



Sécurité Info Architecture Services Web

Un cours de Yann Fornier

Présentation

Yann Fornier

Ingénieur en Aérospatial - Data Manager - Support à la digitalisation dans la stratégie d'entreprise.



Présentation

Yann Fornier

Enseignant en Informatique - Ecole d'Ingénieur, Ecole de Commerce, Université

Domaines privilégiés : Informatique Quantique, Cloud Computing, Blockchain





Cours 1

Conception d'une architecture web

Un cours de Yann Fornier



Plan

Session 1 : Conception d'une architecture Web

- Les critères de choix d'une architecture web (scalabilité, flexibilité, etc.)
- La modélisation des services web avec les diagrammes de séquence et de cas d'utilisation
- Les patterns d'architecture web couramment utilisés (API Gateway, Load Balancer, etc.)
- **Exercice** : concevoir une architecture web en utilisant les différents outils vus en cours



Les critères de choix d'une architecture web

Scalabilité

Flexibilité

Disponibilité

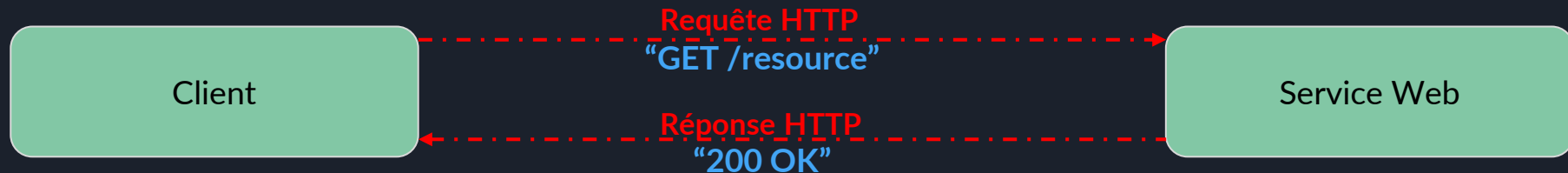
Coût

Sécurité

La modélisation des services web avec les diagrammes de séquences et cas d'utilisation

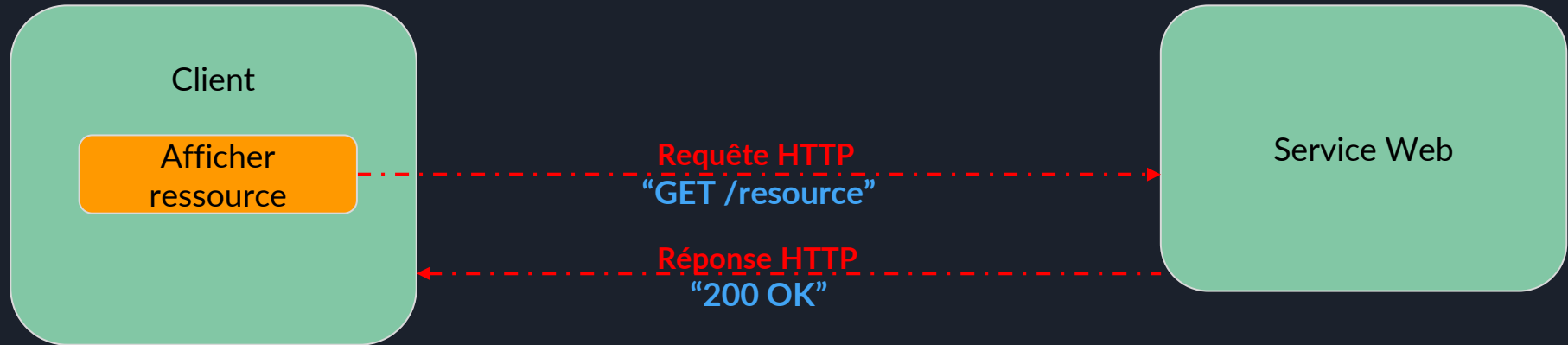
La modélisation des services web peut être faite à l'aide de différents diagrammes, notamment les diagrammes de séquence et de cas d'utilisation.


Le diagramme de séquence permet de modéliser les interactions entre les différents éléments d'un système, en particulier les échanges de messages entre les services. Il permet de visualiser l'ordre dans lequel les messages sont envoyés et reçus, ainsi que les conditions de déclenchement de chaque message.



Le diagramme de cas d'utilisation

Le diagramme de cas d'utilisation permet de modéliser les différentes actions que peuvent réaliser les utilisateurs d'un système, ainsi que les rôles et les responsabilités de chaque acteur impliqué. Il permet de visualiser les limites et les contraintes du système, ainsi que les interactions avec d'autres systèmes ou services.





