

# 1 Administration des projets

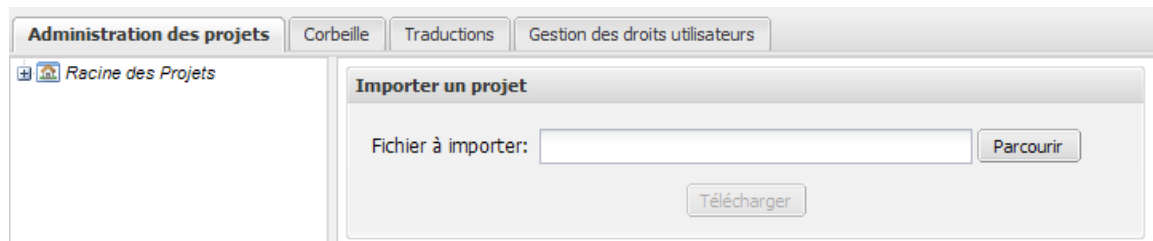
Une fois dans la catégorie « **Administration des projets** », l'utilisateur peut créer, modifier ou supprimer différents types d'éléments, qui permettent d'organiser les analyses.

Ces types d'éléments disponibles sont :

- le Domaine
- le Projet
- le Sous-projet
- l'Entité applicative

L'« administration des projets » présente une arborescence qui regroupe l'intégralité des organisations de projets.

La « racine des projets » est le point d'entrée de cette arborescence. La sélectionner permet d'importer un projet.



**Figure 1. Importer un projet**

Chaque organisation de projet commence par un élément de type Domaine ou Projet.

Créer un de ces éléments se fait en sélectionnant la « racine des projets ». La partie centrale contient alors deux boutons qui permettent de créer un nouveau domaine ou un nouveau projet.

La modification d'un élément existant se fait en sélectionnant l'élément dans l'arborescence.

## 1.1 Créer un nouvel élément

La création d'un nouvel élément se fait depuis l'arborescence. Le menu de création apparaît après un clic droit sur le futur élément « père ».

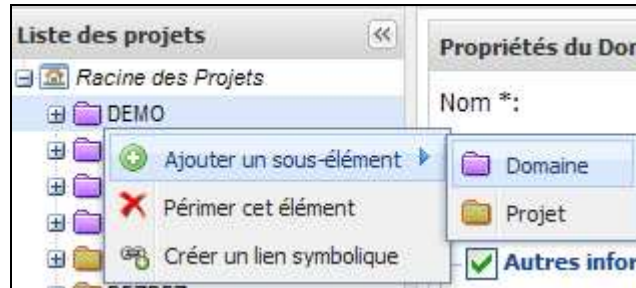


Figure 2. Menu de création d'un élément

La création n'est effective qu'après sauvegarde des informations du nouvel élément.

Type de l'élément père	Types possibles pour l'élément fils
Racine des projets	<ul style="list-style-type: none"> <li>• Domaine</li> <li>• Projet</li> </ul>
Domaine	<ul style="list-style-type: none"> <li>• Domaine</li> <li>• Projet</li> </ul>
Projet	<ul style="list-style-type: none"> <li>• Sous-projet</li> <li>• Entité applicative</li> </ul>
Sous-projet	<ul style="list-style-type: none"> <li>• Sous-projet</li> <li>• Entité applicative</li> </ul>
Entité applicative	

Figure 3. Tableau des types d'éléments fils possibles

## 1.2 Périmé un élément

Périmé un élément se fait via le même menu contextuel que pour créer un nouvel élément.

L'élément périmé n'est pas supprimé physiquement de la base de données, il est retiré de la liste des projets et déplacé dans la corbeille.

## 1.3 Créer un lien symbolique

Il est possible de créer des liens symboliques. Un lien symbolique peut exister entre un domaine et un autre domaine, ou un domaine et un projet.

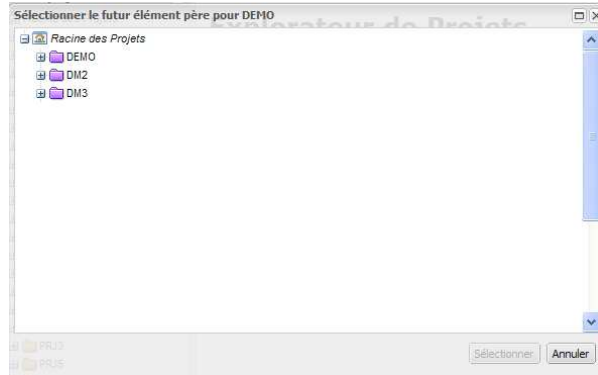



Figure 4. Fenêtre de création du lien symbolique

La fenêtre permet de sélectionner le nouvel élément « père » pour l'élément sélectionné.

