UFR SCIENCES & TECHNIQUES
UNIVERSITÉ DE FRANCHE-COMTÉ

UNIVERSITÉ DE FRANCHE-COMTÉ

Rapport de stage

Licence 3 informatique
Année 2023-2024

Développement d'un logiciel d'édition pour système dédié

Etudiant :
Joris BAUD

Maitre de stage :
Christophe PAGNIER

Tuteur universitaire :
Bruno TATIBOUËT

Remerciements

Je tiens tout d'abord à exprimer mes remerciements envers toutes les personnes qui ont contribué à la réalisation de ce rapport de stage.

Je remercie chaleureusement mon maître de stage, M. Christophe Pagnier, pour son encadrement et son temps tout au long de ce stage.

Je souhaite également adresser mes sincères remerciements à mon tuteur universitaire, M. Bruno Tatibouët, pour son suivi, ses conseils et sa disponibilité.

Je tiens à exprimer ma reconnaissance envers toute l'équipe de l'entreprise Vermot Automation pour son accueil chaleureux, sa collaboration et son partage de connaissances.

Un merci particulier à Fabien Peureux pour ses précieux enseignements sur les techniques de rédaction. Ses cours ont été d'une grande aide pour structurer et présenter ce rapport.

Table des matières

1 Introduction **3**

2 Présentation de l'entreprise **4**
2.1 Historique de l'entreprise 4
2.2 Organisation 4
2.3 Cœur de Métier 5

3 La mission **7**
3.1 Cahier des Charges 7
3.1.1 La Gamme Motion 7
3.1.2 Les Objectifs 8
3.2 Méthode et Outils de Travail 8
3.2.1 Méthode de Travail 8
3.2.2 Logiciel de Développement 9

4 Le travail réalisé **12**
4.1 Gestion des Machines 12
4.1.1 Édition des machine 12
4.1.2 Libraires développée 13
4.2 Gestion des programmes 14
4.2.1 Outil de transfert de programme 14
4.2.2 Visualisation et édition de programme 16
4.3 Gestion des paramètres 18
4.3.1 Outil de connexion rapide 18
4.3.2 Visualisation de paramètres 19
4.4 Outils annexes 21
4.4.1 Permissions et Droits 21
4.4.2 Gestion des utilisateurs 21
4.4.3 Gestion des répertoires et fichiers 23
4.5 Gestion des licences 23

5 Synthèse et Perspectives **25**
5.1 Vente de l'application 25
5.2 Possibilité d'évolution 25
5.3 Retour sur expérience 26

6 Conclusion **27**

1 Introduction

Dans le cadre de la troisième année de Licence d'informatique à l'UFR Sciences et Techniques de Besançon, les étudiants sont tenus d'effectuer un stage dans un service correspondant à leur spécialité au sein d'une entreprise. Ce stage, d'une durée minimale de 10 semaines, vise à compléter la formation au travers d'une expérience professionnelle permettant de mettre en application les connaissances acquises au cours des trois années précédentes.

J'ai effectué mon stage au sein du bureau d'étude de Vermot Automation, située au Valdahon. L'entreprise est spécialisée dans la modernisation des systèmes de décolletage à cames avec pour objectif de répondre à un paysage industriel en constante évolution, marqué par des avancées technologiques rapides et des exigences croissantes en matière d'efficacité et de productivité. L'entreprise se positionne comme un acteur clé dans le domaine de l'automatisation et de l'optimisation des processus de fabrication en créant des environnements de fabrication intelligents, capables de s'adapter rapidement aux évolutions du marché et aux exigences de production. Forte de ses 40 années d'expertise et d'innovation, l'entreprise s'efforce de proposer des solutions adaptées aux besoins spécifiques de ses clients.

En tant que stagiaire, j'ai eu l'opportunité de contribuer à un projet visant à développer un nouvel outil logiciel destiné à améliorer la gestion et la programmation des systèmes MOTION. Cette initiative s'inscrit dans une démarche d'innovation continue de l'entreprise, visant à offrir à ses clients des solutions toujours plus performantes et adaptées.

Dans le cadre de ce projet, l'un des principaux défis était de m'immerger dans un environnement technique afin d'acquérir une compréhension du domaine du décolletage à cames. Cela impliquait de me familiariser avec le vocabulaire spécifique du milieu, de comprendre les besoins et attentes des utilisateurs finaux, ainsi que les spécificités des systèmes MOTION déjà en place.

Ce rapport de stage est le fruit de mon expérience au sein de Vermot Automation. Il vise à présenter de manière détaillée le travail accompli, les défis rencontrés et les résultats obtenus au cours de cette période de stage enrichissante. À travers ces pages, je partagerai quelques points techniques et ma contribution à ce projet.

2 Présentation de l'entreprise

2.1 Historique de l'entreprise

La société Vermot Automation a été fondée en 1984 avec pour mission initiale la réalisation de prestations de maintenance et de dépannage industriel, principalement destinées aux entreprises de décolletage. À ses débuts, l'entreprise se concentrait sur la résolution des problèmes techniques rencontrés par ses clients dans le domaine de la fabrication de pièces mécaniques.

Dans les années qui ont suivi sa création, Vermot Automation a rapidement élargi son champ d'activité. En 1989, elle a établi un bureau d'études interne afin de développer des solutions sur mesure basées sur des automates programmables, répondant ainsi aux demandes spécifiques de ses clients. Cette même année, l'entreprise a lancé la fabrication en série de coffrets de rénovation pour le décolletage, renforçant ainsi sa position en tant que fournisseur de solutions complètes pour l'industrie du décolletage .

En 1993, Vermot Automation a lancé un nouvel équipement appelé l'OVAC [1], qui a permis d'améliorer les performances des machines utilisées dans le processus de décolletage.

Depuis cette période, Vermot Automation a continué à grandir. En 1994, elle a déménagé dans la zone industrielle de Valdahon, consolidant ainsi sa présence dans la région. Au fil des ans, l'entreprise a continué à innover en modernisant ses équipements et en dérivant son système OVAC pour répondre aux besoins changeants de l'industrie.

2.2 Organisation

L'entreprise Vermot Automation est de taille modeste, composée de seulement quatre salariés, ce qui en fait une très petite entreprise (TPE) sous le statut de SAS.

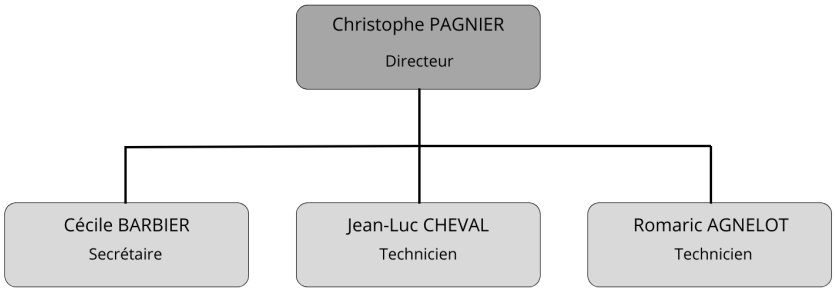

FIGURE 2.1 – organigramme de l'entreprise

En tant que directeur, Christophe Pagnier est responsable de la gestion globale de l'entreprise. Ses missions incluent la supervision des opérations quotidiennes, de la planification des projets,

1. À l'origine Optimisation en Vitesse de l'Arbre à Cames

et la gestion des relations avec les clients et les partenaires.

Cécile Barbier, en tant que secrétaire, est le pivot administratif de l'entreprise. Ses tâches incluent la gestion des communications internes et externes, de la gestion des documents administratifs, du traitement des commandes et des factures, ainsi que de l'accueil des clients et des fournisseurs.

Jean-Luc Cheval, technicien dans l'entreprise depuis 1989 est la mémoire de l'entreprise. Du fait de son ancienneté Jean-Luc connaît très bien les anciens systèmes.

Romaric Agnelot lui aussi technicien depuis 2012 est spécialisé dans les nouveaux systèmes. Au sein de l'atelier, il s'occupe du montage en série des coffrets électrique ainsi que des schémas électrique.

2.3 Cœur de Métier

Le cœur de métier de Vermot Automation repose sur le décolletage à cames. Le décolletage à cames consiste à la réalisation par usinage de petites pièces de micromécaniques par enlèvement de matières à partir d'une barre (de titane, d'acier, d'inox, d'aluminium...). La matière brute introduite dans la zone d'usinage est façonnée par une sérié d'outils coupants qui, selon leur nombre, leur forme et leur disposition, permettront d'obtenir des pièces plus ou moins complexes, de tailles diverses et de précisions variables. Les pièces sont usinées les unes à la suite des autres dans la barre de matière. L'usinage de la pièce est réalisé par des mouvements qui sont générés par un arbre à cames qui obéit à des instructions. Ce procédé est particulièrement utilisé pour la production en grande série de petites pièces complexes, essentielles dans des industries telles que l'automobile, l'aéronautique, et l'horlogerie.

