

Prototypage d'un moteur d'exécution de workflow orienté données

**Travail de Bachelor**

Non Confidentiel

**Département :** TIC

**Filière :** Informatique et systèmes de communication

**Orientation :** Informatique logicielle

Peronetti Eric

23 février 2024

Travail proposé par :

Prénom Nom de la personne confiant l’étude

Nom de l’entreprise/institution

Adresse, NPA Ville

Supervisé par :

Chapuis Bertil

# Préambule

Ce travail de Bachelor (ci-après TB) est réalisé en fin de cursus d’études, en vue de l’obtention du titre de Bachelor of Science HES-SO en Ingénierie.

En tant que travail académique, son contenu, sans préjuger de sa valeur, n'engage ni la responsabilité de l'auteur, ni celles du jury du travail de Bachelor et de l'Ecole.

Toute utilisation, même partielle, de ce TB doit être faite dans le respect du droit d’auteur.

HEIG-VD

Le Chef du Département

Yverdon-les-Bains, le

## Chapitre 1 Cahier des charges

### Introduction

Ce projet de Bachelor vise à concevoir et développer un moteur d'exécution de workflow orienté données, dédié à l'optimisation des processus d'extraction, de transformation et de chargement (ETL). Ce moteur se concentrera sur l'automatisation des tâches répétitives et complexes associées à la manipulation de données.

Les principales technologies utilisées seront :

* Quarkus pour la partie backend avec GraalVM (JIT) pour faire de l’exécution de code (dans un autre langage de programmation) au runtime.
* React avec React Flow pour ce qui est du frontend et de l’affichage de nœuds de notre workflow.

### Objectifs

Les objectifs sont les suivants :

* Conception d'un modèle de données pour représenter les workflows :
  + Développer un modèle de données robuste pour représenter les workflows sous forme de graphes dirigés acycliques (DAG).
  + Modéliser les nœuds et les arcs pour une représentation précise des dépendances et des opérations au sein du workflow.
  + Implémenter une gestion efficace des entrées et sorties pour les différents nœuds du workflow.
* Développement d'un moteur d'exécution :
  + Mettre en œuvre le parallélisme pour une exécution optimisée et accélérée des workflows.
  + Incorporer un système de contexte permettant de partager des ressources au sein du workflow.
  + Établir une gestion robuste des erreurs et des exceptions pour assurer la fiabilité du moteur d'exécution.
  + Incorporer un système de mise en cache et d'évaluation paresseuse pour améliorer les performances et l'efficacité du workflow.
  + Intégrer le support pour plusieurs langages de programmation (dans un premier temps seul le Javascript sera supporté) afin d'accroître la flexibilité dans le développement des workflows.
* Création d'une interface web :
  + Concevoir une interface web intuitive pour l'édition et l'exécution des workflows.
  + Permettre aux utilisateurs de créer, de modifier et d'exécuter des workflows de manière visuelle et interactive.

### Livrables

Les livrables du travail de Bachelor sont les suivants :

* Un rapport intermédiaire
* Un rapport final
* Un résumé publiable
* Une affiche
* Un site web servant à l’édition et la visualisation des workflows
* Un backend pour traiter l’exécution des workflows

### Déroulement du travail

Voici la liste des étapes du travail :

1. Familiarisation avec les technologies utilisées (React / React flow, Quarkus et GraalVM en particulier)
2. Revue de l’existant (Github actions, AWS pipeline, …)
3. Mockup du site web / Architecture haut niveau des différentes parties (frontend et workflow engine)
4. Réalisation d’une structure de donnée convenable pour la représentation d’un workflow et de ses nœuds
5. Recherche pour le workflow engine
   1. Gestion des erreurs
   2. Système de contexte pour partager des ressources (un fichier par exemple qui pourrait être traité par plusieurs nœuds différents)
   3. Mise en cache (couplé avec l’évaluation paresseuse du workflow)
   4. Parallélisation de l’exécution des nœuds
6. Architecture du workflow engine
7. Réalisation du frontend
8. Lien entre le backend et le frontend

