RATP DEV

le cnam

Valère GAY-HEUZEY

Licence générale Sciences Technologie Santé mention Informatique Parcours Informatique Générale

LG025

Alternance 2020 - 2021

Rapport d'activité

à la suite ☐ d'une activité professionnelle
☐ d'un stage *(cocher la case adéquate)*

LICENCE STIC Mention Informatique Générale

MENTION : CHO

CENTRE : CNAM CHOLET

DATE DE DEPOT AU CNAM :
(à remplir par le CNAM)

VOTRE IDENTITE

M, Mme, Mlle : Mr Gay-Heuzey
(nom de jeune fille pour les femmes mariées)

Nom d'usage ou marital :

Prénoms : Valère

Date et lieu de naissance : RENNES (35) le 21 mars 1998

Nationalité : Française

Adresse : 75 Rue Maine

Code postal : 49100 Ville : Angers

Tél. professionnel :
(facultatif)

Téléphone personnel : 06.27.82.14.90

Adresse e-mail

personnelle : valere.gh@hotmail.fr

professionnelle : Valere.GAYHEUZEY@ratpdev.com
(facultative)

N° de la carte d'auditeur du CNAM : 489152

VOTRE FORMATION

1. VOTRE DIPLOME OU TITRE OBTENU, LE PLUS ELEVE

Reportez l'intitulé exact en toutes lettres

Licence Professionnel Système Embarquée et Solution par Application Mobile (SESAM)

Date de l'obtention : 2019-2020
Diplôme **OUI**
Titre homologué **OUI**
X D'établissement public
❑ D'établissement privé
Niveau du diplôme : **VI**

VOTRE EXPERIENCE PROFESSIONNELLE

- Durée totale de votre expérience professionnelle :
- Durée de votre expérience en informatique :

- Succession des postes (emplois) occupés, du plus ancien au plus récent. :

Précisez le nom de l'entreprise, le poste occupé (celui inscrit sur la fiche de paye, niveau de qualification), la fonction tenue, la durée de l'activité.

Emploi occupé	Entreprise et secteur d'activité	Durée de l'activité	Fonctions occupées
Technicien Alarme	GBS Alarme	2016-2018	Apprentie Technicien d'alarme, chargée de la maintenance et installation
Développeur C++	IRIGO	Depuis le 10/11/19	Développeur avec la charge d'un projet de télémaintenance

Sommaire

Introduction

Je m'appelle Valère Gay-Heuzey, apprenti en licence STS (Science Technologie et Santé) au sein du CNAM Cholet

Depuis mon baccalauréat, je suis une formation en alternance afin de multiplier les expériences de travail. J'ai ainsi réalisé 22 semaines de stages en bac professionnel, 2 ans d'apprentissage en BTS et une année en Licence professionnel (électronique et informatique embarquée). J'ai souhaité faire une seconde licence, générale cette fois-ci, afin de perfectionner mes compétences en informatique pur.

Je suis actuellement en contrat avec la société RATP Dev Angers, filiale de RATP Group, en qualité de développeur. Il s'agit de l'entreprise qui m'a accompagné l'année dernière en licence pro. Ce contrat dure 1 an et s'arrête le 28 aout 2021.

Du fait de sa taille et de son positionnement sur le marché français et international, j'ai eu l'occasion de découvrir la complexité très intéressante des rouages de cette entreprise.

De plus j'ai eu à travailler sur un projet riche, abordant de multiples aspects et domaines technique.

Figure 2 : tramway angevin

Présentation de l'entreprise

Description

RATP Dev est une filiale de RATP Group : créée en 2002, elle a pour objectif de développer, exploiter et maintenir de nouveaux réseaux en France, en Europe et dans le monde, en s'appuyant et en déployant l'expertise du groupe RATP. Son rôle consiste à assurer le fonctionnement des services de transport de personnes, et la maintenance du réseau et du matériel. Pour cela, RATP Dev innove afin de proposer des solutions de transports adaptées au contexte urbain dans lequel elles s'inscrivent.

- Nombre de salariés : 20 000 salariés
- Capital Social : 369M€
- Chiffres d'affaires : 1.3Md€

RATP Dev est lié par un contrat de délégation publique à la métropole de Angers Maine et Loire pour une durée de 6 ans (jusqu'en juin 2025). Ce type de contrat implique que la collectivité délègue et confie à une entreprise, la responsabilité et la gestion d'un service public, en l'occurrence le transport public des biens et des personnes.

Dans cette configuration, l'ensemble des services, fournitures et matériels appartiennent à la collectivité. Cela comprend les bus, tramways et taxis utilisés pour le transport de personnes, mais aussi la location des locaux ainsi que les fournitures et pièces de rechanges nécessaires à la maintenance.

