

Exemple d'application commerciale écrite en Rebol



EasyBraine SA
Karim El Founas (Goldevil)

Pour RebolFrance.org
Version du 24/08/2004

Table des matières

| | |
|--|----|
| Introduction..... | 3 |
| Description de la demande du client..... | 3 |
| Description des fonctionnalités..... | 3 |
| Outils existants..... | 4 |
| Infrastructure technique..... | 4 |
| Description de l'implémentation..... | 5 |
| Analyse du projet..... | 5 |
| Choix techniques..... | 5 |
| Choix internes..... | 6 |
| Adaptation infrastructure technique..... | 7 |
| Architecture du logiciel..... | 7 |
| Mise en oeuvre..... | 9 |
| Le framework EEE..... | 9 |
| L'application de gestion des tâches..... | 10 |
| Conclusions..... | 11 |
| Evaluation du déroulement du projet..... | 11 |
| Evolution..... | 12 |
| Commercialisation..... | 13 |
| Annexes..... | 14 |
| Références & Bibliographie..... | 14 |
| Remerciements..... | 14 |

Introduction

Le projet consiste en une application multi-postes, multi-utilisateurs de gestion de projets pour un atelier de menuiserie.

L'objectif principal de l'application est la collecte des temps d'activités par les toutes les personnes intervenant pour la fabrication. Ceci dans le but de connaître en temps réel l'état d'avancement de tous les projets en cours.

Description de la demande du client

Description des fonctionnalités

Chaque projet est découpé en plusieurs parties et sous-parties et sous-sous-parties,...

Par exemple, la fabrication d'un meuble peut se découper ainsi :

- Meuble avec éclairage
 - Construction partie haute
 - Plan
 - Choix des bois
 - Délignage et découpage
 - Finition
 - Placement éclairage
 - Montage en atelier
 - Construction partie basse
 - Plan
 - Choix de bois
 - Délignage et découpage
 - Finition
 - Montage en atelier
 - Livraison et montage sur site
 - Finition sur site

Chaque projet est lié à un client et chaque tâche et sous-tâche est assignée à un utilisateur.

Si la plupart des utilisateurs du logiciel ne doivent qu'encoder leur temps d'activités liés à telle ou telle tâche, d'autres utilisateurs avec des permissions plus élevées s'occupent de la gestion du logiciel. Ils peuvent ajouter, modifier ou supprimer ...

- des comptes utilisateurs
- des clients
- des projets
- des pointages de temps

Ces utilisateurs doivent également pouvoir visualiser les informations recueillies de manière graphique ainsi qu'extraire ces données vers des applications externes (Microsoft Excel).

Outils existants

Actuellement le client utilise des documents papiers remplis par les employés et qui sont ensuite encodés dans des feuilles de calcul Microsoft Excel.

D'autres feuilles de calcul contiennent le listing client, un calculateur d'offre, un système de suivi de projet.

Ce système est relativement difficile à maintenir et en particulier pour les raisons suivantes :

- Difficulté de synchroniser des documents utilisés par plusieurs personnes et dupliqués sur plusieurs machines
- Impossibilité de travailler à plus d'une personne en même temps sur un document.
- Nombreuses erreurs et imprécisions d'encodage à cause de l'absence de validation suffisante.

Infrastructure technique

L'infrastructure technique mise à disposition par le client est la suivante :

- Réseau Ethernet 100 Mbits/s
- Routeur ADSL avec Hub intégré 4 ports
- PC 1 : desktop : poste client
 - Classe Pentium4 2.6GHz - 256Mo RAM - HD 80Go - Graveur CD-ROM - Ethernet 10/100 Mbits/s
 - Microsoft Windows XP - Microsoft Office 2000
- PC 2 : portable : poste client
 - Classe Pentium4M Centrino 1.5 - 512Mo RAM - HD 30Go - Ethernet 10/100 Mbits/s
 - Microsoft Windows XP - Microsoft Office 2000
- PC 3 : portable : poste client
 - Classe Pentium3 500MHz - 192 Mo RAM - HD 6Go - Ethernet 10 Mbits/s
 - Microsoft Windows XP - Microsoft Office 2000
- PC 4 : desktop : serveur
 - Classe Pentium2 200MHz - 32 Mo RAM - HD 4 Go - Pas d'adaptateur réseau
 - Microsoft Windows XP - Microsoft Office 97

Description de l'implémentation

Analyse du projet

Choix techniques

Produit existant ou développement d'une application sur mesure ?

