



# CAHIER D'ANALYSE

## Conteneurisation et gestion des workloads avec Magnum

Rédigé par:

- ESSONO EFFA Etienne Florian
- NGNINTEDEM FEUPI Vaillant

Sous la supervision : M.  
NGUIMBUS Emmanuel

Année Académique 2025 - 2026



# Plan

1. INTRODUCTION
2. CONTEXTE ET OBJECTIFS
3. PÉRIMÈTRE FONCTIONNEL
4. MODÉLISATION UML
5. CONCLUSION



# INTRODUCTION

Ce dossier d'analyse présente l'étude détaillée du système à mettre en place pour le déploiement et la gestion automatisée de clusters Kubernetes ou Docker Swarm via Magnum, avec intégration d'une orchestration hybride (VM + conteneurs) à l'aide de Heat.



# I. Contexte et objectif

## 1. CONTEXTE DU PROJET

L'évolution rapide des architectures applicatives, désormais fortement orientées vers le cloud, les micro-services et les conteneurs, impose aux entreprises de disposer de plateformes capables de gérer efficacement des workloads distribués et dynamiques. Les organisations doivent non seulement accélérer le déploiement des applications, mais également garantir la scalabilité, la résilience, la flexibilité et l'automatisation de l'infrastructure sous-jacente.

Dans un environnement multi-locataire ou hybride, la gestion manuelle de machines virtuelles, réseaux, clusters et applications devient complexe, coûteuse et source d'erreurs. Pour répondre à ces enjeux, OpenStack s'est imposé comme une solution cloud open source complète, et ses services *Magnum* (gestion des clusters de conteneurs) et *Heat* (orchestration d'infrastructure) permettent d'industrialiser, standardiser et automatiser le déploiement d'environnements conteneurisés modernes tels que Kubernetes et Docker Swarm.

Ce projet s'inscrit donc dans un contexte où la conteneurisation est devenue un standard technologique incontournable, et où les entreprises recherchent des solutions leur permettant d'unifier la gestion de leurs infrastructures VM et conteneurs, tout en optimisant l'efficacité opérationnelle et les coûts.



# **I. Contexte et objectif**

## **2. OBJECTIFS DU PROJET**

### **2.1 Objectif Général**

Mettre en place une solution permettant de déployer, orchestrer et automatiser la gestion de clusters Kubernetes ou Docker Swarm via le service Magnum, tout en intégrant des stacks hybrides (VM + conteneurs) grâce au service Heat.

L'objectif final est de fournir une plateforme cloud flexible, scalable et conforme aux standards, facilitant la gestion des workloads conteneurisés dans un environnement OpenStack moderne(cas de Devstack).



# I. Contexte et objectif

## 2.2 Objectifs Spécifiques

1. **Comprendre l'architecture OpenStack et le rôle de Magnum** en Identifier les principaux services OpenStack impliqués (**Nova, Neutron, Heat, Keystone**), Comprendre le rôle de Magnum dans la gestion des clusters de conteneurs.
2. **Déployer des clusters Kubernetes et/ou Docker Swarm automatiquement** via Magnum, en s'appuyant sur des cluster-templates standardisés ou personnalisés.
3. **Intégrer Magnum avec Heat** dans devstack afin de permettre la création de stacks hybrides combinant VMs, réseaux, stockage et clusters de conteneurs dans une même description déclarative.
4. **Automatiser le cycle de vie des clusters**, depuis la création jusqu'à la mise à l'échelle, le monitoring, la sécurisation et la suppression.



# I. Contexte et objectif

## 2.2 Objectifs Spécifiques

**5. Mettre en place un système de monitoring et de supervision** (Prometheus, Grafana ou équivalent) permettant de surveiller l'état, les performances et la santé des clusters en terme de CPU, mémoire, nœuds, pods.

**5. Implémenter des mécanismes de scaling automatique** pour optimiser l'utilisation des ressources.

**8. Livrer une documentation complète**, comprenant l'architecture, les procédures d'installation, les scripts, les tests et les diagrammes UML.



## II. Périmètre Fonctionnel

Nous nous attarderons le plus sur :

- Création de modèles de clusters (cluster templates).
- Provisionnement de clusters (masters/nodes) et récupération des credentials (**kubeconfig**) pour l'accès au cluster afin de mieux **Compréhension du cycle de vie**(**crée des VM, configure Kubernetes, génère les accès (kubeconfig).**).
- Orchestration via Heat : dans le but de Comprendre la **logique d'orchestration (Syntaxe Heat (HOT), Dépendances entre ressources, Déploiement coordonné (VM simple, Cluster Magnum).**).
- Automatisation via CLI, Ansible et/ou Terraform.
- Tests de montée en charge (scale up/scale down), tests de tolérance de panne.  
Monitoring de base (Prometheus + Grafana ou équivalent), logs centralisés.