Les patterns d'architecture web couramment utilisés

API Gateway

Load Balancing

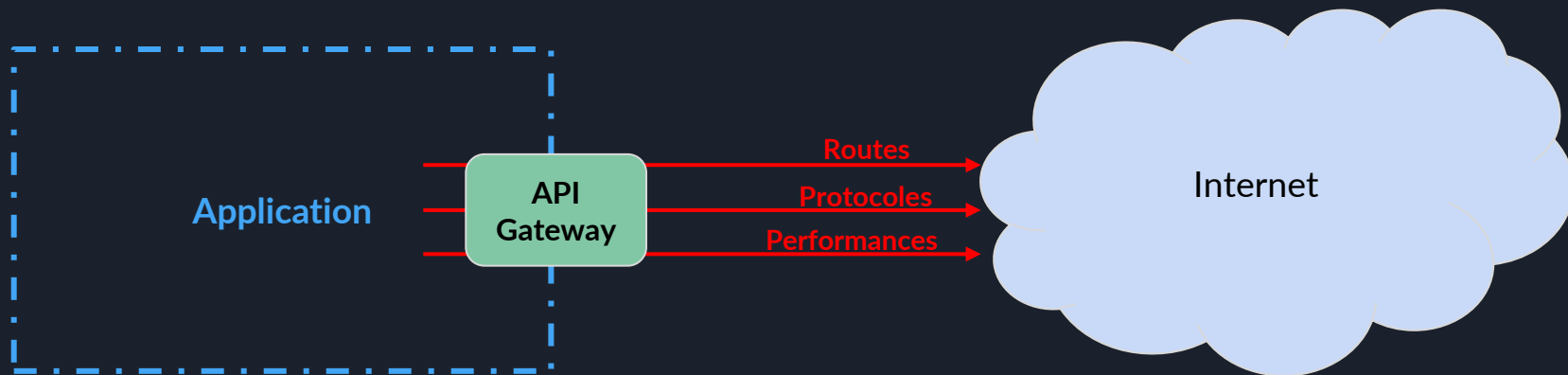
Circuit Breaker

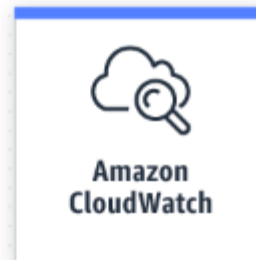
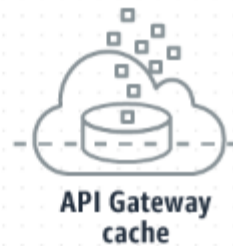
Mise en cache

Event Bus

API Gateway

Une API Gateway est une passerelle de communication qui sert de point d'entrée unique et centralisé pour les différents services et composants de l'application. Elle permet de mettre en place une interface de communication standardisée et normalisée, en gérant les routes, les protocoles, les sécurités, les performances, etc. L'API Gateway s'appuie sur des technologies comme les APIs REST, les WebHooks, les Events, etc. pour gérer les échanges de données et de messages entre les différents acteurs de l'application.







Pourquoi utiliser une API Gateway ?

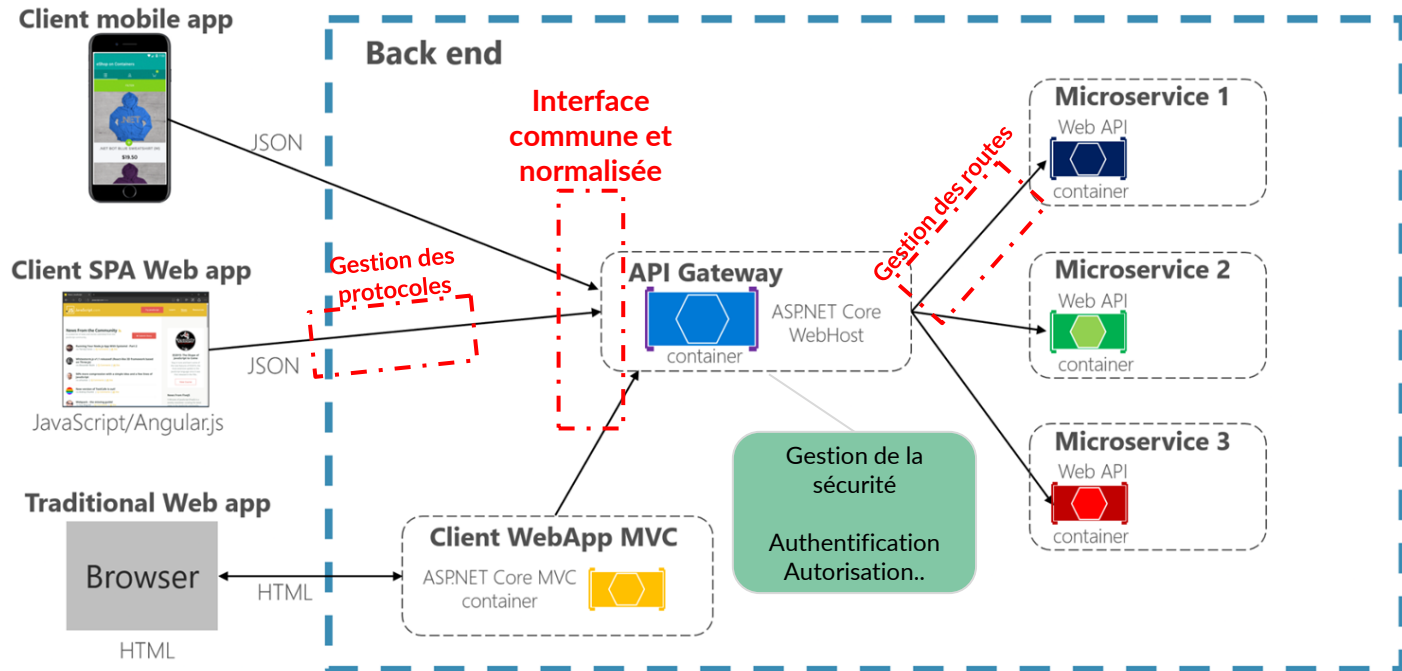
Simplification, sécurisation de
l'intégration et de l'interopérabilité
des services

