ALAPETITE
Florent

Rapport de Stage

Certification d'un vertical adapté aux besoins métiers des secteurs agricole, de la manutention et du levage.

Maître de stage : M. DAGOIS

Professeur référent : Mme. SERRANO ALVARADO

UNIVERSITÉ DE NANTES
FACULTÉ DES
SCIENCES ET TECHNIQUES

Année 2016-2017

Table des matières

I. Remerciements

Je tiens à remercier toutes les personnes qui ont contribué à la réussite de mon stage de fin de troisième année de Licence Informatique

Je remercie tout particulièrement l'entreprise Eskape et son dirigeant M. Laurent Fontenit de m'avoir accordé leur confiance en tant que stagiaire. Je remercie également toutes les personnes que j'ai rencontrées dans les locaux de l'entreprise pour leur accueil très chaleureux qui a facilité mon intégration au sein de l'équipe.

Je tiens à remercier mon maître de stage M. Anthony Dagois et mon référent technique M. Mathieu Corvaisier pour leur présence et leurs conseils. Leur aide précieuse m'a permis d'avancer rapidement sur les différents travaux que j'ai eu à réaliser tout au long de mon stage.

II. Introduction

En tant qu'étudiant de Licence 3 Informatique de l'UFR Sciences et Techniques de Nantes, la validation de mon diplôme s'accompagne de la réalisation d'un stage d'une durée de huit semaines dans un environnement professionnel. Plus qu'une simple étape de la formation dispensée en Licence Informatique, ce stage est un élément primordial dans la réussite du diplôme. En effet, cette expérience particulière vient clore une formation riche en apprentissages réalisée à l'université de Nantes.

J'ai effectué mon stage de fin d'études au sein de l'entreprise Eskape, qui est une entreprise de services du numérique. Celle-ci réalise des projets informatiques qui viennent satisfaire des besoins exprimés par des entreprises clientes de toute la région. Ces projets sont centrés autour du progiciel de gestion intégré Microsoft Dynamics Nav. C'est dans ce contexte particulier que j'ai apporté mon aide à l'entreprise Eskape qui m'a proposé de travailler sur un projet principal : la certification d'un vertical* développé par leur équipe projet. C'est principalement ce projet qui va être détaillé tout au long de ce rapport de stage.

Dans une première partie, je présenterai l'entreprise Eskape qui m'a accueillie pendant ma période de stage puis je parlerai plus en détail du sujet de mon stage et de son contexte. Je développerai ensuite les différentes étapes de la réalisation de mon projet de stage et je terminerai par un bilan humain et technique.

III. Présentation de l'entreprise Eskape

La SARL ESKAPE est une entreprise de services du Numérique (ESN anciennement SSII) fondée en 1998 par Laurent FONTENIT et comprenant une vingtaine de salariés. Eskape est spécialisée dans le développement et la mise en place de solutions autour de progiciels de gestions intégrés* (PGI ou ERP) et notamment celui de Microsoft, Dynamics Nav. Auparavant divisée entre plusieurs services tels que les parties systèmes, réseaux et développement ERP, l'entreprise s'est recentrée il y a quelques années sur la partie ERP en fournissant à ses clients des solutions toujours plus proches de leurs besoins.

Microsoft Dynamics Nav (anciennement NAVISION) est un ERP conçu pour aider les P.M.E. et P.M.I. (entreprises de moins de 150 personnes) dans leurs opérations de gestion quotidiennes. Couplé à une base de données, il est doté de nombreuses fonctionnalités afin de faciliter des actions spécifiques telles que la gestion des achats, des clients, des fournisseurs, des prospects ou encore la logistique et la gestion financière. L'entreprise Eskape conçoit des solutions pour les entreprises utilisatrices de l'ERP de Microsoft en s'appuyant sur les outils de développement propres à Dynamics Nav. Ainsi, la société Eskape a réalisé de nombreux modules pour enrichir l'ERP de nouvelles fonctionnalités, parfois explicitement formulées par les entreprises clientes de l'ESN. Parmi ces modules, on peut notamment citer NAViPaye, un module très paramétrable qui permet d'intégrer les mécaniques de paye et de gestion d'absences directement dans Dynamics Nav.

En tant que partenaire Microsoft, l'entreprise Eskape propose également un suivi personnalisé aux entreprises clientes, ce qui leur permet de compter sur des techniciens réactifs en cas de panne sur une solution développée et mis en place par Eskape. En effet, l'entreprise dispose d'un service d'assistance performant permettant la résolution rapide de problèmes sur l'ERP Dynamics Nav ou les modules intégrés dans celui-ci.

IV. Présentation du sujet du stage

Le sujet principal de mon stage de fin Licence 3 Informatique a été la certification d'un vertical développé par Eskape pour les besoins de leurs clients. En effet, l'entreprise Eskape développe des solutions permettant de répondre à des besoins métiers adressés par des clients de toute la région Centre. Ces dernières années, les équipes d'Eskape ont réalisés, suite à la demande d'un de leurs clients, une solution de gestion intégrée pour les secteurs agricole, de la manutention et du levage : NAViDMS. Ce module (aussi appelé vertical) a été développé en tant que solution spécifique venant répondre à une série de besoins métier particuliers.

Après avoir installé et éprouvé le vertical dans l'entreprise cliente à l'origine du besoin métier, les équipes d'Eskape ont eu une réflexion quant à la poursuite du projet. En effet, le module développé est un module très spécifique mais qui est adapté à toutes les entreprises du secteur des concessions agricoles. En tant que vertical spécifique, Eskape ne peut pas fournir dans les meilleures conditions possibles le vertical à des entreprises du même secteur qui pourraient en faire la demande. C'est pourquoi Eskape souhaitait faire certifier le vertical NAViDMS dans le but pouvoir le proposer en tant que module officiel Dynamics Nav dans toute la France.

Ce programme de certification, appelé **CFMD** (*Certified for Microsoft Dynamics*) permet d'officialiser auprès de Microsoft des modules développés par des entreprises partenaires de l'éditeur. Pour cela, le vertical doit répondre à un certain nombre de critères définis dans un documentation très précise.

V. Réalisation du projet

V.1. Présentation du projet et formation :

Les premiers jours de mon stage au sein de l'entreprise Eskape ont été consacrés à la présentation du projet qui m'a été confié. Les équipes d'Eskape m'ont ensuite formé plus en détail à l'utilisation et au développement sur la nouvelle version de Dynamics Nav (Microsoft Dynamics Nav 2016). J'ai ensuite pu bénéficier d'une présentation détaillée du vertical NAViDMS ce qui m'a permis de mieux comprendre les différents besoins métiers traités dans le module.

Pour le bien du projet, Eskape a mis à ma disposition un ordinateur sous Windows 10 et équipé des outils nécessaires au développement sous Dynamics Nav. J'ai par la suite mis en place les d'autres outils que j'ai été amené à utiliser en installant notamment l'IDE Microsoft Visual Studio 2017. Les équipes d'Eskape m'ont également mis en place un environnement de test au sein de Dynamics Nav en me fournissant une base de données comprenant un jeu de test complet.

V.2. Documentation autour de la certification du vertical :

Dans un premier temps, j'ai consacré mon temps de travail à la recherche d'informations et de documentation autour de la certification CfMD d'un vertical Dynamics Nav. Pour cela, j'ai pu compter sur l'expérience des équipes d'Eskape qui avait déjà fait précédemment certifié un module. Je me suis également documenté avec le guide de certification CfMD officiel de Microsoft, un document très complet et entièrement rédigé en anglais technique qui explique pas-à-pas les étapes du test de certification du vertical.

J'ai ainsi pu apprendre que le test de certification est réalisé par un entreprise externe à Microsoft. De plus, le test est très normalisé et se déroule point par point. Si le vertical ne valide par l'un des points du test, ce dernier est considéré comme échoué et l'entreprise ayant soumis son module a deux mois pour réaliser les modifications et se mettre ainsi en conformité. Grâce à toutes ces informations, j'ai pu débuter rapidement mon travail sur le projet de certification.

V.3. Mise en conformité de la numérotation des objets du vertical :

Mon maître de stage Anthony m'a rapidement indiqué le premier point de la certification sur lequel je devais travailler : la mise en conformité des objets standards et spécifiques du vertical NAViDMS, l'un des points plus importants de la demande de certification. Afin de mieux comprendre ce besoin de mise en conformité des objets, il est nécessaire de comprendre le fonctionnement des objets de l'ERP Microsoft Dynamics Nav.

