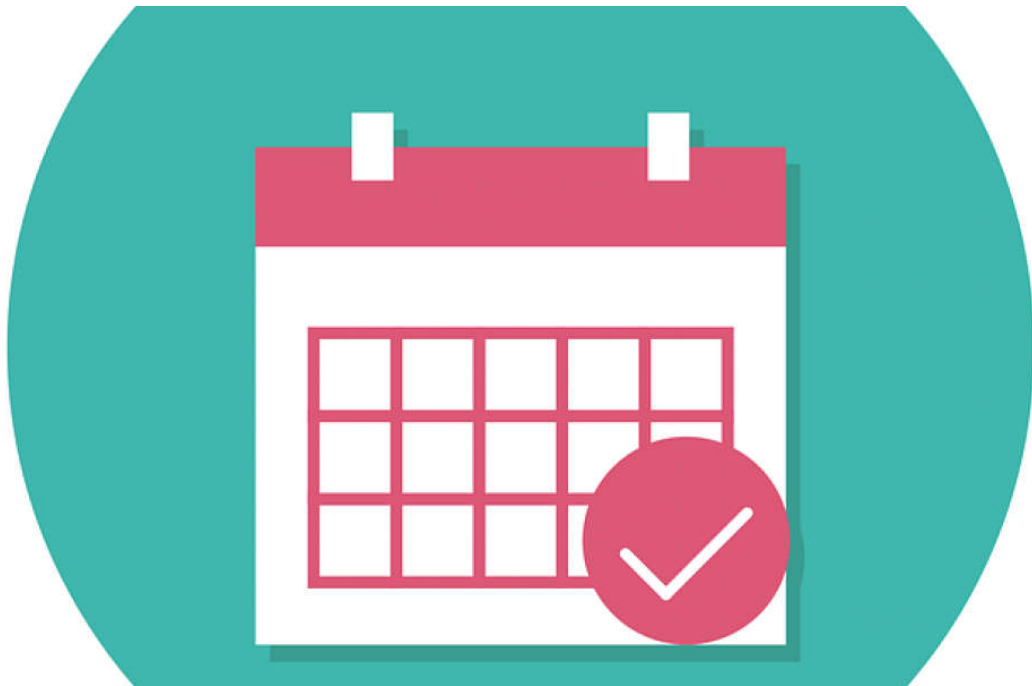


ICalMerge



Candidat : Leonardo Luna
Chef de projet : Jonathan Melly
Expert 1: Ernesto Montemayor
Expert 2: Serge Wenger

Table des matières

1	Analyse préliminaire	4
1.1	Introduction	4
1.2	Objectifs.....	4
1.2.1	Maximum de 10 fichiers source	4
1.2.2	Chargement de fichiers.....	4
1.2.3	Vérification du format automatique	5
1.2.4	Résumé avec le nombre d'événements.....	5
1.2.5	Fusion et Pop-up de fusion	5
1.2.6	Barre de progression	5
1.2.7	Avertissement fichier de destination	5
1.2.8	Vérification de l'intégrité du fichier fusionné.....	5
1.2.9	Utilisation d'un système de versioning	5
1.2.10	Rubrique d'aide.....	6
1.2.11	Explication du format ICal	6
1.2.12	Respect des normes de codage ETML.....	6
2	Analyse / Conception.....	6
2.1	Glossaire	6
2.2	Planification initiale	7
2.3	Conception	8
2.3.1	Maximum de 10 fichiers source	8
2.3.2	Chargement de fichiers.....	9
2.3.3	Vérification du format automatique	9
2.3.4	Résumé avec le nombre d'événements.....	10
2.3.5	Fusion et Pop-up de fusion	10
2.3.6	Barre de progression	12
2.3.7	Vérification de l'intégrité du fichier fusionné.....	13
2.3.8	Avertissement fichier de destination	13
2.3.9	Utilisation d'un système de versioning	15
2.3.10	Rubrique d'aide.....	16
2.3.11	Explication du format ICal	17
2.3.12	Respect des normes ETML	18
2.4	Stratégie de test.....	19
2.5	Risques techniques	22
3	Réalisation.....	23
3.1	Dossier de réalisation	23
3.2	Interface graphique principale	23
3.2.1	Maximum de 10 fichiers source	24
3.2.2	Chargement de fichiers.....	29
3.2.3	Vérification du format automatique	31
3.2.4	Résumé avec le nombre d'événements.....	31
3.2.5	Fusion et pop-up de fusion	32
3.2.6	Barre de progression	33
3.2.7	Avertissement fichier de destination	34
3.2.8	Vérification de l'intégrité du fichier fusionné.....	34

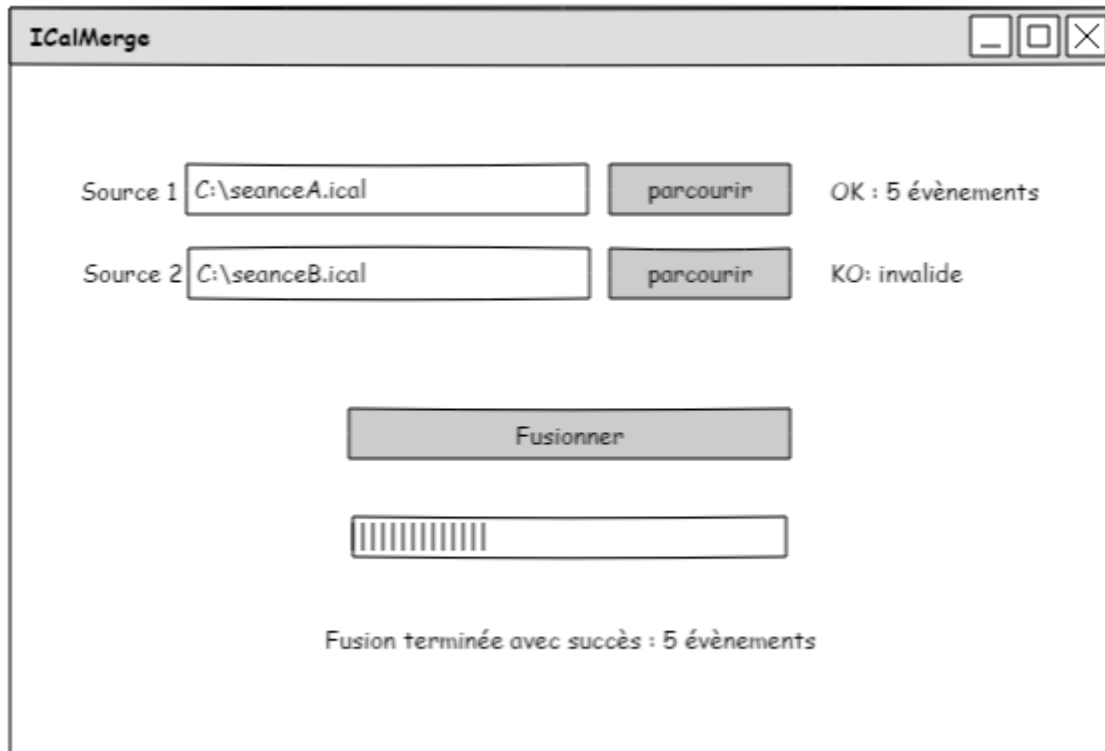
3.2.9	Rubrique d'aide.....	35
3.2.10	Diagramme de classes	36
3.3	Description des tests effectués.....	38
3.4	Erreurs restantes	41
4	Conclusions	41
5	Annexes.....	42
5.1	Sources – Bibliographie.....	42
5.2	Manuel d'utilisation	42
5.3	Table de figures	42
5.4	Liste des documents fournis	43

1 ANALYSE PRÉLIMINAIRE

1.1 Introduction

Selon le cahier des charges : ICalMerge est une solution pour fusionner plusieurs sources de calendrier au format ICAL vers un seul flux. Ceci est particulièrement utile lorsqu'on reçoit plusieurs calendriers pour une conférence par exemple et qu'on veut regrouper cela dans un seul fichier à importer.

Voici le croquis de l'application fourni avec le cahier des charges :



1. Mock-up de l'application

1.2 Objectifs

1.2.1 Maximum de 10 fichiers source

Il sera possible d'ajouter plusieurs sources. Il pourra y en avoir un maximum de dix.

1.2.2 Chargement de fichiers

Il devra être possible d'importer deux chemins de fichiers source. Il sera possible d'ajouter les fichiers source par glisser/déposer, comme demandé dans les points techniques évalués du CDC.

1.2.3 Vérification du format automatique

Lorsque l'utilisateur importera le chemin d'une source, une vérification du format du fichier devra se faire automatiquement.

1.2.4 Résumé avec le nombre d'événements

Si l'importation d'un fichier s'est correctement déroulée, un label affichera le nombre d'événements que contient le fichier choisi. Voir image ci-dessous.



2. Mock-up des contrôles servant à l'importation

1.2.5 Fusion et Pop-up de fusion

Lorsque l'on clique sur le bouton « Fusionner », une pop-up s'ouvrira et permettra à l'utilisateur de choisir la destination du fichier résultant de la fusion.

1.2.6 Barre de progression

Lorsque la fusion s'effectue, une barre de progression se charge en fonction de l'avancement de la fusion. Comme discuté avec M. Melly, le cent pourcent de la barre correspondra au nombre total d'événements à fusionner.

1.2.7 Avertissement fichier de destination

Lorsque la fusion se termine elle affichera un pop-up qui permettra d'exporter le fichier fusionné. Si le fichier de destination existe déjà, il faudra avertir l'utilisateur.

1.2.8 Vérification de l'intégrité du fichier fusionné

Pour vérifier cela, le nombre d'événements créés devra correspondre au nombre d'événements à fusionner.

1.2.9 Utilisation d'un système de versioning

Le projet devra être sauvegardé sur une plateforme de versioning

1.2.10 Rubrique d'aide

Lorsque l'utilisateur pressera sur la touche F1 de son clavier, une page d'aide devra apparaître avec des instructions sur l'utilisation de l'application.

1.2.11 Explication du format ICal

Une explication du format ICal sera requise dans le rapport de projet.

1.2.12 Respect des normes de codage ETML

Le développeur devra suivre les normes de codages de l'ETML.

2 ANALYSE / CONCEPTION

2.1 Glossaire

Git : Logiciel de gestion de versions.