Une première analyse réalisée pour le client, et non détaillée ici, a été de définir quels produits commerciaux existants permettent de répondre à sa demande

Une étude rapide du marché a été réalisée et il s'avère que la plupart des produits ne sont pas adaptés pour les raisons suivantes :

- Le produit est trop cher.
- Le produit n'est pas ciblé sur le pointage de temps qui n'est qu'un module annexe. Du coup cela implique l'acquisition d'autres fonctionnalités et augmente le prix.
- Le produit nécessite trop de modification de l'infrastructure informatique existante. Souvent un serveur relativement puissant est nécessaire.

C'est pour cela qu'un développement sur mesure a été envisagé.

Langage de programmation

Le logiciel sera écrit en Rebol/View 1.2.1 pour les raisons suivantes :

- Rebol est un langage moderne et relativement facile à appréhender, avec une courbe d'apprentissage rapide. Mais il faut aussi admettre que le choix de cette technologie dépasse le simple cadre de se projet.
- La licence de Rebol/View est gratuite
- Aucune des fonctionnalités supplémentaires de Rebol/View/Pro ou Rebol/Command n'est nécessaire pour ce projet.
- La version 1.2.1 est la dernière version stable et officielle de Rebol Technology. Il a été décidé de ne pas migrer vers une version ultérieure avant d'avoir une version officielle stable non-beta (qui sera certainement la 1.3)

Le stockage des données se fera avec MySQL Server 3.23 pour les raisons suivantes :

- MySQL est une base de données offrant un bon niveau de fiabilité, stabilité et performances
- MySQL est disponible gratuitement
- Il existe un protocole MySQL gratuit pour Rebol compatible avec Rebol/View 1.2.1 et MySQL 3.2.23. Les versions ultérieures de MySQL ne sont pas supportées.
- MySQL est une base de données SQL très ouverte est accessible à partir de beaucoup de produits externes (Tableurs, serveur d'application,...)

Comme la base de données est disponible sur le réseau, il est possible d'y accéder avec d'autres outils et dans le projet actuel des feuilles de calcul Excel liées via ODBC permettront d'extraire des données essentiellement pour des informations statistiques.

Ceci permet de limiter la création d'écrans de statistiques complexes au sein de l'application.

Choix internes

Application sur-mesure ou réutilisable ?

Après une première analyse il s'est avéré que le coût de développement d'une telle application est difficilement compatible avec les ressources que le client était prêt à mettre dans ce projet.

Après avoir déjà écarté l'utilisation de logiciels commerciaux, il ne restait plus pour rentabiliser le projet qu'une seule alternative qui est d'amortir l'investissement sur plusieurs clients.

Par conséquent, il a été décidé de concevoir une application dont la plus grande partie est largement réutilisable de manière à en faire un produit commercialisable.

Le choix du Framework

Pour rendre la plus grande partie du code réutilisable il est important d'utiliser des méthodologies de programmations strictes et bien structurées qui séparent au maximum les spécificités d'une implémentation particulière des fonctionnalités de base, plus générales.

Il est aussi bien important de penser l'application telle qu'elle pourrait être utilisée par des clients avec des besoins différents sans se focaliser de trop sur les spécifications exactes du projet actuel.

Il est donc devenu évident que l'application devrait se découper en deux parties :

- Un framework qui propose une série de fonctionnalités générales ainsi que des librairie d'objets/fonctions à orientation business.
- L'application en elle même qui se repose sur le framework et sera la seule partie spécifiquement conçue pour le client.

