

Projet Langages Web

Le Petit Scientifique

19 décembre 2016

Date : 19 décembre 2016

Rédigé par : Amine BOUAZIZ
Jérémy TEVENIN

Rendu à : M. MALLET

Table des matières

1 Objet du projet

Le projet **UML Reverse** vise à reprendre le projet des anciens étudiants du master 1 en ajoutant et en améliorant des fonctionnalités de l'application UML afin de satisfaire les besoins des futurs étudiants. Le projet est décomposé en deux parties :

1.1 Reverse

Le programme produit un diagramme UML à partir d'un code source. Pour cette version, seul le code Java est concerné.

Les diagrammes créés pourront être sauvegardés en XMI ou Plant UML et pourront être modifiés grâce au générateur de diagramme.

1.2 Générateur de diagramme

Le générateur de diagramme permet à l'utilisateur de construire son propre diagramme UML graphiquement.

Il pourra soit en créer un nouveau, soit en charger un à partir d'un projet existant ou d'un fichier PlantUML ou XMI.

L'utilisateur pourra sauvegarder ses diagrammes avec ses paramètres, ou bien les exporter.

2 Documents applicables de référence

2.1 Expression du besoin du projet UML Reverse

L'utilisation des diagrammes UML fait partie des outils de tout développeur informatique. Les applications (ou plugins) de bon niveau permettant d'utiliser efficacement ces diagrammes sont limités dans le domaine du libre, et ne disposent pas de toutes les fonctionnalités utiles.

Remarque : Les logiciels disponibles en libre ne sont pas pérennes. En effet, souvent rachetés par des entreprises, les allers-retours entre versions libres et propriétaires sont très fréquents. Quant aux versions restées libres, elles ont pour défaut leur manque de fonctionnalités, l'ergonomie très discutable et le manque de documentation fiable.

Objectifs

Développer une application ergonomique permettant d'effectuer rapidement un développement informatique intégrant des diagrammes UML. La première phase du projet a été développée l'année dernière, et doit être complétée en exploitant les éléments déjà disponibles. Cette contrainte est fréquemment rencontrée en entreprise où il est demandé de faire évoluer une application existante.

Objectifs imposés

Afin de limiter le périmètre de l'application et d'en faire un outil exploitable, les contraintes imposées sont les suivantes :

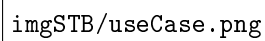
- Améliorer le code existant (Correction de bugs).
- Exporter/Importer au format XML.
- Générer des diagrammes de séquence, d'état et de package. Une interface graphique doit permettre de définir les paramètres graphiques :
 - positionnement des éléments.
 - sélection des éléments affichés/cachés.
- L'application doit permettre le *reverse engineering*, c'est-à-dire l'extraction du diagramme de classe à partir du code source en Java d'une application. Le résultat sera stocké au format XML.
- POC : Réaliser un reverse engineering en exploitant l'introspection JAVA.

3 Lexique

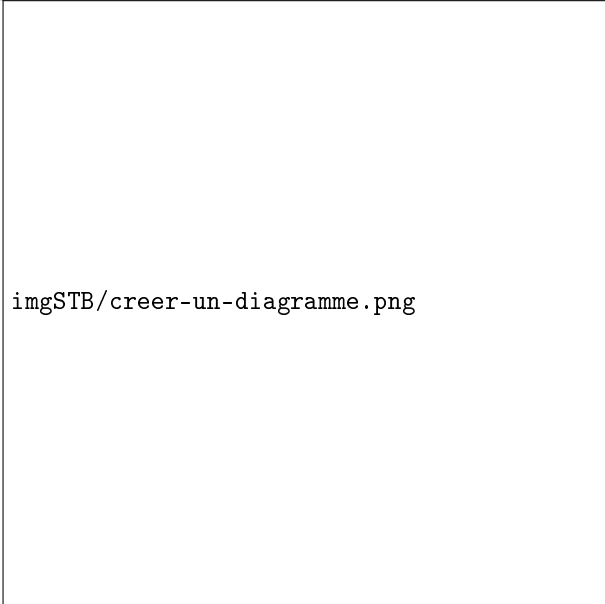
Se référer au document Terminologie.

4 Fonctionnalités


4.1 Diagramme de cas d'utilisation




4.2 Création de diagramme

Nom :	Création de diagramme
Acteurs concernés	Utilisateurs
Description	Crée un diagramme UML
Pré-Conditions	Un projet doit être créé
Evénements déclenchants	
Conditions d'arrêt	Le diagramme a été créé
Description du flot d'événements principal :	
 The diagram area contains a large empty rectangular box, indicating that the flow of events is not yet defined or is represented by a placeholder image. imgSTB/creer-un-diagramme.png	

4.3 Génération de diagramme en reverse

Nom :	Génération de diagramme en reverse
Acteurs concernés	Utilisateurs
Description	Générer un diagramme en reverse
Pré-Conditions	Un diagramme doit être créé
Evénements déclenchants	
Conditions d'arrêt	Le diagramme en reverse a été généré
Description du flot d'événements principal :	
 <p>imgSTB/generer-un-diagramme-reverse.png</p>	

