

# Uniformisation des informations Office 365 et Active Directory

**Projet Master Cyber 2025**

# Sommaire

## Table des matières

Sommaire.....	2
Introduction.....	3
a) Problématique.....	3
b) Motivation de ce projet.....	3
c) Projet déjà existant.....	4
d) Résultat Attendu.....	4
Conception et Organisation du travail (Planning).....	5
a) Découpage du Projet.....	5
b) Méthodologie d'Implémentation.....	5
c) Diagramme de Gantt.....	6
d) Schéma de Conception .....	6
Cahier des charges.....	7
Réalisation .....	8
Test /Validation/Vérification.....	8
Bug /Difficulté rencontré .....	13
Conclusion.....	15
Annexe :.....	16

## Introduction

### a) Problématique

Dans le monde professionnel, l'usage des outils technologiques et applications/solutions cloud est devenu essentiel au quotidien pour permettre la création et la productivité au sein des entreprises. Toutefois, les petites entreprises (PME et TPE) sont rares à disposer d'un service informatique ou d'environnement réseau complet lorsqu'ils ne sont pas dans le domaine de la tech.

Face à la diversification, les outils numériques pour les utilisateurs de ces entreprises, ces utilisateurs doivent mémoriser plusieurs identifiants de connexion pour diverses solutions (mot de passe Windows / Email / Solution Métier / VPN/ ...) et les entreprises doivent également créer de façon répétitive un accès utilisateur pour chaque application métier à l'arrivée d'un nouvel employeur.

Pour pallier ce problème, une liaison LDAP peut être mise en place afin d'uniformiser les mots de passe entre les différents systèmes et de permettre la création commune d'un utilisateur sur l'environnement réseau. Cependant, dans les environnements où cette intégration n'a pas été envisagée dès le départ, la mise en œuvre peut s'avérer complexe/ardue en raison des conflits de synchronisation et d'informations. Problème dont je suis confrontée au quotidien dans mon environnement professionnel avec mes clients.

Ce projet a pour but de créer un programme/outils qui permettra d'uniformiser et de résoudre les problèmes de conflits entre l'un des principaux Annuaire (Active Directory) et l'un des principaux systèmes de Messagerie (Office 365), tous deux développés par Microsoft afin de pouvoir les synchroniser entre eux à l'aide du connecteur Azure AD. Cet outil permettra la création d'un annuaire propre et centralisé pour l'interconnexion des logiciels métiers qui gravitent autour de l'environnement de l'entreprise au travers de la connexion LDAP.

À la fin de ce projet, nous devrions avoir un annuaire centralisé qui permettra d'avoir des identifiants identiques et synchronisés pour chaque application/solution métier d'un utilisateur. Ainsi que la création et suppression des utilisateurs, depuis le tenant Maître.

