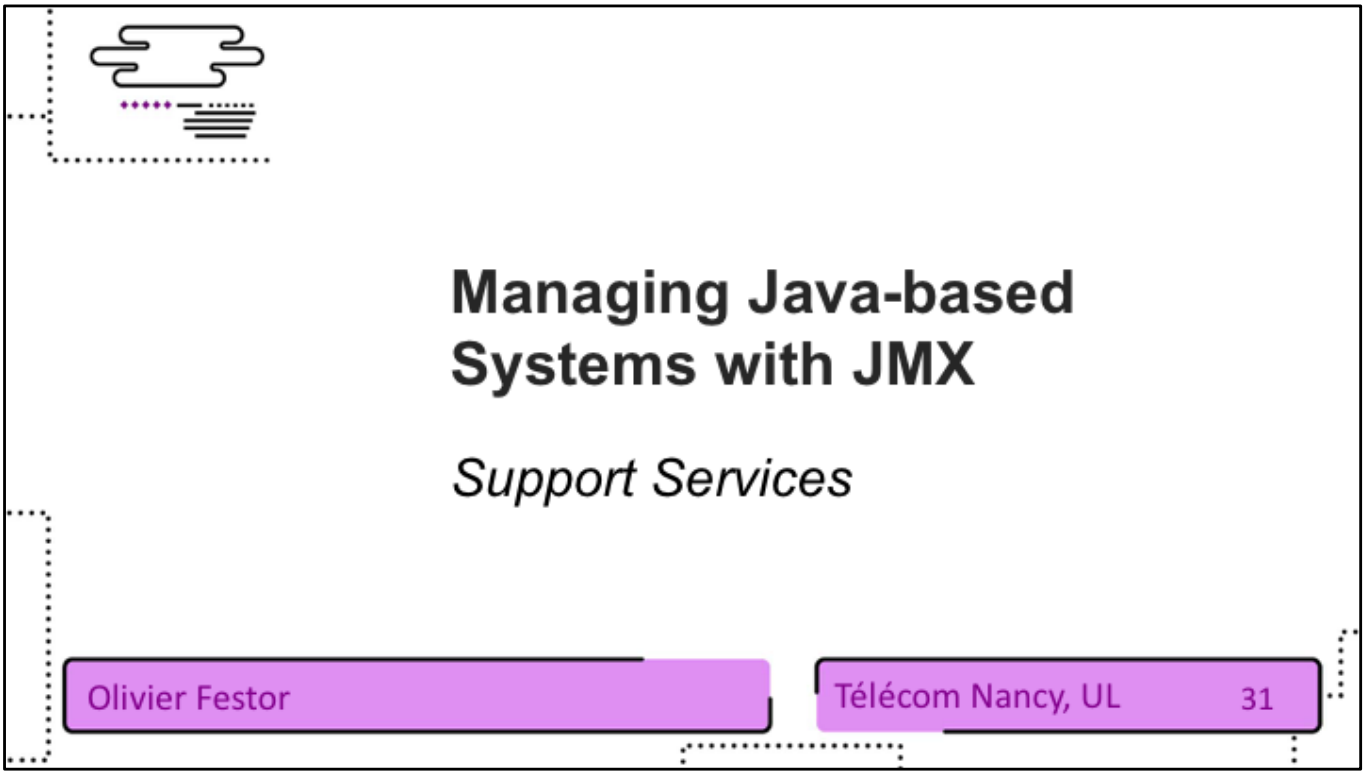
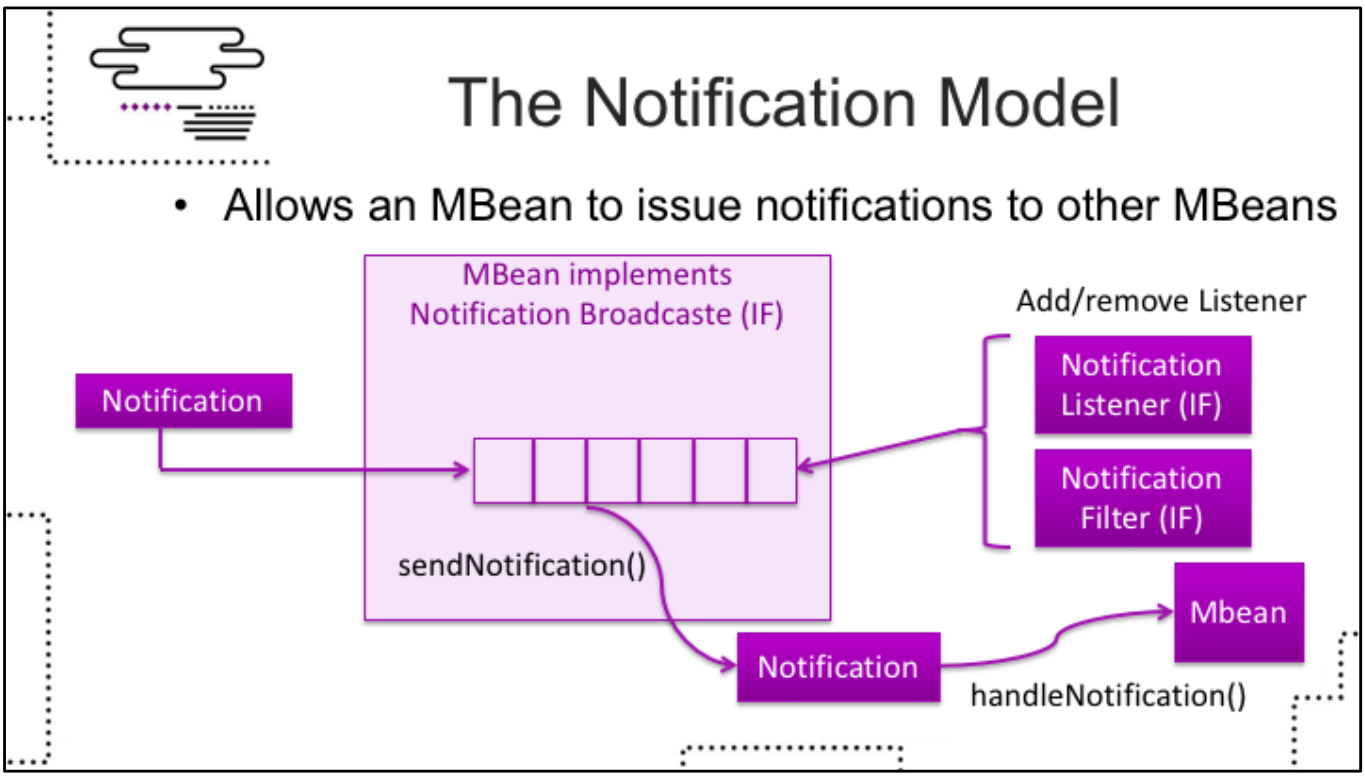


Cette leçon est consacrée aux services complémentaires offerts par la libraire JMX au développeurs de solutions de supervision. 5 services standards sont fournis, certains utilisant le modèle de notifications de l'approche.