Tous les bénéfices reviennent à la métropole qui paie à la société titulaire une redevance de 39M€ par an, pour la gestion du réseau.

<u>Il existe ainsi 3 entités :</u>

- La métropole (collectivité) Angers Maine-et-Loire : le Déléguant
- L'entité commerciale à travers laquelle la métropole vend des services : Irigo
- La société titulaire du contrat : le Délégataire (ou opérateur)

Figure 3 : schéma des relations contractuelles

La société RATP Dev Angers prend en charge un « bassin de vie » de près de 400 000 habitants au total dont 290 000 sont situés sur la ville de Angers. Le réseau de la métropole angevine couvre près 51 000 hectares et 29 communes.

Le précédent opérateur de transports urbains de la métropole d'Angers Maine-et-Loire était la société KEOLIS pendant près de 30 ans. Néanmoins, lors de l'appel d'offres, la société RATP Dev a été retenu, en fessant une offre moins chère que celle de KEOLIS (45M€ pour ces derniers contre 39M€ pour RATP Dev).

L'entreprise RATP Dev se décompose en 7 directions : exploitation, marketing, ressource humaine, qualité service environnement, administration et financière, projet tramway et technique. C'est dans ce dernier pôle que je travaille, dirigé par Mr Karim BENYOUCEF.

Mission Principale : Interface de télémaintenance

Contexte

Un incendie est survenu il y a 2 ans, au sein d'un tramway de la métropole d'Angers, à la suite d'une panne du système d'alimentation de secours : les batteries de secours qui alimentent le tramway, ont pris feu lors et ont brûlé le toit en provoquant uniquement des dégâts matériels.

L'entreprise a décidé de remédier à ce grave dysfonctionnement en analysant les données en provenance des batteries de secours, ils souhaitent ainsi réaliser une analyse sur de longues périodes afin de déterminer une cause à l'incendie.

Je travaille pour RATP Dev depuis maintenant 1 an et demi. Ma mission l'année dernière, consistait à construire une communication entre les tramways et une base de données préexistante. Cette année, afin de rendre accessible ces informations, il m'a été demandé de concevoir une interface homme-machine (IHM) permettant un rendu graphique du contenu de la base de données.

Enjeux

J'ai travaillé sur la base d'un cahier des charges qui a évolué au fil du temps, selon l'avancée des travaux, les ressources disponibles et la volonté de mes supérieurs.

- La première année, le projet a d'abord été pensé pour permettre une analyse des causes de l'incendie avec un archivage massif des informations provenant des batteries, qui a été réorienté par mes supérieurs vers un système de télémaintenance.
- La seconde année, l'interface, actuellement terminée, sera utilisée pour du dépannage et non pour de l'analyse.

En effet, lors d'un incident dans le tramway, un défaut du système ou encore une défaillance humaine, il a été jugé pertinent d'observer plusieurs informations, directement issues du tramway telles que la vitesse ou des tensions électriques. Un opérateur pourra prendre contact par radio avec un conducteur et lui communiquer des informations essentielles sur l'état de son tramway, et ceci en temps-réel.

Cette réorientation de mon projet provient également d'une nouvelle norme sécurité. Le constructeur ALSTOM nous fournit les tramways. Ils imposent désormais un changement tous les 5 ans des batteries. En effet, bien que la cause de l'incendie ne soit toujours pas connue, la durée de vie des batteries est suspectée. Cette nouvelle norme devrait ainsi limiter la survenance de nouveaux incendies.

Description du projet

Le projet qui consiste à créer une solution d'affichage en temps réel, se scinde en 2 parties : la partie fonctionnelle et graphique. J'ai débuté avec cette dernière, plus longue à réaliser. J'ai utilisé le même outil de programmation que l'année dernière : Qt Creator. Il s'agit d'un *frameworks* informatique, fournissant des *bibliothèques*[1] de programmation riche. Il me permet d'être rapide et polyvalent à la création de code. Les langages utilisés sont le C++ (partie fonctionnelle) et le JavaScript (partie graphique) abrégé JS.

Analyse des besoins

J'ai établi le cahier des charges ci-dessous à partir des commentaires provenant des opérateurs qui utiliseront ce système d'interface :

- fournir une interface en temps-réel (quelques secondes de délais) ;
- mettre à disposition des opérateurs l'ensemble des 75 variables actuellement envoyées dans la base de données par chaque tramway ;
- organiser l'interface de manière intuitive afin qu'elles puissent être comprises par tous ;
- la possibilité d'afficher les informations provenant d'un tramway en particulier, choisi parmi la liste de tous ceux dont l'entreprise dispose (17 tramways à Angers) ;
- créer un rendu esthétique de l'interface (animation d'une jauge de vitesse, dégradé de couleurs, etc…) ;
- des contraintes graphiques demandées par les des techniciens, notamment la reprise des mêmes mécaniques d'affichage que celles présentes sur les tableaux de bord des tramways. Cela comprend la mise en place de clignotement sur des voyants précis, et des animations GIF ;
- des systèmes de vérifications et de sécurité de la base de données qui servent à alerter si la connection à été perdue, si le tramway n'envoie pas d'informations ou que la base est vide ;
- enfin, pour permettre au techniciens de visualiser des informations déjà passées, un mécanisme d'horodatage, permettant de sélectionner une date et heure précise.

