



Equipe de projet L2S1

Projet TraceX

Conception Générale

Version du document :	B.05
Date de document :	30/03/2025
Soumis le :	30/03/2025
Auteurs :	ALLAHOUM Abdelmalek Said, HUANG Maxime, KIM Léa, ZHENG Jacques
Type de diffusion :	Document électronique (.pdf)
Confidentialité :	Réservé aux étudiants UFR Maths-Info de l'Université Paris Cité

Table des matières

1.	Introduction.....	2
1.1	Contexte du Projet	2
1.2	Objectif du Document.....	2
1.3	Références.....	2
2.	Architecture Générale du Système	2
2.1	Technologies et Outils Utilisés.....	2
2.2	Interaction entre les Modules	2
2.3	Fonctionnement Général.....	3
3.	Description des Modules.....	4
3.1	Module d'Importation	5
3.2	Module de structure de fichiers.....	5
3.3	Module d'analyse et de traçabilité.....	6
3.4	Module d'extraction des styles	7
3.5	Module d'extraction d'exigences	7
3.6	Module de Visualisation	8
3.7	Module de Génération de Rapport.....	9
4.	Conception de l'Interface Utilisateur	9
5.	Contraintes et Hypothèses	12
5.1.	Contraintes Techniques	12
5.2.	Contraintes de Performance	12
5.2.1.	Traitement rapide des fichiers volumineux	12
5.2.2.	Optimisation des algorithmes d'analyse et de visualisation	13
6.	Plan de Validation	13
6.1.	Tests Unitaires	13
6.2.	Tests d'Intégration	13
6.3.	Recette Fonctionnelle.....	14
7.	Index	14
8.	Références	15

1. Introduction

Ce document présente la conception générale du projet de traçabilité documentaire. Il décrit l'architecture du système, les principaux modules développés ainsi que les contraintes à respecter.

1.1 Contexte du Projet

Le projet vise à automatiser l'analyse documentaire en assurant la traçabilité des exigences définies dans des documents référentiels.

1.2 Objectif du Document

- Définir l'architecture et la structure du système,
- Décrire les principaux modules et leurs interactions,
- Établir les contraintes techniques et fonctionnelles,
- Présenter le plan de validation du projet.

1.3 Références

- Cahier des charges [C.02]
- Cahier de recettes [R.02]
- Manuel d'utilisation [U.02]
- Conception détaillé [D.02]

2. Architecture Générale du Système

L'application a été développée en C++ avec Qt pour l'interface graphique et Pugixml pour l'analyse des fichiers XML.

2.1 Technologies et Outils Utilisés

- **IDE** : Qt creator
- Langage de programmation : C++
- **Qt** : utilisé pour la conception de l'interface utilisateur (fenêtres, menus, interactions graphiques).
- **Pugixml** : bibliothèque optimisée pour le parsing XML, permettant d'extraire efficacement les informations des documents XML.
- **SVN** : Gestion de versions.

