

Projet de Génie Logiciel



ArcelorMittal

Rapport du cas d'étude : ArcelorMittal

Dossier de conception

Table des matières

I. Le dossier de conception	2
II. Le modèle d'architecture global	3
III. Le modèle d'architecture détaillé	4
1. Package model :	4
2. Package dataClasses :	4
3. Package controller :	5
4. Package view :	6
5. Package assets :	6
6. Package test :	6
IV. Les fonctionnalités	7
V. Le modèle conceptuel de données	9

I. Le dossier de conception

Ce document a pour but de présenter à la fois une vision d'ensemble globale et une analyse approfondie des étapes impliquées dans notre projet qui répond à la problématique soulevée par ArcelorMittal. Nous proposons également une série de documents pour présenter notre conception, notamment :

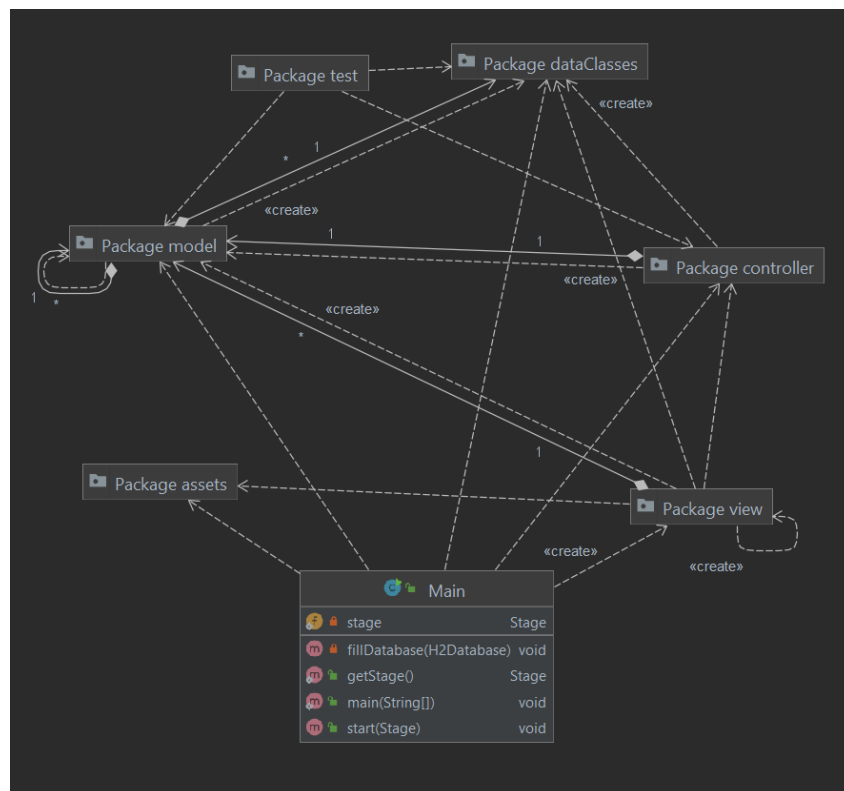
- L'architecture générale
- L'architecture détaillée
- Les fonctionnalités
- Un modèle conceptuel de données

II. Le modèle d'architecture global

Afin de représenter une architecture générale de notre logiciel, voici un diagramme de packages représentant la structure d'un système en groupant les éléments en packages. Il aide à visualiser les dépendances entre les packages pour faciliter la compréhension de la conception du système.

Notre projet est découpé en 6 packages différents qui vont être détaillés ensuite :

1. Package model : contient les différentes classes utiles pour Orowan.exe
2. Package dataClasses : contient les classes des objets Orowan
3. Package controller : contient la classe H2Database en charge de la base de données
4. Package view : contient les classes de view des différents menus
5. Package assets : contient une classe Assets qui charge une seule fois les ressources
6. Package test : contient deux classes utilisées pour tester les fonctionnalités internes



