ÉCOLE CENTRALE LYON

ELC D6 - PLM maquettage numérique
Rapport de projet : VBA Lotus - Catia

EPSA - LASauto (v2)

Élèves :
Schio Michele
Matteï Calixthe

Enseignants :
Didier Lacour
Paul Clozel

3 mai 2020

Table des matières

1 Introduction

1.1 Contexte et motivation

Lotus est un logiciel d'analyse cinématique de la Liaison Au Sol (LAS) d'une voiture. Il permet notamment de définir les points caractéristiques des liaisons types d'une liaison au sol, de définir ces liaisons et d'étudier leur mouvement.

La démarche courante à l'EPSA combine une feuille de calcule Excel avec une génération des pièces par Catalogue CATIA : cette méthode est très fonctionnelle mais également un peu lente. Une fois la configuration de points déterminée (sur Lotus), il est nécessaire d'ouvrir un tableau Lotus, de sélectionner manuellement les coordonnées des points, les copier-coller sur un fichier Excel dédié au paramétrage, enregistrer et fermer le fichier Excel, ouvrir le Catalog LASauto, ouvrir la fenêtre de mise à jour, sélectionner tous les composants dans l'arbre, lancer la mise à jour, attendre quelques minutes, fermer le Catalog, ouvrir l'assemblage de travail, faire une mise à jour et vérifier que tout s'est bien passé.

Ainsi, l'optimisation de la LAS d'un véhicule, typiquement pour un véhicule EPSA, nécessite de nombreuses itérations afin de tendre vers une géométrie et un confinement offrant une dynamique véhicule et un poids optimal. Il est donc primordial pour l'EPSA d'optimiser les mises à jour de LAS afin de gagner un temps considérable. Le but de ce projet est donc de permettre la mise à jour des composants à l'intérieur de l'Assemblage de Liaison au Sol du Véhicule dans Catia (Assemblage qui sera nommé plus tard simplement `Suspension`), mise à jour possible directement en lançant la Macro, par lecture du fichier Lotus, sans intervention manuelle sur l'Assemblage et sans utiliser la technologie de Catalog. Il est également souhaité de pouvoir définir de façon adaptée les pièces de l'assemblage de travail.

L'intérêt de ce rapport est donc de permettre la compréhension du code et de pouvoir utiliser la macro convenablement.

1.2 Structure d'un fichier Lotus

Etant donné que ce projet est fondé sur l'étude d'un fichier Lotus au travers d'une macro Catia, la compréhension de la structure d'un tel fichier est importante. Le fichier d'enregistrement de Lotus est de type `.dat`. Il s'agit d'un fichier Texte structuré afin de pouvoir enregistrer les différents paramètres d'un modèle Lotus. Le fichier est composé de différentes sections, ayant pour titre : `VERSION`, `PARAMETERS`, `TEMP_GRAPHICS`, `TEMP_SETTINGS`, `FRONT SUSPENSION`, `REAR SUSPENSION`, `COMPLIANCE` et `MODAL`. Tout ceci est résumé dans la Fig.1

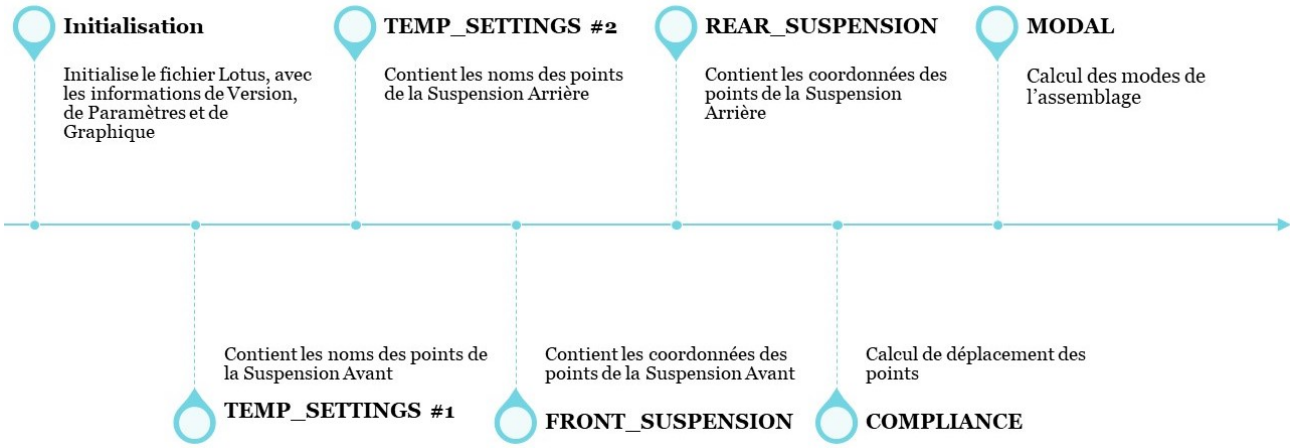

FIGURE 1 – Structure d'un Fichier Lotus

Ainsi, Lotus définit des modèles d'architecture pour différentes géométries de Suspension. Une géométrie de Suspension modifie tout simplement ce modèle et enregistre les nouvelles coordonnées des points dans un fichier `.dat`.

Dans la section `TEMP_SETTINGS` de ce fichier, on retrouve la définition des pièces de la suspension ainsi qu'une liste des points du modèle de suspension choisi.

Dans la section `FRONT SUSPENSION` et `REAR SUSPENSION`, on retrouve les coordonnées des points du modèle de suspension (respectivement pour le train avant et arrière).

Ce sont ces deux types de sections qui vont nous intéresser.

Structure de la section `TEMP_SETTINGS` Un exemple de section `TEMP_SETTINGS` d'un fichier Lotus est présenté en Fig. 2. Après l'entête (ligne 1) et la description du template utilisé pour la création du modèle Lotus (lignes 2 et 3), on retrouve une série de nombres. Le premier indique le nombre de pièces dans le modèle et le deuxième le nombre de points dans le fichier. Dans l'exemple de la Fig. 2, le nombre de pièces est 6 et le nombre de points est 2. Le nom de chaque pièce est listé de la ligne 5 à la ligne 10, suivi de la définition de chaque point dans le modèle. On s'intéresse ici seulement à la lecture du nom du point : on lit donc la ligne 11 (pour le point numéro 1) et on saute 8 lignes pour arriver à la définition du point numéro 2 (ligne 20).

