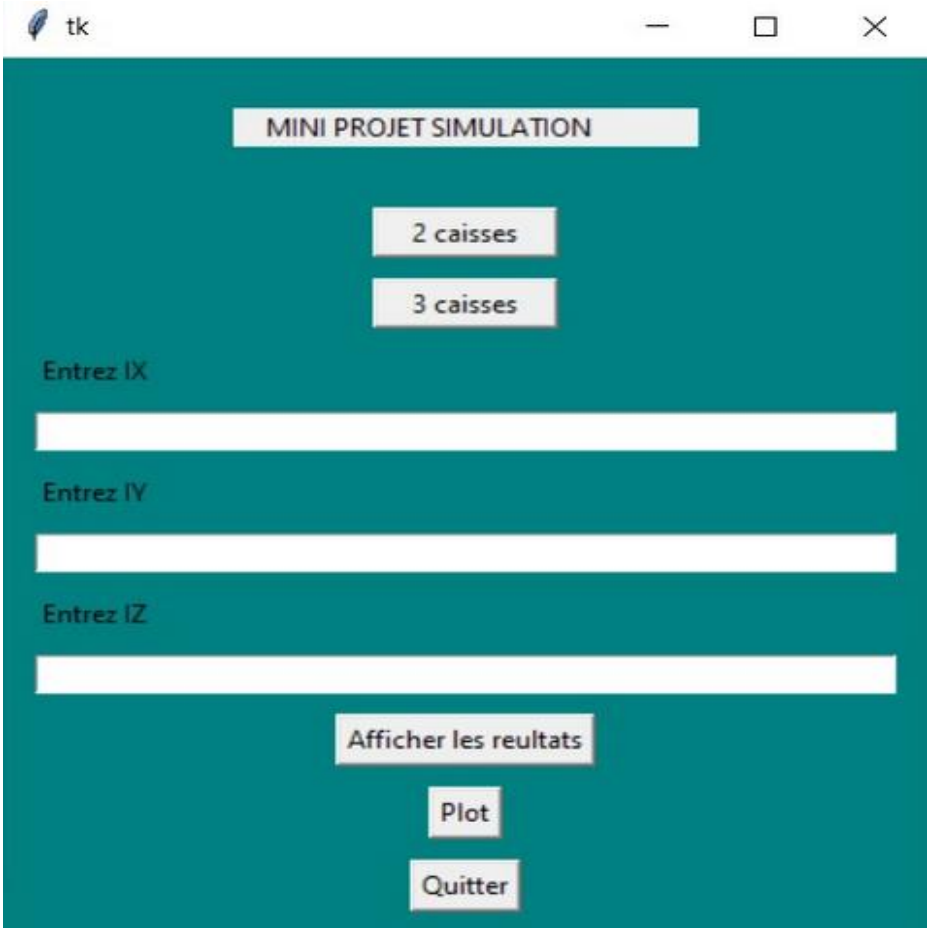

Figure 24 : L'interface graphique

L'interface graphique est réalisable à l'aide de TKinter, une bibliothèque graphique libre d'origine pour le langage Python, permettant la création d'interfaces graphiques.

Prenant tout au long de cette démonstration les valeur 10,100 et 1000 (respectivement) pour IX, IY et IZ.

Commençant par les premiers jours des deux scenarios, le résultat sera comme suit :

jours	NCE	NCP	TFS	TSmoy	TATmoy	TauC1	TauC2	Ratio1	Ratio2
1	342	6	726	8.6374	0.2749	62.259	46.5565	0.9821	0.0179

Figure 25 : simulation premier jour pour 2 caisses

jours	NCE	NCP	TFS	TSmoy	TATmoy	TauC1	TauC2	TauC3	Ratio1	Ratio2
1	329	2	728	8.2097	0.0274	53.2967	34.7527	18.544	0.9973	0.0027

Figure 26 : simulation premier jour pour 3 caisses

Apres le premier jour en entame le cas des 40 jours par le premier scenario, celui des 2 caisses, le résultat est le suivant :

la moyenne des NCP : 9.3250
lintervalle de confiance : [8.2257 10.4243]

jours	NCE	NCP	TFS	TSmoy	TATmoy	TauC1	TauC2	Ratio1	Ratio2
1	342	6	726	8.6374	0.2749	62.259	46.5565	0.9821	0.0179
2	344	12	726	8.6279	0.2965	62.259	48.4848	0.9725	0.0275
3	328	8	726	8.5854	0.2409	57.1625	46.9697	0.9848	0.0152
4	338	13	723	8.6331	0.3846	63.9004	49.7925	0.9668	0.0332
5	347	6	727	8.4323	0.1671	59.9725	48.6933	0.9917	0.0083
6	337	13	727	8.8487	0.3383	61.7607	47.1802	0.9739	0.0261
7	328	4	723	8.7134	0.2317	60.3043	47.0263	0.9862	0.0138
8	319	7	730	8.6708	0.232	60.6849	44.2466	0.9877	0.0123
9	328	13	729	8.5518	0.3049	60.9053	44.856	0.9753	0.0247
10	345	12	728	8.7449	0.371	61.6758	51.0989	0.9657	0.0343
11	342	8	727	8.5819	0.2281	59.5598	48.4182	0.9876	0.0124
12	332	6	729	8.6747	0.253	56.1043	47.7366	0.9822	0.0178
13	327	11	728	8.5688	0.315	59.478	46.4286	0.9794	0.0206
14	337	5	727	8.6736	0.276	61.0729	50.0688	0.9876	0.0124
15	335	12	725	8.594	0.2746	60.2759	47.4483	0.9724	0.0276
16	329	6	725	8.7386	0.2644	59.5862	46.6207	0.9862	0.0138
17	342	9	732	8.7719	0.2836	60.3825	49.1803	0.9863	0.0137
18	323	8	731	8.5635	0.2724	58.9603	43.3653	0.9808	0.0192
19	341	7	730	8.7038	0.2463	60.8219	45.8904	0.9849	0.0151
20	337	16	728	8.8724	0.3264	60.8516	49.0385	0.9684	0.0316
21	335	7	726	8.7045	0.2746	64.876	47.9339	0.9807	0.0193
22	332	8	729	8.488	0.2048	58.9849	48.5597	0.9863	0.0137
23	330	9	732	8.6061	0.2879	61.4754	48.6339	0.9768	0.0232
24	348	9	728	8.6006	0.3017	64.1484	51.2363	0.9821	0.0179
25	320	9	731	8.7937	0.2969	59.9179	42.2709	0.9754	0.0246
26	334	15	726	8.6647	0.2994	63.4986	47.7961	0.9766	0.0234
27	336	8	728	8.7649	0.3214	61.6758	49.4505	0.9725	0.0275
28	334	2	729	8.3952	0.1467	60.9053	46.3649	0.9931	0.0069
29	327	6	731	8.6177	0.2508	60.3283	46.1012	0.985	0.015
30	349	12	728	8.8739	0.3725	64.011	50.2747	0.967	0.033
31	348	18	724	8.727	0.4023	64.0884	48.2044	0.9544	0.0456
32	346	10	729	8.7052	0.3324	63.5117	50.8916	0.9712	0.0288
33	331	13	727	8.6737	0.2689	58.8721	48.0055	0.9739	0.0261
34	337	8	731	8.5223	0.2196	59.6443	47.4692	0.985	0.015
35	332	14	729	8.9307	0.3434	61.5912	48.2853	0.9753	0.0247
36	336	3	730	8.4315	0.1786	59.863	46.3014	0.9945	0.0055
37	337	10	727	8.8843	0.2938	62.1733	48.8308	0.9835	0.0165
38	333	9	728	8.5165	0.2643	59.7527	45.3297	0.9849	0.0151
39	342	12	726	8.7749	0.307	63.7741	49.1736	0.9725	0.0275
40	339	9	729	8.6726	0.2448	63.3745	47.0508	0.9835	0.0165
La moyenne	335.675	9.325	727.875	8.6634	0.2798	61.1111	47.6816	0.9794	0.0206

Figure 27 : résultat du premier scenario

La moyenne NCP est inclus dans notre intervalle de confiance.