V.3.1. Fonctionnement des objets de Dynamics Nav :

Microsoft Dynamics Nav est composé d'une multitude d'objets de types différents et répondant à des fonctions particulières. Parmi les types d'objets Dynamics Nav, on retrouve par exemple les objets suivants :

- Table : table de données (stockées physiquement dans une base de données) comportant des champs de différents types (nombre, texte, …). Les champs d'une table possèdent un identifiant unique (entier).
- Page : mise en forme des informations contenues dans les tables avec divers outils comme des contrôleurs (boutons, champs de texte, …).
- Codeunit : recueil de méthodes utilisables dans Dynamics Nav et pouvant interagir avec les autres objets de l'ERP. Les codeunits ont la particularité de pouvoir être distribués en

tant que Web Services et sont développées dans un langage propre à Dynamics Nav : le C/AL (Client/Server Application Language).

Tous les objets Dynamics Nav ont un identifiant unique (entier) dont la plage correspond à leur type :

- Identifiant entre 1 et 49 999 : **objets standards**.
- Identifiant entre 50 000 et 99 999 : **objets spécifiques**.
- Identifiant entre 100 000 et 999 999 999 : **objets certifiés**.

Microsoft Dynamics Nav est fourni nativement avec une série d'**objets standards**. Ces objets standards sont des objets liés à des besoins génériques d'entreprises. Par exemple, les tables Articles, Clients, Employés et Fournisseurs sont des tables standards de Dynamics Nav. Ces objets ne peuvent pas être supprimés. Seuls des ajouts internes sont permis sur ces objets standards (notamment des ajouts de champs dans les tables standards pour les besoins de nouveaux modules). La plage de numérotation des objets standards est bloquée pour tous les partenaires Microsoft (aucun nouvel objet ne peut être ajouté dans cette plage).

✔	16	Payment Disc. Credit Acc.	Code	20
✔	17	Payment Tolerance Debit Acc.	Code	20
✔	18	Payment Tolerance Credit Acc.	Code	20
✔	19	Add. Fee per Line Account	Code	20
✔	70000	Frais de port Location	Code	20
✔	70001	Additionnal consumption contra	Code	20

Exemple de champs d'une table standard (>49 999 : champs non-standards)

La plage des **objets spécifiques** est une plage libre : les partenaires Microsoft peuvent s'en servir pour développer des objets spécifiques pour leurs clients. Ces objets sont bien souvent très spécialisés.

La plage des **objets certifiés** est également une plage bloquée pour les partenaires Microsoft. En revanche, lorsqu'une entreprise partenaire démarre un processus de certification d'un module, Microsoft modifie la licence développeur de l'entreprise en lui

fournissant une plage de numérotation dans la plage des objets certifiés, lui permettant de migrer ses objets à faire valider dans la plage attribuée.

V.3.2. Analyse du processus de renumérotation des objets :

Le vertical NAViDMS développé par les équipes d'Eskape comportait à la fois des modifications sur les objets standards de Dynamics Nav et une multitude de nouveaux objets jusqu'alors numérotés dans la plage des objets spécifiques (entre 50 000 et 99 999).

Le guide de certification CFMD de Microsoft est très clair sur la numérotation des objets du module à faire certifier :

- Les identifiants des champs ajoutés dans les **tables standards** doivent être compris dans la plage de renumérotation fournie par Microsoft.
- Les nouveaux objets développés pour le module doivent être renumérotés dans la plage fournie par Microsoft.

Le module NAViDMS développé par Eskape étant un module très complet, il était absolument impossible d'imaginer réaliser ces renumérotations à la main du fait de la multitude d'objets ajoutés et modifiés. C'est pourquoi j'ai fait des recherches sur des processus de renumérotation automatique des objets Dynamics Nav.

Différents sites spécialisés proposent des outils à intégrer dans Dynamics Nav (souvent des Codeunits à importer) permettant de réaliser des modifications sur les objets de l'ERP en évitant au maximum la perte de données. Cependant, ces outils sont bien souvent beaucoup plus complets que nécessaires et surtout très couteux. C'est pourquoi j'ai décidé, en accord avec mon maître de stage, de développer moi-même un outil de renumérotation des objets Dynamics Nav que pourraient utiliser facilement les équipes d'Eskape.

V.3.3. Développement de l'outil de migration des objets :

Les objets Dynamics Nav ne peuvent nativement pas être modifiés sans subir un phénomène de perte d'information. Cela est principalement dû au fait que les objets de l'ERP sont très liés entre eux. Par exemple, les objets de type Page sont liés à des objets de type Table (car les pages mettent en forme les données des tables). Ce lien est conservé dans l'ERP grâce à l'identifiant de l'objet lié. Réaliser une modification d'identifiant sur un objet ne garantit pas que les liens de cet objet suivent correctement la renumérotation.

La manière la plus efficace de réaliser de manière automatique cette renumérotation est de réaliser une migration complète des objets à renuméroter. En effet, Dynamics Nav propose un outil d'import et d'export de structures d'objets dans des fichiers. J'ai donc imaginé un moyen de modifier directement la numérotation des objets dans les structures exportées tout en garantissant la conservation des éventuelles données contenues dans les tables de l'ERP avant la migration. Cela se traduit par le scénario suivant :

- Export des différents objets du vertical.
- Modification des fichiers exportés par un programme externe.
- Réimportation des fichiers modifiés dans Dynamics Nav.
- Migration des données vers les objets modifiés.

J'ai décidé de développer l'outil de migration en C#, langage de prédilection des applications ayant pour but d'être exécuté sous le système d'exploitation Windows. J'ai ensuite méthodiquement analysé les fichiers exportés par Dynamics Nav. En effet, Dynamics Nav propose d'exporter les structures d'objets de deux manières : en fichiers FOB (fichiers chiffrés non modifiables) et en fichiers textes classiques (.txt). Les objets exportés en .txt ont une structure très organisée et des délimitations claires entre les différentes informations qui

composent les objets (propriétés, champs, code C/AL, …). Les fichiers .txt générés par Dynamics Nav sont encodés en MS-DOS, un encodage très particulier que j'ai également dû gérer.

```
OBJECT Table 32 Item Ledger Entry
{
  OBJECT-PROPERTIES
  {
    Date=06/01/17;
    Time=15:37:35;
    Modified=Yes;
    Version List=NAVW19.00,NAVFR9.00;
  }
  PROPERTIES
  {
    OnInsert=VAR
               GenJnlPostPreview@1000 : Codeunit 19;
             BEGIN
               GenJnlPostPreview.SaveItemLedgEntry(Rec);
             END;

    CaptionML=[ENU=Item Ledger Entry;
               FRA=criture comptable article];
    LookupPageID=Page38;
    DrillDownPageID=Page38;
  }
  FIELDS
  {
    { 1   ;   ;Entry No.           ;Integer       ;CaptionML=[ENU=Entry No.;
                                                            FRA=Nø s,quence] }
    { 2   ;   ;Item No.            ;Code20        ;TableRelation=Item;
                                                   CaptionML=[ENU=Item No.;
                                                            FRA=Nø article] }
    { 3   ;   ;Posting Date        ;Date          ;CaptionML=[ENU=Posting Date;
                                                            FRA=Date comptabilisation] }
    { 4   ;   ;Entry Type          ;Option        ;CaptionML=[ENU=Entry Type;
                                                            FRA=Type ,criture];
```

Partie de fichier exporté par Dynamics Nav

Pour réaliser la renumérotation des objets, j'ai développé un analyseur syntaxique qui lit le fichier des objets exporté et fait différentes modifications sur ce dernier en fonction de « patterns ». En effet, l'étude des fichiers exportés m'a permis d'en apprendre un peu plus sur la structuration de la création des objets et des liens entre objets. Voici deux exemples de patterns principaux :

- **OBJECT type identifiant nom** : représente la création d'un objet Dynamics Nav avec :
  - **type** : le type de l'objet qui sera créé.
  - **identifiant** : le numéro unique de l'objet.
  - **nom** : le nom de l'objet.

- **Type identifiant;** : représente un lien entre objets avec :
    - **Type**: le type de l'objet avec lequel le lien doit être fait.
    - **identifiant** : le numéro unique de l'objet avec lequel le lien doit être fait.

Ces deux patterns principaux (en plus de quelques autres patterns plus spécifiques) m'ont permis d'automatiser la modification des fichiers exportés ainsi que d'assurer la bonne conservation des différents liens existants entre les objets (voir de migrer les numéros de ces liens si besoin).