Structure de la section `FRONT (REAR) SUSPENSION` Un exemple de section `FRONT SUSPENSION` est présenté en Fig. 3. Après l'entête (ligne 3), on retrouve le code du template utilisé (ligne 2 Fig. 2) et ensuite un tableau de coordonnées des points du Template. Cet exemple utilise un Template avec 27 points, les coordonnées sont donc listées dans les lignes 5 - 31

```
TEMP_SETTINGS
            15
Double Wishbone, Rocker arm damper
             6          2          12          16          3
Lower Wishbone
Upper Wishbone
Track Rod
Pull Rod
Rocker
Upright
Lower wishbone front pivot
Not Defined
Not Defined
Not Defined
Not Defined
Not Defined
     -21.5000          75.0000          163.000
             1            0          1          -1          1          0
             0            0          0
             0            0          0           0          0          0          0
cockpit RT
Not Defined
Not Defined
Not Defined
Not Defined
Not Defined
       0.00000          0.00000          0.00000
             0            0         -1           0          0          0
             0            0          0
             0            0          0           0          0          0          0
             1            2          3
       0.00000          0.00000          0.00000
       1.00000          1.00000          1.00000
```

FIGURE 2 – Structure de la section `TEMP_SETTINGS` avec 2 points

```
EXTENDED_TYRE
             0          0
FRONT SUSPENSION
            15
          -157.000000000          205.000000000          188.000000000
           143.000000000          205.000000000          188.000000000
          -10.1000003815          598.000000000          145.000000000
          -132.000000000          270.000000000          394.000000000
           118.000000000          270.000000000          394.000000000
          -7.03999996185          572.000000000          377.600006104
          -7.03999996185          547.320007324          357.299987793
          -7.03999996185          290.640808105          133.988006592
           44.0676002502          560.000000000          188.000000000
           48.4177017212          144.779998779          230.000000000
          -7.03999996185          270.000000000          380.000000000
          -7.03999996185          292.000000000          206.000000000
         0.00000000000          515.799987793          264.799987793
         0.00000000000          627.000000000          264.799987793
          -7.03999996185          185.000000000          160.000000000
          -8.03999996185          185.000000000          160.000000000
           240.000000000          296.487701416          155.318206787
           230.000000000          410.734710693          384.742401123
           26.4057998657          441.477691650          383.727386475
           155.785202026          497.571502686          321.587005615
           155.000000000          123.876296997          230.478897095
           135.000000000          762.533020020          338.485992432
          -7.03999996185          230.000000000          160.000000000
          -200.000000000          210.000000000          500.000000000
          -200.000000000          210.000000000          29.0699996948
           200.000000000          210.000000000          29.5690002441
           200.000000000          210.000000000          500.000000000
           235.000000000
```

FIGURE 3 – structure de la section `FRONT SUSPENSION`

1.3 Besoins du système LAS vis à vis de la maquette

Les besoins du système liaison au sol sont résumés ci-dessous, représenté par d'abord le pseudo-code de l'application puis le cahier des charges qu'a dû respecter notre application.

Pseudo-code *Inputs, obtenu grâce à l'utilisateur, après lancement de l'Userform* (cf Section 4)

— Path du fichier Lotus .dat
— Path du Wireframe.CATProduct contenant LotusPoint.CATPart
— Paths d'autres Products à mettre à jour

Running

— **Appel du module LectLotus** (cf Section 2),
- Lecture du fichier .dat
- Récupération, dans des listes, des noms et des coordonnées des points du train avant et arrière

— **Appel du module MajCatia** (cf Section 3),
- Ouverture de fichier LotusPoint.CATPart
- **Pour** i=1 à len (nom_points train avant)
  - Si le point existe déjà : Mise à jour de ses coordonnées
  - Sinon : Création du point
- **Pour** i=1 à len (nom_points train arrière)
  - Si le point existe déjà : Mise à jour de ses coordonnées
  - Sinon : Création du point
- Mise à jour de LotusPoints.CATPart
- Mise à jour de Wireframe_Definition.CATProduct
- Mise à jour des autres Products sélectionnées par l'utilisateur
- Fermeture de tous les fichiers Catia

Outputs

— Nombre de points créés et mis à jour pour le train avant
— Nombre de points créés et mis à jour pour le train arrière

Cahier des charges de l'application VBA

— Offrir une interface graphique à l'utilisateur
— Permettre à l'utlisateur la sélection, par explorateur de fichiers, des documents qu'ils souhaitent utilisés, mettre à jour, etc
— Permettre la lecture des points et leurs coordonnées depuis un fichier Lotus `.dat` sélectionné à partir de l'explorateur de fichiers
— Mettre à jour le fichier Catia automatiquement et d'autres fichiers Catia souhaités par l'utilisateur
— Indiquer le nombre d'opérations sur les coordonnées effectuées à l'utilisateur afin qu'il puisse vérifier la bonne exécution du programme
— Anticiper quelques erreurs classiques et les indiquer à l'utilisateur quand il les exécute
— Pouvoir réutiliser les fichiers de conception détaillée de la saison 2020

2 Lecture du fichier Lotus

2.1 Organisation du code

La démarche employée a ainsi été de repérer d'abord les deux sections `TEMP_SETTINGS` afin de récupérer le nom des points pour créer une liste de noms des points de la Suspension Avant, puis les noms des points de la Suspension Arrière.

Ensuite, le code repère la section `FRONT_SUSPENSION` et crée la liste des coordonnées des points de la Suspension Avant, et de manière analogue, la liste des coordonnées des points de la Suspension Arrière est créée.

Pour résumer, voilà la structure synthétique du code :

1. Ouverture du fichier txt
2. Lecture des différentes parties de ce document
3. Sauvegarde des valeurs lues

2.2 Bibliothèques utilisées

Gestion des chemins de fichier avec accents `Option Compare text` est une bibliothèque à mettre avant le `Sub`, car il permet de comparer des chaînes selon un ordre de tri alphabétique en ne tenant pas compte de la casse qui est déterminé par les paramètres régionaux du système informatique utilisé.

TextStream Bibliothèque VBA permettant de créer un fichier TextStream à partir d'un fichier texte. Ce fichier TextStream permet de nombreuses manipulations sur ses lignes sans se soucier d'interférer ou non sur le fichier original. De plus, cette méthode permet d'avoir un objet à passer aux différentes fonctions pour la lecture du texte. Enfin, une fois ce fichier prêt, il est possible de l'enregistrer en tant que fichier txt final.

Dynamic Arrays On utilise la méthode `Dim - ReDim` afin de pouvoir définir une liste et de pouvoir déclarer sa taille réelle en fonction des paramètres lus depuis le fichier Lotus. Il faut faire attention car cette définition se fait en utilisant l'indice de la liste et non sa taille (cf. documentation Microsoft pour `ReDim`)

Conversion de string à double Le caractère de séparation des décimaux dans Lotus est celui du local anglais (le point). Il se peut que dans l'ordinateur où on exécute la macro, le local soit différent de celui anglais et dans ce cas, la conversion d'une *string* en *double* ne se fait pas correctement. Pour cela, on remplace avec `replace` le caractère point avec une virgule lors de la lecture des cordonnées dans le tableau Lotus puisque changer le séparateur dans le local de la machine n'est pas conseillé.

2.3 Lecture du fichier `.dat`

Pour la lecture du morceau de texte présenté en Fig. 2, voir la Subroutine `ReadTempSettings` et ses commentaires. De même pour la lecture du morceau de texte présenté en Fig. 3, voir la Subroutine `ReadSuspension` et ses commentaires.

3 Mise à jour du produit Catia

3.1 Principe de fonctionnement

La Fig. 4 présente la structure type des fichiers de travail : l'idée est de laisser l'utilisateur définir un filaire *ad hoc* dans le produit `WireframeDefinition.CATProduct` en créant des pièces (`Wireframe1.CATPart`, `Wireframe2.CATPart`, ...) avec des projections des points de `LotusPoints.CATPart`. La macro fait tout simplement une mise à jour des points dans `LotusPoints.CATPart` avec les valeurs lus depuis un fichier `Lotus.dat` et appelle une mise à jour complète du produit `WireframeDefinition.CATProduct`.

Etant donné que les points du train avant et arrière ont les mêmes noms, `LotusPoints.CATPart` contiendra deux corps, permettant ainsi de différencier le train avant et arrière lors de la mise à jour.

Ensuite, les pièces filaires (`Wireframe1.CATPart`, `Wireframe2.CATPart`, ...) peuvent être mises dans un produit séparé (`Suspension.CATProduct`) pour la conception détaillée.