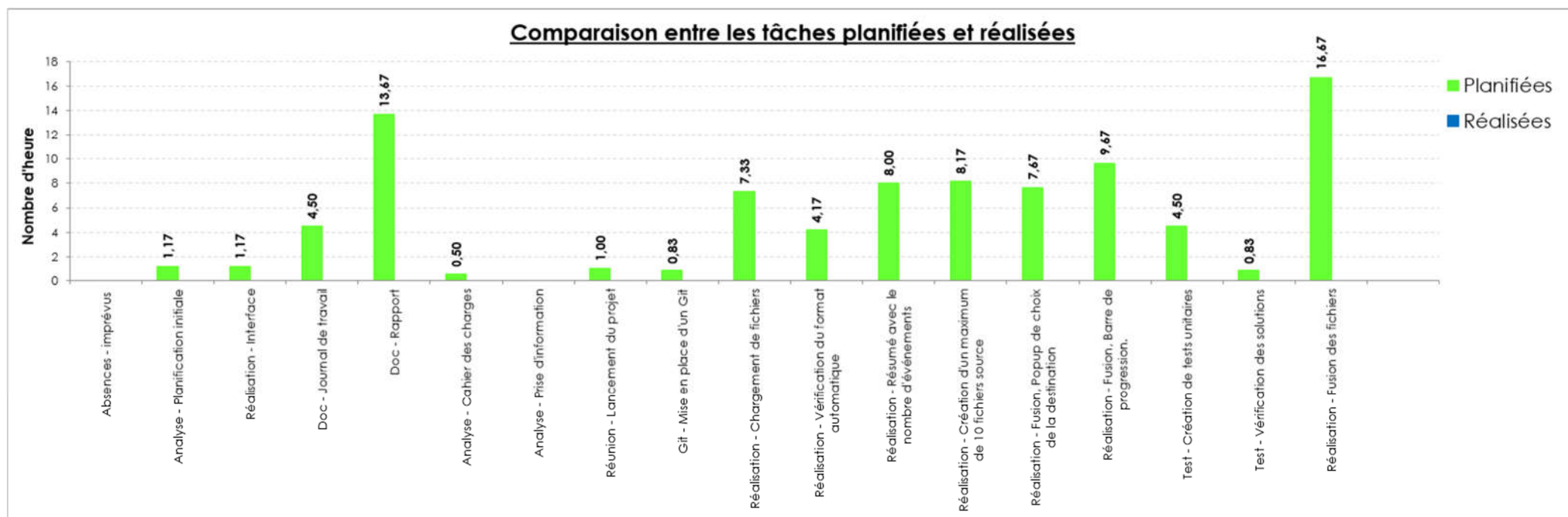
Merger : Une classe qui permet de fusionner les événements de plusieurs calendriers en un seul.

OpenFileDialog : Boîte de dialogue permettant à un utilisateur de sélectionner un fichier.

SaveFileDialog : Boîte de dialogue permettant à l'utilisateur de choisir où importer un fichier.

SourceComponents : C'est une classe qui réunit plusieurs contrôles. Elle permet à l'utilisateur d'importer les fichiers qu'il souhaite fusionner.

2.2 Planification initiale



3. Planification initiale du projet

2.3 Conception

2.3.1 Maximum de 10 fichiers source

Pour commencer avec ce projet il serait bon de s'occuper de la partie graphique. Il faudrait qu'il soit possible d'ajouter jusqu'à 10 sources qui seront au début, purement visuelles. Celles-ci seront gérées par un objet nommé « SourceComponents ».

Un « SourceComponents » sera un objet qui contiendra les contrôles nécessaire pour que l'utilisateur puisse ajouter un fichier source.



4. Mock-up de deux objets SourceComponents

Sur cette image il y'a deux lignes, donc deux fois cet objet :

Cela implique que chaque « SourceComponents » devra se charger de stocker le chemin d'un fichier chargé dans une variable de type string.

A chaque initialisation d'un de ces objet, il s'occupera d'afficher ses contrôles lui-même.

Pour savoir si nous avons atteint le nombre maximum de sources, le formulaire contiendra une liste de « Source Components ».

Pour dix objets dans cette liste, il y'aura dix lignes qui permettront d'ajouter une source. Donc, dix « SourceComponents » correspond à dix sources. Pour ajouter une source il y'aura un bouton prévu à cet effet :

A rectangular button with a light gray border and a light gray background, containing the text 'Ajouter une source' in a dark blue font.

5. Bouton d'ajout d'une source

Il serait dérangement de pouvoir seulement en ajouter mais de ne pas en enlever. Pour cela un bouton de suppression sera prévu à cet effet.

A rectangular button with a light gray border and a light gray background, containing the text 'Retirer une source' in a dark blue font.

6. Bouton de suppression d'une source

Ce bouton impliquera qu'on retira le dernier « SourceComponents » de la liste et que l'on supprimera donc la dernière ligne ajoutée.

2.3.2 Chargement de fichiers

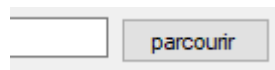
Il y'aura deux façons possibles d'importer un fichier à fusionner :

- Glisser/Déposer le fichier sur le textbox d'une source.



7. Contrôles pour le chargement de source

- Cliquer sur un bouton « parcourir » qui ouvrira une fenêtre permettant de choisir le fichier à importer.



8. Bouton parcourir

2.3.3 Vérification du format automatique

La classe « SourceComponents » contiendra une méthode qui se chargera de vérifier si le fichier indiqué est valide.

Cette méthode se lancera à chaque fois que :

- L'utilisateur modifie le chemin directement via le champ de texte
- Lorsque l'utilisateur importe un fichier source via le bouton « parcourir »
- Lorsque l'utilisateur utilise la fonctionnalité Glisser/déposer

Car toutes ces actions impliquent que l'utilisateur veut charger un fichier à fusionner.

Si le chemin est valide et que le contenu du fichier es valide, le champ texte du « textbox » sera rempli par le chemin du fichier. Dans le cas contraire, le label chargé de donner le nombre d'événement affichera que le fichier est invalide. Cela empêchera l'utilisation du bouton de fusion.

2.3.4 Résumé avec le nombre d'événements

Lorsque l'on importe un fichier .ics valide, le label affichera le nombre d'événements qu'il contient comme suit : OK : 8 événements.

Dans le cas où l'utilisateur essaierait d'importer un fichier non valide, le label affichera : KO : Invalide.

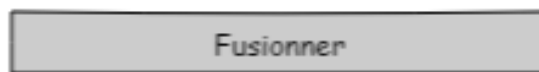
Voici une image de ces deux types de résultats que peuvent retourner ces labels, selon le cahier des charges :



9. Résultat d'analyse de fichier

2.3.5 Fusion et Pop-up de fusion

Lorsque toutes les fichiers sources sont valides, il sera possible de cliquer sur le bouton « Fusionner »



10. Bouton fusionner

Il sera utilisable uniquement, lorsque l'utilisateur aura inséré des fichiers sources valides. Dans le cas où un de ces fichiers aurait 0 événements, le fichier ne sera pas considéré comme valide et devra être remplacé ou l'objet « SourceComponents » devra être effacé. On pourrait très bien faire la fusion avec 0 événements, mais je souhaite écarter cette possibilité car :

- Cela ne servirait à rien de fusionner un fichier vide.
- Cela pourrait potentiellement générer des bugs et imprévus.
- L'utilisateur pourrait s'être trompé de calendrier et en avoir pris un vide.

Lorsque tous les calendriers entrés ont été validés par le programme, le bouton de fusion permettra d'ouvrir une fenêtre d'exportation. L'utilisateur devra choisir l'endroit où il souhaite sauvegarder le fichier fusionné. Si l'utilisateur ferme cette fenêtre, rien ne se passe. Il n'y a pas de fusion. Dans le cas où il désigne un dossier où il souhaite mettre la fusion, une méthode de fusion de lance. Cette méthode parcourra chaque fichier et cherchera toutes les lignes commençant par « BEGIN :VEVENT ».

Car cela implique que les prochaines lignes seront des propriétés liées à l'événement.

```
BEGIN:VEVENT  
DTSTART:20210507T140000Z  
DTEND:20210507T150000Z  
SUMMARY:Titre de la tâche  
END:VEVENT
```

11. Format d'un événement ics

Dans cet exemple la méthode copiera tout de « BEGIN :VEVENT » à « END :VEVENT ». La dernière ligne impliquant la fin de l'événement.

Entre ces deux lignes, il y'en a trois qui sont liées aux caractéristiques de la tâche.

- DSTART : Début de l'événement (date et heure)
- DTEND : Fin de l'événement (date et heure)
- SUMMARY : Titre de l'événement

Toutes ces lignes seront ajoutées à une liste de données. Cette liste contiendra tous les événements à fusionner.

Pour que le calendrier créé soit valide il nécessite deux additions.

Il faut que tous les événements soient entre deux lignes précises. La première qui devra toujours être la première ligne du calendrier :

- BEGIN :VCALENDAR

La dernière définissant la fin du calendrier qui se doit d'être la dernière ligne du fichier :

- END :VCALENDAR

Voici un exemple de calendrier avec deux événements :

```
1 BEGIN:VCALENDAR  
2 BEGIN:VEVENT  
3 DTSTART:20210507T140000Z  
4 DTEND:20210507T150000Z  
5 SUMMARY:Titre de la tâche 1  
6 END:VEVENT  
7 BEGIN:VEVENT  
8 DTSTART:20210507T110000Z  
9 DTEND:20210507T120000Z  
10 SUMMARY:Titre de la tâche 2  
11 END:VEVENT  
12 END:VCALENDAR
```

12. Format d'un calendrier ics

Cet exemple fonctionne car :

- Le fichier commence par « BEGIN :VCALENDAR »
- Les deux événements commencent par « BEGIN :VEVENT »
- La fin des deux événements est définie par « END :VEVENT »
- La dernière ligne annonce la fin du calendrier par « END :VCALENDAR »

Le fichier fusionné aura le même schéma :

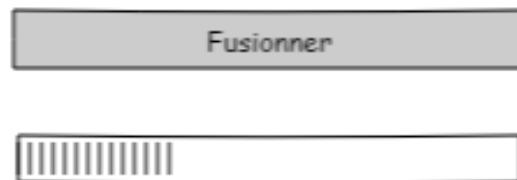
Début du calendrier

- Début événement 1
- Fin événement 1
- Début événement 2
- Fin événement 2

Fin du calendrier

2.3.6 Barre de progression

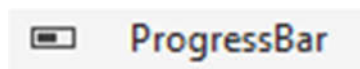
Voici l'emplacement de la barre de progression selon le cahier des charges :



14. Bouton fusionner et barre de chargement

J'ai choisi de garder cet emplacement car il est logique qu'elle s'applique pendant la fusion, donc après avoir interagi avec le bouton « Fusionner ».

J'ai décidé d'utiliser l'outil ProgressBar fourni avec le framework .NET windows form de visual studio.



