Les droits

Table des matières

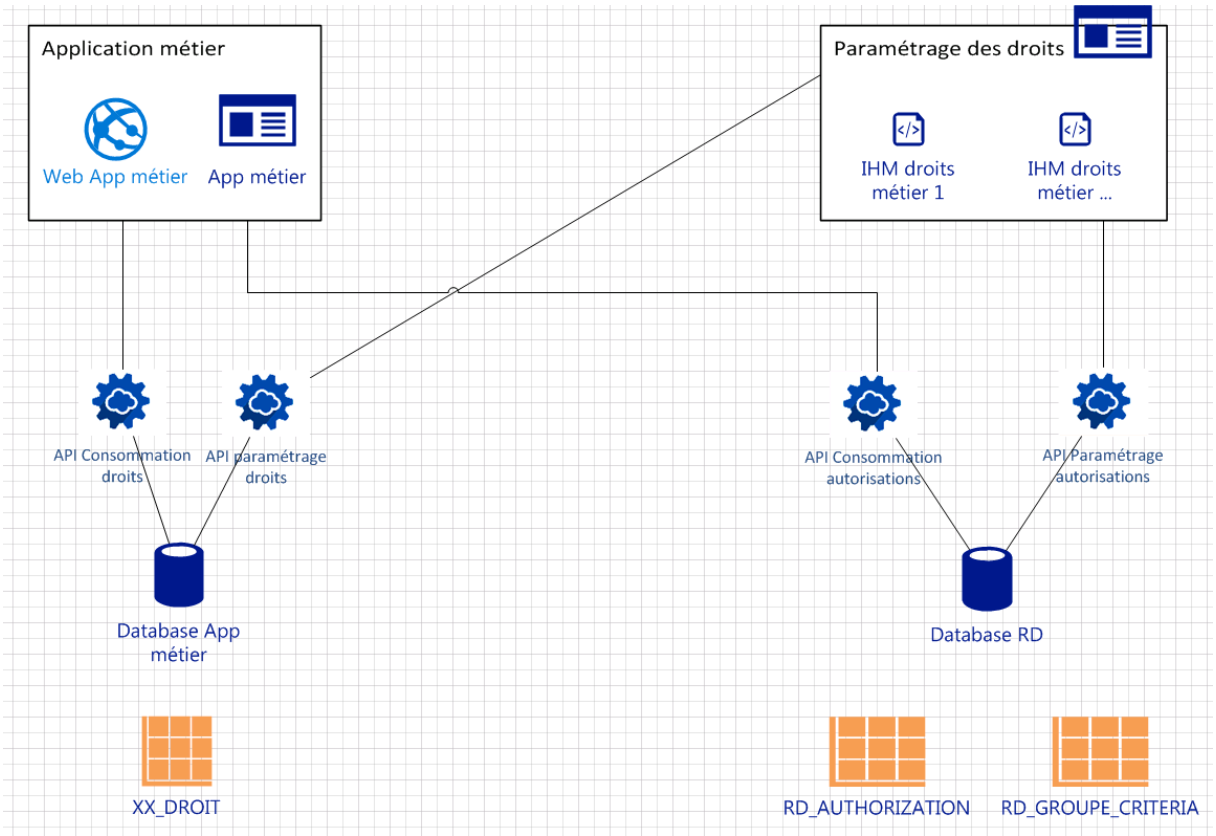
Général

Les droits utilisent deux principes : les **droits** et les **autorisations**.

Un **droit** va représenter le « **quoi** », qui va dépendre du métier auquel il est rattaché. Tandis que **l'autorisation** représente plutôt le « **qui** », en fonction des employés, du service, etc. Elle donne également le **niveau** du droit affecté : lecture, écriture…

Ex : si l'on a un droit sur une catégorie d'article, on pourrait avoir une autorisation pour un employé X et une seconde pour un employé Y qui leur donne le droit d'écriture.

Architecture



Il faut séparer le schéma en deux pour comprendre au mieux le fonctionnement.

Il y a une première partie qui permet de paramétrer les droits et autorisations (sur la droite). Cela se fait par une IHM (par métier) qui se trouve dans la solution des droits en R&D. On peut y ajouter, modifier et supprimer des droits et/ou leur(s) autorisation(s).

La seconde partie (donc celle de gauche) est située au niveau de l'application métier. C'est une application quelconque qui a besoin d'utiliser des droits et qui va utiliser deux API Rest pour aller les chercher en base.

