

Année  
universitaire :  
2023-24

# Rapport de stage



**AKKODIS**

Moreau Alexandre  
IUT Dijon-Auxerre – Département  
Informatique  
Année universitaire : 2023-24

## TABLE DES MATIERES

<b>1</b>	<b>INTRODUCTION.....</b>	<b>3</b>
<b>2</b>	<b>PRESENTATION GENERALE .....</b>	<b>4</b>
2.1.	PRESENTATION DE L'ENTREPRISE .....	4
2.2.	PRESENTATION DU SERVICE .....	4
2.3.	DEFINITION DE LA MISSION .....	5
<b>3</b>	<b>PRESENTATION DU TRAVAIL.....</b>	<b>6</b>
3.1.	OUTILS ET TECHNOLOGIES UTILISES .....	6
	Visual Studio 2019 .....	6
	TortoiseSVN.....	6
3.2.	PRESENTATION DES DIFFERENTES MISSIONS REALISEES .....	7
a.	<i>Les différentes phases de la réalisation.....</i>	<i>7</i>
b.	<i>Difficultés rencontrées.....</i>	<i>10</i>
3.3.	CONCLUSION.....	11
<b>4</b>	<b>ANNEXES .....</b>	<b>12</b>
	<b>RESUME / ABSTRACT .....</b>	<b>18</b>

## TABLE DES ILLUSTRATIONS

FIGURE 1 : LOGO D'AKKODIS .....	4
FIGURE 2 : ARCHITECTURE DES SIMULATEURS .....	5
FIGURE 3 : LOGO VISUAL STUDIO .....	6
FIGURE 4 : LOGO DE TORTOISESVN.....	6
FIGURE 5 : IMAGE D'UNE CLE DANS UN MESSAGE.....	7
FIGURE 6 : IMAGE D'UNE ALTERNATIVE A RECHERCHER .....	8
FIGURE 7 : IMAGE DE L'ALTERNATIVE APRES LE PASSAGE DE MON PROGRAMME .....	8
FIGURE 8 : IMAGE DE LA BARRE DE PROGRESSION .....	9
FIGURE 9 : IMAGE DE L'AIDE .....	9
FIGURE 10 : IMAGE DU EXCEL OBTENU .....	9
FIGURE 11 : ORGANISATION DES TACHES .....	10
FIGURE 12 : DIAGRAMME DE CLASSE .....	16
FIGURE 13 : CAS PARTICULIER RENCONTRE.....	17
FIGURE 14 : RESOLUTION DU CAS PARTICULIER .....	17

## REMERCIEMENTS

Je remercie tout d'abord l'entreprise Akkodis pour m'avoir accueilli au sein de son établissement de Blagnac afin de mener à bien mon stage.

Je remercie M Cyrille BOUDIAS, chef de département Digital Learning et Immersive Solutions et mon tuteur au sein de l'entreprise, qui m'a efficacement encadré durant cette période. Sa disponibilité, sa pédagogie et ses conseils ont guidé ma démarche de travail et participé à la réussite de mon projet.

Mes remerciements s'adressent également à l'ensemble de l'équipe pédagogique de l'IUT de Dijon, pour la qualité de leur enseignement dispensé tout au long de la formation.

# 1 INTRODUCTION

Durant la période allant du 29 janvier au 22 mars 2024, j'ai eu le privilège d'effectuer un stage au sein de l'entreprise Akkodis située au 7 Boulevard Henri Ziegler à Blagnac (31). Ce stage, d'une durée de 8 semaines, s'inscrit dans le cadre de ma formation à l'IUT de Dijon. Cette expérience m'a permis de mettre en pratique les connaissances acquises au cours de mes études, tout en découvrant le fonctionnement quotidien d'une entreprise évoluant dans le secteur du développement de simulateurs de conduite.

Ce rapport vise à présenter mon expérience au cours de cette période. Après avoir exposé le contexte de l'entreprise, je détaillerai les objectifs que j'ai poursuivis dans l'entreprise, avant de présenter les réalisations faites durant le stage.

## 2 PRESENTATION GENERALE

### 2.1. PRESENTATION DE L'ENTREPRISE

Akka Technologies, est un groupe d'ingénierie et de conseil en technologies, fondé par Maurice Ricci en 1999 et dont le siège social est situé à Bruxelles, Belgique.

