LOG3430 - Méthodes de test et de validation du logiciel

Laboratoire 5

Modélisation de tests de charge

Département de génie informatique et de génie logiciel
École Polytechnique de Montréal

Automne 2025

1 Introduction

Les tests de performance sont un type de test permettant de déterminer la vitesse d'un ordinateur, d'un réseau ou d'un périphérique. Ils vérifient la performance des composants d'un système en transmettant différents paramètres dans différents scénarios de charge.

Le test de charge est le processus qui simule la charge réelle d'un utilisateur sur une application ou site Web. Il vérifie le comportement de l'application lors de charges normales et élevées. Ce type de test est appliqué lorsqu'un projet de développement touche à sa fin.

Dans ce laboratoire, vous apprendrez à modéliser la charge d'utilisateurs, comment utiliser un outil de test de charge et exécuter des tests de charge.

2 Objectifs

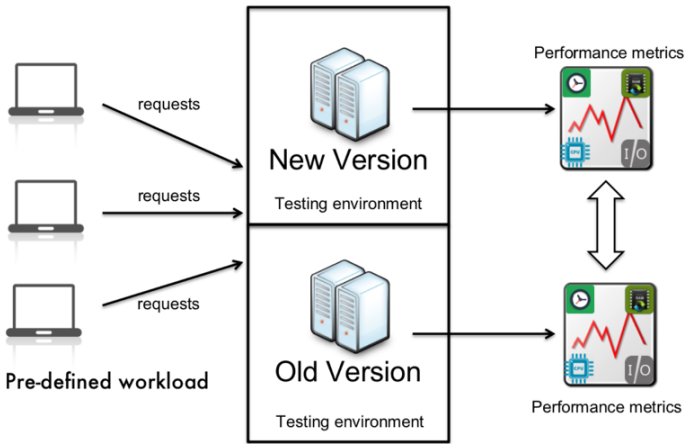
Les objectifs généraux de ce laboratoire sont :

- Modéliser une charge d'utilisateurs
- Utiliser un outil de tests de charges
- Analyser les résultats d'un test de charge

3 Les tests de performance

Les tests de performance/charge consistent en trois processus : (1) concevoir une charge appropriée, (2) exécuter un test de charge et (3) analyser les résultats d'un test de performance/charge.

En pratique, les testeurs configurent l'environnement de test et définissent la charge de travail. Ensuite, les mêmes demandes sont envoyées pour tester le système et les mesures de performance sont collectées. Enfin, les testeurs comparent les deux mesures de performance et identifient la régression des performances (le cas échéant). Ci-dessous, une figure montre le mécanisme des tests de performance/charge :



Les outils

Il existe plusieurs outils pour aider à effectuer des tests de performance/charge.

- **JMeter** - Un outil open source qui peut être utilisé pour les tests de performances et de charge afin d'analyser et de mesurer les performances d'une variété de services.
- **LoadRunner** - Une version de test de performances d'entreprise de Loadrunner et une plate-forme ont permis la normalisation mondiale.
- **ReadLine13** - Une plate-forme de test de charge qui apporte la puissance à faible coût du cloud à JMeter et à d'autres outils de test de charge open source.
- **Locust** - Un outil open source simple et flexible, permettant de créer des tests personnalisés. Sa facilité d'utilisation est renforcée par une interface web interactive et une intégration fluide avec les pipelines CI/CD.

Pour ce TP on va utiliser Locust, pour plus de détails consultez : https://locust.io/

Outils Requis

Système d'exploitation : Windows/MacOS/Linux

Python : version 3.12+

Les tâches

Partie 1 - *Configuration de l'environnement de test*

- Obtenez une copie locale de ce répertoire : https://github.com/Log3430-A2025/TP5.
- Créez un nouvel environnement virtuel et installez les dépendances.
- **Q1.1)** Lancez le serveur web avec `python server/main.py` et visitez `/docs`. Joignez une capture d'écran de cette page.
- Un `locustfile.py` existe déjà. Lancez un test de charge avec celui-ci et accédez à l'interface web de Locust. Consultez la documentation de Locust pour plus d'informations.
- Démarrez un test de charge avec 10 utilisateurs et une montée en charge de 1 utilisateur/seconde.
- **Q1.2)** Joignez une capture d'écran montrant que les requêtes s'effectuent avec succès lorsque le test se rend à 10 utilisateurs.