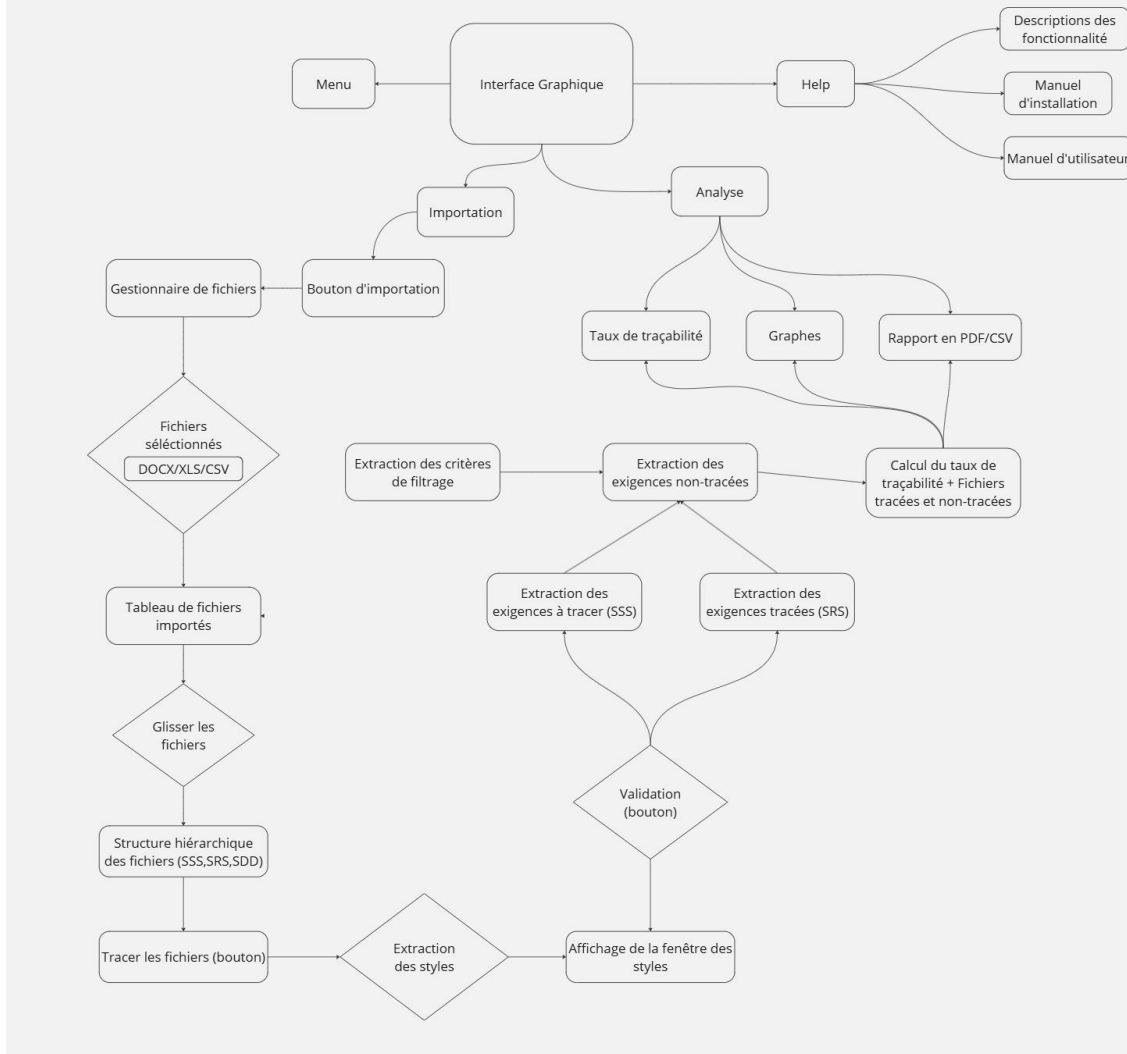
2.2 Interaction entre les Modules

Le système est composé de plusieurs modules :

- **Module de gestion de fichiers** : Création de fichiers, extraction dossiers zip et suppression des fichiers inutiles.

- **Module d'importation** : lecture et extraction des données des documents DOCX, XLSX et CSV.
- **Module d'analyse et de traçabilité** : identification des exigences et établissement des liens entre les fichiers.
- **Module de visualisation** : affichage des liens sous forme de graphe.
- **Module de génération de rapports** : Génération d'un rapport format CSV contenant les exigences non tracées.

Schéma global



2.3 Fonctionnement Général

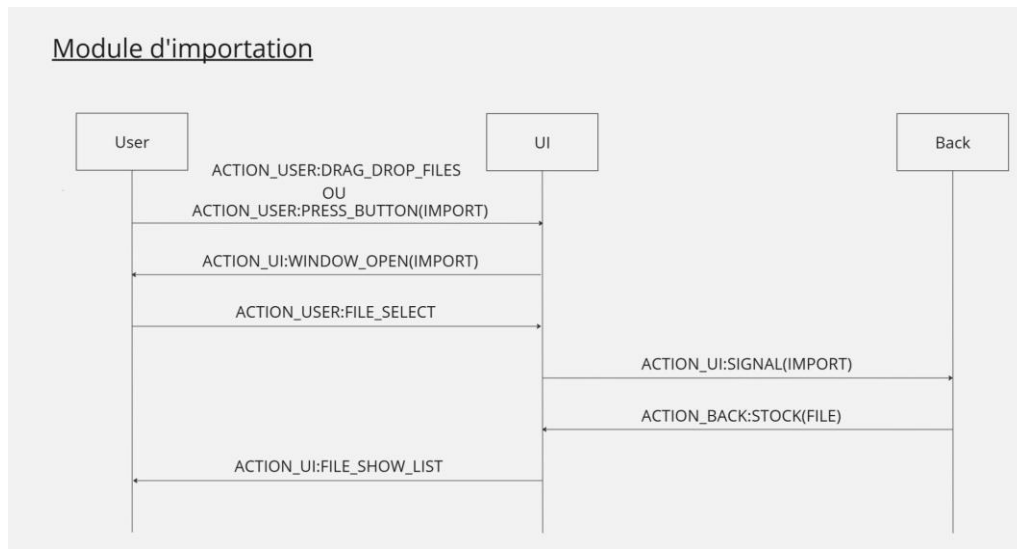
- L'utilisateur importe les fichiers (WORD, EXCEL ou CSV).
- Le module d'analyse extrait les données (les exigences) en parcourant les fichiers XML pour les documents aux formats EXCEL et WORD et en extrayant directement le contenu des documents format CSV.
- Le graphe des relations est généré et affiché à l'utilisateur.
- Des filtres peuvent être appliqués pour ajuster l'affichage en fonction des besoins.