Les tracés des attributs générer par notre code sont les suivant :

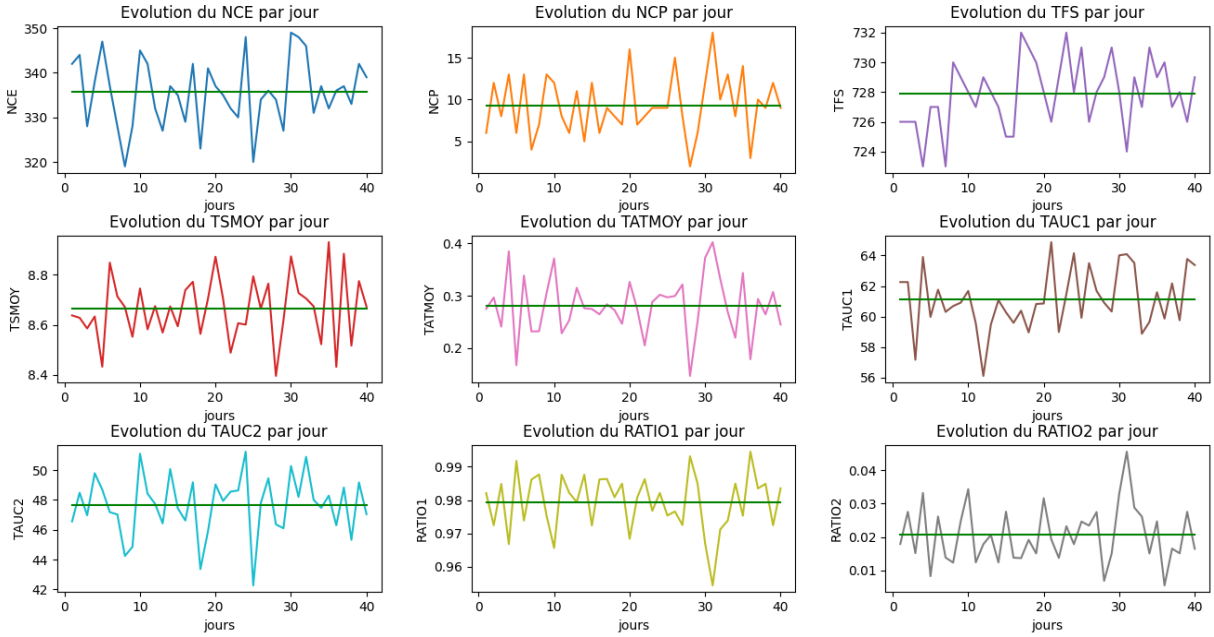

Figure 28 : tracés du premier scenario

Ensuite, nous traitons le cas du deuxième scenario, 3 caisses, ci-dessous on trouve le tableau des résultats d'attributs et leurs tracés.

la moyenne des NCP : 0.9000
lintervalle de confiance : [0.6124 1.1876]

jours	NCE	NCP	TFS	TSmoy	TATmoy	TauC1	TauC2	TauC3	Ratio1	Ratio2
1	329	2	728	8.2097	0.0274	53.2967	34.7527	18.544	0.9973	0.0027
2	357	1	728	8.2997	0.0588	52.7473	37.5	22.8022	0.9945	0.0055
3	350	0	731	8.38	0.04	53.3516	39.9453	20.1094	1.0	0.0
4	341	1	721	8.2287	0.0264	52.8433	39.9445	19.4175	0.9986	0.0014
5	353	2	727	8.4164	0.0368	54.608	37.1389	20.2201	1.0	0.0
6	358	0	729	8.4274	0.0363	54.0466	36.0768	20.439	0.9986	0.0014
7	350	1	729	8.5429	0.0343	53.9095	38.546	20.0274	0.9986	0.0014
8	349	1	732	8.4527	0.0258	52.8689	38.5246	19.9454	0.9959	0.0041
9	341	2	729	8.5543	0.044	50.3429	37.3114	19.6159	0.9945	0.0055
10	351	0	733	8.4074	0.0313	52.3874	39.0177	21.4188	0.9986	0.0014
11	351	1	732	8.2849	0.037	54.6448	41.2568	18.7158	0.9986	0.0014
12	369	2	728	8.3713	0.0434	56.7308	41.6209	24.1758	0.9973	0.0027
13	357	0	731	8.3137	0.0252	53.078	40.4925	20.2462	1.0	0.0
14	355	1	731	8.4141	0.0451	53.3516	36.7989	21.067	0.9973	0.0027
15	369	2	727	8.355	0.0434	55.7084	39.6149	25.8597	0.9972	0.0028
16	353	1	731	8.5439	0.0368	53.8988	41.8605	22.9822	1.0	0.0
17	349	0	729	8.4183	0.0344	54.8697	36.6255	21.6735	1.0	0.0
18	335	0	733	8.5284	0.0358	52.1146	34.1064	17.0532	1.0	0.0
19	338	0	727	8.2751	0.0207	53.9202	38.2393	17.8817	1.0	0.0
20	349	1	730	8.3725	0.0458	51.9178	36.9863	20.9589	1.0	0.0
21	338	1	730	8.432	0.0237	52.8767	38.4932	17.6712	0.9973	0.0027
22	361	2	732	8.4072	0.0332	52.7322	38.9344	23.7705	0.9986	0.0014
23	344	0	724	8.2878	0.0349	52.0718	37.2928	19.8895	1.0	0.0
24	329	0	730	8.2614	0.0061	47.9452	34.3836	18.2192	1.0	0.0
25	362	2	726	8.4669	0.0442	55.5096	38.7052	22.5895	0.9972	0.0028
26	346	1	728	8.2977	0.052	50.5495	38.0495	19.5055	0.9973	0.0027
27	350	0	728	8.2914	0.0429	51.6484	39.1484	23.6264	1.0	0.0
28	348	1	725	8.3793	0.0603	54.6207	37.5172	20.6897	0.9972	0.0028
29	356	1	728	8.4382	0.0309	53.7088	40.7967	19.5055	0.9973	0.0027
30	345	1	727	8.4203	0.0435	53.37	37.9642	20.0825	0.9972	0.0028
31	333	0	725	8.4775	0.018	52.6897	37.7931	16.9655	1.0	0.0
32	333	4	726	8.4595	0.0601	51.5152	40.0826	20.1102	0.9959	0.0041
33	350	1	726	8.16	0.0143	50.0	39.3939	22.5895	1.0	0.0
34	341	0	728	8.4399	0.0293	52.4725	39.5604	19.6429	0.9986	0.0014
35	367	2	726	8.2425	0.0354	55.3719	39.9449	24.6556	0.9972	0.0028
36	356	0	727	8.1966	0.0112	54.0578	38.9271	21.0454	1.0	0.0
37	352	0	726	8.3551	0.0455	52.7548	37.741	21.2121	0.9972	0.0028
38	347	2	730	8.3112	0.0461	55.7534	40.274	18.7671	0.9959	0.0041
39	344	0	726	8.5116	0.0291	53.3058	38.0165	19.9725	1.0	0.0
40	344	0	728	8.1599	0.0291	54.1209	35.0275	18.4066	1.0	0.0
La moyenne	348.75	0.9	728.3	8.3698	0.0355	53.1928	38.3602	20.5518	0.9984	0.0016

Figure 29 : résultat du deuxième scenario

L’ajout d'une 3ème caisse a diminuée le taux, ainsi que notre moyenne des NCP respect le fait d'être inclus à l’intervalle de confiance.