Les fichiers exportés des structures d'objets Dynamics Nav sont de très gros fichiers (plusieurs millions de lignes), j'ai donc dû réfléchir à un scénario de modification de ces fichiers le plus optimal possible. En effet, il est parfois nécessaire de passer plusieurs fois sur certains fichiers dans le but de mettre à jour les liens des objets renumérotés. Le scénario de modification que j'ai implémenté est le suivant :

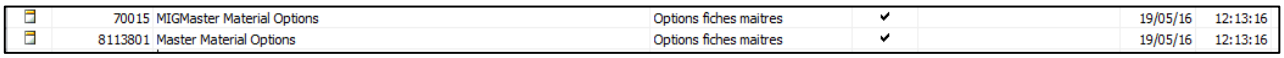
- Renommage des lignes de création d'objets spécifiques (les objets de la plage 50 000 – 99 999).
- Mise à jour des liens des objets spécifiques avec les nouveaux numéros des objets spécifiques.
- Renommage des champs spécifiques ajoutés dans les tables standard et mise à jour des numéros de liens des objets spécifiques renumérotés (opérations conjointes).
- Mise à jour des liens des autres objets standards avec les nouveaux numéros des objets spécifiques.

Afin de mener à bien ce scénario, j'ai choisi d'imposer un découpage particulier des différents objets à modifier. Ainsi, pour réaliser une migration complète de vertical, il faut fournir 3 fichiers différents à l'outil de migration :

- Un fichier .txt contenant tous les **objets spécifiques** exportés (dont les numéros sont dans la plage 50 000 – 99 999).
- Un fichier .txt contenant toutes les **tables standards** exportés (dont les numéros sont dans la plage 1 – 49 999).
- Un fichier .txt contenant tous les **objets standards** exportés (sauf les tables et dont les numéros sont dans la table 1 – 49 999).

Pour conserver les informations de renumérotation des objets et champs afin de réaliser par la suite la mise à jour des différents liens, je me suis servis d'une structure de donnée particulière : les dictionnaires. Cette structure de donnée associe une clé unique avec une valeur. Ainsi, au fil des renumérotation d'objet, j'ai remplis différents dictionnaires (un par type d'objet renuméroté) avec pour clé le numéro avant renumérotation et pour valeur associée le nouveau numéro de l'objet. De cette manière, le programme récupère facilement le nouveau numéro associé à un objet précédemment renuméroté lorsqu'il en rencontre un.

Un des points les plus important de cette renumérotation est la conservation des informations stockées dans le cas d'une migration de tables contenant précédemment des données (migration de clients déjà installés par exemple). Microsoft Dynamics Nav, lors de l'importation d'une structure de tables, n'accepte pas la renumérotation de champs et de tables qui contiennent des données (car cela détruirait les données concernées et qui n'est pas souhaitable). Ainsi, la seule manière de procéder est de faire une migration des données de la table précédente vers une nouvelle table située cette fois dans la bonne plage d'identifiant (l'ancienne table devient alors une table temporaire). Cependant, Dynamics Nav n'accepte pas la création d'objets ayant le même nom, c'est pourquoi le programme que j'ai développé se charge de fournir un fichier importable qui réalise la modification des noms des tables à migrer en ajoutant la chaîne « MIG » devant les noms de tables.



Exemple de table à migrer

Il en est de même pour les champs spécifiques des tables standards. En effet, la numérotation des tables standards n'est pas modifiée mais les champs spécifiques ajoutés pour le bien du vertical dans celles-ci doivent nécessairement être renumérotés. C'est pourquoi j'ai appliqué la même méthode qu'avec les tables spécifiques : renommage de l'ancien champ contenant les données et création d'un nouveau champ dans la bonne plage d'identifiant.

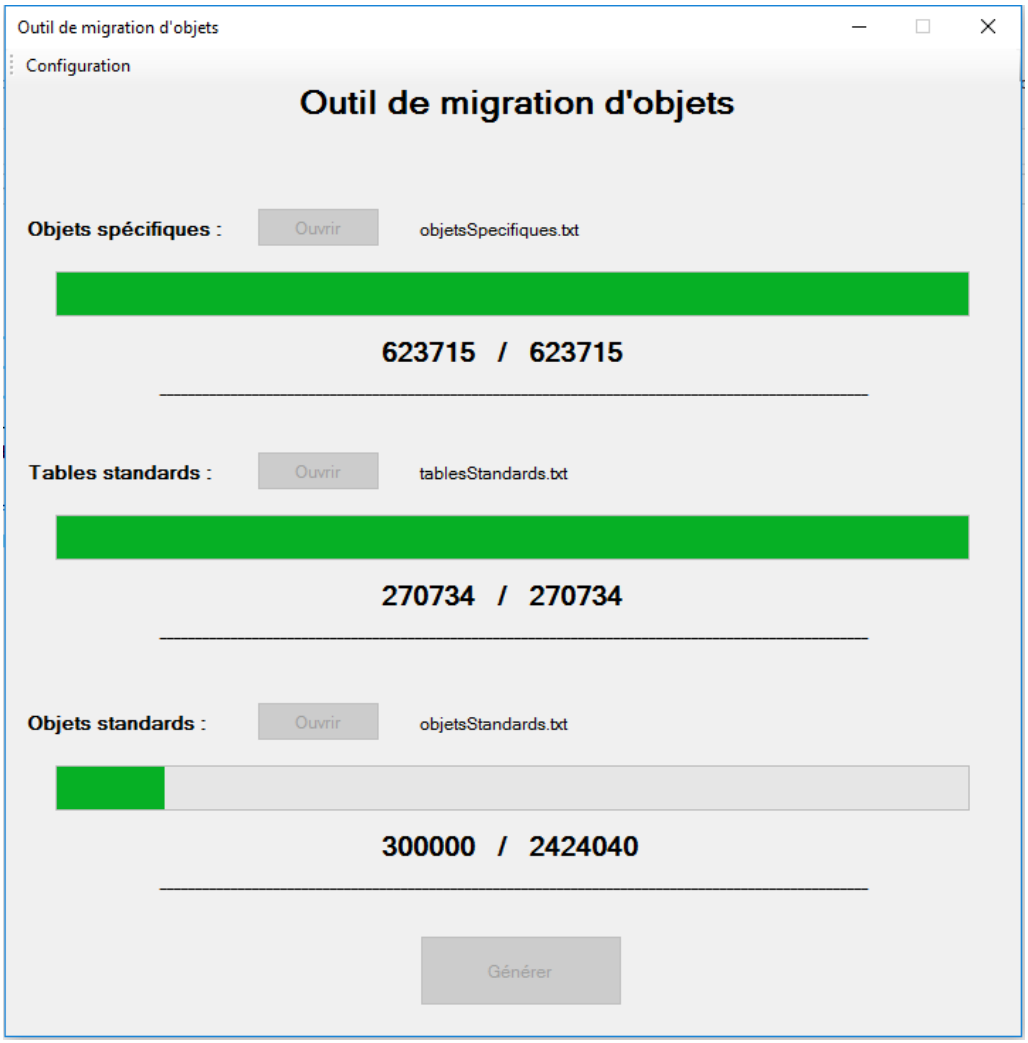
✔	60099	MIGcompany code	Code	20	
✔	70000	MIGInit Negative Qty to ship	Boolean		MCO Le 12-07-2016 => ESKCA1.0
✔	70001	MIGEDI CNH Code	Code	2	GR le 06/12/16 => Export EDI pour CNH
✔	8113767	company code	Code	20	
✔	8113778	Init Negative Qty to ship	Boolean		MCO Le 12-07-2016 => ESKCA1.0
✔	8113779	EDI CNH Code	Code	2	GR le 06/12/16 => Export EDI pour CNH

Exemple de champs de table standard à migrer

V.3.4. Création d'une interface pour l'outil :

L'outil de migration que j'ai développé a vocation à traiter de très gros fichiers (plusieurs millions de lignes et de caractères) ce qui peut potentiellement prendre beaucoup de temps (de l'ordre de 5 à 10 minutes de traitement). C'est pourquoi j'ai souhaité développer une interface pour l'outil permettant d'afficher en temps réel la progression du traitement des fichiers. Les interfaces d'applications C# (aussi appelées Windows Forms) fournissent une série de contrôleurs et d'éléments visuels dynamiques natifs permettant de créer des interfaces simples et lisibles.

Comme énoncé précédemment, l'outil a besoin de 3 fichiers très différents qui seront traités dans un ordre particulier. J'ai donc divisé l'interface de l'outil en 3 parties correspondant aux traitements des 3 fichiers. Chaque partie propose dans un premier temps de donner à l'outil le fichier demandé avec les boutons « Ouvrir » permettant, par le biais de méthodes de gestion d'évènements, d'afficher à l'utilisateur un **FolderBrowserDialog** (recherche de fichier via l'explorateur Windows classique) et ainsi de récupérer le chemin du fichier à traiter. Chaque partie est également composée d'une barre de progression qui se met à jour en temps réel.