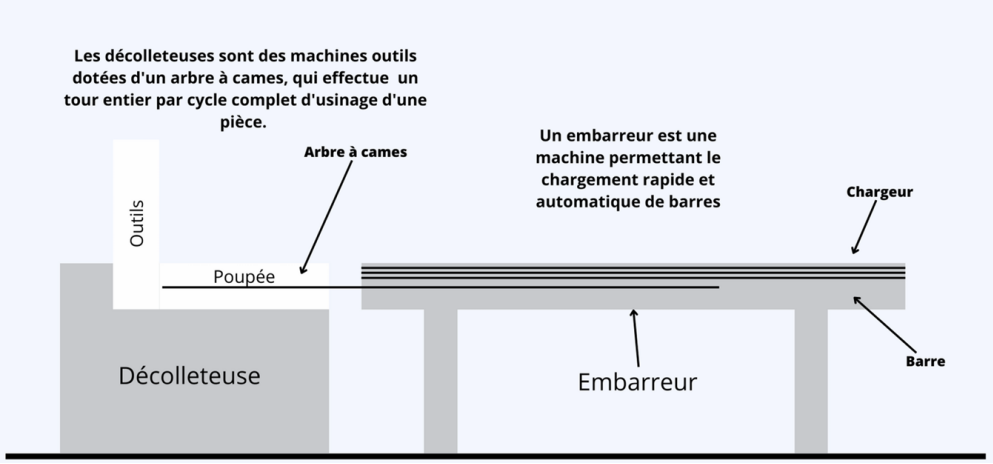

FIGURE 2.2 – Schéma d'une décolleteuse à cames

Vermot Automation se spécialise dans la modernisation et l'automatisation de ces machines de décolletage à cames, en intégrant des coffrets technologiques pour améliorer leurs performances et leur efficacité. Ces coffrets permettent d'ajouter des systèmes de contrôle numérique aux machines existantes, souvent anciennes, permettant aux entreprises de réaliser des économies

substantielles en évitant l'achat de nouvelles machines. Les coffrets apportent plusieurs avantages : ils améliorent la précision des machines en permettant des ajustements plus fins des paramètres, augmentent la productivité en automatisant certaines tâches et en optimisant les processus, et ils offrent une flexibilité accrue grâce à une reprogrammation facile des machines pour produire différentes pièces sans modifications matérielles complexes. De plus, la digitalisation des machines permet un suivi en temps réel des performances et des diagnostics, facilitant la maintenance proactive et réduisant les temps d'arrêt imprévus.

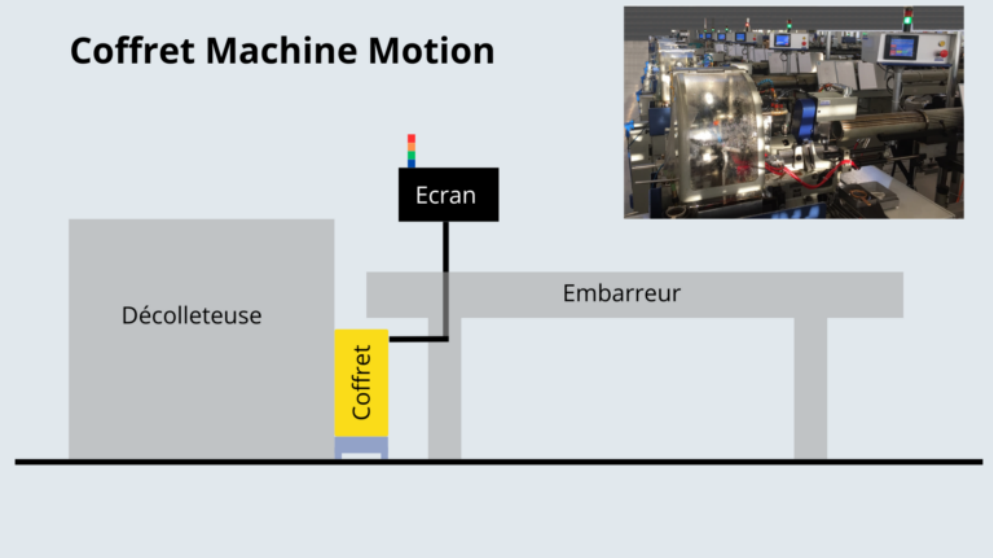

FIGURE 2.3 – Ajout du coffret sur une décolleteuse à cames

Les entreprises susceptibles de faire appel aux services de Vermot Automation proviennent de divers secteurs industriels, principalement situés en Suisse et en Haute-Savoie, où l'industrie du décolletage est prédominante. Située à Valdahon en Franche-Comté, Vermot Automation peut intervenir rapidement pour installer ses produits, effectuer de la maintenance et assurer le dépannage.

3 La mission

3.1 Cahier des Charges

3.1.1 La Gamme Motion

Un des produits phares de l'entreprise est la gamme Motion. Ce sont des coffrets de rénovation conçus pour moderniser les machines de décolletage traditionnelles. Il en existe plusieurs catégories tel quel les Coffrets Machine Motion, STB Motion ou bien les Barboy Motion qui ont chacun leurs spécificités. Ces coffrets permettent d'ajouter des fonctionnalités avancées comme la gestion électronique des mouvements, augmentant ainsi considérablement la précision et la vitesse des opérations de décolletage. L'intérêt des coffrets réside dans leur capacité à prolonger la durée de vie des machines existantes tout en améliorant leurs performances. La gamme Motion intègre un écran tactile permettant de commander diverses fonctions comme le suivi production en temps réel avec des informations sur le nombre de pièces produites et l'état de la machine ou bien de programmer jusqu'à dix programmes de décolletage différents qui vont faire succéder une suite de plusieurs mouvements sur les différents composants (arbre à cames, poupée, embarreur...) dans le but d'usiner une pièce. Parmi les options disponibles, certaines améliorent significativement l'efficacité des machines, telles que le pilotage du séparateur à godets pour une gestion optimisée de l'échantillon et de la production, la motorisation et le pilotage des appareils pour une gestion avancée de la vitesse, et des fonctionnalités de communication pour transférer les données de production vers un logiciel ERP. [1]. D'autres options incluent l'ajout de relais auxiliaires pour piloter des électrovannes et des palpeurs, et l'intégration de colonnes lumineuses pour des indications visuelles immédiates. En moyenne, cette technologie augmente la productivité de 30%.

FIGURE 3.1 – Interface OVAC MOTION

1. Plus de détail sur les ERP : `https://www.gestisoft.com/fr/blogue/qu-est-ce-qu-un-logiciel-erp`

3.1.2 Les Objectifs

L'objectif de mon stage est de concevoir entièrement un logiciel multifonctionnel nommé VSoftMotion qui devra répondre aux exigences spécifiques d'un utilisateur novice ou confirmé. Pour atteindre cet objectif, le développement se concentrera sur plusieurs aspects clés.

Premièrement, la conception d'une interface utilisateur intuitive. Celle-ci devra être conviviale et ergonomique, afin de permettre aux utilisateurs, qu'ils soient opérateurs machines ou administrateurs, de naviguer facilement à travers les différentes fonctionnalités du logiciel. Une attention particulière sera portée à la disposition des éléments de l'interface, à la clarté des instructions et à la facilité de navigation entre les différentes sections.

Ensuite, l'aspect central du logiciel sera la gestion des programmes des machines MOTION. Pour les clients. Ceux-ci auront la possinilité d'importer et exporter des programmes entre un ordinateur et une machine. De plus, les clients devront avoir la possibilité d'éditer directement les fichiers de programme pour adapter les processus de fabrication à leurs besoins spécifiques. Cette fonctionnalité déjà présente sur les machines devra être développée de manière à être similaire dans le logiciel tout en restant intuitif et accessible même pour les utilisateurs moins expérimentés.

Pour les administrateurs, le logiciel devra offrir des outils de gestion avancés pour les fichiers de paramètres des machines. Cela inclura la possibilité d'importer, d'exporter et de visualiser ces fichiers. L'objectif est de permettre aux administrateurs de maintenir une configuration précise et cohérente des machines, tout en facilitant la gestion des mises à jour des machines et de connaitre les contraintes de celle-ci.

De plus, le logiciel devra inclure un système de gestion des machines pour permettre aux utilisateurs de suivre et d'enregistrer facilement leur machine, même en l'absence de connexion avec celle-ci. Cela implique la création d'un lieu de stockage de données, avec des informations détaillées sur chaque équipement ainsi que la possibilité de stocker le fichier de paramètres associés. L'objectif est de fournir aux utilisateurs et administrateur un moyen efficace de visualiser facilement leur parc de machines.