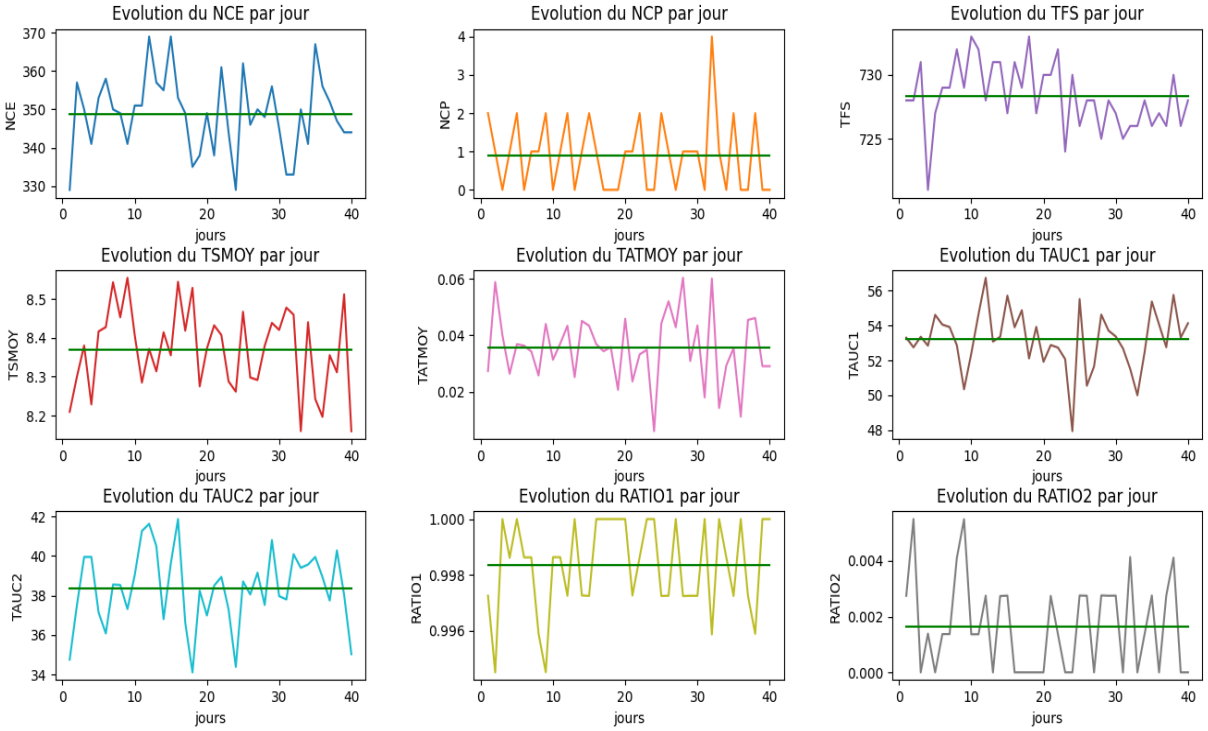

Figure 30 : tracés du deuxième scenario

Cette étape doit se terminer par une validation qui consiste par exemple à comparer les résultats fournis par le modèle aux résultats du système réel si celui-ci existe. Or nos attributs et résultats sont dans les normes donc notre modèle est dans les normes, d'où nous arrêtons la simulation.

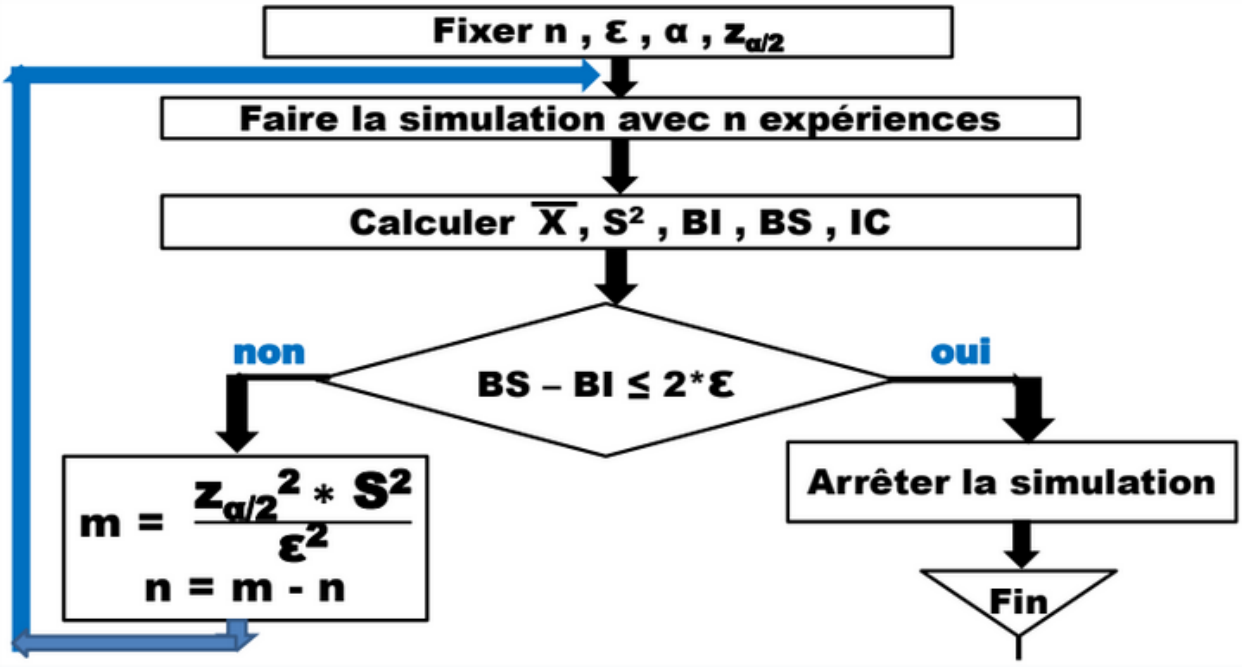

Figure 31 : Critère d'arrêt de la simulation

5. Interprétations :

La prise de décision se fait après comparaison des deux scenarios pour pouvoir déterminer le meilleur entre eux.

Cette dernière étape est importante pour l'étude de simulation. En effet, il va falloir présenter les résultats de l'étude pour qu'ils soient compréhensibles

Pour ce faire nous analysons chaque scénario (phase 1) pour déterminer ses caractéristiques et comme conclusion des deux analyses nous arrivons à choisir celui qui semble convenable et logique (phase 2).