Un lien symbolique permet d'avoir un même élément présent dans autant de domaines que nécessaire. Ainsi cet élément peut impacter la qualité de plusieurs domaines.

Un élément présent via un lien symbolique est représenté par une icône identique à un élément « normal », avec l'élément graphique  en surimpression.

## 1.4 Les propriétés d'un domaine

Un domaine représente un regroupement fonctionnel. Ses caractéristiques sont :

- Le **nom** : Le nom du domaine.
- Une **description** : Une description du domaine.
- La **date de création** : La date de création du domaine.
- La **date de dernière modification** : La date de dernière modification du domaine.
- Le **poids** : le poids du domaine dans le calcul de la qualité du domaine « parent »

Une section « **Autres informations** » regroupe deux champs qui peuvent servir à personnaliser un domaine en lui ajoutant diverses informations. Cette section apparaît en cochant la boîte à cocher « Autres informations ».

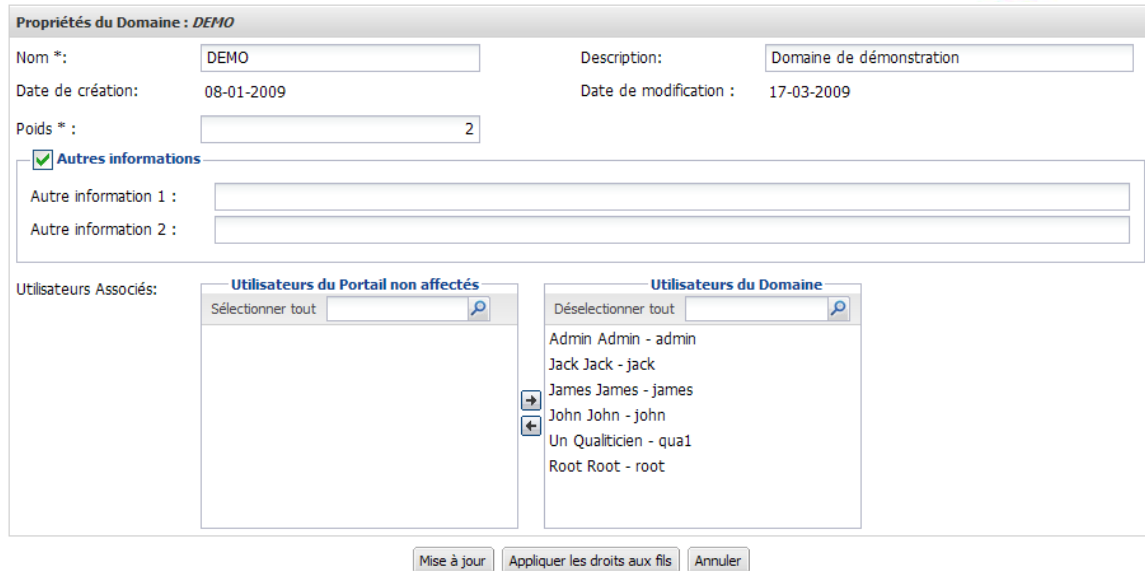


Figure 5. Propriétés d'un domaine

L'utilisateur doit indiquer quels utilisateurs ont le droit d'accéder au projet via la section représentée dans la Figure 6. La liste des utilisateurs affectables correspond à la liste des utilisateurs ayant accès à l'élément père, ou, dans le cas où l'élément père est la racine des projets, à la liste des utilisateurs enregistrés dans le portail CAQS.

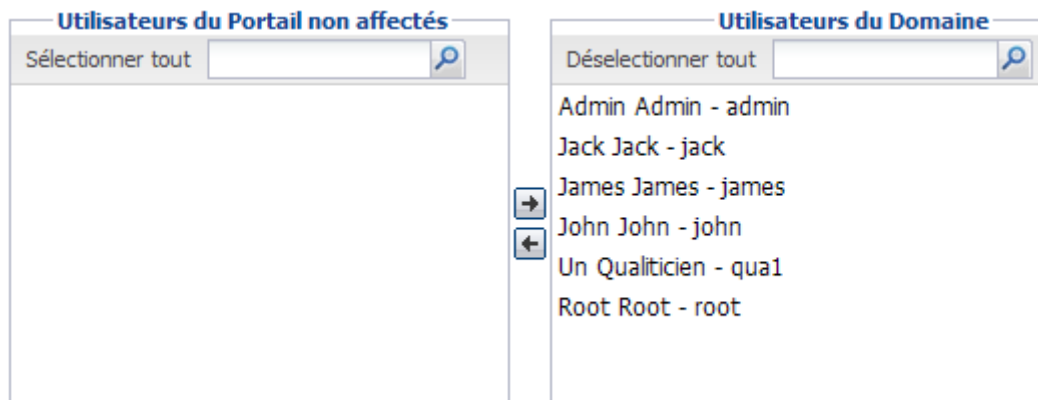


Figure 6. Gestion des droits et actions disponibles

L'action « **Mise à jour** » permet la sauvegarde des informations qui viennent d'être modifiées.

