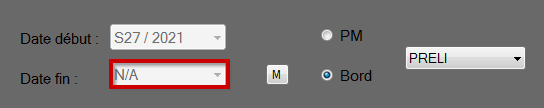
## Liste des améliorations / corrections à apporter :

### Empêcher la réservation d’équerres et de tins sur un chantier bord si la date de fin n’est pas définie

**Description :** La date de fin d’un chantier bord par défaut est ‘0001-01-01’. Celle-ci est donc antérieure à la date de début du chantier bord. Réserver une équerre ou un tin dans ces conditions ne doit pas être possible.



**Solutions envisagées :**

1 ) Si la date n’est pas valide, on affiche un message à l’utilisateur lui demandant de modifier la date.

2 ) Si la date n’est pas valide, on cache ou on désactive le bouton pour réserver.

Attention toutefois, si l’utilisateur change la date, il faudra alors remettre le bouton dans son état initial.

3 ) On définit que la date de fin d’un chantier bord par défaut est la même que la date de début du chantier bord.

### Création d’un nouveau bloc pour un navire

**Description :** Lorsque l’utilisateur saisit le repère d’un bloc, si celui-ci n’existe pas pour le navire sélectionné, l’utilisateur doit être en mesure de créer ce bloc.

*Remarque :* L’utilisateur doit alors renseigner au minimum les dates de début et de fin du chantier PM

Solution envisagée : Lorsque le bloc n’existe pas, on affiche un Panel avec des ComboBox pour sélectionner les dates de début et de fin ainsi qu’un bouton pour valider.

**Événement(s) concerné(s) :**

tb\_rechercheBloc\_TextChanged L437

nouveauBoutonValiderCreationBloc\_Click N/A

**Existant :**

**BDDQuery :**

public Bloc newBloc(navire, repere, dateDebutPm, dateFinPm) L1644

Mise à jour du PDF d’un bloc

**Description :** Le PDF d’un bloc doit être mis à jour lorsque les données concernant ce dernier sont modifiées : changement des dates de début/fin du bloc, réservation d’équerres, déréservation d’équerres, prêt d’équerres associées à un bloc.

**Événement(s) concerné(s) :**

btn\_modificationDateDebut\_Click L653

btn\_modificationDateFin\_Click L713

btn\_validerReservation\_Click L864

btn\_modifNombreEquerres\_Click L1769

btnValiderPretEquerre (si different du btn déjà existant) N/A

**Existant :**

**GenerateurPdf**

public bool genererPdf(navire, pm, repere, semaineDD, anneeDD, semaineDF, anneeDF, reservations, prets)

Emplacement du bouton pour rendre la main

**Description :** Le bouton pour « rendre la main » doit être plus visible pour ne pas être oublié.

*Remarque :* Si l’utilisateur clique sur un bouton du menu principal autre que celui destiné à la réservation d’équerres, de tins ou destiné au prêts d’équerres; l’utilisateur doit automatiquement rendre la main.

**Existant :**

\_utilisateurActuel.Main = false;

\_BDDQuery.updateUtilisateur(\_utilisateurActuel);

Valeur par défaut du NumericUpDown correspondant au nombre de tins à réserver

**Description :** Parfois (quand il n’y a aucune réservation de tins, par exemple) la valeur du NumericUpDown est out of range.

**Événement(s) concerné(s) :**

btn\_reservation\_Click L1040

nud\_nombre.Value = 0; L1091

### Prêter des équerres

**Description :** Il doit être possible de prêter des équerres. Il y a 2 différents modes de prêts :

* Un prêt classique :

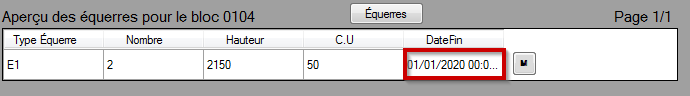
L’utilisateur choisit les équerres à prêter de la même manière qu’il réserve des équerres (saisie de la hauteur, de la charge…). Ces équerres doivent être disponibles sur une période déterminée par l’utilisateur. L’utilisateur doit pouvoir saisir une remarque concernant le prêt.