4.4 Exportation/Importation de diagramme

Nom :	Exportation/Importation de diagramme
Acteurs concernés	Utilisateurs
Description	Exporter/Importer un diagramme UML
Pré-Conditions	Un diagramme UML doit être créé
Événements déclenchants	
Conditions d'arrêt	L'exportation/Importation a été réalisé
Description du flot d'événements principal :	
 <p>imgSTB/exporter-un-diagramme.png</p>	


4.5 Exigences fonctionnelles de l'IHM

Cas d'utilisation de l'IHM		
Identification	Description	Priorité
IHM_10	Changer le point d'apparition d'une entité	Indispensable
IHM_20	Pouvoir mettre plusieurs flèches entre deux classes	Indispensable
IHM_30	Exporter un diagramme au format XMI	Indispensable
IHM_40	Importer/charger un diagramme UML écrit en XMI compatible dans un projet	Indispensable
IHM_50	Valider un diagramme UML grâce à un validateur	Optionnelle


4.6 Exigences fonctionnelles des générations de diagrammes

Cas d'utilisation des diagrammes		
Identification	Description	Priorité
DIA_10	Créer un nouveau diagramme de séquence	Indispensable
DIA_20	Créer un nouveau diagramme de paquetages	Indispensable
DIA_30	Créer un nouveau diagramme d'états	Indispensable
DIA_40	Générer un diagramme de séquence par <i>reverse engineering</i>	Indispensable
DIA_50	Réaliser une étude sur l'introspection java	Indispensable

4.7 Cas d'utilisation de modification d'un diagramme de séquence




imgSTB/diagramme-de-sequence.png



imgSTB/diagramme-de-sequence(suite).png

4.7.1 Schéma fonctionnel d'utilisation



imgSTB/E-R-Sequence.png

4.7.2 Précisions

- Un acteur contient un nom.
- Un objet contient un nom (un unique mot) et possède une ligne de vie.
- Une note contient un texte.
- Une relation est une flèche.
- Un cadre d'itération contient un texte et regroupe des messages.
- Un cadre else suit immédiatement un cadre alt. Sans alt, il ne peut y avoir de else.
- Un message est une flèche.
- Déplacer un objet déplace aussi tout ce qui lui est lié, c'est-à-dire les flèches, les cadres d'itération et sa ligne de vie.

4.7.3 Liste des cas d'utilisation supplémentaires

4.7.4 Liste des cas d'utilisation

Modification d'un diagramme de séquence		
Numéro	Description	Priorité
SEQ_10	Créer un acteur. Ce nom est une chaîne de caractères quelconque	Indispensable
SEQ_20	Modifier le nom d'un acteur. Ce nom est une chaîne de caractères quelconque	Indispensable
SEQ_30	Ajouter une référence à un acteur	Optionnelle
SEQ_40	Créer une note	Indispensable
SEQ_50	Modifier le texte d'une note	Indispensable
SEQ_60	Créer un cadre d'itération (alt, for...)	Indispensable
SEQ_70	Modifier les messages contenus dans un cadre d'itération	Indispensable
SEQ_80	Créer une relation entre deux acteurs	Indispensable
SEQ_90	Modifier le texte de la relation	Indispensable
SEQ_100	Déplacer un acteur	Indispensable
SEQ_110	Déplacer une relation	Optionnelle
SEQ_120	Supprimer un acteur	Indispensable
SEQ_130	Supprimer un élément	Indispensable