Le groupe est positionné sur l'ensemble des secteurs d'activités industriels et tertiaires, à savoir notamment aéronautique, automobile, défense, énergie, ferroviaire, spatial, système d'information et télécommunications.

Le groupe dispose de sociétés implantées dans 30 pays du monde dont Belgique, France, Allemagne, Espagne, Inde, Italie, Maroc, Roumanie, Royaume-Uni, Chine, Amérique du Nord et Suisse.



Figure 1 : Logo d'Akkodis

En juillet 2021, le groupe Franco-Suisse Adecco annonce l'acquisition d'Akka Technologies pour 2 milliards d'euros. En mars 2022, Modis, l'activité de services de haute technologies du groupe Adecco, a été combinée avec Akka Technologies pour devenir une marque commerciale internationale dénommée Akkodis.

Les principaux actionnaires d'Akkodis sont la famille Ricci avec 38.5% ainsi que la Compagnie Nationale à Portefeuille (société d'investissement et holding belge) avec 21.4%.

En 2021, le chiffre d'affaires d'Akkodis France s'élève à 182 millions d'euros. Akkodis France est dirigée par Yves-Marie Boissonnet depuis 2023.

Akkodis France est composée d'une équipe de 9 000 employés, participant ainsi à la présence mondiale d'Akkodis qui compte au total 50 000 employés à travers le monde.

### 2.2. PRESENTATION DU SERVICE

Le service Akkodis Digital Learning et Immersive Solutions, étant autrefois la société Operantis, est spécialisée dans la conception et le développement de produits de formations multimédias dédiés aux grands industriels.

Les solutions développées s'inscrivent dans une logique de formation permettant aux apprenants d'atteindre des objectifs pédagogiques.

Akkodis Digital Learning et Immersive Solutions est un des acteurs majeurs de l'apprentissage par les nouvelles technologies et la simulation. Akkodis Digital Learning et Immersive Solutions a développé sa propre plateforme de simulation pour modéliser différents systèmes réels (aviation, ferroviaire...).

## 2.3. DEFINITION DE LA MISSION

Pour la création des différents simulateurs de conduite, Akkodis possède une base commune qui est la plateforme OSK (Operantis Simulateur Kernel). Cette plateforme OSK est dotée d'un système de messagerie Serveur-Client qui permet au client, tel que la partie 3d, la partie pédagogie ou autres, et au serveur de communiquer entre eux.

Ce système de messagerie est fonctionnel mais peut être optimisé et rationalisé. Il est nécessaire également de connaître par qui sont envoyés et reçus les différents messages et s'ils sont utilisés.

Je me vois donc confier la tâche de créer un outil qui modifie le programme existant pour optimiser et rationaliser le système de messagerie et qui recense les différentes informations des messages. Ce programme doit pouvoir être appliqué sur tous les simulateurs d'Akkodis.

Ce programme devra être utilisable sur l'invite de commande.