Contraintes

J'ai conçu et déterminé les étapes nécessaires à l'aboutissement du projet. L'accès au tramway était soumis à des contraintes de disponibilité (atelier, maintenance). J'ai réalisé ce projet en autonomie avec l'appui de mon tuteur.

[1] *Frameworks : logiciel de programmation permettant de faciliter le travail d'un programmeur. Cet outil va mettre en avant des éléments de code et en permettre une lecture rapide et claire. Bibliothèque : ensemble de code déjà écrit, permettant de réaliser des mises en œuvre et implémentation complexe de manière efficace, ex : Mise en place de communication avec base de données.*

Réalisation du projet

Partie graphique

Le logiciel Qt Creator apporte un moteur graphique[2], nommé QML (Qt Modulation Language) qui permet de réaliser des interfaces et applications complexes, portable sur mobile et ordinateur. Il s'utilise grâce au langage JavaScript avec des spécificités propres à Qt.

Mon premier objectif pour l'interface, a été de m'inspirer du tableau de bord des tramways. A partir de cela j'ai organisé mon interface en plusieurs zones distinctes :

- les alarmes dit « IOS », qui rassemblent les différents états critiques du tramway (perte d'alimentation électrique, perte de communication etc.…) ;
- les informations secondaires (défaut mineur, quelle cabine est occupée) ;
- les modes dans lesquels se trouve le tramway ;
- les tensions électriques générales des batteries.

Cette organisation permet également de bien séparer le code car chacune des zones correspond à un fichier séparé des autres : chacun de ses fichiers contient des « éléments » qui lui est propre. Un « élément » ou « item » est un morceau de code qui correspond à un élément graphique. Par exemple l'item « voyant », écrit dans le code, fait apparaitre un cercle noir sur l'interface graphique.

Chaque item est contenu dans un « module », catalogue d'items, et possède un identifiant unique utilisé pour modifier les caractéristiques de ce dernier. S'il existe des catalogues d'items permettant de réaliser beaucoup d'applications, il est parfois nécessaire de créer un item de toute pièce (implémentation). Le terme implémentation est un terme couramment utilisé en informatique et désigne la création d'un morceau de code, qui répond à un besoin spécifique du développeur.

J'ai notamment dû créer un item horloge, permettant l'affichage d'une horloge dans l'interface. Elle est construite à partir d'items basique : d'un cercle, qui représente le cadre, de 3 barres représentant les aiguilles ainsi que de divers autres items, améliorant le rendu graphique. Construire ses propres items demande un temps conséquent pour un développeur, mais ils sont réutilisables et peuvent s'intégrer ensuite aux items classiques fournis dès le départ. De plus, il est également possible de construire des items sur la base de ceux que l'on a déjà créés. Il convient alors de créer soi-même des modules, ce qui est un coût de temps important.

Conformément au cahier des charges, l'interface doit être « réactive », c'est-à-dire qu'elle réalise des animations, des clignotements, fondu de couleurs, etc...

Le langage JavaScript est un langage spécifiquement employé pour être réactif. Cependant, ce n'est pas lui qui ordonne les animations. Il s'agit de la partie C++ (partie fonctionnelle) qui détermine et indique au JS quelles actions faire.

[2] *Moteur graphique : outil qui sert à encapsuler l'API (c'est-à-dire la partie fonctionnelle du code) sur laquelle il repose, fournir des fonctions d'affichage plus évoluées, gérer automatiquement la "scène" de manière que le rendu soit optimisé et que l'utilisateur soit déchargé de cette gestion.*

L'une des composantes les plus difficiles de ce projet a été de faire communiquer la partie fonctionnelle avec la partie graphique soit faire communiquer la partie C++ avec le JavaScript (on parle ici de *binding* entre le C++ avec le JS, terme anglais signifiant « attacher ».)

Le code placé en C++ est différent du code JS. Ces codes respectifs sont placés à des endroits différents et n'échangent pas entre eux par défaut. Le Framework Qt Creator offre 2 solutions pour réaliser cette communication : la communication dynamique, et la communication statique.