L'action « **Annuler** » permet d'annuler les modifications effectuées sur le projet depuis la dernière sauvegarde. Ce bouton est désactivé tant qu'aucune modification n'est faite.

L'action « **Appliquer les droits aux fils** » applique les droits du projet à ses éléments fils. De plus, elle sauvegarde les modifications apportées au projet.

## 1.5 Les propriétés d'un projet

Un projet est constitué d'au moins un élément fils. Les caractéristiques d'un projet sont identiques à celles d'un domaine et ne seront pas redéfinies dans cette section.

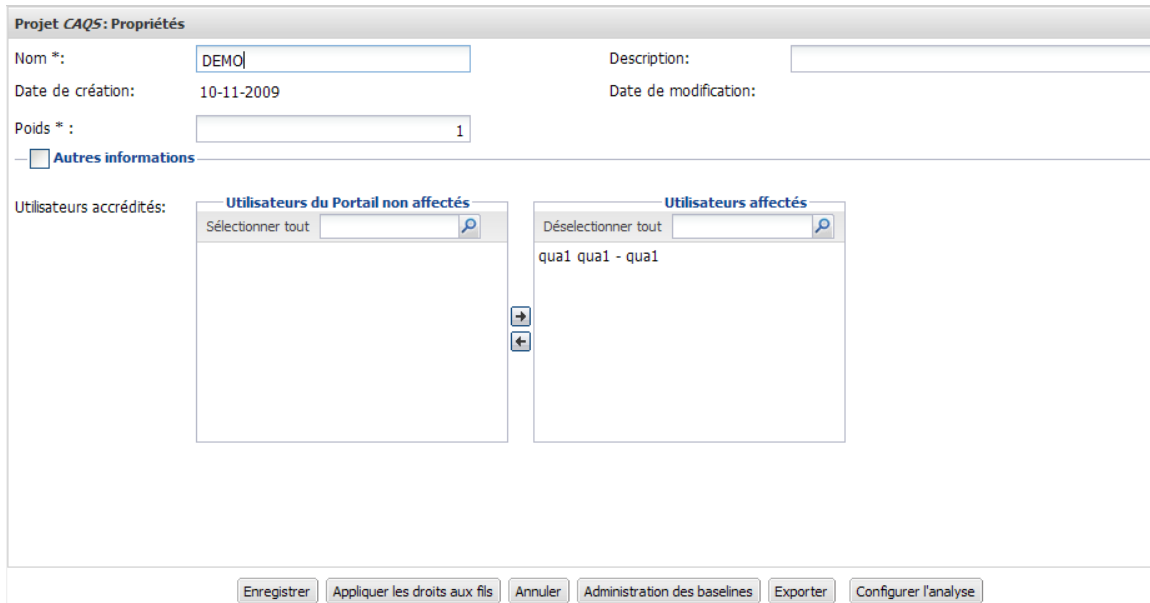


Figure 7. Propriétés d'un projet

Les utilisateurs pouvant potentiellement avoir accès au projet ne sont que ceux ayant déjà accès à l'élément père direct dans l'arborescence. Cela signifie que les seuls droits pouvant être affectés au projet sont ceux déjà affectés au domaine parent.

## 1.6 Les propriétés d'un sous-projet

Un sous-projet est un élément fils d'un projet ou d'un autre sous-projet. Il peut contenir d'autres sous-projets ou une ou plusieurs entités applicatives.

Les propriétés d'un sous-projet qui sont identiques à celles d'un projet, ne seront pas redéfinies dans cette section.

Le **poids** d'un sous-projet indique quel sera son coefficient dans le calcul des objectifs au niveau de l'élément père (projet ou sous-projet).