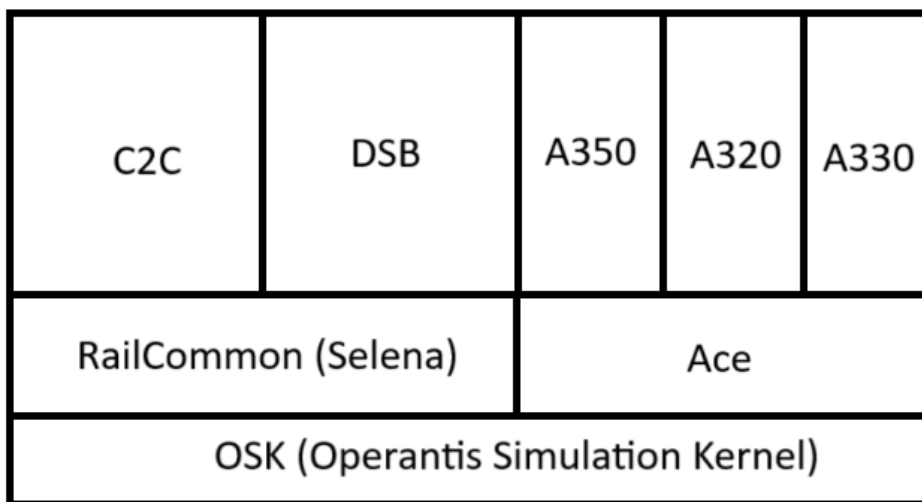


Figure 2 : Architecture des simulateurs

## 3 PRESENTATION DU TRAVAIL

### 3.1. OUTILS ET TECHNOLOGIES UTILISES

Afin de réaliser le développement, j'ai eu recours à un ordinateur portable fourni par l'entreprise, sous Windows 10, relié au réseau Ethernet de l'entreprise.

Concernant les logiciels utilisés, pour l'ensemble du stage, ils ont tous été installés sur l'ordinateur ; de ce fait, je n'ai utilisé que des logiciels fournis par l'entreprise. En voici la liste :

- Développement de l'application : Visual Studio 2019 professionnel
- Logiciel de gestion de version : TortoiseSVN

#### Visual Studio 2019



Figure 3 : Logo Visual Studio

Visual Studio est un ensemble complet d'outils de développement permettant de générer des applications web ASP.NET, des services web XML, des applications bureautiques et des applications mobiles.

Visual Basic, Visual C++, Visual C# utilisent tous le même environnement de développement intégré (IDE), Visual Studio Code, qui leur permet de partager des outils et facilite la création de solutions faisant appel à plusieurs langages.

Par ailleurs, ces langages permettent de mieux tirer parti des fonctionnalités du *framework*<sup>1</sup> .NET

#### TortoiseSVN



Figure 4 : Logo de TortoiseSVN

TortoiseSVN est un logiciel populaire, client de SVN<sup>2</sup>.

En s'intégrant dans l'explorateur de fichier de Windows, il offre aux utilisateurs de Windows une interface graphique permettant de réaliser la plupart des tâches qu'offre SVN en ligne de commande.

L'explorateur Windows s'enrichit des fonctionnalités suivantes :

Superposition d'icônes aux répertoires et fichiers permettant de visualiser instantanément l'état (à jour, modifié, en conflit...)

Menu contextuel permettant de committer<sup>3</sup> ou d'updater<sup>4</sup>, à l'échelle d'un fichier, d'une sélection de fichiers, ou encore d'un répertoire.

<sup>1</sup> Un *framework* (cadre de travail) est une bibliothèque logicielle facilitant le développement d'applications.

<sup>2</sup> SVN est un logiciel qui permet de stocker un ensemble de fichiers en conservant la chronologie de toutes les modifications qui ont été effectuées dessus.

<sup>3</sup> Committer permet d'envoyer sur un serveur les changements qui ont été effectués.

<sup>4</sup> Updater permet de récupérer sur un serveur les données qui ont pu être modifiées par un tiers.

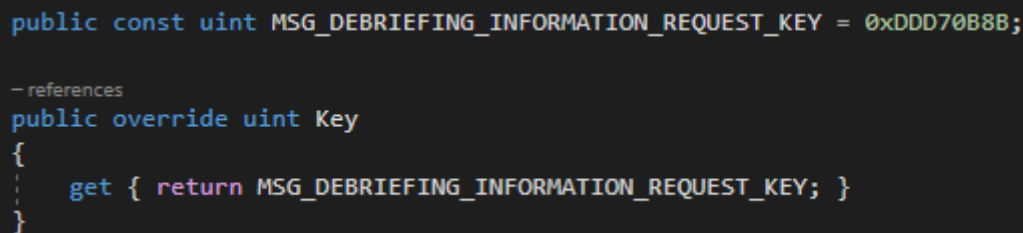
## 3.2. PRESENTATION DES DIFFERENTES MISSIONS REALISEES

### a. Les différentes phases de la réalisation

Après mon installation dans le service, on m'a présenté le sujet sur lequel j'allais travailler, qui consiste à modifier des fichiers de code pour qu'il soit optimisé et de recenser les différents messages du système de messagerie.

