

# Système de gestion de données distribué

## Rapport de projet

Cours de systèmes distribués  
M1 Informatique Fondamentale  
ENS de Lyon  
Printemps 2018

[guillaume.coiffier@ens-lyon.fr](mailto:guillaume.coiffier@ens-lyon.fr)

L'intégralité du code source est disponible à l'adresse suivante :  
<https://github.com/GCoiffier/Distributed-Data-Manager-Erlang>

## Table des matières

<b>1</b>	<b>Description du programme</b>	<b>1</b>
1.1	Topologie du réseau . . . . .	1
<b>2</b>	<b>Initialisation du serveur</b>	<b>1</b>
<b>3</b>	<b>Méthode de stockage</b>	<b>1</b>
<b>4</b>	<b>Interface client</b>	<b>1</b>
<b>5</b>	<b>Gestion des requêtes d'un client par les noeuds de requête</b>	<b>1</b>
<b>6</b>	<b>Robustesse du réseau</b>	<b>1</b>
<b>7</b>	<b>Complexité de communication</b>	<b>1</b>

# 1 Description du programme

## 1.1 Topologie du réseau

Nous avons fait le choix d'imposer une topologie précise à notre réseau, plutôt que de gérer une topologie arbitraire. Les conséquences de ce choix sont discutées en chapitre 5.

On distinguera, dans le réseau, trois types d'agents :

- Le noeud **master**, responsable des connections entrantes au réseau.
- Les noeuds de requête (**query**), responsables de la gestion des requêtes de stockage et d'accès aux données.
- Les noeuds de stockage (**storage**) responsables... du stockage des données.

Le noeud **master** est relié à l'intégralité des noeuds **query**. Ces derniers forment une clique. Dans la pratique, ils ont simplement accès à la liste des Pid de tous les autres noeuds **query**. Enfin, les noeuds de stockages sont reliés à un ou plusieurs noeuds **query** et forment leurs fils.

## 2 Initialisation du serveur

Le noeud master (module `server.erl`) est responsable de l'Initialisation du serveur. Il est lancé lorsque l'on compile le module `server`. Il a la tâche d'initialiser N noeuds de requêtes, qui vont à leur tour initialiser M noeuds de stockage chacun. Les valeurs par défaut de N et M sont respectivement 10 et 5.

## 3 Méthode de stockage

Le stockage des données est assuré par un  
Fonctionnement du programme

## 4 Interface client

L'interface du client est décrite par le module `client.erl`. Elle comprend les fonctions suivantes :

- `connect/1`
- `send_data/2`
- `fetch_data/`
- `release_data`

## 5 Gestion des requêtes d'un client par les noeuds de requête

Discussion sur les performances et la robustesse de la solution choisie

## 6 Robustesse du réseau

## 7 Complexité de communication