Après un rapide tour d'horizon des technologies Rebol, il s'avère qu'aucun framework orienté vers les applications de gestion n'existe. Par contre, il existe déjà des technologies de plus bas niveau déjà développées comme des protocoles d'accès à des bases de données, des brokers ou encore des serveurs d'application web.

Il devenait donc évident que la conception du framework ferait partie intégrante du projet même s'il devait intégrer des technologies déjà développées.

Evidemment, la conception du framework allait largement augmenter le temps de développement de l'application mais cela améliore les chances de rentabilité par la commercialisation ultérieure d'autres produits semblables.

Type de licence

Etant donné que le produit n'est pas développé exclusivement pour ce client, la propriété intellectuelle du code appartient à EasyBraine SA. Par contre, en vue de garantir la pérennité de son investissement, en cas de disparition de EasyBraine SA, le client possédant une documentation technique et le code source aura la possibilité d'obtenir le support d'une autre société ou personne compétente.

Dans ce cadre-ci il a été également concédé que la licence du logiciel n'était pas définie par rapport à un nombre de postes clients et/ou serveurs.

Néanmoins, l'objectif est de proposer par la suite une licence commerciale plus restrictive.

Adaptation infrastructure technique

L'infrastructure technique a du être légèrement modifiée pour répondre à l'installation de l'application.

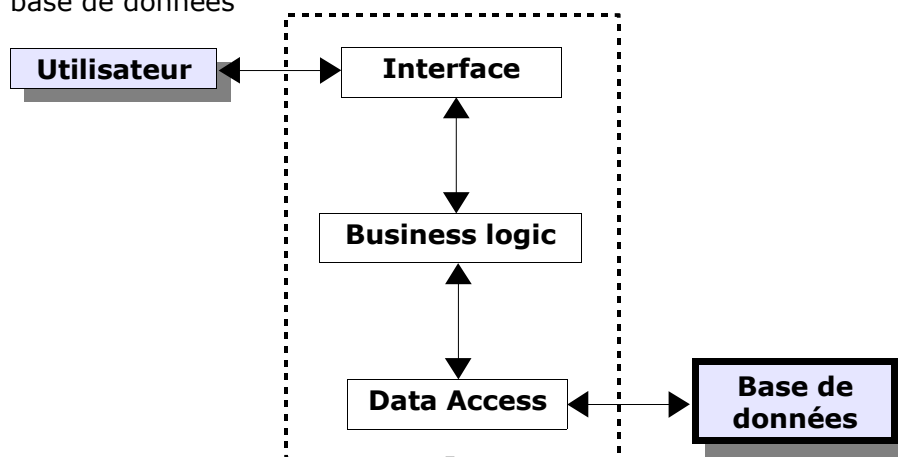
- Ajout d'un Hub 10Mbit/s 4 ports
- Modification du PC 4 (pour le rendre apte à servir de serveur de base de données)
 - Réinstallation de Windows 98 SE
 - Installation d'une carte réseau 10 Mbits/s
 - Ajout de 64 MO de RAM pour amener le total à 96Mo
 - Ajout d'un ventilateur supplémentaire

Le problème de l'absence de backup a été aussi abordée avec le client et un système de copie entre postes complété par un backup sur CD-ROM sera installé ultérieurement.

Architecture du logiciel

L'application sera structurée selon le modèle 3-tiers mais sans impliquer un client léger. L'idée est de découper chaque ensemble de fonctionnalités en 3 couches.