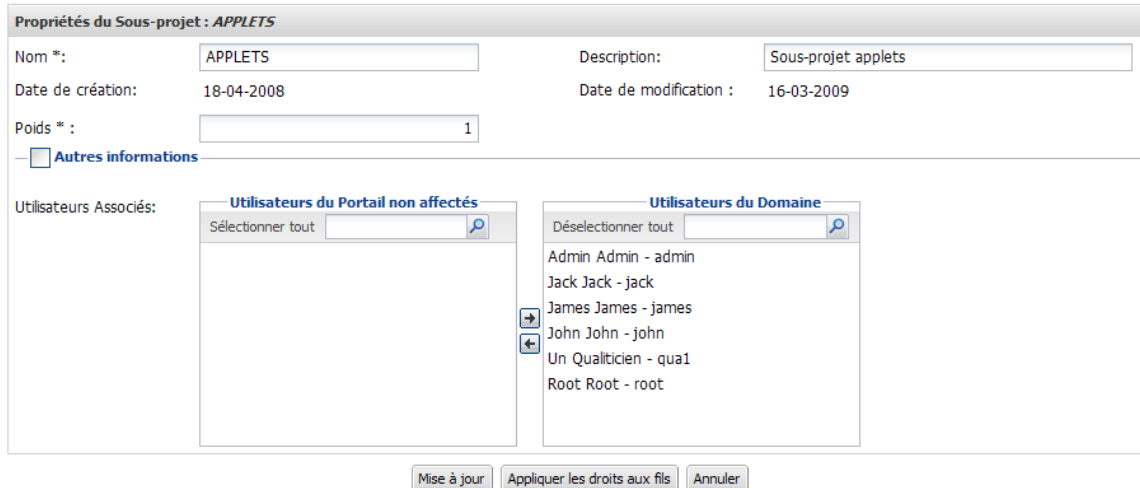


Figure 8. Propriétés d'un sous-projet

Les utilisateurs pouvant potentiellement avoir accès au sous-projet ne sont que ceux ayant déjà accès à l'élément père direct dans l'arborescence. Cela signifie que les seuls droits pouvant être affectés au sous-projet sont ceux déjà affectés au projet.

## 1.7 Les propriétés d'une entité applicative

Une entité applicative représente un élément qui sera analysé. Elle appartient à un sous-projet ou à un projet.


En plus des propriétés **nom**, **description**, **date de création**, **date de modification** et « **autres informations** » précédemment définies, une entité applicative a les propriétés suivantes :

- **Répertoire Source** : Il contient l'application (ou la partie d'application) à analyser. Le répertoire indiqué doit contenir le sous-répertoire suivant :
  - o **src** : Ce répertoire doit contenir l'arborescence complète des fichiers source à analyser, tels qu'ils ont été définis. Nous verrons dans une section ultérieure comment architecturer sont contenu.
- **Librairies** : Elle définit le répertoire qui contient l'ensemble des bibliothèques utilisées dans le projet. Nous développerons cette propriété plus loin.
- **Poids** : De la même manière que pour un sous-projet, le poids définit le coefficient lors du calcul d'une entité applicative au sein d'un sous-projet.
- **Modèle qualimétrique** : Cette propriété définit le modèle qualimétrique utilisé par l'entité applicative.
- **Dialecte associé** : Le dialecte associé à l'entité applicative correspond à la « version » du langage correspondant à l'entité applicative qui va être analysée.
  - o Dans le cas d'un dialecte « VB », un champ « **Fichier de configuration projet** » devient disponible. Ce champ doit contenir le chemin vers le fichier de définition du projet Visual Studio (sln, etc.), nom du fichier inclus. **Ce chemin doit être relatif au répertoire « src ».**

- **Téléchargement des sources** : ce champ permet de télécharger vers le serveur un fichier ZIP. Ce fichier doit contenir les sources à analyser, placées dans un répertoire « src », et optionnellement un répertoire « bin ». Le contenu de ce fichier sera décompressé dans le répertoire défini par la propriété « Répertoire Source ».
- **Référentiel SCM** et **Module** : Ces deux propriétés permettent de définir l'accès aux sources si celui-ci doit se faire via un gestionnaire de version de type CVS ou Subversion, entre autres.

Seuls les utilisateurs ayant des droits d'accès à l'élément père peuvent potentiellement avoir le droit d'accéder à l'entité applicative.

Une entité applicative n'a pas d'élément fils.

Dans le cas où le téléchargement des sources échoue, les erreurs survenues sont disponibles en cliquant sur l'icône .