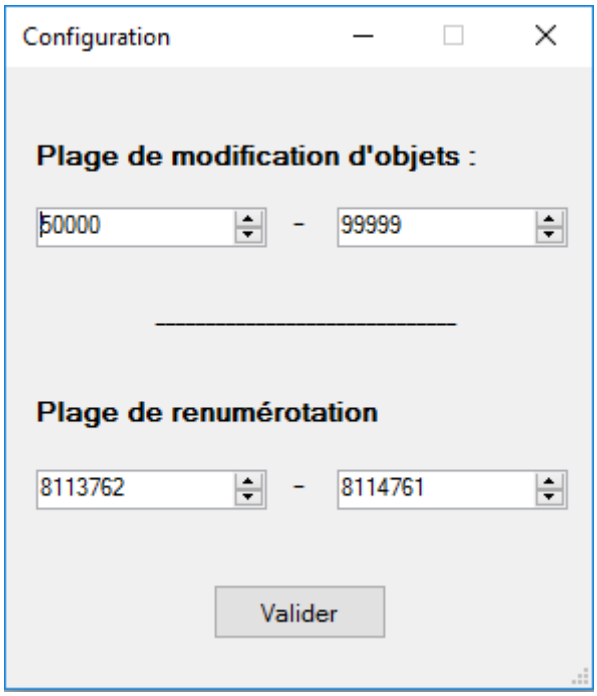


Interface de l'outil de migration

Les interfaces des Windows Forms sont générées sur des **UI Thread** (User-Interface Thread*). Exécuter des opérations coûteuses sur des UI Thread est rarement une bonne idée. En effet, lorsqu'un traitement est lancé sur un UI Thread, tout le thread est bloqué en attendant la fin du traitement. Ainsi, si des mises à jour de l'interface sont réalisées pendant l'exécution du traitement coûteux, l'UI Thread ne réalise pas la modification de l'interface puisqu'il est bloqué en attendant la fin du traitement. Pour pallier à ce problème, les Windows Forms disposent d'un objet particulier qui permet de lancer des opérations coûteuses sur un thread d'arrière-plan afin de ne pas bloquer l'exécution de l'UI Thread : les **BackgroundWorkers**. Un BackgroundWorker déclenche un traitement qui lui est fourni sur un nouveau thread. De cette manière, l'opération coûteuse se déroule sur le thread d'arrière-plan et ce dernier peut

communiquer des modifications à réaliser sur l'interface à l'UI Thread (mise à jour de la barre de progression ou d'un label de l'interface par exemple).

Dans un dernier temps, j'ai travaillé autour du paramétrage de l'outil de migration. En effet, j'ai choisi de faire en sorte que l'outil soit le plus général possible afin que les équipes d'Eskape puissent s'en servir pour renuméroter tous les objets qu'ils souhaitent vers les plages de numéro dont ils ont besoin. J'ai donc ajouté à l'outil un fichier de configuration XML* qui est lu avant de lancer le traitement. Le fichier de configuration conserve les numéros qui doivent être renommés (plage de modification d'objet) et la plage de renumérotation qui doit être utilisée. Le fichier de configuration peut être modifié très simplement via le bouton « Configuration » qui fournit à l'utilisateur un nouveau Windows Form lui permettant d'enregistrer la configuration qu'il souhaite appliquer au traitement des fichiers de migration.



Interface de configuration de l'outil de migration

V.3.5. Développement des méthodes de migration des données :

Les fichiers qui ressortent de l'outil de migration que j'ai développé permettent de répondre aux deux scénarios possibles de la migration des objets d'un vertical vers la plage de numérotation fournie par Microsoft. Ces deux scénarios sont les suivants :

- Importation simple des objets dans le but de réaliser une nouvelle installation sur une base vierge de toute donnée liée au vertical à installer.
- Importation avec migration des données présentes dans les tables d'un vertical déjà installé chez un client. Ce scénario doit permettre de migrer l'intégralité des données de la base sans aucune perte tout en mettant les objets à jour (avec les bons identifiants).

La partie importation est réalisée par le biais de la modification des structures d'objets exportés depuis le vertical initial par l'outil de migration C#. En ce qui concerne la partie de migration des données, j'ai envisagé plusieurs solutions. En effet, Dynamics Nav peut exporter les données d'une table dans des fichiers de types différents (XML, CSV, etc …). Cette solution pouvait être envisageable mais pouvait également poser des problèmes lors de la réimportation des données. En accord avec les équipes d'Eskape, je me suis donc orienté vers une autre solution, plus proche cette fois-ci de l'ERP Dynamics Nav et de ses possibilités.

Microsoft Dynamics Nav a la particularité d'être un ERP modifiable à l'extrême. Un grand nombre de traitements spécifiques peuvent être réalisées directement dans l'ERP grâce notamment à son langage de développement interne particulier : le C/AL. Il s'agit d'un langage très axé sur les données. Le C/AL comporte nativement des méthodes qui permettent de réaliser facilement et en respectant le cycle de vie d'une donnée des modifications sur les différentes informations stockées dans les tables de l'ERP (qui sont physiquement stockées sur une base de données couplée à l'ERP). On peut notamment citer la présence de nombreux triggers (déclencheurs) entièrement modifiables s'exécutant lors de la réalisation de différents évènements classiques du cycle de vie d'une donnée (trigger OnInsert, OnDelete par exemple).

Nous avons précédemment parlé des différents types d'objets de Dynamics Nav et notamment des Codeunit, qui sont des « boîtes à outils » de procédures appelables à partir de n'importe quel objet de Nav. Pour réaliser la migration des données d'une base sur laquelle le vertical était précédemment installé et utilisé, j'ai créé un Codeunit. Ce Codeunit réalise les deux migrations de données nécessaires :

- La migration des données des champs spécifiques précédents des tables standards vers les nouveaux champs créés pour l'occasion dans les bonnes plages de numérotation.
- La migration complète des tables spécifiques précédentes vers les tables spécifiques numérotés dans la bonne plage fournie par Microsoft.

Pour cela, j'ai développé des algorithmes bouclant sur toutes les tables ayant besoin d'une migration. Ces tables sont repérables par la sous-chaîne « MIG » présente dans le nom de la table ou le nom du champ à migrer. Lorsqu'une table doit être migrer, les différentes procédures que j'ai écrites réalisent une boucle sur tous les **Record** d'une table à migrer. Un **Record** est une variable particulière qui stocke un tuple présent dans la table. Des traitements sont ensuite réalisés sur ces Records afin de migrer les données d'un champ à l'autre ou d'une table à l'autre ce qui assure la bonne continuité des données lors de la migration du vertical. Il s'agit également d'un traitement très coûteux en termes de temps puisque les tables qui doivent être migrés peuvent parfois comporter plusieurs millions de tuples.

J'ai également développé une série de méthode plus spécifiques à la migration du vertical NAViDMS permettant de faire suivre les identifiants d'objets stockés dans les tables de Dynamics Nav. En effet, certaines tables de l'ERP sont des tables de paramétrages qui stockent parfois des identifiants d'objets. Afin d'assurer la bonne continuité des différents paramétrages mis en place, j'ai donc réalisé des méthodes qui migrent automatiquement ces identifiants. Pour cela, l'utilisateur doit fournir au Codeunit des fichiers (.txt) conservant les liens entre anciens et nouveaux identifiants d'objets. Ces fichiers sont directement générés lors d'une

renumérotation par le programme C# que j'ai développé (un fichier de sortie par type d'objet renuméroté).

V.4. Outil d'analyse de code :

Après avoir mis en place la renumérotation automatique des objets de Dynamics Nav, les équipes d'Eskape ont souhaité pouvoir compter sur un outil d'analyse des codes C/AL des objets Dynamics Nav afin de traquer les éventuels identifiants d'objets stockés en dur dans les codes. En effet, le vertical étant d'abord prévu pour répondre à un besoin spécifique, les équipes d'Eskape ont parfois directement écrits dans le code les identifiants d'objets lors notamment d'appels particuliers sur ces objets (la méthode C/AL **RUN** permet par exemple de lancer un objet Nav à partir de son identifiant).

Il est parfois impossible de définir quel type d'objet est lié à un numéro stocké en dur dans les codes C/AL, c'est pourquoi la renumérotation automatique n'est pas fiable à 100%. L'outil d'analyse que j'ai développé se charge donc de faire remonter les lignes contenant des numéros qui ont été précédemment renumérotés qui pourraient potentiellement poser problème par leur présence dans le code et que l'outil de migration n'a pas pu modifier. Pour cela, l'analyseur prend en entrée un fichier exporté Dynamics Nav et un fichier contenant l'ensemble des numéros précédemment modifiés (généré en sortie par l'outil de renumérotation).

J'ai profité de l'ajout de cette nouvelle fonctionnalité pour mettre à jour l'interface de l'outil de migration et le préparer à l'ajout de futures fonctionnalités en créant notamment un menu principal à partir duquel l'utilisateur peut naviguer vers les interfaces des différentes fonctionnalités qu'il souhaite utiliser.