- Interface : ne gère que l'interface utilisateur
- Business logic : s'occupe de la logique même des entités de l'application
- Data acces : se limite au stockage et à la manipulation des objets dans la base de données



Il faut ensuite définir comment ces différents éléments vont se répartir dans l'infrastructure physique. Sur quel PC tournera

La base de données en elle-même est destinée à être installée sur le seul PC qui est susceptible de tourner sans arrêt (le PC 4)

On pouvait imaginer installer une partie de la logique applicative sur le serveur. Il existe alors deux approches :

- une application avec une interface web (client très léger)
- une application serveur connecté à des applications clients Rebol (client riche)

Mais comme le serveur à disposition est trop peu performant, il a été décidé de ne pas envisager l'installation d'une partie de la logique applicative dessus.

Par conséquent, chaque poste exécutera l'application complète. Seule la base de données est centralisée

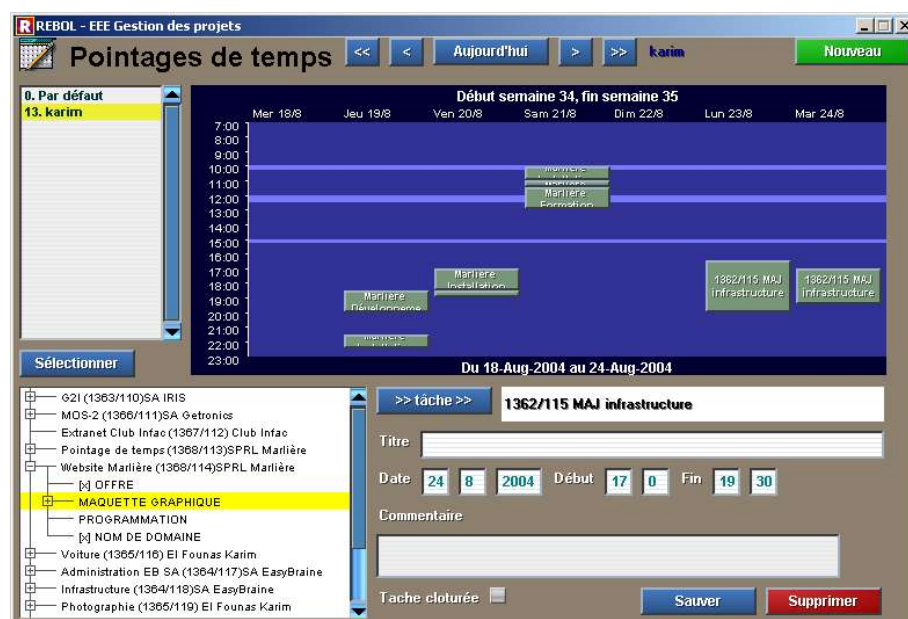


Illustration 1 Ecran principal d'encodage des temps d'activités

Mise en oeuvre

Le framework EEE

Le framework a été baptisé EEE qui est un acronyme de EasyBraine E-Business Engine.

Règles

Le framework doit imposer une structuration des composants :

- Arborescence des fichiers dans le répertoire de déploiement est bien précisé.
- La manière de nommer les fichiers est fixée
- La structure de la base de données est partiellement fixée
- Structuration de librairies selon l'architecture 3-tiers

La manière de programmer au sein du framework est également soumise à une série de règles :

- Utilisation de langue anglaise tant pour les choix de mots clés que pour les commentaires. Ceci de dans le but d'ouvrir les évolutions futures au plus grand nombre de développeurs
- Utilisation des entêtes de fichier rebol avec l'obligation d'utiliser certains attributs tels que "name", "title" ou encore "version"
- Conception des objets business sur le même modèle avec notamment une fonction constructeur.
- Manière d'indenter un script
- ...

Fonctionnalités

Le Framework doit offrir une série de fonctionnalités de bases :

- Une gestion de licence dans le but de permettre de gérer la diffusion du logiciel. L'idée est d'utiliser une combinaison d'un numéro de licence et d'une clé encryptée.
- Un chargement de librairies en mémoire au démarrage
- Un structuration de l'application dans une arborescence de fichier
- Un pool de connexions aux bases de données facilitant l'intégration de ces dernier.
- Un mécanisme de gestion des erreurs de plus haut niveau que les erreurs Rebol
- Un mécanisme de verrouillage de ressource (locking) qui est vital pour les applications multi-utilisateurs
- Un mécanisme de sequence automatique destiné à fournir une numérotation automatique aux applications.