Gestion des routes et des
protocoles

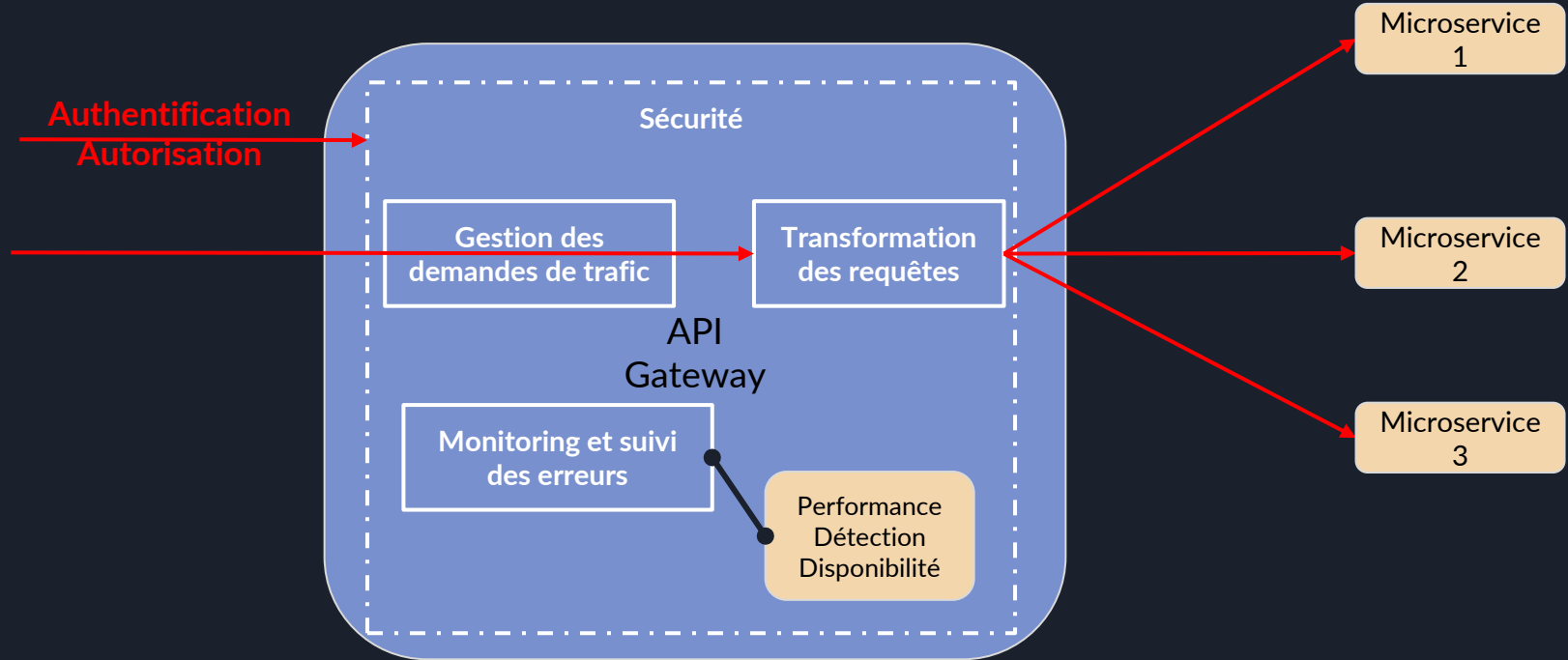
Gestion des sécurités et des
performances

Pourquoi utiliser une API Gateway ?

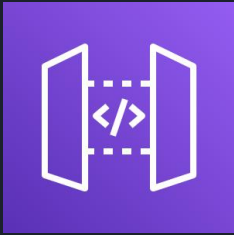
Using a single custom **API Gateway service**



Exemples d'utilisation d'une API Gateway



Exemples de logiciels d'API Gateway



AWS
API Gateway



Tyk.io



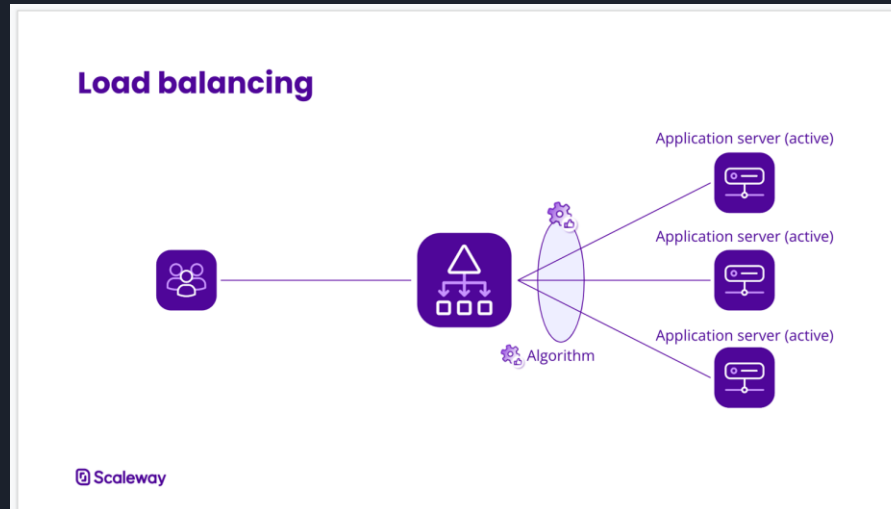
Express
Gateway



Kong

Load Balancing

Le load balancing est une technique utilisée pour répartir la charge de travail sur plusieurs serveurs ou instances de traitement afin d'optimiser la performance et la disponibilité d'un système. Il est souvent utilisé dans les environnements de production pour gérer les pics de charge et assurer la continuité de service.

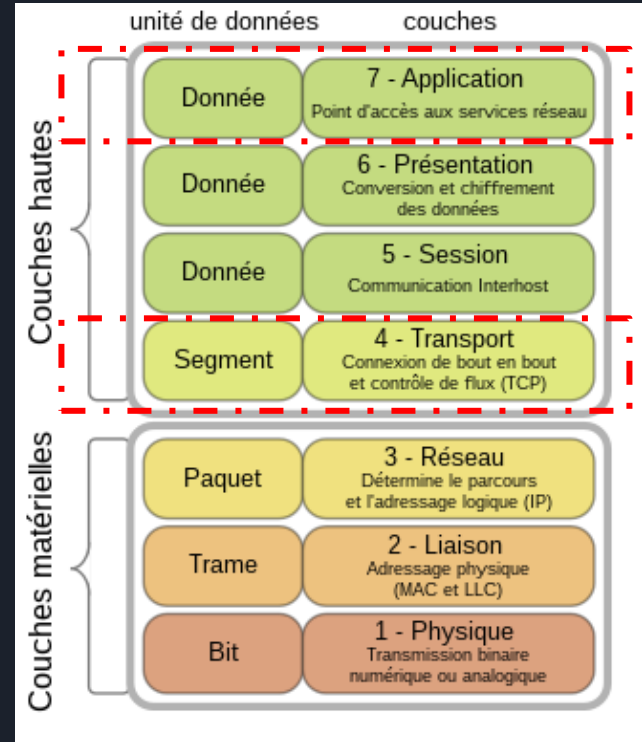


Load Balancing

Il existe plusieurs types de load balancing, notamment le load balancing de niveau 7 (basé sur les couches 7 du modèle OSI) et le load balancing de niveau 4 (basé sur les couches 4 du modèle OSI).