**Table des matières**

[Préambule I](#_Toc160200584)

[Chapitre 1 Cahier des charges II](#_Toc160200585)

[1.1 Introduction II](#_Toc160200586)

[1.2 Objectifs II](#_Toc160200587)

[1.3 Livrables II](#_Toc160200588)

[1.4 Déroulement du travail II](#_Toc160200589)

[Résumé 1](#_Toc160200590)

[Chapitre 2 Introduction 2](#_Toc160200591)

[2.1 Gestion des entrées et sortie 2](#_Toc160200592)

[2.2 Gestion des erreurs 2](#_Toc160200593)

[2.3 Structure de données 2](#_Toc160200594)

[2.4 Gestion du cache 2](#_Toc160200595)

[2.5 Parallélisation 2](#_Toc160200596)

[2.6 Contexte 2](#_Toc160200597)

[2.7 Site web 2](#_Toc160200598)

[Chapitre 3 Conclusion 4](#_Toc160200599)

[Bibliographie 5](#_Toc160200600)

[Authentification 6](#_Toc160200601)

[Annexes 7](#_Toc160200602)

# Résumé

Cliquez ou appuyez ici pour entrer du texte.

## Introduction

Le but de ce travail de Bachelor est de réaliser un moteur d’exécution pour des workflows. Les points de réflexion principaux de ce travail seront :

* **Gestion des entrées et sortie** : Comment gérer les types qui rentrent et sortent des différents nœuds ? Comment faire pour que les types soient corrects alors qu’il est possible de changer de langage ? Que faire si un nœud ne retourne rien mais que l’on veut quand même indiquer qu’il doit s’exécuter avant un autre nœud ?
* **Gestion des erreurs** : Que faire lorsqu’un nœud est marqué comme ayant une erreur ? Stopper tout le workflow ? Continuer l’exécution au maximum ?
* **Structure de données**: Il est nécessaire d’avoir une structure de données adaptée pour la représentation du workflow et de ses nœuds.
* **Gestion du cache** : Le principe de l’évaluation paresseuse dans notre cas est de ne pas exécuter des nœuds que l’on a déjà exécuté et que l’on sait que le résultat reste inchangé.
* **Parallélisation** : On doit déterminer pour quels nœuds l’on peut effectuer leurs tâches en parallèle. On ne peut pas exécuter en même temps le nœud parent et son enfant par exemple étant donné que l’on a des dépendances sur le résultat de l’autre nœud.
* **Contexte** : Comment gérer le fait que plusieurs nœuds peuvent accéder au même fichier par exemple et, étant donné qu’il y’a du parallélisme possible. Comment faire si plusieurs nœuds veulent accéder au même fichier en même temps.
* **Site web** : Quel serait la meilleure interface possible pour un utilisateur ? Comment lui montrer ce qu’il se passe actuellement dans le workflow ? Quels sont les nœuds qui posent des problèmes ?

Toutes ses points vont être traités dans ce chapitre d’introduction.

### Gestion des entrées et sortie

### Gestion des erreurs

### Structure de données

### Gestion du cache

### Parallélisation

### Contexte

### Site web

.

## Conclusion

Cliquez ou appuyez ici pour entrer du texte.

# Bibliographie

# Authentification

Le soussigné, Peronetti Eric, atteste par la présente avoir réalisé seul ce travail et n’avoir utilisé aucune autre source que celles expressément mentionnées.

Peronetti Eric

Yverdon-les-Bains, le Sélectionner une date

# Annexes

**Glossaire**

**JIT**: Just in time

**ETL**: Extract, transform and load

**Index**

**Table des figures**

Aucune entrée de table d'illustration n'a été trouvée.

**Liste des tableaux**

Aucune entrée de table d'illustration n'a été trouvée.

**Liste des codes sources**

Aucune entrée de table d'illustration n'a été trouvée.