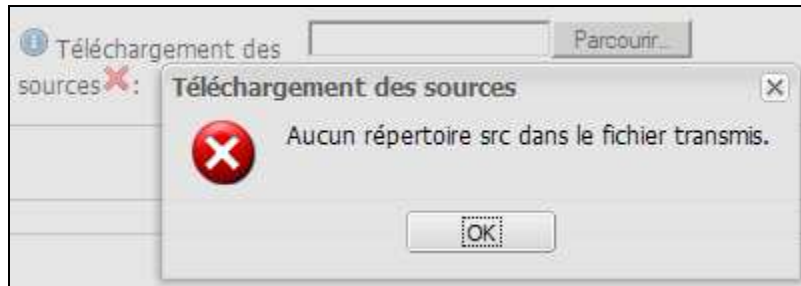


Figure 9. Exemple d'erreur de téléchargement de sources

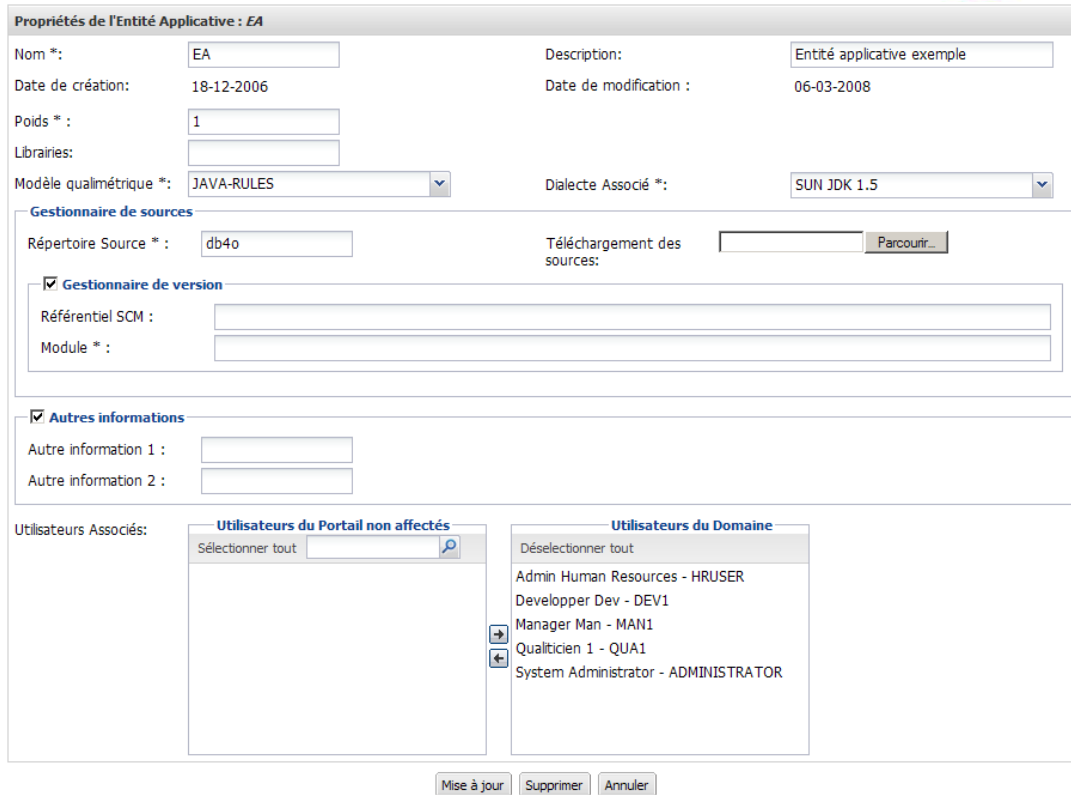


Figure 10. Propriétés d'une entité applicative

### 1.7.1 Structure du répertoire source.

Le répertoire « Source » (défini au niveau de l'entité applicative) doit contenir plusieurs sous-répertoires :

- Un répertoire **src** qui contiendra les sources de l'entité applicative.
- Un répertoire **bin** (optionnel) qui contiendra les versions compilées des sources de l'entité applicative.

Afin que l'analyse d'un projet se passe correctement, il faut architecturer les répertoires de la manière suivante pour les entités applicatives contenant du code :

- **Java**
  - o le répertoire **src** doit contenir les fichiers « .java » se trouvant dans une arborescence représentant les packages. Un répertoire doit être créé pour un package et doit contenir les sources de ce package java. Ainsi une classe nommée « p1.p2.p3.Classe01 » se trouvera dans le répertoire  
« Répertoire Source »/src/p1/p2/p3
  - o le répertoire **bin** (optionnel) doit contenir la même arborescence que le répertoire **src**. La différence est qu'il contiendra les fichiers « .class » (java compilés) au lieu des fichiers « .java ».
- **C/C++**



- Le répertoire **src** doit contenir les fichiers sources du projet **pré-compilés** en **conservant les commentaires** (extension .i ou .ii).
  - Le répertoire **bin** est ignoré pour les analyses de code C/C++.
- **COBOL**
  - Se référer à la section dédiée 2.3, page 12.
- **VB6**
  - Le répertoire **src** doit contenir les fichiers sources à analyser.
  - Le répertoire **bin** est ignoré pour les analyses de code VB6.
- **C#**
  - Le répertoire **src** doit contenir les fichiers sources à analyser. Ces fichiers doivent porter l'extension « .cs ».
  - Le répertoire **bin** est ignoré pour les analyses de code C#.

