

# Fonctionnement du cache - Météo

## Sommaire

[Fonctionnement et impact sur les performances sur le Broker](#)  
  [Fonctionnement du cache et de ses composants](#)  
    [Composants](#)  
      [Cache intra-processus](#)  
      [Composant de mise à jour des météo](#)  
      [Gestionnaire de stockage](#)  
  [Calcul des informations](#)  
  [Schéma résumé du fonctionnement du cache](#)

## Fonctionnement et impact sur les performances sur le Broker

### Fonctionnement du cache et de ses composants

Lors d'une demande d'affichage d'une page de météo dans l'UI de Visualisation, un appel est fait au module de la WebUI : **webui-module-element-weather**.

Dans l'optique de conserver de bonnes performances, et ce, même dans le cas où un très grand nombre de requêtes étaient envoyées, le module utilise un système de cache, permettant de conserver l'état des météo afin de ne pas les recalculer à chaque demande.

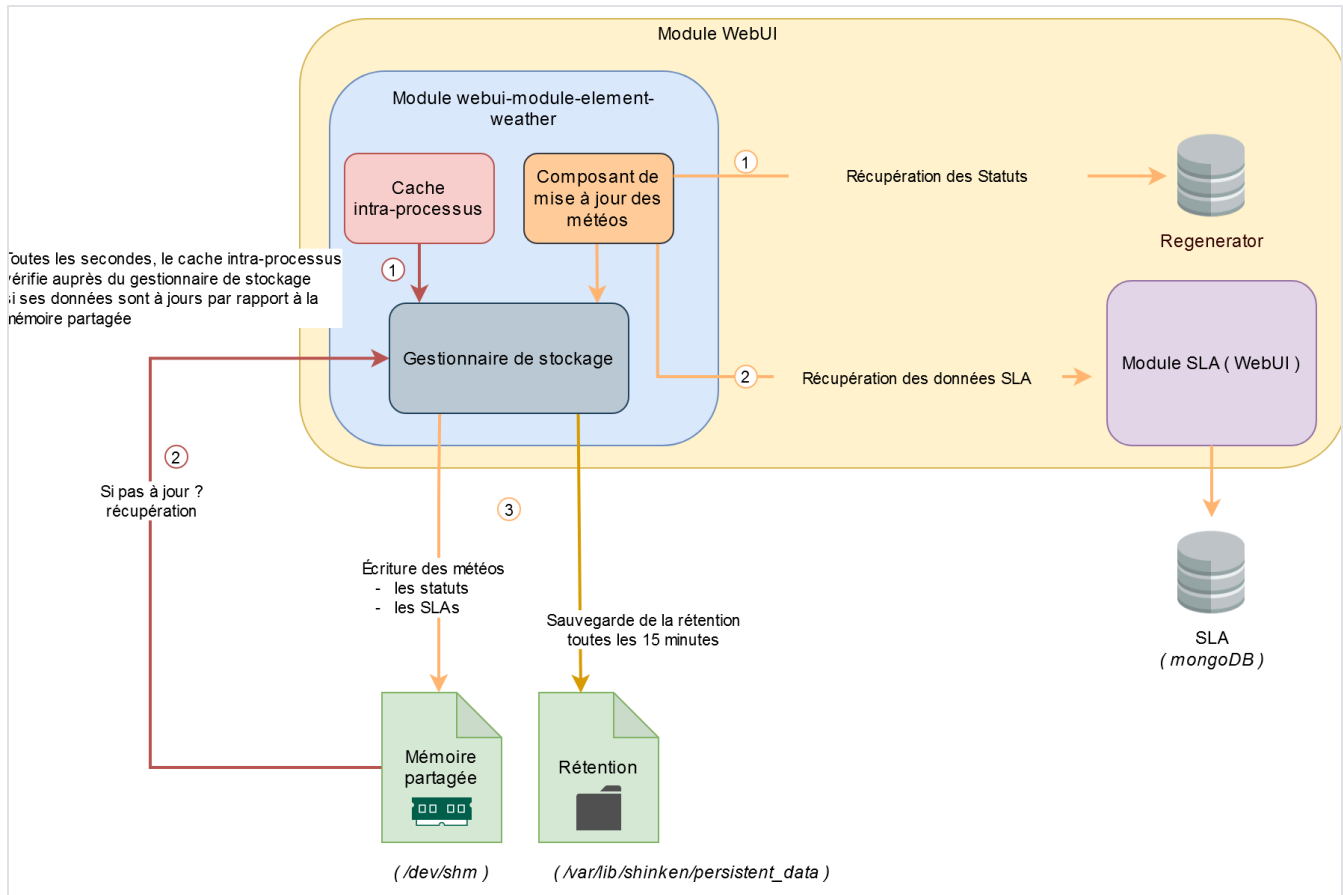
Son fonctionnement est expliqué en détail dans les chapitres suivants.