* Un prêt associé à un bloc :

Il arrive parfois que des équerres se libèrent avant la fin du bloc. Hors, une réservation bloque les équerres sur toute la durée du bloc. Pour pouvoir profiter des équerres avant la fin du bloc, on ‘triche’ en prêtant une équerre à un bloc plutôt qu’en la réservant.

Ainsi, les dates de début et de fin du prêt doivent coïncider avec les dates du bloc.

La liste des équerres prêtées à un bloc est consultable par les membres de l’équipe Coque Métallique. La date de fin doit respecter le format ‘S07 / 2018’ et prendre en compte une marge de une semaine.



*Remarque :*

- Toutes les dates sélectionnées par les membres de l’équipe CM tiennent déjà compte des marges imposées.

- En base de données, les attributs ***dateDebut*** et ***dateFin*** doivent être valorisés et ce même si ***dateDebut*** équivaut à la date de début du bloc.

**Existant :**

**BDDQuery :**

public List<string> getAllPretsBloc(bloc, pm, affichageSimplifie)

Modifier les lignes 1509 et 1520 pour afficher les dates dans le format imposé

listeDonneesDgv.Add(Convert.ToString(dr["dateFin"]))

### Sélection des dates

**Description :** Toutes les dates, sauf la date de début du chantier bord d’un bloc, doivent pouvoir être modifiées.



En base de données, les dates de début et de fin d’un bloc sont stockées sans prendre en compte la marge imposée.

On peut déjà modifier les dates seulement des erreurs surviennent parfois :

* Si on sélectionne la dernière semaine d’une année, l’année stockée en BDD n’est pas toujours la bonne.
* Lorsque l’on change une date, il arrive que la date stockée en BDD soit correcte mais que celle affichée ne le soit pas

**Événement(s) concerné(s) :**

btn\_modificationDateDebut\_Click L653

btn\_modificationDateFin\_Click L713

### Page de gestion

**Description :** Tous les onglets de la page gestion (Équerres / Types d’équerres / Tins / Travaux / Navire / Transport / Utilisateurs / Paramètres) doivent permettre d’ajouter et de modifier les données.

Tous les attributs, sauf les ID, sont modifiables.

Attention toutefois, les données saisies ou sélectionnées par l’utilisateur doivent faire l’objet d’un contrôle. Ces dernières devront obligatoirement être cohérentes (Ex : Un type d’équerre avec plusieurs propriétés ne peut pas être de type ‘statique’).

Gestion des équerres

Création d’une nouvelle équerre :

* Le repère de l’équerre doit être de la forme <lettre(s)><chiffre(s)> (Ex : ERB123)
* Si l’équerre est d’un nouveau type, le type doit être créé au préalable

Modification d’une équerre :

* Le repère de l’équerre peut changer. Il doit toutefois être unique et respecter la forme indiquée précédemment. La partie <lettre(s)> doit être identique pour toute équerre du même type.

Gestion des types d’équerres

Création d’un type d’équerre :

* C’est lors de la création d’un type d’équerre que ses propriétés (hauteur, charge et moyens de transports) sont définies.
* La valeur de l’attribut ***reglageHauteur*** doit correspondre avec les propriétés saisies. (Ex : Une équerre ne peut pas être fixe si elle possède plusieurs propriétés).
* Chaque propriété doit être associée à un ou plusieurs moyens de transport.

Gestion des tins

Il doit être possible de créer un nouveau type de tin.

Le nom ainsi que le nombre de tins d’une catégorie peut être modifié. Ce nombre peut être associé à la date actuelle ou à une date sélectionnée par l’utilisateur.

Gestion des navires

Il doit être possible de créer et de renommer les navires.

Gestion des transports

Il doit être possible de créer et de renommer les moyens de transports. On ne peut pas créer de nouvelle classe. Les 3 classes possibles sont A, B et C.

Gestion des travaux

Des travaux peuvent être réalisés sur une équerre. Les dates de début et de fin des travaux doivent être renseignées. Il doit être possible d’ajouter une remarque.