Optionnellement, il est possible de pré-configurer le répertoire source avant une analyse. Cette pré-configuration peut être un déplacement de fichiers sources dans le répertoire source depuis un autre répertoire, une compilation des fichiers sources. Cela peut être aussi la récupération d'un ensemble de sources depuis un gestionnaire de configuration quelconque (CVS, ClearCase, etc.) ou l'exécution d'un programme annexe.

Ces actions sont exécutées sous la forme de tâches ANT<sup>1</sup>. L'ensemble des actions exécutables consiste donc en l'ensemble des tâches ANT existantes et peut être facilement étendu.

Les tâches ANT sont définies dans un fichier « build.xml » générique et qui s'adapte à la grande majorité des cas d'utilisation. Cependant, si l'analyse nécessite des options très spécifiques, il est possible de redéfinir les tâches ANT dans un fichier « build.xml » spécifique qui sera placé dans le « Répertoire Source ». Ce fichier contiendra l'ensemble des tâches ANT à exécuter et devra être conforme à la syntaxe définie par la fondation Apache<sup>2</sup> pour la rédaction de ce type de fichiers<sup>3</sup>.

## 1.7.2 Le répertoire Librairies

Le répertoire « Librairies » contient les bibliothèques utilisées par le code source à analyser pour l'entité applicative.

Il est utilisé dans les analyses de code suivantes :

- **Java**
  - Le répertoire Librairies peut contenir l'ensemble des fichiers « .jar » utilisés par le code source lié à l'entité applicative.
- **VB6**
  - Le répertoire doit contenir les fichiers « .ocx » utilisés par le code source lié à l'entité applicative.

---

<sup>1</sup> <http://ant.apache.org/>

<sup>2</sup> <http://www.apache.org>

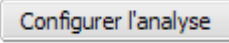
<sup>3</sup> <http://ant.apache.org/manual/index.html>