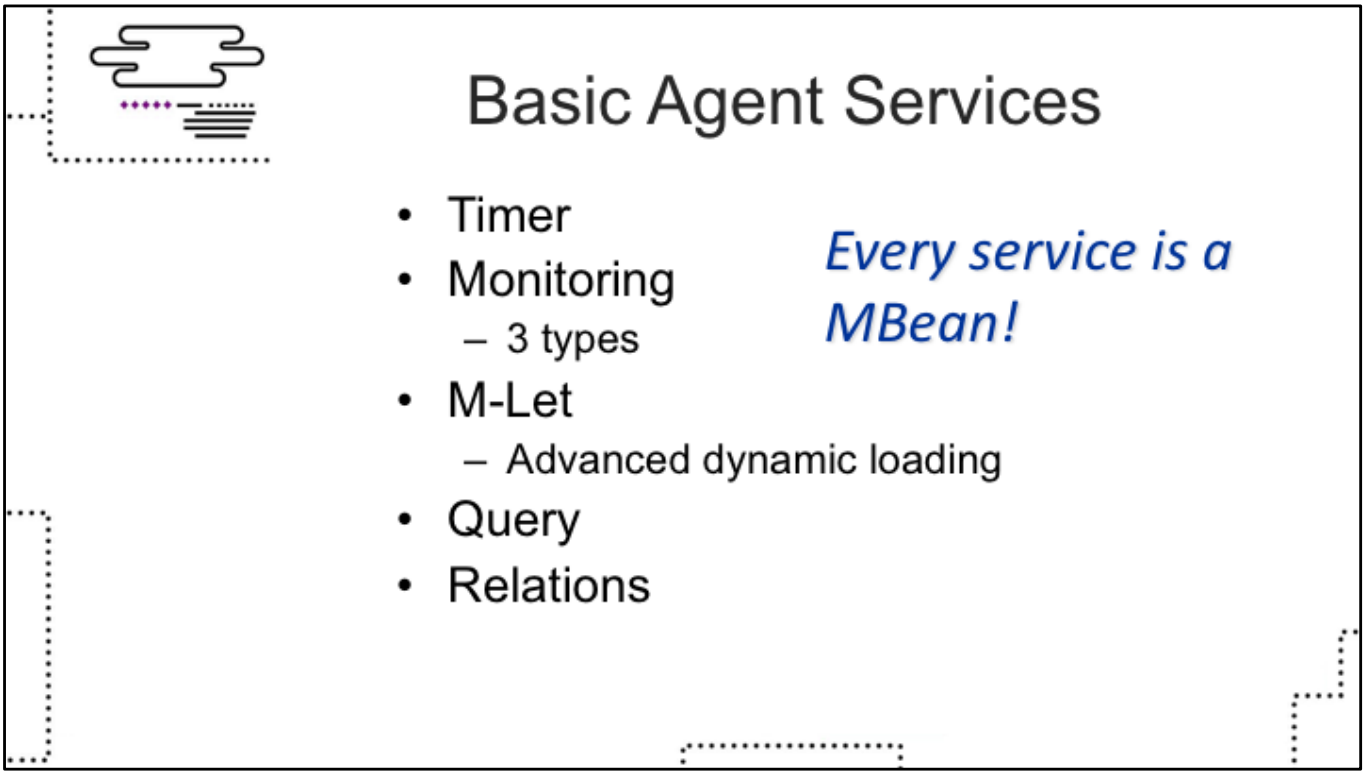
Le modèle de notifications de JMX est celui du patron de conception « Observer » bien connu dans la littérature. Les développeurs d'interfaces en Java ne seront pas dépaysés.

On y retrouve les acteurs suivants :

- Notification (la classe peut être sous-classée). Celle-ci est constituée de :
  - un type: une chaîne de caractères qui définit le type de la notification par exemple fr.loria.pc.alarm,
  - un numéro de séquence (numéro incrémenté et géré par la source de la notification),
  - une estampille,
  - un message (une chaîne de caractères qui décrit la cause),
  - UserData : toute donnée complémentaire nécessaire au traitement de la notification (un Java « Object »),
- NotificationListener (IF) qui réceptionne les notifications et les traite au travers d'une méthode *handlENotification,*
- NotificationFilter (IF) qui, associé à une souscription, permet de savoir si une notification passe un filtre avant émission. Le filtre est évalué par la méthode notificationEnabled,
- NotificationBroadcaster (IF) qui définit l'interface nécessaire à la gestion de l'émission de notifications. Celle-ci comprend les méthodes suivantes :
  - getNotificationInfo qui énumère tous les types de notifications que la source peut émettre,

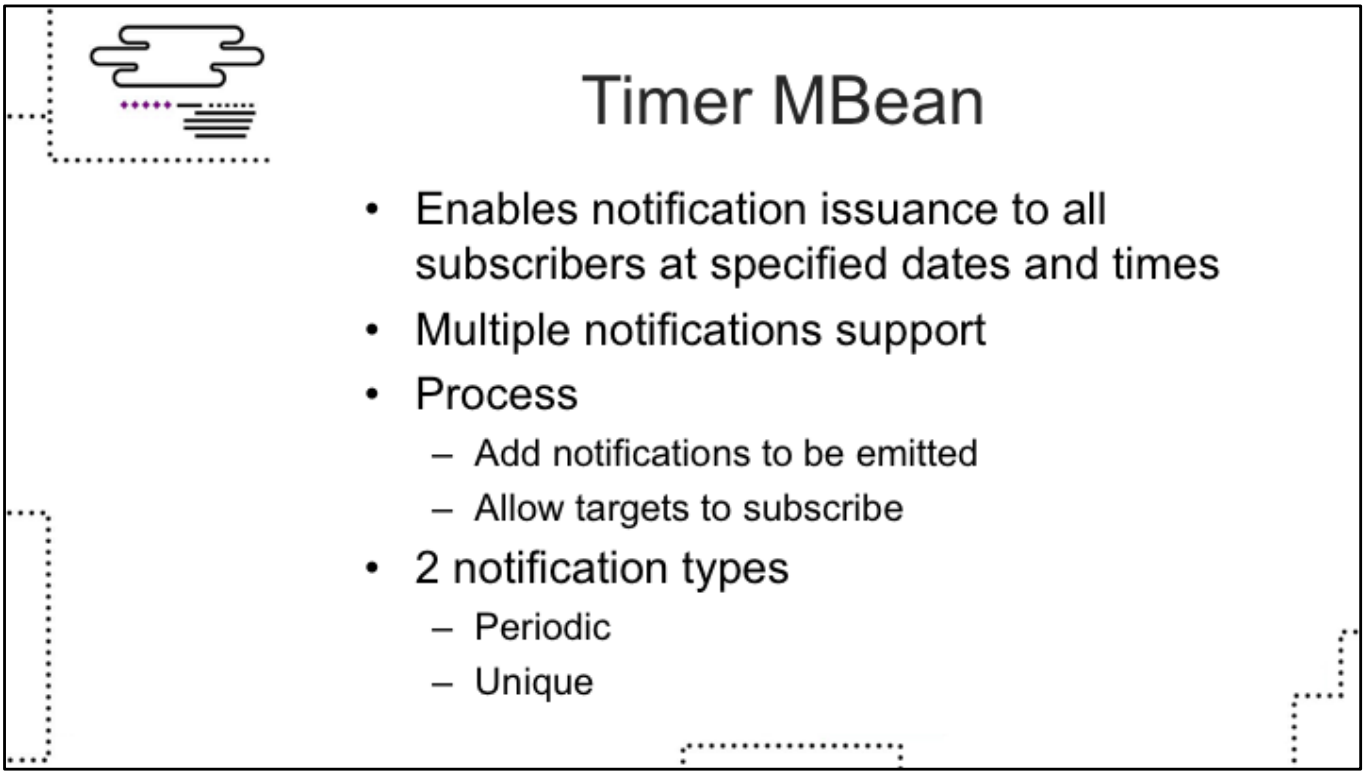
- add/RemoveNotificationListener qui gèrent les abonnements/désabonnements aux notifications de la source,
- sendNotification qui gère l'envoi effectif.

Tout objet de supervision qui veut émettre et gérer directement des notifications doit implémenter un NotificationBroadcaster.



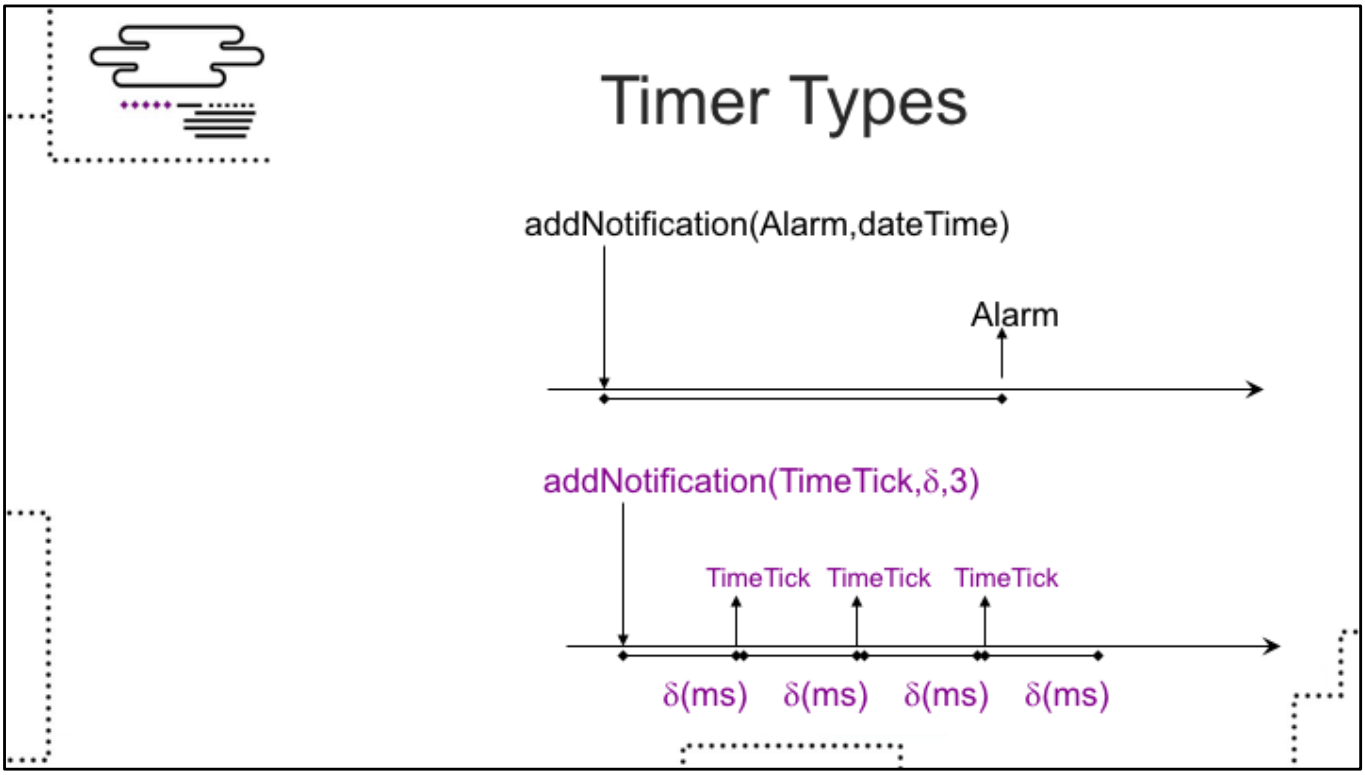
5 services composent les services de base fournis aux développeurs d'objets et d'agents de supervision, en plus de la gestion du cycle de vie des objets bien sûr. Les deux premiers (*Timer* et *Monitoring*) utilisent le service de notifications de JMX basé sur le modèle décrit précédemment. Il faut y ajouter un service de chargement à distance de Mbeans, un service de requêtes et un service de gestion de relations entre Mbeans.