V.5. Outil d'ajout de Version List :

Un autre point important de la certification est l'ajout d'une **Version List** aux objets du vertical qui doit être certifié. Cette Version List atteste de l'appartenance d'un objet Dynamics Nav à un vertical particulier et contient également la version du vertical. Un même objet peut avoir plusieurs Version List. C'est notamment le cas des objets standards qui peuvent être enrichis selon les critères évoqués précédemment. Afin de préciser que l'objet standard a été modifié pour le vertical, il est nécessaire d'ajouter la Version List du vertical à celui-ci.

Type	ID	Name	Caption	Modifié	Liste versions
	200	Job Journal Templates	Modèles feuille projet		NAVW17.10
	201	Job Journal	Feuille projet		NAVW19.00.00.45243
	275	Job Journal Template List	Liste modèles f. projet		NAVW17.10

Exemple de Version List (Liste versions)

Du fait de la complexité du vertical NAViDMS, de très nombreuses modifications ont été réalisées sur les objets standards. Les équipes d'Eskape m'ont donc demandé de créer un programme réalisant l'ajout automatique d'une Version List sur, d'une part les nouveaux objets du vertical, et d'autre part les objets standards qui ont été modifiés. Pour cela, j'ai de nouveau enrichi l'outil de migration que j'ai développé d'une nouvelle fonctionnalité : l'outil d'ajout de Version List, qui modifie un fichier Dynamics Nav exporté en ajoutant si besoin la bonne Version List dans les propriétés de l'objet.

L'outil d'ajout de Version List réalise deux traitements différents en fonction du type d'objet à modifier :

- Les objets spécifiques sont directement modifiés pour ajouter la Version List si celle-ci n'est pas déjà présente.

- Les objets standards nécessitent un traitement plus poussé : les objets standards sont analysés un par un et comparés à un objet standard de référence (objet

standard non modifié). Si l'outil détecte une modification dans l'objet, celui-ci ajoute la Version List à l'objet.

L'outil d'ajout de Version List prend en entrée un fichier contenant les objets Dynamics Nav à modifier. Dans le cas d'objets standards, un fichier de référence contenant les mêmes objets doit être également fourni. Il fournit en sortie une liste des objets modifiés par rapport aux objets de référence.

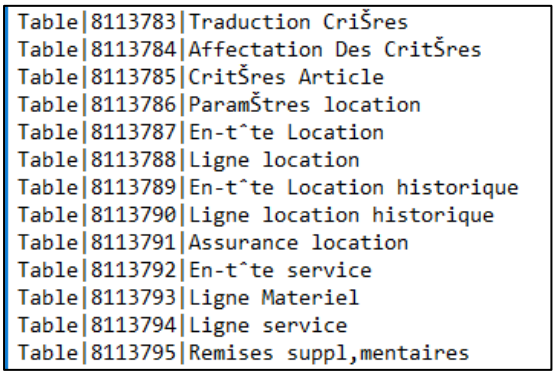
Comme l'outil de renumérotation, l'outil d'ajout de Version List est configurable afin de passer n'importe quelle chaîne de Version List au programme pour la modification.

V.6. Outil de génération de documentation Dynamics Nav :

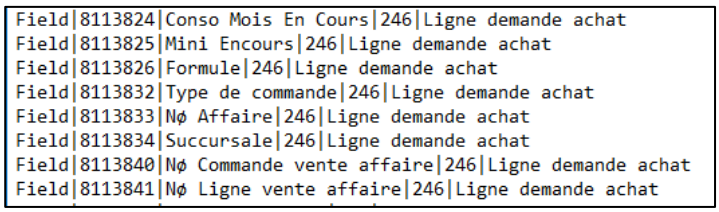
Le test de certification requiert que chaque table et champ ajouté dans un vertical soit documenté. Dans le cas d'un vertical comme NAViDMS, cela représente un nombre très conséquent de documentation interne à réaliser. La documentation interne de Dynamics Nav fonctionne conjointement avec un serveur Web qui héberge les fichiers de documentation en HTML. Lorsqu'un utilisateur demande l'ouverture de la documentation interne d'une table ou d'un champ, le client Nav ouvre un navigateur et affiche la page HTML correspondant à la documentation souhaitée.

La documentation interne des champs et tables de Dynamics Nav n'est pas un outil très utilisé, c'est pourquoi les équipes d'Eskape m'ont demandé de réaliser un programme permettant de générer automatiquement des pages de documentation HTML très basiques (avec uniquement la description du nom de la table ou du champ). Pour cela, j'ai une nouvelle fois enrichi l'outil de migration que j'ai développé d'une nouvelle fonctionnalité : l'outil de génération de documentation Dynamics Nav.

J'ai dans un premier temps cherché un moyen simple et efficace de trier les objets pour lesquels une documentation doit être générée. Je me suis une nouvelle fois rapproché des fonctionnalités offertes par Dynamics Nav et ses Codeunit. Les tables et champs nécessitant la génération d'un fichier de documentation sont les tables et champs précédemment renumérotés dans la plage fournie par Microsoft. Je me suis servi de cette information pour développer dans le Codeunit de migration une nouvelle méthode permettant cette fois de générer un fichier comportant toutes les tables et champs pour lesquels une documentation doit être générée. Cette méthode travaille autour des tables systèmes de Dynamics Nav qui stockent les informations de tous les objets et champs présents dans la base. En filtrant correctement sur les identifiants de ces champs et tables (les paramètres de la méthode sont les identifiants minimum et maximum de la plage renuméroté), la méthode génère un fichier .txt contenant les informations à inscrire dans la documentation HTML à générer.



Exemple de ligne d'information de table généré



Exemple de ligne d'information de champ généré

Ce fichier contenant toutes les tables et les champs pour lesquels une documentation doit être générée est ensuite fourni à l'outil de génération de documentation Dynamics Nav que j'ai développé dans l'outil de migration C# sur lequel j'ai travaillé tout au long de mon stage. Le programme va générer des fichiers HTML à partir de templates* de fichiers de documentation de tables et de champs.

V.7. Documentation de l'outil de migration :

L'outil de migration que j'ai développé pour les besoins de certification d'Eskape comporte une documentation technique interne au code C# de l'outil qui décrit toutes les méthodes que j'ai développé. De plus, le Codeunit de migration que j'ai réalisé est également commenté pour expliciter chaque étape de la migration en séparant la partie de migration plus générale de la partie spécifique à NAViDMS.

J'ai également réalisé une documentation précise des différentes possibilités de l'outil tout en expliquant son fonctionnement et la procédure classique de migration du vertical NAViDMS que j'ai mis en place tout au long de mon stage. La documentation est disponible en annexe du rapport (voir Annexe 2 : documentation de l'outil de migration).

V.8. Poursuite des étapes de certification :

Une fois les modifications spécifiques réalisées sur les différents objets du vertical NAViDMS dans le but de faire certifier ce dernier, j'ai continué mon travail autour de la documentation officielle de certification en triant les points non-réalisés et obligatoires pour la certification.

Comme expliqué précédemment, la certification permet à un module généraliste d'être officialisé ce qui implique que les entreprises partenaires de Microsoft qui travaillent autour de l'installation et du développement de l'ERP Microsoft Dynamics Nav obtiennent la possibilité de proposer à leurs clients le vertical certifié. Il s'agit d'un avantage conséquent pour Eskape puisque le module NAViDMS pourra être proposé dans toute la France. Cette ouverture du vertical à la revente dans tout le pays implique la réalisation de différents documents

permettant d'expliquer avec précision aux futures entreprises revendeuses (aussi appelées VAR*) les procédures d'installation, de désinstallation et de paramétrage.

Certaines informations spécifiques doivent également être fournies au moment du test à l'entreprise qui se charge de la certification, notamment un document Excel comprenant tous les objets modifiés et ajoutés par le vertical accompagné de ses caractéristiques (identifiant, nom, version list, distribution en tant que WebService, …).

V.9. Travaux complémentaires :

Ayant terminé mon projet quelques jours avant la fin de mon stage, j'ai pu apporter mon aide aux équipes d'Eskape sur quelques travaux complémentaires très intéressants et variés. J'ai notamment travaillé sur la réalisation de développements liés à des fiches d'écarts qui sont des documents techniques utilisés par Eskape pour expliciter une modification ou un ajout de fonctionnalité demandé par un client dans l'environnement Nav. Les fiches d'écarts décrivent avec précision le besoin du client et proposent des pistes de développement pour réaliser la modification.