J'ai donc débuté par un programme en C# sur Visual Studio 2019 qui allait modifier les fichiers de code. Mon maître de stage m'a conseillé de récupérer les noms des classes à modifier depuis les DLL (Dynamic Link Library) grâce à la réflexion qui permet à un code de lire les DLL. Il faut savoir que tous les messages héritent de la classe « MSGBase » et que la réflexion me permet de connaître pour une classe particulière de quelle classe elle hérite. Donc j'ai pu récupérer tous les noms des classes qui héritent de « MSGBase ».

Il me fallait maintenant chercher dans les fichiers de code les classes dont j'avais trouvé le nom et pour ce faire j'ai utilisé les Expressions Régulières. Donc j'ai parcouru tous les fichiers, ligne par ligne, pour trouver l'emplacement ainsi que la ligne de début et de fin de la classe. Pour ce faire j'ai utilisé quelques outils que j'ai codé tels que : un outil qui me permettait de savoir la fin d'une classe, une fonction qui trouve tous les fichiers présents dans un chemin avec une extension précise, et aussi une fonction qui me permet de récupérer les noms des classes trouvées. Ensuite j'ai créé les clés pour les messages avec Adler32, une fonction de hachage, sous le format suivant :



```
public const uint MSG_DEBRIEFING_INFORMATION_REQUEST_KEY = 0xDDD70B8B;

- references
public override uint Key
{
    get { return MSG_DEBRIEFING_INFORMATION_REQUEST_KEY; }
}
```

Figure 5 : Image d'une clé dans un message

Après avoir créé les clés, j'ai conçu une fonction qui ajoute les clés dans les fichiers des classes correspondantes.

Pour faire l'ajout, il me fallait plusieurs outils tels que, un outil qui décale toutes les classes qui sont à modifier présentes dans le même fichier par le nombre de lignes qui ont été ajoutées, et une fonction qui indente l'ajout pour que la lecture du code soit plus agréable. J'ai également réalisé une fonction qui permet d'ajouter du contenu à une ligne précise.

J'ai ensuite ajouté une classe « ClassToUpdate » qui contient la classe à modifier et au lieu d'ajouter la clé directement au fichier, j'enlève l'ancienne classe et je la remplace par la classe que j'ai modifiée avec la nouvelle clé.

J'ai ensuite ajouté une fonction qui ignore les commentaires présents dans le code pour que mon programme ne prenne pas un commentaire comme étant une classe à modifier.

Après m'être occupé des classes, je me suis occupé des fonctions dans le code qui utilisent les messages modifiés précédemment. Pour cela j'ai commencé par ajouter une classe « FunctionToUpdate » qui contient la fonction à modifier et j'ai commencé à faire la recherche



des fonctions à modifier grâce aux Expressions Régulières pour trouver le début et la fin des alternatives qui sont sous le format suivant :

```
if (msg is MSGLessonSelectionRequest)
{
    RaiseOnLessonSelectionRequest(registerKey);
}
if (msg is MSGLessonInitializationExtendedIsRequired)
{
    RaiseTaskFolderOffsetIsRequiredBy(registerKey);
}
else if (msg is MSGEditorRequestSave)
{
    RaiseEditorRequestSave(msg);
}
```

Figure 6 : Image d'une alternative à rechercher

J'ai ensuite réalisé une méthode pour ajouter des switch case dans les fonctions correspondantes. Pour créer les switch case, il fallait récupérer le contenu du « if » et à quel message il est attribué, pour arriver au résultat suivant :

```
switch(msg.Key)
{
    case MSGLessonSelectionRequest.MSG_LESSON_SELECTION_REQUEST_KEY:
    {
        RaiseOnLessonSelectionRequest(registerKey);
    }
    break;

    case MSGLessonInitializationExtendedIsRequired.MSG_LESSON_INITIALIZATION_EXTENDED_IS_REQUIRED_KEY:
    {
        RaiseTaskFolderOffsetIsRequiredBy(registerKey);
    }
    break;

    case MSGEditorRequestSave.MSG_EDITOR_REQUEST_SAVE_KEY:
    {
        RaiseEditorRequestSave(msg);
    }
    break;
}
```

Figure 7 : Image de l'alternative après le passage de mon programme

J'ai ensuite commencé la création du fichier Excel qui regroupe les informations des messages, comme leurs noms, leurs fichiers et leurs emplacements dans les fichiers.