Le load balancing de niveau 7 prend en compte les informations de l'application, comme les en-têtes HTTP, pour déterminer comment répartir la charge de travail.

Le load balancing de niveau 4 se base uniquement sur les informations de transport, comme l'adresse IP et le numéro de port, pour répartir la charge de travail.



Le modèle OSI

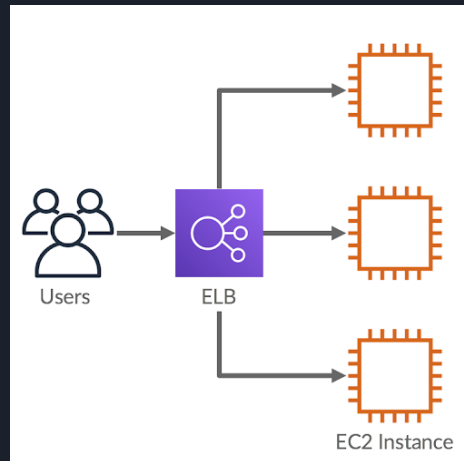


Load Balancing

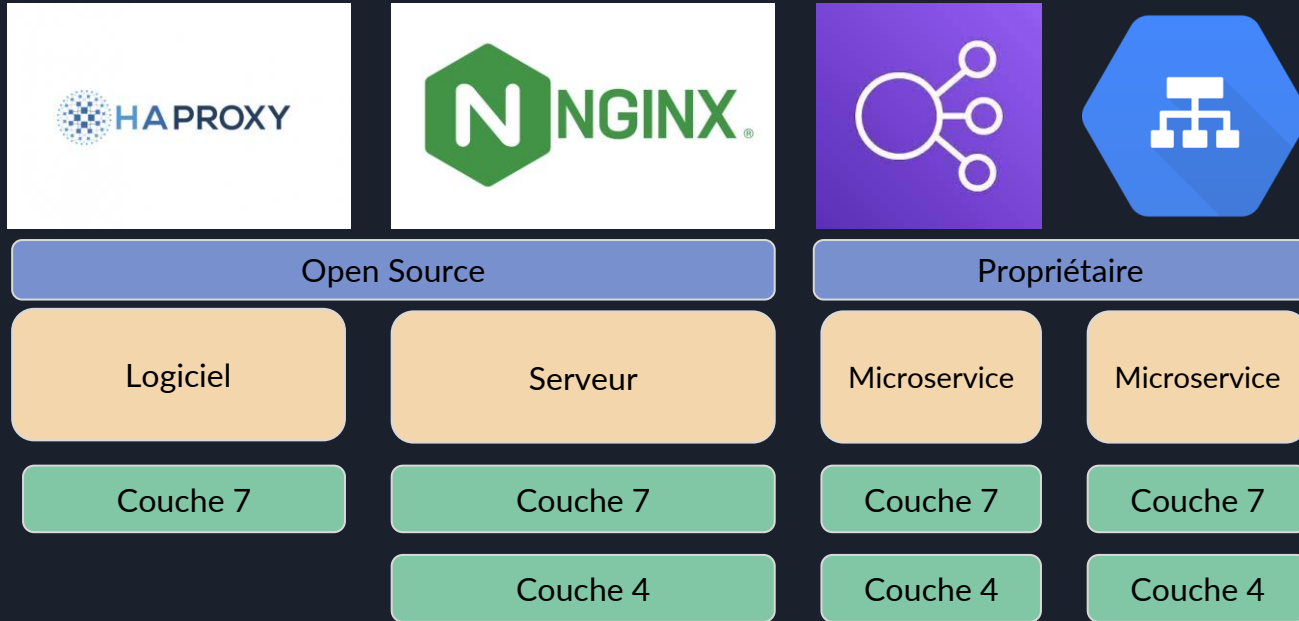
Le load balancing peut être mis en place de différentes manières, comme en utilisant un équilibreur de charge dédié ou en utilisant un logiciel de load balancing sur un serveur existant. Il est également possible de configurer le load balancing en utilisant un service de cloud computing, comme Amazon Elastic Load Balancing ou Google Cloud Load Balancer.

Le Load Balancer

Un load balancer (équilibreur de charge) est un **composant** ou un **service** qui permet de répartir la charge de travail sur plusieurs serveurs ou instances de traitement afin d'optimiser la performance et la disponibilité d'un système. Il est généralement utilisé dans les environnements de production pour gérer les pics de charge et assurer la continuité de service.