3. Description des Modules

Type	Définition
ACTION_USER	Action de l'utilisateur
ACTION_UI	Action de l'UI de l'application
ACTION_BACK	Action du backend de l'application
DRAG_DROP_FILES	Glisser/déposer les fichiers
PRESS_BUTTON()	Appui sur un bouton
WINDOW_OPEN()	Ouvre une fenêtre
FILE_SELECT	Sélection de fichier
SIGNAL()	Lecture d'un signal sur l'UI par le Back
STOCK()	Stockage les données récupérées
FILE_SHOW_LIST	Affichage des fichiers sous forme de liste (tableau)
FILE_SLIDE	Glisser/déposer et ordonner les fichiers selon leur type
FILE_SHOW_TREE	Affichage d'arborescence des fichiers
EXTRACT()	Extraction des informations du document
CALCULATE_TRACA	Calcul du taux de traçabilité
GRAPH_SHOW	Affichage du graphe
GRAPH_SHOW_TRACA	Affichage de traçabilité sur le graphe
USE()	Utilisation de données récupérées auparavant
RAPPORT_SHOW	Affichage du rapport

miro

3.1 Module d'Importation



Ce module permet à l'utilisateur d'importer des fichiers (docx,xlsx,csv) nécessaires à la traçabilité des documents.

Une fois les fichiers importés, ils apparaissent dans un tableau qui affiche des informations utiles, telles que :

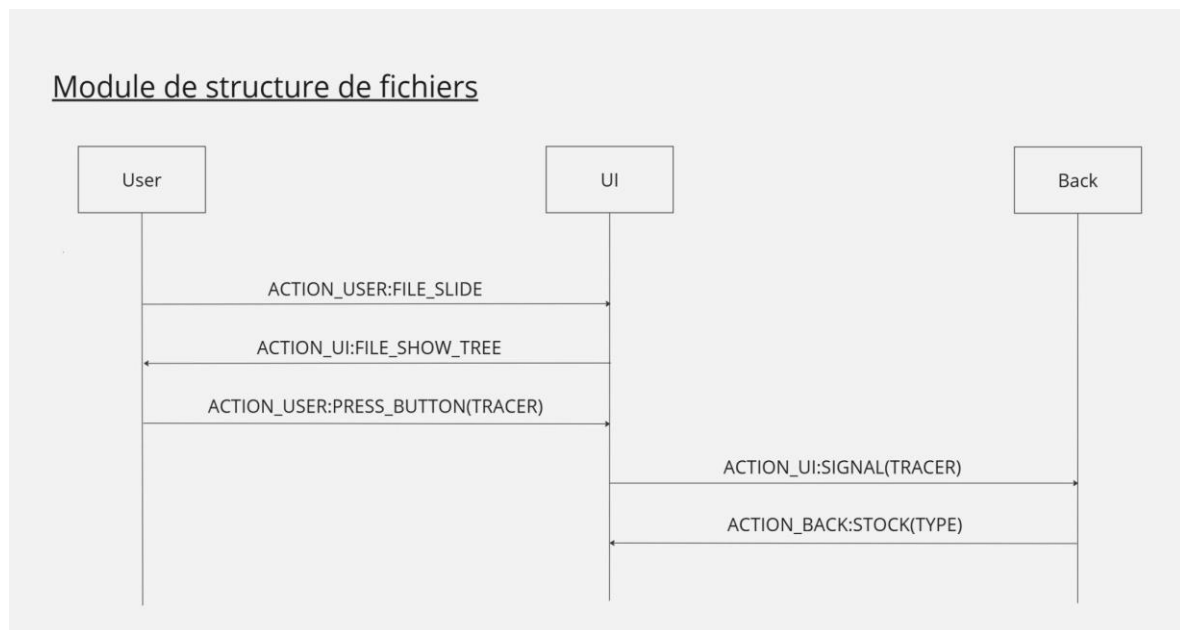
- Le **nom** + extension du fichier.
- Le **chemin d'accès** du fichier.
- Une **icône** de "poubelle" pour supprimer le fichier.