Les bibliothèques

Il propose quelques bibliothèques standard :

- **internaute** : internaute est un autre nom pour "user" . Avec l'expérience l'utilisation de mots-clés très courants dans beaucoup de langage m'a amené à utiliser un autre nom.
- **permissions** : gestion des permissions des utilisateurs pour les ressources de l'application.
- **langage** : apporte le support du multilinguisme sans limitation du nombre de langues mais avec les limitations du Rebol. En effet ce dernier ne supporte pas des codages de caractères étendus comme l'UTF-8
- **gui** : Objets graphiques. Pour l'instant uniquement les skins d'Etienne Laurent
- **mysql** : Le protocole MySQL de Doc Kimbel

Et enfin, on retrouve d'autres bibliothèques directement liées à l'application à créer :

- **relation**: gestion des objets "relation" (c-à-d les clients, fournisseurs,...) et de leurs coordonnées (objets "coordinate")
- **task** : gestion des objets "tâches", "modèles de tâches" et "temps d'encodage"
- **graphic** : quelques objets graphiques de type "calendrier", "planning".

Chaque bibliothèque d'objet est déclinée en trois parties (c-à-d trois fichiers) correspondant au trois couches d'application. Par exemple pour les relations :

- **relation.r** : l'objet « relation » au sens business logic.
- **relation_db.r** : stockage et manipulation de l'objet dans la database (data access)
- **relation_view.r** : gère l'affichage de l'objet ou d'une liste d'objet au sein d'un écran (interface)

L'application de gestion des tâches

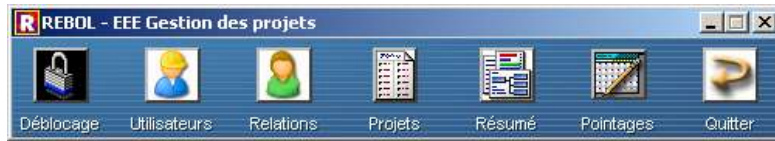


Illustration 2 Le menu principal de l'application

L'application Rebol

L'application en elle-même est répartie en plusieurs modules qui utilisent intensivement le framework EEE. Rassemblés dans un même répertoire, les modules ont chacun un nom explicite :

- mainmenu.r : menu principal
- login.r : entrée dans l'application par le login
- internaute_manager.r : gestion des utilisateurs
- relation_manager.r : gestion des clients et des coordonnées
- task_manager.r : Gestionnaire des projets
- task_summary.r : Affichage du statut des projets en cours
- timesheet_week.r : Ecran d'encodage des temps d'activités

Les feuilles de calcul

Comme la base de données est disponible sur le réseau, il est possible d'y accéder avec d'autres outils.

Dans le projet actuel des feuilles de calcul Excel liées via ODBC permettent d'extraire des données essentiellement pour des informations statistiques.

De plus, comme le client utilise déjà Excel intensivement, il est déjà familiarisé avec ses possibilités et peut facilement créer de nouvelles feuilles à partir des données encodées via l'application Rebol.

Néanmoins quelques feuilles Excel de base ont été fournies :

- relations.xls : C'est simplement le listing client
- projets.xls : projets en cours avec temps prévu et totaux des temps d'activités réalisés.
- timesheet.xls : Ensemble des pointages de temps par utilisateur.

Conclusions

Evaluation du déroulement du projet

Difficultés rencontrées

Comme mentionné, l'application est divisée en 3 couches distinctes :

- Interface
- Business logic
- Data acces

La partie Business logic a été développée en premier et ceci dans un temps inférieur à ce qui a été prévu. La facilité d'apprentissage du langage Rebol est l'élégance de son code sont pour beaucoup dans ce bénéfice.

