**RAPPORT ANALYSE DU PROJET**

**INTRODUCTION**

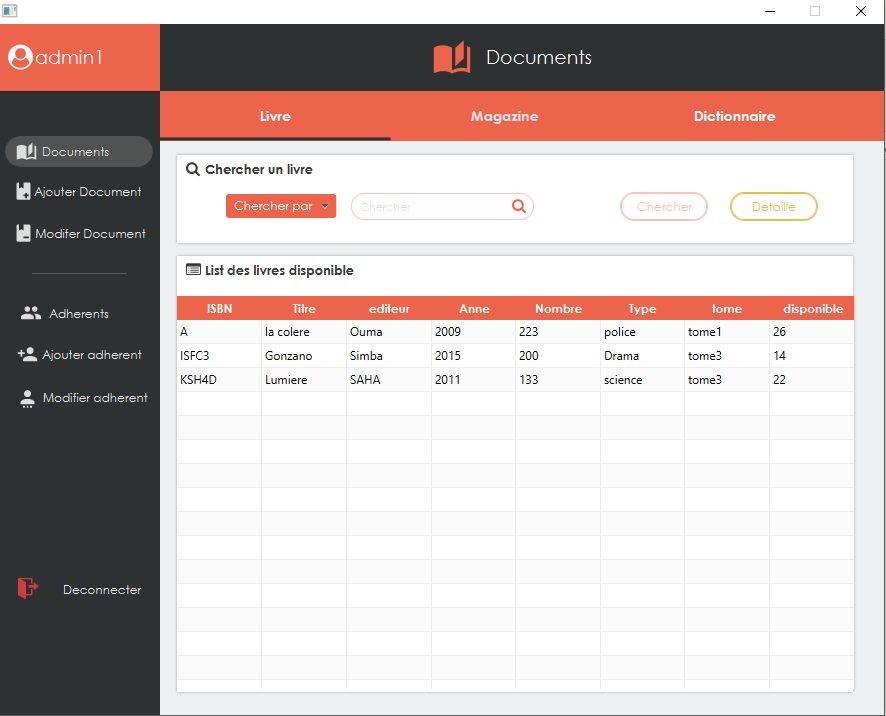
Personne ne peut douter aux jours d’aujourd’hui que l’informatique est une révolution fondamentale et innovante qui touche considérablement la vie humaine. En effet, loin d’être un phénomène effervescent, ou une tendance passagère, l’informatique est exploitée dans tous les aspects de la vie. Aucun domaine n’est resté à l’abri de cette politique qui est utilisé par des entreprises pour satisfait les besoins de ces consommateurs, a cette effet de nombres produits logiciel respectant des besoins fonctionnels de l’utilisateur et misent en production s’avèrent coûteuse pour ce dernier au niveau de la maintenance . Le problème qui y découle est celui de savoir comment analyser et retrouver tout éventuel problèmes dans une application?

Dans l’optique d’apporter une solution à cette interrogation, nous évoluerons suivant cette approche : de prime abord, nous présenterons le projet choisi, ensuite nous ferons l’analyse proprement dite du projet.

**Présentation du projet choisi**

En vu de mettre en pratique les notion d’évaluation de produit logiciel nous avons utiliser une **Application de Gestion d'une bibliothèque** libre et accessible sur **github**

**interface administrateur de l’application**



**ANALYSE DU PROJET**

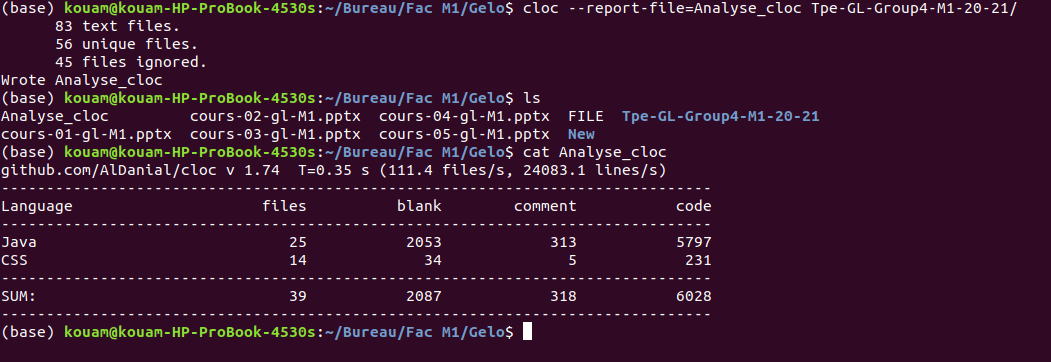
Pour analyser notre projet, nous nous sommes référé aux outils d’analyse de projet Open source et avons utilisé **cloc, sloccount, O3SMeasures** chacun ayant des particularités.