Voici les différents avantages de cette approche :

- définition du filaire *ad hoc* sur un produit dédié : possibilité de définir des nouvelles pièces dans le filaire sans modifier la macro, permettant d'anticiper l'apparition de nouveaux sous-systèmes dans la suspension du véhicule, non présents lors des itérations précédentes (par exemple, ajout d'une barre anti-roulis)
- une mise à jour de `Suspension.CATProduct` ne modifie pas les pièces filaires puisque ces dernières ont été définies dans le contexte de `WireframeDefinition.CATProduct` : autrement dit la mise à jour des points est indépendante du fichier de conception détaillé
- Possibilité d'adapter la construction des pièces filaires aux exigences de forme : placement des repères au centre des pièces

3.2 Description et organisation du code

Dans le module *majCatia*, la subroutine *maj_ Catia* permet de mettre à jour les points présents dans `LotusPoints.CATPart` avec les noms et les coordonnées lues depuis le fichier Lotus, comme expliqué dans la section précédente.

Grâce à l'ouverture du produit `WireframeDefinition.CATProduct`, nous pouvons accéder directement au fichier `LotusPoints.CATPart`, étant déjà chargé dans la mémoire. Ainsi, le code fait tout d'abord une scan des points déjà présents dans `LotusPoints.CATPart` afin de pouvoir distinguer la création d'un nouveau point de la mise à jour d'un point existant. Ensuite, nous faisons une comparaison entre les noms présents dans la liste globale `frontNames` (créée avec la lecture du ficher Lotus.dat pour le train avant) et `rearNames` (de manière analogue pour le train arrière) avec ce qui était présent dans les deux corps de `LotusPoints.CATPart` :

- si le point existe déjà, ses coordonnées sont mises à jour en passant par les paramètres de l'objet
- sinon, un nouveau point est créé en utilisant la méthode *hybridShapeFactory* et ajouté dans l'arbre de la pièce

Petit détail concernant la conception à l'EPSA : la conception détaillée est faite par convention sur la partie gauche du véhicule, puis mis en symétrie. Néanmoins, les points Lotus sont définis à droite : nous avons simplement changer le signe de la coordonnée `Y` afin de réaliser la symétrie.

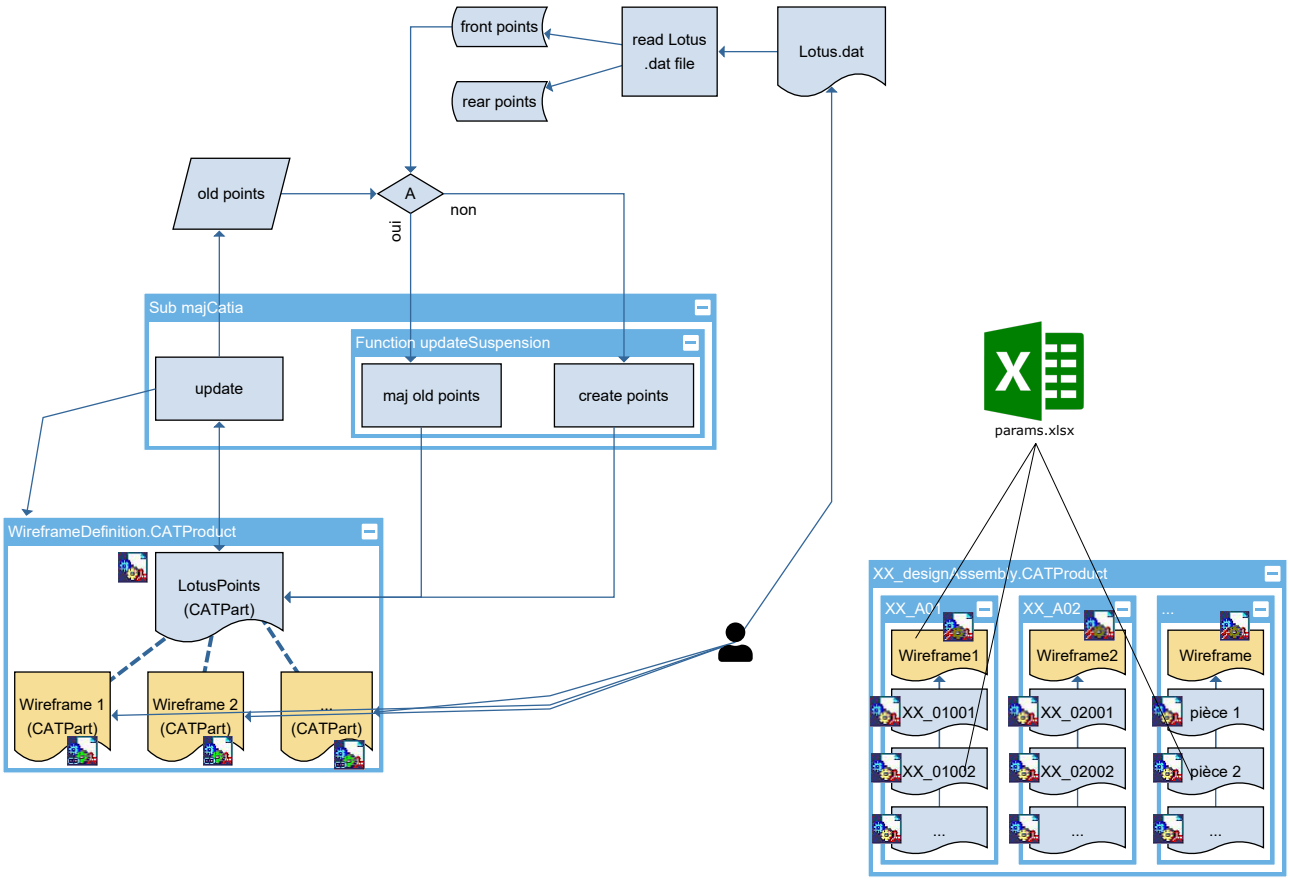

FIGURE 4 – Représentation schématique du principe de fonctionnement de la macro

3.3 Précautions et points faibles

Premièrement, nous remarquons que l'identification des points se base sur leur nom. Pour changer cette propriété, il faut rentrer directement dans le paramétrage du logiciel Lotus.

On remarque deuxièmement que cette macro ne peut pas effacer des points dans `LotusPoints.CATPart`. Pour ce faire, il faut ouvrir manuellement la pièce dans Catia et supprimer les point dans l'arbre. L'intérêt de ceci est d'éviter que la macro efface les mauvais points si l'utilisateur n'est pas précautionneux et nomme de la même manière les nouveaux points et ceux qu'il a supprimé dans Lotus.

4 Interface Graphique

4.1 Cahier des Charges spécifique

— Sélection dynamique du fichier Lotus depuis l'explorateur de fichiers puisque ce fichier va changer très souvent (versionnement des fichiers Lotus, donc versionnement de la LAS de le voiture, se fait par enregistrement de différents fichiers)
— Affichage des messages d'erreur et du résumé de mise à jour
— Bouton mise à jour de l'assemblage
— Sélection dynamique du fichier Catia depuis l'explorateur du fichier : mémoriser le parcours d'enregistrement des fichiers `WireframeDefinition.CATProduct` car il ne vont pas changer très souvent.
— Possibilité à l'utilisateur de sélectionner d'autres produits à mettre à jour après la mise à jour des points
— Conservation des derniers produits updatés lors de l'itération précédente, afin de faire gagner du temps à l'utilisateur lors des nombreuses itérations.