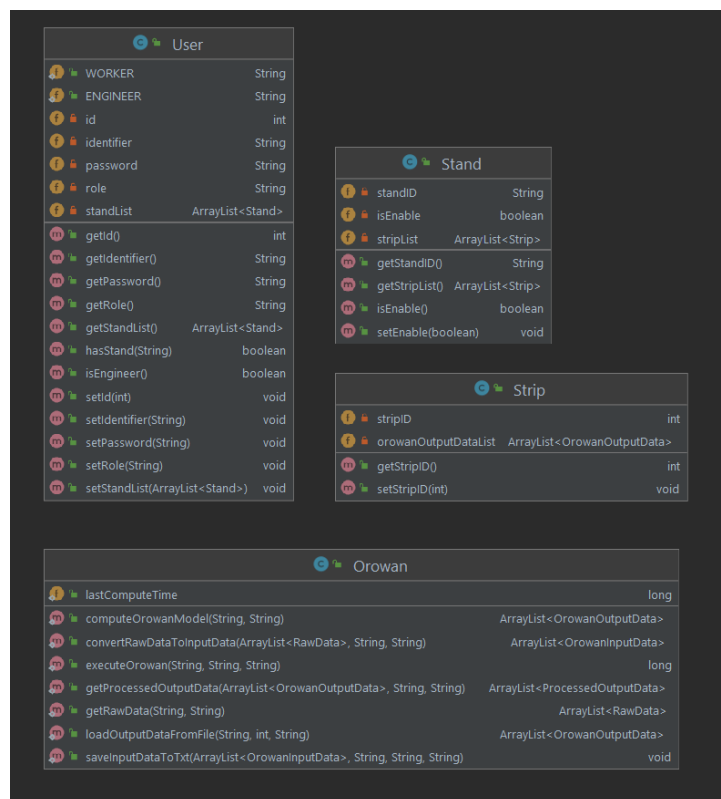
III. Le modèle d'architecture détaillé

Dans cette partie, nous allons présenter plus en détail chaque package et ses classes associées.

1. Package model :

Il contient les classes suivantes :

- Orowan : permet d'exécuter Orowan.exe et de sauvegarder et charger les données utiles pour son exécution
- User, Stand et Strip qui permettent d'instancier des objets contenant les informations des utilisateurs, stand et strip pour une utilisation plus pratique de ces données



2. Package dataClasses :

Ce package contient les 4 classes des 4 type de données manipulées par Orowan et l'application. **RawData** sont les données brutes qu'on lit dans les fichiers du type « 1939351_F2 ». **OrowanInputData** sont les données converties de ces données brutes que l'on peut passer en entrée d'Orowan.exe. **OrowanOutputData** ont les données de sortie d'Orowan.exe. Ces dernières permettent de générer les données moyennes **ProcessedOutputData**.

RawData	OrowanOutputData	OrowanInputData
Lp int	rollSpeed double	rollSpeed double
MatID int	stand String	cas int
stand String	matId int	He double
xTime double	casId int	Hs double
xLoc double	errors String	Te double
EnThick double	offsetYield double	Ts double
ExThick double	friction double	diam_WR double
EnTens double	rollingTorque double	WRyoung double
ExTens double	sigmaMoy double	offset_ini double
RollForce double	sigmaIni double	mu_ini double
FSlip double	sigmaOut double	Force double
Diameter double	sigmaMax double	G double
RolledLengthForWorkRolls double	forceError double	
youngModulus double	slipError double	
BackupRollDia double	hasConverged boolean	
RolledLengthForBackupRolls double		
mu double		
torque double		
averageSigma double		
inputError double		
LubWFIUp double		
LubWFILo double		
LubOilFIUp double		
LubOilFILO double		
WorkRollSpeed double		

ProcessedOutputData
xTimeMS float
rollingSpeed double
sigma double
friction double
stand String
stripID int

3. Package controller :

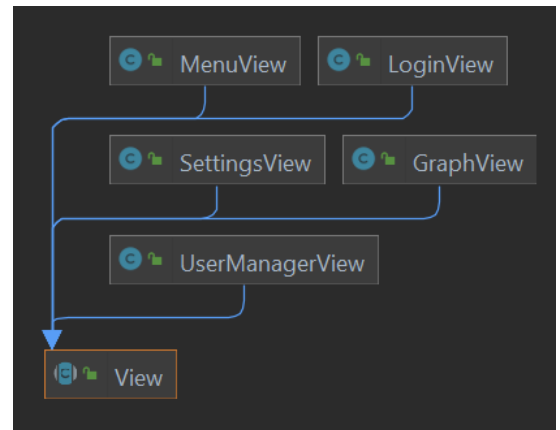
Ce package contient la classe H2Database qui est à la charge de toutes les opérations avec la base de données (BDD) h2 :

- Création de la BDD
- Connexion avec la BDD
- Récupération d'informations dans la BDD
- Ajout, retrait, mise à jour des informations dans la BDD