J'ai ensuite fait la gestion des événements sur le même principe que les messages, en sachant que tous les événements héritent de la classe « SimulationContextClientBaseEvent ». Et avec cette méthode de recherche basé sur les Expressions Régulières, j'ai pu trouver les fonctions, qui utilisent les événements qui ont été modifiés, auxquels je rajoute les switch case.

Puis j'ai ajouté une classe « ToUpdate » qui est héritée par « ClassToUpdate » et « FunctionToUpdate » pour faciliter et uniformiser l'ajout aux classes et aux fonctions.

J'ai ensuite repris ma fonction de recherche pour que le programme puisse trouver toutes les fonctions qui utilisent les événements et tous les messages. Pour ce faire, je regarde toutes les classes pour trouver les fonctions qui contiennent des mots clés choisis au préalable.

Je me suis ensuite occupé de la gestion de l’affichage pour qu’on puisse utiliser mon programme directement à partir de l’invite de commande. Pour gérer les arguments fournis depuis la console, j’ai créé une fonction qui lit ces arguments. J’ai également fait une fonction qui affiche une barre de progression pour savoir à quelle étape le programme en est.

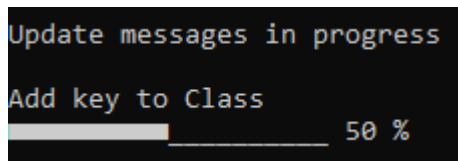


Figure 8 : Image de la barre de progression

J’ai aussi créé une aide pour savoir quel argument inscrire pour que le programme se lance, sous le format suivant :

```
C:\DEV\SVN\DSB\RailCommon\Kernel\src\TOOLS\UpdateMSG\UpdateMSG\bin\Debug>UpdateMSG.exe /?
-dllDebugPath or -dllReleasePath [path dll folder]
-searchPath [folder search path]
-update [all : Update messages and events | messages : Update only messages | events : Update only events | excel : Create excel with messages information]
-clean : Remove all "#if USE_KEYS" in cs and csproj
-help or /? for help
```

Figure 9 : Image de l'aide

J’ai également géré des exceptions, qui par exemple envoient l’aide en cas de problème dans la commande pour lancer le programme.

J’ai ensuite ajouté des directives préprocesseurs aux fonctions que j’avais modifiées et au fichier csproj qui contenait la classe avec la fonction modifiée pour pouvoir utiliser les directives préprocesseurs. Dans ce but, j’ai ajouté une classe « ProjToUpdate » qui hérite de « ToUpdate », et j’ai également fait une méthode « Clean » pour enlever les directives préprocesseurs.

Puis j’ai repris mon fichier Excel pour ajouter des informations telles que savoir par qui est émis le message et qui le reçoit entre le Serveur et le Client. Celui-ci se présente sous ce format :

Nom du MSG	Fichier du MSG	Ligne de debut	Emmeteur	Recepteur
MSGActionCompleted	C:\DEV\SVN\DSB-Tests\RailCommon\Kernel\src\VPT.RTDS\Operantis.VPT.RTDS.SimulationContextControlLib\MSGActionCompleted.cs	21	Server	Client
MSGCancelAutomaticNavRequest	C:\DEV\SVN\DSB-Tests\RailCommon\Kernel\src\VPT.RTDS\Operantis.VPT.RTDS.SimulationContextControlLib\MSGAutomaticNav.cs	21	Server	Client
MSGCallTrainer	C:\DEV\SVN\DSB-Tests\RailCommon\Kernel\src\VPT.RTDS\Operantis.VPT.RTDS.SimulationContextControlLib\MSGCallTrainer.cs	21	Server	Client

Figure 10 : Image du Excel obtenu

J’ai également dû fournir une documentation qui explique comment utiliser mon programme.

(Voir Annexe page 12)

Le diagramme ci-dessous décrit les tâches effectuées durant mes 8 semaines de stage :



Figure 11 : Organisation des tâches

## b. Difficultés rencontrées

La plus grande difficulté rencontrée lors de cette mission a concerné la réalisation d'une fonction de recherche des différents fichiers à modifier, dû au fait qu'il y avait un grand nombre de fichiers à modifier, près de cinq cents fichiers, et qu'il fallait trouver leurs points communs pour que mon programme sache les reconnaître.