13. Barre de progression windows form

Elle se présente ainsi avec un taux de complétion de 50%



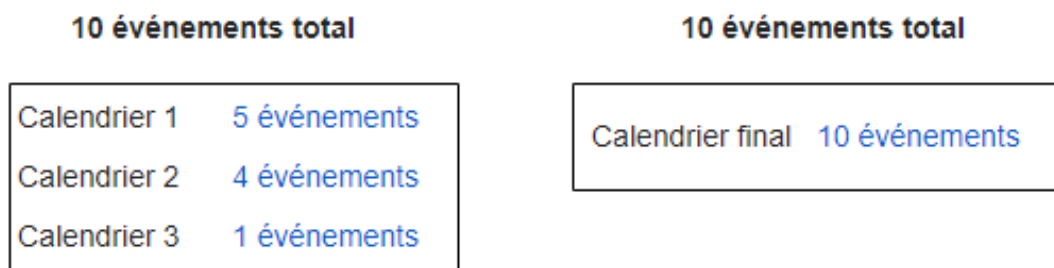
15. Objet barre de progression windows forms

2.3.7 Vérification de l'intégrité du fichier fusionné

La vérification s'effectue dès que tous les événements ont été fusionnés dans un seul fichier. Ce fichier devra être parcouru pour qu'on y compte le nombre d'événements final.

Pour définir si la fusion a fonctionné, il faudra que le nombre d'événements contenu soit égal au nombre total d'événements de chaque calendrier source.

Voici un schéma représentant une fusion réussie :



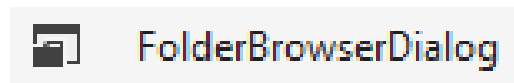
16. Schéma importation réussie

Dans le cas où il n'y aurait pas d'égalité, un message erreur s'affichera à l'utilisateur. Il lui sera expliqué que les calendriers n'ont pas été fusionnés.

2.3.8 Avertissement fichier de destination

Cette fonctionnalité s'applique lorsque l'utilisateur clique sur le bouton « Fusionner ». S'il choisit un emplacement où un fichier du même nom existe déjà, on affiche un avertissement.

Pour cela, nous allons utiliser un outil du framework .NET windows form de visual Studio. Il s'appelle un « FolderBrowserDialog ».



17. Objet FolderBrowserDialog

Il permet d'ouvrir une fenêtre qui permettra à l'utilisateur de sélectionner un dossier. Cela sera utile pour définir où l'on voudrait stocker le calendrier (.ics) résultant de la fusion.

Voici un test que j'ai effectué pour pouvoir déterminer son utilisation. J'ai créé un FolderBrowserDialog et j'ai laissé le nom par défaut « folderBroswerDialog1 ».

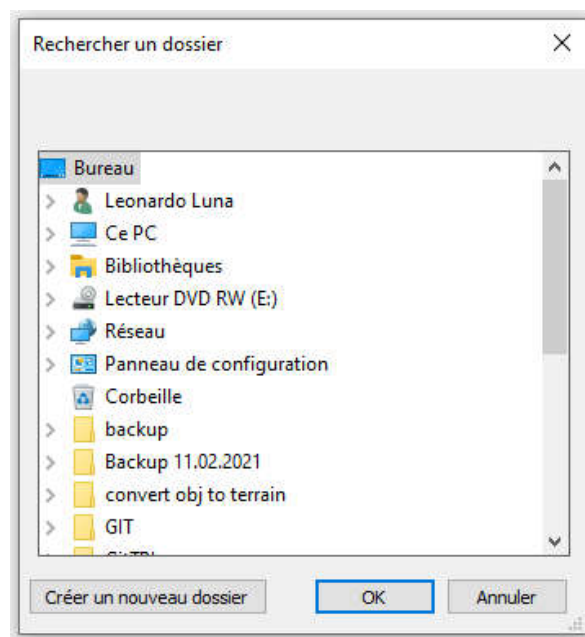
J'ai utilisé la méthode ShowDialog() pour pouvoir afficher la fenêtre souhaitée.

Dans le cas d'une implémentation, le nom de cet objet serait approprié à son utilisation. Cela n'est qu'un test pour but de comprendre comment utiliser cet objet.

Voici la commande utilisée :

```
folderBrowserDialog1.ShowDialog();
```

Voici la fenêtre que cela ouvre :



18. Boîte de dialogue

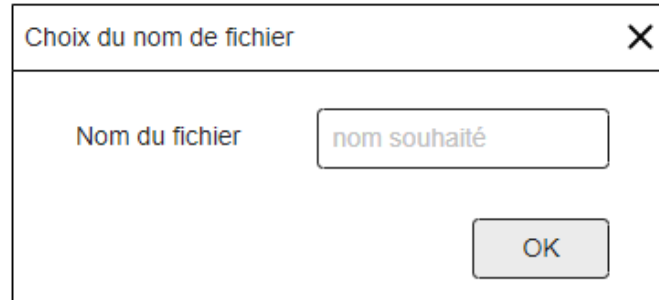
Pour choisir un dossier, il faut cliquer sur celui souhaité et puis sur le bouton « OK ». Pour aller chercher un dossier qui se trouverait à l'intérieur d'un autre il faudra cliquer sur l'icône de flèche à côté du dossier parent.



19. Icône flèche

Une fois que l'utilisateur a pu choisir le dossier qui va contenir la fusion de calendrier, il sera possible de retrouver son chemin. Il faut utiliser la variable « SelectedPath » qui retourne le chemin du dossier sous la forme d'un string.

L'utilisateur pourra choisir le nom du fichier fusionné. Pour cela, un nouveau formulaire s'ouvrira. Il contiendra un label «Nom du fichier» ainsi qu'un champ de texte qui lui permettra d'écrire le nom souhaité. Un bouton « OK » lui permettra d'envoyer sa réponse au programme.



The image shows a standard Windows-style dialog box. The title bar at the top reads 'Choix du nom de fichier' and has a close button (X) on the right. The main area of the dialog contains a label 'Nom du fichier' on the left and a text input field on the right. The input field contains the text 'nom souhaité'. Below the input field, centered, is a button labeled 'OK'.

20. Formulaire de choix du nom de fichier

Des vérifications s'effectueront.

Si le chemin entré précédemment est toujours valide et que le nom de fichier n'est pas déjà utilisé, le fichier (.ics) pourra être créé à l'endroit désigné.

Si le nom est déjà utilisé à l'emplacement, un message s'affichera pour que l'utilisateur choisisse un autre nom.

Si le nom est valide mais que l'utilisateur a supprimé une partie du chemin, il faudra qu'il entre un nouveau chemin.

À tout moment le processus de d'exportation pourra s'arrêter en fermant le formulaire ci-dessous ou en fermant la fenêtre du « FolderBrowserDialog » de la page précédente.

Ce sera utile si par exemple, l'utilisateur se rappelle qu'il souhaite ajouter un autre calendrier à fusionner. Cela implique qu'il ne souhaite pas poursuivre la fusion.

2.3.9 Utilisation d'un système de versioning

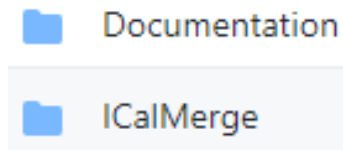
Pour le système de versioning j'ai décidé d'utiliser un git.

J'y accéderai via le logiciel GitHub Desktop.

Le git s'appelle : ICalMerge_TPI

Voici le lien pour y accéder : https://github.com/Fwai/ICalMerge_TPI

Je l'utiliserai tout au long du projet car il me servira à stocker le programme et sa documentation.

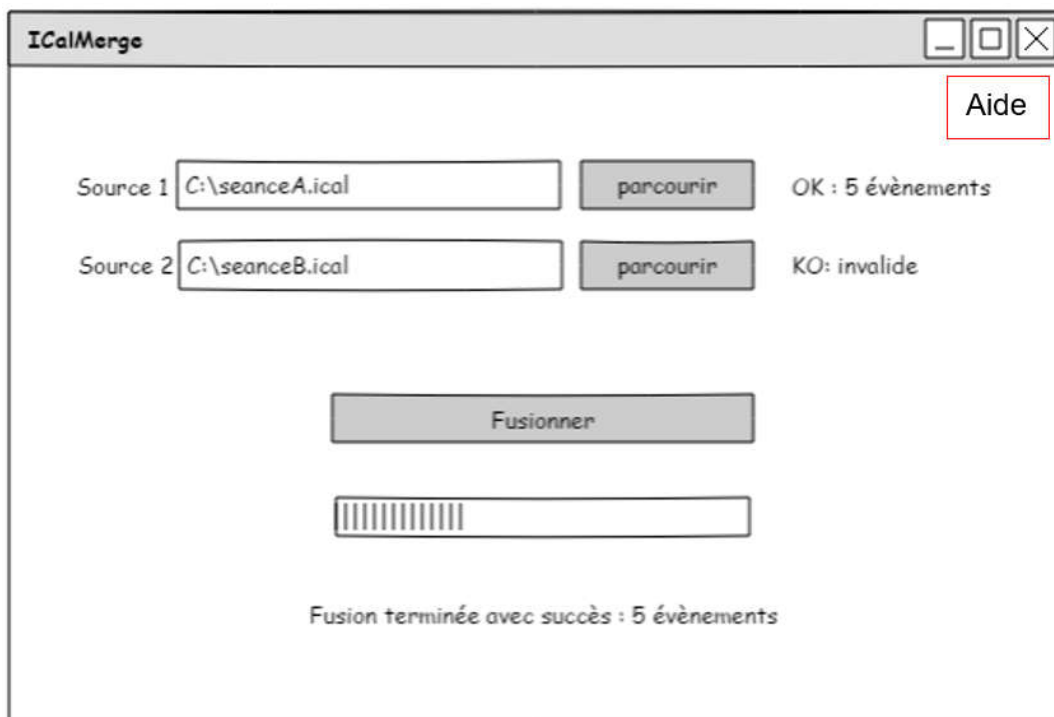


21. Fichiers principaux du git

2.3.10 Rubrique d'aide

La rubrique d'aide sert à ce que l'utilisateur comprenne assurément comment utiliser l'application.

Elle sera accessible via un label « Aide » situé en haut à droite de l'interface. Il sera aussi possible de cliquer sur la touche « F1 » pour ouvrir l'aide.



22. Label d'aide sur la page principale

Cela aura pour conséquence d'afficher un formulaire qui contiendra plusieurs informations sur les différents composants de l'interface.

2.3.11 Explication du format iCal

Pour qu'un fichier soit considéré au format iCal, il doit contenir les lignes suivantes :

Ligne	Emplacement	Définit
BEGIN:VCALENDAR	Doit être la première ligne du fichier	Le début du calendrier
END:VCALENDAR	Doit être la dernière ligne du fichier	La fin du calendrier

Ces deux lignes permettent de définir le début et la fin du calendrier iCal.