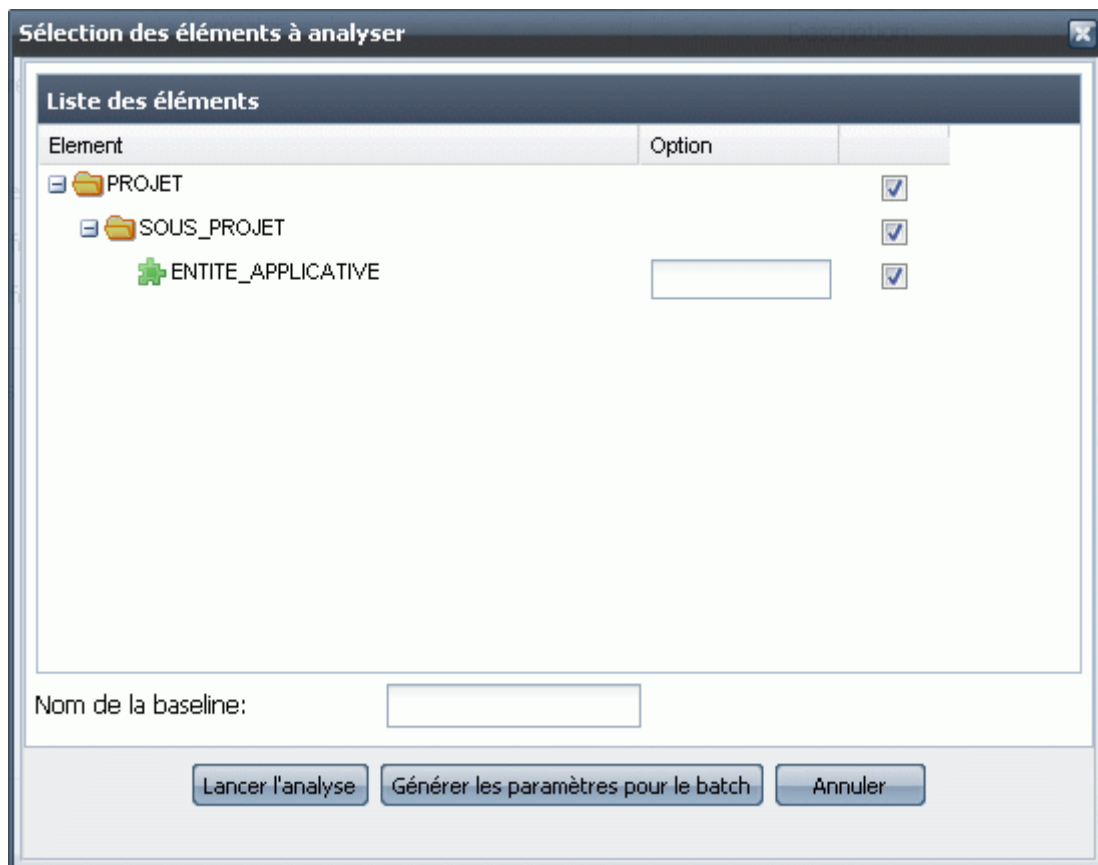
## 2 Analyser un projet

*L'analyse d'un projet ne doit être faite qu'une fois celui-ci entièrement défini et configuré.*




Une analyse est représentée dans CAQS par une « **baseline** ». Celle-ci a un identifiant qui doit être unique. L'unicité de cet identifiant ne dépend pas du projet. Cela signifie que deux projets différents ne peuvent avoir deux baselines ayant le même identifiant.

Le lancement d'une analyse se fait depuis la page de définition du projet à analyser.

En cliquant sur le bouton , une fenêtre s'ouvre qui permet de configurer l'analyse.



Sélection des éléments à analyser

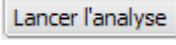
Element	Option	
 PROJET		<input checked="" type="checkbox"/>
 SOUS_PROJET		<input checked="" type="checkbox"/>
 ENTITE_APPlicative	<input type="text"/>	<input checked="" type="checkbox"/>

Nom de la baseline:

Figure 11. Fenêtre de configuration d'une analyse

Cette fenêtre permet de sélectionner les entités applicatives à analyser. Un champ « **Option** » est disponible pour chacune de ces entités applicatives.

## 2.1 Analyse interactive

Cliquer sur le bouton  permet de lancer l'analyse.

Après le lancement de l'analyse, une fenêtre apparaît, qui récapitule les informations sur l'analyse.

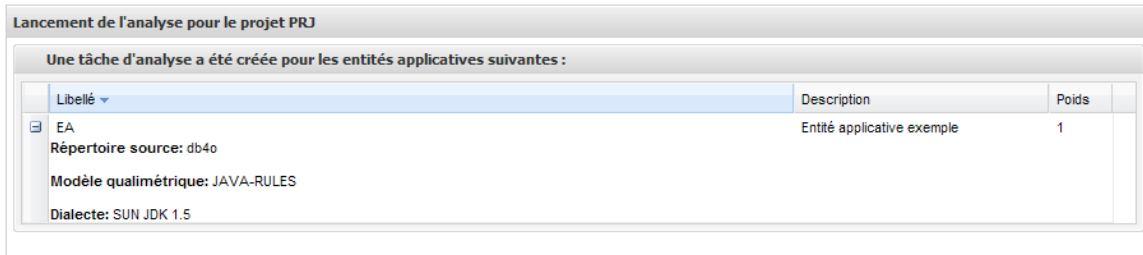
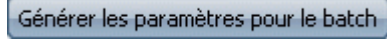


Figure 12. Récapitulatif du lancement d'une analyse

Un message, situé dans la zone de message, permet de connaître l'avancement de l'analyse.

## 2.2 Analyse en batch

Cliquer sur le bouton  permet d'afficher la liste des valeurs de paramètres nécessaires à l'exécution de la même analyse, en batch, via une ligne de commande.



Ces valeurs doivent être passées en paramètres du script fourni dans l'installation de CAQS (côté serveur):

- Unix/Linux : <CAQS\_DIR>/Traitements/ant/batch/antOnly/analysis.sh
- Windows : <CAQS\_DIR>\Traitements\ant\batch\antOnly\analysis.bat

NB : <CAQS\_DIR> représente le répertoire dans lequel CAQS a été installé.

Exemple d'utilisation de ce script sous Unix (l'ensemble de la commande tient sur une seule ligne):

```
caqs> analysis.sh -DprojectId=20090605162158984139025 -DprojectName=PROJET -  
DbaselineName= -DuserId=qua1 -Demail=xxx@company.com -Dealist=20090605162212453139029 -  
Deaoptionlist=
```

## 2.3 Analyse d'un projet Cobol

L'analyse d'un projet Cobol se fait à l'aide de l'outil DevEntreprise.