Gestion des utilisateurs

* Le nom, le prénom ainsi que le rang d’un utilisateur peuvent être modifiés.
* Il existe actuellement 4 rangs. La suppression du rang « Manager [3] » a été envisagée
* Il est impossible de modifier le rang de quelqu’un ayant un rang supérieur
* Il est impossible d’attribuer à une personne un rang supérieur à celui possédé par l’utilisateur

Existant : voir DataGridView\_listeUtilisateurs.txt (Documentation\Logiciel). Le code qui s’y trouve permet de modifier (visuellement uniquement) le rang des utilisateurs. Peut-être faut-il décommenter certaines lignes.

Modification des paramètres

La modification des paramètres est une partie un peu complexe qui mérite d’être expliquée de manière assez détaillée :

Le logiciel doit être capable de proposer à l’utilisateur de modifier certains paramètres concernant les emplacements de certains fichiers ou la connexion à la base de données.

Par soucis de simplicité, nous nommerons **paramètres fichiers** les paramètres concernant les emplacements de certains fichiers et **paramètres base de données (BDD)** les paramètres concernant la connexion à la base de données.

Les paramètres permettant de se connecter à la base de données ont été saisis directement dans le code de la classe ConnexionBDD.

static private string \_user = "root";

static private string \_password = "root";

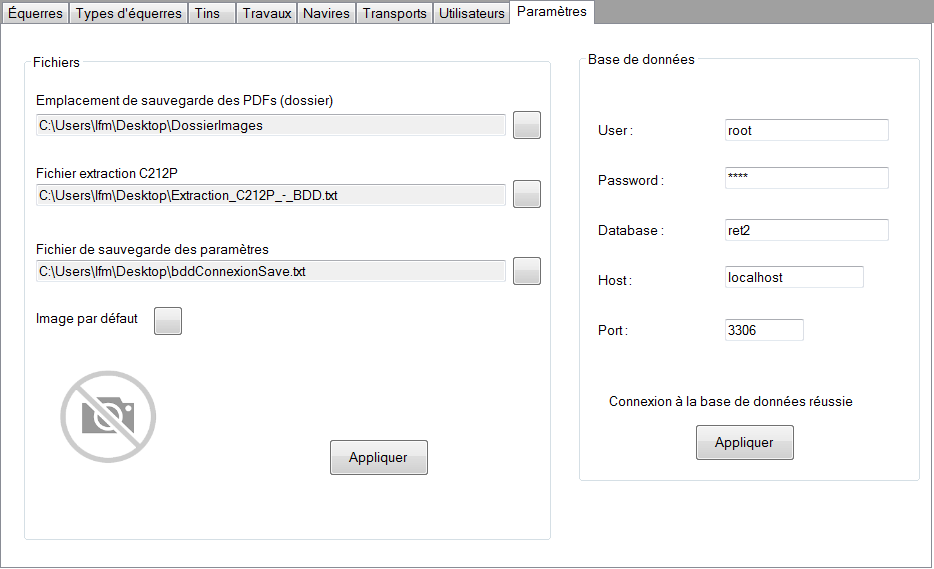
static private string \_database = "ret2";

static private string \_host = "localhost";

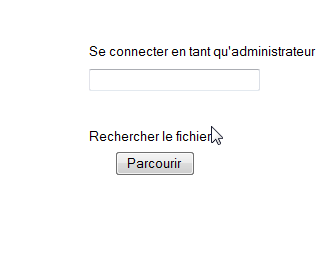
static private string \_port = "3306";

Si l’emplacement de la BDD vient à changer, il serait alors impossible pour le logiciel de s’y connecter (à moins de modifier le code).

C’est pour cela qu’il faut proposer à l’utilisateur de modifier les paramètres BDD depuis l’interface graphique.



Si le logiciel démarre et est dans l’incapacité de se connecter à la base de données, il faut alors faire apparaître ce panel ***: pnl\_parametres\_connexion***



Sans BDD, impossible de connaître le rang de l’utilisateur actuel. Par conséquent on demande un mot de passe à l’utilisateur. Si le mot de passe saisi est le bon, on propose à l’utilisateur de modifier les paramètres BDD.