Enfin, la gestion des permissions sera un aspect crucial du développement du logiciel. Cela consistera à mettre en place un système de contrôle d'accès robuste et modulable, permettant de définir les niveaux d'autorisation pour les différents utilisateurs utilisant le logiciel. Cela à pour objectif que chaque utilisateur ait accès aux fonctionnalités appropriées, en fonction de son rôle et de ses responsabilités dans l'organisation.

3.2 Méthode et Outils de Travail

3.2.1 Méthode de Travail

Pour mener à bien ce projet de développement, nous avons adopté une méthode de travail structurée et collaborative. Christophe Pagnier, le directeur de Vermot Automation, qui est également le seul à posséder des compétences en développement au sein de l'entreprise, a joué un rôle central dans ce processus. M Pagnier me fournissait des consignes claires et détaillées sur les fonctionnalités et les améliorations à apporter. Ces directives étaient documentées dans

un fichier nommé "Suivi de projet" ou alors directement sur un journal de note personnelle.

La documentation de suivi de projet a permis de conserver une trace précise de toutes les demandes et exigences spécifiques. En parallèle, je maintenais un autre document où étaient notées quotidiennement les tâches accomplies, les défis rencontrés et les points bloquants. Ce journal de bord servait non seulement à suivre l'avancement du projet, mais aussi à identifier les obstacles techniques ou opérationnels et à élaborer des solutions pour les surmonter.

Une fois les consignes mises en œuvre, mon travail était présenté dans un premier temps à Monsieur Pagnier pour une première validation. Cette phase de validation interne était cruciale pour s'assurer que les développements réalisés correspondaient aux attentes initiales et respectaient les normes de qualité de l'entreprise. Une fois cette validation effectuée, nous passions à une phase de tests plus approfondie impliquant les autres membres de l'équipe.

Les membres de l'entreprise étaient impliqués dans la phase de test afin de recueillir leurs retours sur les nouvelles fonctionnalités et les améliorations à apporter. Leurs commentaires étaient précieux pour identifier les éventuels bugs, les incohérences ou les améliorations possibles. Cette étape permettait de garantir que le logiciel répondait non seulement aux spécifications techniques, mais aussi aux besoins pratiques des utilisateurs finaux.

Après avoir intégré les retours et effectué les ajustements nécessaires, une nouvelle version du logiciel était finalisée. Chaque version validée du logiciel était soigneusement archivée dans un dossier dédié sur le serveur de l'entreprise. Cette pratique a permis de garantir la traçabilité des différentes versions et facilitait la gestion des mises à jour. En parallèle, une notice explicative détaillée était rédigée pour chaque version, décrivant les nouvelles fonctionnalités, les corrections apportées et les instructions d'utilisation.

3.2.2 Logiciel de Développement

Pour le développement de ce projet, il m'a été demandé d'utiliser Delphi, un environnement de développement intégré [2] déjà en usage pour les applications existantes chez Vermot Automation. Delphi est reconnu pour sa rapidité, sa flexibilité et sa capacité à produire des applications robustes et performantes. Il propose une interface utilisateur intuitive qui facilite la prise en main de l'outil et qui permet de commencer à construire les premiers visuels d'un logiciel rapidement et efficacement. Bien que je n'avais jamais utilisé Delphi auparavant, j'ai trouvé l'outil très intuitif et quelques petits tutoriels trouvé sur internet ont suffi pour me familiariser avec l'outil et commencer le développement.

Delphi utilise deux types de fichiers principaux pour gérer les interfaces utilisateur et le code associé : les fichiers .dfm et .pas. Les fichiers .pas contiennent le code source écrit en Pascal Object, qui définit la logique de l'application et les comportements des composants visuels. Les fichiers .dfm [3], quant à eux, sont des fichiers de définition de formulaire qui stockent la disposition et les propriétés des composants visuels sur les formulaires. En d'autres termes, les fichiers .dfm décrivent comment l'interface utilisateur est structurée visuellement, tandis que les fichiers .pas définissent ce que fait l'application.

Delphi présente plusieurs avantages notables. Grâce à ses composants visuels réutilisables et

2. Plus de détail sur les IDE : `https://www.redhat.com/fr/topics/middleware/what-is-ide`
3. Delphi Form File

à sa bibliothèque VCL [4], il permet un développement rapide d'interfaces utilisateur. De plus, Delphi offre une capacité multi-plateforme, permettant de développer des applications pour Windows, macOS, iOS et Android avec une base de code unique.

Cependant, Delphi a aussi ses inconvénients. Le coût de sa licence professionnelle (1869 euros) peut être un frein, surtout pour les petites entreprises ou les développeurs indépendants. Il est également moins flexible en termes d'intégration avec d'autres outils et services comparé à des environnements de développement open-source comme Eclipse ou Visual Studio Code, et dispose de beaucoup moins de tutoriels ou de documentation par rapport à un atelier de génie logiciel tel que WinDev.

Pour pallier ces problèmes, j'ai pris la décision d'utiliser Delphi uniquement pour le design des pages (les fichiers .dfm) et de gérer l'ensemble du code source (.pas) grâce à Visual Studio Code. J'ai choisi Visual Studio Code pour son interface conviviale et ses nombreuses extensions et en particulier GitHub Copilot, qui a pour rôle de fournir une assistance intelligente dans l'écriture du code. Son rôle est de suggérer une autocomplétion du code de manière contextuelle. Cela m'a permis de gagner du temps et ainsi d'être beaucoup plus productif.

Pour ce qui est de Delphi, que j'utilise pour la construction des différentes fenêtres du logiciel, l'utilisation est relativement simple. On commence par placer des composants visuels sur une fenêtre de l'application. Ces composants peuvent être des boutons, des champs de texte, des menus, etc. Chaque composant a des propriétés qui peuvent être configurées, telles que la taille, la couleur ou le texte affiché. En plus de ces propriétés, chaque composant peut réagir à des événements, comme un clic de souris ou une entrée de clavier.

Pour ce qui est du Pascal Object utilisé pour le code source c'est un langage orienté objet dérivé du Pascal créé par la société Borland. Il permet de structurer le code en classes et objets, facilitant ainsi la modularité et la réutilisation du code. Les classes peuvent contenir des champs, des propriétés, des méthodes et des événements, permettant de définir diffèrent comportements. Dans le cadre de Delphi, il est utilisé pour implémenter la logique fonctionnelle derrière les interfaces utilisateur, rendant les applications interactives et dynamiques. Par exemple, un bouton peut être associé à une méthode qui s'exécute lorsqu'il est cliqué, et cette méthode peut à son tour interagir avec d'autres composants ou effectuer des opérations sur des données.

4. La Bibliothèque VCL (Visual Component Library) est un framework écrit en Pascal Objet par Borland pour les outils de développement Delphi et C++ Builder

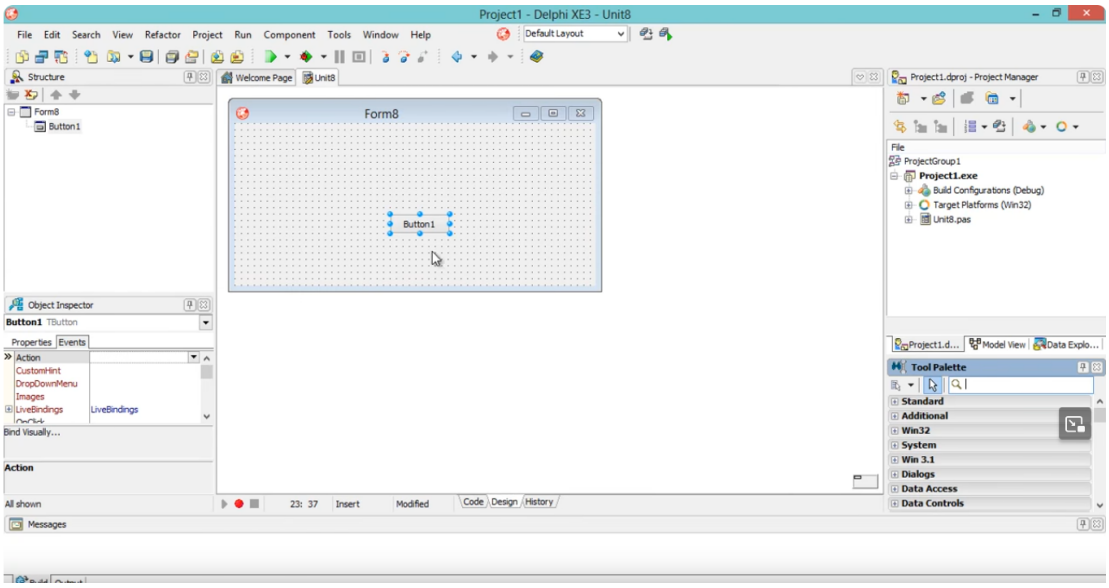

FIGURE 3.2 – Interface Delphi

Une fois cette phase de développement terminé, il est possible de compiler le projet grace à Delphi. Cette phase de compilation crée un fichier exécutable utilisable indépendamment des fichiers source.