Les fonctionnalités principales du bouton d'importation sont la suivantes:

- **Importation multiple** : L'utilisateur aura la possibilité de sélectionner plusieurs fichiers en une seule fois ou bien de faire un glisser-déposer depuis son gestionnaire des fichiers.
- **Affichage dynamique des fichiers importés** : chaque fichier importé est ajouté au tableau en temps réel.
- **Préparation pour la hiérarchisation** : les fichiers présents dans le tableau permettent de les filtrer selon leur type ultérieurement dans l'arbre hiérarchique grâce à un système de glisser-déposer.

Ce module est essentiel pour rassembler tous les documents nécessaires à l'analyse et à la traçabilité des exigences.

3.2 Module de structure de fichiers



Ce module permet à l'utilisateur de visualiser et d'organiser les fichiers importés sous forme d'une structure hiérarchique, reflétant les relations père/fils entre les différentes exigences (par exemple : SSS → SRS → SDD). Une fois les fichiers chargés, l'utilisateur peut interagir avec eux dans une interface graphique qui représente leur organisation sous forme d'arbre.

Les interactions principales de ce module sont les suivantes :

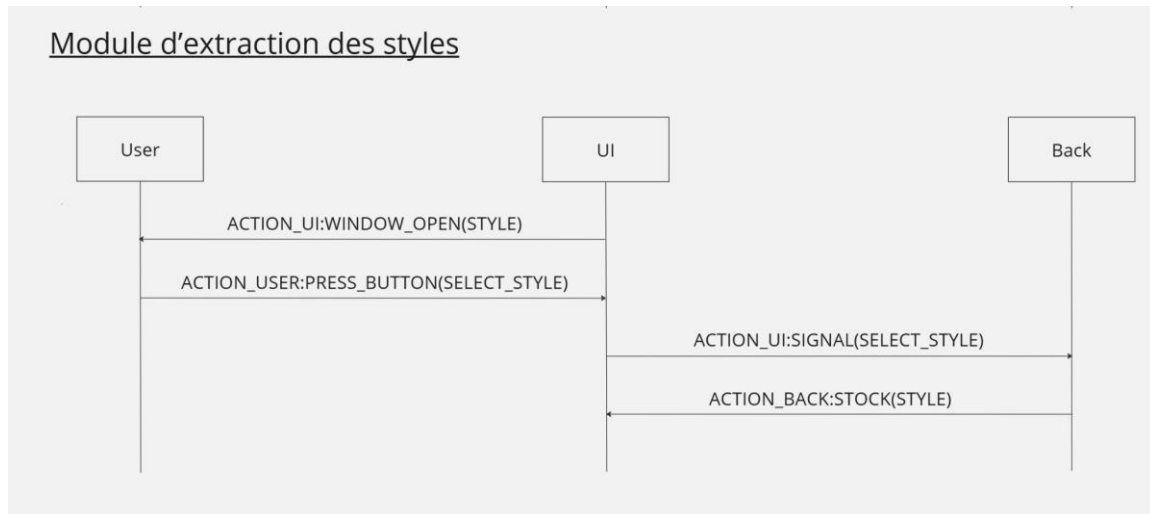
- **Glissement de fichier** : L'utilisateur dépose un fichier dans l'arbre de données, permettant son ajout dans la structure.
- **Affichage arborescent automatique** : L'interface analyse l'endroit où le fichier à été déposé afin d'y afficher.
- **Création de relations hiérarchiques** : L'utilisateur peut créer manuellement ou automatiquement des liens entre les fichiers selon leur type (ex. un fichier SRS enfant d'un SSS).
- **Déclenchement du traçage** : Une fois les liens définis, un bouton dédié permet d'envoyer un signal à la couche back-end pour stocker la hiérarchie et préparer le graphe de traçabilité.

Ce module joue un rôle central dans la préparation de l'analyse, car il établit la base structurelle sur laquelle se construisent les relations de traçabilité entre exigences.

3.3 Module d'analyse et de traçabilité

Ce module analyse le contenu des documents importés par l'utilisateur afin d'établir des liens de traçabilité entre les exigences qu'ils contiennent.