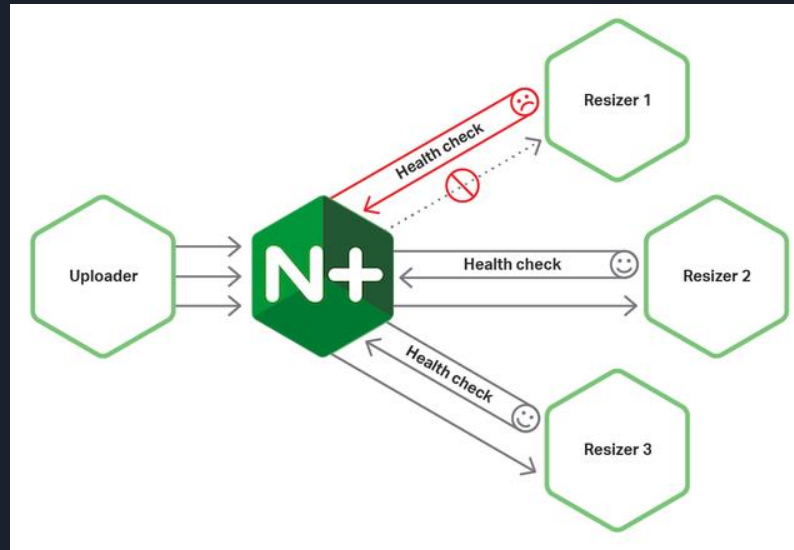


Les technologies du Load Balancer



Le Circuit Breaker

Circuit Breaker : ce pattern consiste à mettre en place un mécanisme de protection pour prévenir les erreurs de cascades et les dégradations de performance dans un environnement distribué. Le Circuit Breaker permet de couper temporairement l'accès à un service défaillant, afin de limiter les dommages et de permettre une récupération progressive.





Le Circuit Breaker

Un circuit breaker est un patron de conception qui permet de protéger une application contre les pannes de sous-systèmes en désactivant temporairement les appels de service à ces sous-systèmes qui échouent de manière répétée. On l'utilise généralement en parallèle d'un Load Balancer pour améliorer les performances et la (haute) disponibilité.



Processus du circuit breaker

Le circuit breaker suit un processus simple :

1. Lorsqu'un appel de service échoue, le circuit breaker enregistre cet échec.
1. Si un nombre suffisamment élevé d'appels de service échouent de manière consécutive, le circuit breaker passe en mode "ouvert" et bloque tous les appels de service futurs vers le sous-système en question.
1. Pendant que le circuit breaker est ouvert, il surveille en continu si les appels de service réussissent à nouveau.
1. Si un nombre suffisamment élevé d'appels de service réussissent de manière consécutive, le circuit breaker passe en mode "fermé" et autorise à nouveau les appels de service vers le sous-système.



Exemples de circuit breaker

Dans une application de commerce en ligne, un circuit breaker peut être utilisé pour protéger l'application contre les pannes du système de paiement. Si le système de paiement échoue de manière répétée, le circuit breaker peut être activé pour bloquer temporairement les appels de service vers ce système et empêcher l'application de se bloquer.

Exemples de circuit breaker



Bibliothèque open source,
développée par Netflix.

Cible : applications Java.



Bibliothèque open source

Plus légère et facile
d'utilisation

Cible : applications Java.