**Cloc**

Permet de compter les différences de lignes physiques de code source dans le fichiers donnés (fichiers zip, tar, et tout projet git) et / ou récursivement sous le répertoires donnés. Il est entièrement écrit en Perl, en utilisant uniquement des modules de la distribution standard. cloc compte les lignes vierges, les lignes de commentaires et les lignes physiques du code source dans de nombreux langages de programmation et produits le résultat sous plusieurs format.

nous l’avons installer sur notre distribution linux en utilisant la commande **sudo apt-get install -y cloc**. Nous avons accès a son manuelle d’utilisation avec **man cloc**

Analyse et sauvegarde du rapport dans le fichier **Analyse\_cloc**

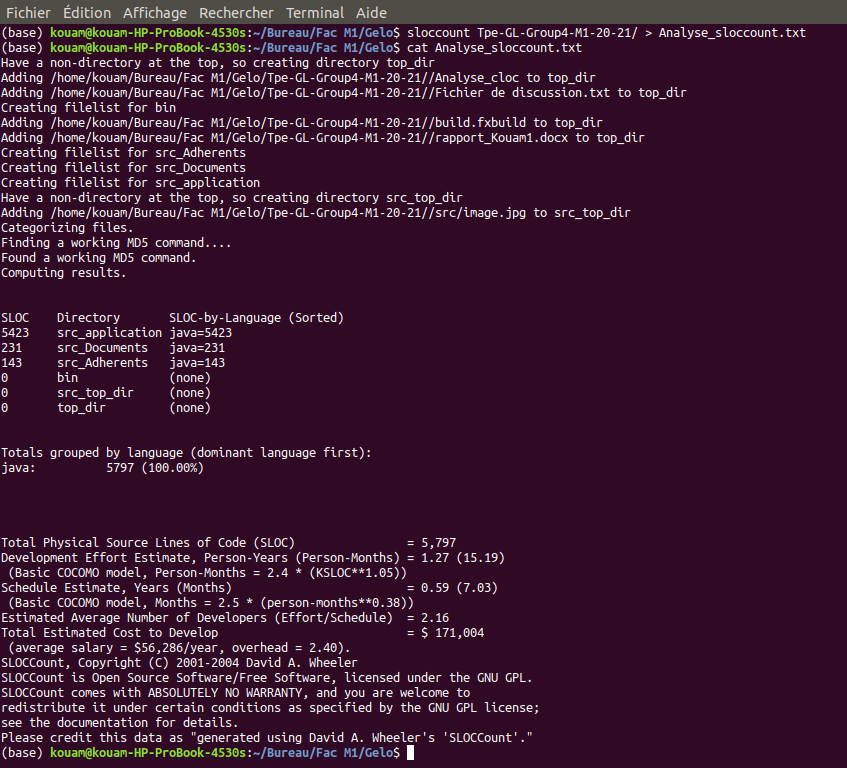


**sloccount**

Nous permet d’estimer même l'effort, le temps et le coût de développement d’un produit logiciel. Sans options, il utilisera le modèle de base **COCOMO**, qui effectue ces estimations uniquement à partir du nombre de lignes de code.