Il a été aussi difficile d'ajouter les modifications effectuées dans les classes ou les fonctions, car elles devaient être ajoutées à une ligne spécifique de la fonction ou de la classe, tout en ne réécrivant pas sur les autres fonctions ou classes.

J'ai su à chaque fois trouver des solutions efficaces à chacune des difficultés rencontrées. Cette expérience m'a permis de développer ma capacité à résoudre les problèmes de manière autonome, renforçant ainsi mes compétences professionnelles.

### 3.3. CONCLUSION

---

Le stage est une partie importante de la scolarité en BUT, car il permet à la fois de valider ses compétences techniques dans un cadre réel, et de découvrir le monde du travail en entreprise.

Cela s'est révélé être le cas car mes connaissances théoriques acquises en cours à l'IUT m'ont permis de réaliser les différentes tâches demandées et mener à bien mon projet de stage. De plus, appliquer les connaissances à une situation réelle est stimulant.

Pour ma part, le stage s'est très bien passé et j'ai beaucoup apprécié la mission et l'équipe. Il aurait été également intéressant d'être dans une équipe avec plusieurs développeurs, cela m'aurait en effet permis de davantage développer des compétences de travail en équipe, mais je ne regrette pas le choix du stage car le sujet était très intéressant.

## 4 ANNEXES

2024

# Documentation UpdateMSG

OUTIL D'OPTIMISATION, DE RATIONALISATION ET DE RECENSEMENT  
ALEXANDRE MOREAU

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Utilisation de l'outil</b>	<b>2</b>
2.1	Les différents arguments	2
2.2	La validation des modifications	3

## 1 Introduction

UpdateMSG est un outil d'optimisation, de rationalisation et de recensement pour le système de messagerie de la plateforme OSK.

L'outil utilise l'invite de commande pour que l'utilisateur puisse rentrer différents arguments.

## 2 Utilisation de l'outil

### 2.1 Les différents arguments

a) *Argument « -help » ou « / ? »*

Les arguments -help et / ? permet d'afficher tous les arguments utilisables pour l'outil et ce qui faut renseigner.

b) *Argument « -DLLDebugPath » ou « -DLLReleasePath »*

L'argument « -DLLDebugPath » permet de connaître les dossiers contenant les DLL qui ont été générés par la compilation en mode Debug, l'argument « -DLLReleasePath » fait la même chose mais dans les dossiers générés par la compilation en mode Release. Il faut leur renseigner le chemin du dossier des DLL.

c) *Argument « -SearchPath »*

L'argument « -searchPath » permet de connaître les dossiers qui doivent être modifiés par l'outil. Il faut lui renseigner le chemin du dossier.

d) *Argument « -Update »*

L'argument « Update » permet de savoir ce qui faut modifier, il a besoin pour fonctionner de au moins un Argument « -DLLDebugPath » ou « -DLLReleasePath » et de au moins un Argument « -SearchPath »

Quatre arguments sont possibles « all » qui modifie tous les messages et tous les événements, « messages » qui modifie seulement les messages, « events » qui modifie seulement les événements et « excel » qui génère un fichier Excel avec toutes les informations des messages.

En modifiant les messages ou les événements une directive préprocesseur va être ajoutée aux fonctions modifiées et au csproj qui sont rattaché aux fonctions pour pouvoir tester le nouveau code tout en gardant l'ancien code.

Exemple :

```
C:\DEV\SVN\DSB\RailCommon\Kernel\src\TOOLS\UpdateMSG\UpdateMSG\bin\Debug>UpdateMSG.exe -searchPath C:\DEV\SVN\DSB-Tests
-dllDebugPath C:\DEV\SVN\DSB-Tests\RailCommon\Kernel\src\VPT.RTDS\Operantis.VPT.RTDS.SimulationContextControlLib -update
excel _
```

```
C:\DEV\SVN\DSB\RailCommon\Kernel\src\TOOLS\UpdateMSG\UpdateMSG\bin\Debug>UpdateMSG.exe -searchPath C:\DEV\SVN\DSB-Tests
-dllDebugPath C:\DEV\SVN\DSB-Tests\RailCommon\Kernel\src\VPT.RTDS\Operantis.VPT.RTDS.SimulationContextControlLib -update
all
```

#### e) Argument « -clean »

L'argument clean permet d'enlever les directives préprocesseurs ajoutés lors de la modification et l'ancien code. Il a besoin de au moins un Argument « -SearchPath » pour fonctionner.