La partie Data access a été terminée dans un temps relativement court également

Par contre, la partie interface a pris beaucoup plus de temps que prévu. C'était un premier projet en Rebol et l'offre s'était basé la connaissance de Rebol/Core et sa documentation pour l'évaluation du temps de développement.

Mais en fin de compte, beaucoup de difficultés sont survenues dans la mise au point de la partie graphique.

Essentiellement pour les raisons suivantes :

- L'incroyable pauvreté de la documentation des dialectes VID et DRAW en particulier le manque de tutoriaux.
- Le manque de certains objets graphiques (tree view, zone texte évoluée,...)
- Les limitations des objets graphiques existants (impossible de mettre deux fois le même libellé dans une textlist, taille limite d'une listbox choice, ...)

Il a bien fallu surmonter ces limitations. Pour cela plusieurs techniques ont été utilisées :

- Echange d'informations sur des forums de discussion Internet.
- Utilisation de composants graphiques de partie tierces (skins d'Etienne Alarent)
- Remplacement d'objets graphiques par d'autres moins adaptés mais utilisables

A l'heure ou ces lignes sont rédigées, il reste une longue liste d'améliorations et de bugs à corriger même si un prototype fonctionnel a été fourni au client.

On relèvera en particulier que l'application gère mal ma mémoire et cette dernière n'arrête pas d'augmenter et entraîne un ralentissement de l'application. Il est difficile de trouver facilement ou se trouve le problème, même avec des outils tels que Anamonitor.

Les choix technologiques

Malgré les problèmes rencontrés pour l'interface graphique, je suis très content d'avoir choisi Rebol.

J'espère que le langage évoluera encore beaucoup non pas au niveau de ses principes mais surtout au niveau des outils de développement disponibles (IDE,

debug, ...) Il n'y a qu'un niveau des composants VID que j'attends le plus de la version Rebol/View 1.3 qui est annoncée.

Je suis également content aussi de la base de données MySQL avec laquelle j'ai déjà réalisé une série de projets. Elle se satisfait assez bien des ressources limitées de l'ordinateur sur lequel elle a été installée et ses temps de réponse sont assez bons.

Et enfin, d'un point de vue purement personnel, je me suis familiarisé avec le langage Rebol par ce projet. J'ai trouvé très intéressant de pouvoir travailler sur ce projet qui peut prendre plus d'envergure qu'une simple implémentation pour un seul client.

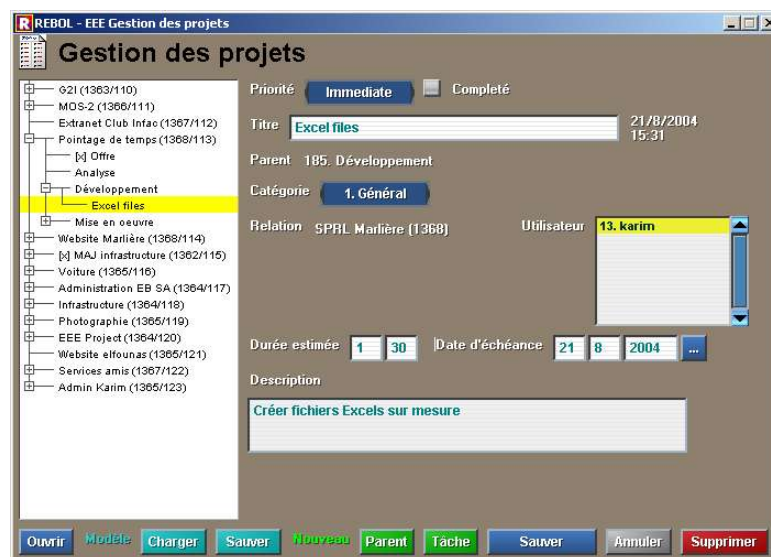


Illustration 3 Ecran d'encodage des projets

Evolution

Evolution de l'application

La conception de l'application m'a pris bien plus de temps que ce que j'ai facturé mais l'objectif avoué est de vendre l'application de gestion des tâches à d'autres clients mais aussi de continuer à augmenter ses possibilités. C'est son d'ailleurs son évolution qui dictera l'évolution du Framework EEE.