Bibliothèque open source,
développée par Lyft

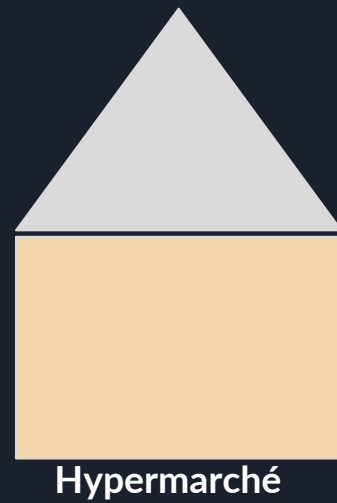


Mise en cache

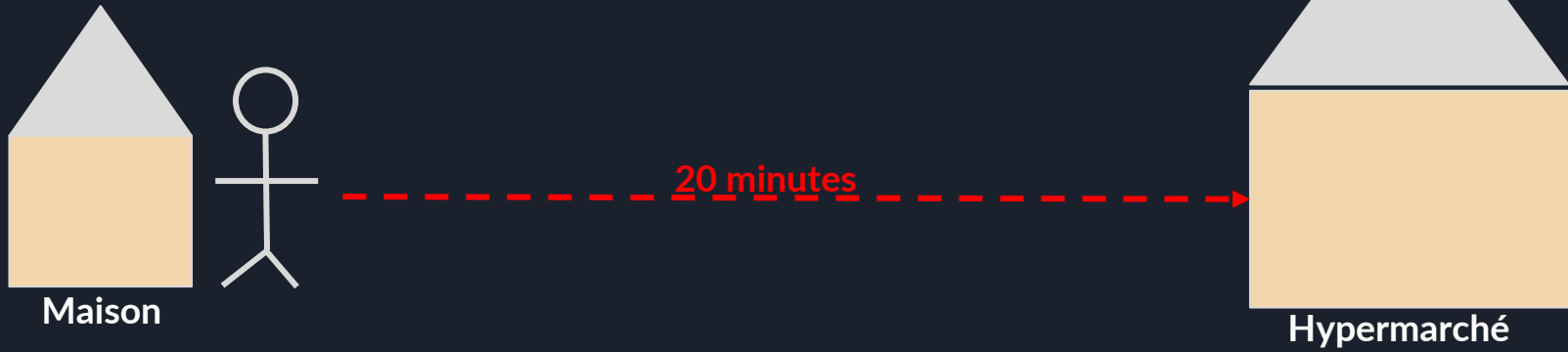




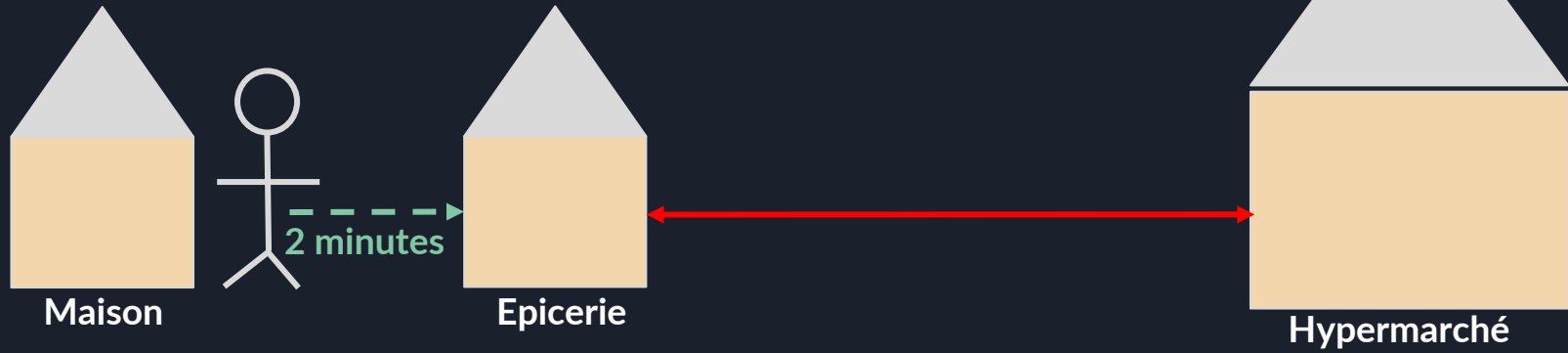
Mise en cache



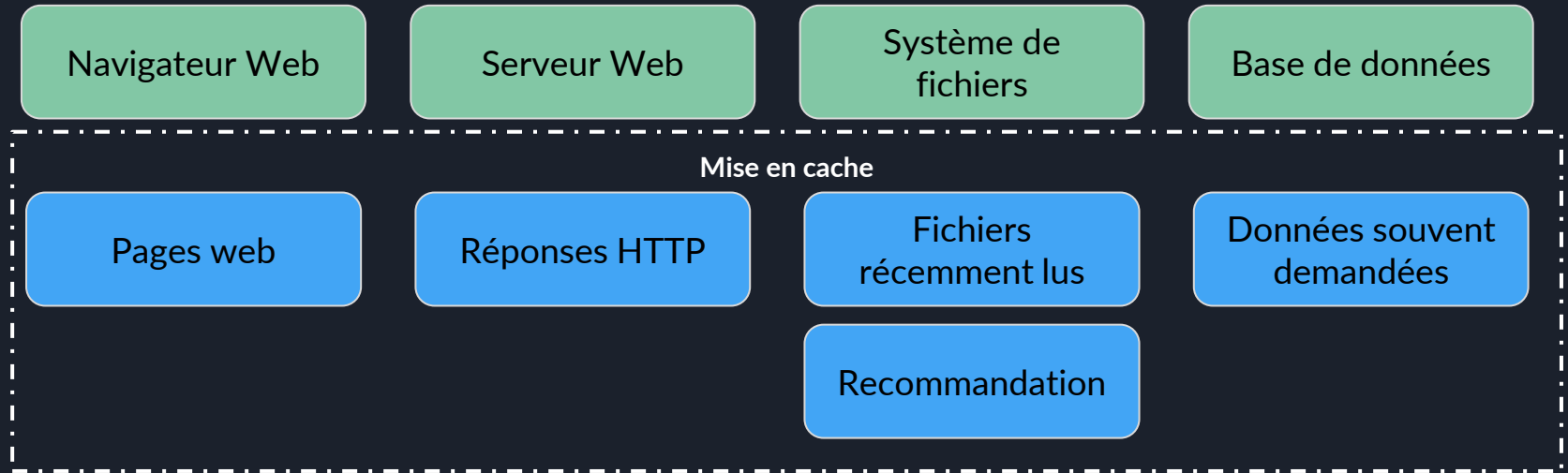
Mise en cache



Mise en cache



Exemples de mise en cache



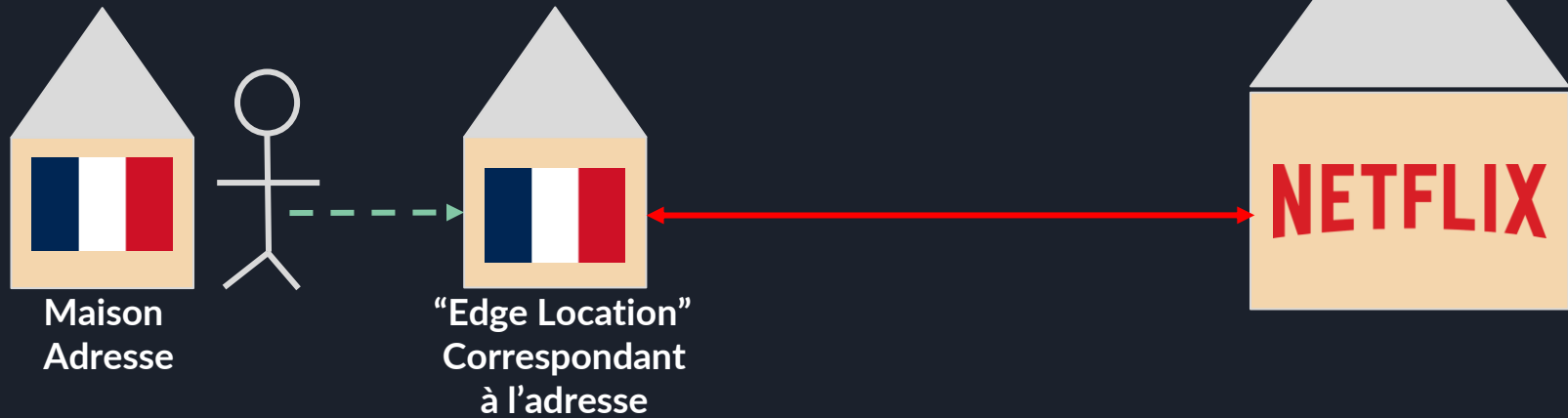


NordVPN®

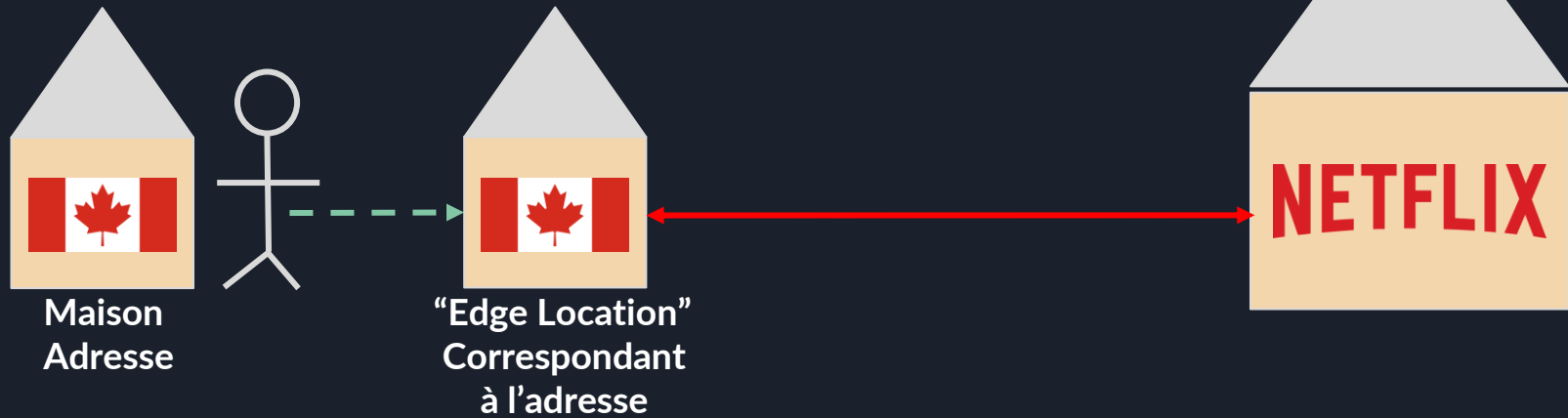


“Pourquoi en utilisant un VPN, vous pouvez accéder au catalogue Netflix du monde entier ?”

Mise en cache