Exemple :

```
C:\DEV\SVN\DSB\RailCommon\Kernel\src\TOOLS\UpdateMSG\UpdateMSG\bin\Debug>UpdateMSG.exe -searchPath C:\DEV\SVN\DSB-Tests
-clean _
```

## 2.2 La validation des modifications

Après l'utilisation d'un argument update ou clean si des fichiers sont modifiés alors la liste des modifications et le nombre de modifications s'afficheront et vous aurez le choix de valider ou non, en marquant soit Y soit N, les modifications. Si vous valider, en marquant Y, les modifications s'appliqueront aux fichiers correspondants et un fichier apparaîtra avec toutes les fonctions qui ont été modifiées et différentes informations concernant les fonctions ou la recherche des fichiers a modifié.

```
Operantis.VPT.RTDS.ReviewRetryPanel.csproj C:\DEV\SVN\DSB-Tests\RailCommon\Kernel\src\VPT.RTDS\Operantis.VPT.RTDS.Review
RetryPanel\Operantis.VPT.RTDS.ReviewRetryPanel.csproj: 1 change
Operantis.VPT.RTDS.SerialCommunicationBridge.csproj C:\DEV\SVN\DSB-Tests\RailCommon\Kernel\src\VPT.RTDS\Operantis.VPT.RT
DS.SerialCommunicationBridge\Operantis.VPT.RTDS.SerialCommunicationBridge.csproj: 1 change
Operantis.VPT.RTDS.SoundPlayerLib.csproj C:\DEV\SVN\DSB-Tests\RailCommon\Kernel\src\VPT.RTDS\Operantis.VPT.RTDS.SoundPla
yerLib\Operantis.VPT.RTDS.SoundPlayerLib.csproj: 1 change
RailCommon.InstructorSupervision.csproj C:\DEV\SVN\DSB-Tests\RailCommon\RailCommon\RailCommon.InstructorSupervision\Rail
Common.InstructorSupervision.csproj: 1 change
Total number of changes: 493
Confirm changes: Y/N
y
Changes made
Duration : 186 s
```





Figure 12 : Diagramme de classe

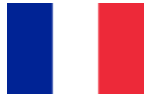
```
..}␣  
..else if (evt.isSimulationContextDeclareVarsEvents ||␣  
.....evt.isSimulationContextUpdateVarsEvents)␣  
..{␣  
.....OnVarsEvent((SimulationContextVarsEvents)evt);␣  
..}␣
```

Figure 13 : Cas particulier rencontré

```
case SimulationContextDeclareVarsEvents.SIMULATION_CONTEXT_DECLARE_VARS_EVENTS_KEY:␣  
{␣  
    OnVarsEvent((SimulationContextVarsEvents)evt);␣  
}  
break;␣  
case SimulationContextUpdateVarsEvents.SIMULATION_CONTEXT_UPDATE_VARS_EVENTS_KEY:␣  
{␣  
    OnVarsEvent((SimulationContextVarsEvents)evt);␣  
}  
break;␣
```

Figure 14 : Résolution du cas particulier

## RESUME / ABSTRACT



Ce rapport décrit mon stage de deuxième année de BUT informatique au sein de l'entreprise Akkodis, à Blagnac. Dans ce cadre, j'ai été chargé de développer une macro permettant de d'ajouter des clés à des classes spécifiques et de remplacer les alternatives « if else » en « switch case ». Durant ces 8 semaines, j'ai pu ainsi découvrir divers moyens de recherche dans les fichiers et réaliser la tâche demandée.



This report outlines my second-year internship in computer science at Akkodis company, located in Blagnac. In this context, I was tasked with developing a macro to add keys to specific classes and to replace "if else" alternatives with "switch case". Over the course of these 8 weeks, I was able to explore various methods of file searching and successfully accomplish the assigned task.