Mickaël BARON
École Nationale Supérieure de Mécanique et d'Aérotechnique
LIAS – EA 6315
1 avenue Clément Ader BP 40109
86961 Chasseneuil Cedex – France
Tél : +33 (0)5 49 49 80 73
baron@ensma.fr

Étude et optimisation du placement et de la synchronisation de fonctions de simulation distribuée sur infrastructure Kubernetes

Localisation : Site du Futuroscope, France
Date de début : début 2027 (flexible)
Durée : 6 mois
Encadrants : Mickaël BARON, Henri BAUER, Emmanuel GROLLEAU et Frédéric RIDOUARD *(l'encadrant dont le nom est souligné est l'encadrant référent : baron@ensma.fr)*
Mots clés : simulation distribuée, Kubernetes, synchronisation périodique et mémoire partagée.

Contexte du stage

Les simulations de systèmes complexes (systèmes cyber-physiques, jumeaux numériques, etc.) reposent fréquemment sur un ensemble de fonctions périodiques, potentiellement dotées d'un état interne (stateful) ou non (stateless), et s'exécutent de manière distribuée et communiquent en échangeant un grand nombre de variables (entrées/sorties). Pour concilier performance et cohérence temporelle, ces fonctions sont regroupées et affectées à des nœuds d'exécution. Chaque nœud exécute séquentiellement les fonctions qui lui sont assignées et fait progresser la simulation localement, jusqu'à un point de synchronisation où l'ensemble des nœuds échange les données produites/consommées et se resynchronisent.

Ce type d'architecture s'appuie sur un middleware de simulation qui assure à la fois le stockage local des données lues et écrites par les fonctions colocalisées (accessibles alors en mémoire partagée) ainsi que la sérialisation, le transfert réseau et la synchronisation des données lorsque les fonctions communicantes sont hébergées sur des nœuds distincts.

Dans ce contexte, le placement des fonctions sur les nœuds, tout comme leur ordonnancement local, constituent des leviers déterminants de l'efficacité globale de la simulation et de sa représentativité du procédé simulé. En effet, ces deux décisions déterminent directement l'équilibre entre calcul local et communication réseau. D'un côté, maximiser le parallélisme en isolant chaque fonction sur un nœud dédié réduit le temps de calcul local et facilite l'accès à des ressources spécifiques comme le GPU. Cette stratégie a cependant un coût : elle multiplie les synchronisations réseau, ce qui accroît la latence globale et sollicite davantage le plan de contrôle. À l'inverse, maximiser le regroupement en colocalisant sur un même nœud les fonctions fortement couplées permet de limiter ces synchronisations réseau grâce à l'exploitation de la mémoire partagée locale. Cette approche allonge mécaniquement le temps d'exécution séquentielle de chaque nœud et peut restreindre l'accès à certaines ressources spécialisées.

Il existe donc un compromis à arbitrer entre granularité du parallélisme et surcoût de syn-

Mickaël BARON
École Nationale Supérieure de Mécanique et d'Aérotechnique
LIAS – EA 6315
1 avenue Clément Ader BP 40109
86961 Chasseneuil Cedex – France
Tél : +33 (0)5 49 49 80 73
baron@ensma.fr

chronisation, dépendant du graphe de dépendances des fonctions (qui lit/écrit quoi), de leur coût de calcul, du volume de données échangées, et de la topologie/latence du réseau sous-jacent.

Objectifs du stage

L'objectif de ce stage est d'étudier comment transposer ce modèle d'exécution sur une infrastructure basée sur Kubernetes, où chaque fonction serait déployée comme un ou plusieurs Pods (unité atomique correspondant généralement à un conteneur), avec :

- un mécanisme d'échange de données adapté (mémoire partagée intra-pod versus communication réseau inter-pods/inter-nœuds) ;
- une stratégie de placement des pods (affinité/anti-affinité) reflétant les décisions de partitionnement des fonctions ;
- un mécanisme de synchronisation périodique entre pods, s'appuyant potentiellement sur les primitives Kubernetes natives (Services, ConfigMaps) ou sur un intergiciel additionnel déployé au sein du cluster.

Réalisation attendue

- Étudier les mécanismes Kubernetes pertinents (placement, communication, synchronisation) et leur correspondance avec le modèle d'exécution cible.
- Modéliser une étude de cas représentative, incluant un graphe de dépendances entre fonctions et plusieurs scénarios de partitionnement contrastés.
- Mettre en place un cluster Kubernetes comme banc d'essai, incluant les outils de mesure (latence, réseau, calcul).
- Développer et déployer l'étude de cas selon différentes stratégies de placement et de synchronisation.
- Généraliser les résultats sous forme de recommandations ou de prototype réutilisable.

Profil du candidat

Le candidat doit être en Master 2 en Informatique ou en dernière année de préparation d'un diplôme d'ingénieur spécialité Informatique et doit disposer de bonnes connaissances en systèmes distribués, réseaux et Kubernetes/conteneurisation.

Documents à fournir

Les deux premiers documents suivants sont indispensables ; sans eux, la candidature sera rejetée.

- Curriculum Vitae ;
- Notes de Master ou équivalent (si possible avec le classement) ;
- Autres documents utiles pour appuyer la candidature (ex. : projets, GitHub).