# III. Modélisation UML

## 1. Listes des acteurs et rôles

| Acteurs                                      | Rôles  |
|--|--|
| Administrateurs Cloud<br>(Effa et Feupi)     | Configure OpenStack, gère les projets, quotas, réseaux, clés, politiques de sécurité, et valide les templates. |
| Ingénieur DevOps<br>(Ngnintedem Feupi)       | Automatisation des déploiements, création/gestion des clusters, intégration CI/CD, scaling, monitoring.        |
| Développeur d'Applications<br>(EFFA Etienne) | Déploie des applications sur les clusters Kubernetes/Swarm fournis.  |
| Service Magnum                               | Service OpenStack responsable de la gestion complète des clusters (création, scaling, suppression).            |
| Service Heat                                 | Orchestration des stacks hybrides (VM + clusters). Exécute les templates HOT.                                  |

# III. Modélisation UML

## 2. Description textuelle d'un cas d'utilisation

### UC01 – CRÉER UN CLUSTER KUBERNETES VIA MAGNUM

| Élément          | Description  |
|------------------|--|
| TITRE            | UC01 – Créer un cluster Kubernetes   |
| RÉSUMÉ           | Permet à un administrateur ou ingénieur DevOps de créer un cluster Kubernetes en utilisant un cluster-template Magnum.   |
| ACTEUR(S)        | Administrateur Cloud, Ingénieur DevOps, Service Magnum   |
| VERSION          | 1.0  |
| PRÉCONDITION     | OpenStack fonctionnel, cluster-template déjà configuré, quotas disponibles, réseau accessible.   |
| SCÉNARIO NOMINAL | <ol style="list-style-type: none"><li>1. Utilisateur S'authentifie sur <b>Keystone</b> via l'interface Horizon ou le CLI.</li><li>2. Utilisateur Consulte les modèles disponibles : magnum cluster-template-list. Il choisit un template (<u>ex</u>: k8s-fedora-coreos).</li><li>3. Utilisateur Saisit la commande de création en spécifiant le nom, le template et le nombre de nœuds (--node-count 2 --master-count 1).</li><li>4. <b>Magnum</b> Réceptionne la requête, valide les droits de l'utilisateur et génère dynamiquement un <b>Template Heat (HOT)</b>.</li><li>5. <b>Heat</b> Reçoit le template de Magnum et crée une <b>Stack</b>. Il appelle <b>Neutron</b> pour le réseau, <b>Cinder</b> pour les volumes et <b>Nova</b> pour les VMs.</li><li>6. <b>Nova/Glance</b> Lance les instances à partir de l'image (<u>ex</u>: Fedora-CoreOS). Au démarrage, un script de configuration (<b>Cloud-init</b> ou <b>Ignition</b>) installe les composants Kubernetes.</li></ol> |

# III. Modélisation UML

## 2. Description textuelle d'un cas d'utilisation

### UC01 – CRÉER UN CLUSTER KUBERNETES VIA MAGNUM

|                             |   |
|-----------------------------|---|
|                             | <ol style="list-style-type: none"><li>7. <b>Magnum</b> Surveille l'état de la Stack Heat. Une fois que tous les nœuds répondent "Ready", il passe le statut du cluster à CREATE_COMPLETE.</li><li>8. Utilisateur Récupère le fichier de configuration (magnum cluster-config &lt;name&gt;) pour piloter son cluster avec kubectl.</li></ol> |
| SCÉNARIIS<br>ALTERNATIFS    | Template invalide ; quotas insuffisants ; réseau indisponible ; échec Nova.   |
| POST-CONDITION DE<br>SUCCÈS | Le cluster Kubernetes est opérationnel.   |
| POST-CONDITION<br>D'ÉCHEC   | Stack interrompue, log d'erreurs généré, cluster non créé.  |