J'ai par exemple eu la chance de travailler sur le développement de fonctionnalités d'un programme tournant sur des PSM (terminaux portables avec lecteurs de code-barres). Ce développement m'a permis de mettre en œuvre des technologies différentes de celles que j'ai pu utiliser en travaillant sur mon projet comme par exemple le développement sous l'OS Microsoft Windows CE (Embedded Compact) qui est une version minimaliste de Windows conçue pour des systèmes embarqués ou encore la mise en place d'appels à des Web Services de type SOAP qui est un protocole d'appel de procédures à distance (remote procedure call) basé sur une architecture XML.

VI. Bilan humain et technique

Le stage de fin de troisième année de Licence Informatique est un passage obligé qui finalise une année d'enseignement riche en apprentissages. Plus qu'un simple module, ce stage en milieu professionnel fait partie intégrante de la formation complète proposée en Licence Informatique.

Mon stage au sein de l'entreprise Eskape a été une expérience particulièrement enrichissante et motivante. L'accueil chaleureux qui m'a été réservé par les équipes des différents pôles d'Eskape m'a permis de m'intégrer rapidement dans l'entreprise. De plus, le projet que m'a proposé mon maître de stage était un projet à la fois technique et intéressant qui a nécessité l'apprentissage de nouveaux outils que je n'avais jusqu'alors jamais eu l'occasion d'utiliser. Grâce à toutes les connaissances que j'ai acquises au cours de mes années de formations et à l'aide précieuse de mon maître de stage et des membres de l'équipe, j'ai pu mener à bien les différentes tâches qui m'ont été demandée au cours des huit semaines de stage dans l'entreprise de services du numérique.

Réaliser un stage de deux mois au sein d'une entreprise dans le domaine de l'informatique est un atout indéniable dans la suite de notre parcours professionnel. Cette expérience très positive m'a confortée dans mon choix de me diriger vers le développement informatique de projets répondant à des besoins précis. De plus, réaliser mon stage dans le cadre d'une entreprise dynamique m'a permis d'en apprendre un peu plus sur le monde professionnel auquel je serai confronté dans quelques années.

ANNEXES

VI.1. Lexique :

Vertical : solution Dynamics Nav développée par une entreprise partenaire de Microsoft dans le but de répondre à une série de besoins métiers spécifiques. Un vertical reste un module qui se veut généraliste pour pouvoir s'installer facilement dans tous types d'entreprises d'un même secteur d'activité ayant des besoins semblables.

Progiciels de gestion intégré : abrégé PGI ou en anglais ERP (Enterprise Resource Planning) est une application très complète couplée à une base de données qui permet aux entreprises de consulter, conserver et réaliser des actions classiques de gestion d'entreprise.

Thread : processus léger qui réalise des traitements en parallèle d'autres processus. Les threads sont beaucoup utilisés dans la conception d'interfaces graphiques car ils permettent de réaliser des actions de manière asynchrone (notamment des modifications graphiques pendant qu'un autre thread exécute un traitement ou un action).

XML : Extensible Markup Langage (Langage de balisage extensible). Il s'agit d'un langage de balisage très structuré et hiérarchique. La lecture et l'écriture de fichiers XML peuvent se réaliser à partir de requêtes XQuery qui sont gérées facilement et efficacement dans des packages natifs en C#.

Template : modèle facilement adaptable et paramétrable. Dans ce cas précis, les templates de documentation permettent de générer des fichiers HTML très semblables et dont le contenu est paramétrable lors de la génération.

VAR : value-added resellers ou en français DVA : distributeur à valeur ajouté. Le terme désigne un revendeur qui exerce également une activité de service.

VI.2. Documentation utilisateur de l'outil de migration :

Outil de migration d'objets Dynamics Nav

Documentation

Modification et analyse d'objets Dynamics Nav

eskape
Présent pour votre Futur

Table des matières

I. Informations principales de l'outil

1. Utilisation et mise en place

L'outil de migration d'objets Dynamics Nav (MigrationTool) permet de réaliser la renumérotation d'objets spécifiques de Dynamics Nav en assurant la bonne conservation des liens dans les objets standards et spécifiques.

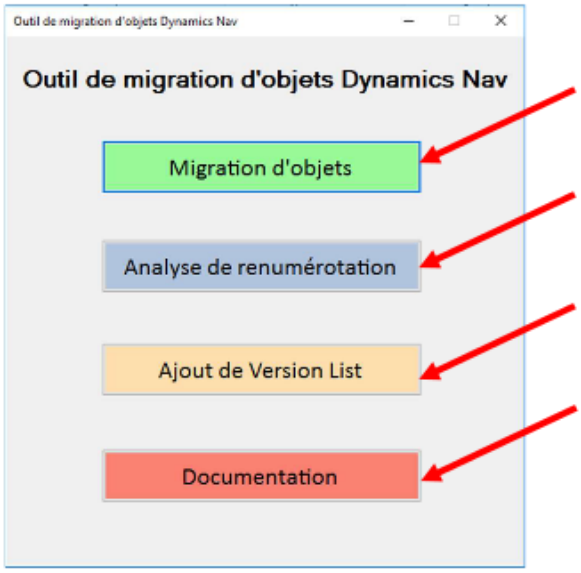
L'outil permet également d'analyser un fichier exporté Dynamics Nav renuméroté afin de chercher si des numéros d'objets sont écrits en dur dans un code C/AL.

Enfin, l'outil permet d'analyser des objets afin de définir s'ils ont été modifiés et d'ajouter une Version List personnalisée à ces objets ainsi que générer de la documentation Dynamics Nav automatique pour des nouvelles tables ou champs.

L'outil de migration d'objets doit toujours être dans le même dossier que son fichier de configuration config.xml.

2. Menu principal

Le menu principal redirige vers les interfaces permettant de lancer les traitements des trois fonctionnalités de l'outil.



Renumérotation et migration d'objets (voir II.1).

Analyse des objets renumérotés (voir II.2).

Ajout d'une Version List personnalisée (voir II.3).

Générer de la documentation automatique (voir II.4).

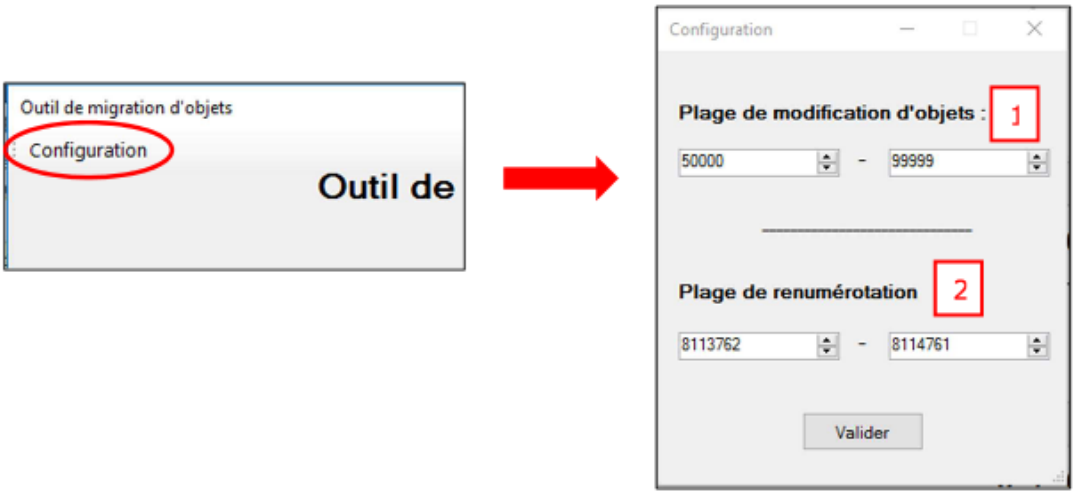
II. Fonctionnalités de l'outil de migration

1. Migration et renumérotation d'objets Dynamics Nav

Attention : DANS LE CAS D'UNE MIGRATION COMPLETE NAVIDMS :
- Importer d'abord le Codeunit 90001 Outil Migration (MigrationTool.fob).