H2Database
addStand(String) void
addStandForUser(int, String) void
addStrip(int, String) void
addUser(String, String, boolean) void
disableStand(String) void
enableStand(String) void
getAllStands() ArrayList<Stand>
getInstance() H2Database
getStands_aux(PreparedStatement, ArrayList<Stand>) ArrayList<Stand>
getStrips(String) ArrayList<Strip>
getUserByUsername(String) User
getUserIdentifier() String
getUserStands() ArrayList<Stand>
getUserStands(int) ArrayList<Stand>
getUsers() ArrayList<User>
isRawDataEmpty(int, String) boolean
isUserEngineer() boolean
loadOrowanData(int, String) ArrayList<OrowanOutputData>
loadProcessedOutputData(int, String) ArrayList<ProcessedOutputData>
loadRawData(int, String) ArrayList<RawData>
loadRawData_aux(int, String, ArrayList<RawData>) ArrayList<RawData>
loadUserStands() ArrayList<Stand>
loginUser(String, String) boolean
refreshUserStands() void
removeStandForUser(int, String) void
removeUser(int) void
setUpDatabase() void
setUserIsEngineer(boolean) void
updateStand(String, boolean) void
updateUser(int, String, String, boolean) void
writeOrowanData(ArrayList<OrowanOutputData>) void
writeProcessedOutputData(ArrayList<ProcessedOutputData>) void
writeSensorData(ArrayList<RawData>) void

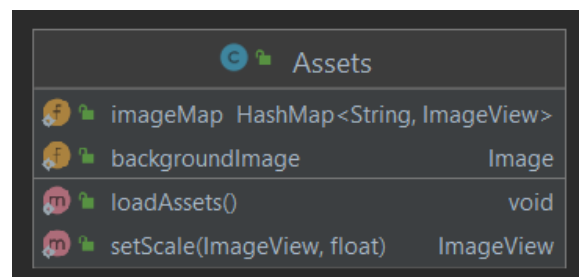
4. Package view :

Ce package contient les 5 classes de Vue utilisées pour la conception des menus. Ces 5 classes héritent d'une superclasse View qui elle-même hérite de la classe Scene de javafx. Cela permet de généraliser certaines informations communes à toutes nos Vue.



5. Package assets :

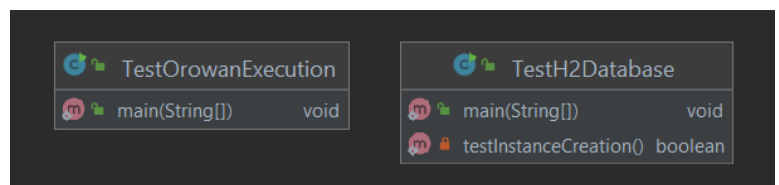
Ce package contient la classe Assets qui possède deux attributs *statics* imageMap (qui contient l'ensemble des images utilisées dans le projet, identifiées par une clef String) et backgroundImage (qui contient l'image de fond de l'écran de connexion).



Ces attributs sont *statics* afin de ne les charger qu'une seule fois et de pouvoir les récupérer à partir de toutes les autres classes.

6. Package test :

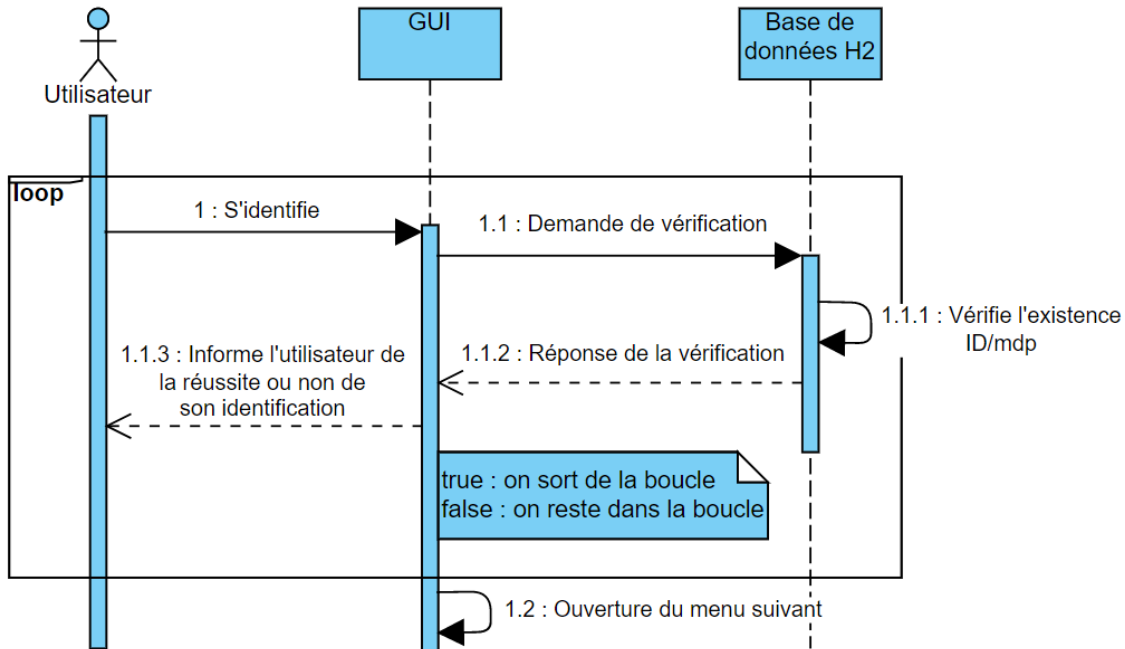
Ce package contient les deux classes de test que nous avons utilisé.