4.8 Cas d'utilisation de modification d'un diagramme de paquetage

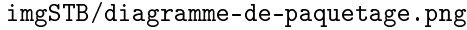
imgSTB/diagramme-de-package.png

UML Reverse

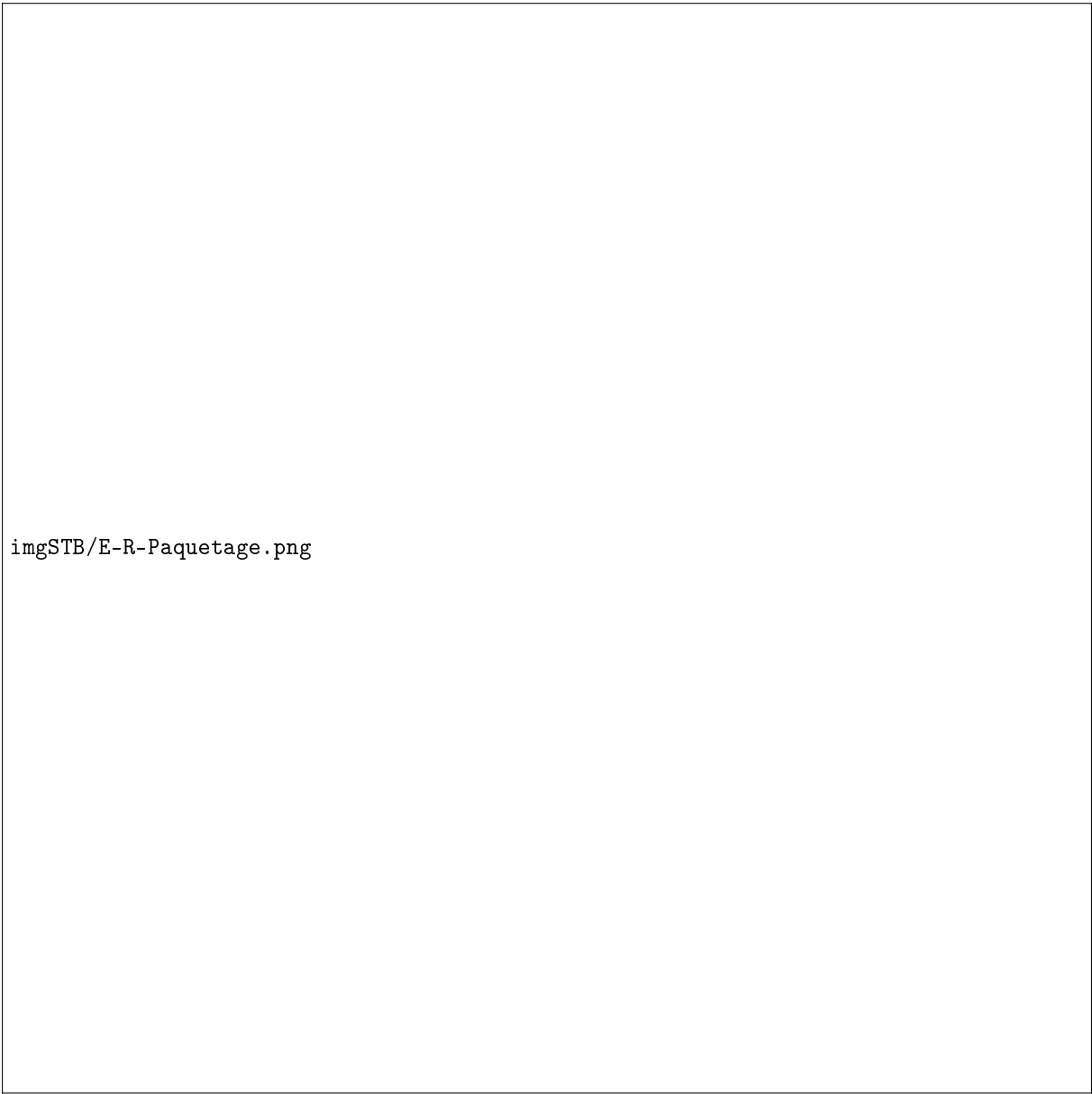
imgSTB/diagramme-de-package(suite).png

UML Reverse

4.8.1 Liste des cas d'utilisation

Modification d'un diagramme de paquetage		
Identification	Description	Priorité
PAQ_10	Créer un paquetage	Indispensable
PAQ_20	Modifier le nom d'un paquetage	Indispensable
PAQ_30	Ajouter un élément à un paquetage	Indispensable
PAQ_40	Modifier le nom des éléments du paquetage	Indispensable
PAQ_50	Modifier la visibilité des éléments du paquetage	Indispensable
PAQ_60	Supprimer un élément d'un paquetage	Indispensable
PAQ_70	Créer une note	Indispensable
PAQ_80	Modifier le texte d'une note	Indispensable
PAQ_90	Redimensionner un paquetage	Optionnelle
PAQ_100	Créer une relation entre deux paquetages	Indispensable
PAQ_110	Supprimer une relation entre deux paquetages	Indispensable
PAQ_120	Supprimer un paquetage	Indispensable
Exemple de diagramme de paquetage :		
		

4.8.2 Schéma fonctionnel d'utilisation



imgSTB/E-R-Paquetage.png

4.8.3 Précisions

- Les relations se font entre les paquetages

5 Exigences

5.1 Exigences fonctionnelles

Identifiant	Description
EXF_20	L'interface graphique doit être fonctionnelle, pratique et adaptée aux besoins.
EXF_30	Le <i>reverse</i> permet de générer un diagramme à partir d'un fichier compatible Java 7.
EXF_40	L'application implémente tous les éléments d'un diagramme UML2 pour les diagrammes compatibles.
EXF_50	À tout moment de l'édition d'un diagramme non vide, un fichier PlantUML ou XMI compilable peut être généré à partir du diagramme.
EXF_60	L'utilisateur ne peut pas avoir deux entités du même nom dans un diagramme.

5.2 Exigences de qualités

Identifiant	Description
EXF_70	Le code de l'application doit être modulaire.

5.3 Exigences techniques

Identifiant	Description
EXF_10	L'application est codée en Java.
EXF_80	L'application doit fonctionner sur les ordinateurs de l'Université.
EXF_90	L'application finale doit respecter la licence open source.