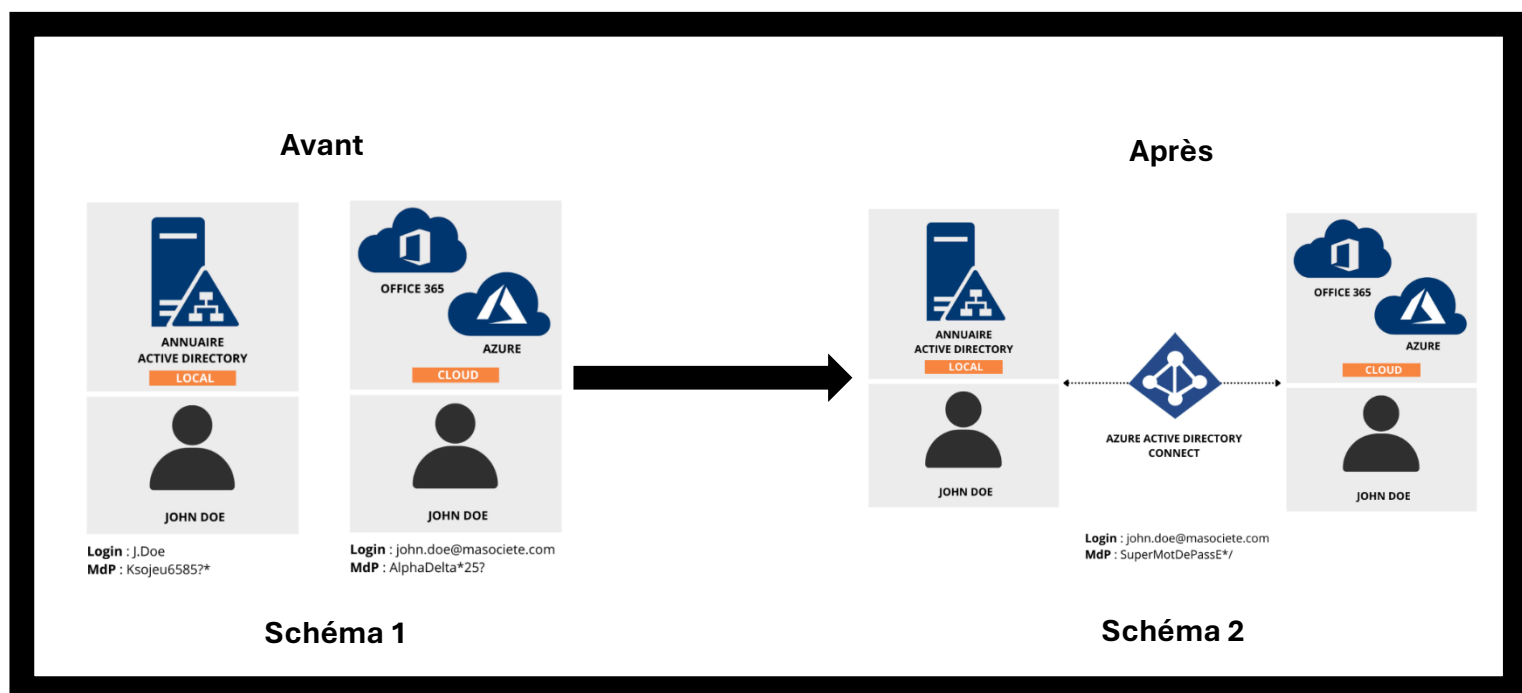
### b) Motivation de ce projet

Ma motivation principale dans ce projet réside d'être dans la continuité de mon travail en entreprise. Mon but étant de réussir à automatiser mon travail et résoudre la problématique vue en entreprise, pour finalement proposer une solution qui puisse être déployée/intégrée dans l'environnement d'un client et lui apporter une plus-value.

### c) Projet déjà existant

Il n'existe à ma connaissance qu'un seul projet pour permettre une synchronisation entre un Active Directory et un tenant Office 365, le connecteur Azure AD développer par Microsoft (renommé récemment Entra ID). Toutefois ce dernier souffre de lacune lorsque les 2 environnements sont déjà existants. Il existe bien des forums, des articles ou des tutoriels pour résoudre de manière individuelle les problèmes de synchronisation d'un utilisateur en fonction du paramètre/champs problématiques. Mais il n'existe aucun dépôt gît proposant une solution qui permettrait d'automatiser la résolution du problème de synchronisation entre les 2 tenants existant pour une multitude d'utilisateurs. Cela sera donc le but de ce projet.

### d) Résultat Attendu



## Conception et Organisation du travail (Planning)

### a) Découpage du Projet

La solution sera développée en modules indépendants, permettant un suivi de l'avancement et une intégration progressive des fonctionnalités :

- **Interface Graphique (Main) :**  
*Durée estimée : 8 à 16 heures*  
Développement d'une interface graphique simplifiée.
- **Importation et Connexion aux Environnements :**  
*Durée estimée : 4 à 6 heures*  
Configuration des connexions et chargement des modules permettant la communication avec l'Active Directory et Office 365..
- **Récupération des Données :**  
*Durée estimée : 6 à 8 heures*  
Extraction des informations utilisateurs depuis Office 365 et l'Active Directory.
- **Comparaison, Harmonisation des données :**  
*Durée estimée : 8 heures*  
Analyse des données extraites afin d'identifier et corriger les incohérences pour uniformiser les informations et Mise à jour des deux environnements avec les données harmonisées.
- **Gestion de la Synchronisation et des Erreurs (Bonus) :**  
*Durée estimée : 3 à 4 heures*  
  
Lancement de la synchronisation complète avec retour d'erreurs potentielles. Chaque étape fera l'objet de tests préliminaires avant la synchronisation définitive.

### b) Méthodologie d'Implémentation

Pour garantir la robustesse et la sécurité de la solution avant son déploiement en production, une approche expérimentale sera adoptée par la mise en place :