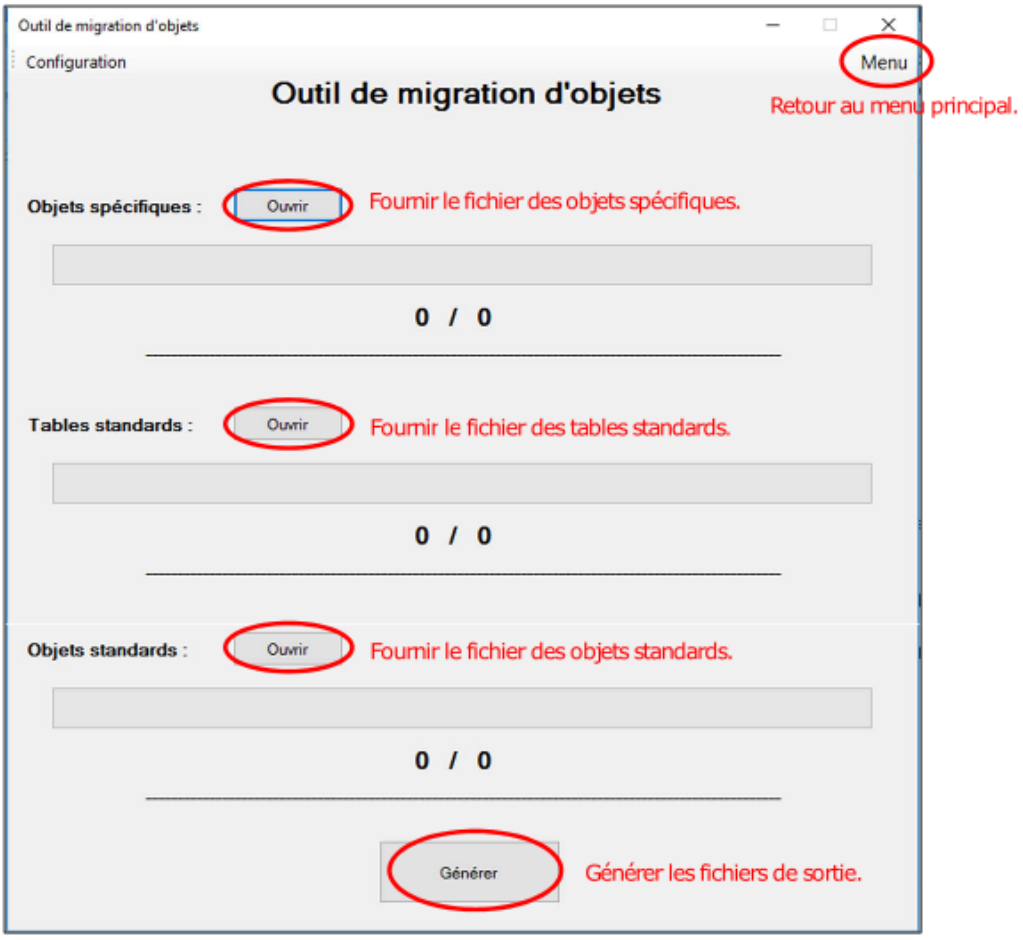
Exporter les objets en 3 fichiers .txt distincts :
a. Un fichier contenant tous les objets spécifiques.
b. Un fichier contenant uniquement les tables standards.
c. Un fichier contenant tous les objets standards (sauf les Tables et les MenuSuites non-modifiés).

Configurer la renumérotation à réaliser : bouton Configuration en haut à gauche de l'interface.



1. Plage de modification d'objets : les objets et champs de cette plage seront renumérotés.
2. Plage de renumérotation : les objets et champs seront renumérotés dans cette plage.

Fournir à l'outil de migration les 3 fichiers .txt exportés dans leurs emplacements respectifs.



Lancer la modification avec le bouton **Générer**. Une interface de sélection de dossier de destination apparaîtra.

La génération produit différents fichiers :

Mig_Modified_ + nom du fichier d'origine : fichier de migration des objets renumérotés.

Modified_ + nom du fichier d'origine : fichier des objets renumérotés.

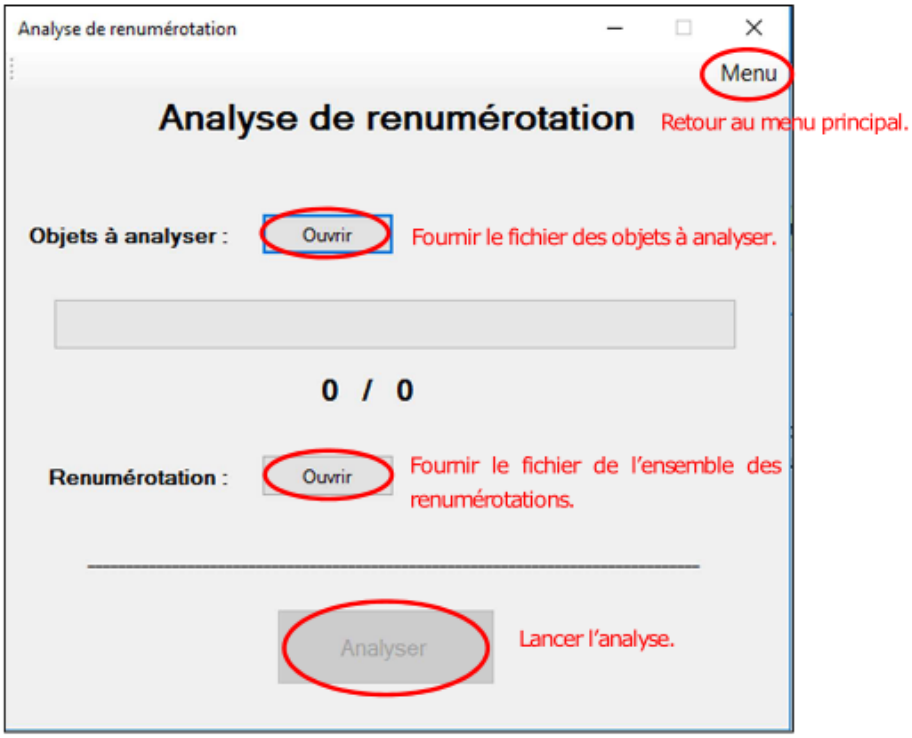
Ensemble_Renumerotation.txt : fichier contenant l'ensemble des identifiants renumérotés.

Renumerotation_ + type objet : fichier contenant la correspondance ancien identifiant / nouvel identifiant renuméroté pour le type d'objet.

2. Analyse des objets renumérotés

Fournir à l'outil d'analyse le fichier à analyser.

Fournir à l'outil le fichier contenant les numéros qui ont été renumérotés (Ensemble_Renumerotation.txt généré par l'outil de migration et de renumérotation).



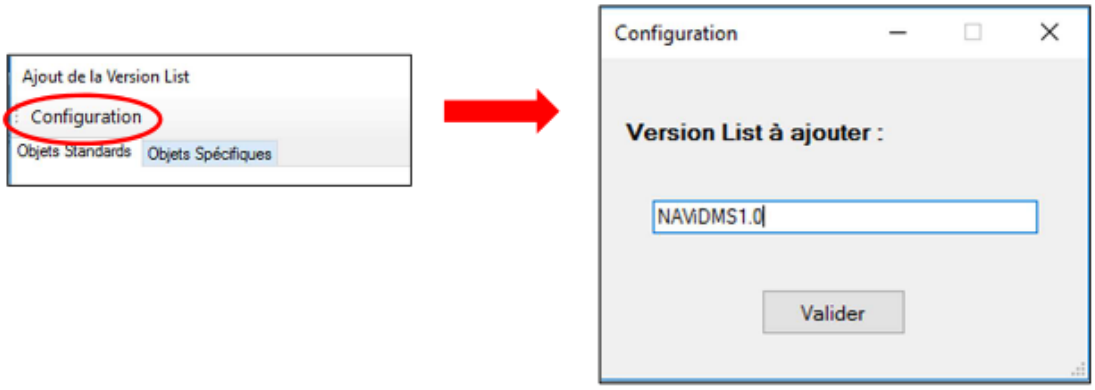
Lancer l'analyse avec le bouton **Analyser**. Une interface de sélection de dossier de destination apparaîtra.

L'outil d'analyse produit un fichier de sortie contenant le numéro de ligne et la ligne dans laquelle est présent un numéro qui a été renuméroté précédemment.