4 Le travail réalisé

Avant de commencer le développement, il a fallu cerner les besoins et définir la structure du projet une fois compilé. Nous avons très vite cerné la nécessité de créer des fichiers externes pour stocker les données de machine, d'utilisateur et les paramètres logiciels variables propres à chaque client. Pour stocker ces données, il a été choisi d'utiliser des fichiers au format .json. Ce format, largement utilisé et reconnu, présente l'avantage d'avoir déjà des bibliothèques existantes, capables de parser les données. Grâce à ces bibliothèques, nous n'avons pas à écrire de code pour interagir avec ces fichiers. Nous disposons aussi d'un fichier LICENCE.TXT qui contient les informations de licence du logiciel ainsi que la clé d'activation du logiciel afin d'éviter des copies illicites.

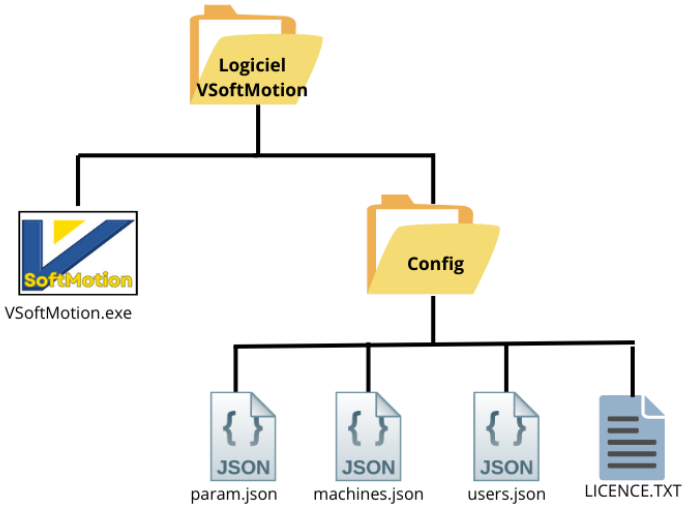

FIGURE 4.1 – Arborescence de la configuration

4.1 Gestion des Machines

4.1.1 Édition des machine

Le logiciel est conçu pour communiquer avec un parc de machines. Pour simplifier la connexion, un administrateur viendra renseigner et sauvegarder les informations de chaque machine via une interface dédiée appelée 'Editeur Machine'. Cela permet d'enregistrer des machines pour, à l'avenir, s'y connecter plus facilement en sélectionnant simplement la machine souhaitée. Les informations de chaque machine sont stockées dans le fichier machines.json, incluant le nom de la machine, un numéro d'identification, le type de contrôleur, l'adresse IP, l'identifiant de

connexion, le mot de passe, le port et le type de machine. Un fichier de paramètres est également associé à chacune d'entre elle.

Lors de l'ajout d'une machine, si la connexion est réussie, un fichier de paramètres présent sur celle-ci peut être sélectionné. En cas d'échec de la connexion, un fichier de paramètres peut être choisi sur l'ordinateur, cependant, il n'y aura aucune garantie que ces paramètres correspondent à celui réellement présent sur la machine et doit donc être effectué avec réflexion. Les informations enregistrées peuvent être modifiées ou supprimées, et les paramètres peuvent être visualisés via la même interface.

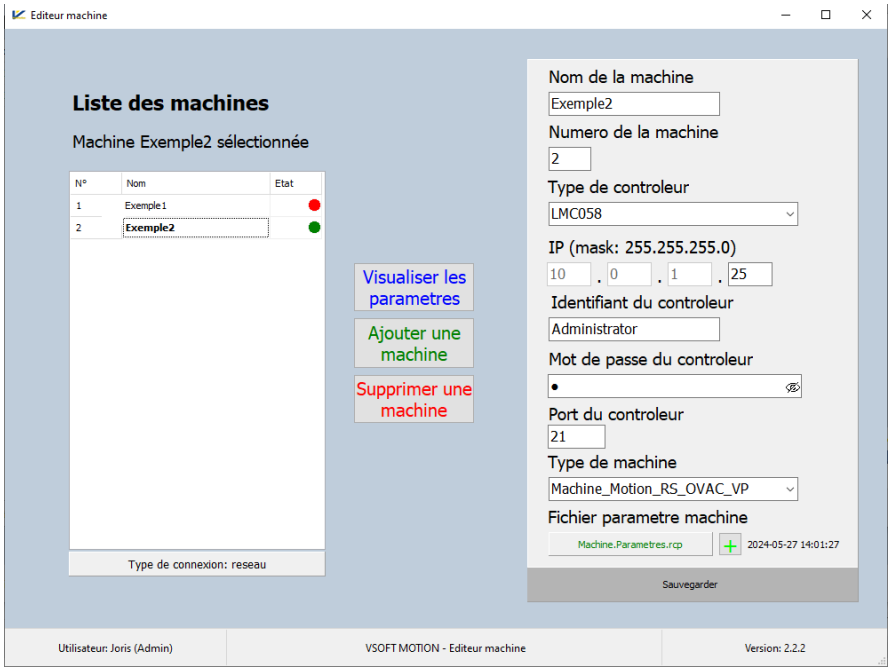

FIGURE 4.2 – Interface éditeur machine

L'intérêt d'enregistrer les machines est aussi qu'une fois celle-ci enregistrée, il sera possible de travailler sur des programmes sans être connecté à celle-ci, tout en ayant conscience des contraintes qu'elle impose via son fichier de paramètres associés.

4.1.2 Libraires développée

Chaque machine est représentée comme un objet dans le logiciel. Pour gérer ces objets de manière efficace, une librairie a été développée avec des fonctions spécifiques afin de faciliter les differentes manipulations. Cette librairie permet de créer des objets représentant les machines et de gérer leurs paramètres de façon structurée et centralisés.

```
TMachine = class
public
    FName: string;
    FNumero : integer;
    FIP: string;
    FUser: string;
    FPassword: string;
    FPort: Integer;
    FHasLoadParameters: Boolean;
    FNameOfFileParameters: string;
    FPathOfFileParameters: string;
    FAssociatedDate: string;
    FParameters: TDictionary<string, string>;
    FTypeMachine: TypeMachine;
    FTypeControleur: TypeControleur;
```

Cet objet TMachine contient plusieurs attributs essentiels tels que le nom de la machine (FName), son numéro (FNumero), son adresse IP (FIP), ses identifiants de connexion (FUser et FPassword), ainsi que des informations sur le fichier de paramètres associé (FNameOfFileParameters, FPathOfFileParameters, FAssociatedDate) et le type de machine et de contrôleur (FTypeMachine, FTypeControleur). Les paramètres de la machine sont stockés dans un dictionnaire (FParameters).

Pour manipuler ces objets, nous avons préalablement développé dans la librairie diverses fonctions. Par exemple, pour vérifier si une machine est joignable, on utilise la fonction Ping qui envoie une requête à l'adresse IP d'une machine donnée pour vérifier si elle répond :

```
function Ping(machine: TMachine): Boolean;
```

Une autre fonction permet de récupérer la valeur d'un paramètre spécifique stocké dans le dictionnaire FParameters de l'objet TMachine passé en paramètres :

```
function getValueOfParameters(machine: TMachine; key: string): string;
```

4.2 Gestion des programmes

4.2.1 Outil de transfert de programme

Cet outil sera accessible à tout type d'utilisateur et se doit d'être compréhensible par un novice.

Dans un premier temps, l'utilisateur choisira la machine sur laquelle il souhaite se connecter (Cette machine doit avoir été préalablement enregistrée dans l'éditeur de machines.). Si la

connexion est réussie, l'utilisateur visualisera les fichiers sur le PC ainsi que les fichiers programmes sur la machine.

Pour garantir une compréhension facile pour tous, un composant similaire à l'explorateur de fichiers Windows a été choisi, car c'est un outil familier que la plupart des personnes savent utiliser. Dans Delphi, ce composant est constitué de 2 sous-composants. Le premier est un ShellListView et représente la liste des fichiers d'un répertoire qui correspond à la partie droite de l'explorateur de fichiers, et le second est un ShellTreeView, qui représente l'arborescence des répertoires.