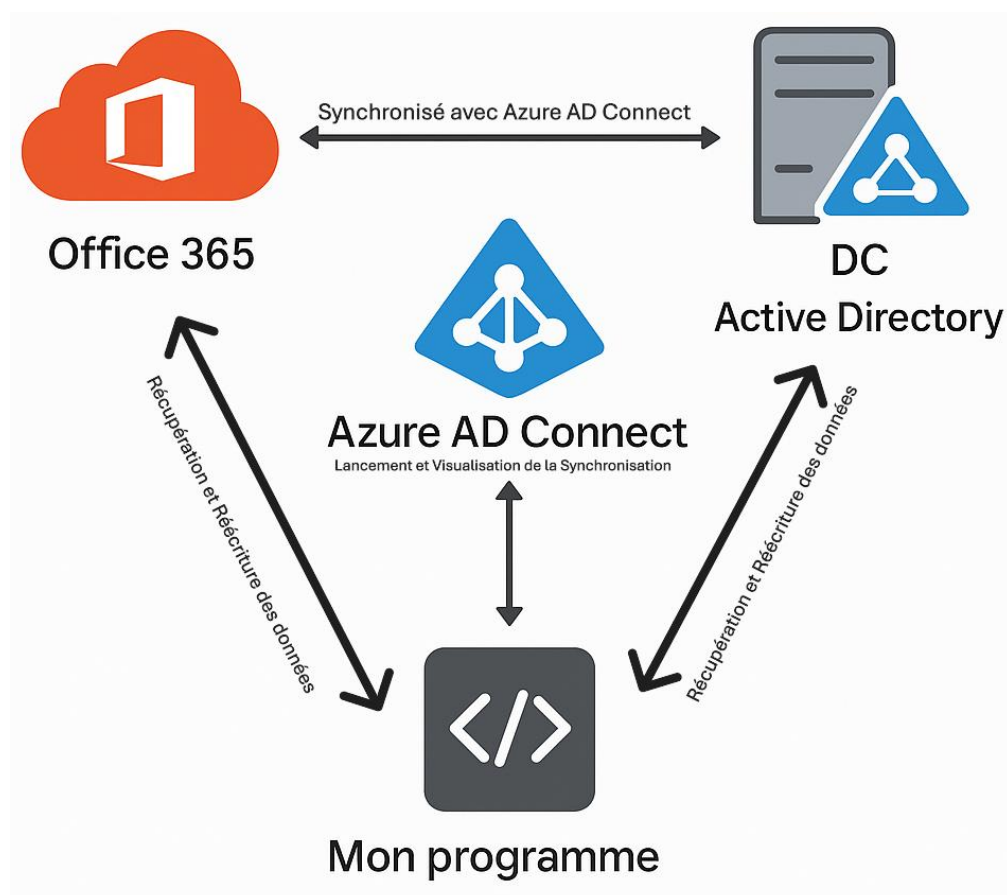
- **Un Tenant Office 365**
- **Une Machine Virtuelle (VM) :**  
Installation d'une VM hébergeant un Active Directory et le connecteur Azure AD, permettant de simuler un environnement de production pour réaliser des tests en conditions réelles.

### c) Diagramme de Gantt

## Project Master Diagramme de Gantt



### d) Schéma de Conception



## Cahier des charges

Critère	Description	Exigence Spécifique
<b>Exécution sur Serveurs Windows</b>	La solution doit être directement exécutable sur des serveurs Windows, sans nécessiter l'installation de logiciels tiers qui pourraient compromettre la stabilité ou la sécurité du système.	Aucune installation tierce requise
<b>Utilisation de PowerShell</b>	La mise en œuvre reposera sur PowerShell, le langage natif de Windows, permettant une interaction directe avec Active Directory et le tenant Office 365 grâce aux modules appropriés.	Utilisation exclusive de PowerShell
<b>Sécurité et Fiabilité</b>	Pour éviter toute corruption des données critiques (AD et Office 365), des mécanismes de sécurité seront intégrés. Cela inclut la confirmation avant réécriture des données.	Empêcher le plantage d'un environnement en production
<b>Gestion des Erreurs et des Exceptions</b>	La solution intégrera une gestion des erreurs et exceptions, avec la génération de rapport .csv/.txt et/ou logs détaillés afin de faciliter l'identification des erreurs et de faciliter les interventions manuelles.	Rapports d'erreurs en CSV et/ou logs
<b>Interface Graphique Simple</b>	Une interface graphique simple pour naviguer entre les différentes fonctionnalités du programme.	Interface utilisateur intuitive
<b>Extensibilité</b>	L'outil sera conçu pour pouvoir intégrer ultérieurement de nouvelles fonctionnalités (gestion automatique des licences, création d'alias, etc....).	Modularité prévue / Ouvert à l'ajout de fonctionnalité

## Réalisation

Je détaillerais ci-dessous des parties de mon code que je trouve pertinente et qui sont représentatif de mon rendu final. Je vais cependant détailler ici quelques points pour comprendre le fonctionnement de mon programme.

Dans un premier temps, ce programme est lancé à partir d'une session administrateur d'un serveur DC possédant un active directory avec un Shell lui aussi en administrateur et les polices de sécurité en mode non restreint grâce à cet commande : « Set-ExecutionPolicy Unrestricted ».

De plus mon script a besoin de 2 fichiers annexe pour fonctionner, un fichier Configuration.txt qui contient 4 information :

- Le nom de domaine Office365 qui sera appliqué aux utilisateurs de l'AD (car on ne peut évidemment pas faire l'inverse avec un domaine local)
- Le chemin absolu vers un fichier. Csv, contenant les informations des utilisateurs, a synchronisé. C'est ce fichier qui permet de m'indiquer quels utilisateurs je dois synchroniser et ainsi éviter de synchroniser les administrateurs/les prestataires ou autres...
- L'identifiant et le mot de passe d'un administrateur Office 365 ayant les droits d'accès à Entra ID (anciennement Azure) et les droits de modifications sur les utilisateurs.

Une fois tous les prérequis validés, alors il est possible de lancer mon projet, celui-ci va installer/importer et se connecter à tous les modules nécessaires à son exécution. Vous trouverez dans l'Annexe de ce document « **CAPTURE DE CODE : FONCTION TYPE D'IMPORTATION ET D'INITIALISATION DES MODULES** » le premier extrait de mon code que je souhaite présenter.

