Institut Supérieur d'Informatique
et de Mathématiques de Monastir
Année Universitaire 2025-2026

Niveau : L2_INFO
Proposé par : Dr. Nada Haj Messaoud

# TP2 : Simulation avec SimPy

## Objectifs du TP

1. Comprendre les concepts de simulation stochastique, dynamique, et à événements discrets avec SimPy.
2. Implémenter le modèle pour collecter les métriques (temps d'attente moyen, temps de réponse moyen, longueur de file, utilisation des infirmières).
3. Tester différents scénarios pour analyser l'impact sur les performances.

| | |
|---|---|
| SimPy | **SimPy**, une bibliothèque Python pour simulations à événements discrets. |
| DataFrame Pandas | Bibliothèque puissante pour la manipulation et l'analyse de données. Fournit des structures de données comme les DataFrames qui facilitent le traitement de données tabulaires. |
| NumPy | Bibliothèque de calcul numérique en Python. Fournit des objets tels que les arrays pour manipuler des matrices et effectuer des opérations mathématiques. |

La classe `simpy.Environment()` est l'élément fondamental de SimPy. Elle permet de :

- Créer un environnement pour coordonner les processus et gérer les ressources.
- Simuler l'écoulement du temps.
- Organiser et synchroniser les événements dans une simulation à événements discrets.

Un processus dans SimPy est simplement une fonction Python qui "cède" le contrôle avec `yield` à l'environnement, en attendant que certaines conditions soient remplies (comme le passage du temps ou la disponibilité d'une ressource).

`simpy.Resource(env, capacity)` : Représente une ressource avec une capacité limitée.

`env.process(generator_function)` : Lance un processus (générateur Python) dans l'environnement. Les processus modélisent les flux (par exemple, arrivées ou activités).

`env.timeout(time)` : Suspend un processus pendant un certain temps (en unités de temps simulées). Simule une attente ou une durée d'activité.

`resource.request()` : Demande l'accès à une ressource. Si la ressource est occupée, l'entité attend dans une file.

`env.now` : donne le temps actuel pour calculer les durées.

`env.run()` est utilisé pour démarrer l'exécution de la simulation. Cela fait avancer le temps simulé et exécute tous les processus jusqu'à ce qu'il n'y ait plus de processus actifs ou qu'on veut décider de limiter le temps d'exécution.

## Problème 1 : Station-service avec plusieurs pompes

**Description** : Une station-service a plusieurs pompes à essence. Les voitures arrivent à des moments aléatoires et doivent attendre si toutes les pompes sont occupées. Chaque voiture prend un certain temps pour se ravitailler. Nous voulons simuler la station-service pour déterminer :

- Le nombre moyen de voitures dans la file d'attente.
- Le temps d'attente moyen d'une voiture avant de se ravitailler.

**Enoncé :**

1. La station-service dispose de **3 pompes**.
2. Les voitures arrivent toutes les **5 à 10 minutes** (aléatoire).
3. Chaque voiture prend **3 à 5 minutes** pour se ravitailler.
4. Simuler la station pendant **12 heures (720 minutes)**.

- Implémenter le programme de simulation en utilisant la bibliothèque SimPy.
- Tracer La courbe indiquant le nombre de voitures arrivées à la station pendant 12 heures.

```
import simpy
import random

def voiture(env, name, station):
    print(f"{name} arrive à la station à {env.now:.2f}")

    with station.request() as request:
        yield request
        print(f"{name} commence à se ravitailler à {env.now:.2f}")
        yield env.timeout(random.randint(3, 5))  # Temps de ravitaillement
        print(f"{name} quitte la station à {env.now:.2f}")

def processus_arrivee_voitures(env, station):
    voiture_id = 0
    while True:
```

```
        yield env.timeout(random.randint(5, 10))  # Arrivée aléatoire des voitures
        voiture_id += 1
        env.process(voiture(env, f"Voiture {voiture_id}", station))

env = simpy.Environment()
station = simpy.Resource(env, capacity=3)
env.process(processus_arrivee_voitures(env, station))
env.run(until=720)
```

## Problème 2 : Simulation d'une clinique avec infirmières

Ce Problème modélise une clinique où des patients arrivent pour une consultation avec une infirmière. Les paramètres de base sont :

- **Taux d'arrivée** : 1 patient toutes les 5 minutes (distribution exponentielle).
- **Durée de consultation** : Moyenne de 6 minutes (distribution exponentielle).
- **Nombre d'infirmières** : 3.
- **Durée de la simulation** : 8 heures (480 minutes).
- **Nombre d'exécutions** : 5.

Nous allons tester trois scénarios :

- **Scénario de base** : Paramètres ci-dessus.
- **Scénario 1** : Taux d'arrivée plus rapide (1 patient toutes les 3 minutes).
- **Scénario 2** : Réduire à 2 infirmières.

**Métriques calculées** :

- Temps d'attente de chaque patient dans la file.
- Temps passé par chaque patient avec une infirmière.
- Temps d'attente moyen dans la file.
- Temps total moyen (attente + consultation).
- Longueur moyenne de la file d'attente.

### Question 1 : Simulation stochastique vs déterministe

**a)** Expliquer la différence entre une simulation stochastique et déterministe. Pourquoi le modèle de la clinique est-il stochastique ?

**b)** Comment modifieriez-vous le modèle pour le rendre déterministe ? Quel impact cela aurait-il sur les résultats ?

**c)** Dans quels cas une simulation déterministe serait-elle préférable pour analyser la clinique ?

**d)** Définir ce qu'est une simulation dynamique et une simulation statique. Le modèle de la clinique est-il dynamique ou statique ? Justifiez.

Dy - evolution dans le temps avec interactions

e) Quelle est la différence entre une simulation à événements discrets et une simulation continue ? Le modèle de la clinique est-il discret ou continu ?

f) Comparer les résultats des trois scénarios (base, arrivée plus rapide, 2 infirmières). Quel scénario offre les meilleures performances pour les patients ?

g) Comment la variabilité stochastique affecte-t-elle les résultats entre les exécutions ? Pourquoi est-il important de faire plusieurs exécutions ?

h) En vous basant sur les résultats, proposer une amélioration pour la clinique réelle.

➔ Implémenter un modèle SimPy pour simuler la clinique et collecter les métriques demandées.

- Définir les Paramètres globaux
- Implémenter la Fonction pour gérer le processus d'un patient
- Implémenter la Fonction pour générer les arrivées de patients
- Implémenter la Fonction pour exécuter une simulation
- Implémenter la Fonction pour exécuter un scénario
- Lancer les scénarios

➔ Tester les trois scénarios et comparer leurs performances.