3.4 Module d'extraction des styles



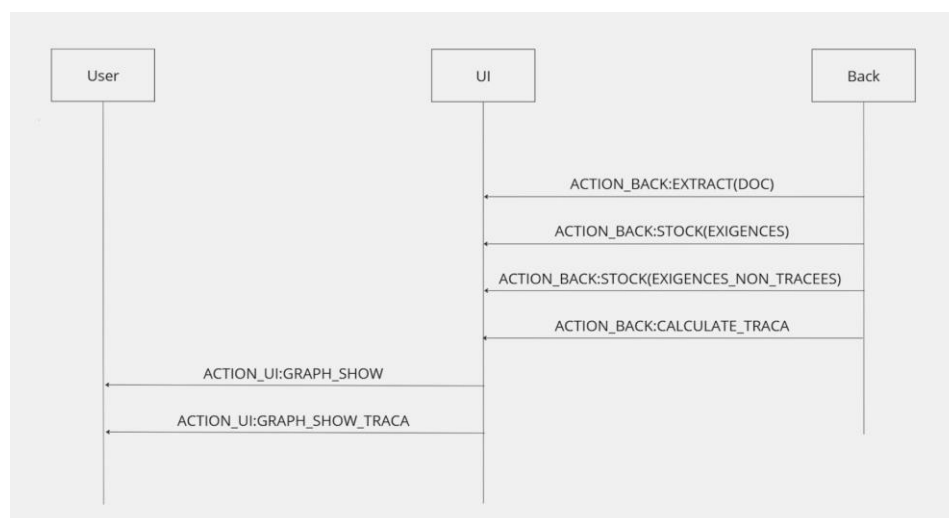
- Fonctionnement du module d'extraction des styles

Ce module est chargé d'extraire les styles des fichiers DOCX importés et de les afficher pour permettre à l'utilisateur de sélectionner celui qui correspond au style des exigences.

Après l'import des documents, les fichiers de styles "styles.xml" pour chaque type de document (SSS, SRS et SDD) au format DOCX seront analysés afin d'extraire les styles utilisés. Ces styles seront ensuite présentés à l'utilisateur sous forme de cases à cocher pour qu'il puisse en sélectionner un.

Après avoir sélectionné un style pour chaque type de document (SSS, SRS, SDD), ce dernier sera attribué à l'attribut statique "**style**" de chaque classe.

3.5 Module d'extraction d'exigences



- Fonctionnement du module d'extraction d'exigences

— Document au format DOCX :

Ce module permet d'extraire les exigences des documents **SSS, SRS, SDD** en identifiant le style utilisé pour les définir.

Les fichiers XML seront analysés afin d'extraire les exigences écrites selon ce style pour ensuite les stocker dans l'attribut **"exigences"** de la classe qui convient parmi les classes SSS, SRS ou SDD. Pour les documents "SRS", des données supplémentaires, telles que la cible, les clients pour lesquels l'exigence est nécessaire et le statut (développé ou non développé), seront également extraites et stockées dans les attributs appropriés.

— Document au format XLSX :

Ce module permet aussi d'extraire les exigences définies dans un document au format Xlsx.

Les fichiers XLSX sont des archives compressées au format ZIP contenant plusieurs fichiers XML et d'autres ressources. Parmi ces fichiers XML, ce module analysera **sharedStrings.xml**, qui contient le contenu de chaque cellule (les exigences), afin d'en extraire les données nécessaires grâce à une fonction adaptée à ce format.

— Document au format CSV :

Ce module permet également d'extraire les exigences définies dans un document au format CSV.

Les fichiers CSV seront traités comme des fichiers texte simples, où chaque ligne correspond à une exigence. Le contenu de chaque ligne sera lu et les exigences ainsi extraites seront stockées dans l'attribut **"exigences"**.

3.6 Module de Visualisation

— Génération du graphe :

Après que l'analyse des documents soit faite, la génération du graphe se fera sous forme de nœuds représentant chaque document SSS/SRS/SDD reliés par des arêtes. Chaque arête contiendra le taux de traçabilité entre les deux nœuds.

— Filtrage des liens :

Les exigences seront filtrées selon certains critères (cibles, statut (développé ou pas), client pour lesquels l'exigence est nécessaire, expression régulière) extraits d'un fichier SRS. L'option de filtrage sera proposée sous forme de cases à cocher permettant à l'utilisateur de pouvoir en sélectionner une ou plusieurs.

— Interaction utilisateur :

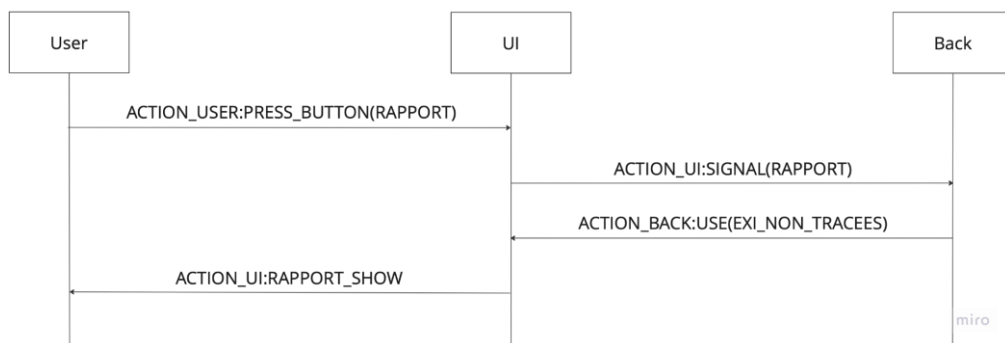
Le zoom, sélection et exploration du graphe. Un bouton “+” et bouton “-” qui permettra à l'utilisateur de zoomer/dé-zoomer le graphe en entier.