Il s'agit d'une fonction d'importation type qui demande en variable d'entrer les identifiants et mots de passe à Office 365. Tout d'abord, il détectera si le module en l'occurrence « AzureAD » est présent sur le serveur, si ce n'est pas le cas, il l'installera avant de l'importer. Puis il essaiera si nécessaire à son fonctionnement de se connecter avec les identifiants fournis et d'ouvrir une session vers Office365.

Le code est identique à l'exception des Noms pour les modules :

- MsolService : permet de récupérer des informations complémentaires dans Office365
- ImportExcel : permet d'ouvrir et de manipuler des fichiers .csv
- Active Directory : permet de manipuler les users dans l'ActiveDirectory

Une fois, l'initialisation fait mon programme va réaliser une deuxième tâche en fond avant d'afficher le menu de sélection. La récupération de la liste des utilisateurs dans le CSV à l'aide du code dont la capture est en annexe « **CAPTURE DE CODE : FONCTION RECUPERATION DES UTILISATEURS A SYNCHRONISER.** »



Cette fonction prend en paramètre le chemin absolu du fichier csv et le nom de domaine tous deux présent dans le fichier de configuration. A partir de là, il va essayer d'ouvrir le .csv en considérant que le délimiteur est un « ; ».

Puis dans un deuxième temps il va parcourir le fichier CSV et va créer un Objet pour chaque utilisateur dans lequel il y aura 4 champs importants :

- UserName (équivalent à samAccountName)
- UserPrincipalName
- ProxyAddresses
- Mail

Il s'agit là de la première partie de l'harmonisation des données. Le connecteur Azure AD utilise ces 4 champs dans l'attribut d'objet d'un utilisateur AD en plus de l'Object ID pour faire la liaison avec son équivalent dans Office365. Ainsi lors de la création de l'Object je vais remplacer le domaine.local par le domaine Office365 qui lui est connu en dehors de l'organisation. (connu dans les DNS public) .

Une fois toutes ces étapes passées, la phase d'initialisation est terminée et le menu s'affiche. Voir dans l'annexe du document « **CAPTURE DE FONCTIONNEMENT : MENU DU PROJET** ». La liste des utilisateurs s'affichera une première fois avant le menu. Puis ce dernier nous proposera plusieurs options : synchroniser un, plusieurs ou tous les utilisateurs de la liste/Modifier un utilisateur dans la liste/Réafficher les utilisateurs/Quitter (permet de fermer proprement le programme et de terminer les sessions Office365) et surtout vérifier la concordance des utilisateurs. Voir dans l'annexe du document : « **CAPTURE DE CODE : FONCTION DE VERIFICATION DE LA CONCORDANCE** »

Cette fonction pourtant simple va me permettre grâce à un Get-User et Get-AzureADUser de vérifier si l'utilisateur que j'ai récupéré à partir de mon .CSV est bien présent dans l'Active Directory. Il s'agit là d'une étape essentielle et d'un garde-fou qui permet de vérifier les erreurs dans le .CSV fournit par le responsable IT du client et d'éviter tous crash ou toute corruptions des informations d'un utilisateur.

Finalement une fois, les utilisateurs récupérés et leurs concordances prouvées, je peux lancer la synchronisation de mes utilisateurs, l'élément clé de mon programme. Voir dans l'annexe du document : « **CAPTURE DE CODE : FONCTION DE PRINCIPALE DE REECRITURE DANS OFFICE365 ET ACTIVE DIRECTORY** ».

Dans la première partie de ce code, je vais donc parcourir et remplacer les informations des utilisateurs de L'AD par ceux que j'ai modifié à partir de mon fichier CSV avec un Set-UserAD -Identity -Replace.

Puis dans un deuxième temps, je vais récupérer l'ObjectID de l'utilisateur en cours pour le convertir en Base64 avant de l'envoyer vers Office365 avec un Set-AzureADUser. L'ObjectID étant le dernier paramètre restant après les 4 premiers cité précédemment pour synchroniser un utilisateur entre les 2 environnements. Comme il m'est impossible de créer un ObjectID de rien, il me faut obligatoirement récupérer celui de l'AD pour l'envoyer vers Office365. Une fois l'ObjectID copier l'utilisateur est ajouté au Groupe de sécurité GrpSync\_AzureAD qui permet à l'utilisateur d'être synchronisé.

Ceci était la dernière étape du programme une fois fait, il suffit d'attendre le prochain cycle de synchronisation automatique toutes les 30 minutes par défaut ou de lancer une synchronisation avec une ligne de commande ou en passant par la console du connecteur AzureAD, vous pourrez également confirmer avec cette dernière le nombre d'User synchroniser.

## Test /Validation/Vérification

L'utilisation de ce projet étant cadrée dans un environnement complexe, un tenant Office365 non synchronisé et la présence d'un Active Directory installé sur un serveur DC avec un connecteur AzureAD connecté déployer sur ce dernier. J'ai donc réalisé en amont une démonstration (disponible en annexe du projet) pour démontrer la validation du projet.