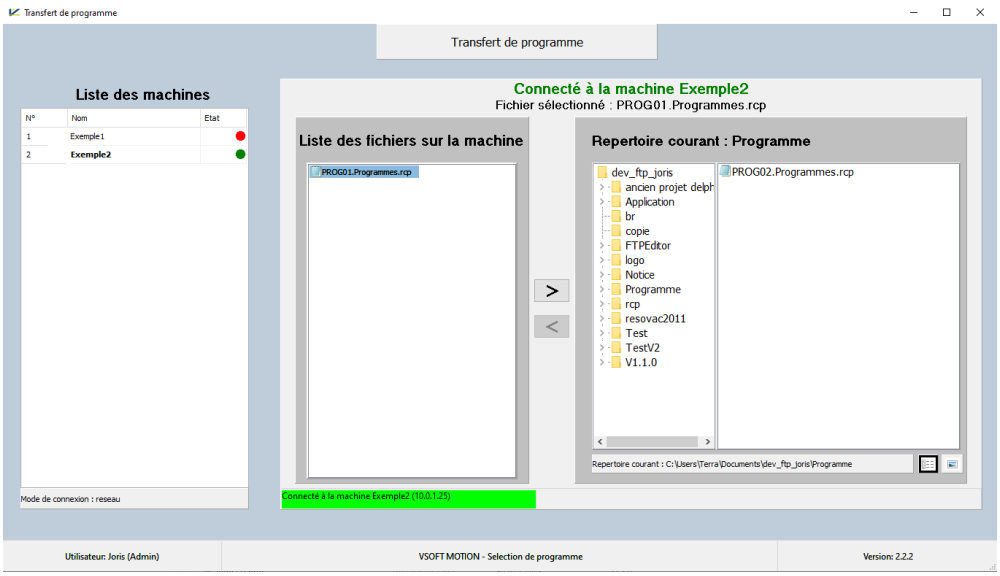

FIGURE 4.3 – Interface de transfert de programme

Pour se connecter à la machine, le logiciel utilise le composant IdFTP. Celui-ci permet grâce à une adresse IP, des identifiants de connexion et un port de se connecter à une machine via le protocole FTP ou SFTP. L'un des défauts du ShellListView est qu'il ne peut être utilisé que pour afficher les fichiers sur l'ordinateur. Donc, dans l'état actuel, il serait impossible de visualiser directement les fichiers présents sur la machine. Pour palier ce problème, à chaque connexion, les fichiers qui contiennent les programmes sont récupérés et stockés dans un dossier temporaire sur le PC. Le contenu de ce dossier temporaire sera ensuite affiché à l'utilisateur puis détruit à la fermeture du logiciel.

Cette méthode permet d'avoir une visualisation des fichiers similaire pour la machine et l'ordinateur, mais elle peut créer des problèmes de synchronisation. En effet, une fois connecté, si de nouveaux programmes ou des programmes déjà présents sur la machine sont ajoutés ou modifiés, le dossier temporaire ne sera plus à jour. C'est pourquoi, à chaque action de l'utilisateur, une vérification sera effectuée pour s'assurer que le contenu du répertoire temporaire est similaire à celui du répertoire de la machine. Pour ce faire, le nombre de fichiers, le nom des fichiers, puis leur contenu seront vérifiés en calculant un hash qui sera comparé entre le PC et la machine. En cas de problème de synchronisation, l'utilisateur sera informé et invité à se reconnecter pour

mettre à jour les informations.

Les problèmes de synchronisation ne se limitent pas aux différences entre la machine et le répertoire temporaire. Ils peuvent également survenir lorsque des changements sont effectués sur le PC via l'explorateur de fichiers Windows. Par exemple, la création d'un dossier doit être détectée afin de mettre à jour le ShellListView et le ShellTreeView. Pour cela, des listeners (écouteurs en français) sont utilisés. Lorsqu'un changement est effectué dans les répertoires de fichiers, ces écouteurs déclencheront la mise à jour des différents composants, garantissant ainsi que l'affichage reste synchronisé avec les modifications apportées.

Si tout s'est bien déroulé, l'utilisateur peut alors transférer des fichiers de l'ordinateur vers la machine et inversement en cliquant sur des flèches ou alors en glisser-déposer (drag and drop) entre les 2 composants.

4.2.2 Visualisation et édition de programme

La visualisation et l'édition de programme proposent 2 fonctionnalités. La possibilité de créer un nouveau programme ou alors d'ouvrir un programme déjà existant.

Dans le cas de la création d'un nouveau programme, nous allons fournir un fichier programme vierge que l'utilisateur va venir associer une machine préalablement enregistrée dans l'éditeur machine pour laquelle il souhaite créer ce programme. Cela aura pour effet d'associer les contraintes et spécificité de la machine dans l'editeur de programme.

Dans le cas de l'ouverture d'un fichier préalablement transféré sur le PC, les contraintes de la machine sont directement inscrites dans le fichier programme, nous allons proposer directement à l'utilisateur de choisir un fichier avec l'extension .Programmes.rcp enregistré sur son ordinateur. Cette extension a été créée par l'entreprise pour stocker les données des programmes en respectant une certaine syntaxe. Chaque ligne du fichier correspond à un paramètre spécifique du programme, et suit le format clé ;valeur. Voici un exemple de la structure du fichier :

```
...
ProgRecette.darbre;100
ProgRecette.attente_vit_pp;1
ProgRecette.vitppchg;1500
ProgRecette.multi_pos_actif;0
ProgRecette.typ_gest_aac;0
ProgRecette.opt_incr;0
ProgRecette.num_tran_incr;0
ProgRecette.valeur_incr;0
ProgRecette.vitfraiselente;100
...
```

Cette structure clé-valeur est représentée en Pascal par un TDictionary que l'on appelle aussi Map ou tableau associatif.

Pour manipuler au mieux les données, nous créons dans notre librairie un objet TProgramme qui aura pour objectif de renseigner les différentes informations de celui-ci.

```
TProgramme = class
public
    FName: string;
    FPath: string;
    FParameters: TDictionary<string, string>;
end;
```

Cet objet contient plusieurs attributs : le nom du fichier (FName), son chemin (FPath), et un dictionnaire des paramètres (FParameters). Pour remplir ce dictionnaire, le programme utilise une fonction développée préalablement dans la librairie de fonctions :

```
function LoadProgrammeConfig(const path: string): TProgramme;
```

Pour réaliser à bien sa mission, cette fonction utilise le principe de parsing. Cela consiste à lire le contenu d'un fichier ligne par ligne, à diviser chaque ligne en fonction d'un séparateur que l'on va spécifier, puis de stocker ces éléments dans une structure de donnée quelconque. Dans le cas présent, la fonction ouvre le fichier spécifié par le chemin et lit son contenu ligne par ligne. Chaque ligne est divisée en deux parties : la clé (nom du paramètre) et la valeur (valeur du paramètre), séparées par un point-virgule. Les paires clé-valeur sont ensuite stockées dans le dictionnaire FParameters de l'objet TProgramme. Une fois, tous les paramètres extraits et stockés, l'objet TProgramme est retourné pour être utilisé.

Ensuite, diverses méthodes permettent d'accéder à ce dictionnaire. Par exemple, la fonction `getValueOfProgramme` permet de récupérer la valeur d'un paramètre d'un TProgramme donné, et la fonction `setValueOfProgramme` permet de modifier la valeur d'un paramètre.

```
function getValueOfProgramme(programme: TProgramme; key: string): string;

function setValueOfProgramme(programme: TProgramme; key: string;
    value: string): boolean;
```

Ces fonctions permettent une manipulation aisée des paramètres du programme, facilitant ainsi leur visualisation et leur édition dans le logiciel.

Une fois l'abstraction des données effectuée, il nous a fallu concevoir l'interface de visualisation du programme. Pour ce faire, nous avons choisi de réaliser une interface similaire à celle déjà présente sur la machine afin de ne pas déstabiliser l'utilisateur qui a l'habitude de l'utiliser.

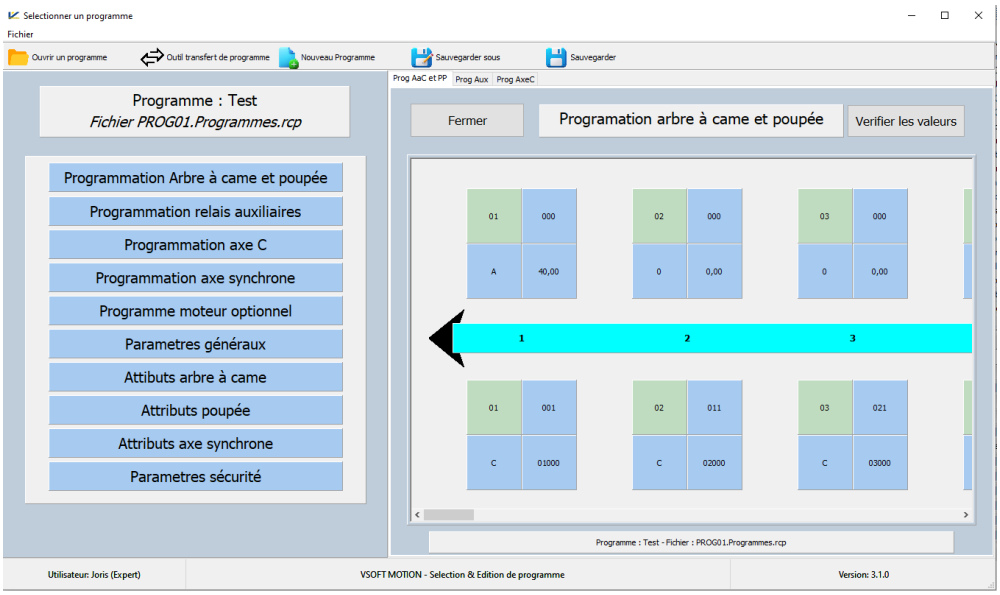

FIGURE 4.4 – Interface d'édition de programme

À chaque clic sur un des menus (situé à gauche sur la figure 4.4), le menu correspondant viendra s'insérer dans un composant TPageControl. Cet objet va nous permettre de gérer les différentes sections de l'interface de manière organisée et intuitive. Le TPageControl fonctionne comme un ensemble d'onglets, où chaque onglet va contenir les pages de menus qui ont été ouvertes de manière organisée et intuitive.