A l'instar des connecteurs et des adaptateurs, les services sont eux-mêmes des Mbeans et sont invocables au travers de l'interface de supervision.



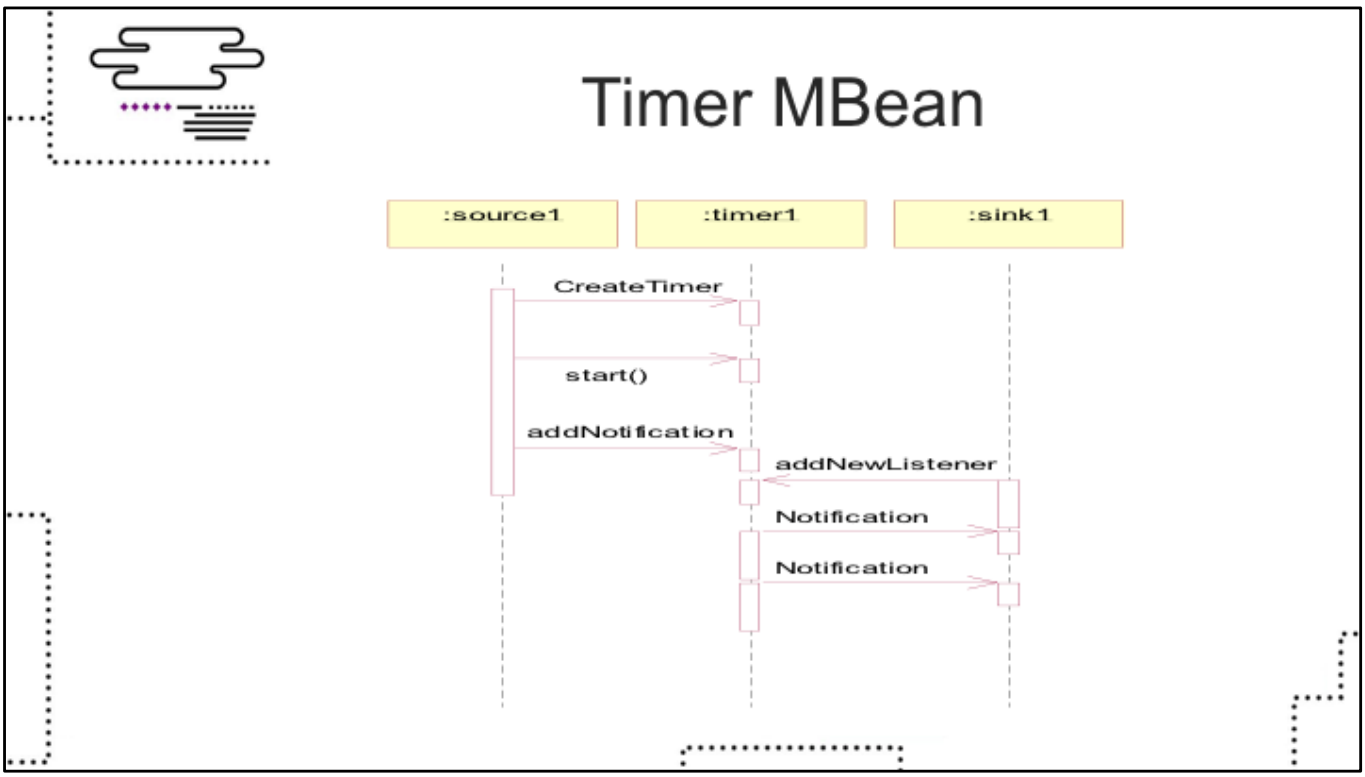
Le service de timer gère une horloge programmable assortie d'un réveil pour les objets de supervision.
Il permet à tout objet géré de lui demander des notifications temporelles. On peut y définir des nouvelles notifications et tout objet géré peut y souscrire. Deux modes sont possibles pour les notification ;: périodique ou apériodique (une unique notification émise à un instant déterminé).
Un système de gestion du cycle de vie du *timer* est fourni. Il comprend les méthodes :

- start(),stop() pour démarrer, respectivement stopper un timer,
- isActive() qui permet de connaître l'état d'un timer,
- sendPastNotifications(bool) permet à un souscripteur de demander que les notifications qui ont pu être émises avant la souscription soient transmises au souscripteur, sinon elles seront ignorées.

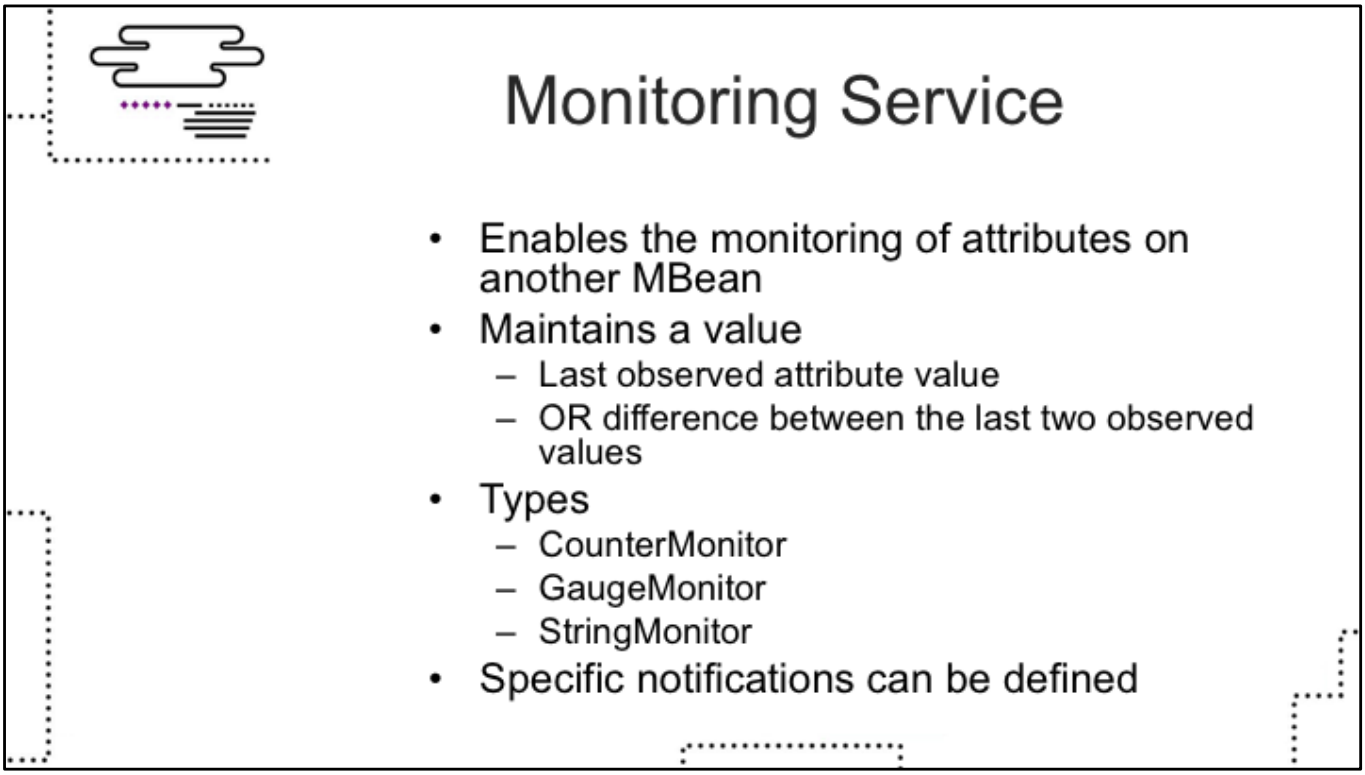


Les deux types de fonctionnement du service de timer sont :