- **Phase 1 :**

Commençant par le premier scenario, celui du 2 caisses, nous analysons :

Scénario 1					
Moyenne NCE	Moyenne NCP	Moyenne TFS	Moyenne Temps de Séjour	Moyenne Temps d'attente	
335,675	9,325	727,875	8,663422788	0,279835819	

Tableau 4 : Moyennes d'indicateurs - scénario 1

Scénario 1	
TauxC1	TauxC2
61%	48%

Tableau 5 : Taux de caisses pour 1ère scénario

Scénario 1	
Ratio1	Ratio2
0,979418368	0,020581632
0,998694761	0,001305239

Tableau 6 : Ratio du 1er scénario

Ci-dessous, nous trouvons une comparaison entre NCE et NCP durant les 40 jours de simulation:

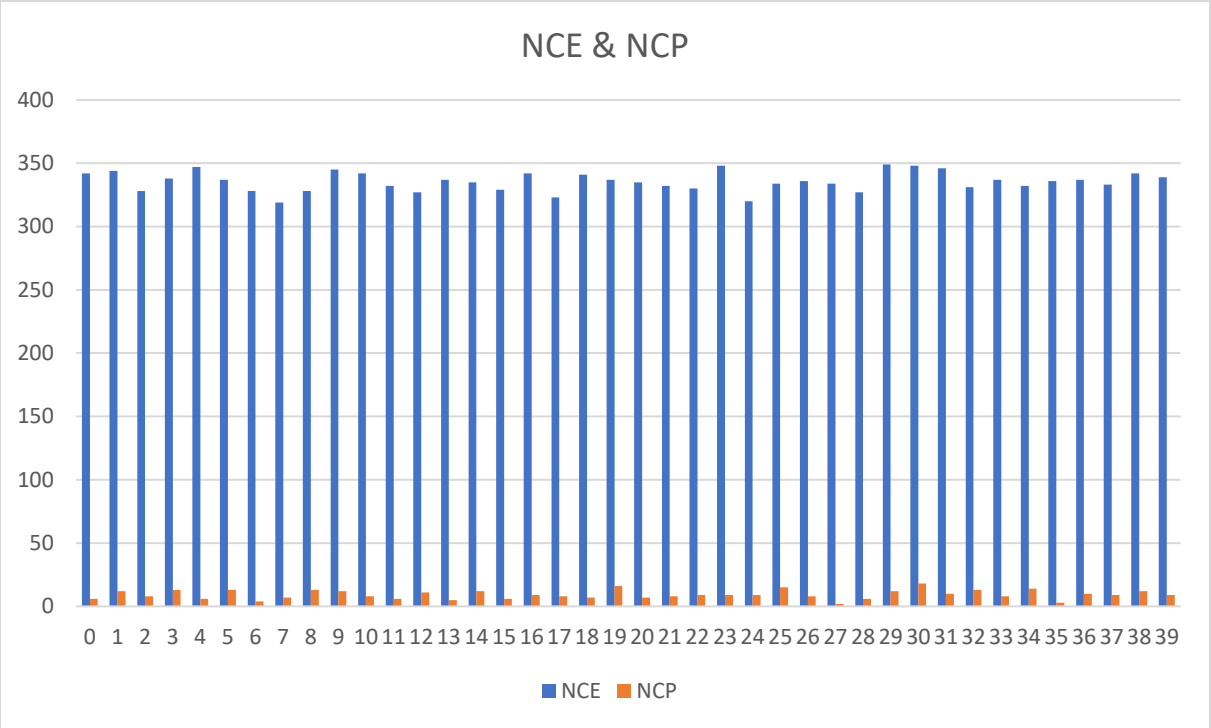

Figure 32 : Parallélogramme de NCE & NCP du 1er scénario

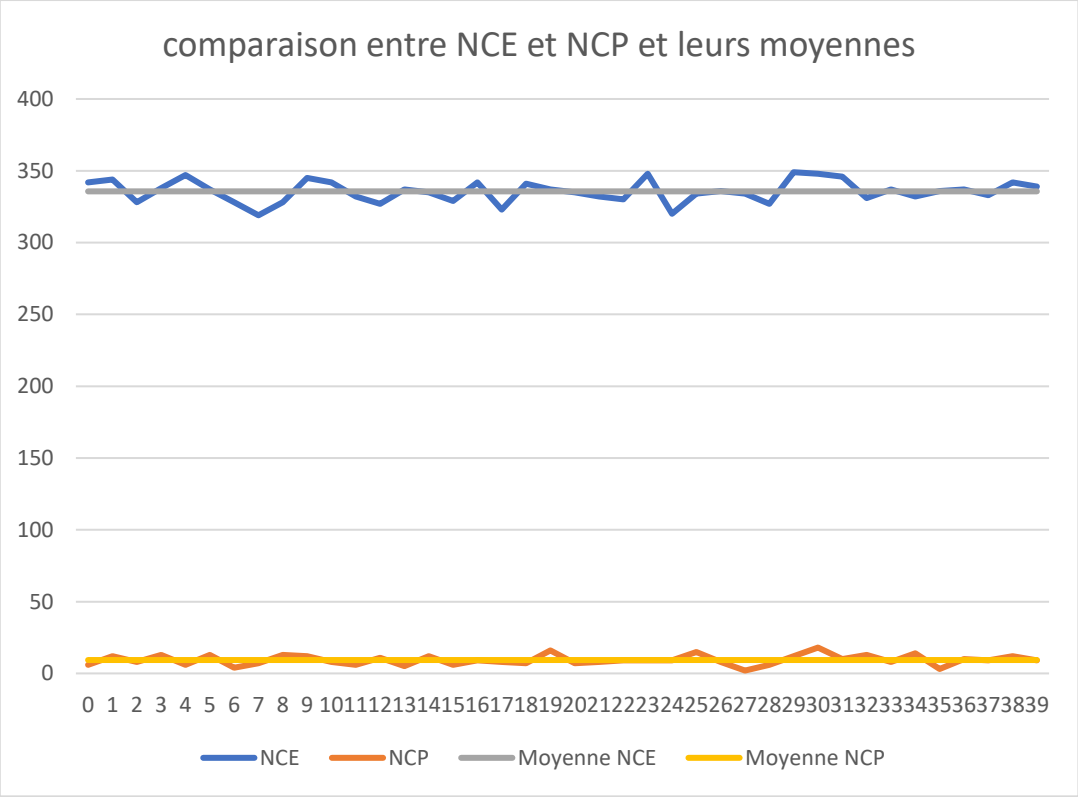

Figure 33 : Courbe de NCE & NCP et leurs moyennes

Nous observons que le nombre des clients perdus est négligeable devant le nombre des clients entrés.

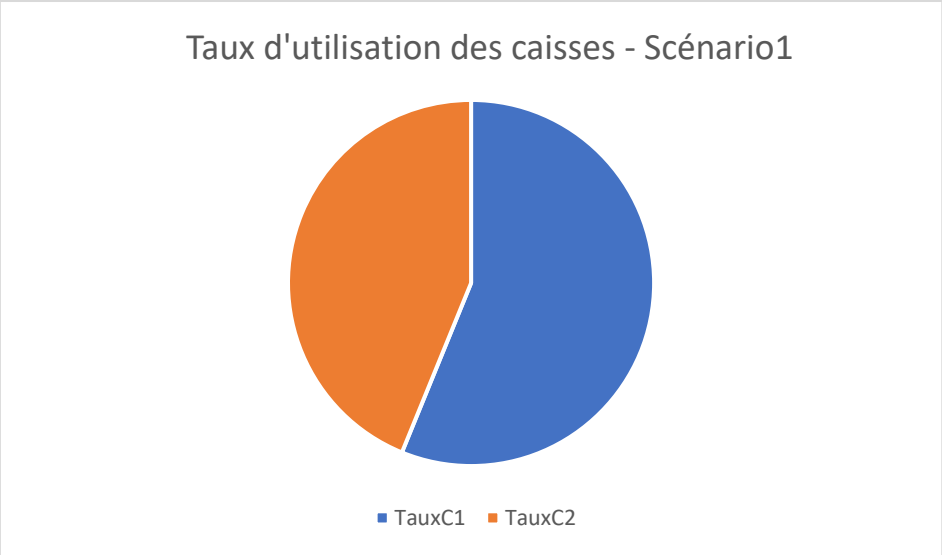

Figure 34 : Taux d'utilisation des caisses pour le 1er scénario

Nous observons que le taux d'utilisation de la caisse 1 est plus important que la caisse 2.