La vidéo de présentation se déroule comme suit :

1. Dans un premier temps, on me voit créer 3 Utilisateurs UserAzureTest 10 à 12 dans l'Active Directory puis dans le tenant Office365 (l'utilisateur n°10 a été créé de la même manière en amont, mais n'est pas montré dans le montage final).
2. Dans un deuxième temps, je réalise la synchronisation d'UserAzureTest10 sans l'application de mon projet en amont. On visualise alors que l'utilisateur est dupliqué avec un domaine en AAAFRANCECARS.onmicrosoft.com. C'est précisément l'erreur que l'on cherche à éviter, actuellement le connecteur AzureAD Connect n'arrive pas à faire la liaison entre les utilisateurs du domaine AD et celui du tenant Office365, car les informations telles que l'adresse SMTP, l'UserPrincipaleName, le nom de domaine ou l'objectID ne corresponde pas entre eux. Résultat plutôt qu'essayer de réécrire les données, le connecteur va simplement créer un nouvel utilisateur avec le domaine par défaut de Microsoft pour résoudre le conflit. Résultat, nous avons un utilisateur dupliqué, et bien que nous ayons la possibilité de changer l'adresse email principal de l'utilisateur synchronisé pour corriger le problème, nous ne pourrions pas garder l'historique des emails. C'est donc un échec.
3. Dans un troisième temps, je réalise l'application de mon projet sur mes 3 utilisateurs de test, mon programme va récupérer les informations des utilisateurs ciblés (ObjectID, UserPrincipaleName) et réécrire les informations dans le tenant Office365 et AD pour que les informations correspondent de chaque côté, il va aussi ajouter mes utilisateurs au groupe de sécurité GroupSync\_AzureAD qui filtre les utilisateurs synchronisés pour qu'il soit synchronisé au prochain cycle du Connecteur.

Finalement, vous pourrez constater que les 3 utilisateurs tests créés à l'occasion, sont bien passés en synchronisation démontrant la réussite de ce projet et prouvant que cette procédure pourra alors être réitérée pour des utilisateurs réels afin de synchroniser l'ensemble des utilisateurs du domaine.

### Capture Finale après actualisation :

7 utilisateurs trouvés

<input type="checkbox"/>	Nom d'affichage ↑↓	Nom d'utilisateur principal ↑↓	Type d'utilisateur	Synchronisatio...	Identités	Nom de l'entrep
<input type="checkbox"/>	 <b>TEST</b>	TEST1@francecars.fr 	Membre	Non	AAAFRANCECARS.onmicrosoft.c...	
<input type="checkbox"/>	 <b>Test a supprimer- Prenez RDV avec</b>	TestRDVCCile@francecars.fr 	Membre	Non	AAAFRANCECARS.onmicrosoft.c...	
<input type="checkbox"/>	 <b>Test11 UserAzure</b>	UserAzureTest11@francec... 	Membre	Oui	AAAFRANCECARS.onmicrosoft.c...	
<input type="checkbox"/>	 <b>Test5 UserAzure</b>	UserAzureTest5@franceca... 	Membre	Oui	AAAFRANCECARS.onmicrosoft.c...	
<input type="checkbox"/>	 <b>Test10 UserAzure</b>	UserAzureTest10@francec... 	Membre	Oui	AAAFRANCECARS.onmicrosoft.c...	
<input type="checkbox"/>	 <b>Test12 UserAzure</b>	UserAzureTest12@francec... 	Membre	Oui	AAAFRANCECARS.onmicrosoft.c...	
<input type="checkbox"/>	 <b>A supprimer Test - Prenez RDV avec</b>	CcileLANEZPrenezRDVave... 	Membre	Non	AAAFRANCECARS.onmicrosoft.c...	



## Bug / Difficulté rencontré

Dans ma vidéo de démonstration, on peut apercevoir ceci à la suite de l'application de mon programme, les erreurs/bugs que je vais expliquer ci-dessous.

```
Entrez votre choix : 1
3c3bf4e9-f2ac-4109-84eb-beaa4e299a2e CN=Test10 UserAzure,OU=Test_Nancy,OU=Test_Agences,OU=Test_AzureAD,DC=francecar,DC=com s54rLgWu/U6N+C9W3yB5tg==
Account Environment TenantId TenantDomain AccountType
-----
admin.infitec@francecars.fr AzureCloud 5322e483-a80c-4b2e-b279-c9997329883c francecars.fr User
Set-AzureADUser : Error occurred while executing SetUser
Code: Request_BadRequest
Message: Another object with the same value for property immutableId already exists.
RequestId: 6d18b332-0bc5-4d6a-8812-f0bbc27c4695
DateTimeStamp: Wed, 09 Apr 2025 14:26:55 GMT
Details: PropertyName - ImmutableId, PropertyErrorCode - ObjectConflict, ConflictingObjects - User_239cea5b-fc3f-4c1e-b8a4-ba1f251eec06
HttpStatusCode: BadRequest
HttpStatusDescription: Bad Request
HttpResponseStatus: Completed
Au caractère C:\Infitec\scriptAzureAD\ProjetAzureAD\Sync.ps1:25 : 9
+ Set-AzureADUser -ObjectId $Azure.ObjectId -ImmutableId $Immut ...
+ ~~~~~
+ CategoryInfo          : NotSpecified: (:) [Set-AzureADUser], ApiException
+ FullyQualifiedErrorId : Microsoft.Open.AzureAD16.Client.ApiException,Microsoft.Open.AzureAD16.PowerShell.SetUser