# III. Modélisation UML

## 2. Description textuelle d'un cas d'utilisation

### UC02 – DÉPLOYER UNE STACK HYBRIDE (VM + CLUSTER) VIA HEAT

| Élément          | Description   |
|------------------|---|
| TITRE            | UC02 – Déployer un stack hybride via Heat   |
| RÉSUMÉ           | Ce cas permet de créer simultanément des VMs et un cluster conteneurisé en utilisant un template Heat (HOT).  |
| ACTEUR(S)        | Ingénieur DevOps, Service Heat, Service Magnum  |
| VERSION          | 1.0   |
| PRÉCONDITION     | Template Heat valide, ressources suffisantes, réseau créé.  |
| SCÉNARIO NOMINAL | <ol style="list-style-type: none"><li>1. Utilisateur Soumet le template <b>HOT (YAML)</b> qui définit une ressource <u>OS::Nova::Server</u> (ex: Base de données) et une ressource <u>OS::Magnum::Cluster</u> (app conteneurisée).</li><li>2. <b>Heat Engine</b> Analyse le template et crée un <b>graphe de dépendances</b>. Il identifie ce qui doit être créé en premier (ex: le réseau).</li><li>3. <b>Heat</b> -&gt; <b>Neutron</b> Provisionne le réseau privé, le sous-réseau et le routeur qui seront partagés par les VMs et le cluster.</li><li>4. <b>Heat</b> -&gt; <b>Nova</b> Lance la création de la VM de base de données. Heat attend que Nova confirme que la VM est en état "ACTIVE".</li><li>5. <b>Heat</b> -&gt; <b>Magnum</b> Envoie la requête de création du cluster Kubernetes/Swarm. Heat transmet l'ID du réseau déjà créé pour que le cluster y soit rattaché.</li></ol> |

# III. Modélisation UML

## 2. Description textuelle d'un cas d'utilisation

### UC02 – DÉPLOYER UNE STACK HYBRIDE (VM + CLUSTER) VIA HEAT

|                             |  |
|-----------------------------|--|
|                             | <ol style="list-style-type: none"><li>6. <b>Magnum</b> Pilote sa propre stack interne pour déployer les nœuds du cluster.</li><li>7. <b>Heat Engine</b> Agrège les statuts. Une fois que la VM Nova est prête ET que le cluster Magnum est opérationnel, <u>la stack globale</u> passe en CREATE_COMPLETE.</li><li>8. Ingénieur DevOps Accède à l'application : les conteneurs dans le cluster Magnum peuvent maintenant requêter la base de données sur la VM Nova via l'IP privée.</li></ol> |
| SCÉNARI<br>ALTERNATIFS      | Dépendances manquantes, erreurs réseau, quotas insuffisants, échec de création du cluster.   |
| POST-CONDITION<br>DE SUCCÈS | Stack hybride opérationnelle et fonctionnelle.   |

# III. Modélisation UML

## 2. Description textuelle d'un cas d'utilisation

### UC03 – METTRE À L'ÉCHELLE (SCALING) UN CLUSTER

| Élément             | Description  |
|---------------------|--|
| TITRE               | UC03 – Mettre à l'échelle un cluster   |
| RÉSUMÉ              | Permet d'augmenter ou réduire dynamiquement le nombre de nœuds dans un cluster.  |
| ACTEUR(S)           | Ingénieur DevOps, Service Magnum   |
| VERSION             | 1.0  |
| PRÉCONDITION        | Cluster déjà opérationnel, ressources disponibles.   |
| SCÉNARIO<br>NOMINAL | <ol style="list-style-type: none"><li><b>Demande</b> : L'ingénieur DevOps envoie une requête pour modifier le nombre de nœuds (ex : passer de 3 à 5 workers).</li><li><b>Modification infra</b> : Magnum pilote la création (ou la suppression) des VMs workers nécessaires via l'infrastructure OpenStack.</li><li><b>Mise à jour logique</b> : Le cluster Kubernetes/Swarm détecte automatiquement les nouveaux nœuds et les intègre.</li><li><b>Rééquilibrage</b> : L'orchestrateur (ex : Kubernetes) déplace certains conteneurs (pods) vers les nouveaux nœuds pour optimiser les performances.</li></ol> |