Afin d'améliorer l'ergonomie de l'éditeur de programme, il est possible de sortir les fiches du composant TPageControl pour les afficher sous forme de fenêtres flottantes. Cette fonctionnalité permet à l'utilisateur de personnaliser l'espace de travail selon ses préférences et de disposer les éléments de l'interface de manière optimale.

De plus, l'utilisateur peut ouvrir plusieurs éditeurs programmes en simultané afin de les comparer ou de copier les informations contenues dans les fiches. Par la suite, il pourra venir coller les informations préalablement copiées dans la fiche d'un autre programme.

Une fois le programme édité et les informations qu'il contient vérifier, l'utilisateur peut alors sauvegarder son programme ce qui viendra écraser l'ancien ou alors l'enregistrer sous dans le répertoire souhaité

4.3 Gestion des paramètres

4.3.1 Outil de connexion rapide

L'outil de connexion rapide est destiné aux administrateurs et experts. Il permet de se connecter aux machines sans avoir besoin de les enregistrer au préalable. Seules les informations essentielles à la connexion sont requises : le modèle de la machine, l'adresse IP, le port et les identifiants de connexion. Une fois ces informations saisies, l'interface de transfert de programme

s'affiche, mais cette fois avec un filtre permettant de visualiser, en plus des programmes, les fichiers de paramètres.

Il est possible d'importer ses fichiers pour les visualiser et de les exporter sous réserve de la vérification d'un mot de passe.

Ces fichiers sont des fichiers à manipuler avec précaution, car ils contiennent des informations que l'on peut considérer sensible pour plusieurs raisons. Premièrement, ces informations participent au bon fonctionnement de la machine et exporter un fichier d'une autre machine pourrait créer des incompatibilités dans la nouvelle. Secondement, ces fichiers contiennent les différentes options de la machine qui ont été au préalablement vendu par Vermot Automation. Si ces options monnayables peuvent être changer par tous, alors cela générera un manque à gagner pour l'entreprise.

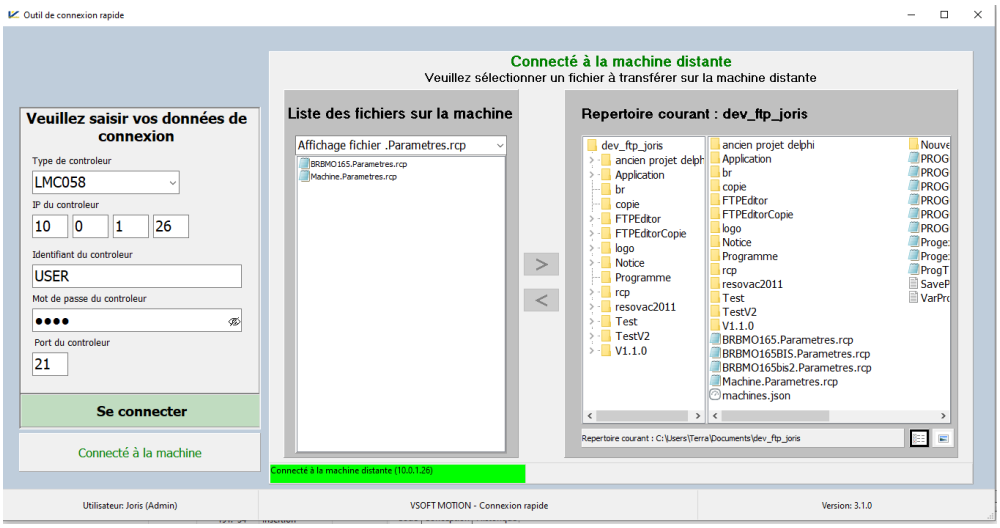

FIGURE 4.5 – Interface de connexion rapide

4.3.2 Visualisation de paramètres

L'idée est la même que pour la visualisation de programmes. L'objectif est de représenter les paramètres via une interface similaire au menu caché des machines qui contiennent ces paramètres.

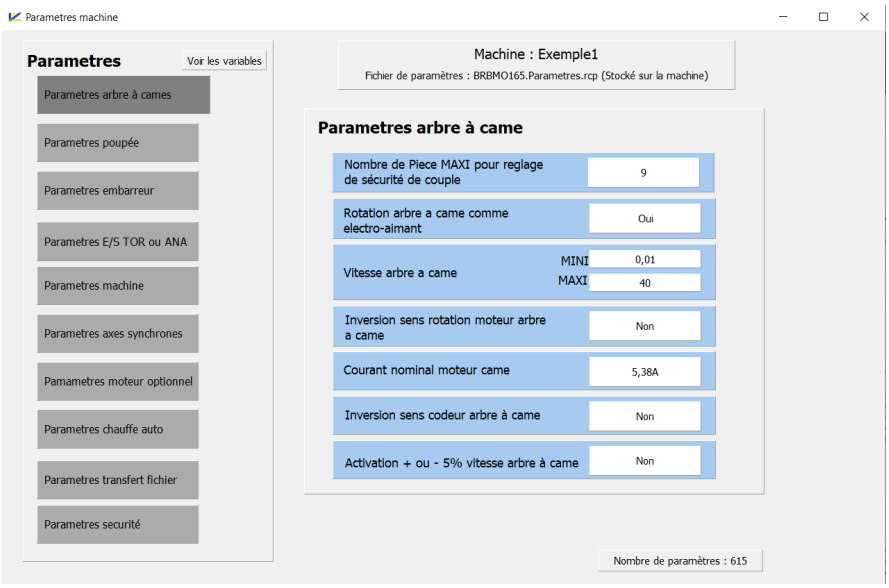

FIGURE 4.6 – Interface de visualisations des paramètres

Via cette interface nous retrouvons une majorité des paramètres machine mais pas tous ! De plus, le fichier paramètre peut être amené a évoluer dans le temps lors de nouvelle version des machines MOTION. De ce fait, un second affichage a été créé pour visualiser l'ensemble des paramètres dans un tableau sous la forme clé-valeur.

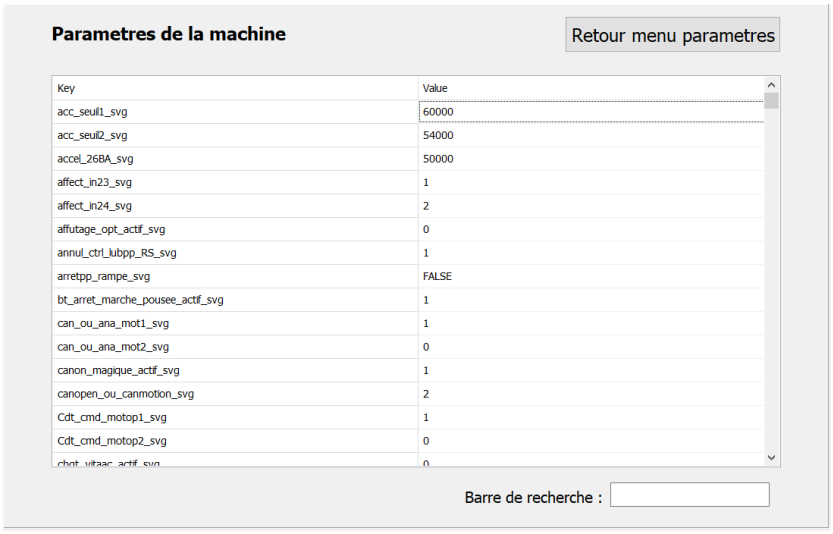

FIGURE 4.7 – Interface de la listes des paramètres machine

4.4 Outils annexes

4.4.1 Permissions et Droits

Comme précisé tout au long de ce rapport, les utilisateurs du logiciel peuvent être des décolleteurs, des administrateurs de parcs de machines, ou des membres de Vermot Automation. Les différentes catégories d'utilisateurs ne doivent alors pas avoir les mêmes permissions.

Nous avons pris la décision de hiérarchiser les utilisateurs en trois sections : les utilisateurs, les administrateurs et les experts. Les utilisateurs ont accès aux fonctions de base, les administrateurs disposent de droits supplémentaires pour gérer les machines et les paramètres, et les experts ont les permissions les plus étendues, leur permettant de modifier la configurations du logiciel lui-même.