Mise en cache



Exemples de logiciels de mise en cache



Memcached



Redis



Varnish



Squid

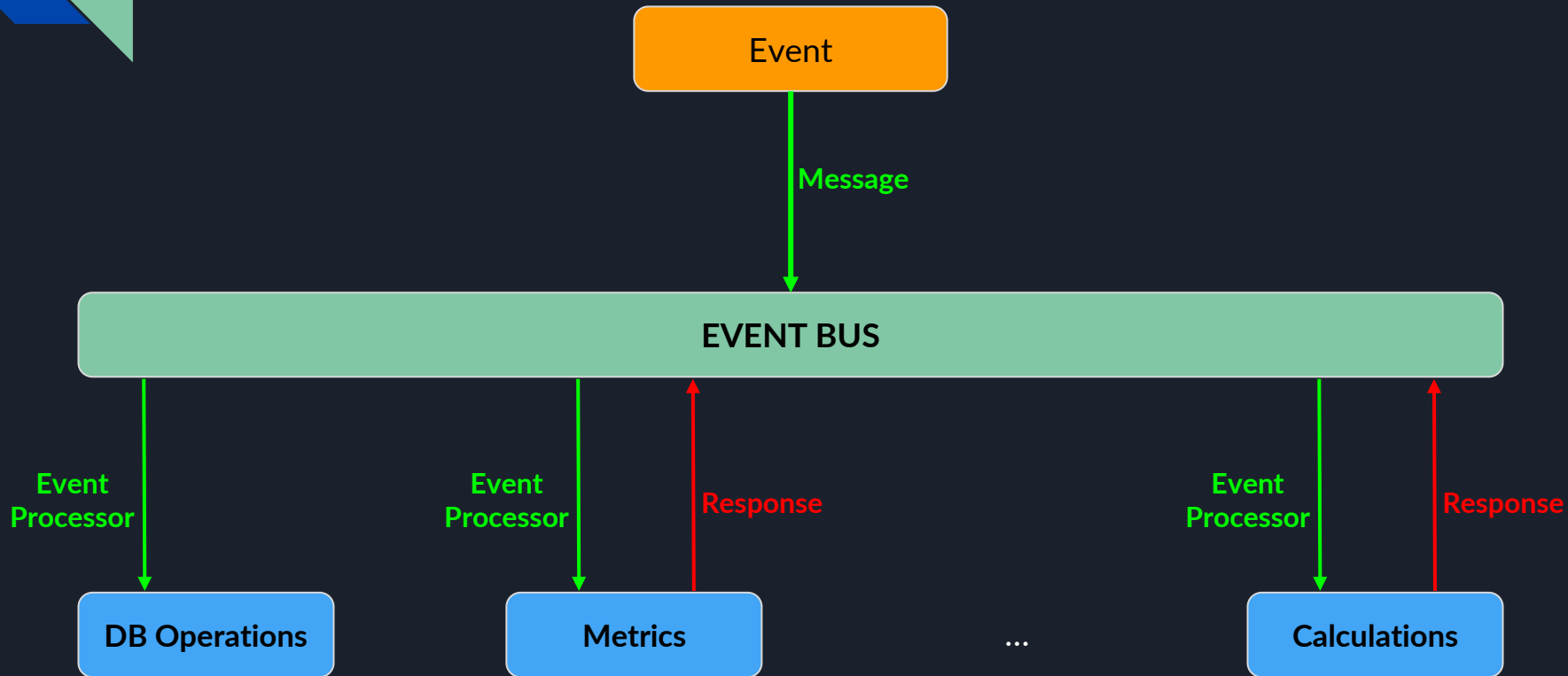


L'Event Bus

Ce pattern consiste à mettre en place un mécanisme de **communication asynchrone** entre les différents éléments d'une application, en utilisant **un bus de messages ou un système de notification**. L'Event Bus permet de découpler les échanges de messages entre les services, de manière à améliorer la flexibilité et la robustesse de l'application.