# III. Modélisation UML

## 2. Description textuelle d'un cas d'utilisation

### UC03 – METTRE À L'ÉCHELLE (SCALING) UN CLUSTER

|                             |  |
|-----------------------------|--|
| SCÉNARIO<br>ALTERNATIFS     | <ol style="list-style-type: none"><li>1. <b>Nœuds non joignables</b> : Les nouvelles VMs sont créées mais ne parviennent pas à rejoindre le cluster à cause d'un problème réseau.</li><li>2. <b>État DEGRADED</b> : Le cluster reste fonctionnel mais avec moins de nœuds que prévu, ce qui interrompt l'opération de scaling.</li></ol> |
| POST-CONDITION<br>DE SUCCÈS | Cluster mis à l'échelle correctement.  |
| POST-CONDITION<br>D'ÉCHEC   | Cluster dans un état DEGRADED ou scaling interrompu.   |

# III. Modélisation UML

## 2. Description textuelle d'un cas d'utilisation

### UC04 – Automatiser le déploiement d'un cluster via script/API

| Élément          | Description   |
|------------------|---|
| TITRE            | UC04 – Automatisation du déploiement de clusters.   |
| RÉSUMÉ           | Automatiser la création de clusters via des scripts CLI (Bash, Python, Terraform, Ansible) pour réduire l'intervention manuelle et améliorer la reproductibilité.   |
| ACTEUR(S)        | Ingénieur DevOps, Service Magnum, Service Heat.   |
| DATE DE CRÉATION | 20/11/2025.   |
| VERSION          | 1.0.  |
| PRÉCONDITION     | Scripts disponibles ; accès API opérationnel ; cluster-template déjà configuré.   |
| SCÉNARIO NOMINAL | <ol style="list-style-type: none"><li>1. L'ingénieur DevOps exécute un playbook (Ansible) ou un script (Terraform/Bash).</li><li>2. Le script appelle l'API via la commande <code>magnum cluster-create</code> avec les paramètres requis (nom, template, taille).</li><li>3. Magnum contacte Heat pour orchestrer le déploiement de l'infrastructure.</li><li>4. Nova, Neutron et Cinder provisionnent les ressources (VMs, réseaux, stockage).</li><li>5. Le script récupère automatiquement le fichier <code>kubeconfig</code> une fois le cluster prêt.</li></ol> |



# III. Modélisation UML

## 2. Description textuelle d'un cas d'utilisation

### UC04 – Automatiser le déploiement d'un cluster via script/API

|                             |  |
|-----------------------------|--|
| SCÉNARIO<br>ALTERNATIFS     | <p><b>Modèle invalide</b> : Le template spécifié n'existe pas ou est corrompu.</p> <p><b>Mauvais paramètres</b> : Erreur dans les arguments du script (<u>ex</u>: nombre de nœuds négatif).</p> <p><b>API Keystone indisponible</b> : Échec de l'authentification empêchant l'appel API.</p> |
| POST-CONDITION<br>DE SUCCÈS | Un cluster fonctionnel est déployé de manière totalement automatique.  |
| POST-CONDITION<br>D'ÉCHEC   | Échec du script avec génération de logs d'erreur ; aucune ressource n'est créée ou laissée en suspens.   |

# III. Modélisation UML

## 2. Description textuelle d'un cas d'utilisation

### UC05 – SUPPRIMER UN CLUSTER

| Élément                  | Description   |
|--------------------------|---|
| TITRE                    | UC04 – Supprimer un cluster.  |
| RÉSUMÉ                   | Permet à un utilisateur autorisé de supprimer proprement un cluster Magnum.   |
| ACTEUR(S)                | Administrateur Cloud, Ingénieur DevOps, Service Magnum.   |
| VERSION                  | 1.0.  |
| PRÉCONDITION             | Le cluster existe et n'est pas utilisé par un service critique.   |
| SCÉNARIO NOMINAL         | <ol style="list-style-type: none"><li>1. L'utilisateur exécute la commande magnum cluster-delete.</li><li>2. Magnum identifie et supprime les composants logiciels (workers, masters).</li><li>3. Nova procède à la suppression des Machines Virtuelles.</li><li>4. Heat nettoie les stacks résiduelles (réseaux, volumes).</li></ol> |
| SCÉNARIOS ALTERNATIFS    | <p><b>Ressources verrouillées :</b> Certaines ressources ne peuvent être supprimées car elles sont utilisées ailleurs.</p> <p><b>Dépendances actives :</b> Des liens externes empêchent la suppression complète.</p>  |
| POST-CONDITION DE SUCCÈS | Le cluster est totalement supprimé et les ressources (quotas) sont libérées.  |
| POST-CONDITION D'ÉCHEC   | Le cluster reste dans un état de suppression partielle (ressources orphelines).   |