Set-ADUser : L'attribut ou la valeur de service d'annuaire spécifié n'existe pas
Nom du paramètre : msDS-ConsistencyGuid
Au caractère C:\Infitec\scriptAzureAD\ProjetAzureAD\Sync.ps1:32 : 9
+ Set-ADUser -Identity $user.UserName -Replace @{"msDS-Consiste ...
+ ~~~~~
+ CategoryInfo          : InvalidArgument: (UserAzureTest10:ADUser) [Set-ADUser], ArgumentException
+ FullyQualifiedErrorId : ActiveDirectoryCmdlet:System.ArgumentException,Microsoft.ActiveDirectory.Management.Commands.SetADUser

04e1ef41-0d1e-4422-8824-4692232a1d6d CN=Test11 UserAzure,OU=Test_Nancy,OU=Test_Agences,OU=Test_AzureAD,DC=francecar,DC=com CrPI98+770m5P6IY7vYgwA==
Set-ADUser : L'attribut ou la valeur de service d'annuaire spécifié n'existe pas
Nom du paramètre : msDS-ConsistencyGuid
Au caractère C:\Infitec\scriptAzureAD\ProjetAzureAD\Sync.ps1:32 : 9
+ Set-ADUser -Identity $user.UserName -Replace @{"msDS-Consiste ...
+ ~~~~~
+ CategoryInfo          : InvalidArgument: (UserAzureTest11:ADUser) [Set-ADUser], ArgumentException
+ FullyQualifiedErrorId : ActiveDirectoryCmdlet:System.ArgumentException,Microsoft.ActiveDirectory.Management.Commands.SetADUser

b28292ea-7706-4d61-b4c4-ee2463b3a2e2 CN=Test12 UserAzure,OU=Test_Nancy,OU=Test_Agences,OU=Test_AzureAD,DC=francecar,DC=com dmEwnA28KEcheI3os58FLg==
Set-ADUser : L'attribut ou la valeur de service d'annuaire spécifié n'existe pas
Nom du paramètre : msDS-ConsistencyGuid
Au caractère C:\Infitec\scriptAzureAD\ProjetAzureAD\Sync.ps1:32 : 9
+ Set-ADUser -Identity $user.UserName -Replace @{"msDS-Consiste ...
+ ~~~~~
+ CategoryInfo          : InvalidArgument: (UserAzureTest12:ADUser) [Set-ADUser], ArgumentException
+ FullyQualifiedErrorId : ActiveDirectoryCmdlet:System.ArgumentException,Microsoft.ActiveDirectory.Management.Commands.SetADUser

--- Menu de selection ---
1. Lancer le programme pour tous les utilisateurs
2. Sélectionner plusieurs utilisateurs à traiter
```

La première erreur rencontrée « Messages : Another object with the same value for property immutableId already exists. » Signifie qu'un utilisateur avec le même ObjectID que j'essaie de réécrire existe déjà et qu'il est impossible d'en avoir 2 avec le même ID. Cette erreur ne vient pas du programme, mais de ma démonstration précédente. Dans la deuxième partie de démonstration de mon projet, j'ai d'abord synchronisé UserAzureTest10 sans appliquer mon programme avec de démontrer l'erreur. L'utilisateur a alors été dupliqué avant d'être supprimé par la suite.

Cependant, j'ai négligé le fait que lorsque qu'on supprime un utilisateur celui-ci n'est pas supprimé de manière définitive, car Microsoft laisse la possibilité de restaurer un utilisateur pendant 30 jours. De ce fait, l'utilisateur n'a pas été supprimé de manière définitive provoquant ce conflit. Après suppression réelle de l'utilisateur et resynchronisation, l'utilisateur a pu passer en synchroniser.

La deuxième erreur/Bug rencontré et qui apparaît pour mes 3 utilisateurs et du au fait que j'essaie de réécrire l'attribut « msDs-ConsistencyGuid » pour chaque utilisateur alors que ce champs est vide. L'origine de ce problème vient d'une utilisatrice longue durée déjà synchronisée qui est le

responsable IT de la société au niveau national et qui intervient aussi dans d'autres sociétés du même groupe. L'utilisateur devant donc naviguer entre plusieurs sociétés et utiliser du MFA son ordinateur à tendance à écrire plusieurs champs dans l'active directory, dont celui-ci. Or, il s'agit d'un champ qui provoque des erreurs dans la synchronisation et qui a nécessité que je réécrive les informations manuellement pour corriger l'erreur. J'ai par la suite essayé d'implémenter une ligne de code dans mon projet pour pousser ce champ lors de la synchronisation initiale et éviter que le problème réapparaisse. Or et c'est là une erreur de ma part, j'ai considéré l'utilisateur qui faisait exception comme étant une norme et j'ai essayé de l'appliquer à tous les utilisateurs alors que ce champ était vide. Finalement, j'ai simplement supprimé ma ligne qui modifiait cette variable, cette dernière n'étant pas pertinente.