Dans les 2 cas il s'agit de faire la même chose : dire du C++ au JavaScript, quoi faire et à qui.

Les modifications sont faites sur les items. Par exemple dire au voyant n°4 de s'allumer. Pour appeler ce voyant on utilise son identifiant unique. C'est le même principe pour tous les autres items.

Ce que l'on indique à l'item, va dépendre de sa nature : en fonction de celle-ci, il lui est associé une ou des propriétés. Un voyant a une propriété booléenne « active » (qui peut être égale à 0 ou 1): si l'on souhaite l'allumer, on passe sa valeur de 0 à 1, ce qui aura pour effet de passer la couleur du voyant de noir, à rouge.

Si l'on souhaite modifier un texte d'affichage, par exemple une valeur de tension, on fait appel à sa propriété texte. Il en va ainsi pour chaque item.

Il existe pour chaque item des propriétés communes, telles que son opacité ou sa taille.

Quand l'action à réaliser est identifiée ainsi que l'item considéré, il faut choisir la manière de communiquer. La communication dynamique est la plus simple à mettre en place. Il faut pour cela recréer un équivalent de l'item, dans le code C++. On lui indique son nom, et dans quel fichier JS aller le chercher. Une fois cette mise en place faite, on peut modifier ses propriétés.

Cette méthode est la plus simple car l'on déclare simplement ces 2 composantes. Cependant elle présente un désavantage car chaque fois que l'on souhaite modifier un item, il faut repréciser son nom, ainsi que le fichier JS.

La communication statique est plus lourde en matière de code. Il ne s'agit plus simplement de recréer l'item JS dans le code C++, mais un seul et même item qui sera commun aux 2 langages. Pour cela il faut définir un module, dédié uniquement à ce programme.

Une fois cette création de module effectué, il n'y a pas de « reprécisions » à apporter sur l'item. Son utilisation est alors simplifiée.

Le très grand avantage de la communication statique réside dans le fait que les items peuvent être modifiés depuis le C++ ou le JS, ce qui est impossible avec la 1ère solution. En effet, la communication dynamique ne fonctionne que du C++ vers le JS, ce qui peut poser problème.

Il existe toutefois des manières pour contourner se problèmes, mais elles deviennent moins pratiques à l'utilisation.

Afin de me former, j'ai mis en place les 2 méthodes.

```
164     QPair <QObject*, int> m_choixMode;
```

```
85      if (m_choixMode.first){
86
87          m_choixMode.first = rootItem->findChild<QObject*>("choixMode");
88
89          switch(m_choixMode.second){
90          case MODECHOIX_ETEINT:
91              m_choixMode.first->setProperty("active", false);
92              break;
93          case MODECHOIX_CLIGNOTEMENT:
94              m_choixMode.first->setProperty("active", !m_choixModeBlink);
95              m_choixModeBlink = !m_choixModeBlink;
96              break;
97          default:
98              break;
99          }
100     }
101 }
```

Figure 6 : Exemple de code permettant le clignotement du voyant « Choix de Mode »

Le code ci-dessus présente le code de clignotement d'un voyant « Choix de Mode ». Le clignotement n'est pas un comportement possible par défaut. Il faut passer la propriété « active » de ce voyant de 0 à 1 régulièrement, ce qui crée mécaniquement un clignotement.

La technique de communication employée est statique. La ligne 166 (en haut de l'image), montre la création d'une variable C++. Celle-ci s'appelle m_choixMode.

m_choixMode contient à la fois l'item C++ (*QObject**) et une autre valeur. Pour utiliser l'item je ferai : m_choixMode.first.

La ligne 87 associe l'item C++ à l'item JS (je *bind* les 2 items ensembles). Je demande à l'item C++ d'aller chercher dans le fichier JS « rootItem », l'item choixMode.

Les lignes 91 et 94 affectent une valeur booléenne à l'item C++.

Dans le premier cas il s'agit d'un *False* ce qui correspond à un 0 informatique, le voyant est éteint. Cette ligne s'effectue si l'on souhaite que le voyant soit simplement éteint, pas de clignotement.

Dans le deuxième cas, on souhaite que le voyant clignote. On affecte à l'item C++, la valeur contenue dans « m_choixModeBlink ». Cette valeur passe de 0 à 1 régulièrement, par l'intermédiaire d'un timer (un minuteur) que j'ai programmé.

Partie fonctionnelle

La partie fonctionnelle, bien que plus rapide à réaliser, n'en demeure pas moins tout aussi complexe si ce n'est plus, que la partie graphique. Celle-ci est réalisée en C++.

Le rôle de cette partie est d'interroger régulièrement la base de données qui enregistre les valeurs issues du tramway. C'est elle également qui coordonne l'affichage de l'IHM, qui alerte l'opérateur en cas d'erreur, qui contrôle les différent timer, etc....