3. Ajout d'une Version List personnalisée

a)Paramétrage

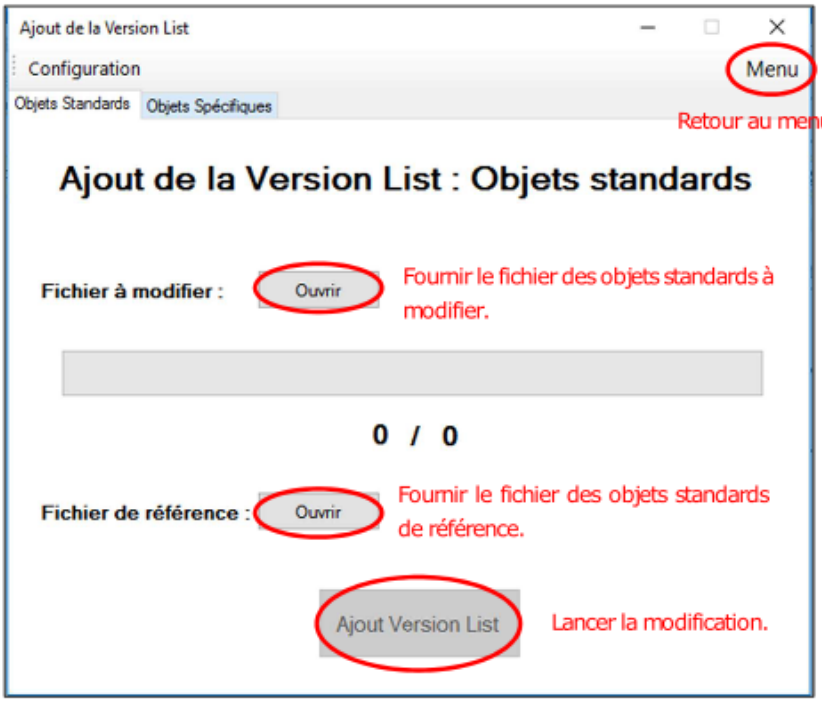
Configurer la Version List à ajouter : bouton **Configuration** en haut à gauche de l'interface.



b) Ajout d'une Version List pour les objets standards

Fournir à l'outil d'ajout de Version List le fichier .txt des objets standards à modifier.

Fournir à l'outil d'ajout de Version List un fichier .txt des objets standards de référence (objets standards sans modifications spécifiques).

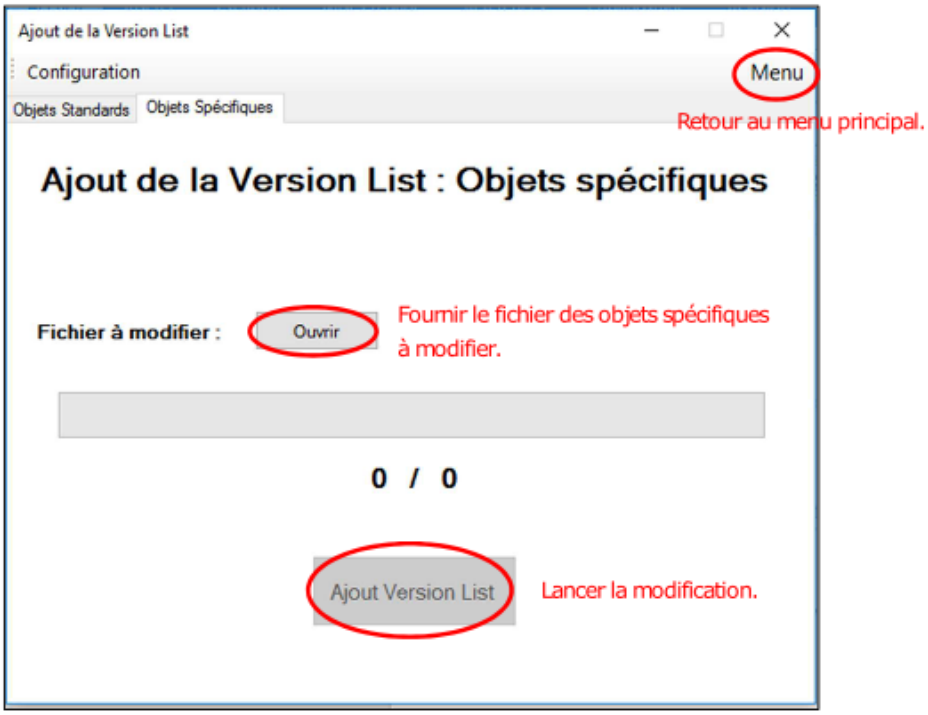


Lancer la modification avec le bouton **Ajout Version List**.

L'outil d'ajout de Version List modifie directement le fichier fourni. L'outil génère également un fichier ListeObjetsModifiés.txt dans le dossier source du fichier fourni qui liste tous les objets standards qui ont été modifiés.

c) Ajout d'une Version List pour les objets spécifiques

Fournir à l'outil d'ajout de Version List le fichier .txt des objets spécifiques à modifier.



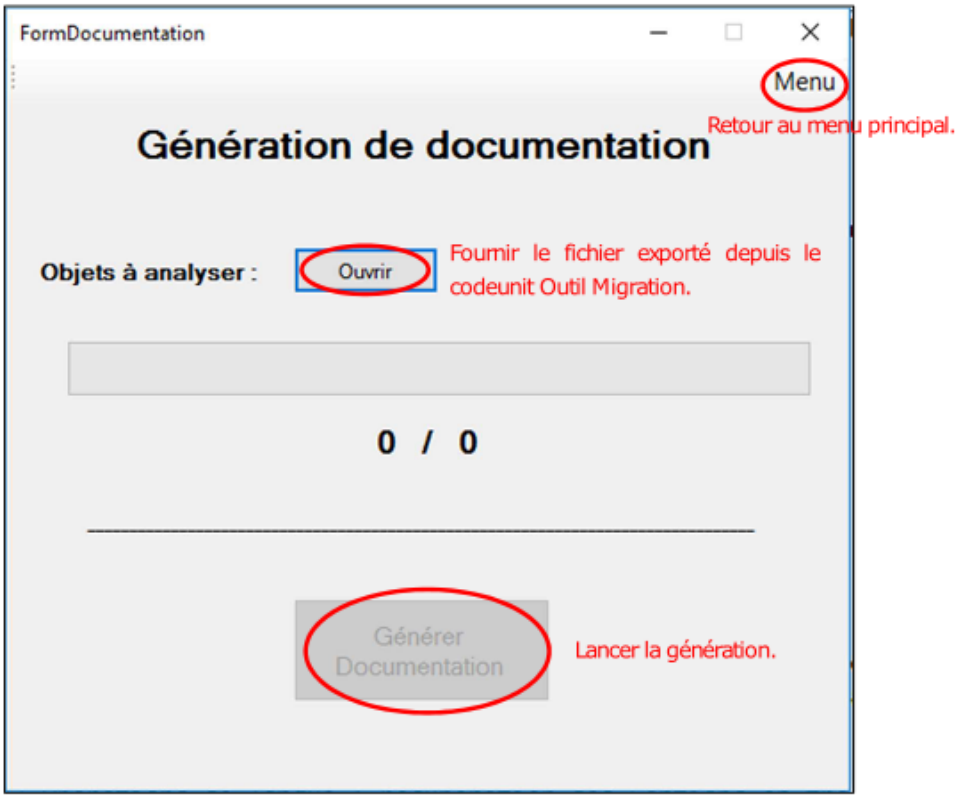
Lancer la modification avec le bouton **Ajout Version List**.

L'outil d'ajout de Version List modifie directement le fichier fourni.

4. Génération automatique de documentation Dynamics Nav

Lancer la méthode **ExportForDocumentation(MinID, MaxID)** du codeunit Outil Migration. Sauvegarder le fichier généré par la méthode.

Fournir à l'outil de documentation automatique le fichier exporté par la méthode précédente du codeunit Outil Migration.



Lancer la modification avec le bouton **Générer Documentation**. Une interface de sélection de dossier de destination apparaîtra.

L'outil fourni en sortie une série de fichiers de documentation HTML générés automatiquement. Pour mettre en place les fichiers sur le serveur d'aide Dynamics Nav, copier tous les fichiers dans le dossier contenant les pages d'aide sur le serveur Nav.

III. Procédures d'import des objets NAViDMS

1. Procédure d'import sans migration des données

1. Importer les fichiers dont le nom commence par Modified_,

2. Compiler tous les objets.

3. Faire le paramétrage.

4. Faire un RUN sur le codeunit ManagementCompany.

5. Importer le codeunit généré.

2. Procédure d'import avec migration des données

1. Supprimer tous les objets spécifiques qui ne sont pas des tables (sauf la page 70089 Workshop Group Center).

2. Renommer la page 70089 Workshop Group Center en Workshop Group Center_.

3. Importer le fichier de migration des tables spécifiques (Mig_Modified_ + nom du fichier des objets spécifiques).

4. Importer les objets spécifiques renumérotés (Modified_ + nom du fichier des objets spécifiques).

5. Importer les objets standards modifiés (Modified_ + nom du fichier des objets standards).

6. Importer le fichier de migration des tables standards (Mig_Modified_ + nom du fichier des tables standards).

7. Compiler tous les objets.

8. Faire le paramétrage de la table 2000000072 Profile : modifier les numéros des tables spécifiques (voir fichier RenumerotationTables.txt).

9. Faire un RUN sur le codeunit Outil Migration.

10. Supprimer les tables spécifiques dans la plage non-souhaitée, la page 70089 et le codeunit Outil Migration.

11. Importer le fichier des tables standards modifées (Modified_ + nom du fichier des tables standards).

12. Compiler (avec l'option Force) les tables standards.