Nous l’avons installer sur notre distribution linux en utilisant la commande **sudo apt-get install -y sloccount**. Nous avons accès a son manuelle d’utilisation avec **man sloccount**

Analyse et sauvegarde du rapport dans le fichier **Analyse\_sloccount**



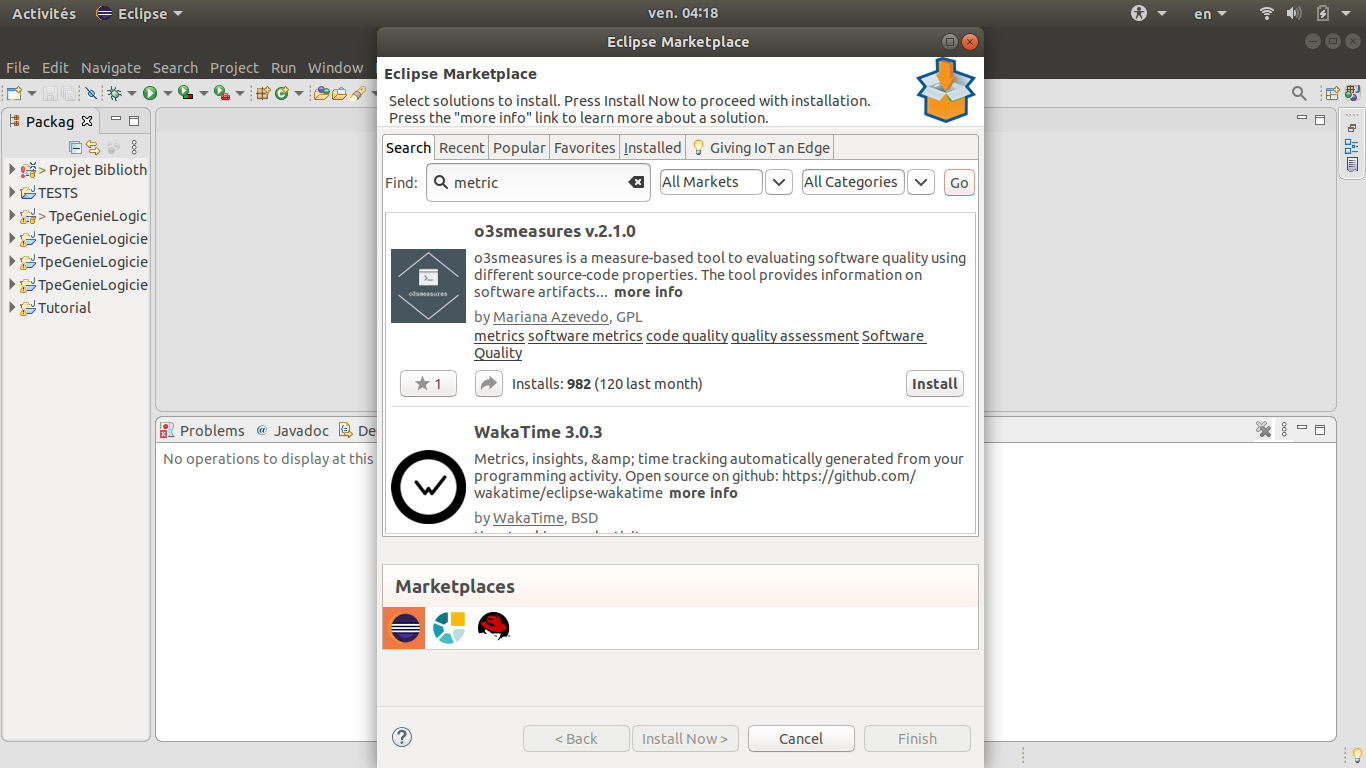
**O3SMeasures**

c’est d’outil principal d’analyse que nous avons utilisé. Il est basé sur des mesures pour évaluer la qualité d'un logiciel à l'aide de différentes propriétés de code source. L'outil fournit des informations sur les artefacts logiciels en utilisant les 15 mesures les plus citées dans les études empiriques pour étudier la qualité interne des logiciels au cours de la dernière décennie + 11 mesures de soutien (telles que le nombre de classes, le nombre de références de méthodes, le nombres d’attributs etc.).