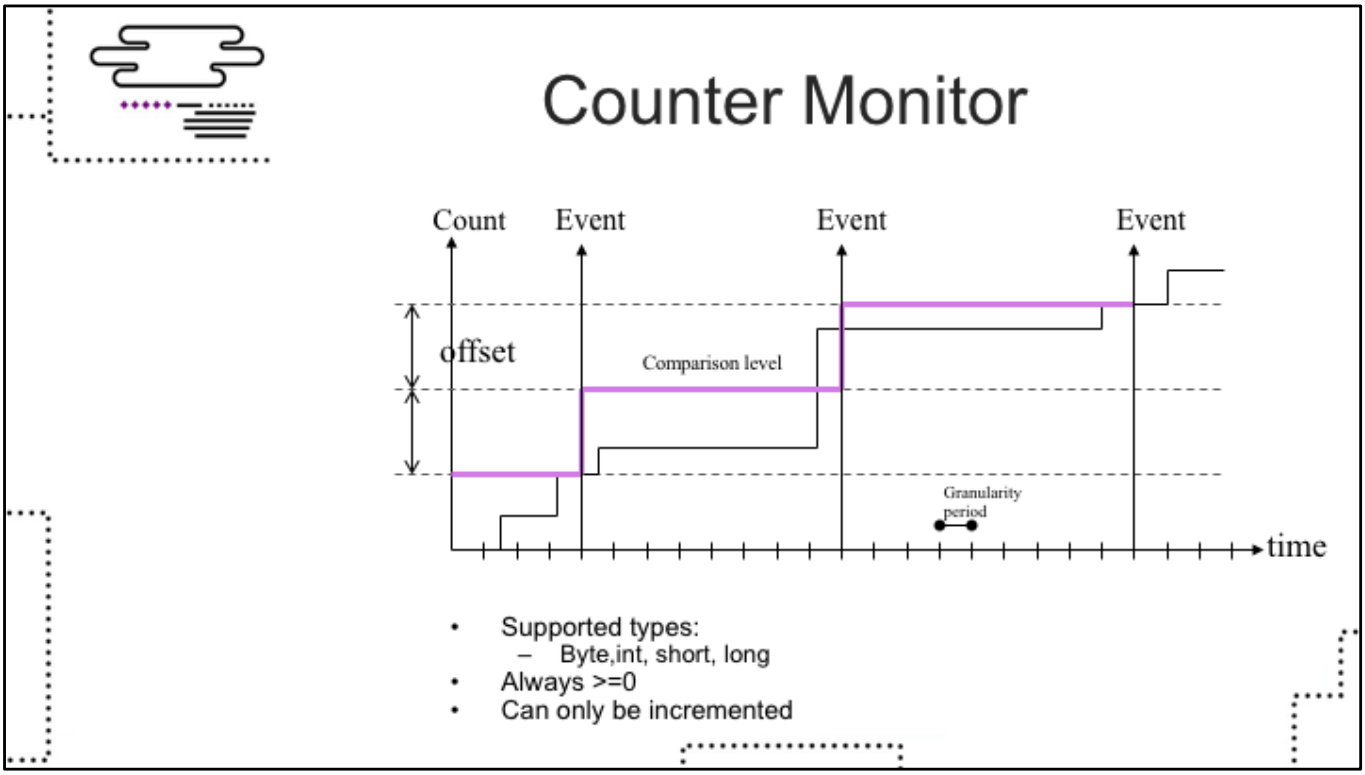
- Le mode apériodique qui engendre un déclenchement unique à une date et heure précises,
- Le mode périodique. Dans celui-ci, une notification est émise à échéance de chaque période. Les périodes sont données aux timers lors de leur paramétrage. Elle sont exprimées en millisecondes.



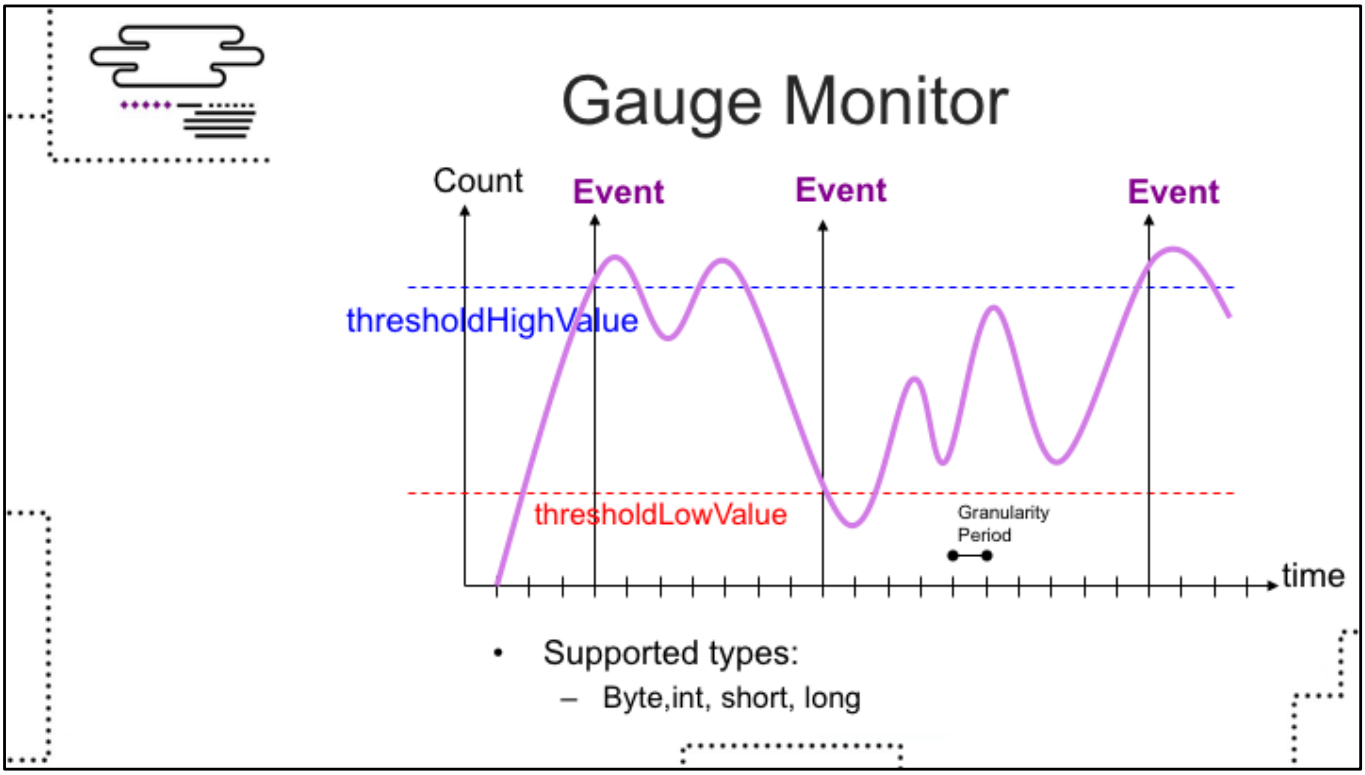
Voici le séquencement des appels de méthodes entre les différents acteurs exploitant un service de timer. On y trouve la source qui créé le timer, le paramètre et le démarre. Le second acteur est le puit (sink) qui va recevoir des notifications. Pour cela, il s'abonne au service avec la méthodes addNewListener. Une fois cet abonnement réalisé, il va recevoir les notifications du service de timer.



JMX offre un service de monitorage. Celui-ci permet à une application de supervision de déléguer la surveillance d'un attribut d'un objet de supervision à l'agent. Le service est totalement paramétrable tant sur la méthode d'observation (fréquence, forme et nature des données collectées) que sur le comportement (émission de notifications sur changement, sur seuil, par exemple). Trois services implémentent ce service de monitorage et fournissent des comportements prédéfinis : un service de surveillance de compteurs, un service de surveillance de jauges et un service de surveillance de chaînes de caractères.