4.2 Définition de l'interface

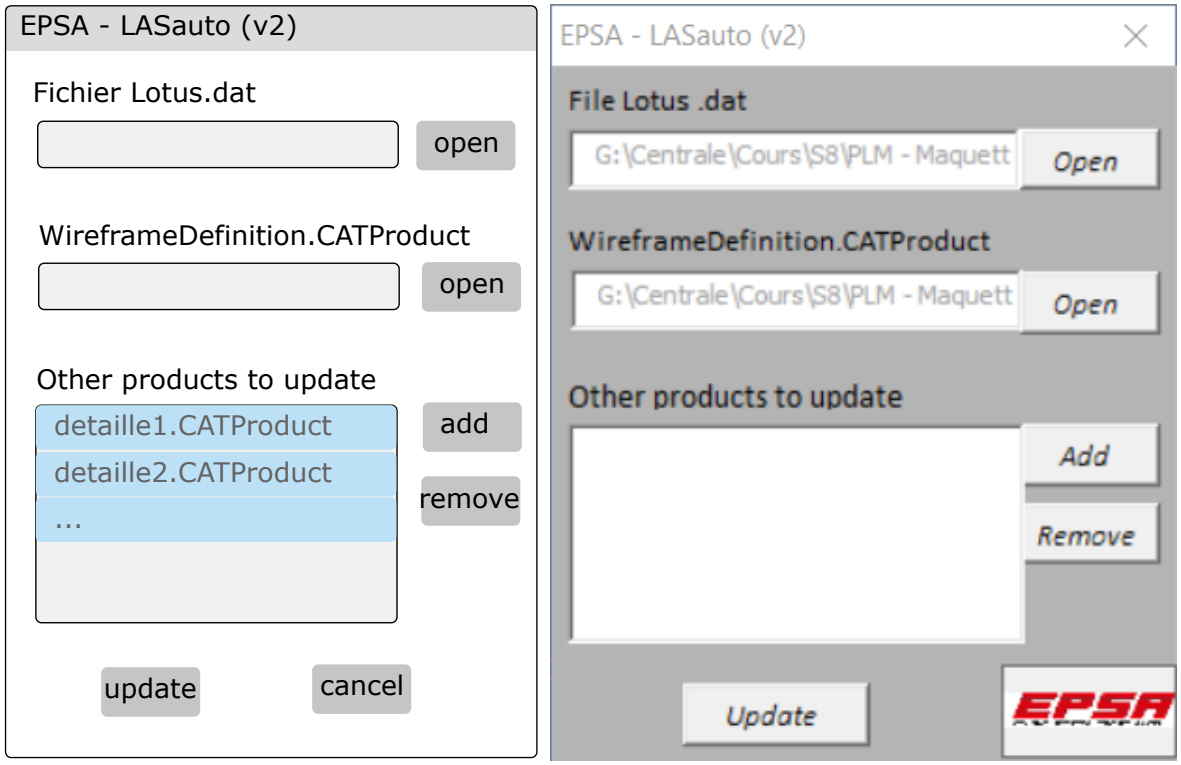

FIGURE 5 – Représentation schématique (gauche) et réalisation de l'interface (droite) de l'interface de la Macro

La Fig. 5 présente la structure souhaitée pour l'interface de la Macro : trois sections se définissent.
Premièrement, on retrouve le champ pour la sélection du fichier Lotus `.dat`.
Deuxièmement, on trouve le champ pour la sélection du Produit Catia pour la définition du

filaire à partir des points. La macro va automatiquement repérer à l'intérieur de ce Produit la pièce `LotusPoints.CATPart` afin de mettre à jour les points.
Troisièmement, on retrouve une section pour la sélection multiple des produits à mettre à jour suite au changement des points LAS : le choix est fait par l'utilisateur en fonction du travail demandé.

On remarque que l'utilisateur peut ne pas avoir besoin de changer tous les fichiers chaque fois puisque seulement le fichier `Lotus.dat` est sensé varier. On souhaiterait donc que cette application garde en mémoire les adresses des fichiers utilisés lors de la dernière utilisation tout en laissant à l'utilisateur la possibilité de les modifier en cas de besoin. Pour ce faire, un nouveau fichier `params.txt` est créé et mis à jour dans le même répertoire de stockage de la bibliothèque de la macro (plus de détails dans la section 4.3).

On remarque également que le bouton "Cancel" prévu dans la maquette de l'interface se matérialise dans le bouton de fermeture de la fenêtre de la macro.

4.3 Enregistrement des dernières positions des fichiers

Afin de simplifier l'utilisation de la macro, nous avons donc décidé de stocker la dernière valeur de la position du dernier fichier `WireframeDefinition.CATProduct` dans la première ligne du fichier `params.txt` et de laisser à l'utilisateur la possibilité de changer cette position à travers l'interface graphique. On implémentera la même solution pour l'ensemble des produits que l'utilisateur ajoutera dans le champ *Other products to update* de l'interface graphique et également pour le dernier fichier Lotus utilisé, afin de pouvoir les charger la fois suivante à l'ouverture de la macro.

Afin de charger ces paramètres lors du lancement de l'application, on utilise la subroutine `UserForm_Activate` et on implémente une simple lecture du fichier texte params.txt avec un objet de type *TextStream*. Pour l'enregistrement des positions des fichiers avant la fermeture de l'application, on utilise la subroutine *UserForm_ QueryClose* et on remplace tout simplement les nouvelles positions dans le fichier texte. La structure du fichier `params.txt` se définit donc de façon simple :

1. Path du dernier fichier `WireframeDefinition.CATProduct` dans la première ligne
2. Path du dernier fichier Lotus `.dat` dans la deuxième ligne
3. Path des derniers produits supplémentaires à mettre à jour à partir de la troisième ligne

Ainsi, afin de n'écrire qu'une fois dans le fichier `params.txt`, étant donné que la manipulation d'un fichier txt en VBA est plutôt lourde, nous passons par l'intermédiaire d'un tableau, appelé *update_products* dans notre code, tableau dans lequel nous ajoutons ou enlevons le path des produits ainsi ajoutés ou enlevés. Ensuite, nous copions termes à termes les paths dans `params.txt` à la fermeture de l'UserForm.

4.4 Messages d'erreurs

Les différentes erreurs pris en compte par cette Macro sont listés ci-dessous. Néanmoins, si des erreurs surviennent lors de la mise à jour des produits supplémentaires, la macro se bloque et ne termine pas l'exécution de son code. Nous n'avons pas pu régler ce problème.

— Si l'utilisateur oublie de renseigner la path d'un fichier Lotus et clique sur Update, la macro l'oblige à en choisir un
— Même méthode s'il oublie de choisir le path de `WireframeDefinition.CATProduct`.

5 Exemple d'utilisation : voiture de la saison 2020

On présente ici une application de la Macro afin de tester son bon fonctionnement et de montrer la démarche typique pour son utilisation au sein de l'EPSA. Cet exemple servira également à aider à l'évaluation de ce projet. Seulement une partie du train avant gauche est présentée par la suite : la reconstruction de l'assemblage complet aurait demandé trop de temps.