— Génération des statistiques :

Représentation des statistiques sur le taux de traçabilité inter-documentaire (SSS, SRS et SDD). La représentation des statistiques sera faite sous forme d'un histogramme (diagramme en bâtons). L'axe d'abscisse représente le fichier tracé et à tracer. L'axe des ordonnées représente l'équation nombre d'exigences tracées/nombre totale d'exigence) *100.

3.7 Module de Génération de Rapport

Module de génération de rapport

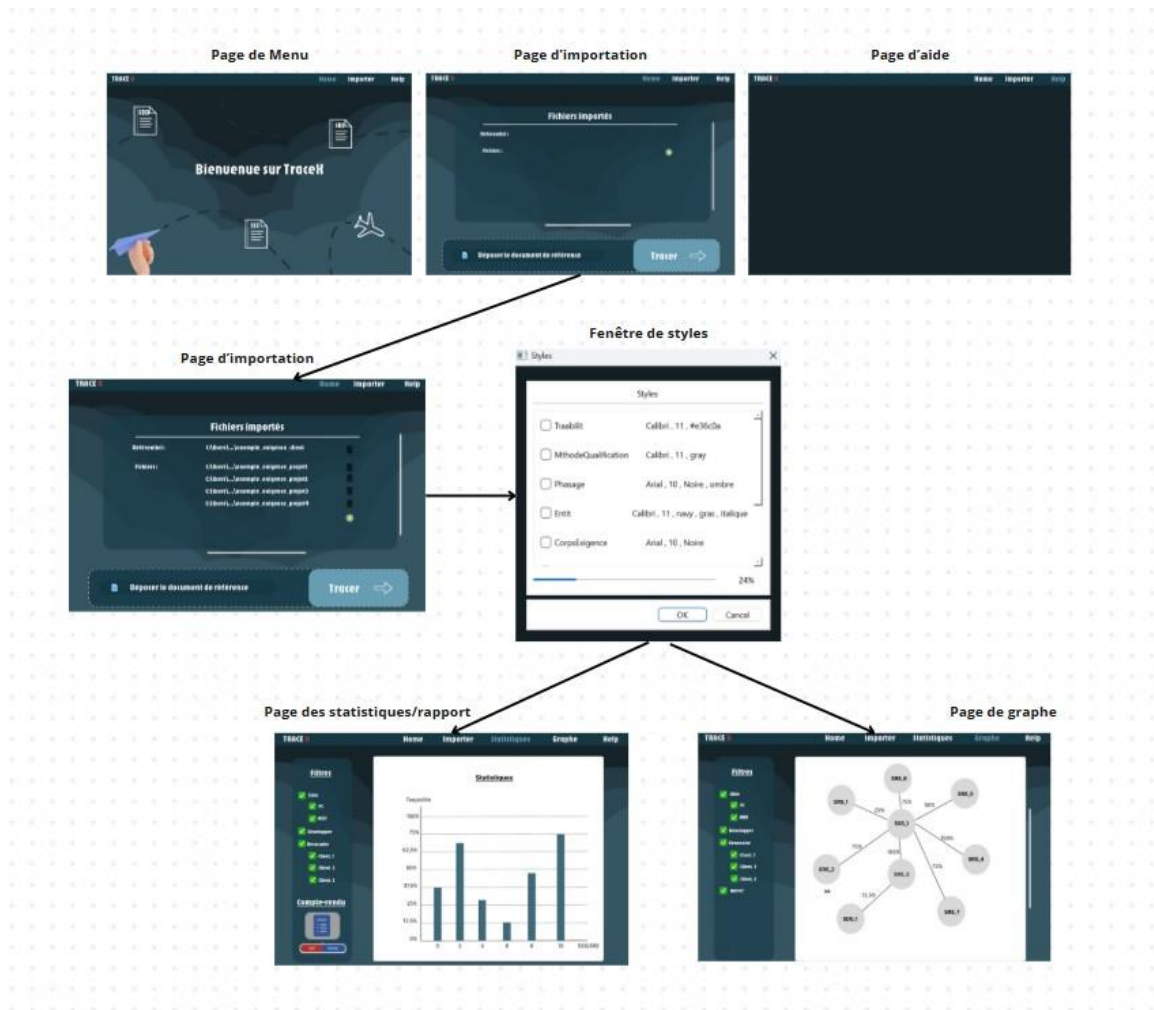


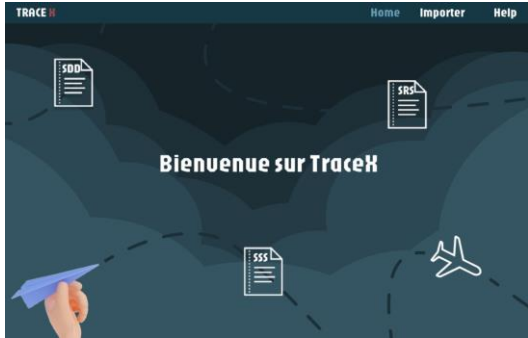

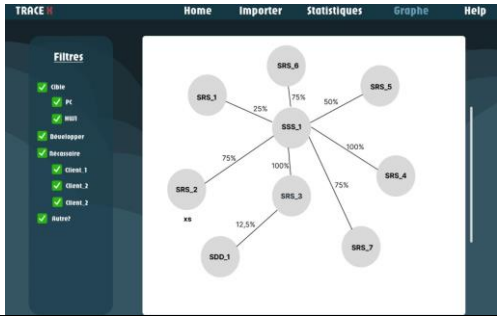
- Formats d'exportation : Après l'analyse des documents et l'identification des exigences non tracées, un rapport récapitulatif de ces exigences sera généré au format CSV ou PDF.

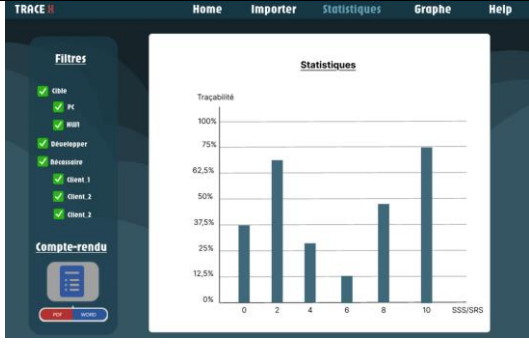
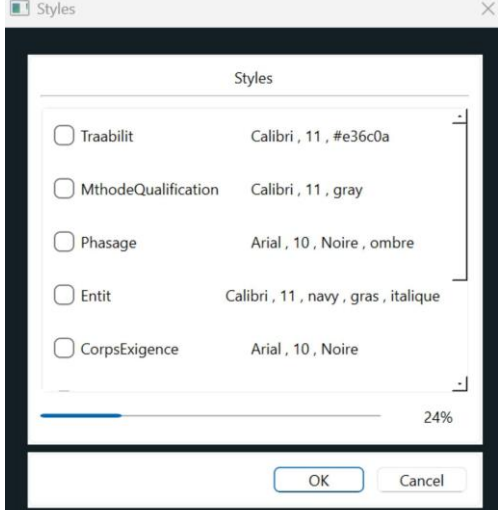
4. Conception de l'Interface Utilisateur

L'interface utilisateur sera développée avec Qt et offrira les fonctionnalités suivantes :

- Importation et gestion des fichiers (spécialement en format DOCX, XLSX et CSV),
- Affichage du graphe de traçabilité.
- Filtrage des exigences.
- Navigation intuitive et ergonomique.



Maquette	Description
	<p>La page d'accueil est la première page lors de l'ouverture de l'application. Il contient:</p> <ul style="list-style-type: none"> - Une barre de menu