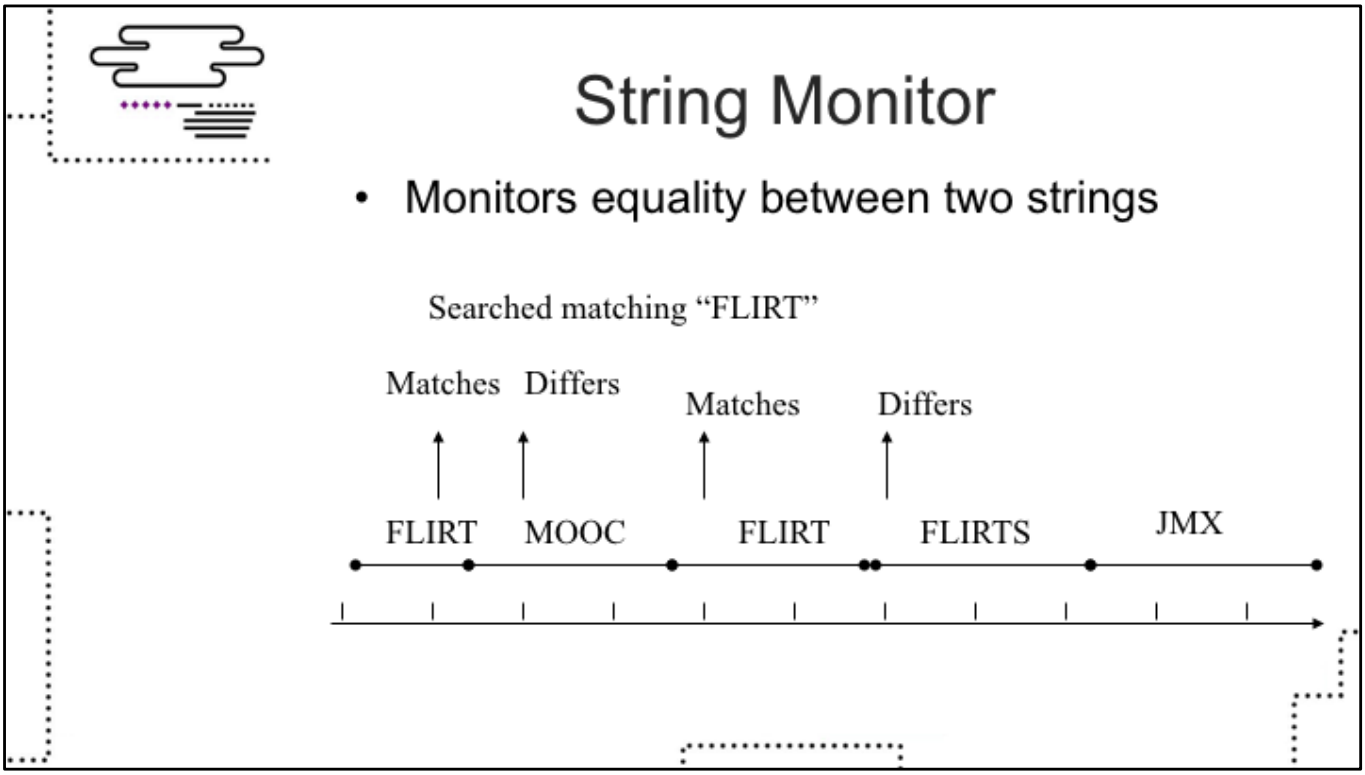


Le premier type de service de monitorage permet de surveiller l'état d'un compteur.
Le compteur a la caractéristique suivante : il ne peut que croître dans le temps jusqu'à atteindre un sommet qui le remettra éventuellement à son minimum, 0.
Surveiller un compteur revient donc à mettre en place des sentinelles sur des passages de seuil (*offset* sur notre illustration).



Le deuxième type de moniteur fourni dans JMX permet de surveiller des jauges à savoir des entiers qui oscillent entre une valeur minimale et maximale comme on peut le voir sur le schéma ici. Les attributs supervisables sont de type octet (8 bits), entier (32 bits), court (sur 16 octets) et long (64 bits soit 8 octets).
Deux paramètres sont définis pour ce service, une valeur de seuil bas (*thresholdLowValue)* et seuil haut (*thresholdHighValue).*
Les déclenchements de notifications sont réalisés dans les conditions suivantes :

- Lorsque la jauge croît et que sa valeur dépasse le seuil haut (cas du premier et du dernier événement),
- Lorsque la jauge diminue et que sa valeur passe le seuil bas.

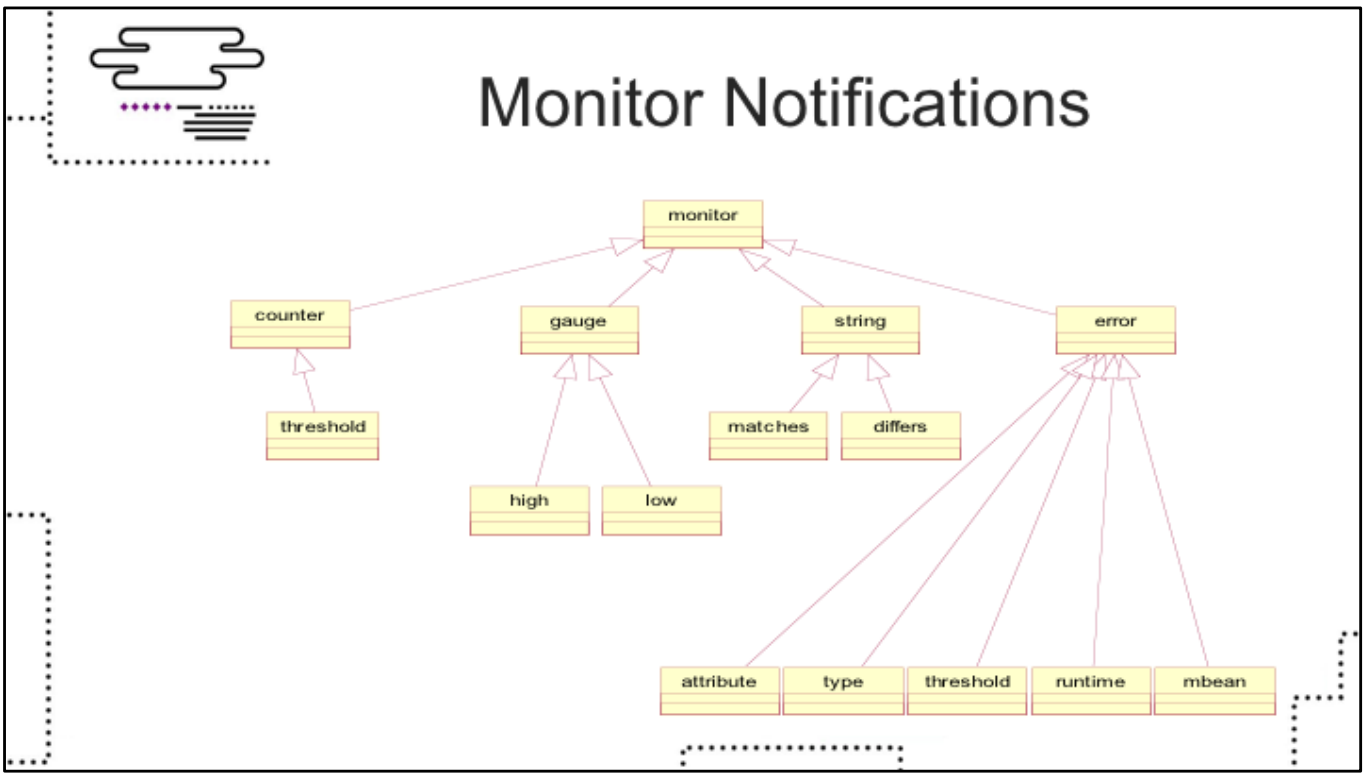


Le troisième et dernier type de service de monitorage est celui des chaînes de caractères. Comme son nom l'indique, il surveille un attribut d'objet de supervision de type String et le compare à une référence qui lui a été donnée lors de l'initialisation du service.