5.1 Création des pièces filaires

Dans un premier temps, il est nécessaire de créer les pièces filaires dans `WireframeDefinition.CATProduct` en utilisant les points à l'intérieur de `LotusPoints.CATPart`. Ci-dessous une liste complète des filaires à créer, présent dans la Fig. 6 avec une capture d'écran du *Bureau* CATIA permettant de comprendre l'architecture de notre Produit. [1]

- Châssis (frame)
- Triangles
- Biellettes de suspension, de direction et pince
- Basculeurs
- Barres anti-roulis [2]
- Colonne de direction
- Roues

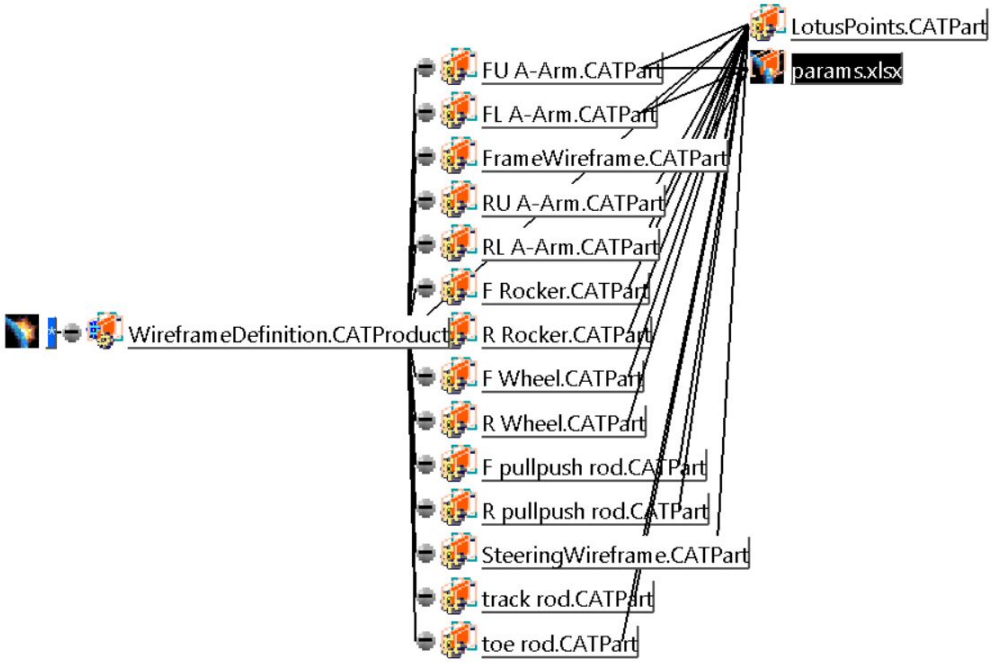

FIGURE 6 – Capture du *Bureau* de `Wireframe_Definition.CATProduct`, montrant une structure similaire à la Figure 4

1. afin d'obtenir la bonne structure, il est conseillé de cacher tout les dossiers "éléments de référence" dans l'arbre et de ne jamais sélectionner les éléments depuis l'affichage de la maquette mais de cliquer directement les éléments dans l'arbre du produit

2. ces filaires n'apparaissent pas à l'heure actuelle mais peuvent être mis en place très facilement dans le `WireframeDefinition`

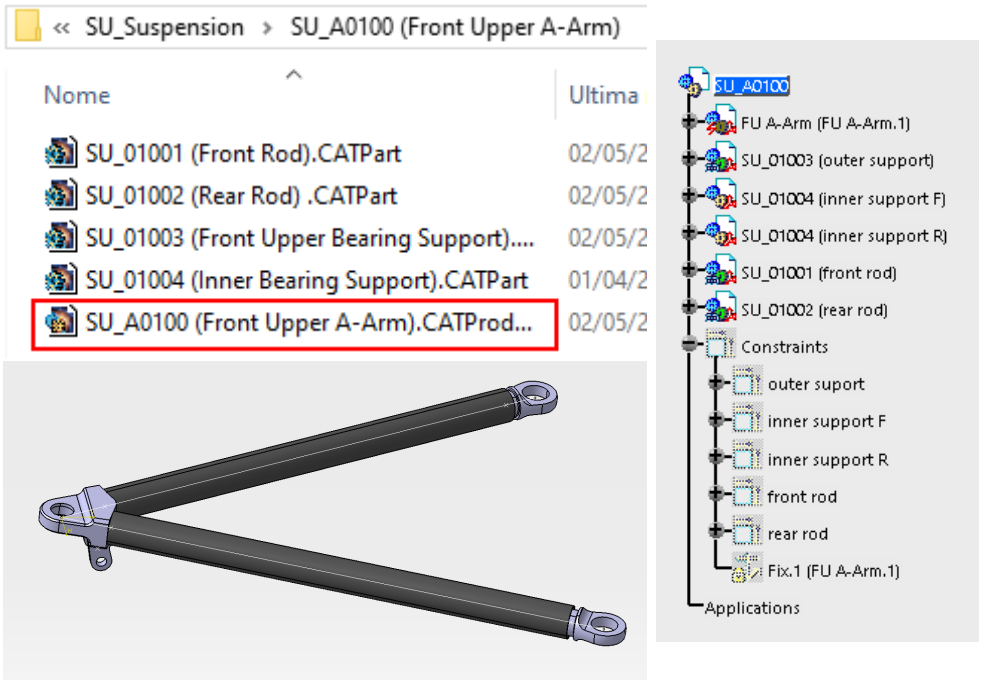

FIGURE 7 – Capture d'écran du produit pour la conception détaillée du triangle avant haut gauche

Nous remarquons que dans cette étape, l'utilisateur peut définir une orientation personnalisée des repères des pièces. Cela est très utile en phase de conception détaillé afin d'avoir des produits bien faits par rapport aux directions principales du véhicule. En effet, les axes du repère Lotus sont associés à différents grandeurs caractéristiques de la dynamique du véhicule et il est par conséquent impératif que les assemblages soient bien orientés.

5.2 Création des produits pour la conception détaillée

Dans un deuxième temps, on s'occupe de la création des fichiers pour la conception détaillée des pièces. Pour ce faire, on place, à la racine de chacun de ces produits, la pièce filaire correspondante et on la fixe avec une contrainte de fixation. Au fur et au mesure que l'on ajoute des pièces dans les produits, nous conseillons de suivre les règles suivantes :

- tout positionnement de pièce à l'intérieur d'un produit de conception détaillée doit se faire par rapport à la pièce filaire à la racine du produit
- ranger les contraintes de chaque pièce dans un sous-dossier et cacher l'affichage graphique des contraintes

Tout le long de la conception détaillée, on peut définir un fichier de paramètres de conception `params.xlsx` qui permettra à l'utilisateur de stocker de façon méthodique tous les paramètres inter-corrélés à l'intérieur des différents produits. Il s'agit de paramètres comme l'espacement des basculeurs ou le décalages des tubes des triangles par rapport au centre de la rotule : ces valeurs ont la caractéristique de changer très souvent pendant la période de conception du véhicule. L'exemple d'un triangle de suspension est présenté en Fig. 7.