Par exemple, pour exporter un fichier iCal sur le calendrier Google, il est obligatoire que la première ligne soit « BEGIN:VCALENDAR », autrement le fichier est invalide.

En revanche il est possible d'écrire n'importe quoi après la fin du calendrier « END:VCALENDAR ».

La validité du fichier dépend entièrement de la manière dont on décide le traiter.

À la suite de la première ligne peuvent se trouver des caractéristiques propres au calendrier. Il est possible d'en rajouter et d'en créer soi-même. Cela sera quand même compatible avec d'autres applications car la ligne sera ignorée. Exemple :

Ligne	Définit
CALSCALE:GREGORIAN	Permet de définir l'agenda comme calendrier grégorien
ID:	Identifiant du calendrier
AUTHOR:	Nom de la personne qui a créé le calendrier
TITLE:	Titre du calendrier

Pour définir un événement, les lignes suivantes devront se trouver entre celles du calendrier :

Ligne	Emplacement	Définit
BEGIN:VEVENT	Entre BEGIN:VCALENDAR et END:VCALENDAR	Le début d'un événement
END:VEVENT	Doit être la dernière ligne du fichier	La fin d'un événement

Pour attribuer des données à l'événement, il est possible d'insérer les lignes suivantes :

Ligne	Définit
DTSTART:	La date et l'heure du début de l'événement
DTEND:	La date et l'heure du fin de l'événement
LOCATION:	L'endroit où se déroule l'événement
SUMMARY:	Titre de l'événement

Ces données devront se trouver entre BEGIN:VEVENT et END:VEVENT.

2.3.12 Respect des normes ETML

Les normes ETML seront respectées quand il sera pertinent de le faire.


Vous pourrez les trouver à cet endroit :

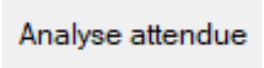
ICalMerge_TPI\Documentation\6.NormesDeCodageETML\
I-ConventionsDeCodageV3.5.0.pdf

2.4 Stratégie de test

Les tests ont été effectués directement via l'application ICalMerge.
Pour qu'un test soit validé, la fonctionnalité testée devra fournir le résultat désiré selon ses critères de validation.

1. Maximum de dix fichiers source	
Contexte	L'utilisateur devra être capable d'ajouter des sources. Cependant il pourra en ajouter un maximum de dix.
Procédure	Pour tester cette fonctionnalité, il sera uniquement nécessaire d'utiliser le bouton « Ajouter une source ». C'est grâce à ce contrôle que la fonctionnalité est utilisable.
Critères de validation	<ul style="list-style-type: none">• L'utilisateur peut ajouter des sources en cliquant sur le bouton « Ajouter une source »• Si l'utilisateur essaie d'ajouter plus de dix sources, un message d'erreur s'affiche.• Le message d'erreur explique la raison de l'erreur

2. Chargement de fichiers	
Contexte	L'utilisateur pourra charger les fichiers qu'il souhaite fusionner.
Procédure	<p>Les contrôles de la classe « SourceComponents » seront testés. Ils se présentent sous cette forme :</p>  <p>23. Un SourceComponents</p> <p>Ils devront permettre le chargement des données.</p> <p>Pour cela nous allons utiliser un des deux SourceComponents créés par défaut sur le formulaire principal.</p>
Critères de validation	<ul style="list-style-type: none"> • Cliquer sur le bouton « parcourir » ouvre une boîte de dialogue et permet à l'utilisateur de choisir un fichier à importer • La boîte de dialogue possède un filtre qui permet uniquement d'importer des fichiers ics. • La boîte de dialogue possède un second filtre permettant d'afficher tous les fichiers. • Lorsque l'importation par boîte de dialogue est terminée, le chemin du fichier s'affiche sur le champ textuel. • Cliquer deux fois sur le champ textuel permet d'ouvrir la boîte de dialogue. • Le chemin peut être entré à la main

3. Vérification du format	
Contexte	Lorsque l'utilisateur ajoute une source, il doit savoir si le fichier entré est valide.
Procédure	<p>Pour tester cela, il faudra utiliser un des deux SourceComponents affichés par défaut.</p> <p>Les vérifications seront affichées via le label suivant qui se trouve à la droite du bouton « parcourir »</p>  <p>24. Label analyse attendue</p>
Critères de validation	<p>Lorsque l'utilisateur, modifie le champ textuel ou charge un fichier, une vérification se produira. Elle aura deux possibilités différentes :</p> <ul style="list-style-type: none"> • Si le fichier est conforme, le label « Analyse attendue » affiche « OK : » suivi de son nombre d'événement. • Si le chemin du fichier n'existe pas ou qu'il ne contient pas d'événement, le label « Analyse attendue » Affiche « KO : » suivi du motif de l'invalidation.

4. Fusion des fichiers	
Contexte	L'utilisateur pourra fusionner les fichiers source qu'il aura entré au préalable.
Procédure	<p>Pour tester cela, des calendriers Google au format .ics seront utilisés. Ils contiendront plusieurs événements chacun.</p> <p>Les fichiers seront importés via les des SourceComponents.</p> <p>La fusion se lancera lorsque l'utilisateur cliquera sur le bouton « Fusionner ».</p> <p>Un test unitaire sera effectué. Il servira à vérifier que l'objet « Merger » fonctionne avec des données fictives.</p>
Critères de validation	<ul style="list-style-type: none"> • Lorsque l'on clique sur le bouton « Fusionner », la fusion se lance. Pour en témoigner, la barre de progression se remplit jusqu'à que la fusion se termine. • Une fois que la fusion est terminée, un label affichera le nombre d'événements fusionnés. Il se situera en dessous de la barre de progression. • Le test unitaire est validé avec les données fictives.

5. Exportation de la fusion	
Contexte	Lorsque le programme a fini de fusionner les fichiers, il sera possible d'exporter le fichier résultant de la fusion.
Procédure	Pour cela, plusieurs fichiers seront fusionnés. Une fois que la fusion sera terminée, nous pourrions vérifier l'exportation.
Critères de validation	<ul style="list-style-type: none"> • Une boîte de dialogue s'ouvre. • Il est possible d'entrer le nom du fichier. • Il est possible de choisir un dossier de destination. • Si le fichier existe déjà, un message prévient l'utilisateur. Il peut soit remplacer le fichier existant soit changer de nom. • Lorsque l'on clique sur le bouton « ok » de la boîte de dialogue, le fichier a été enregistré avec le nom désiré à l'endroit désiré avec l'extension.ics

6. Exportation de la fusion	
Contexte	Lorsque le programme a fini de fusionner les fichiers, il sera possible d'exporter le fichier résultant de la fusion.
Procédure	Pour cela, plusieurs fichiers seront fusionnés. Une fois que la fusion sera terminée, nous pourrions vérifier l'exportation.
Critères de validation	<ul style="list-style-type: none">• Une boîte de dialogue s'ouvre.• Il est possible d'entrer le nom du fichier.• Il est possible de choisir un dossier de destination.• Lorsque l'on clique sur le bouton « ok » de la boîte de dialogue, le fichier a été enregistré avec le nom désiré à l'endroit désiré avec l'extension.ics

7. Page d'aide	
Contexte	L'utilisateur peut recevoir de l'aide via un formulaire.
Procédure	La page d'aide sera testée via le formulaire principal. C'est depuis cette page que l'aide pourra s'ouvrir.
Critères de validation	<ul style="list-style-type: none">• Lorsque l'utilisateur clique sur F1, la page d'aide s'ouvre.• Lorsque l'utilisateur clique sur le label « aide », la page d'aide s'ouvre.• La page d'aide contient un SourceComponents qui permet à l'utilisateur d'importer un fichier.

2.5 Risques techniques

La barre de progression :

Je n'ai encore jamais fait de barre de progression auparavant sur windows forms. Il se pourrait que je peine à trouver la manière dont je dois l'utiliser. Il faudrait que je me documente davantage pour être à l'aise avec.

3 RÉALISATION

3.1 Dossier de réalisation

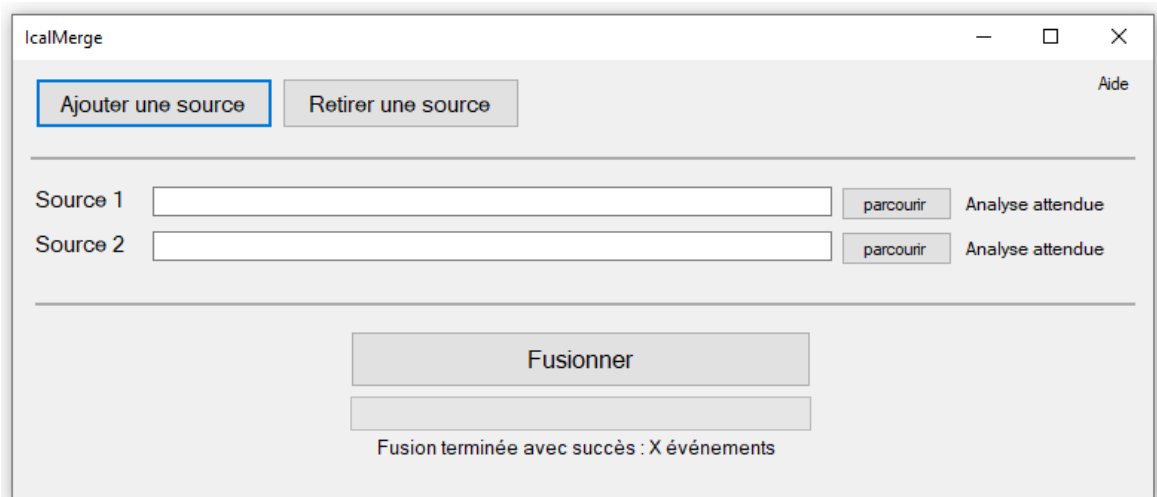
La méthode gestion de projet est Kanban.
Cette méthode a été réalisée via trello.

Voici le lien permettant d'y accéder :
<https://trello.com/b/EdRijZqc/icalmerge>

3.2 Interface graphique principale

L'interface se base entièrement sur le mock-up fourni dans le cahier des charges. Néanmoins, pour des raisons de confort d'utilisation et de fonctionnalités supplémentaires requises, certains contrôles ont été ajoutés.