Installation et utilisation

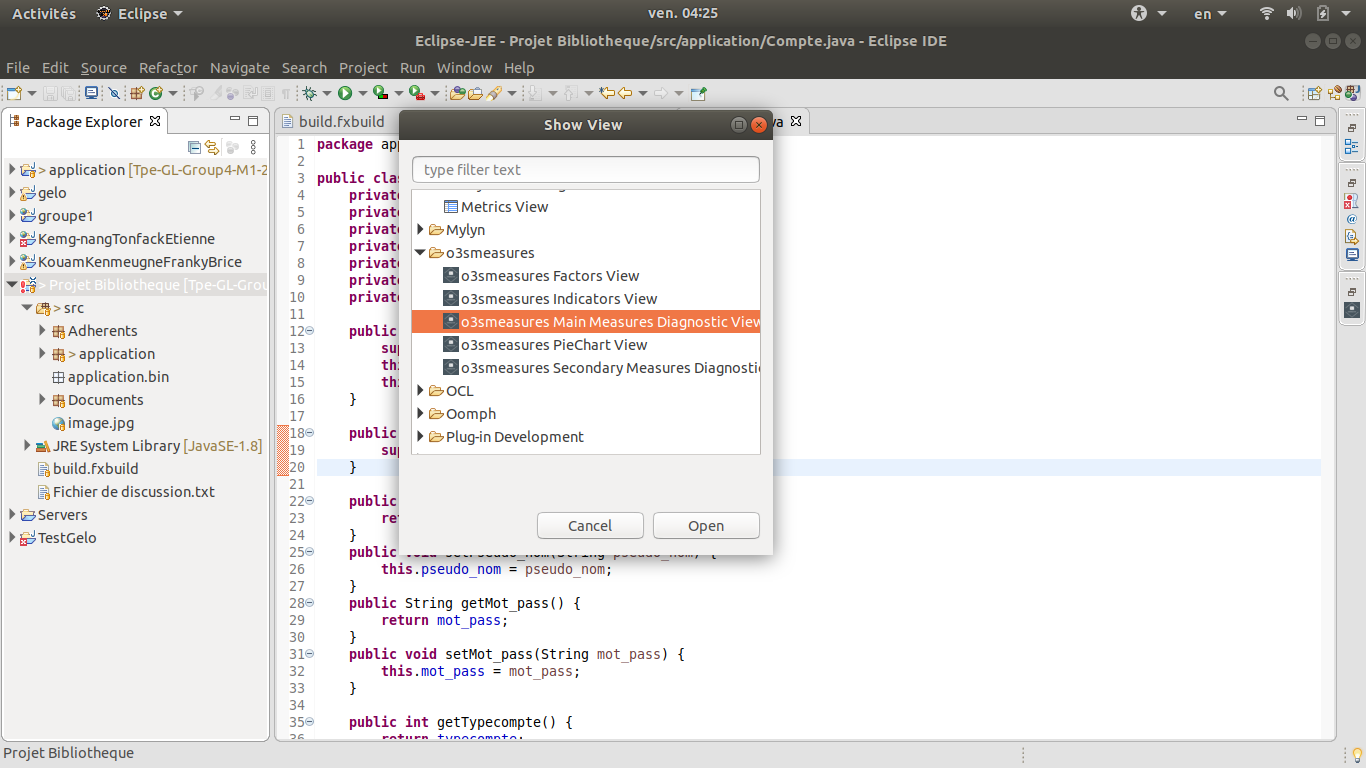
**O3SMeasures** s’installe comme plugin dans Eclipse