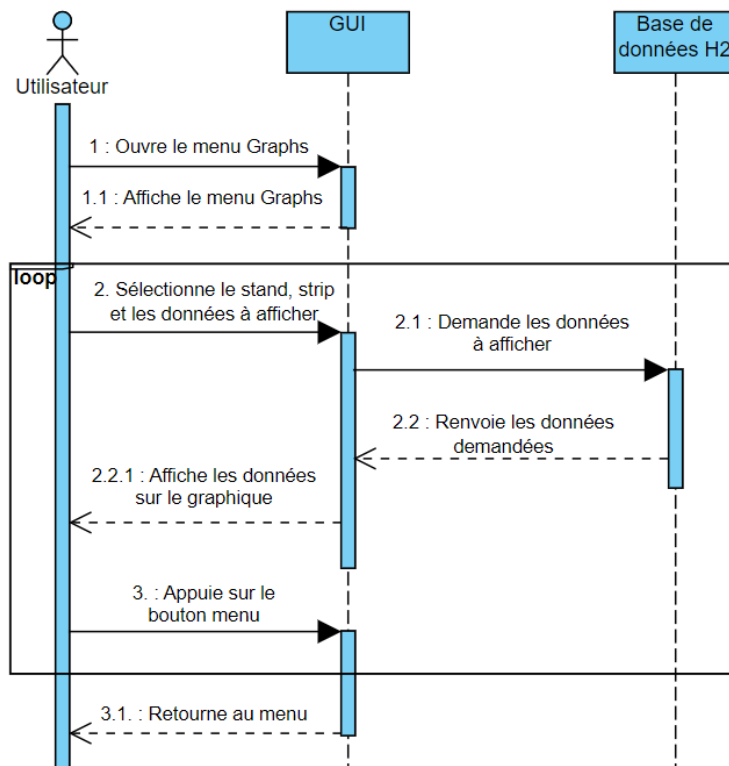
TestOrowanExecution pour tester le bon fonctionnement des fonctions créées pour la création des données de sorties d'Orowan.exe et **TestH2Database** utilisée pour tester la création de l'instance de la base de données.