Voir ensuite le cas de 3 caisses :

Scénario 2					
Moyenne NCE	Moyenne NCP	Moyenne TFS	Moyenne Temps de Séjour	Moyenne Temps d'attente	
344,625	0,85	728,2	8,350130082	0,033964919	

Tableau 7 : Moyennes d'indicateurs - scénario 2

Scénario 2		
TauxC1	TauxC2	TauxC3
53%	38%	9%

Tableau 8 : Taux d'utilisation des caisses - scénario 2

Scénario 2	
Ratio1	Ratio2
0,998695	0,001305

Tableau 9 : Ratio - scénario 2

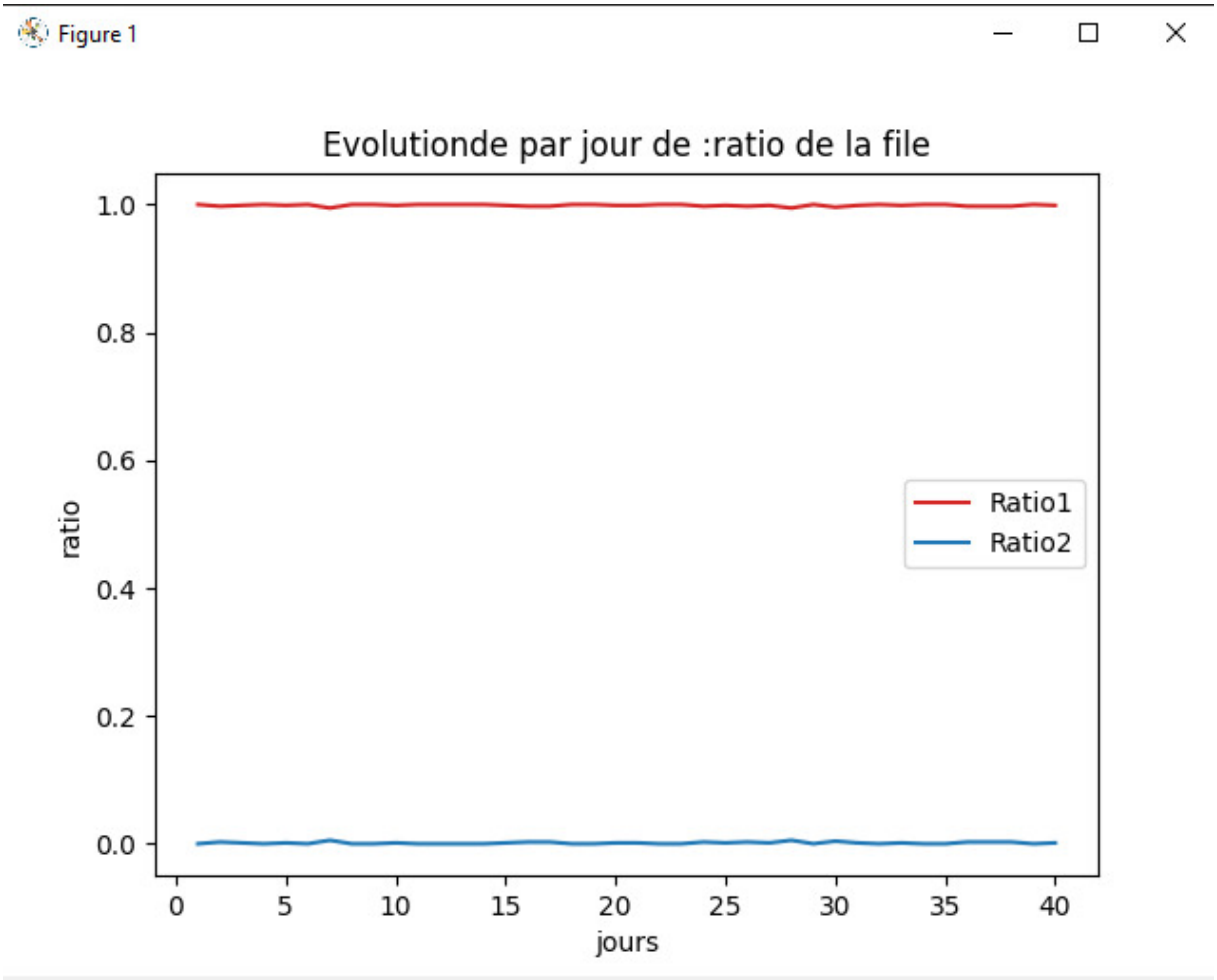

Figure 35 : Evolution des ratios – scénario

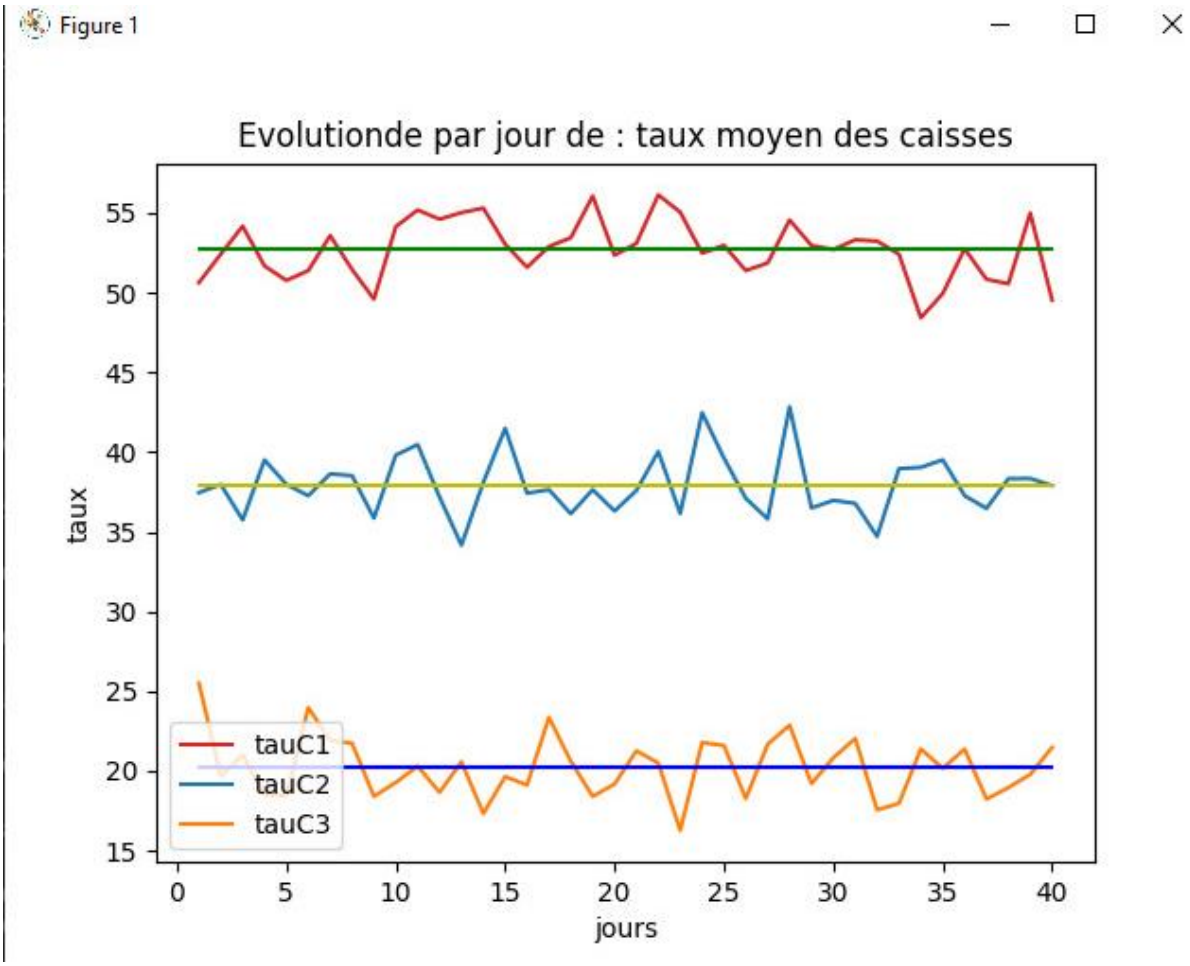