Cette analyse est liée à une ou plusieurs collections définies dans DevEntreprise. Ainsi, avant de lancer toute analyse, il est nécessaire de vérifier au préalable les paramètres suivants :

- Définition correcte de la connexion à DevEntreprise dans CAQS.
- Définition correcte et complète de la ou des collections à analyser dans DevEntreprise

### 2.3.1 Définition de la connexion à DevEntreprise

Cette définition se fait au niveau du plugin DevEntreprise de CAQS.

Elle ne doit être faite qu'une seule fois, ou après un changement d'un des paramètres de connexions.

Ce plugin se trouve dans le répertoire d'installation de CAQS, dans la sous-arborescence « Traitements/deventreprise/plugin ». Il faut configurer le fichier « config.properties » en y indiquant les paramètres de connexion à la base SQLServer utilisée par DevEntreprise et les paramètres de connexion au serveur Metadata.

Les paramètres suivants doivent être renseignés :

- **sqlserver.servername** : Nom du serveur SQLServer
- **sqlserver.username** : Nom de l'utilisateur habilité à se connecter à la base de données SQLServer.
- **sqlserver.password** : Mot de passe de l'utilisateur habilité à se connecter à la base de données SQLServer.
- **sqlserver.databaseName** : Nom de la base de données contenant les données du Metadata.
- **metadataServer** : Nom du serveur contenant le Metadata Analyser
- **metadataServer.port** : Port d'écoute du Metadata Analyser
- **metadata.installPath** : Répertoire d'installation partagé du Metadata Analyser
- **learning.maxfailed** : Nombre d'erreurs de learning permises avant de déclarer le learning comme échoué.
- **metadataServer.deleteColl** : Placer à 1 pour demander une suppression des entités précédemment collectées avant de lancer une analyse.

### 2.3.2 Définition des paramètres d'analyse pour une entité applicative donnée

L'analyse de sources Cobol pour une entité applicative est faite en fonction de collections définies dans DevEntreprise.

Ces collections doivent donc être impérativement correctement définies dans DevEntreprise pour que l'analyse se fasse sans erreur.

Il suffit d'indiquer, pour une entité applicative données, quelles collections devront être analysées, via le champ « Autre information 1 ».

Ce champ doit être renseigné en utilisant le format suivant :

- collections=[*Nom de la ou des collections à analyser*]

Dans le cas où plusieurs collections devront être analysées, les noms de celles-ci devront être séparés par des virgules.

Par exemple :

collections=TEST1,TEST2,TEST3

### 3 Administration des analyses

Il est possible de renommer ou de supprimer une analyse d'un projet.

Pour supprimer une analyse, il suffit de sélectionner dans la gestion des projets le projet qui la définit. Un bouton **Administration des baselines** est disponible. En cliquant dessus, une fenêtre apparaît qui liste les analyses associées à ce projet.

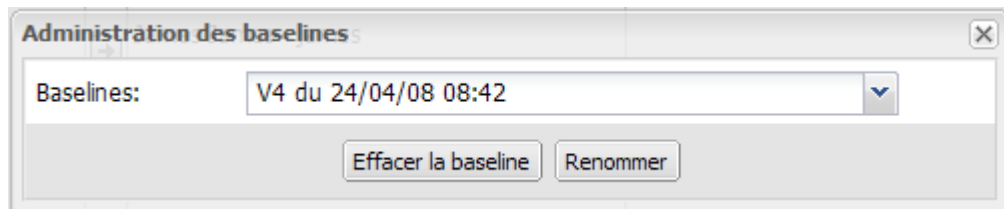


Figure 13. Gestion des analyses


Le bouton **Effacer la baseline** permet d'effacer la baseline sélectionnée. Contrairement aux autres éléments, les baselines sont supprimées physiquement de la base de données.

En cliquant sur le bouton **Renommer** après avoir sélectionné une baseline, il est possible de renommer la baseline sélectionnée.

## 4 Import/Export de projets


### 4.1 Export de projets

Exporter un projet se fait en le sélectionnant dans l'administration des projets.

Le bouton  permet de lancer l'export d'un projet. Cet export est lancé de manière asynchrone et une fois complétée, est téléchargeable depuis le bouton dédié dans le message correspondant.

## 4.2 Import de projets

L'import se fait en sélectionnant la « racine des projets » depuis l'administration des projets. L'import commence au clic sur le bouton « Télécharger ».

 Le fichier à importer doit être un fichier précédemment exporté depuis l'interface d'administration de CAQS.

## 5 Import/Export de modèles

L'import/export de modèles qualimétriques a été déplacé dans l'éditeur de modèles qualimétriques. Se référer à la documentation de l'éditeur de modèles qualimétriques pour plus d'informations.