Je pense qu'un module logiciel de gestion commerciale (et de comptabilité très allégée) enrichirait le plus l'application par rapport à la demande des clients.

De plus, comme j'utilise l'application pour ma propre utilisation je vais mettre en priorité certains développement orientés vers la gestion commerciale et la comptabilité.

Les bibliothèques sur lesquelles j'aimerais travailler en priorité sont :

- relation : apporter un système de différenciation clients/fournisseurs/prospects
- commande : Objet complexe à formalisé surtout s'il faut qu'il englobe les différentes phases de la vie d'une commande, de l'offre à la facturation.

Evolution du framework

Outre la résolution des bugs, l'amélioration de la stabilité et des performances, l'objectif est de faire évoluer le framework EEE dans les voies suivantes :

- Corriger et optimiser le code. C'est le premier programme Rebol réalisé et il est clair qu'il est possible de faire nettement mieux.
- L'adapter à Rebol View 1.3 dès que ce dernier sera disponible
- Ecrire une autre couche data access destiné à la connectivité vers RebDB. RebDB est une base de donnée écrite en Rebol et qui permettra d'en faire une version standalone ne nécessitant pas de serveur de base de donnée externe.
- Ecrire une autre couche interface destinée au Web, cette fois. Il sera certainement intéressant de se baser ou s'inspirer de Magic! et RSP.
- Transformer le framework EEE pour obtenir un serveur d'application réparti. L'idée est de pouvoir séparer les différentes couches sur des machines différentes. Cela permettrait de déplacer plus de traitement sur un serveur et d'apporter de meilleures performances à des poste clients plus légés. Il est imaginable de multiplier les processus voire de faire du clustering de manière à supporter des charges importantes (utile pour le web). Des technologies comme Soccer ou Rugby peuvent être intéressantes pour ce type de choses.
- Etudier l'intégration d'autres technologies Rebol tels que Steel ou LadyBird.
- Et bien évidemment, ajouter d'autres modules de gestion d'entreprise (gestion commerciale, comptabilité,...). Ceci permettrait d'élargir l'offre commerciale basée sur le framework.

Commercialisation

Ce projet ne constitue pas le centre de l'activité professionnelle pour l'instant et je ne sais pas dans quelle mesure et dans quel sens il évoluera effectivement.

Il a été envisagé de vendre le framework EEE en tant que produit mais le marché potentiel est semble bien trop petit pour cela. Par conséquent, le coeur du framework sera diffusé sous une licence open source avec quelques librairies business de base (internautes & permissions).

Ainsi, si d'autre développeurs sont intéressés par ce Framework, ils pourront faire évoluer ce dernier de manière plus rapide. Il leur sera possible de fournir des produits commerciaux basés dessus mais aussi de l'enrichir de librairies business s'ils le désirent.



Annexes

Références & Bibliographie

Rebol Programmation - O. Auverlot (Edition Eyrolles – ISBN 2-212-11017-0)
Magazine Login – Posse Presse
EasyBraine SA (<http://www.easybraine.com>)
Rebol Technologies (<http://www.rebol.com>)
Rebol Ressources (<http://www.rebol.org>)
MySQL AB (www.mysql.com)
MySQL Protocol (<http://rebol.softinnov.org/mysql/>)
RebolFrance (<http://www.rebolfrance.org>)
Steel (<http://www.rebol.it/~steel/>)
LadyReb (<http://www.ladyreb.org/>)

Remerciements

O. Auverlot (pour RebolFrance)
Etienne Laurent (pour les skins)
Nenad Rakocevic (MySQL protocol)
Romano Paolo Tyenca (Anamonitor)
Les membres du forum de discussion Codeur.org