IV. Les fonctionnalités

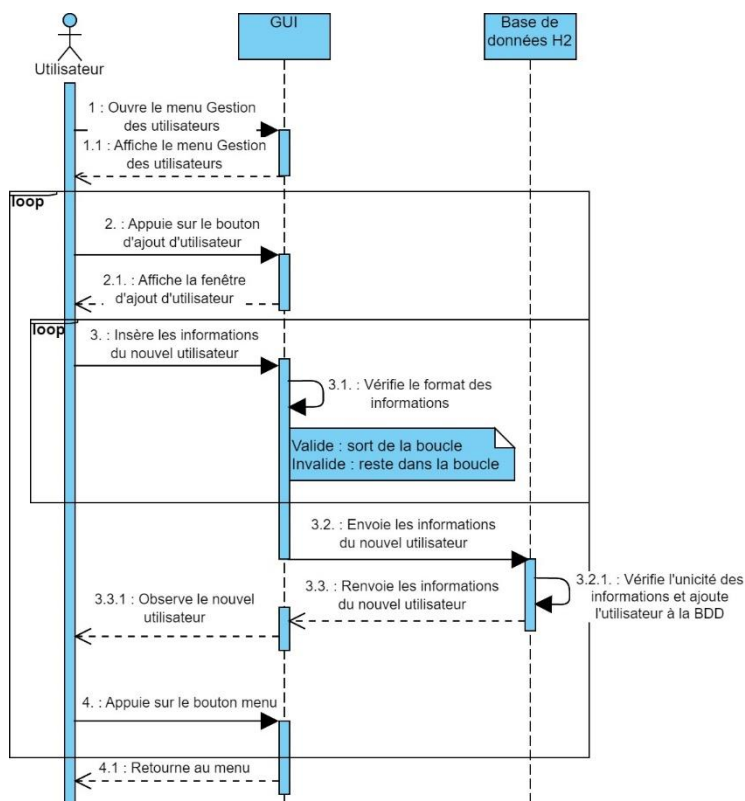
Authentification d'un utilisateur



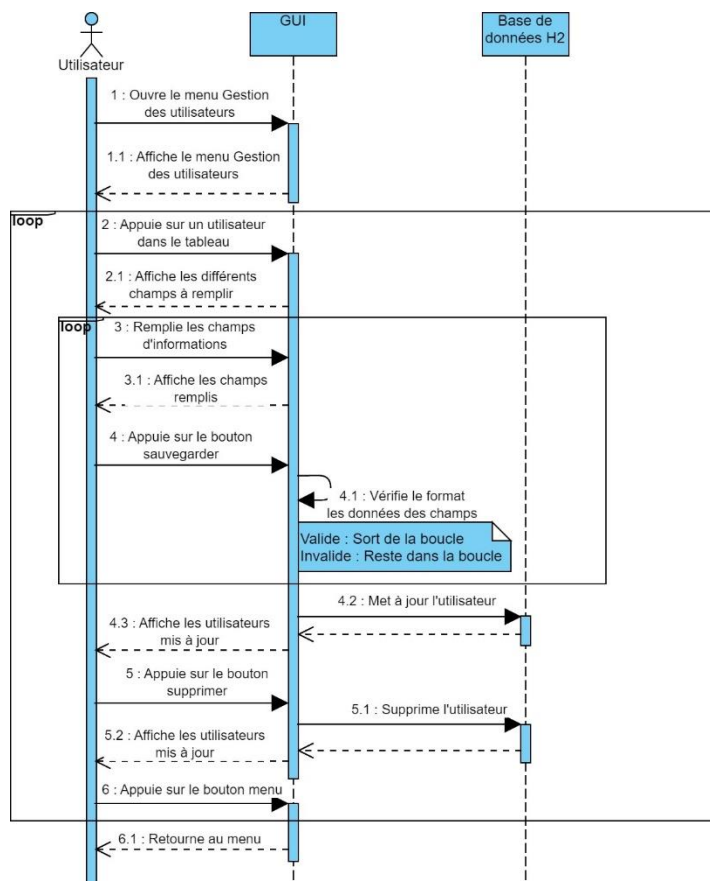
Affichage des graphiques



Ajout d'un utilisateur



Modification d'un utilisateur



V. Le modèle conceptuel de données

Le package dataClasses vu précédemment contient les 4 classes des 4 type de données manipulées par Orowan et l'application.

Elles ont été conçues pour récupérer sous forme d'objet les données des lignes des fichiers texte qui contiennent les données brute, d'entrée de Orowan.exe et de sortie.

ProcessedOutputData elle, a été conçue pour stocker et calculer les données moyennes qui sont affichées sur les graphiques de l'application.

RawData	OrowanOutputData	OrowanInputData
Lp int	rollSpeed double	rollSpeed double
MatID int	stand String	cas int
stand String	matId int	He double
xTime double	casId int	Hs double
xLoc double	errors String	Te double
EnThick double	offsetYield double	Ts double
ExThick double	friction double	diam_WR double
EnTens double	rollingTorque double	WRyoung double
ExTens double	sigmaMoy double	offset_ini double
RollForce double	sigmaIni double	mu_ini double
FSlip double	sigmaOut double	Force double
Diameter double	sigmaMax double	G double
RolledLengthForWorkRolls double	forceError double	
youngModulus double	slipError double	
BackupRollDia double	hasConverged boolean	
RolledLengthForBackupRolls double		
mu double		
torque double		
averageSigma double		
inputError double		
LubWFIUp double		
LubWFIlo double		
LubOilFIUp double		
LubOilFIlo double		
WorkRollSpeed double		

ProcessedOutputData
xTimeMS float
rollingSpeed double
sigma double
friction double
stand String
stripID int