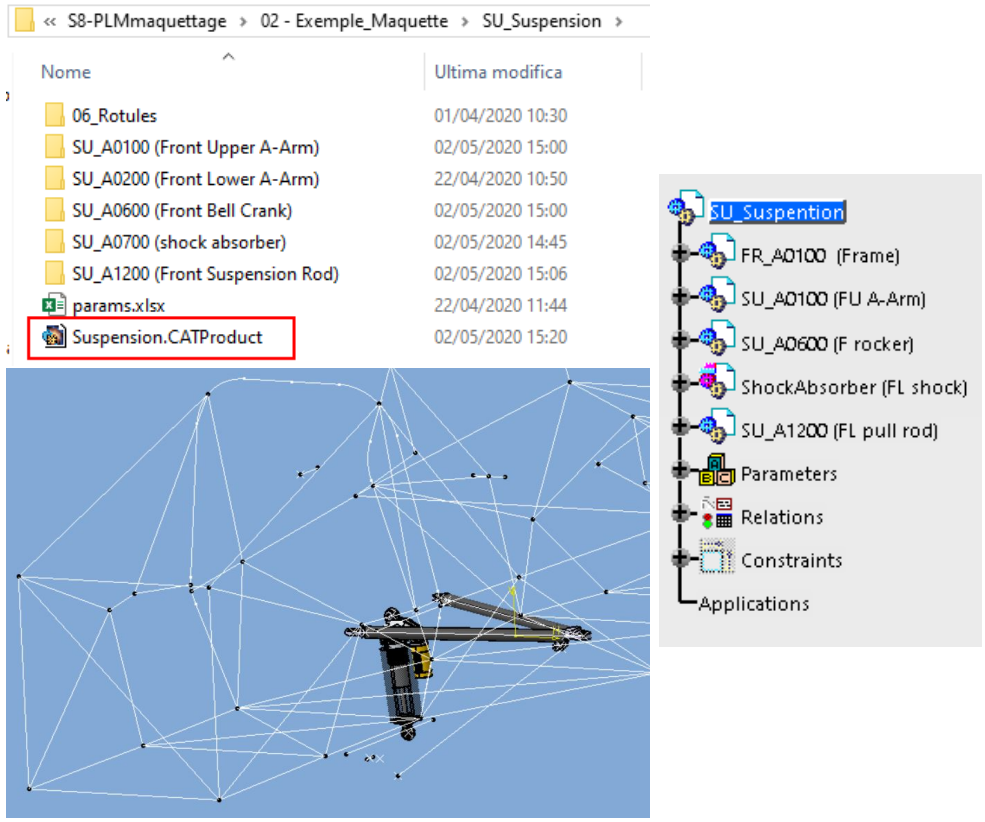

FIGURE 8 – Capture d'écran de l'Assemblage LAS qui est mis à jour par la macro

5.3 Création d'un assemblage global pour la Liaison au Sol

Enfin, nous ajoutons les produits de conception détaillée à l'intérieur d'un assemblage général pour la totalité du système de Liaison au Sol du véhicule (Fig. 8). Pour ce faire, on place en fixation globale le produit détaillé du châssis et on accroche graduellement tous les produits détaillés.

Afin de pouvoir étudier le mouvement et les collisions éventuelles entre les différentes organes du véhicule, certaines liaisons peuvent être paramétrées au niveau de l'assemblage `Suspension.CATProduct` : c'est l'exemple du débattement de l'amortisseur (paramètre `FL offset`) dans l'arbre du produit.

De même, certaines grandeurs caractéristiques (comme par exemple la voie et empattement) pourront être implémentées comme applications de type *mesure* dans l'arbre de ce produit global afin de pouvoir surveiller leur valeur tout le long de la conception.

5.4 Une proposition pour l'assemblage châssis

Afin de démontrer la versatilité de l'architecture choisie, il est également possible d'insérer dans des assemblages d'autres sous-systèmes, certains filaires de la LAS. En effet, un véhicule comprend de nombreuses interactions, et la prise en compte de ces interfaces fonctionnelles est primordiale.

Nous proposons donc le placement du filaire des triangles de suspension et de la colonne de direction dans celui du châssis. Ces filaires seront mis à jour par la macro.

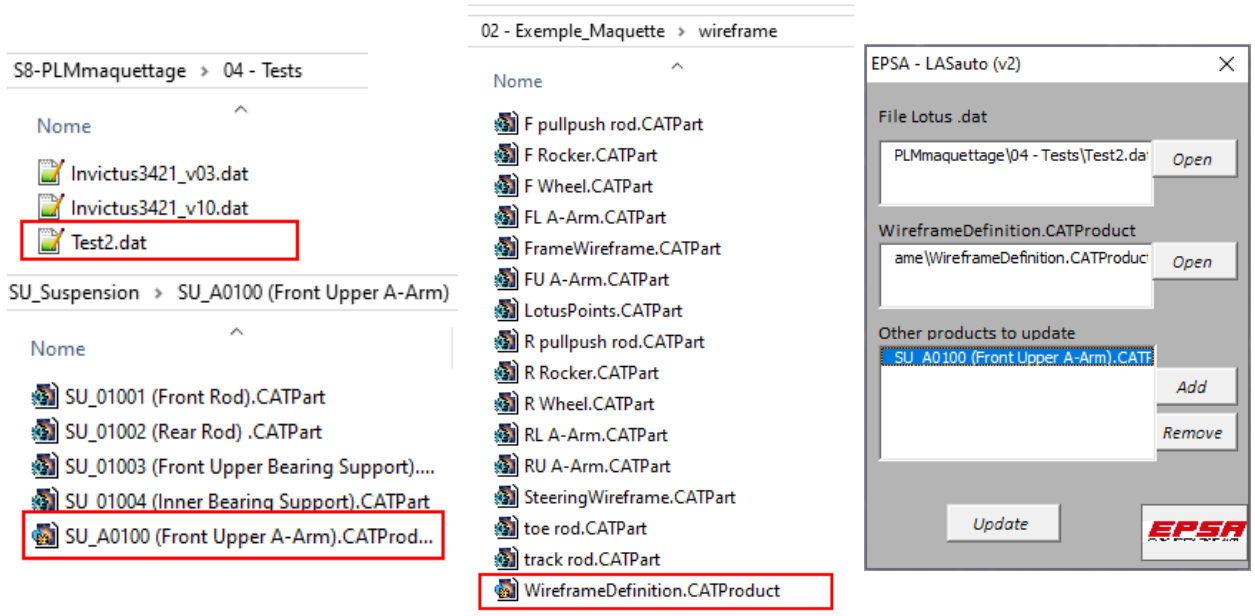

FIGURE 9 – exemple de test de la macro

5.5 Mise en application

Voici les étapes à suivre pour tester la Macro, un exemple est donné en Fig. 9

1. Installez la macro comme spécifié dans le guide d'installation, présent dans la section A.2.
2. Lancez la macro comme indiqué dans ce même guide.
3. Sélectionnez le fichier Lotus `Test1.dat`, rangé dans le dossier *04 - Tests*
4. Sélectionnez le produit `WireframeDefinition`, dans le dossier *02 - Exemple_ Maquette > wireframe*
5. Ajoutez les produits que vous souhaitez mettre à jour dans le dossier *02 - Exemple_ Maquette*, par exemple le fichier `SU_A0100 (Front Uper A-Arm).CATProduct`
6. Mettez à jour. Vous aurez normalement des messages d'informations sur le contenu de la mise à jour.
7. Fermez la macro et observez l'assemblage `Suspension`
8. Relancez la macro et répétez les étapes précédentes la concernant, en choisissant cette fois ci le fichier `Test2.dat` dans le dossier *04 - Tests*. Vous devriez voir notamment que vos choix précédents ont été mis en mémoire par la macro et que vous n'aurez pas à sélectionner à nouveau `WireframeDefinition.CATProduct` et `Suspension.CATProduct`.
9. Mettez à jour et observez l'assemblage `Suspension`. Vous pourrez noter que l'architecture a bel et bien été mis à jour.