**installation**

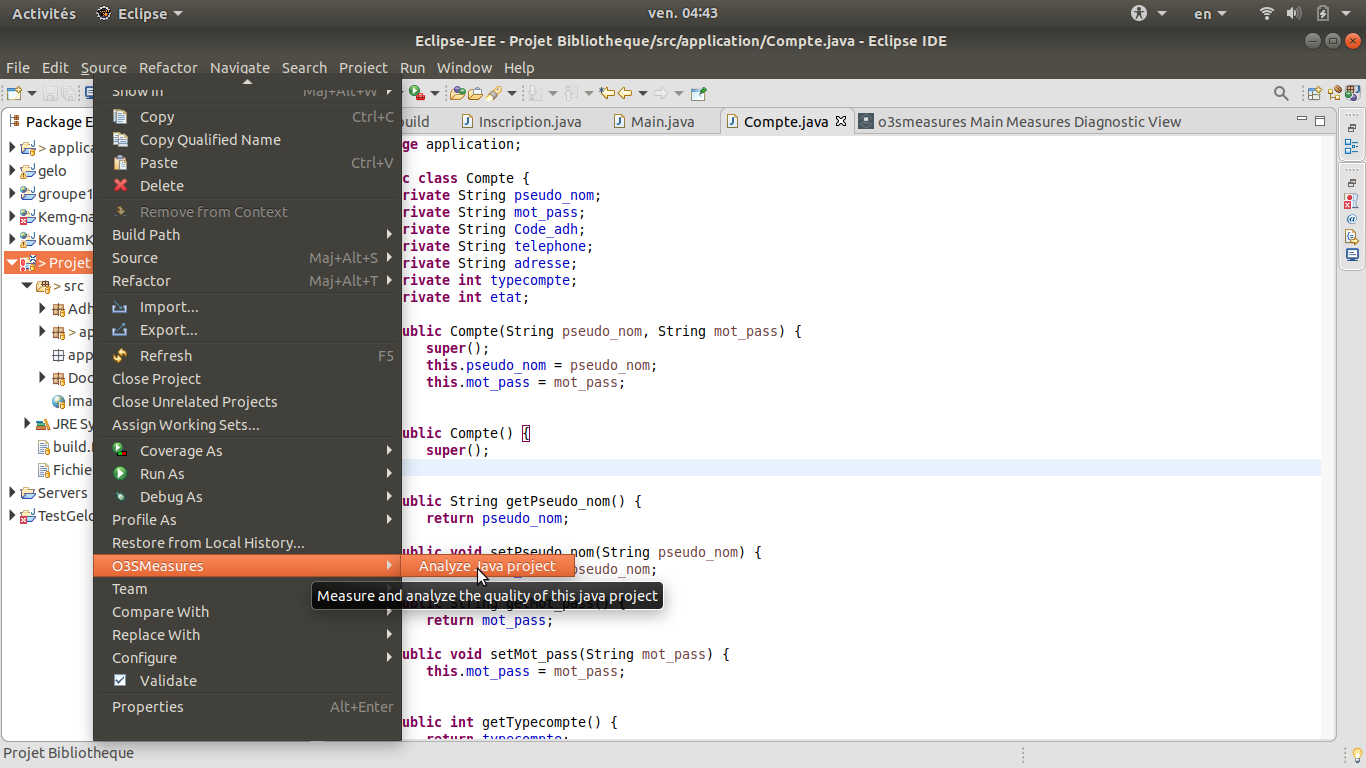
****

**Utilisation**

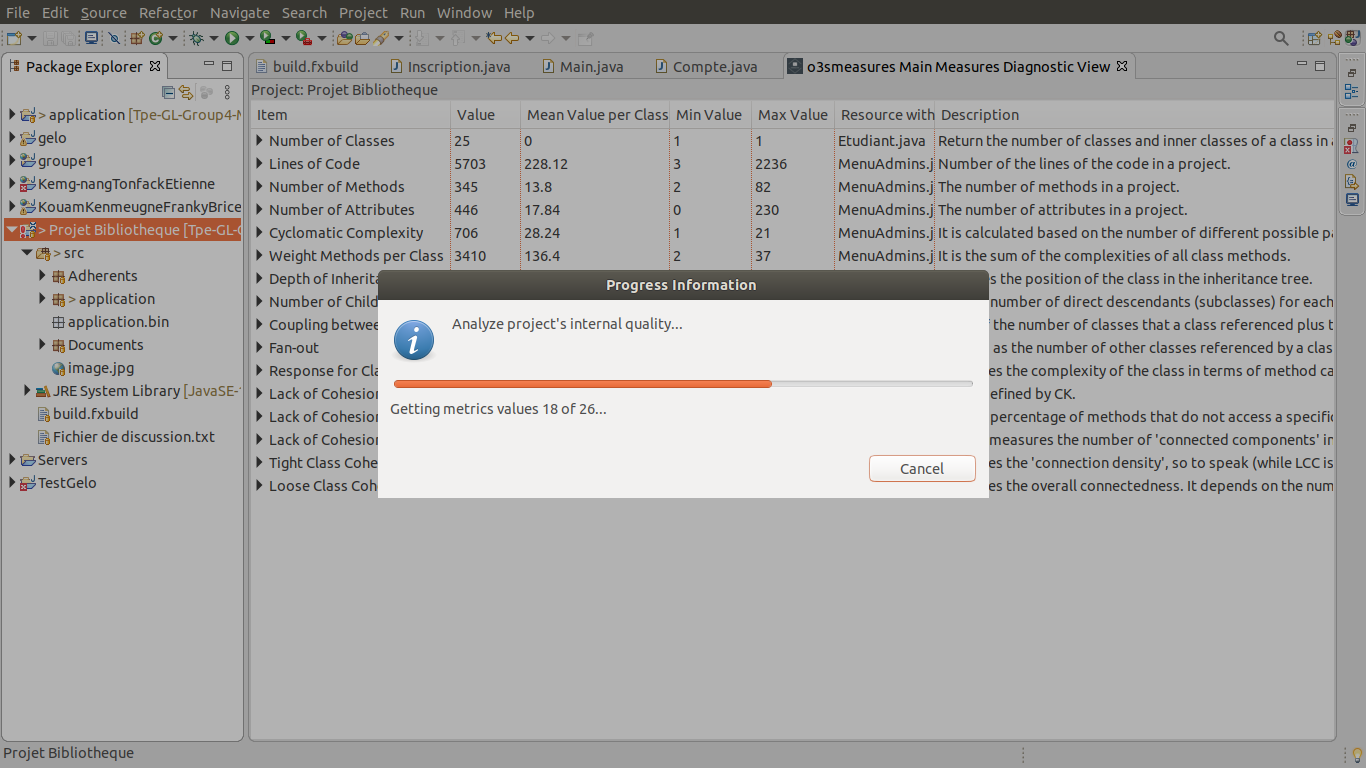