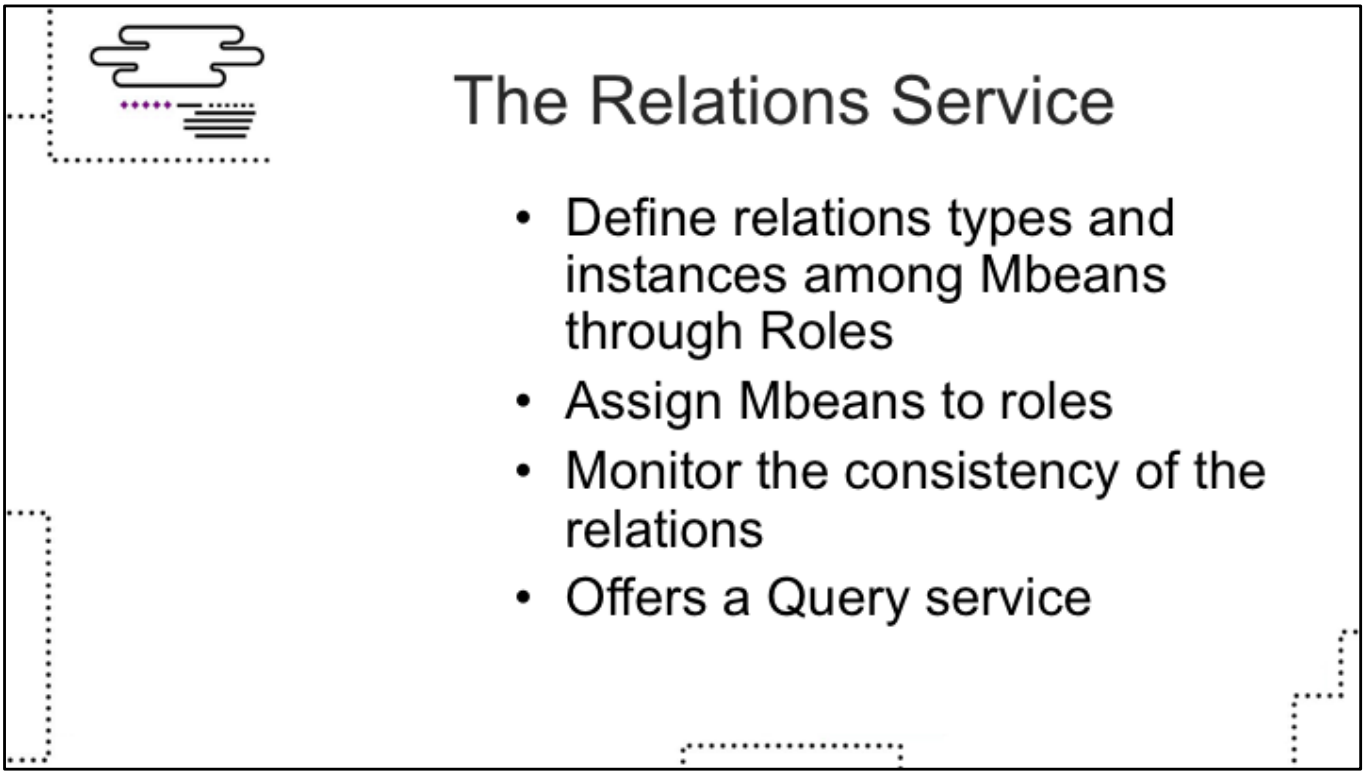
Deux notifications peuvent être émises par ce service :

- Matches : l'attribut surveillé est identique à la valeur paramètre,
- Differs : la valeur de l'attribut surveillé est différente.

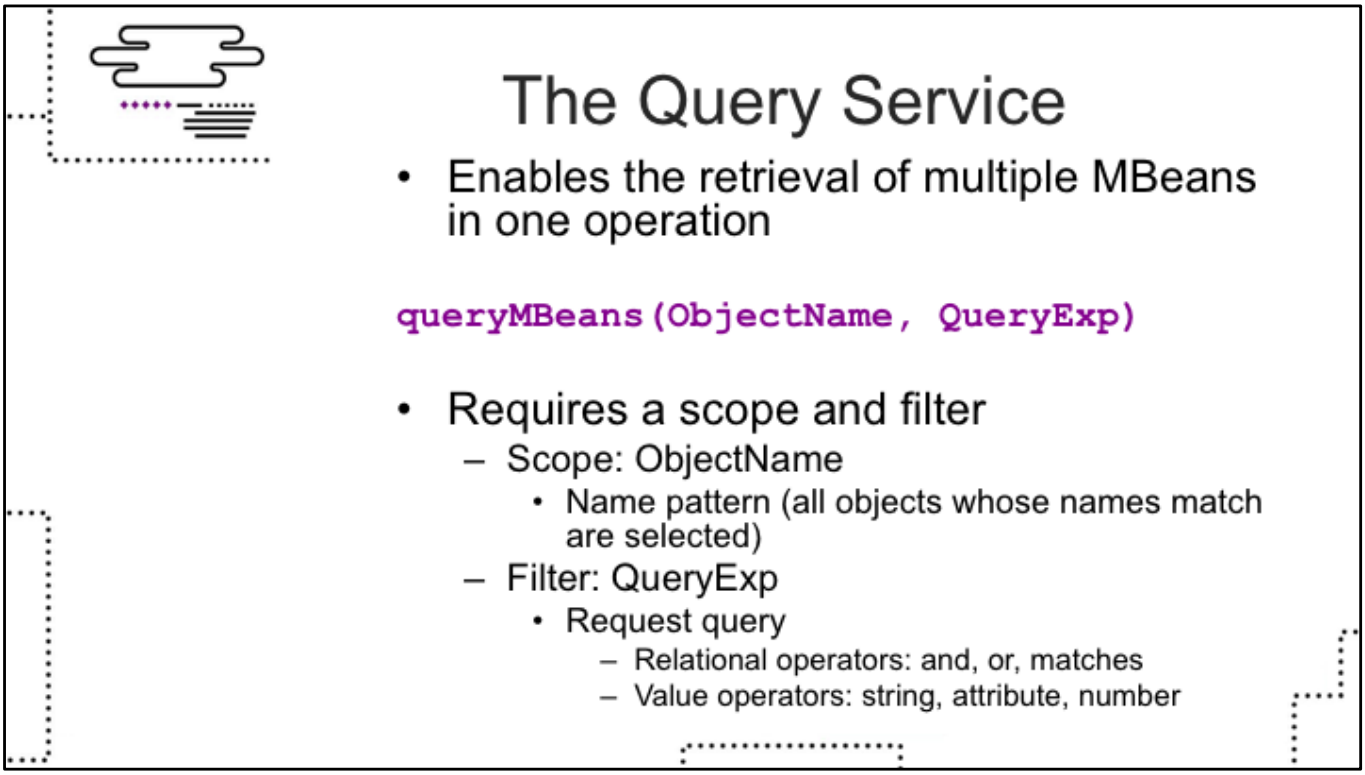
Les notifications ne sont émises que sur changement d'observation. Par exemple sur les périodes 3 et 4, deux observations sont réalisées. Toutes deux diffèrent mais seule la première donne lieu à émission d'une notification. Il en est de même pour "Matches" sur les périodes 5 & 6.



En résumé, le service de monitorage dispose d'une hiérarchie de notifications comme nous l'avons vu dans la présentation des différentes types. Il comprend également 5 notifications d'erreurs qui permettent de traiter tous les cas d'impossibilité de monitorer dans de bonnes conditions un attribut (il n'existe pas, son type est incompatible, les seuils sont incompatibles, l'objet supervisé n'existe pas …).

