Michel de Broux
Simon Lardinois
Victor Lecomte

Modifications de la conception

LSINF1225
Groupe V

Charles Momin
Valentin Rombouts
Harold Somers

Nous allons présenter ici les modifications que nous avons faites en passant de l'étape de conception à l'étape d'implémentation, et leur justification. Nous avons refait le schéma relationnel et le schéma UML pour les illustrer.

1 Internationalisation de la base de données

Comme mentionné dans le rapport 3 (implémentation Android), nous avons dû adapter la base de données à une approche multilingue. Nous avions déjà joint notre nouvelle base de données avec ses instructions de création ; nous joignons maintenant à ce rapport-ci le schéma relationnel refait (fichier `relationnel-corrige.pdf`). Nous avons décidé de garder les noms de champs existants en français pour mettre en évidence le fait que la structure principale du modèle présenté dans le rapport 1 n'a été en rien modifieé.

2 Modifications du schéma UML

Lors de l'implémentation, nous avons modifié quelques éléments mineurs du diagramme de classe : nous avons ajouté quelques opérations et modifié quelques autres, pour s'adapter à la plateforme utilisée, par oubli lors de la conception, ou pour rendre l'implémentation plus aisée. Nous insistons toutefois sur le fait que ces changements sont anécdotiques, et que la structure principale de notre programme est restée solide et identique.

Nous joignons à ce rapport une version corrigée de l'ancien diagramme de séquence, dans le fichier `uml-dao-corrige.png` (principalement, nous avons enlevé les getters/setters non pertinent pour ce type de document) et une nouvelle version mettant en évidence les changements, dans le fichier `uml-dao-new.png`. Cette nouvelle version marque avec un `+` les opérations ajoutées, avec un `-` les opérations enlevées (rendues inutiles par d'autres) et avec un `*` les opérations modifiées.

Passons rapidement en revue les raisons pour les différents changements :

— `Utilisateur` : Nous avons ajouté le champ `motDePasse` pour permettre à l'utilisateur d'afficher le mot de passe actuel dans les paramètres d'utilisateur. De plus, nous imaginons qu'il sera utile plus tard pour effectuer des opérations en s'identifiant auprès du serveur central. Nous avons supprimé le champ `langue` car le système Android préconise que l'application s'adapte directement à la langue du téléphone.

— `Client` : L'opération `viderPanier` n'existe plus dans l'interface dans client, car elle est appelée automatiquement quand on appelle `confirmerPanier`. L'opération `ouvrirCommande` a été ajoutée pour envelopper une procédure qui devait auparavant se faire directement via `DAOCommande`.

— `Manager` : L'opération `changerGrade` peut se faire simplement et directement par un appel `setGrade` dans `DAOUtilisateur`. Nous avons gardé la classe Manager car le fait qu'un utilisateur l'instancie signifiait qu'il était manager, et parce que nous n'excluons pas qu'avec un ajout ultérieur de fonctionnalités, cette classe obtienne des méthodes propres et intéressantes comme celles de `Serveur`.

— `Détail` : Le détail doit contenir son `idDétail` pour que l'on puisse modifier des informations qui y sont liées dans la base de données et la `dateAjouté` pour qu'elle puisse être affichée dans l'interface.

— `Consommation` : Nous avons dû ajouter le `nomAffichage` pour la traduction de l'application tout en gardant le `nom` pour la base de données. Étant donné que nous n'avions besoin du type que pour son icône, nous avons changé `type` en `icôneType`.

— `Ingrédient` : Nous avons ajouté `utilisationActuelle` pour retenir l'utilisation déduite par le panier tout en gardant la valeur réelle du stock pour l'afficher dans la consultation du stock. Nous avons ajouté deux opérations liées à cela : `getRestant` qui donne la différence entre le stock et l'utilisation panier, et `appliquerUtilisation` qui déduit l'utilisation du stock à la confirmation du panier.

— `DAOUtilisateur` : L'opération `créer` ne prend plus la langue, supprimée de la base de donnée, et crée un client par défaut. Les opérations `isLoginMdpCorrect` et `connecter` ont été fusionnées en une seule opération `tenterConnexion`. Les opérations `loginEstPris` et `emailEstPris` ont été ajoutées pour les vérifications à l'enregistrement. L'opération `getTousGrades` a été créée pour donner une vue d'ensemble des grades des utilisateurs dans le panneau de gestion des utilisateurs.

— `DAOCommande` : Nous avons fusionné les opérations `getPour` et `getOuvertePour`, qui étaient assez simples et toujours utilisées ensemble.

— `DAOConsommation` : Une opération `getParNom` a été créée car Android ne permet pas de passer facilement des données autres que numériques et string entre activités sans qu'il ne soit recréé. Pour donner à une activité une consommation, nous devions donc lui donner son nom.

— `DAOIngrédient` : Par facilité d'implémentation, l'opération `getInsuffisants` n'examine plus le panier pour savoir qu'elles ingrédients sont devenus insuffisants, mais parcourt simplement le stock en vérifiant les `utilisationActuelle`. L'opération `getSousSeuil` nous a permis de créer une nouvelle vue affichant exclusivement les ingrédients en-dessous de leur stock seuil. L'opération `appliquerUtilisations` remplace `décompter` avec notre représentation séparée des utilisations panier. L'opération `getParNom` permet de lier les produits aux ingrédients qu'ils utilisent tout en gardant une seule instance des ingrédients lors du chargement du menu.