Il s'agit d'une couche de communication indépendante qui permet de **découpler** les différents services et composants de l'application, de manière à ce qu'ils n'aient pas à se connaître ou à se dépendre les uns des autres.

L'Event Bus



Mise en place d'un Event Bus

Un bus de messages

Permet de souscrire et publier des messages asynchrones en utilisant un **broker***



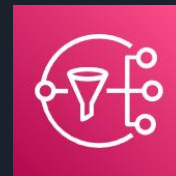
Un bus d'événements

Permet de souscrire et publier des événements asynchrones en utilisant un **système de gestion d'événements**



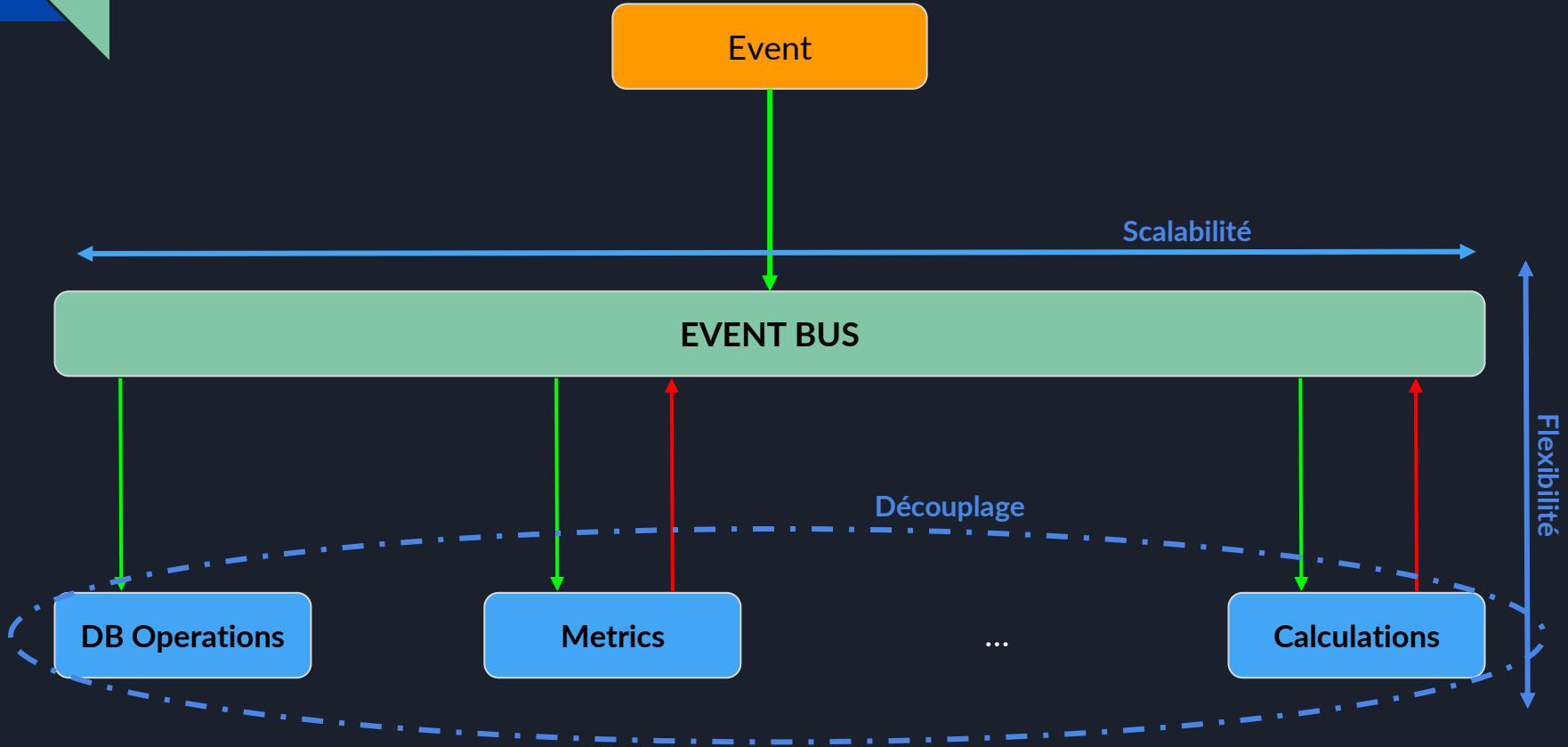
Un bus de notification


Permet de souscrire et publier des notifications asynchrones en utilisant un **système de gestion de notification**



*broker : Serveur avec lequel les clients

Avantages d'un event Bus





Exercice concevoir une architecture web en utilisant les différents outils vus en cours

Vous pouvez utiliser un outil de diagramme de cas d'utilisation comme Lucidchart ou Draw.io pour réaliser votre modélisation. Voici quelques éléments à prendre en compte lors de la création de votre diagramme :

Prenez le cas d'une application de e-commerce qui permet de acheter en ligne des produits.

Identifiez les différents services nécessaires à l'application (par exemple : service de gestion des produits, service de gestion des commandes, service de gestion des utilisateurs, etc.)

Utilisez des diagrammes de séquence et de cas d'utilisation pour modéliser les interactions entre ces services.

Choisissez un pattern d'architecture web qui convient à votre application (par exemple, API Gateway, Load Balancer, Circuit Breaker, etc.) et justifiez votre choix.

Dessinez le diagramme de votre architecture en utilisant les outils vus en cours (par exemple, draw.io).

Rédigez un rapport décrivant votre architecture et les résultats de vos tests.