6 Conclusion

Le but de ce projet était donc de permettre la mise à jour simple et rapide d'un Assemblage Catia portant sur une liaison au sol, dimensionnée cinématiquement sur un logiciel annexe nommé Lotus. En passant par de la programmation VBA, ce résultat est rempli et permet à l'utilisateur une utilisation ergonomique, à l'aide d'un UserForm simple et épurée. La programmation VBA et les fonctionnalités que nous avons incorporé permettent une automatisation, une répétabilité et une versatilité conséquente. Vous trouverez ci-dessous les innovations futures de *LASAuto*, que nous n'avons pas pu implémenter :

Développements futurs

1. Faire un parallèle de l'assemblage `Suspension` avec le module MÉCAMASTER.
2. Optimiser le rangement des système non symétriques de la voiture (exemple de l'architecture Ackermann pour le système de direction).
3. Filtrer les points de type *centre de gravité (CG)* dans Lotus.
4. Enlever les bugs qui apparaissent lorsqu'un Product ne veut pas se mettre à jour.

A Annexe

A.1 Paramétrage CATIA pour les références entre pièces

La fig. 10 présente les options à cocher afin d'avoir une bonne transmission des références entre les pièces des assemblages.

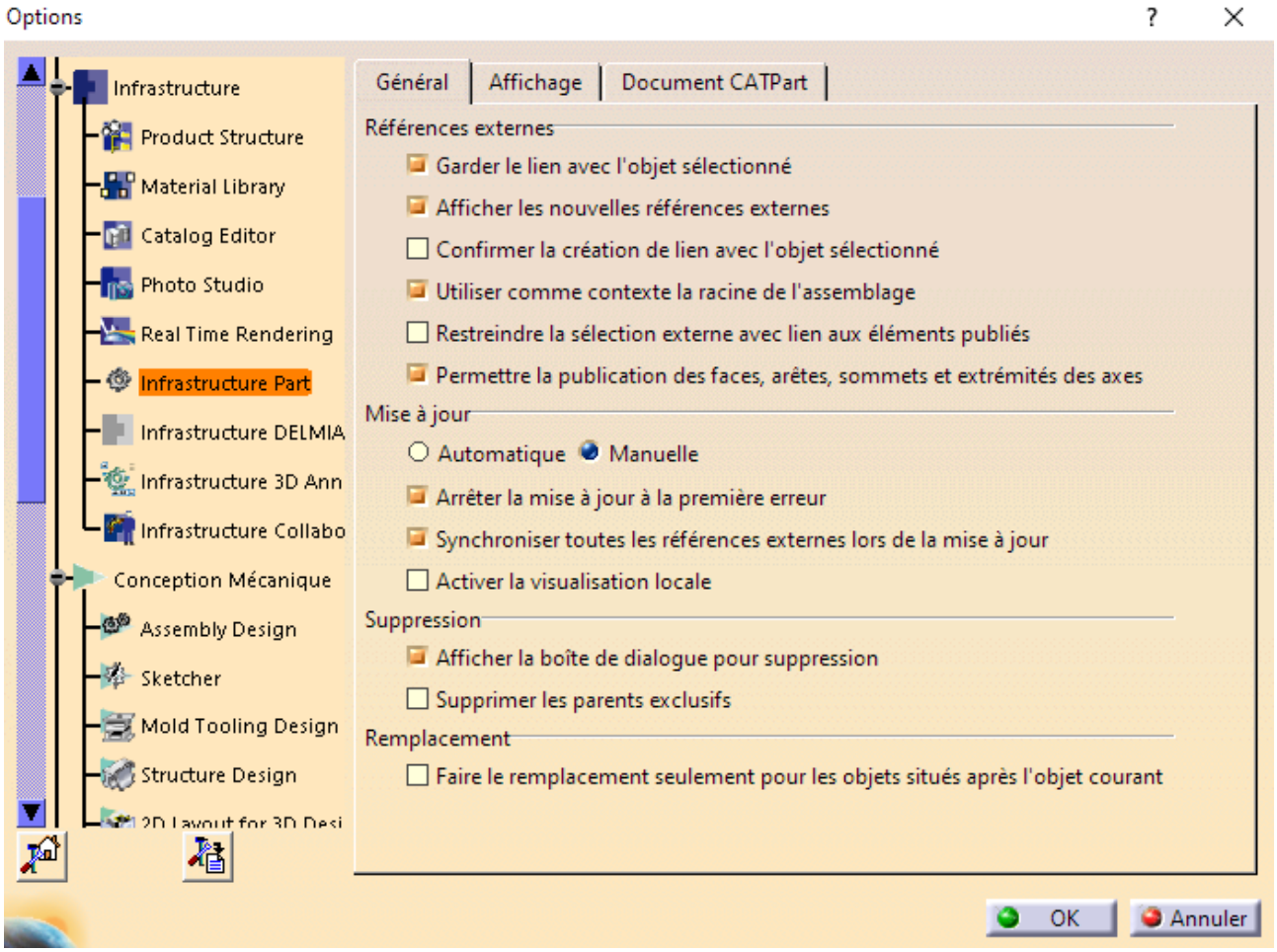

FIGURE 10 – Préférences Catia conseillées

A.2 Installation de la Macro

Vous trouverez, dans les pages suivantes, la démarche nécessaire afin d'installer la macro dans le répertoire de travail de la suspension d'un nouveau véhicule.

Guide de survie pour la macro EPSA – LAS Auto :

Introduction

Le but de ce document est d'aider à l'installation et l'utilisation de la macro d'automatisation des itérations LAS à l'EPSA.

Cette macro a été réalisée en 2020 et est une version automatisée pour la MAJ des points LAS en utilisant Lotus. Cette macro utilise donc de la programmation VBA et fait suite au 1er LASAuto de l'EPSA qui reposait sur l'utilisation d'un Catalog.

Eléments à respecter :

- Ne pas avoir déplacer de documents qui sont dans le dossier mère de la macro
- Bien avoir construit le Product *Suspension.CATProduct* comme indiqué dans le rapport

Ajouter la macro à la bibliothèque

Il est nécessaire d'ajouter la macro à la bibliothèque des macros de CATIA.

Aller dans **Outils** -> **Macro** -> **Macros**

Cette fenêtre s'ouvre :

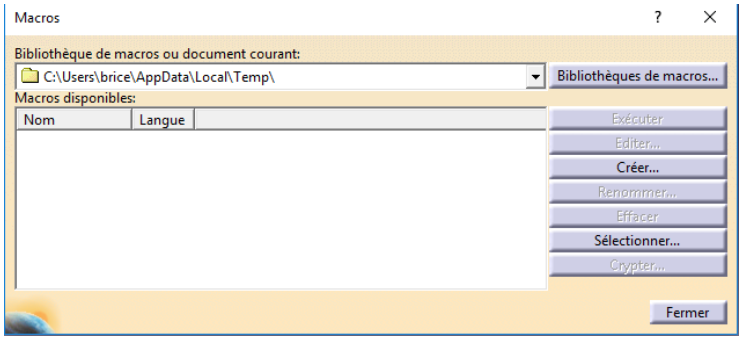

Figure 1 : Fenêtre des macros

Cliquer sur **Bibliothèques de macros** -> sélectionner Projets VBA dans **Types de bibliothèque**

Cliquer sur **Ajouter une bibliothèque existante** et sélectionner **LASAuto.catvba** dans le dossier \01 - VBAproject.

La macro d'automatisation des itérations LAS est maintenant dans votre bibliothèque.