Figure 36 : Evolution de taux moyens – scénario 2

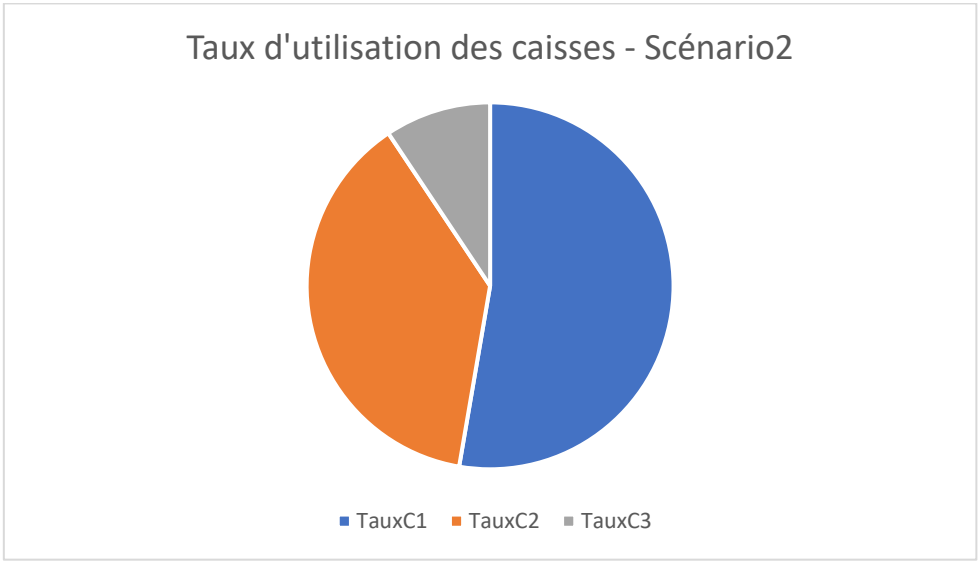

Figure 37 : Taux d'utilisation des caisses - scénario 2

- **Phase 2 :**

Pour comparer les deux scenarios , nous commençons tout d'abord par comparer les taux des caisses.

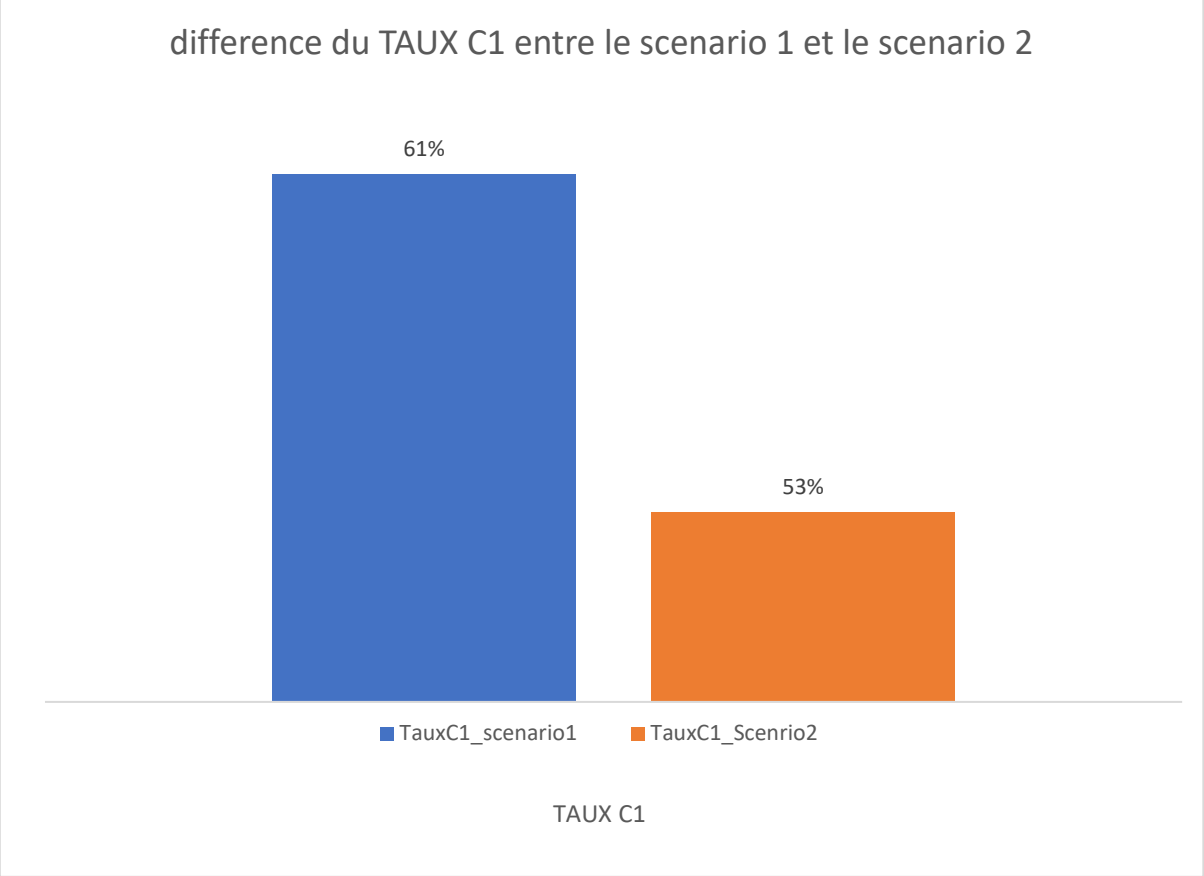

Figure 38 : Taux d'utilisation de la caisse 1

Nous observons que lors de l'ajout d'une 3ème caisse le taux d'utilisation de la caisse 1 diminue de 8%, et c'est presque le même dans le cas de deux caisses donc il n'y a pas une grande différence.