Pour garantir l'évolution et l'adaptation du logiciel, les zones nécessitant des permissions sont configurables par les utilisateurs experts via une interface dédiée. Cela permet une flexibilité maximale et assure que le logiciel peut répondre aux besoins changeants des utilisateurs et des nouvelles exigences des clients.

4.4.2 Gestion des utilisateurs

Afin de se connecter au logiciel avec les bonnes permissions, une gestion des comptes utilisateurs a été mise en place. Ces comptes peuvent être créés via une interface réservée aux experts (ou aux administrateurs si les permissions ont été modifiées via l'interface précédente). Dans cette interface, il est possible de consulter la liste des utilisateurs, ainsi que d'ajouter, de modifier, de supprimer ou de réinitialiser le mot de passe d'un compte existant.

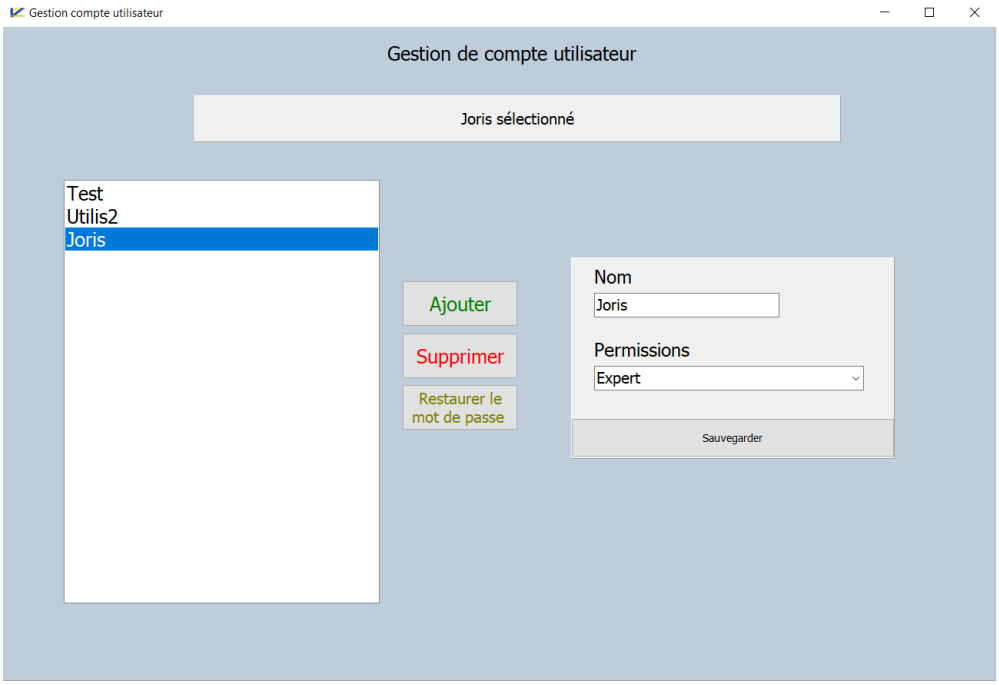

FIGURE 4.8 – Interface de gestion de compte

Les comptes sont ensuite enregistrés dans le fichier users.json, qui contient la liste des utilisateurs avec leur login, leur mot de passe hashé et leur niveau de permission : 0 pour utilisateur, 1 pour administrateur et 2 pour expert.

```
{
 "username": "Joris",
 "password": "532eaabd9574880dbf76b9b8cc00832c20a6ec113d682299550d7a6e0f345e25",
 "permission": 2,
}
```

Le fait de hasher des données consiste à calculer une empreinte unique d'une chaîne de caractères (dans ce cas précis, un mot de passe). Les fonctions de hachage ont diverses applications, notamment la possibilité de calculer l'empreinte unique d'un fichier pour comparer sa similarité à un autre ou de stocker les mots de passe de manière protégée dans une base de données. [1]

Chaque fois qu'un utilisateur souhaite se connecter au logiciel, une page de connexion lui demande son identifiant et son mot de passe. Si l'utilisateur ne s'est jamais connecté avec son compte, il lui sera demandé de spécifier un mot de passe pour les connexions futures. Cette gestion permet de garantir que chaque utilisateur accède uniquement aux fonctionnalités qui correspondent à son niveau de permission, assurant ainsi une utilisation sécurisée et appropriée du logiciel.

1. Plus de détail sur les fonctions de hachage : `https://fr.wikipedia.org/wiki/Fonction_de_hachage`

4.4.3 Gestion des répertoires et fichiers

La gestion des répertoires et fichiers dans le logiciel est utile pour organiser et accéder rapidement aux différents éléments nécessaires pour les opérations quotidiennes. Cette fonctionnalité permet de définir les chemins par défaut lors de la sélection d'un programme, par exemple.

De plus, via cette interface, on peut changer le fichier machines.json qui contient les configurations de machine. Cela sera utile à Vermot Automation, car permettra de gérer plusieurs fichiers machines.json, chacun correspondant au parc de machine de chacun de ses clients. En cas de dépannage, grâce au numéro de la machine, le logiciel fournira une interface claire toutes les spécificités de la machine impactée sans nécessité une intervention du client.

4.5 Gestion des licences

Lors du premier lancement du logiciel, la validation de la licence s'effectue grâce à une clé d'activation. Voici les étapes clés de ce processus :

Tout d'abord, le programme récupère le numéro du disque sur lequel est installée l'application. Ce numéro est unique et sert à générer les clés de licence. Pour garantir la sécurité et la reproductibilité des résultats, la graine [2] (aussi appelé seeds) du générateur de nombres aléatoires est définie avec ce numéro de disque.

Ensuite, un nombre aléatoire n1 est généré avec la fonction `Random` dans une plage allant jusqu'à 10,000,000,000. À partir de ce nombre n1, on calcule n2 comme étant la différence entre 10,000,000,000 et n1. Un autre calcul est effectué pour obtenir n3 en ajoutant n1 au numéro de disque. Ces deux valeurs, n2 et n3, sont alors communiquées à l'utilisateur.

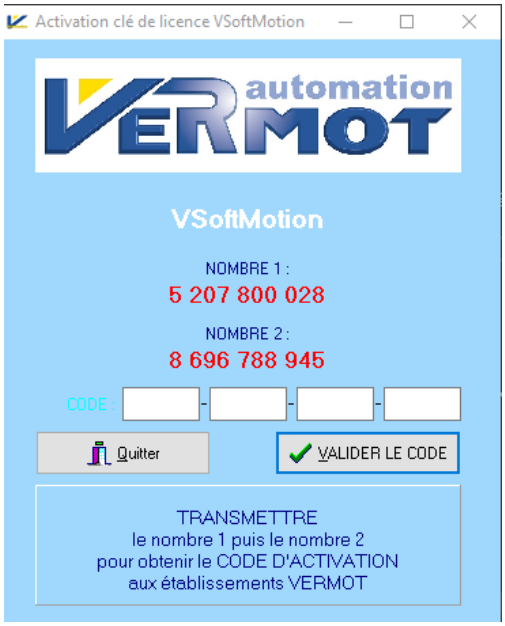

FIGURE 4.9 – Saisie de la clé d'activation

2. Plus d'informations sur les graines : `https://fr.wikipedia.org/wiki/Graine_al%C3%A9atoire`

L'utilisateur transmet ces deux nombres à l'entreprise pour obtenir une clé d'activation. L'entreprise utilise ces valeurs pour retrouver le nombre aléatoire initial n1 en calculant n1 = 10,000,000,000 - n2 et pour retrouver le numéro de disque en calculant le numéro de disque = n3 - n1.

À partir de ces deux valeurs, l'entreprise crée une clé d'activation. Pour cela, les numéros de disque et le numéro aléatoire sont convertis en une suite de 16 caractères hexadécimaux grâce à un certain processus.

Cette clé est ensuite transmise à l'utilisateur qui la saisit dans l'interface prévue à cet effet.

En parallèle, l'ordinateur sur lequel est exécuté le VSoftMotion calcule la clé d'activation théorique et la compare à celle reçue de l'utilisateur. Si les clés correspondent, l'utilisateur est autorisé à utiliser le logiciel, et la clé d'activation est enregistrée dans le fichier `licence.txt` pour faciliter les utilisations ultérieures. Dans le cas de la figure 4.9, la clé d'activation générée sera 1744-9F2D-301A-0161.

Lors des prochaines utilisations, le logiciel vérifie si la clé d'activation stockée dans `licence.txt` correspond à celle attendue. Si les clés diffèrent, cela indique que le logiciel a été déplacé sur un autre disque dur ou que le fichier `licence.txt` a été modifié manuellement. Dans ce cas, l'utilisation du logiciel est bloquée, et l'utilisateur doit demander une nouvelle activation pour réutiliser le logiciel. Ce processus permet de protéger le logiciel et assurer que celui-ci a été acheté auprès de Vermot Automation.