* création de la vue



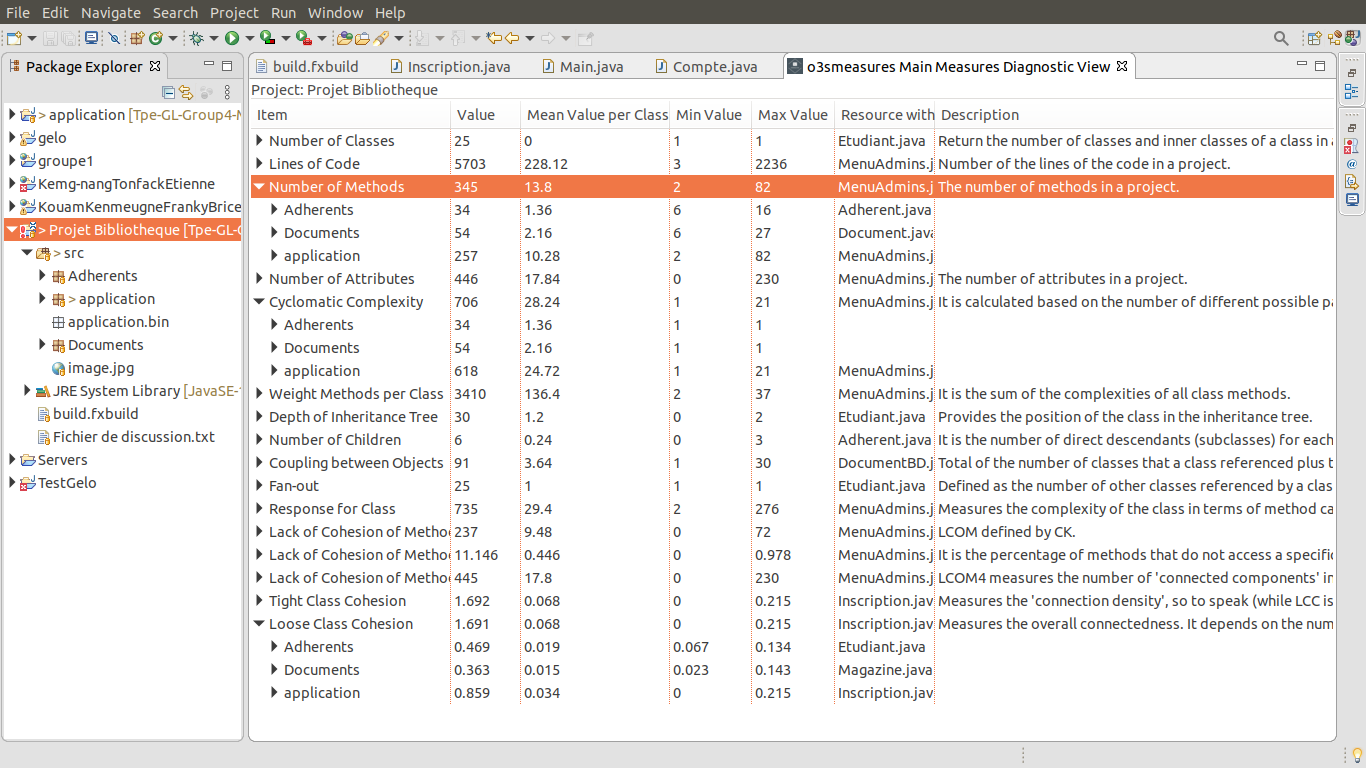
* lancement de l’analyse

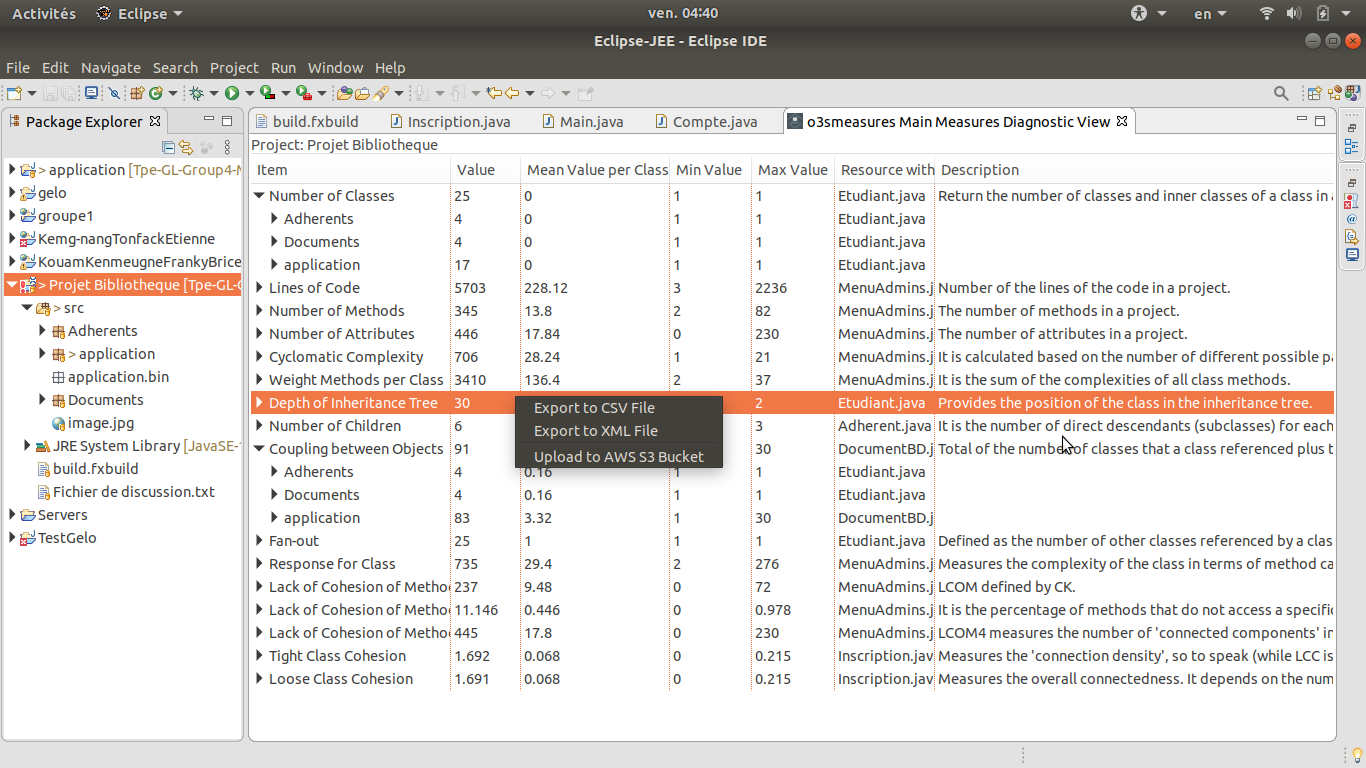


* Analyse en cours



* Résultat analyse



* Exportation des résultats
* Contenu finale de notre répertoire