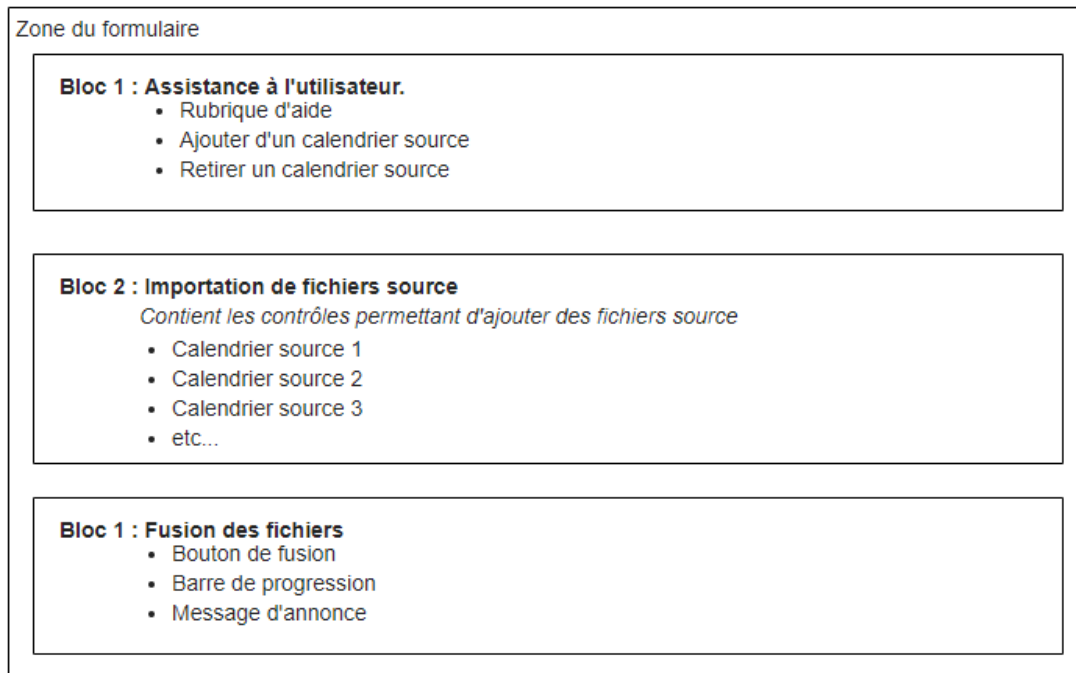
Voici l'interface graphique créée :



25. Interface principale IcalMerge

L'interface est composée de trois blocs. Chacun contient des contrôles qui servent à des buts précis.

Ces blocs sont définis par des panels. Cela permet de changer la disposition des contrôles par groupes en ayant le minimum d'impact possible sur les autres blocs.



26. Schéma de l'interface principale IcalMerge

3.2.1 Maximum de 10 fichiers source

Lorsque l'application démarre, deux emplacements de fichiers source sont créés par défaut. Voici à quoi ils ressemblent :

Source 1	<input type="text"/>	<input type="button" value="parcourir"/>	Analyse attendue
Source 2	<input type="text"/>	<input type="button" value="parcourir"/>	Analyse attendue

27. Les deux SourceComponents de base

Pour arriver à ce résultat, nous créons deux objets nommés « SourceComponents ». Chaque objet de cette classe regroupe les contrôles nécessaires à l'affichage. Cette classe permet à l'utilisateur d'ajouter un calendrier à fusionner. Un objet permet d'ajouter une source. Donc plusieurs objets, implique plusieurs sources.

Composition d'un SourceComponents :

Variables principales :

```
// Ce sont tous les contrôles qui vont être affichés visuellement à l'utilisateur.
private Label lblSourceName; // Permet d'afficher le titre et le numéro de la source. Exemple "Source 2"
private TextBox tbSourcePath; // Permet à l'utilisateur de voir le chemin de son fichier et de le changer à la main
private Button btnBrowse; // Permet à l'utilisateur d'ouvrir une fenêtre lui permettant d'importer un fichier
private Label lblEventResult; // Permet d'afficher le nombre d'événements que contient le fichier.
```

28. Variables principales d'un SourceComponents

Les variables ont été encapsulées.

Le constructeur de la classe :

```
public SourceComponents(PnlContainer pnlContainer, sbyte location, PnlFusion pnlFusion, Form mainForm)
{
    // On ajoute un espacement seulement pour les sources additionnelles. Les deux premières sources ont déjà une place suffisante.
    if (location >= 2)
    {
        // On redimensionne le formulaire contenant les SourceComponents, le formulaire principal et le panel servant à la fusion.
        // Cela permet de faire la place nécessaire à l'ajout d'une source.
        pnlContainer.Size = new System.Drawing.Size(pnlContainer.Size.Width, pnlContainer.Size.Height + 30);
        pnlFusion.Location = new System.Drawing.Point(pnlFusion.Location.X, pnlFusion.Location.Y + 30);
        mainForm.Size = new System.Drawing.Size(mainForm.Size.Width, mainForm.Size.Height + 30);
    }

    // Affichage des contrôles
    ShowSourceControls(pnlContainer, location);
}
```

29. Le constructeur d'un SourceComponents

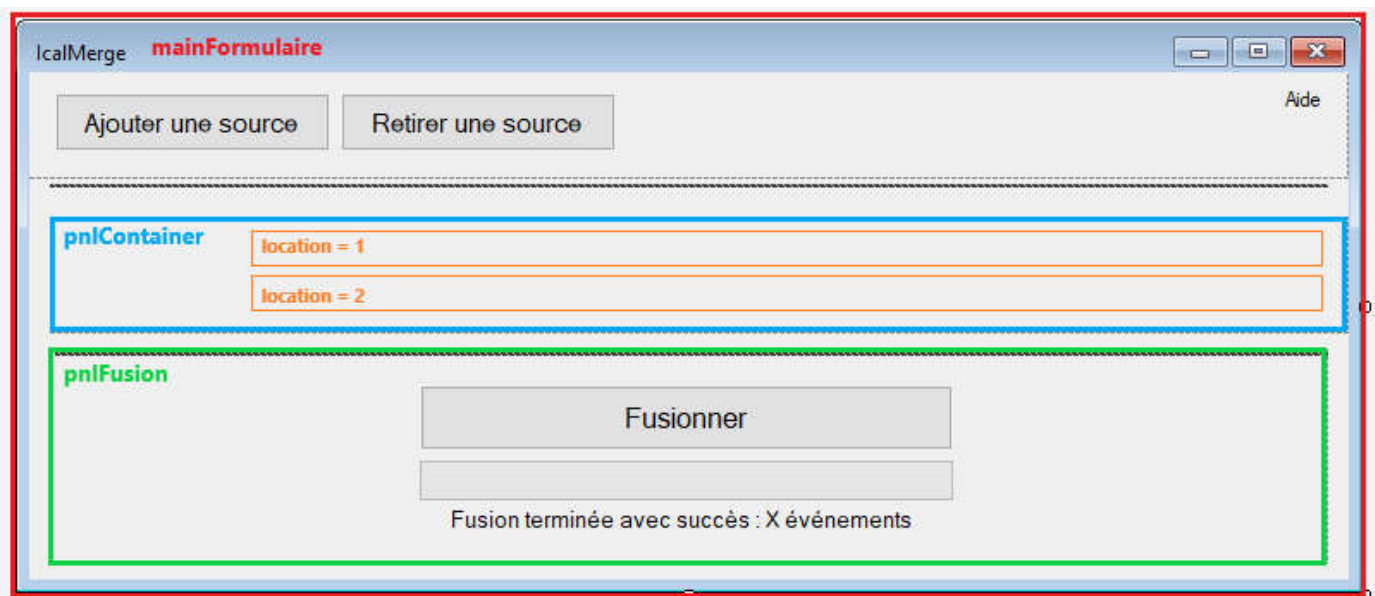
pnlContainer : Définit le panel qui va contenir les sources.

location : Définit le numéro de la source. Si le programme contient trois sources, en ajouter une nouvelle lui donnera location = 4.

pnlFusion : Définit le panel qui contient les contrôles permettant à l'utilisateur de fusionner les fichiers. On en a besoin car, à fois que l'on veut ajouter un fichier source, il faudra baisser ce panel.

mainForm : Définit le formulaire principal. Car à chaque fois que l'on veut ajouter une source, il faudra agrandir le formulaire principal.

Voici une représentation graphique de ces valeurs :



30. Schéma de la conception du formulaire principal

La méthode ShowSourceControls :

```
/// <summary>
/// Permet d'afficher tous les contrôles nécessaires à la récupération d'un fichier source.
/// </summary>
/// <param name="pnlContainer">Panel qui va contenir les contrôles</param>
/// <param name="location">numéro de la position</param>
1 reference
private void ShowSourceControls(PnlContainer, sbyte location)
{
```

31. En-tête de la méthode ShowSourceControls

Cette méthode est appelée par le constructeur. Son contenu pourrait être directement dans le constructeur, mais cela permet de séparer les différentes étapes du processus.

Elle sert à afficher tous les contrôles nécessaires pour que l'utilisateur puisse rajouter une source. Ces contrôles sont stockés dans la classe « SourceComponents ». C'est ceux qui ont été montrés à la page précédentes sous « Variables Principales ».

Voici les contrôles concernés :

Source 1	<input type="text"/>	<input type="button" value="parcourir"/>	Analyse attendue
----------	----------------------	--	------------------

32. Contrôles d'un SourceComponents

Limitation à 10 fichiers source :

Un fichier source correspond à un SourceComponents.

Pour cela, le formulaire principal contient une liste de SourceComponents.

```
/* Définit une liste de sources. Les sources ont des composants visuels  
ainsi que des valeurs pouvant stocker le chemin d'une source définie par l'utilisateur. */  
List<SourceComponents> listSources;
```

33. Liste de SourceComponents

Lorsque l'on clique que le bouton « ajouter une source » :

Si le nombre de sources disponibles actuelles est inférieur à 10

- Crée un SourceComponents et l'ajoute à la liste

Si le nombre de sources est de 10

- Affiche un message d'erreur avertissant l'utilisateur que le nombre maximal a déjà été atteint

C'est la méthode « AddSource » qui s'occupe de cela.

```
private void AddSource()  
{  
    // Vérifie si l'on a atteint la limite de sources maximale.  
    if (listSources.Count == 10)  
    {  
        // Comme il y'a déjà le maximum de sources, nous prévenons l'utilisateur grâce à un message box.  
        MessageBox.Show("La limite de fichiers source a été atteinte");  
    }  
    else  
    {  
        // Création d'une nouvelle source  
        SourceComponents newSource = new SourceComponents(pnlSources, Convert.ToSByte(listSources.Count()), pnlFusion, this);  
        listSources.Add(newSource);  
    }  
}
```

34. Méthode AddSource

Comme il est possible d'ajouter des sources, il devrait aussi être possible d'en supprimer.