Au lancement de l'application, le programme va tout d'abord générer l'interface. Il va ensuite tenter de se connecter à la base de donnée, si cela n'est pas possible il va afficher un message d'erreur afin de prévenir l'opérateur.

Les communications avec 2 machines, utilise le schéma de serveur-client. La base de données est un serveur. Son rôle est d'enregistrer et de fournir des informations à d'autres machines, que l'on appel client. Ces clients interrogent le serveur pour obtenir des informations.

Les échanges qui opérent entre eux, utilisent un protocle de communication. Il s'agit d'un esemble de règles, que les 2 machines respectent, afin de se comprendre et s'envoyer des informations. Il en existe de tous types et pour tous les usages. Le protocole SQL (Structured Query Langage) et le plus couramment utilisé pour ce genre d'échanges.

Pour établir la communication avec le serveur, il me faut créer un client. Dans le code, j'indique à quel adresse IP se trouve le serveur. je lui renseigne également un identifiant et un mot de passe. Ainsi, le serveur sait quel client est connecté, et m'autorisera ou non, l'accès à ses informations.

Une fois la connection établie, je lance un timer que j'ai créé. Ce timer est équivalent à l'horloge de mon programme. Lorsque celui-ci atteint 600ms, il repasse à 0 et redémarre. On dit qu'il *timeout*. A chaque timeout, le programme lance une série de 75 échanges avec la base de données, soit une fois pour chaque variable. Ces échanges sont extrêment courts, et ne durent que quelques millisecondes chacuns. Leur temps cumulé est inférieur à 600ms.

A l'issue de chaque échange, le programme récupère 4 informations : le nom de la variable, sa valeur, le numéro tramway duquel il est issu, et son heure d'enregistrement dans la base de données. Dès qu'une variable est obtenue, le programme associe son nom à un item C++ qu'il associe immédiatement à un item JS. Cette asso

Une fois cette association faite, le programme utilise la valeur obtenu de la BDD, pour la placer dans l'item C++, qui l'affecte à l'item JS.

Ainsi, toute les 600ms, 75 nouvelles valeurs sont obtenues, et sont retransmises visuellement sur l'interface.

Figure 8 : Exemple de cas d'erreur

Une fois la connections établit, le programme va démarrer un timer de 600ms. A chaque « interruption » du timer (c'est-à-dire toute les 600ms) le programme va venir interroger la BDD afin de recueillir les dernières informations.

Afin de procéder à cette interrogation, il est nécessaire d'utiliser un *protocole de communication* commun entre le programme et la BDD. Il s'agit du protocole SQL, protocole très largement utilisé dans l'industrie, spécialement dédié à ce genre d'application. Il fonctionne sur le principe dit de « requête SQL », il s'agit d'un message informatique envoyée vers la BDD, contenant les différents éléments d'informations voulu. Cependant n'ayant que peu d'expérience avec ce protocole, un collègue du service informatique m'a assisté afin de créer les requêtes nécessaires.

```
m_request = "SELECT ROW_NUMBER() OVER(ORDER BY v.NOM_VARIABLE ASC),"
            " v.NOM_VARIABLE, v.VALEUR, v.DATE_J, v.NUM_TRAM FROM"
            " W_TRAM_AUTOMATE v, (SELECT NOM_VARIABLE, max(DATE_J) "
            "AS MAX_DATE FROM W_TRAM_AUTOMATE WHERE DATE_J <= '"
          + sl_getDatePeriod() + "' GROUP BY NOM_VARIABLE ) VMAX WHERE"
                                 " v.NOM_VARIABLE= VMAX.NOM_VARIABLE AND "
                                " v.DATE_J=VMAX.MAX_DATE AND NUM_TRAM = 10"
           + m_numeroRame + " ORDER BY NOM_VARIABLE ASC;" ;
```

Figure 9 : Requête permettant l'obtention des 75 variables présents sur l'IHM

BDD : Base de données

Protocole de communication : Ensemble de règle définit et commune à toute les machines, permettant la communication entre différent systèmes.

La figure 9 présente la requête complète permettant l'obtention des 75 variables nécessaire à l'IHM.

Il s'agit d'une requête de sélection (« SELECT »), qui permet de sélectionner les informations que l'on souhaite. La commande « ORDER BY v.NOM_VARIABLE » permet de récupérer les valeurs en fonction de leurs noms et dans l'ordre alphabétique (« ASC »).

La requête est complétée par « v.NOM_VARIABLE, v.VALEUR, v.DATE_J, v.NUM_TRAM », ce qui signifie que l'on veut également sa valeur, sa date d'émission et le numéro de tramways à laquelle elle appartient (parmi les 17 tramways angevin).