Enfin, une autre de mes difficultés rencontrées fut pour la création d'interface graphique pour mon application. Bien qu'ayant développé et récupéré des lignes de code de précédent projet, j'ai sous-estimé la complexité de la mise en place de ce dernier et voyant la tâche chronophage que serait son implémentation, j'ai décidé de réaliser une simple console dans l'interface PowerShell pour dans un premier temps réaliser mes 1ers tests puis pour l'ensemble du projet.

## Conclusion

Je suis satisfait de ce projet qui a permis de démontrer ma capacité de répondre au besoin d'un client (ici le besoin d'uniformiser et de synchroniser les informations de l'active directory avec Office 365) au travers d'un projet. Par rapport à d'autres projets que j'ai pu suivre ou réalisé, celui-ci fut très différent car il n'appliquait pas de grande connaissance ou défis en programmation, mais une connaissance approfondie des 2 environnements et une grande capacité d'autonomie et de recherche pour trouver des solutions aux problèmes de communications et de synchronisation entre ces différents environnements. Microsoft étant de mon point de vue très bon pour retourner une erreur/la cause, mais ne donne que très rarement la solution pour résoudre l'erreur. Il est donc nécessaire de savoir chercher sur les différents forums et documentation disponible pour trouver la solution.

Ce projet bien qu'étant fonctionnel de mon point de vue reste ouvert aux améliorations, certaine partie du code comme l'ajout de l'utilisateur à des groupes de sécurité sont codés en dur et sont des spécifié développer pour le client à sa demande. Il s'agirait de trouver une solution pour permettre plus de flexibilité au code.

De plus, l'implémentation d'une réelle interface graphique pourrait être un plus, mais du fait de sa complexité sur PowerShell, la mise en place de ces différentes améliorations nécessiterait de multiplier par 2 la durée de développement du projet pour peu d'ajouts fonctionnel.

## Annexe :

- Lien GitHub du Livrable : <https://github.com/GNAlexandre/Projet-Master-Uniformisation-des-informations-Office-365-et-Active-Directory/tree/main>
- Schéma 1 et 2 : Source : <https://www.openhost-network.com/blog/tout-savoir-sur-azure-ad-connect-pour-votre-entreprise/>
- Image créer joint au projet et disponible sur le GitHub
- Vidéo de Démonstration du Projet : « [Projet Master Vidéo Démonstration.mp4](#) » joint au projet



## CAPTURE DE CODE : FONCTION TYPE D'IMPORTATION ET D'INITIALISATION DES MODULES

```
function InitialiserAzureAD {  
    param (  
        [string]$LoginOffice365,  
        [string]$PasswordOffice365  
    )  
  
    #Initialisation du Module AzureAD  
    Write-Host "### Initialisation du Module AzureAD ###"  
  
    # Détection du module AzureAD  
    if (-not (Get-Module -Name AzureAD -ListAvailable)) {  
        Write-Host "Le module AzureAD n'est pas installé"  
        Write-Host "Installation du module AzureAD"  
        Install-Module -Name AzureAD -Force -AllowClobber  
        Write-Host "Module AzureAD installé"  
    }  
    else {  
        Write-Host "Le module AzureAD est déjà installé"  
    }  
  
    # Importation du module AzureAD  
    Import-Module AzureAD  
  
    # Connect to Azure AD avec les identifiants Office365  
    $SecurePassword = ConvertTo-SecureString $PasswordOffice365 -AsPlainText -Force  
    $Credential = New-Object System.Management.Automation.PSCredential ($LoginOffice365, $SecurePassword)  
  
    try {  
        Connect-AzureAD -Credential $Credential  
        Write-Host "Connexion a AzureAD reussie"  
    }  
    catch {  
        Write-Host "Erreur lors de la connexion a AzureAD"  
        Write-Log -logFilePath $logFilePath -message "Erreur lors de la connexion a AzureAD : $_"  
    }  
  
    # Retour à la ligne  
    Write-Host "-----"
```