Partie 2 - *Modélisation de la charge globale*

Un fichier `logs.txt` vous présente des logs du serveur. À partir de ces logs, vous devrez modéliser la charge utilisateur en un test de charge reproductible. Vous aller utiliser deux méthodes : La charge globale et une charge par chaine de Markov. Considérez les données d'utilisation de tous les utilisateurs. Vous pouvez utiliser un IA générative **uniquement** pour vous aider à extraire ces données (par exemple, pour produire un script). Vous pouvez aussi utiliser tout autre méthode de votre choix pour extraire vos données des logs.

— **Q2.1)** Faites l'extraction des informations nécessaires à la conception d'une charge globale à partir des logs. Expliquez votre approche et présentez les résultats obtenus.
— **Q2.2)** Créer un nouveau test de charge dans un nouveau fichier `locustfile_global.py`. Consulter la documentation de Locust pour plus d'informations sur comment définir votre test de charge. Utilisez un délais de 1 à 5 secondes entre les requêtes d'un utilisateur. Joignez une capture d'écran de votre code de test de charge.
— **Q2.3)** Exécutez le test de charge avec un maximum de 10 utilisateurs et une montée de 1 utilisateur par seconde. Joignez une capture d'écran montrant que votre test s'exécute avec succès.

Partie 3 - *Modélisation de la charge par chaine de Markov*

— **Q3.1)** Faites l'extraction des informations nécessaires à la conception d'une charge par chaine de Markov à partir des logs. Expliquez votre approche et présentez les résultats obtenus.
— **Q3.2)** Créer un nouveau test de charge dans un nouveau fichier `locustfile_markov.py`. Vous aurez certainement besoin de consulter la documentation sur les TaskSet pour vous aider. Utilisez un délais de 1 à 5 secondes entre les requêtes d'un utilisateur. Joignez une capture d'écran de votre code de test de charge.
— **Q3.3)** Exécutez le test de charge avec un maximum de 10 utilisateurs et une montée de 1 utilisateur par seconde. Joignez une capture d'écran montrant que votre test s'exécute avec succès.

Partie 4 - *Exécution et analyse des résultats*

— **Q4.1)** Pour chacun des tests de charge que vous avez fait, déterminez le nombre maximal d'utilisateurs supporté par le serveur sur votre machine. Donnez vos résultats et expliquez votre méthode.
— **Q4.2)** Exécutez de nouveau chacun des tests avec comme nombre maximal d'utilisateurs la moitié de ce qui est supporté par votre ordinateur. Assurez vous de laisser le test rouler assez longtemps pour récolter assez de données sur la quantité de requêtes exécutées. Comparez les ratios (fréquences) des requêtes obtenus par chaque méthode. Il se peut que vous ayez besoin de calculer certains ratios vous-mêmes.
— **Q4.3)** À la lumière de vos expériences durant le laboratoire, comparez les deux méthodes de modélisation de charge. Présentez des avantages ou désavantages que vous avez rencontrés. Est-ce que les deux méthodes peuvent être utilisées dans n'importe quel contexte ?

4 Livrables attendus

Les livrables suivants sont attendus :

- Un rapport pour le laboratoire. Le rapport doit contenir :
  - Vos réponses à la question 1.1 : (1 points)
  - Vos réponses à la question 1.2 : (2 points)
  - Vos réponses à la question 2.1 : (3 points)
  - Vos réponses à la question 2.2 : (2 points)
  - Vos réponses à la question 2.3 : (1 points)
  - Vos réponses à la question 3.1 : (3 points)
  - Vos réponses à la question 3.2 : (2 points)
  - Vos réponses à la question 3.3 : (1 points)
  - Vos réponses à la question 4.1 : (2 points)
  - Vos réponses à la question 4.2 : (1 points)
  - Vos réponses à la question 4.3 : (1 points)
  - Qualité du rapport : (1 point).
- Le rapport doit être soumis au format PDF, nommé `matricule1_matricule2_lab5.pdf`.
- L'absence d'une capture d'écran entraînera une non-notation de cette question.

Seulement une personne de l'équipe doit remettre le travail. Déposez le rapport pdf directement. Le rapport doit contenir le titre et numéro du laboratoire, les noms et matricules des coéquipiers ainsi que le numéro du groupe.

5 Information importante

1. Consultez le site Moodle du cours pour la date et l'heure limites de remise des fichiers. **Consultez l'onglet choix d'équipe afin d'avoir accès à la remise, le travail doit se faire en équipe de 2 étudiants.**
2. Un retard de [0,24h] sera pénalisé de 10%, de [24h, 48h] de 20% et de plus de 48h de 50%.
3. Aucun plagiat n'est toléré. Vous devez soumettre uniquement le code et les rapports de couverture de code réalisé par les membres de votre équipe.