La variable « sl_getDatePeriod() » est une variable d'ajustement dans le cas où l'opérateur aurai souhaité obtenir des informations passé, la variable contient dès lors la date choisi par l'opérateur. Par défaut, cette valeur correspond au moment présent.

De la même manière, la variable « m_numeroRame » correspond au numéro de tramway sur lequel on souhaite obtenir des informations. Par défaut, cette valeur est 1.

Une fois reçus, les valeurs sont indexé et « découpé » afin d'être manipulé.

A noter que les valeurs sont reçus une à une, à la suite.

Figure10 : Valeurs reçu avant manipulation

```
int index =  data.mid(2,2).toInt();
int value = data.mid((data.indexOf("VALEUR ")+ 7),
                     data.indexOf("DATE") - (data.indexOf("VALEUR ")+ 7)).toInt();
```

Figure11 : Indexation et découpage de la valeur pour exploitation

La figure 10 montre une valeur (ici la n°64) avec son nom (mode_Bat_arch), sa valeur (0), sa date d'émission (le 05 Février 2021 à 16 :28) et son numéro de tramway (le numéro 16) ;

La figure 11 présente comment ses informations brute, sont recueillie. Afin de créer une indexation, la *trame* récupère (fonction mid pour milieu) à partir du 2ième caractère (les 2 premiers caractère étant des espaces vides) et sur les 2 caractères suivant (ici le caractère 6 et 4). Une fois récupérer, ses 2 caractères sont enregistrer dans une variable « index ».

De la même manière, la valeur de la donnée est isolée grâce à la commande mid et est enregistré dans une variable « value ».

La fonction « .toInt ()» permet de convertir le caractère dit alphanumérique (par exemple 6 et 4) en valeur purement numérique, afin d'en faciliter l'exploitation.

Plusieurs UE enseigné au CNAM m'ont été utile pour ce projet :

L'UE « NFP107 – Système et gestion de bases de données » : sur la partie SQL du projet, notamment dans la rédaction des requêtes.

L'UE « NFP121 – Programmation avancée » : sur les aspects fonctionnels du projet, en particulier la mise en relation entre le langage C++ et Javascript.

Trame : « message informatique » répondant aux normes d'un protocole de communication, il s'agit ici d'une trame SQL, le terme « trame » étant plus généralement utilisé

Mission Secondaire : Dématérialisation de formulaire

Contexte

Au sein du pôle technique de RATP Dev, il existe une sous-direction appelé MR (pour Matériel Roulant). Leur travail consiste au maintien du bon fonctionnement des tramways. Parmi leurs missions, ils ont la charge de réaliser périodiquement des opérations de maintenance, également appelées formulaire préventif de maintenance.

Ces opérations de maintenances sont imposées par notre constructeur : ALSTOM. Il décide des opérations à réaliser, et nous indique à quelle fréquence est-ce qu'ils doivent être réalisés. Cette fréquence est déterminée en fonction du kilométrage des tramways.

Tous les 15.000km, une série d'opérations est nécessaire. Il en va de même tous les 30.000km, 60.000, 120.000, 180.000, 300.000, 360.000, 600.000 et 900.000km.

Le pas de maintenance de 15.000km (le plus faible) est atteint très régulièrement, imposant ainsi des opérations très régulièrement.

Ses formulaires sont pour l'heure, au format papier. Néanmoins, ce support pose un problème de logistique. En effet, l'archivage de ces formulaires, requiert un espace croissant de stockage. De plus, l'analyse post-opération est plus complexe, car tout le travail d'exploitation doit se faire manuellement

Enjeux

Afin de répondre à cette problématique de logistique et d'exploitation, mes supérieurs ont souhaité utiliser des formulaires numériques.

J'ai été désigné pour réaliser cette transition. Un outil de dématérialisation m'a été proposé : KIZEO Forms. Il s'agit d'une société française, qui a développé cet outil précisément pour ce genre de missions.

Les objectifs de cet outil sont :

- Une utilisation fluide et ergonomique des formulaires pour les techniciens chargé de les remplir
- Un stockage efficace et automatique de ces formulaires
- La possibilité d'exploiter ultérieurement ces documents

Par ailleurs, si cette dématérialisation convient aux différentes équipes, cette solution sera étendue aux sous-directions utilisant également des formulaires, et rencontrant des problématiques similaires.

J'ai eu la responsabilité de la validation de l'outil KIZEO. Après un temps de prise en main de l'outil, j'ai communiqué à mes supérieurs les différentes possibilités qu'offre cet outil, et confirmer que celui-ci convenait à leurs attentes.