Ajouter une barre d'outils contenant le bouton de lancement de la macro

1. Ajouter une nouvelle barre d'outils vide

Ouvrir l'atelier **Assembly Design**

Aller dans le menu **Outils** -> **Personnaliser** -> Onglet **Barre d'outils** -> Bouton **Nouvelle**

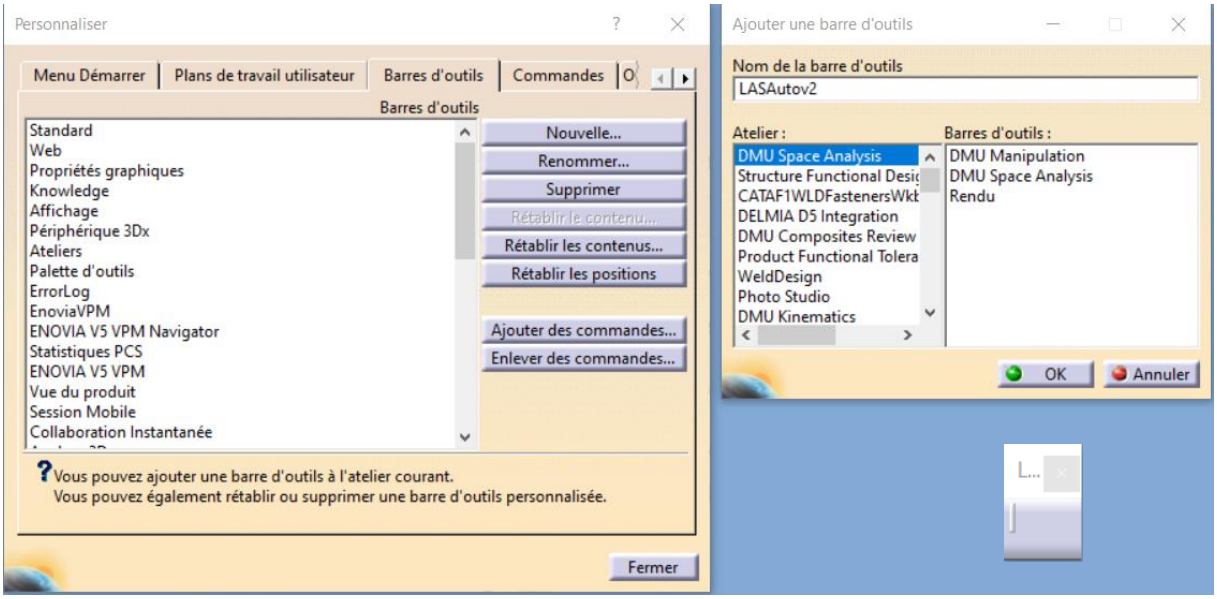

Figure 1 : Insertion d'une nouvelle barre d'outils

Ainsi, après avoir renommé votre barre d'outils et cliqué sur OK, vous devriez avoir une nouvelle barre d'outils vide.

Si vous ne la trouvez pas, clique droit sur n'importe quel espace vide de vos barres d'outils par défaut, cherchez là dans l'arbre, et cacher/afficher là pour la voir réapparaître.

2. Insérer le bouton de commande de la macro

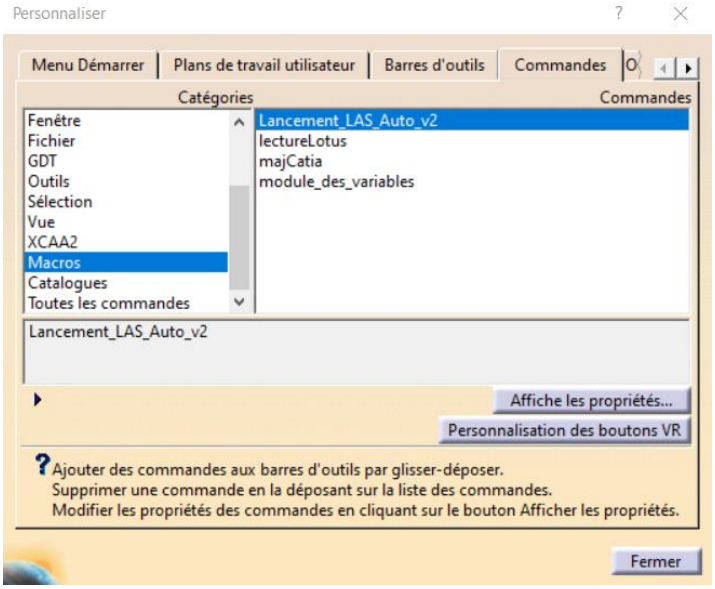

Figure 5 : Insertion du bouton de commande de la macro

Toujours dans **Outils** -> **Personnaliser,** allez dans l'onglet **Commandes** -> catégories **Macros**

Faites glisser **Lancement_LAS_Auto_v2** dans la nouvelle barre d'outils.

*(Il est possible de modifier l'image de ce bouton dans votre barre d'outils en cliquant sur **Affiche les propriétés** quand vous avez sélectionné **Lancement_LAS_Auto_v2**, puis observer la zone **Icones.** Vous pouvez soit changer par un logo déjà existant dans Catia, soit insérer le logo que nous avons préparé à cet effet, mais cette possibilité avec le logo préparé ne fonctionne pas toujours selon les ordinateurs pour des problèmes de programmation interne à CATIA_V5. Le fichier prévu s'appelle Wheel_Pictogramme.bmp et se situe dans le dossier 01_VBAProject)*

Maintenant, si vous cliquez sur le bouton, la macro se lance et une fenêtre type 'Userform' va s'ouvrir, expliqué dans le paragraphe suivant.

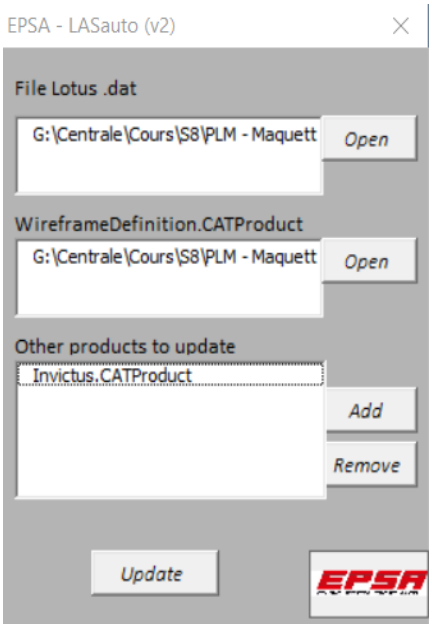

Figure 6 : Userform utilisé dans la macro

Si vous avez lu le rapport, vous savez que la macro permet de sélectionner le **fichier Lotus** avec la configuration LAS que vous souhaitez implanter, permet de sélectionner le **Wireframe** qui contient **Lotus_Point** qui lui-même contient les points Lotus, et que vous pouvez sélectionner **d'autres produits** que vous souhaitez updater.

Normalement, l'Userform est bien pensé et les boutons sont parlants : les boutons *Open* respectifs permettent de sélectionner le fichier .dat de Lotus et le fichier Wireframe.CATProduct, et le bouton *Add* et *Remove* permettent d'ajouter ou de supprimer des CATProducts à mettre à jour. Vous pouvez ensuite appuyer sur le bouton *Update*.

Notez qu'il faut absolument avoir sélectionné un fichier Lotus et un Wireframe.CATProduct pour pouvoir updaté les points, mais que l'update d'autres Products n'est pas obligatoire

Ne pas renommer les points que vous utilisez dans Lotus_Point, sinon la macro ne les mettra pas à jour ! De plus, renommez les points également dans Lotus, c'est comme si vous créez un nouveau point dans Catia, donc ne les renommez pas !