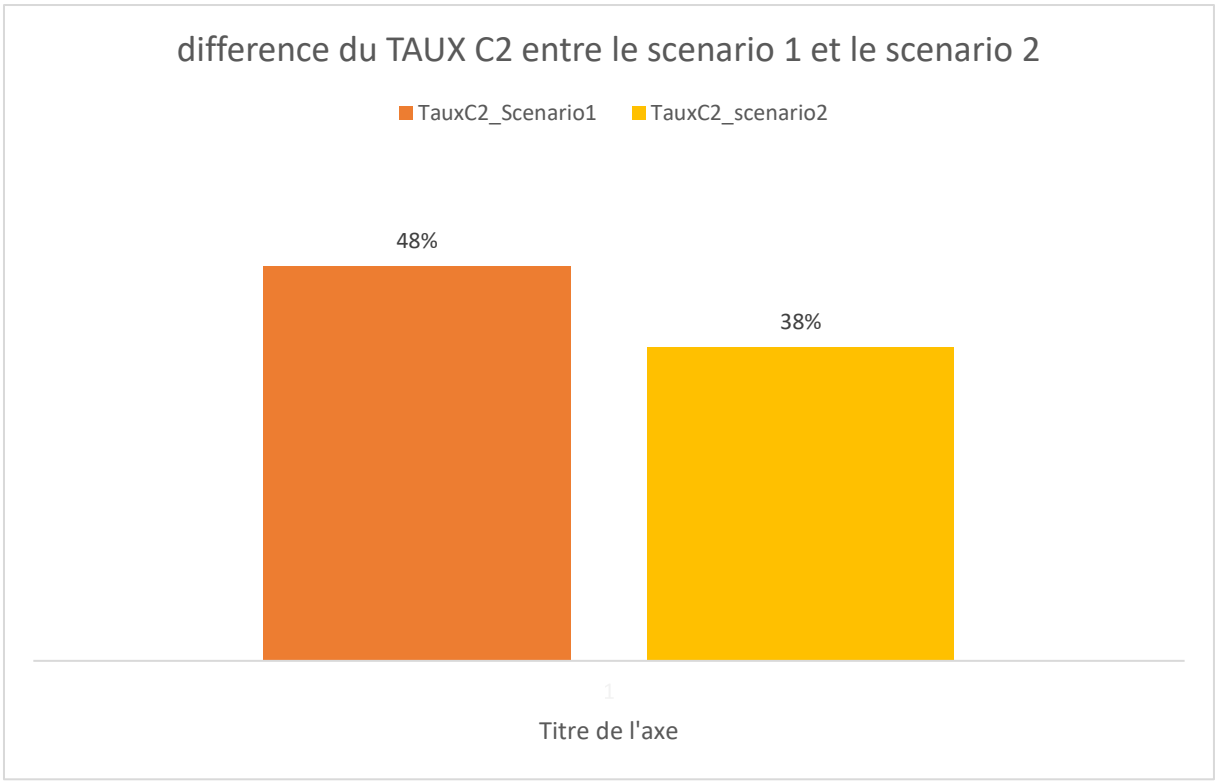

Figure 39 : Taux d'utilisation de la caisse 2

Voir ensuite une comparaison entre les ratios :