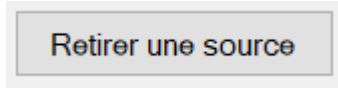
Pour cela j'ai ajouté une liste qui contient les contrôles de la classe SourceComponents. Cette liste se remplit à la fin de la méthode « ShowSourceControls ».

```
listControls = new List<Control>();  
// On regroupe les contrôles dans une liste  
listControls.Add(LblSourceName);  
listControls.Add(tbSourcePath);  
listControls.Add(btnBrowse);  
listControls.Add(lblEventResult);
```

35. Ajout des contrôles à la liste de contrôles

Cela permettra de faire plusieurs actions sur chaque contrôle en une fois. Cela à l'aide d'une boucle « foreach ».

Le bouton « Retirer une source » permet d'effacer une source.



36. Bouton de suppression d'une source

L'événement clic va appeler une méthode nommée « RemoveSource ».

```
/// <summary>
/// Se produit lorsque l'on clique sur le bouton de suppression d'une source
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 reference
private void BtnRemoveSource_Click(object sender, EventArgs e)
{
    RemoveSource();
}
```

37. Méthode clic

La méthode RemoveSource permet la suppression d'une source :

```
/// <summary>
/// Permet d'initier la suppression d'une source. Se produit seulement si il
/// y en a plus que deux
/// </summary>
1 reference
private void RemoveSource()
{
    if (listSources.Count == 2)
    {
        MessageBox.Show("Le programme nécessite deux sources au minimum");
    }
    else
    {
        listSources[listSources.Count - 1].Destruct(pnlSources, pnlFusion, this);
        listSources[listSources.Count - 1] = null;
        listSources.RemoveAt(listSources.Count-1);
    }
}
```

38. Méthode de suppression d'une source

S'il y'a plus que deux sources, on lance la méthode « Destroy » du dernier SourceComponents contenu dans la liste de sources. Après cela, nous retirons l'objet concerné à la liste. Pour définir le dernier élément, nous prenons le nombre d'objet dans la liste moins un.

La méthode « Destruct », redimensionne les contrôles du formulaire principal. Le but est d'effacer la place qui était nécessaire à l'emplacement de la source. Ensuite une boucle « foreach » permet de cacher tous les éléments visuels. Car l'utilisateur considère ne plus en avoir besoin.

```

/// <summary>
/// Permet de détruire tous ses contrôles et de rétablir
/// </summary>
/// <param name="pnlContainer">Panel qui contient les contrôles de cet objet</param>
/// <param name="pnlFusion">Panel en dessous de pnlContainer</param>
/// <param name="mainForm">Formulaire principal</param>
1 reference
public void Destruct(Panel pnlContainer, Panel pnlFusion, Form mainForm)
{
    pnlContainer.Size = new System.Drawing.Size(pnlContainer.Size.Width, pnlContainer.Size.Height - 30);
    pnlFusion.Location = new System.Drawing.Point(pnlFusion.Location.X, pnlFusion.Location.Y - 30);
    mainForm.Size = new System.Drawing.Size(mainForm.Size.Width, mainForm.Size.Height - 30);

    foreach(Control control in listControls)
    {
        control.Hide();
    }
}

```

39. Méthode de destruction d'une source

3.2.2 Chargement de fichiers

L'utilisateur a quatre moyens d'importer un calendrier.

1. Clic sur le bouton « parcourir »

Cliquer une fois sur le bouton « parcourir » ouvrira une fenêtre qui permettra à l'utilisateur de sélectionner un fichier avec une extension .ics.



41. Clic sur le bouton parcourir

```

/// <summary>
/// Permet d'ouvrir une boîte de dialogue qui permettra à l'utilisateur de choisir un fichier.
/// </summary>
/// <param name="sender">Contrôle qui appelle la méthode</param>
/// <param name="e">Détermine le type de méthode. Permet de donner les informations
/// sur la manière dont l'utilisateur utilise la souris</param>
2 references
private void ClickOpenFile(object sender, MouseEventArgs e)
{
    // Vérifie que l'utilisateur aie importé un fichier
    if (opfdOpenFile.ShowDialog() == DialogResult.OK)
    {
        TbSourcePath.Text = opfdOpenFile.FileName;
    }
}

```

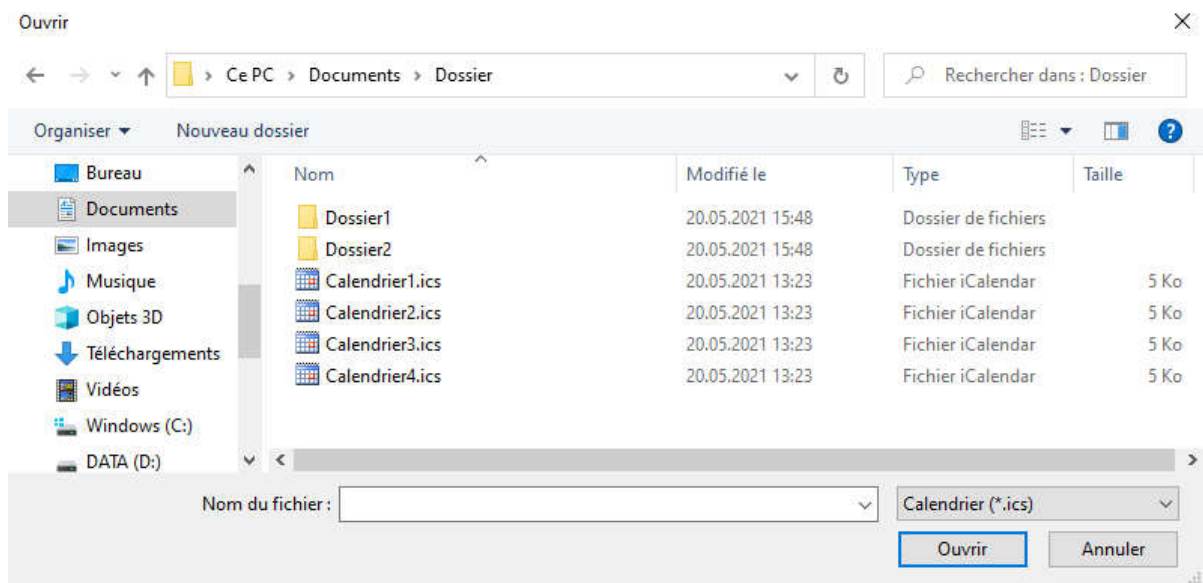
40. Méthode qui gère l'ouverture d'un fichier

Fonctionnement :

Lorsque l'utilisateur clique sur le bouton, la méthode suivante se charge d'ouvrir une boîte de dialogue. Elle permettra à l'utilisateur de choisir un fichier ics à importer.

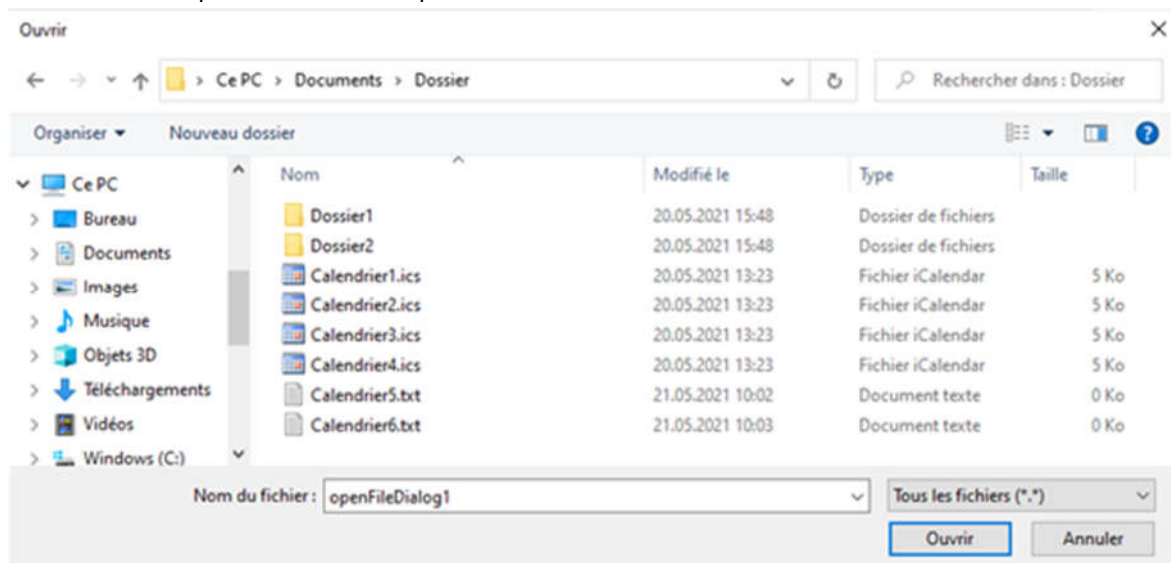
La fenêtre est issue de la classe « OpenFileDialog ».

Initialement, un filtre a été appliqué pour autoriser uniquement les fichiers ics. Suite à la demande de M.Melly, un autre filtre permettra d'entrer tout type de fichier. Cela sera utile dans le cas où certaines données au format ics se trouveraient dans d'autres types de fichiers. Exemple avec filtre ics :



42. Un OpenFileDialog avec filtre « ics »

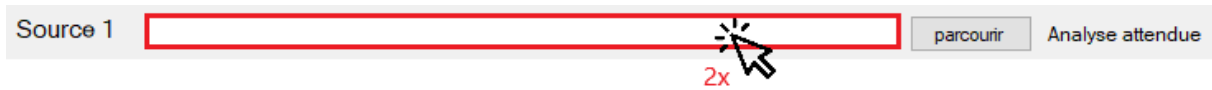
Exemple avec filtre pour autoriser tous les fichiers:



43. Un OpenFileDialog avec filtre "Tous les fichiers"

2. Cliquer deux fois sur le champ textuel

La même boîte de dialogue s'ouvrira à l'utilisateur lorsqu'il cliquera deux fois sur le champ textuel.



45. Clic double sur le champ textuel

Fonctionnement

Lors de la création du champ textuel, on lui assigne la méthode « ClickOpenFile » présentée précédemment. Cela impliquera que la méthode se lance à chaque fois que l'on clique deux fois dessus. Dans ce cas, la variable tbSourcePath correspond au champ textuel ci-dessus.

```
// On ajoute la méthode qui permettra de choisir un fichier via un OpenFileDialog  
tbSourcePath.MouseDoubleClick += ClickOpenFile;
```

44. Attribution de la méthode "ClickOpenFile" à "MouseDoubleClick"

3. Écriture du chemin

L'utilisateur peut aussi simplement écrire le chemin à son fichier dans le champ textuel.

Fonctionnement

A chaque fois que l'utilisateur modifie le texte du champ textuel, une vérification se fera. Ainsi il saura à tout moment s'il a entré un calendrier valide.