### Composants

Le fonctionnement du cache se décompose en trois composants :

- Le **cache intra-processus**, propre à chaque module, c'est celui-ci qui gère le stockage entre la demande de l'utilisateur et la mémoire partagée ;
- Le **composant de mise à jour des météo** qui, toutes les 60 secondes ( *qui peut être redéfini dans la configuration du module* ), va mettre à jour les météo dont il est le responsable ( *réparties entre tous les modules* ) ;
- Le **gestionnaire de stockage**, qui va avoir pour rôle
  - d'enregistrer et lire des météo dans la mémoire partagée ainsi que
  - de sauvegarder la rétention des données présentes dans la mémoire partagée.

C'est par lui que passent le **cache intra-processus** et le **composant de mise à jour des météo** pour sauvegarder et récupérer les météo.



## Cache intra-processus

Chaque module **webui\_module\_service\_weather** possède sa propre version des données (*directement en mémoire*) appelée **cache intra-processus**.

- Dans celle-ci sont stockées l'entièreté des météo actives.
- Toutes les secondes, ce composant va interroger le **gestionnaire de stockage** afin de savoir si les données qu'il garde en mémoire sont à jour. Si ça n'est pas le cas, alors il récupère la dernière version des météo auprès du **gestionnaire de stockage**.
- À chaque demande d'un utilisateur, le module va renvoyer la météo sauvegardée dans ce composant, permettant ainsi de n'avoir aucun calcul à effectuer

## Composant de mise à jour des météo

Afin de maintenir les météo à jour, ce composant est présent sur chaque module **webui\_module\_service\_weather**.

- Ce composant étant disponible sur chaque module, les météo sont réparties de manière équitable entre toutes ses instances.
- Chaque instance sera alors responsable de mettre à jour ses météo.
- Chaque instance va toutes les 60 secondes (*qui peut être redéfini dans la configuration du module*), recalculer les données des météo dont il est le responsable (*statut, contexte et tendance SLA*) pour les passer au **gestionnaire de stockage** afin qu'il les sauvegarde en mémoire partagée.

Exemple :

- Il y a deux WebUI ayant chacune un module **webui\_module\_service\_weather**
- Il y a un total de 5 météo
- La WebUI 1 aura à sa charge 2 météo
- La WebUI 2 aura à sa charge 3 météo

Les informations de la météo seront mises à jour comme décrit dans le chapitre [Calcul des informations](#).

## Gestionnaire de stockage

C'est le composant qui fait l'intermédiaire entre l'espace de stockage contenant les météo (*stockées dans `/dev/shm/shinken`*) et les autres composants.

Les autres composants vont alors s'adresser à lui pour chacune de ces raisons :

- Sauvegarder une nouvelle météo;
- Récupérer une météo ;
- Mettre à jour une météo existante ;
- Récupérer la date de dernière modification d'une météo dans l'espace de stockage qu'il gère ;

Les fichiers sont sauvegardés :

- Dans `/dev/shm/shinken/service_weather/daemons/broker/modules/webui/modules/webui_module_service_weather/weather_data`
- Ils sont nommés de la sorte : `shinken_WEATHER_UUID.json`
  - `WEATHER_UUID` correspond à l'identifiant de la météo.
  - Il est unique par météo.



Le dossier `/dev/shm` est un répertoire directement monté dans la mémoire vive de la machine. Ceci permet une rapidité d'écriture et de lecture bien plus importante qu'en passant par le disque classique.

L'espace en mémoire partagée offre un accès plus rapide aux météo, cependant elle est vidée à chaque redémarrage du serveur.

Pour palier ce problème, le **gestionnaire de stockage** effectue une rétention toutes les 15 minutes sur un espace de stockage persistant à l'endroit suivant : `/var/lib/shinken/persistent_data`

Cela permet qu'après un redémarrage, si le **gestionnaire de stockage** ne trouve pas une météo dans la mémoire partagée, il est capable de la récupérer depuis le disque avant de la ré-écrire en mémoire partagée.

Les fichiers sont sauvegardés :

- Dans `/var/lib/shinken/persistent_data/service_weather/daemons/broker/modules/webui/modules/webui_module_service_weather/weather_data`
- Ils sont nommés de la sorte : `shinken_WEATHER_UUID.json`
  - `WEATHER_UUID` correspond à l'identifiant de la météo.
  - Il est unique par météo.



Les fichiers enregistrés dans `/dev/shm` étant des fichiers directement écrits en mémoire, il est très vivement déconseillé de les modifier manuellement.

## Calcul des informations

Les informations affichées par la météo sont récupérées à différents endroits :

- Les informations de statut sont en mémoire dans le regenerator ( *dans la WebUI* ), ce qui implique que l'on bloque la lecture, les données de supervisions ( *ou broks* ) ne sont donc pas traitées pendant ce temps ;
- Les informations des SLA sont dans la base MongoDB ce qui implique des appels en lecture sur la base pour aller chercher les informations du jour et le calcul du pourcentage SLA ;

Il est possible si la base MongoDB est trop chargée de ne pas afficher les données SLA sur la page.

Il est aussi possible de réduire le nombre de widgets maximal des météo dans le cas où les appels aux météo sont trop long et qu'ils impactent la lecture des broks.

- ( via le paramètre `broker__module_webui__module_service_weather__default_max_widget_number`, voir la page [Configuration du module webui-module-service-weather](#) )

## Schéma résumé du fonctionnement du cache