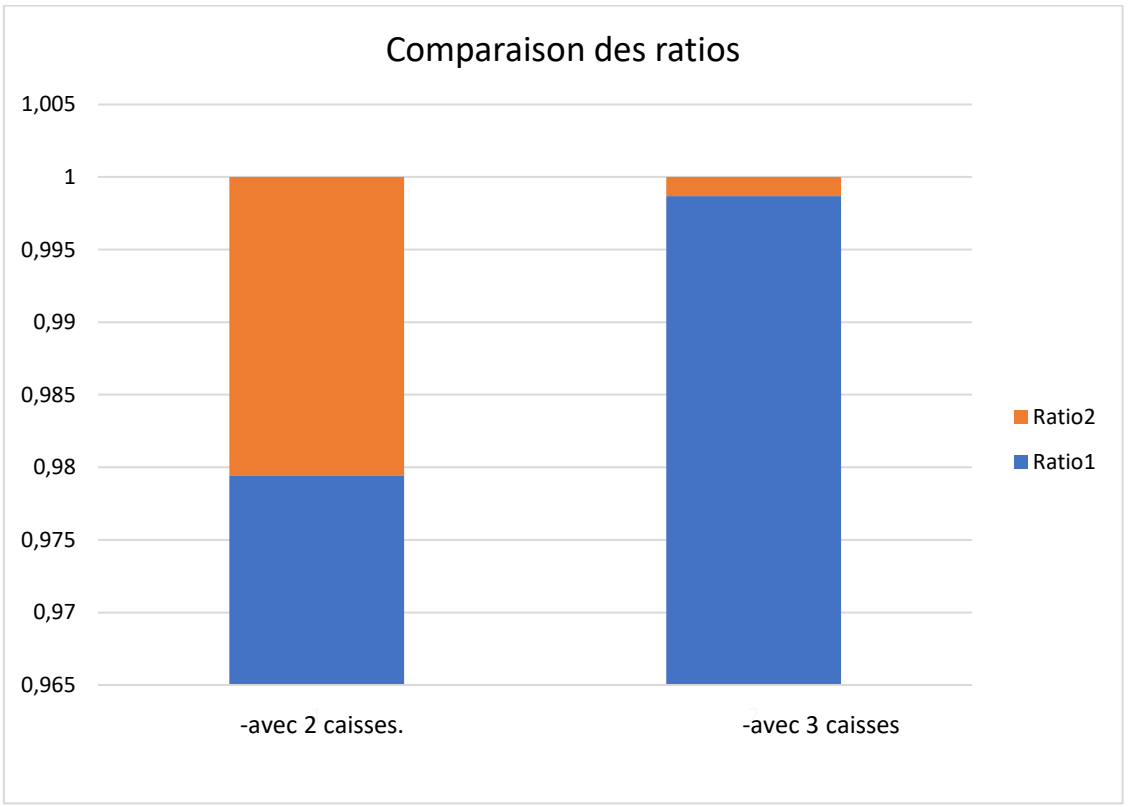

Figure 40 : Comparaison des ratios des deux scénarios

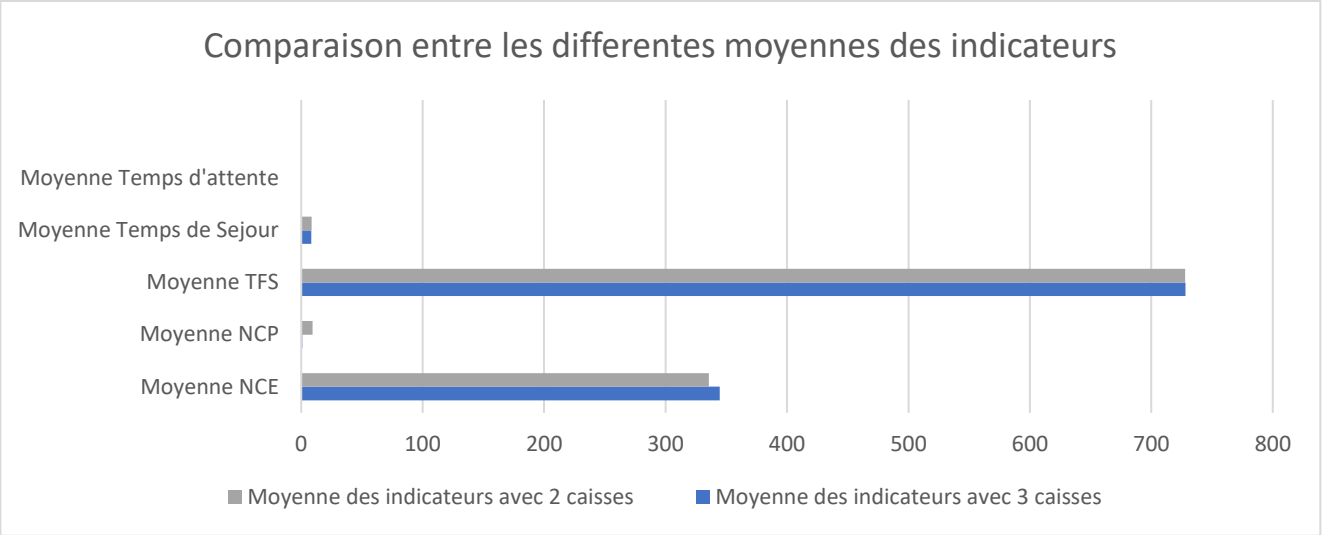

Figure 41 : Comparaison entre les différentes moyennes des indicateurs

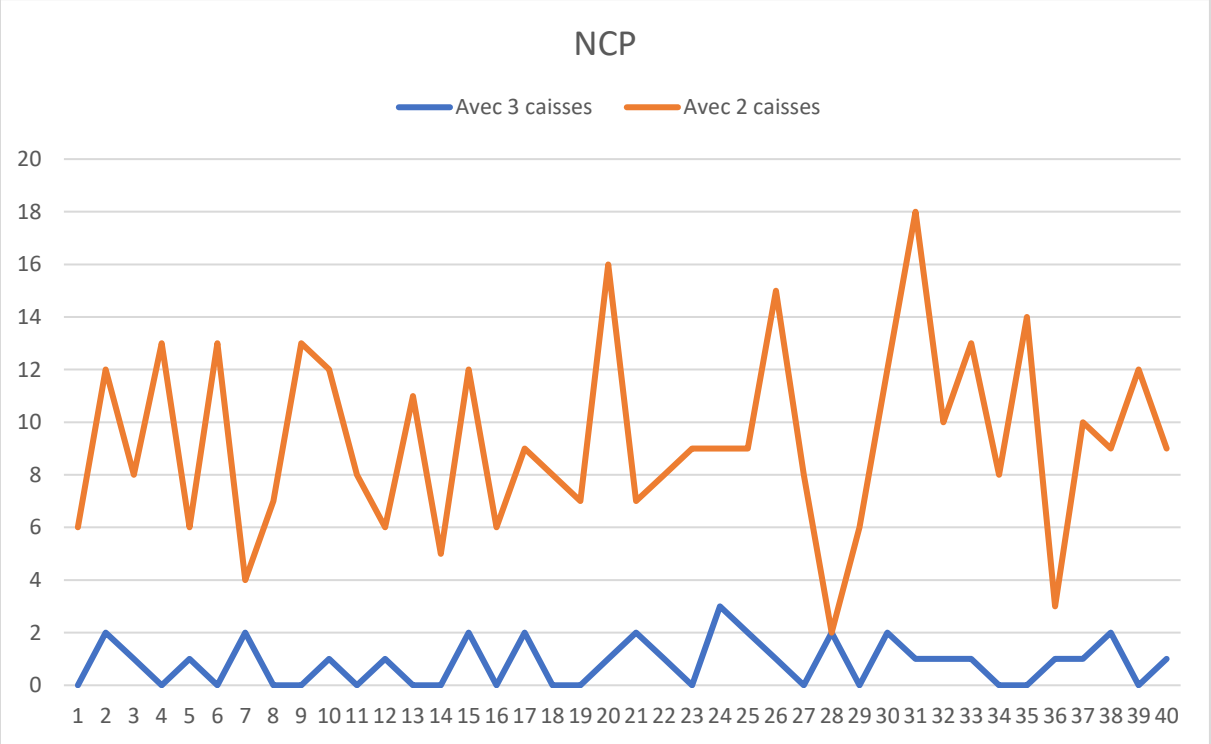

Tableau 10 : Comparaison de NCP des deux scénarios

Lorsque nous disposons de 3 caisses, nous remarquons que nous gagnons plus de clients que lorsque nous disposons seulement de 2 caisses, mais il n'y a pas beaucoup de différence.

L'un des composants essentiels dans cette simulation est le temps d'attentes :

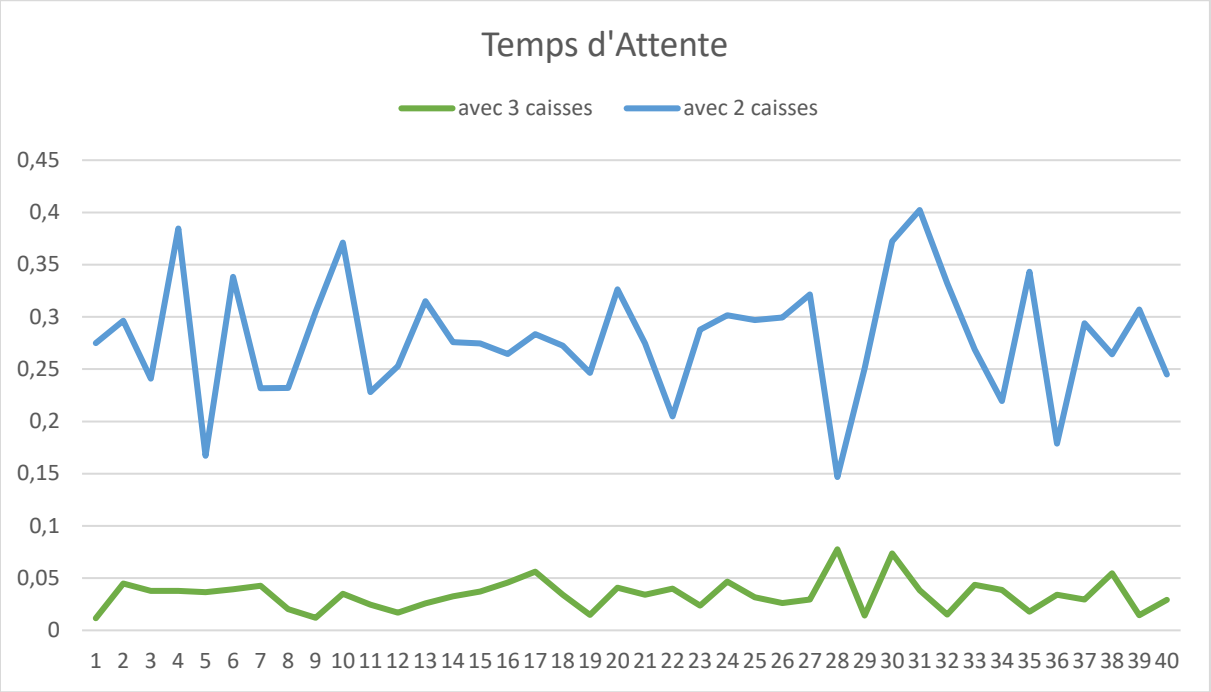

Figure 42 : Comparaison du temps d'attente

Nous remarquons que les clients passent moins du temps dans la file d'attente lorsque nous disposons de 3 caisses, alors nous avons satisfait le besoin du client.

Conclusion

Après avoir programmer un code en langage python qui nous permet de faire une simulation de 40 journées et de conclure les différents indicateurs pour les deux scenarios, nous avons réalisé grâce à Excel les différents graphes pour mieux visualiser les résultats de la simulation. D'où nous avons remarqué que dans notre cas l'ajout de la caisse 3 est une charge plus qu'un bénéfice.

Le travail sur ce mini projet a été pour nous très utile car il ne se contente pas uniquement sur la programmation mais sur plusieurs notions de simulations, ainsi qu'il fait appel à des indicateurs statistiques que nous avons vu dans le cours de simulation de cette année et de statistiques. Ainsi, ça été une occasion pour découvrir d'une part l'interface graphique tkinter de python et d'autre part de maitriser les bases de simulations et les appliquer sur un exemple concret de caisses des magasins. Ajoutons que le travail collectif sur ce mini projet nous a poussé à mieux organiser la méthodologie de travail et à développer l'esprit d'équipe chez chacun de nous.