	<p>La page d'importation permet à l'utilisateur d'importer et de trier les fichiers. Il contient :</p> <ul style="list-style-type: none"> - Un bouton d'importation - Un tableau contenant l'ensemble des fichiers importés - Un arbre affichant les fichiers organisés selon le tri de l'utilisateur - Un bouton "tracer" qui permet d'extraire les styles - Une barre de menu
	<p>La page de graphe permet à l'utilisateur de visualiser les liens de traçabilité entre les documents analysés. Il contient:</p> <ul style="list-style-type: none"> - Un graphe (noeuds/arrêtes) - Un Menu de filtres (selon le cible, nécessaire, développé ou non) - Une barre de menu
	<p>La page de statistique permet à l'utilisateur de visualiser le pourcentage de liens de traçabilité entre les fichiers. Il contient:</p> <ul style="list-style-type: none"> - Un histogramme (axe des abscisses = fichier/fichier et axe des ordonnées = pourcentage de traçabilité)

	<ul style="list-style-type: none"> - Une barre de menu
	<p>La fenêtre de styles permet à l'utilisateur de trier les styles d'exigences. Il contient:</p> <ul style="list-style-type: none"> - Une liste de case à cocher (checkbox) contenant chaque styles d'exigences - Une barre de menu

5. Contraintes et Hypothèses

5.1. Contraintes Techniques

- Utilisation de Qt pour l'interface graphique.
- Compatibilité multiplateforme (Windows, Linux, MacOS).