Il a été convenu que le logiciel devra générer un fichier .txt sur le bureau de l’utilisateur ayant rétabli la connexion à la BDD. Ce fichier texte contiendra les nouveaux paramètres (valides) de la connexion à la BDD. L’utilisateur devra alors partager ce fichier sur un emplacement publique, de manière à ce que

tous les autres utilisateurs puissent y avoir accès. Il devra aussi remplacer depuis l’interface graphique l’emplacement du fichier de sauvegarde des paramètres.

Chaque autre utilisateur devra alors cliquer sur « Parcourir » puis sélectionner le fichier .txt.

Le logiciel se connectera alors à la BDD, grâce aux informations contenues dans le fichier, à chaque démarrage.

**Mise à jour et sauvegarde de la base de données**

**Description :** Il doit être possible à l’utilisateur de mettre à jour la base de données (par rapport au fichier extraction C212P) à n’importe quel moment. Si la base de données a été mise à jour de cette manière, la date de la dernière mise à jour ne doit pas changer.

De la même manière, l’utilisateur doit être capable de sauvegarder les données de la BDD dans un fichier (.bak par exemple) et d’utiliser celui-ci comme source lors de la réinstallation de la BDD.

**Existant :**

**BDDQuery :**

public void updateBDD()

public void updatePersistanceParametresDateMiseAJour()

### Historique des modifications

Le logiciel doit être assez autonome. Il doit parfois effectuer des modifications sans en avertir l’utilisateur. Toutefois, quand une modification n’est pas possible / envisageable, l’utilisateur doit en être averti.

Prenons un cas simple : Soit deux blocs, B1 et B2 et une équerre E.

Le chantier du bloc B2 commence juste après la fin du chantier B1.

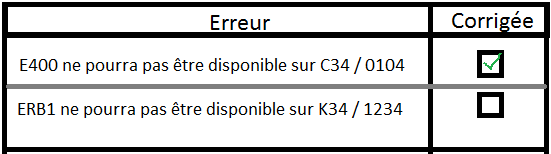
E est réservée sur B1 et sur B2. Seulement, le chantier de B1 a du retard et ne pourra être fini avant que B2 commence. L’équerre E ne peut pas quitter le chantier de B1 avant la fin de ce dernier. Elle ne sera donc pas disponible sur le chantier du bloc B2.

On doit alors regarder si une équerre, du même type que E, est disponible pour la durée du chantier de B2. Si c’est le cas, on remplace E par une autre équerre de même type dans les réservations de B2. Si ce n’est pas le cas, on doit en avertir l’utilisateur.

Celui-ci doit pouvoir consulter la liste des équerres qui posent problèmes (et donc des modifications à apporter par la suite).

L’historique des modifications pourra être trié par date et/ou par navire. Pour ce faire, il serait judicieux d’envisager un DataGridView ou une ListView, avec une CheckBox à l’extrémité de chaque ligne, à cocher si une correction a bien été apportée.

De manière assez simplifiée

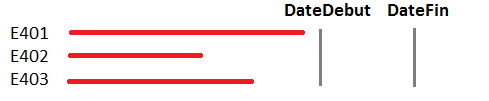


Si des modifications doivent être apportées, un voyant doit en avertir les utilisateurs.

### Optimiser la réservation des équerres

**Description :** Lors de la réservation d’une équerre, l’utilisateur choisit entre les plusieurs types qui lui sont proposés. Le programme réserve alors la « première équerre qui vient » ayant pour type le type choisi. Le but ici, est d’optimiser la réservation des équerres de manière à ce qu’une équerre passe le moins de temps possible sans réservation. Le programme ne prendrait alors plus « la première équerre qui vient » mais plutôt celle qui n’a pas été réservée / prêtée depuis le plus longtemps.

Dans cette situation, on préférera réserver l’équerre E402 en priorité, puis l’équerre E403



**Existant :**

**BDDQuery :**

public List<int> getEquerresDisponibles(bloc, pm, hauteur, charge) L1192

La requête doit être modifiée.

Note : Avant tout insert/update en BDD, la valeur des textBox, richTextBox, maskedTextBox, ainsi que de tous les objets avec une ‘length’ variable doivent faire l’objet d’une vérification. La length d’un composant ne doit pas pouvoir être supérieure à celle autorisée en BDD.