Web services

La composition des web services est la suivante :

- Consommation des droits
    - GetDroitsById(int idDroit) : IEnumerable<IDroit>
    - GetDroitsByDesignation(string designation) : IEnumerable<IDroit>

- Consommation des autorisations
    - GetAuthorization(int idDroit, EApplication application, Employe employe, bool withExplicitExclusion) : EEAuthorizationLevel
    - GetAdministrators(IEnumerable<(int idDroit, EApplication application)> droits) : IEnumerable<Employe>

- Paramétrage des droits
    - SetDroit(IDroit droit)
    - DeleteDroit(IDroit droit)
    - UpdateDroit(IDroit droit)

- Paramétrage des autorisations
    - SetAuthorization(GroupeCritere critere, IDroit droit, EEAuthorizationLevel level)
    - DuplicateAuthorizations(IDroit source, IDroit destination)

Tables

1. **RD_DROIT**

   Elle permet de définir pour un métier les droits associés à leur(s) paramètre(s). C'est pour cette raison que chacun va posséder des colonnes différentes liées à leur propre paramétrage des droits. Elle doit avoir obligatoirement un IDT_DROIT (clé primaire).

   Ex : La logistique pourrait avoir une colonne IDT_COMMANDE pour avoir des droits spécifiques à chaque commande, alors qu'en référentiel produit, une colonne IDT_ARTICLE serait utilisé

2. **RD_GROUPE_CRITERIA**

   Cette table définit qui a les droits, cela peut aller de l'activité jusqu'à l'employé, en passant par le service, etc…

   Ex : Pour le référentiel produit, il n'y a que le service informatique qui peut accéder aux articles d'une certaine catégorie.

   Il y a un système de file intégré aux droits. Les colonnes « IDT_GROUPE » et « IDT_FILE » permettent de différencier deux façons différentes de gérer les droits des files.

La première est renseignée lorsque l'on souhaite laisser l'accès à une file sur un critère de cette table, donc à un service, à un employé etc…

La seconde n'est pas nulle quand on veut gérer les droits de la file via une autorisation qui va définir le niveau du droit.

3. **RD_AUTHORIZATION**

L'autorisation, quant à elle, fait la liaison entre le droit et le groupe de critères. Elle permet de savoir le niveau de l'autorisation (lecture, écriture, etc.) et l'application concernée.

Ex : Le droit 6 avec les critères 42 est accessible uniquement en lecture pour l'application Logistique

4. **Autres tables**

Ici vont être répertoriées les tables utilisées par les droits mais qui ne sont pas forcément impactantes ou qui ne sont pas utilisées directement par la solution des droits.

Il y a notamment les tables utilisées pour les traductions, à savoir

- RD_LANGUE_REGION
- RD_PAYS
- RD_LANGUE

Ces tables sont utilisées d'une part la table « RD_GROUPE » qui possède une désignation qui doit être traduite, d'autre part avec la table « RD_TYPE_FILE » qui contient également une désignation et qui sert à définit le type de file.

Implémentation

Pour gérer les droits dans sa solution, il y a plusieurs étapes à respecter.

Tout d'abord, il faut aller dans la solution se situant dans « *ERP/RH/Parametres/Version/V3* »

Il s'agira ensuite de récupérer la solution et la lancer.

Une fois fait, **créez un projet** « RH.IHM.NomDuMetier ». Dans celle-ci, il faudra créer vos propres UserControl, et ceux-ci devant **hériter** de « IDroitControl » et implémenter les interfaces requises.

C'est alors à vous de développer les interfaces nécessaires à la gestion de vos droits, donc tout ce qui est de la création de droits, la modification etc…

Point important, il ne faut **aucune dépendance** vers les projets de votre application. Donc pas de services ou de références pointant vers la solution métier concernée par les droits. Tout devra passer par des API Rest.

Ce projet devra également contenir deux classes :

- La première est un DroitProvider, qui hérite de IDroitProvider, et sur laquelle vous allez implémenter les interfaces nécessaires. Une fois fait, ce provider vous servira à aller

chercher vos droits en utilisant les méthodes « GetDroitsByDesignation() » ou « GetDroitsByIds() »

- La seconde est votre classe de droits métiers qui devra hériter de l'interface « IDroit » en implémentant ce qui a besoin de l'être, utilisant le DroitProvider créé ci-dessus.

Pour terminer, lorsque vous avez besoin d'aller chercher les autorisations, il faudra passer par l'API Rest qui permet l'accès à ces données. Vous ne devez utiliser en aucun cas l'interface IServiceDroits qui est devenu obsolète.