Mise en place des formulaires

Création des formulaires

Je suis responsable de la réalisation de ce projet, ainsi je suis invité à prendre connaissance du résultat que souhaite obtenir les techniciens remplissant ces formulaires ainsi que les responsables chargé de l'exploitation futur de ces documents.

KIZEO Forms repose sur le principe du « drag-and-drop » pour la création de formulaire.

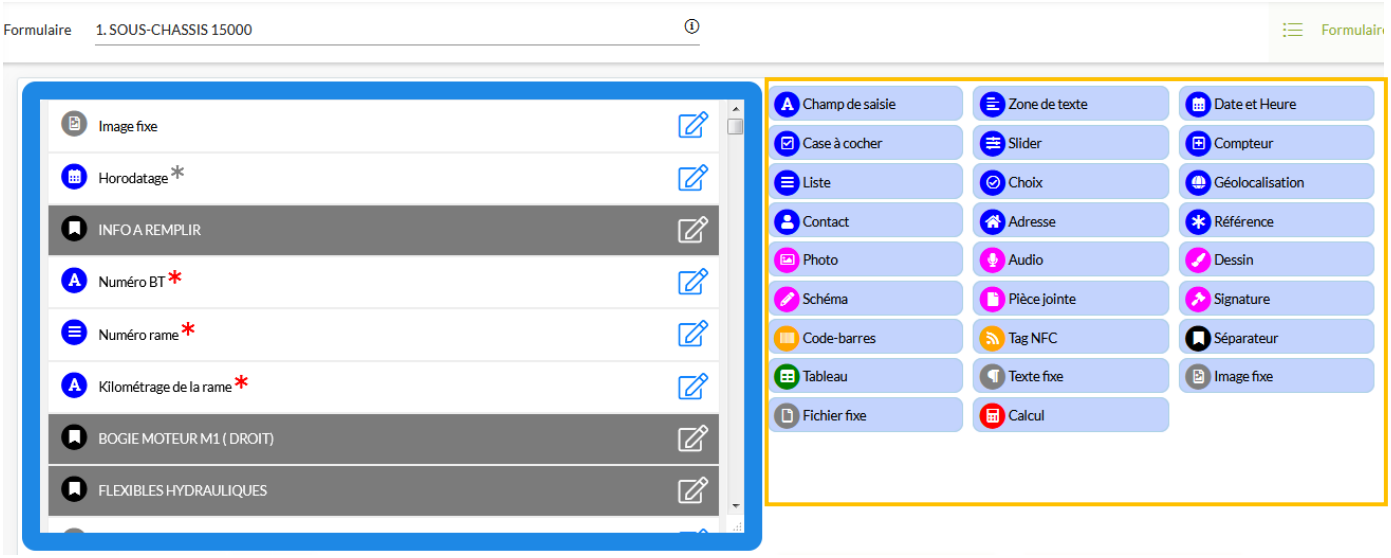

Figure12 : fenêtre de création des formulaires Kizeo Forms

L'encart orange à droite présente les outils disponibles pour cette création.

Ainsi, l'on peut placer des éléments de texte, tel que le nom d'une opération ou bien d'une procédure. Créer des listes à choix multiples que le technicien remplira au moment de compléter le formulaire, mais il est également possible d'utiliser des outils puissant.

Par exemple, KIZEO fournit un *item* de géolocalisation. Cet item va automatiquement identifier la localisation du technicien lorsqu'il remplira le formulaire.

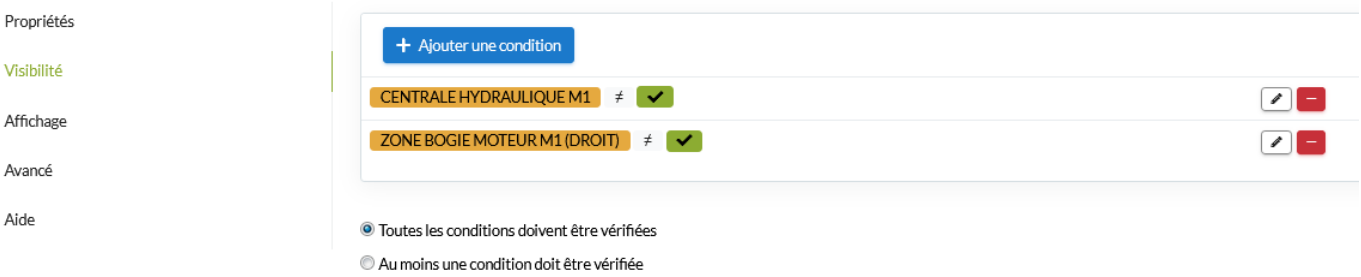

Figure13 : fenêtre de paramétrage des items KIZEO

De plus KIZEO fournit un panel de paramètres, permettant d'ajuster les items selon les besoins. Ainsi j'ai pu rendre des champs texte invisible si certaines ont été cocher. Cela sert au technicien lorsqu'ils ont remplit une partie du formulaire, afin de réduire la taille de celui-ci.