5.2. Contraintes de Performance

5.2.1. Traitement rapide des fichiers volumineux

Quand on travaille avec de gros fichiers Word, Excel ou CSV, le temps de traitement peut vite devenir un frein. Pour éviter ça :

- **Lecture intelligente** : Plutôt que de charger tout le fichier en mémoire d'un coup, on lit par morceaux (ligne par ligne pour CSV, ou directement les sections utiles dans Word/Excel). Cela permet de démarrer l'analyse plus vite sans attendre la fin du chargement complet.
- **Analyse ciblée** : On n'extrait que les données nécessaires (par

exemple, les exigences) sans parcourir les parties inutiles du fichier. Pour un fichier Word, inutile de charger tout le style ou les images si on ne veut que le texte dans document.xml.

- **Économie de ressources** : En libérant rapidement les données qui ne servent plus et en évitant les copies inutiles, on garde la mémoire légère, même avec des fichiers massifs.

5.2.2. Optimisation des algorithmes d'analyse et de visualisation

Une fois les données extraites, il faut les analyser et les afficher sans ralentissement :

- **Tri et recherche rapides** : Plutôt que de comparer chaque exigence une par une, on organise les données dès le début (par exemple en listes triées ou en tables de hachage) pour les retrouver instantanément.
- **Mise à jour progressive** : Si l'utilisateur ajoute ou modifie un fichier, on ne relance pas toute l'analyse depuis zéro. Seules les nouvelles données sont analysées et intégrées au reste.

6. Plan de Validation

6.1. Tests Unitaires

- Vérification des fonctionnalités de chaque module : L'idée est de tester chaque méthode de chaque module séparément (tests unitaires) pour s'assurer que tout fonctionne correctement avant d'intégrer l'ensemble.

Avec QtTest, on peut :

- Valider le comportement attendu de chaque fonction.
- Détecter rapidement les régressions après une modification.
- Automatiser les tests pour ne pas tout vérifier manuellement à chaque changement

6.2. Tests d'Intégration

- Validation des interactions entre les modules : Vérifier que les données importées dans le module d'importation sont correctement transférées et manipulées par le module d'analyse et de traçabilité.
- Cohérence des données entre l'importation, l'analyse et la visualisation : Chaque module doit envoyer et recevoir les bonnes informations dans le bon format.
- Tester le fonctionnement utilisateur dans son ensemble : Vérifier si les actions d'un module (comme l'importation d'un fichier) provoquent bien les effets attendus dans d'autres modules (comme l'affichage de ces fichiers ou leur analyse).

6.3. Recette Fonctionnelle

- Pour garantir la fiabilité et la robustesse du logiciel, chaque module sera soumis à des tests unitaires à l'aide de la bibliothèque QtTest. Cette approche permettra d'assurer que chaque fonctionnalité fonctionne correctement de manière indépendante avant l'intégration globale.
- Pour valider l'ergonomie et l'efficacité de ton application, il te faut organiser des tests utilisateur bien structurés. Voici une méthode complète en plusieurs étapes :
 - **Scénario 1** : Importer un fichier SSS et un fichier SRS.
 - **Scénario 2** : Glisser déposer et trier les fichiers importés et vérifier l'affichage de l'arborescence des fichiers
 - **Scénario 3** : Sélectionner le style des exigences et lancer l'analyse puis vérifier que les nœuds se créent correctement.
 - **Scénario 4** : Appliquer un filtre sur les exigences et vérifier la mise à jour de l'affichage.

7. Index

Architecture Générale du Système	2
Analyse et Traçabilité	3.2
Conception de l'Interface Utilisateur	4
Contraintes Techniques	5.1
Contraintes de Performance	5.2
Génération du Graphe	3.3
Génération de Rapports	3.4
Importation des Fichiers	3.1
Interaction entre les Modules	2.2
Objectif du Document	1.2
Plan de Validation	6
Références	1.3
Recette Fonctionnelle	6.3

8. Références

— Documentation technique :

- pugixml "Light-weight C++ XML processing library" : <https://pugixml.org/docs/manual.html>
- Qt Documentation "Qt 6.5 C++ GUI framework" : <https://doc.qt.io/qt-6/>

— Tutoriel et formations :

- **Tutoriels sur Qt/Python** (chaînes diverses sur YouTube).
- **LinkedIn learning** : C++ avancé et développement d'applications avec Qt.
- **OpenClassroom** : Programmation orienté objet C++.

— Outils d'assistance :

ChatGPT : Aide pour le choix des bibliothèques selon le besoin