## III. Modélisation UML

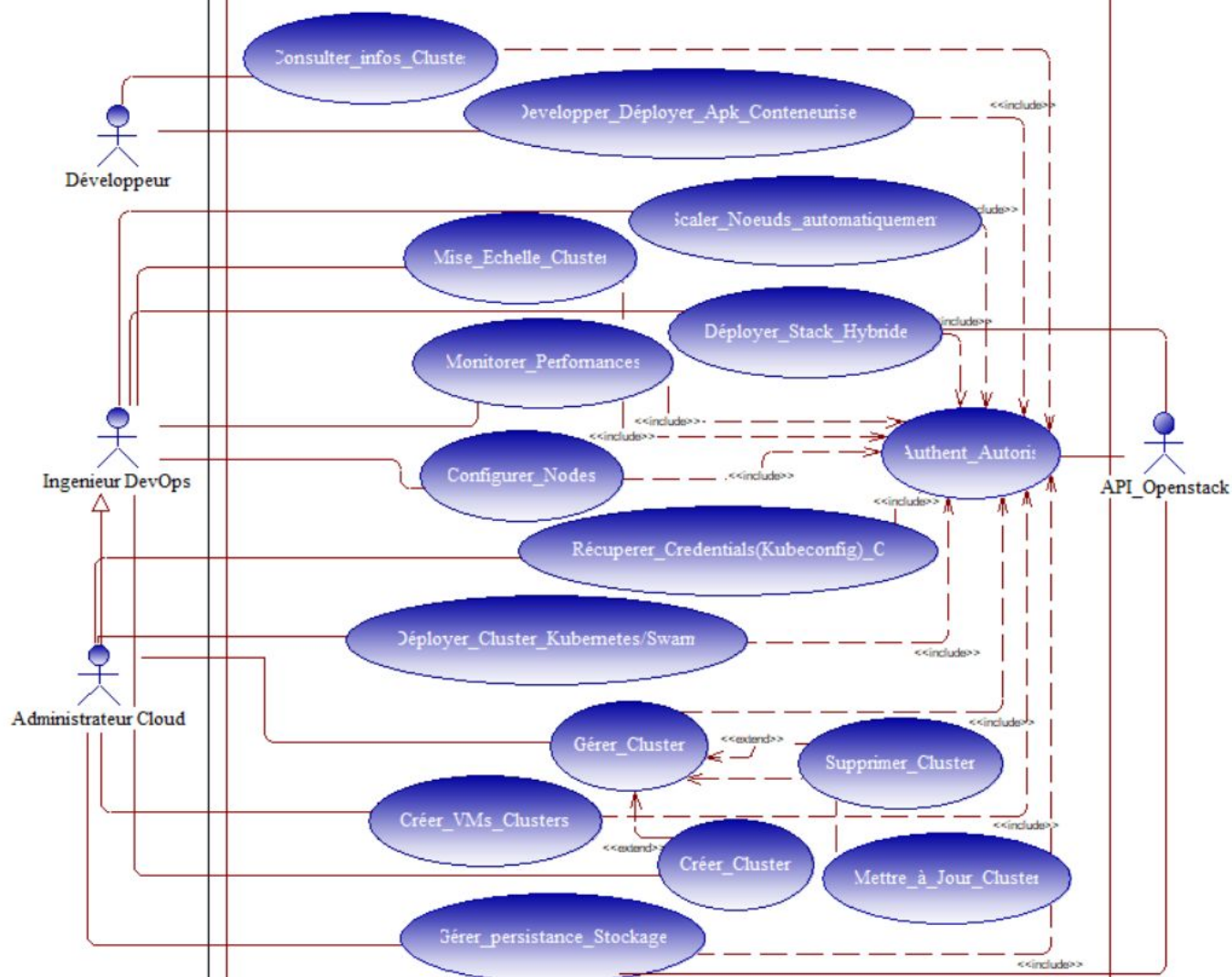
Diagramme de Cas d'Utilisation :

**Acteurs principaux :**

- Admin Cloud,
- Ingénieur DevOps,
- Développeur.

**Cas d'utilisation principaux :**

- Gérer clusters,
- scaler cluster,
- déployer stack.
- Récupérer les credentials d'un cluster





## Conclusion

**Ce dossier constitue la base d'analyse complète permettant de passer à la phase d'implémentation et de conception opérationnelle. Les acteurs, cas d'utilisation et logiques système sont désormais définis pour permettre la création d'un système Magnum + Heat robuste et opérationnelle en environnement de test.**