3.2.3 Vérification du format automatique

La vérification s'effectue de la manière suivante :

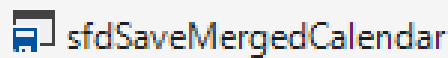
Étape	Description
1	L'utilisateur a importé un fichier
2	Vérification de la validité du chemin de fichier. S'il ne l'est pas, nous affichons un message d'erreur. S'il est valide, nous passons à l'étape suivante.
3	Vérifie qu'un événement a été inscrit.
4	Vérifie le nombre d'événements. S'il y'en a aucun, un message d'erreur s'affiche. Dans le cas contraire un label affichera le nombre d'événements.

3.2.4 Résumé avec le nombre d'événements

Si la vérification a été validée, un label affichera le nombre d'événements trouvés lors de l'analyse.

3.2.5 Fusion et pop-up de fusion

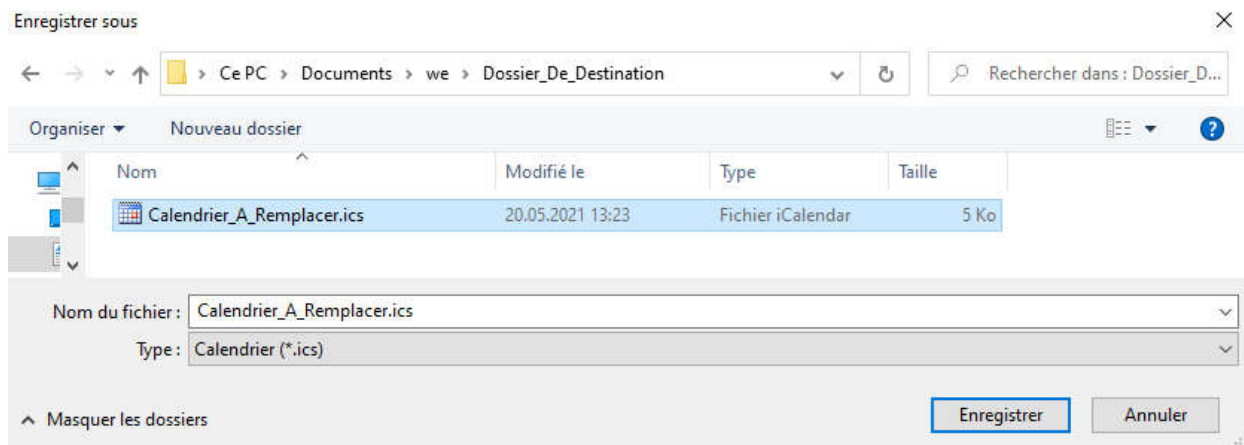
Pour gérer la fusion de fichier, le formulaire fait appel à un objet nommé « Merger ». C'est un objet qui s'occupe de ressortir tous les événements de chaque fichier et de les rassembler en un seul. Si le nombre d'événements correspond au nombre total d'événements à fusionner, il appelle un objet nommé « SaveFileDialog ». Il permet à l'utilisateur de choisir où enregistrer un fichier. Pour cela, l'objet ouvre une boîte de dialogue qui montrera les dossiers de l'ordinateur ainsi que les fichiers au format .ics uniquement. Dans ce cas-là je l'ai renommé « sfdSaveMergedCalendar ».



46. Objet SaveFileDialog

Il y'a deux manières d'enregistrer le fichier fusionné :

1. En choisissant un dossier et un nom de fichier
Cela créera un nouveau fichier au nom sélectionné à l'endroit sélectionné.
2. En cliquant sur un calendrier existant, cela permet de remplacer le contenu tout en gardant son nom.



47. Remplacement du contenu d'un fichier

3.2.6 Barre de progression

Voici son emplacement sur l'application :



48. Emplacement de la barre de progression

Variables de la barre :

- « **Maximum** » définit le 100% de la barre
- « **Minimum** » définit le 0% de la barre
- « **Value** » définit l'état actuel de la barre.

Comme discuté avec le chef de projet, ces valeurs devraient être définies selon le nombre de tâches que l'on souhaite fusionner. Cela se manifeste sous cette forme :

- « **Maximum** » définit le maximum de tâches à importer
- « **Minimum** » définit par le nombre 0. Car cela définit que l'on a importé 0 événements pour l'instant.
- « **Value** » définit par le nombre d'événements que l'on a déjà fusionné. Si l'on a fusionné douze événements, cette valeur sera égale à douze. Cette valeur sera actualisée à chaque importation d'un événement.

3.2.7 Avertissement fichier de destination

Pour l'exportation, j'ai utilisé un objet natif à Windows Forms. Il s'appelle un « OpenFileDialog ». Il permet d'exporter un fichier avec ses données.

Si l'utilisateur choisit un nom de fichier et qu'il existe déjà à l'endroit choisi, un message avertira l'utilisateur que le fichier existe déjà. Il pourra ainsi choisir de le remplacer ou pas.

3.2.8 Vérification de l'intégrité du fichier fusionné

Pour la vérification, le fichier à vérifier est parcouru.

Il y'a trois étapes à la vérification :

Critère	Description
1. Chemin du fichier	Le chemin du fichier est vérifié. Car l'utilisateur peut écrire un chemin à la main. Cela implique qu'il puisse faire des erreurs lorsqu'il écrit l'emplacement.
2. Contenu du fichier	A chaque fois que l'on détecte un début d'événement et une fin, cela compte comme un événement.
3. Le nombre d'événements que contient le fichier	Lorsque l'on a fini de parcourir le fichier, on vérifie le nombre d'événements sortis. S'il y'en a aucun, le fichier n'est pas validé. Autrement il l'est.

3.2.9 Rubrique d'aide

En cas de besoin, l'utilisateur peut faire appel à une aide.
Il lui suffira de cliquer sur la touche « F1 » ou sur le label « Aide » sur la page principale.

Voici la page d'aide :

Aide

— □ ×

Aide ICalMerge

Le programme sert à fusionner plusieurs calendriers ical entre eux.
Pour cela, il vous faudra suivre les étapes ci-dessous.

1. Nombre de calendriers

Vous devrez définir le nombre de calendriers à fusionner.
Pour cela vous avez deux boutons qui permettent d'en ajouter et d'en effacer.
Minimum: deux sources
Maximum: dix sources

Retirer une source

2. Importation des sources

Vous pourrez importer les fichiers en appuyant sur le bouton "parcourir".
Vous pouvez tester cette fonctionnalité via l'exemple suivant:

Source 1

parcourir

Analyse attendue

Cela vous permettra de choisir le fichier à fusionner.
Le label "Analyse attendue" vous indiquera si la fusion a fonctionné ou pas.
OK: importation réussie
KO: importation échouée
Dans le cas où vous obtiendriez un KO, la raison sera précisée.

3. Fusion des calendriers

Le bouton "Fusionner" vous permettra de lancer la fusion.
Pour pouvoir l'utiliser, il vous faut avoir importé toutes les sources et qu'elles soient "OK".

Fusionner

Pendant la fusion, une barre de progression vous permettra de suivre la progression de la fusion.
Voici à quoi elle ressemble quand elle est chargée à 50%.

Lorsque la fusion sera terminée,
il vous faudra choisir le nom du fichier, ainsi que l'endroit où le stocker.
Après cela, il vous suffira de cliquer sur "ok" pour terminer l'exportation.

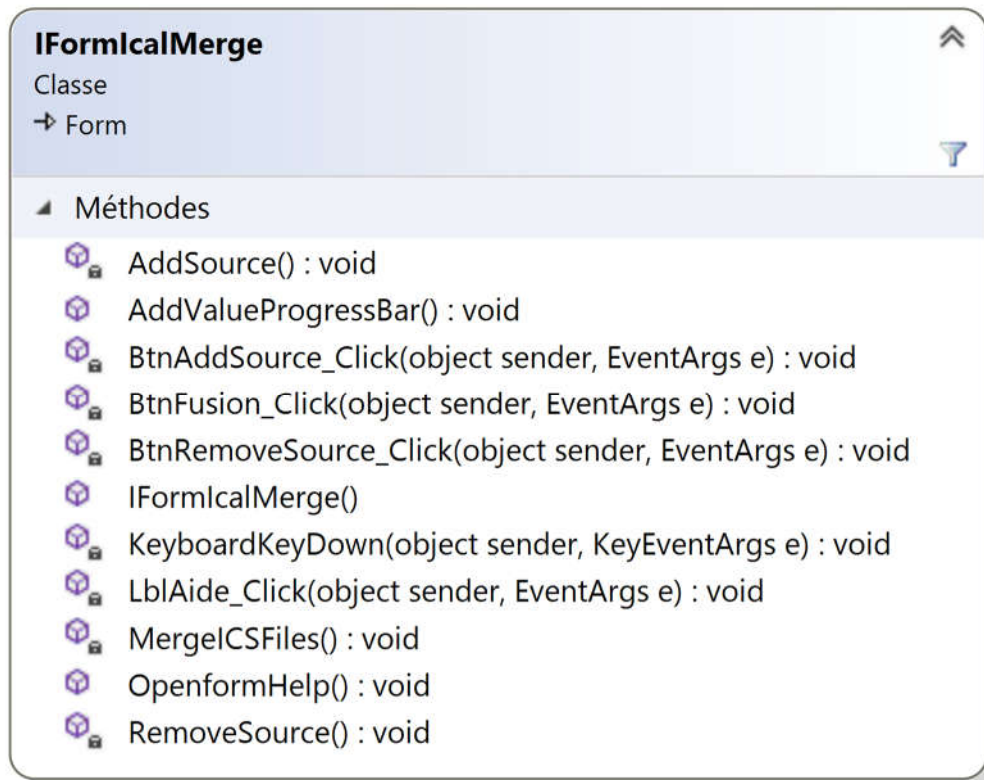
Fermer l'aide

49. Interface de la page d'aide

Le deuxième point contient des contrôles interactifs. Cela permet à l'utilisateur de charger un fichier test. Ainsi le label « Analyse Attendue » affichera le résultat de l'analyse et l'utilisateur pourra voir en direct les résultats « OK » ou « KO ».