Il est important de noter que la sécurité de ce système repose en grande partie sur la confidentialité des calculs effectués et des données utilisées pour générer la clé d'activation. Cette approche a été choisie, car le logiciel peut fonctionner sans connexion Internet, rendant impossible la vérification en temps réel de la validité de la clé par une connexion à une base de données distante.

En termes d'évolution, il serait envisageable d'intégrer des fonctionnalités supplémentaires dans la clé d'activation, comme une date limite d'utilisation du logiciel et des options supplémentaire dans le logiciel. Cela permettrait de renforcer encore la sécurité et le contrôle sur les licences et de proposer des périodes d'essais gratuites.

5 Synthèse et Perspectives

5.1 Vente de l'application

La vente de l'application s'adresse principalement aux entreprises possédant la gamme de machines MOTION, avec un potentiel de 30 à 50 clients intéressés. Ces entreprises cherchent à optimiser leur production et à moderniser leurs processus, et le logiciel VSoftMotion est conçu pour répondre précisément à ces besoins. En centralisant la gestion des programmes et des paramètres machines depuis un ordinateur, l'application offre un gain de temps considérable pour les opérateurs et les administrateurs.

Il est important de noter que Vermot Automation génère la majorité de ses revenus non pas à travers les logiciels, mais via la vente de produits MOTION et les services associés. En proposant ce logiciel, l'objectif premier n'est pas de générer des revenus, mais de renforcer la position de l'entreprise sur le marché et d'offrir une solution complète aux clients, afin de se démarquer de la concurrence.

Pour promouvoir le logiciel, il est prévu de le présenter lors de salons professionnels et sur les réseaux sociaux de l'entreprise. Cette stratégie a pour objectif de démontrer concrètement les avantages et les fonctionnalités de l'application. En outre, il est aussi envisagé de proposer une phase de test gratuite d'une certaine période pour permettre aux entreprises de découvrir et d'apprécier les bénéfices du logiciel avant de s'engager dans un achat. Dans la même idée, il est aussi envisagé de proposer des réductions sur le prix du logiciel en cas d'achat de nouveaux coffrets MOTION.

Le prix de vente de l'application reste encore à définir, mais l'objectif est de fournir un outil relativement abordable et efficace qui complète l'offre de produits MOTION et renforce la fidélité de la clientèle.

5.2 Possibilité d'évolution

Une des premières évolutions envisagées et qui à commencé à être développé est l'intégration d'une fonctionnalité permettant d'imprimer un programme. Cette feuille contiendrait les differentes information sur le programme, permettant de le communiquer.....

Une autre idée est de propose la possibilité de scanner des feuilles de programme qui sont crée par... L'objectif de ceci serait de generer un fichier avec les contraintes imposé par l'extension .Programmes.rcp afin de virtualiser ces feuilles de programme. Pour ce faire l'idée est d'utilise un API de Reconnaissance optique qu'on appelle aussi OCR. L'OCR (Optical Character Recognition) est une technologie qui permet de reconnaître et de convertir les caractères imprimés ou manuscrits en texte électronique. Cette technologie utilise des algorithmes de reconnaissance de formes pour analyser les images de documents et extraire les caractères, les mots et les phrases et , a patir de cela récuperer les informations interessante pour virtualiser notre programme.

La gestion des traductions est une autre évolution interessante En intégrant un système de traduction multilingue, le logiciel pourrait être utilisé par des opérateurs parlant différentes

langues, rendant l'application plus accessible à un public international et augmentant ainsi le marché potentiel.

Enfin, l'idée de la traçabilité des pièces à aussi été abordé lors de nos discussion avec Monsieur Pagnier, notamment pour l'industrie pharmaceutique. En ajoutant des outils de traçabilité. Le logiciel pourrait enregistrer et suivre chaque étape du processus de production, incluant la machine qui a créé la pièce, le programme utilisé et l'auteur du programme. Cette fonctionnalité fournirait des rapports détaillés et des historiques de production, permettant aux entreprises (En particulier du milieu pharmaceutique) de garantir la qualité et la conformité de leurs produits.

Rajouter la transmisions de connaissance du dev Autonome

5.3 Retour sur expérience

Travailler sur ce projet a été une expérience extrêmement enrichissante et formatrice. L'un des principaux défis était d'utiliser Delphi, un environnement de développement que je n'avais jamais utilisé auparavant, et de coder en Pascal, un langage que je ne connaissais pas. J'ai pu mettre en avant mes compétences personnelles ainsi que celles acquises en programmation objet avancé au semestre 4 pour la réalisation de ce projet.

Un aspect notable de ce projet a été la gestion des fichiers de configuration, un domaine que nous avions abordé plus superficiellement en cours. Concevoir une solution qui permet de gérer les utilisateurs, les paramètres et plusieurs fichiers de configuration machine pour différents clients avec la possibilité d'être évolutif a été particulièrement intéressant.

La mise en place concrète de la gestion de projet a également été elle aussi une facette très intéressante de cette expérience. Comprendre les besoins des utilisateurs et les traduire en fonctionnalités logicielles se sont avérées plus complexes que prévu. J'ai réalisé qu'il était souvent difficile de cerner précisément les attentes de mon maître de stage, car les besoins ne sont pas toujours clairement exprimés et peuvent évoluer au fil du temps. Cela a exigé une communication plus régulière et plus pointilleuse en dessinant le résultat souhaité en amont pour s'assurer que nous étions sur la même longueur d'onde.

J'ai également constaté que les utilisateurs peuvent avoir des niveaux de compréhension technique très variables, et qu'il est essentiel de simplifier les explications et de les guider étape par étape. Cette réalisation m'a aidé à adapter ma manière de structurer les informations, utiliser des mots simples et à être plus pédagogique.

6 Conclusion

Pour conclure, ce stage a été bien plus qu'une simple expérience technique. C'était ma première incursion dans le monde professionnel de l'informatique, une transition stimulante de l'environnement académique à celui de l'entreprise. Travailler sous la direction d'un patron m'a offert un aperçu précieux de ce que signifie être un membre productif et valorisé d'une équipe. Savoir que mon travail avait un impact concret et positif sur le fonctionnement de l'entreprise a ajouté une dimension de gratification personnelle à cette expérience professionnelle.

C'était aussi l'occasion de voir comment mes compétences pouvaient être mises en œuvre pour répondre aux besoins réels. Chaque fonctionnalité développée contribuaient directement à la réalisation des objectifs de l'entreprise. Cette prise de conscience a renforcé ma motivation et mon engagement envers le projet, car je savais que mes efforts étaient appréciés.

Travailler avec Delphi, apprendre le langage Pascal ainsi que m'immerger dans un domaine technique auquel je ne connaissais rien ont constitué des défis stimulants. La gestion des fichiers de configuration et la conception d'une solution flexible pour répondre aux besoins ont été des aspects particulièrement intéressants de ce projet. Cette expérience m'a également confronté à la réalité de la communication pratique, où comprendre précisément les attentes du client s'avère souvent complexe.

Bien que le résultat final puisse encore évoluer et être amélioré, je suis satisfait de ce que j'ai accompli et ai pu répondre à une grande majorité des demandes de l'entreprise. Ce stage m'a permis de développer de nouvelles compétences et d'acquérir une précieuse expérience dans le domaine du développement logiciel.

Résumé

La troisième année de Licence informatique à l'Université de Franche-Comté propose, pour son sixième semestre, un stage d'une durée minimale de 10 semaines. J'ai effectué ce stage chez Vermot Automation, une entreprise spécialisée dans la modernisation d'anciennes décolleteuses à cames située à Valdahon. Mon rôle au sein du bureau d'étude de l'entreprise a été de concevoir, à l'aide de Delphi, un logiciel multifonctionnel nommé VSoftMotion permettant de gérer de puis un logiciel, un parc de machines dans l'objectif de pouvoir visualiser à distance leurs paramètres, et de créer ou modifier des programmes de décolletage pour les machines précédemment enregistrées.

Mots clés

Développement logiciel, Interface utilisateur, Décolleteuse à cames, Delphi, Pascal object, MOTION , fichier de configuration, Stage, Licence informatique, Vermot Automation

Abstract

The third year of the Computer Science Bachelor's program at the University of Franche-Comté includes, during the sixth semester, a mandatory internship lasting a minimum of 10 weeks. I completed this internship at Vermot Automation, a company located in Valdahon that specializes in modernizing old cam-operated lathes. My role within the company's design office was to develop, using Delphi, a multifunctional software named VSoftMotion. This software allows for managing a fleet of machines from a computer to remotely visualize their parameters and create or modify machining programs for the previously registered machines

Keywords

Software development, User interface, Cam-operated lathe, Delphi, Object Pascal, MOTION, Configuration file, Permissions, Internship, Computer Science Bachelor's, Vermot Automation