## CAPTURE DE CODE : FONCTION RECUPERATION DES UTILISATEURS A SYNCHRONISER.

```

> Import.ps1
11 function ImporterEtParcourirFichierCSV {
12
13     # Initialisation de la liste des utilisateurs
14     $ListeUtilisateurs = @()
15     $Compteur = 0
16
17     # Essaie d'importer le fichier CSV
18     try {
19         $csv = Import-Csv -Path $NomFichier -Delimiter ";"
20     }
21     catch {
22         # Log et affichage de l'erreur en cas d'échec
23         Write-Log -logFilePath $logFilePath -message "Erreur lors de l'importation du fichier CSV : $_"
24         Write-Host "Erreur lors de l'importation du fichier CSV : $_"
25         Write-Host "Vérifiez le chemin du fichier"
26         return $null
27     }
28
29     # Parcourir le fichier CSV et ajouter les utilisateurs à la liste
30     foreach ($user in $csv) {
31         $Utilisateur = New-Object PSObject -Property @{
32             UserNbr           = $Compteur
33             UserName          = $user.UserAD
34             UserPrincipalName = $user.UserAD + "@" + $NomDomaine
35             ProxyAddresses    = "SMTP:" + $user.UserAD + "@" + $NomDomaine
36             Mail              = $user.UserAD + "@" + $NomDomaine
37             MFA               = $user.MFA
38         }
39         $Compteur++
40         $ListeUtilisateurs += $Utilisateur
41     }
42
43     # Retourne la liste des utilisateurs
44     return $ListeUtilisateurs
45 }

```

## CAPTURE DE FONCTIONNEMENT : MENU DU PROJET.

```

### Initialisation du Module Import-Excel termine ###
### Initialisation du Module ActiveDirectory ###
Le module ActiveDirectory est déjà installé
### Initialisation du Module ActiveDirectory termine ###

--- Liste des utilisateurs ---

Numero de l'utilisateur : 0
Nom de l'utilisateur : UserAzureTest10
UserPrincipalName : UserAzureTest10@francecars.fr
ProxyAddresses : SMTP:UserAzureTest10@francecars.fr
Mail : UserAzureTest10@francecars.fr
-----
Numero de l'utilisateur : 1
Nom de l'utilisateur : UserAzureTest11
UserPrincipalName : UserAzureTest11@francecars.fr
ProxyAddresses : SMTP:UserAzureTest11@francecars.fr
Mail : UserAzureTest11@francecars.fr
-----
Numero de l'utilisateur : 2
Nom de l'utilisateur : UserAzureTest12
UserPrincipalName : UserAzureTest12@francecars.fr
ProxyAddresses : SMTP:UserAzureTest12@francecars.fr
Mail : UserAzureTest12@francecars.fr
--- Menu de selection ---
1. Lancer le programme pour tous les utilisateurs
2. Selectionner plusieurs utilisateurs a traiter
3. Selectionner un utilisateur a traiter
4. Modifier les informations d'un utilisateur
5. Supprimer un utilisateur de la liste
6. Afficher la liste des utilisateurs
7. Verifier la concordance des informations
8. Quitter
--- Fin du menu de selection ---
Entrez votre choix :

```

## CAPTURE DE CODE : FONCTION DE VERIFICATION DE LA CONCORDANCE

```
#Fonction Vérification de concordance des informations
#Cette fonction a pour but de vérifier que les informations dans l'AD et dans AzureAD sont concordantes
function VerifierConcordance {
    param (
        [Parameter(Mandatory = $true)]
        #Prends en paramètre un utilisateur
        [object]$user
    )

    #Vérifie l'existence de l'utilisateur dans l'AD
    try {
        $UserAD = Get-ADUser -Identity $user.UserName
    }
    catch {
        if ($UserAD -eq $null) {
            Write-Host "L'utilisateur $($user.UserName) n'existe pas dans l'AD"
            return $false
        }
    }

    #Vérifie l'existence de l'utilisateur dans Office365
    try {
        $UserAzureAD = Get-AzureADUser -ObjectId $user.UserPrincipalName
    }
    catch {
        if ($UserAzureAD -eq $null) {
            Write-Host "L'utilisateur $($user.UserName) n'existe pas dans Office365"
            return $false
        }
    }

    return $true
}
```

## CAPTURE DE CODE : FONCTION DE PRINCIPALE DE REECRITURE DANS OFFICE365 ET ACTIVE DIRECTORY

```
foreach ($user in $ListeUtilisateurs) {
    try {
        #Modifier information de l'utilisateur dans l'AD
        Set-ADUser -Identity $user.UserName -Replace @(UserPrincipalName = $user.UserPrincipalName; ProxyAddresses = $user.ProxyAddresses; Mail = $user.Mail )

        #Ajout de l'utilisateur à un groupe
        AjouterUtilisateurGroupe -NomGroupe "GroupesSync_AzureAD" -NomUtilisateur $user.UserName

        #Récupération de l'ObjectID dans Azure
        $Azure = Get-AzureADUser -ObjectId $user.UserPrincipalName | Select-Object ObjectId

        #Conversion de l'ObjectGUID en base64
        $UserAD = Get-ADUser -Identity $user.UserName
        $ImmutableId = [System.Convert]::ToBase64String($UserAD.ObjectGUID.tobytearray())
        Write-Host $Azure.ObjectId, $UserAD, $ImmutableId

        #Modification de l'ImmutableID dans Azure
        Set-AzureADUser -ObjectId $Azure.ObjectId -ImmutableId $ImmutableId
    }
}
```