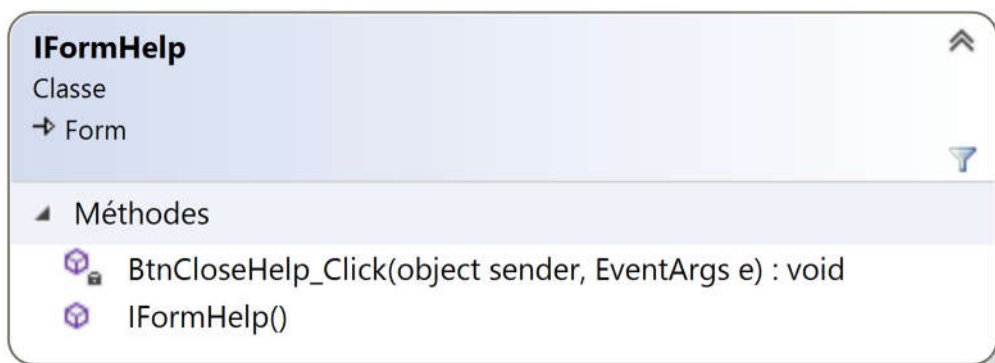
3.2.10 Diagramme de classes

Formulaire principal :



50. Méthodes du formulaire principal

Formulaire d'aide :



51. Méthodes du formulaire d'aide

Classe SourceComponents :

SourceComponents
Classe

▲ Propriétés

- 🔧 AllLines { get; set; } : string[]
- 🔧 BtnBrowse { get; set; } : Button
- 🔧 EventsNumber { get; set; } : int
- 🔧 IsFileValidated { get; set; } : bool
- 🔧 LblEventResult { get; set; } : Label
- 🔧 LblSourceName { get; set; } : Label
- 🔧 OpfdOpenFile { get; set; } : OpenFileDialog
- 🔧 TbSourcePath { get; set; } : TextBox

▲ Méthodes

- 🔧 AllowDrop(object sender, DragEventArgs e) : void
- 🔧 ClickOpenFile(object sender, MouseEventArgs e) : void
- 🔧 Destruct(PnlContainer, PnlFusion, Form MainForm) : void
- 🔧 DropFileOver(object sender, DragEventArgs e) : void
- 🔧 ShowSourceControls(PnlContainer, sbyte location) : void
- 🔧 SourceComponents(PnlContainer, sbyte location, PnlFusion, Form MainForm, OpenFileDialog opfdOpenFile)
- 🔧 TextBoxTextChanged(object sender, EventArgs e) : void
- 🔧 VerifyFileIntegrity() : void

52. Variables encapsulées et méthodes de la classe SourceComponents

Classe Merger :

Merger
Classe

▲ Propriétés




- 🔧 StrAllMergedLines { get; set; } : string







▲ Méthodes



- 🔧 AddContentToFuse(string[] allLines) : void
- 🔧 DefineMainForm(IFormIcalMerge MainForm) : void
- 🔧 FinishCalendar() : void
- 🔧 Merger()
- 🔧 ResetContentToFuse() : void




53. Variables et méthodes de la classe Merger






3.3 Description des tests effectués





1. Maximum de dix fichiers source	
Critère de validation	Résultat
L'utilisateur peut ajouter des sources en cliquant sur le bouton « Ajouter une source »	
Si l'utilisateur essaie d'ajouter plus de dix sources, un message d'erreur s'affiche.	
Le message d'erreur explique la raison de l'erreur.	




2. Chargement de fichiers	
Critère de validation	Résultat
Cliquer sur le bouton « parcourir » ouvre une boîte de dialogue et permet à l'utilisateur de choisir un fichier à importer	
La boîte de dialogue possède un filtre qui permet uniquement d'importer des fichiers ics.	
La boîte de dialogue possède un second filtre permettant d'afficher tous les fichiers.	
Lorsque l'importation par boîte de dialogue est terminée, le chemin du fichier s'affiche sur le champ textuel.	
Cliquer deux fois sur le champ textuel permet d'ouvrir la boîte de dialogue.	
Le chemin peut être entré à la main	

3. Vérification du format	
Critère de validation	Résultat
Si le fichier est conforme, le label « Analyse attendue » affiche « OK : » suivi de son nombre d'événements.	
Si le chemin du fichier n'existe pas ou qu'il ne contient pas d'événement, le label « Analyse attendue » Affiche « KO : » suivi du motif de l'invalidation.	

4. Fusion des fichiers	
Critère de validation	Résultat
Lorsque l'on clique sur le bouton « Fusionner », la fusion se lance. Pour en témoigner, la barre de progression se remplit jusqu'à que la fusion se termine.	
Une fois que la fusion est terminée, un label affichera le nombre d'événements fusionnés. Il se situera en dessous de la barre de progression.	
Le test unitaire est validé avec les données fictives.	

5. Exportation de la fusion	
Critère de validation	Résultat
Une boîte de dialogue s'ouvre.	
Il est possible d'entrer le nom du fichier.	
Il est possible de choisir un dossier de destination.	
Si le fichier existe déjà, un message prévient l'utilisateur. Il peut soit remplacer le fichier existant soit changer de nom.	
Lorsque l'on clique sur le bouton « ok » de la boîte de dialogue, le fichier a été enregistré avec le nom désiré à l'endroit désiré avec l'extension.ics	

6. Exportation de la fusion	
Critère de validation	Résultat
Une boîte de dialogue s'ouvre.	
Il est possible d'entrer le nom du fichier.	
Il est possible de choisir un dossier de destination.	
Lorsque l'on clique sur le bouton « ok » de la boîte de dialogue, le fichier a été enregistré avec le nom désiré à l'endroit désiré avec l'extension.ics	

7. Page d'aide	
Critère de validation	Résultat
Lorsque l'utilisateur clique sur F1, la page d'aide s'ouvre.	
Lorsque l'utilisateur clique sur le label « aide », la page d'aide s'ouvre.	
La page d'aide contient un SourceComponents qui permet à l'utilisateur d'importer un fichier.	

3.4 Erreurs restantes

Il ne reste pas d'erreurs connues à ce jour. Les tests ont été validés et n'ont en présenté aucunes.

4 CONCLUSIONS

Toutes les fonctionnalités souhaitées dans le cahier des charges ont été implémentées.

Ce projet m'a permis de mieux visualiser ce que le monde professionnel souhaite voir d'un informaticien. Que cela soit grâce aux critères TPI ou aux dialogues avec mes experts et mon chef de projet.

J'ai eu des difficultés pour la mise en place d'un système de glisser/déposer. Je ne l'avais pas mis dans les difficultés potentielles car j'ai sous-estimé cette tâche.

Les améliorations potentielles porteraient sur :

- La rapidité de la fusion. Pour permettre un plus grand nombre d'événements (exemple : 10'000)
- La page d'aide pourrait être implémentée directement sur la page principale. Cela serait sous la forme d'un tutoriel qui se calquerait sur le formulaire principal. Cela permettrait de faire une fusion, étape par étape.
- La fusion pourrait gérer d'autres types de fichiers pour offrir une plus grande variété de services.

Si le projet était à refaire, je planifierai plus de temps pour le rapport et moins pour la réalisation. J'essaierais de rendre l'interface plus dynamique et que la vérification des fichiers soit plus complète et prenne en compte plus de paramètres.

Concernant la planification initiale et finale :

J'avais imaginé que la réalisation prendrait à peu près le double de ce qu'elle a réellement pris. Pour cette raison, la tâche qui m'a pris le plus de temps fût la documentation. Pour pouvoir remplir le maximum des critères TPI, j'ai dû utiliser ce temps supplémentaire à l'écriture de mes différents documents livrables. En annexe vous pourrez trouver la comparaison entre la planification initiale et le journal de travail.

5 ANNEXES

5.1 Sources – Bibliographie

Image de la première page

Lien : <https://www.groupedelasalle-reims.com/le-college/infos-pratiques/calendrier-2020-2021>

C# Tutorial – Drag and Drop Text files into a RichTextBox | FoxLearn

Lien : <https://www.youtube.com/watch?v=RCnHLu0sp5c>

Extracting Path from OpenFileDialog path/filename

Lien : <https://stackoverflow.com/questions/439007/extracting-path-from-openfiledialog-path-filename>

Tutoriel de création d'un installateur:

Lien : [How to Create Setup.exe in Visual Studio 2019 | FoxLearn - YouTube](https://www.youtube.com/watch?v=RCnHLu0sp5c)

Icônes définissant si un test est validé ou non:

Lien du « validé » : https://www.flaticon.com/free-icon/check_463574?term=check&page=1&position=20&page=1&position=20&related_id=463574&origin=search

5.2 Manuel d'utilisation

Il se trouve directement sur l'application. Pour y accéder il faut appuyer sur la touche « F1 » ou en cliquant sur le label « Aide ».

5.3 Table de figures

1. Mock-up de l'application	4
2. Mock-up des contrôles servant à l'importation	5
3. Planification initiale du projet	7
4. Mock-up de deux objets SourceComponents	8
5. Bouton d'ajout d'une source	8
6. Bouton de suppression d'une source	8
7. Contrôles pour le chargement de source	9
8. Bouton parcourir	9
9. Résultat d'analyse de fichier	10
10. Bouton fusionner	10
11. Format d'un événement ics	11
12. Format d'un calendrier ics	11
13. Barre de progression windows form	12
14. Bouton fusionner et barre de chargement	12
15. Objet barre de progression windows forms	12
16. Schéma importation réussie	13
17. Objet FolderBrowserDialog	13
18. Boîte de dialogue	14
19. Icône flèche	14

20. Formulaire de choix du nom de fichier.....	15
21. Fichiers principaux du git.....	16
22. Label d'aide sur la page principale	16
23. Un SourceComponents	20
24. Label analyse attendue.....	20
25. Interface principale IcalMerge.....	23
26. Schéma de l'interface principale IcalMerge	24
27. Les deux SourceComponents de base.....	24
28. Variables principales d'un SourceComponents	24
29. Le constructeur d'un SourceComponents.....	25
30. Schéma de la conception du formulaire principal	25
31. En-tête de la méthode ShowSourceControls.....	26
32. Contrôles d'un SourceComponents	26
33. Liste de SourceComponents	27
34. Méthode AddSource	27
35. Ajout des contrôles à la liste de contrôles.....	27
36. Bouton de suppression d'une source.....	28
37. Méthode clic	28
38. Méthode de suppression d'une source	28
39. Méthode de destruction d'une source.....	29
40. Méthode qui gère l'ouverture d'un fichier	29
41. Clic sur le bouton parcourir.....	29
42. Un OpenFileDialog avec filtre « ics »	30
43. Un OpenFileDialog avec filtre "Tous les fichiers".....	30
44. Attribution de la méthode "ClickOpenFile" à "MouseDownClick".....	31
45. Clic double sur le champ textuel.....	31
46. Objet SaveFileDialog	32
47. Remplacement du contenu d'un fichier.....	32
48. Emplacement de la barre de progression	33
49. Interface de la page d'aide	35
50. Méthodes du formulaire principal.....	36
51. Méthodes du formulaire d'aide.....	36
52. Variables encapsulées et méthodes de la classe SourceComponents	37
53. Variables et méthodes de la classe Merger.....	37

5.4 Liste des documents fournis

- Planification initiale
- Journal de travail
- Bilan de la planification/réalisation
- Abstract/Résumé du projet