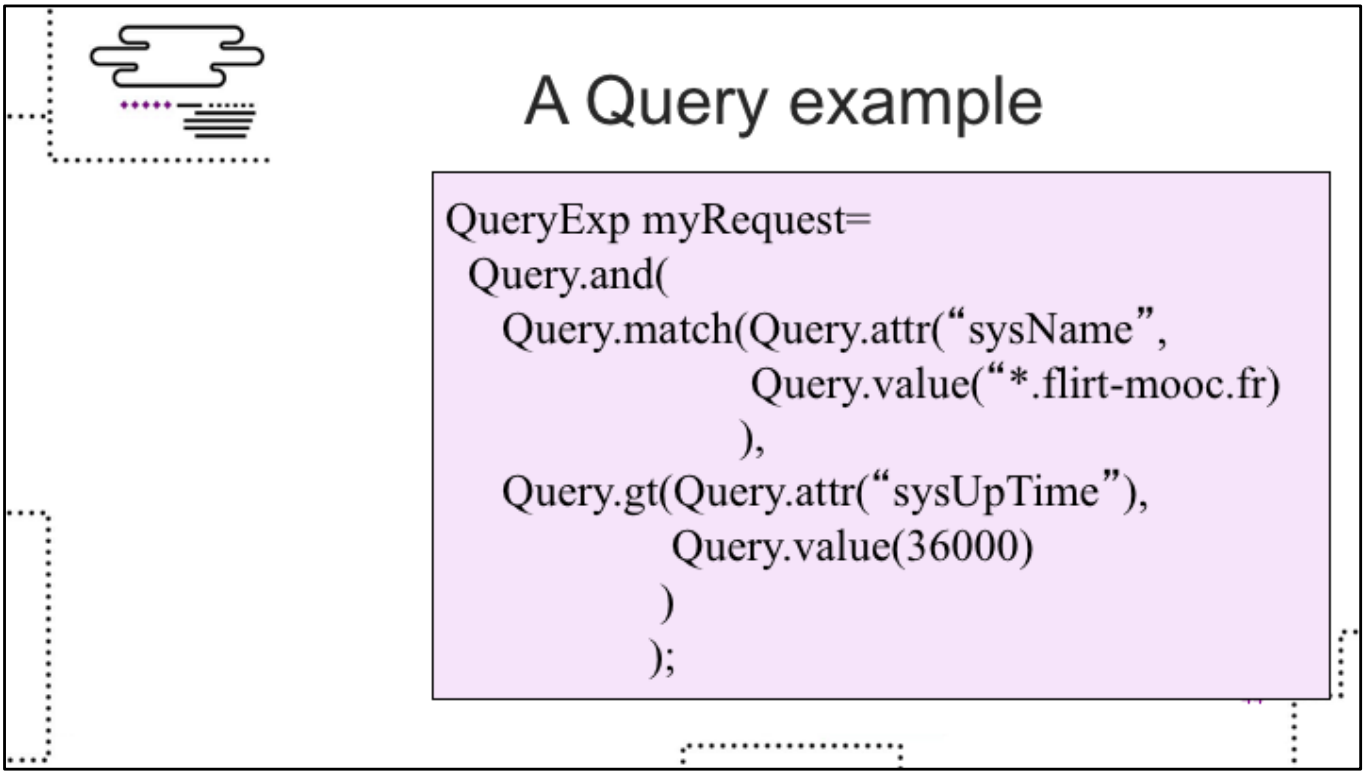


Le service de relations définit des liens entre différentes instances de Mbeans au travers d'un système de rôles. Les objets de supervision sont associés à des rôles puis le service assure la supervision de la consistance de ces relations et offre un service d'interrogation des relations via un langage de requêtes dédié.

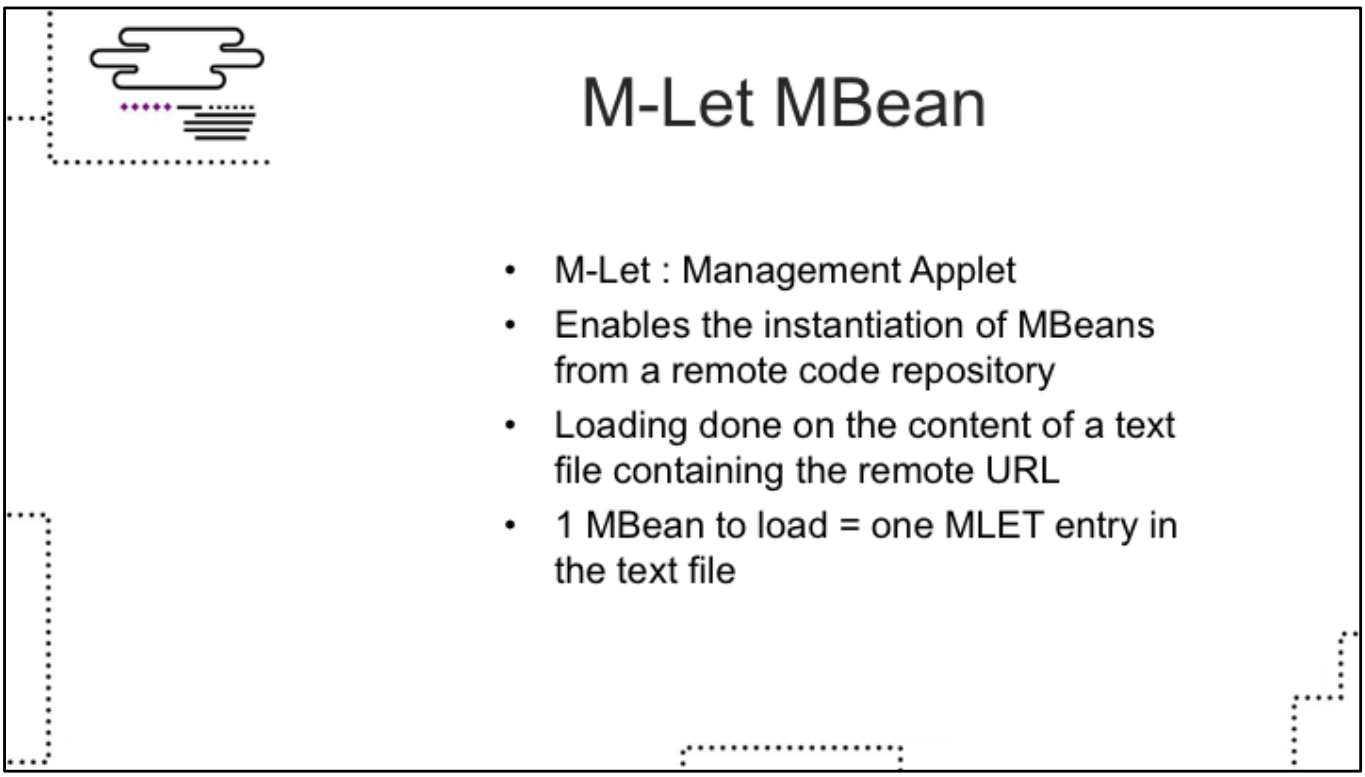


Le service de requêtes permet de sélectionner un ensemble d'objets de supervision qui ont un pattern de nom spécifique et qui répondent à des critères exprimés suivant une expression logique. Les deux paramètres de ce service sont :

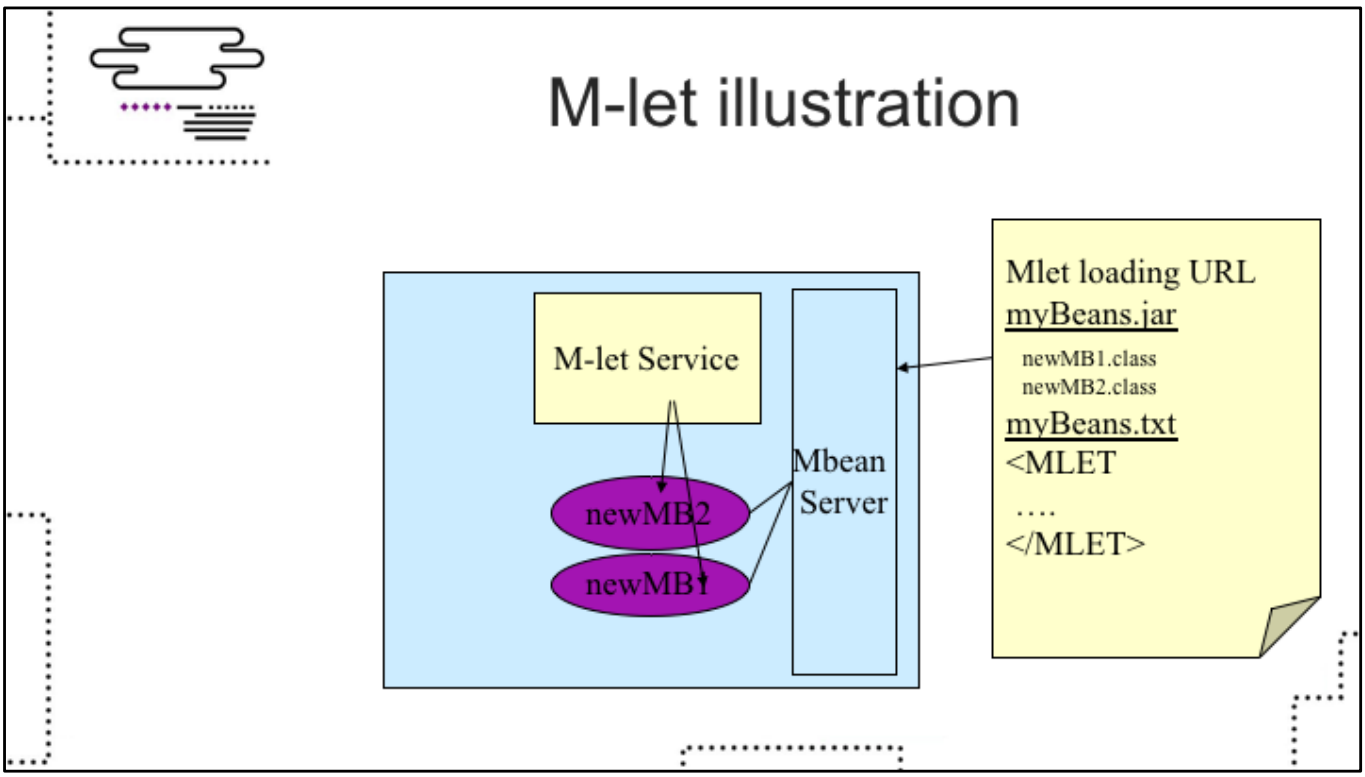
- Le paramètre de portée qui sélectionne tous les objets dans un sous-arbre enraciné par le nom.
- Le filtre qui exprime les conrtaintes attendues sur les objets à remonter.