Item : Elément fournit par KIZEO Forms répondant à un besoin. L'item image va permettre à l'opérateur créant le formulaire de placer une image fixe (exemple : le logo de l'entreprise).

Archivage des formulaires

Sur chaque formulaire que complète un technicien, il existe un bouton permettant (lorsque le formulaire est entièrement complété) de le transférer en BDD.

Ce transfert va en premier lieu déposer le formulaire remplit, au sein des BDD de KIZEO. Une fois déposé chez KIZEO, une conversion du formulaire va être opérée. En effet les différents items de KIZEO sont indexés grâce à une balise d'identification (voir figure 14)

Figure14 : Balise d'identification des éléments

Il est possible de fournir un fichier Excel à KIZEO, qui reprend ces numéros de balises. Ce fichier Excel va remplir automatiquement les valeurs inscrit par les techniciens, à l'endroit où son placer les balises correspondante.

Par exemple, si une balise s'appelle « ##valeur_chaudière## » et qu'il s'agit d'une valeur qu'un technicien à rentrer. Le fichier excel va automatiquement remplacer sa balise par la valeur inscrit par le technicien.

Ainsi, lorsqu'un responsable souhaite étudier un formulaire, il obtiendra une copie du fichier Excel avec les valeurs remplit automatiquement.

Pour accéder à ce fichier Excel, il est nécessaire pour le responsable de le trouver sur le site internet de KIZEO. Cependant il est possible de mettre en place un système de mail automatique qui lui permettra de l'avoir directement sur sa boite mail.

De plus, il est possible de transférer automatiquement se fichier Excel via un *FTP* automatique. Ce transfère FTP va enregistrer une copie du document Excel sur le réseau intranet de RATP Dev, permettant ainsi un gain considérable de temps et d'organisation.

Formation KIZEO

Au vu de la quantité considérable de formulaire à transformer, il m'a été demandé de préparer des procédures d'utilisation et de création de formulaires. De plus, je vais avoir la charge de former directement un collaborateur à la prise en main de cet outil.

L'UE « GND100 Management de projet » m'a été utile pour se projet, en particulier sur la consultation des équipes technique MR et ma capacité à recenser leurs besoins.

FTP : File Transfer Protocol, est un protocole de communication dédier au transfert de fichier peu volumineux tel que des fichiers texte, Excel et PDF.

Conclusion

Je suis en collaboration avec la société RATP Dev Angers depuis maintenant 2 ans. Cette collaboration m'a permis de comprendre beaucoup de chose vis-à-vis du monde de l'entreprise et du monde professionnel.

J'ai eu la confiance de mes supérieurs pour la conception et la réalisation de 2 projets particulièrement conséquent, et à ce titre, j'éprouve beaucoup de fierté et de reconnaissance.

Par ailleurs, j'ai pu développer mes compétences en informatique (principalement grâce au projet de télémaintenance) de manière très importante. Etre le seul développeur informatique de ma société m'a forcé à puiser dans de nouvelles ressources et à adapter mes méthodes de travail. De cette manière j'ai énormément progressé, notamment en prise d'initiative.

De plus, ces 2 missions m'ont offert la chance d'être responsable de l'avancement de projet qui concerne plusieurs dizaines de personnes. J'ai eu la charge d'organiser des réunions, de solliciter des avis. J'ai pu fournir mon expertise, imposer mon avis et solutions à mes collègues.

En matière de management et d'organisation, je considère avoir beaucoup progressé.

Je remercie tout particulièrement Mr. Yann Brossard, mon responsable depuis 2 ans, qui m'a demandé et offert une grande autonomie de travail, répondant systématiquement à toute mes questions. Il m'a également assisté dans mes obligations scolaires, m'obligeant à le prioriser face au travail professionnel, et me soutenant au quotidien.

Je remercie Jeremy pour m'avoir aidé à mettre en place le système de télémaintenance, me donnant des compléments d'informations sur le fonctionnement du tramway ainsi que de la documentation.

Je remercie également Mathieu du Service informatique pour m'avoir aidé occasionnellement, notamment sur les requêtes SQL, ainsi que pour d'autres éléments technique.

Je remercie le reste de l'équipe informatique, ainsi que de l'équipe CFA avec qui nous travaillons en proximité immédiate, pour leur accueil chaleureux et bienveillant.

Je remercie également les techniciens MR avec qui j'ai travaillé pour mettre en place les formulaires KIZEO et qui m'ont encouragé dans mon travail.