

Extrait du Easter-eggs - Spécialiste GNU/Linux

<http://www.easter-eggs.com>

Application de statistiques

- Études de cas - Développement applicatif -

Date de mise en ligne : mardi 14 novembre 2006

Easter-eggs - Spécialiste GNU/Linux

Sommaire

- [Problématique](#)
- [Démarche](#)
- [Technologies](#)

Problématique

Fin 2005, Laforêt Immobilier a consulté Easter-eggs pour la réalisation d'un système de reporting d'activité de ses agences. Cet outil devait à la fois exécuter les requêtes pré-configurées, mais aussi être un éditeur d'état pour créer de nouvelles requêtes.

Connecté à une base de données relationnelle pour collecter les données de ventes, le logiciel à développer devait être hautement configurable, pour permettre de choisir dynamiquement des indicateurs et les statistiques à calculer. Les rapports devaient être éditables sous la forme de tableaux, dans un format tableur ou pdf.

Le cahier des charges fourni à Easter-eggs était pour l'essentiel un classeur excel, contenant l'ensemble des statistiques utilisées. Les lignes décrivaient des indicateurs (chiffres d'affaires, durées des ventes, prix de ventes, marges, etc...) et les colonnes des périodes de temps (année, trimestre, mois, etc...) ou des zones géographiques (régionale ou nationale).

Démarche

Le premier réflexe lors de tout projet de développement est de regarder ce qui existe déjà en libre pouvant répondre aux besoins du client. Notre attention s'est portée sur le projet [Agata](#). Ce projet libre dispose d'un bon outil graphique de gestion de requêtes, nécessitant tout de même un minimum de connaissances SQL. Destiné à l'origine à l'extraction de données, il est peu adapté pour exécuter des traitements. L'affichage des rapports finaux n'est pas non plus assez développé. L'application est tellement aboutie que toute modification aurait été très lourde. Nous avons aussi examiné la librairie [Rlib](#). Il s'agit d'une librairie assez puissante dont nous nous sommes finalement passée, parce qu'elle aurait ajouté une couche supplémentaire dans notre logiciel.

Après avoir décidé de construire une solution dédiée, et après avoir été sélectionné par le client sur la base d'une maquette, nous sommes entrés dans un cycle de production itératif. En effet, il nous fallait absolument mieux définir le besoin des utilisateurs et valider que notre travail y répondrait convenablement. Aussi avons-nous commencé par produire un prototype, qui a été tout de suite manipulé par les utilisateurs. Puis, au fil des réunions, le prototype s'est progressivement enrichi de nouvelles fonctionnalités (nouveaux états prédéfinis, fonctions statistiques, type de colonnes, etc.), tout en intégrant les remarques du client au fur et à mesure.

Au final le logiciel développé permet de créer des rapports, sur des périodes de temps et localisations géographiques choisies. Un rapport est constitué de colonnes et de lignes, que l'utilisateur doit créer individuellement. Les lignes sont assemblées sous forme de groupes, un groupe étant constitué de critères géographiques ou de critères sur les données (types de biens, caractéristiques du bien, etc...). L'utilisateur peut enregistrer les rapports qu'il crée. Il peut aussi exécuter des rapports prédéfinis.

Technologies

Nous avons assez rapidement choisi de faire une application web, pour répondre à des problématiques de déploiement. Nous avons sauté sur l'occasion pour écrire notre première application utilisant les technologies [Ajax](#). C'était à vrai dire l'unique moyen d'obtenir une application dynamique, tout en restant web et multi plates-formes.

Pour le développement, nous avons adopté un système [MVC](#), en séparant le modèle, le coeur de l'application qui génère des rapports, de son contrôleur (l'ensemble des fonctions qui manipulent le modèle, dans le cas présent il s'agit de la création du rapport), et de la vue, qui est l'interface graphique (composants HTML et Ajax).

La principale difficulté rencontrée a été la comptabilité avec Internet Explorer. Tous nos développements ont été effectués et testés avec le moteur [Gecko](#). Nous avons bien sûr réalisé des tests avec Internet Explorer, mais les retours client nous ont démontré que ces tests n'étaient pas assez poussés.

Si nous devions écrire cette application aujourd'hui, sans doute réutiliserions-nous davantage de composants que nous l'avons fait la première fois. Il existe maintenant de nombreuses bibliothèques Ajax, ou PEAR, qui nous auraient fait gagner du temps, et qui auraient probablement résolu les problèmes de comptabilité de navigateur. Néanmoins, en nous en passant, nous avons acquis une expertise certaine sur ces technologies.