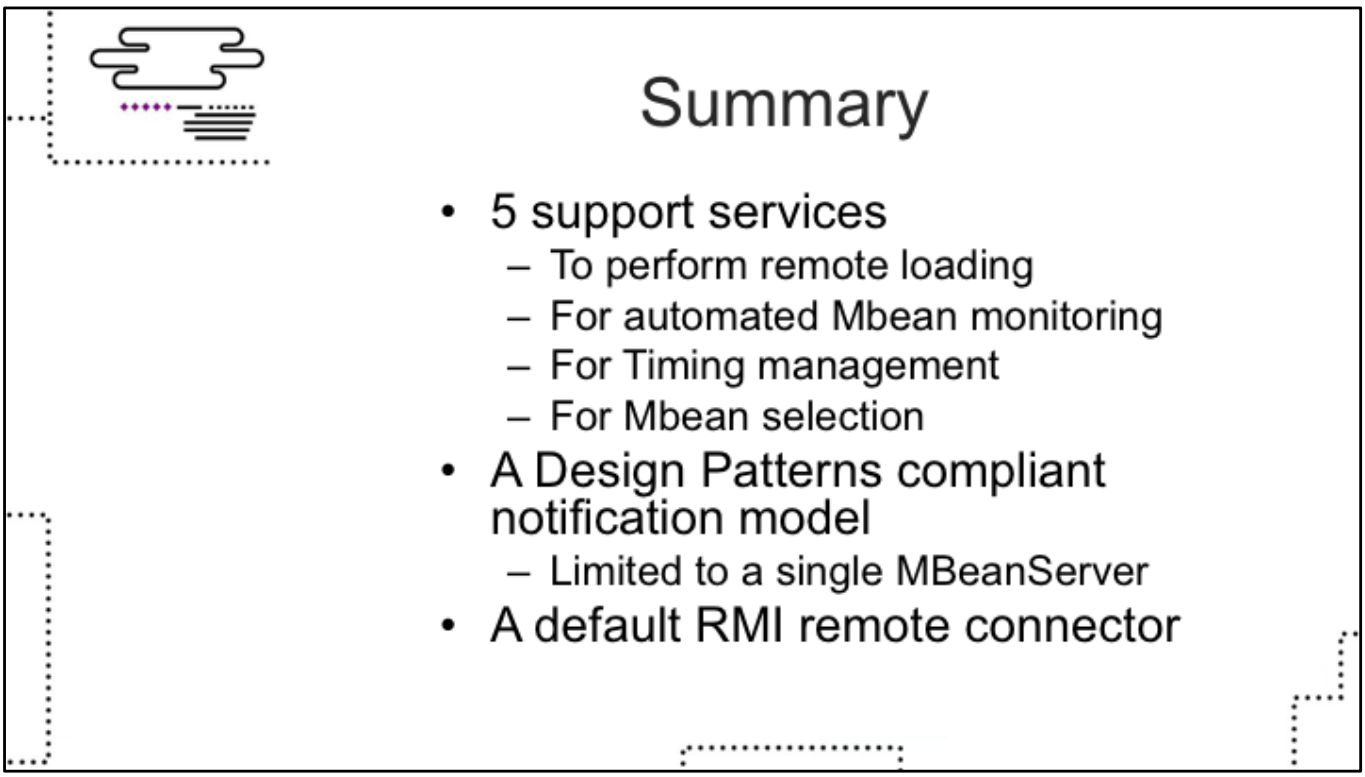
Voici un exemple simple d'une expression. On recherche toutes les instances d'objets de supervision dont le nom contient le suffixe flirt-mooc.fr et qui sont activés depuis plus de 10 heures (le sysUpTime est ici supposé être exprimé en secondes).



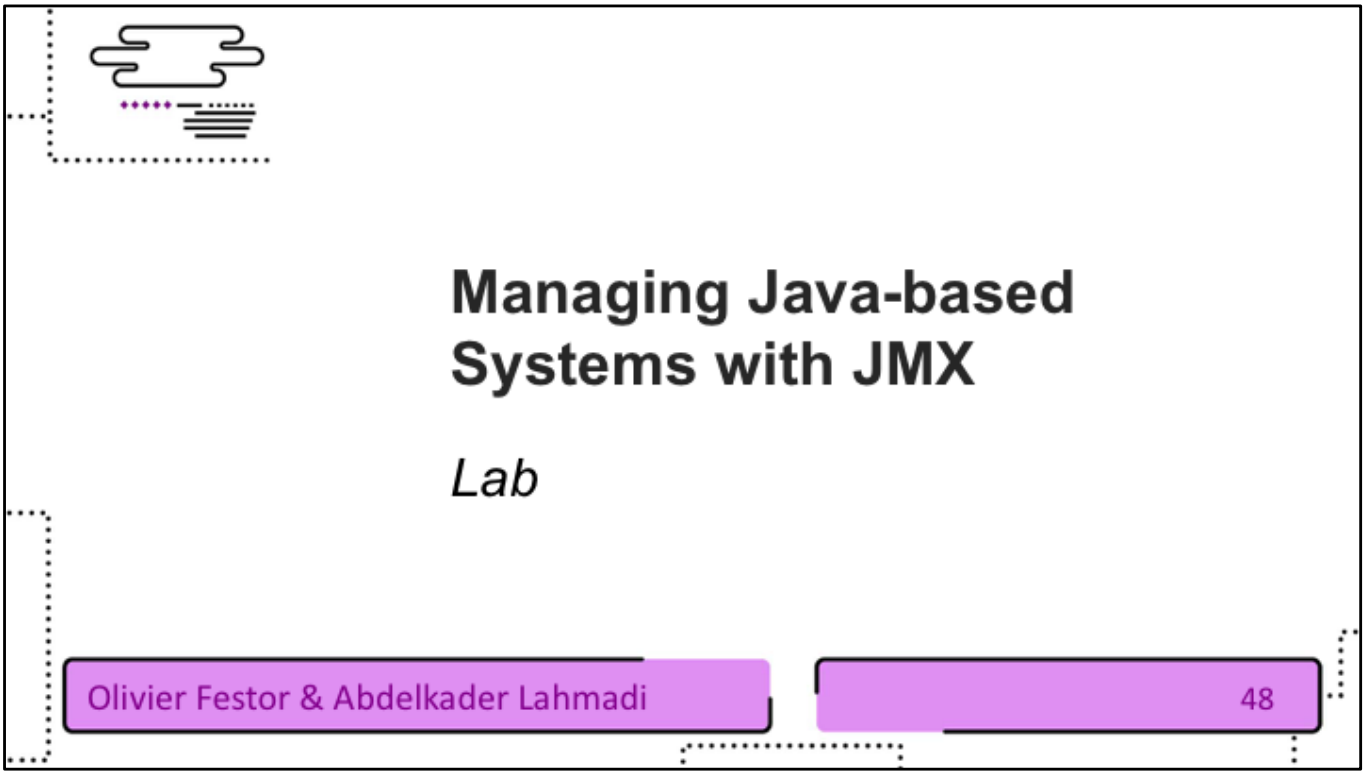
M-let est le service qui permet de charger à distances des Mbeans dans un agent qui va assurer la liaison avec les objets de supervision et les objets applicatifs présents.



Les objets de supervision sont fournis dans une archive (fichier .jar). Les paramètres de chargement ainsi que les instructions d'instanciation et de couplage sont donnés dans des méta-données associées (fichier .txt) et *snipplet* (extrait de code XML)
<MLET>



Comme nous l'avons vu dans cette séquence, les services supports de JMX sont nombreux et riches. Ils permettent de charger à distance des objets gérés, d'en monitorer des attributs, de les sélectionner grâce à un service de requêtes.
Le service offre également une gestion complète de timers et un système de notifications puissant.



L'objectif du lab est de mettre en œuvre sur un serveur spécifique, l'ensemble des concepts vus dans toutes les séquences du cours.