|  |  |
| --- | --- |
| Dédicace |  |

**A**

|  |  |
| --- | --- |
| Remerciements |  |

Ce travail a connu la participation de plusieurs personnes qui, d’une manière ou d’une autre y ont laissé leurs empreintes. Ma profonde gratitude va ainsi à l’endroit des personnes ci-après :

- le Directeur de l’**ESSEC**, Professeur **KAMDEM Emmanuel** pour tout son dévouement à faire du temple de Management une école toujours leader ;

- le Président du **CARPA** Monsieur **Dieudonné BONDOMA YOKONO** qui nous a permis de nous imprégner du monde professionnel au sein de la prestigieuse structure qu’il dirige ;

- Le coordonnateur du secrétariat technique du **CARPA**, Monsieur **Marcel Blaise MBELLA** ainsi que les Experts et le personnel administratif qui le constitue pour avoir facilité mon intégration au sein dans ce groupe ;

- Notre encadreur académique le **Professeur DONTSI** pour sa disponibilité à apporter les éclaircissements à notre travail ;

- Notre encadreur professionnel **M. NDJANA Willy Freddie** Expert technique au **CARPA**  pour sa patience, ses précieux conseils sa, capacité à nous transmettre les connaissances nécessaires à la pratique professionnelle durant notre stage académique et professionnel;

- A mes enfants pour leur soutien moral ;

- Tous mes collègues stagiaires ;

- A tous ceux qui ont œuvré de près ou de loin à la réalisation de ce rapport de stage nous vous remercions et que le Dieu Tout-puissant vous bénisse.

|  |  |
| --- | --- |
| Sommaire |  |

[Dédicace 3](#_Toc515660504)

[Remerciements ii](#_Toc515660505)

[Sommaire iii](#_Toc515660506)

[Liste des tableaux vi](#_Toc515660507)

[Liste des figures vii](#_Toc515660508)

[Acronymes et abréviations viii](#_Toc515660509)

[Quelques définitions ix](#_Toc515660510)

[Résumé xi](#_Toc515660511)

[Introduction générale 1](#_Toc515660512)

[Chapitre 2 Première Partie : Phase d’imprégnation 3](#_Toc515660513)

[Chapitre 1 : CONTEXTE ET PROBLEMATIQUE 4](#_Toc515660514)

[Chapitre 2 : GENERALITES 5](#_Toc515660515)

[Introduction 5](#_Toc515660516)

[2.1. Rapid Application Development 6](#_Toc515660517)

[2.2. L’outil Ruby On Rails 6](#_Toc515660518)

[2.3. Le langage Ruby 7](#_Toc515660519)

[4.1. Architecture et création d’un projet avec Ruby On Rails 9](#_Toc515660520)

[2.4.1 Création des classes et modèles 9](#_Toc515660521)

[2.5. Scaffolding 10](#_Toc515660522)

[2.5.1. Running 10](#_Toc515660523)

[2.6. Avantages de Rails 11](#_Toc515660524)

[2.7. L’outil Git 11](#_Toc515660525)

[2.7.1 Git 12](#_Toc515660526)

[2.8 Le pattern MVC 12](#_Toc515660527)

[2.8.1. Définition 12](#_Toc515660528)

[2.8.2. Présentation du modèle MVC 13](#_Toc515660529)

[2.8.3. Principe du MVC 13](#_Toc515660530)

[2.9. Constructeur d’un projet UML 14](#_Toc515660531)

[2.10. Le Développement Agile 14](#_Toc515660532)

[2.10.1. Les principes du Manifeste Agile 14](#_Toc515660533)

[2.10.2. Modélisation Agile(AM) 15](#_Toc515660534)

[Conclusion 16](#_Toc515660535)

[Chapitre 3 Deuxième Partie : Phase technique 17](#_Toc515660536)

[Chapitre 3 : ANALYSE ET CONCEPTION 18](#_Toc515660537)

[Introduction 18](#_Toc515660538)

[3.1. Cahier de Charges 18](#_Toc515660539)

[3.1.1 Fonctionnalités attendues de l’application 18](#_Toc515660540)

[3.1.1 Besoins de l’application 18](#_Toc515660541)

[Chapitre 4 : IMPLEMENTATION 20](#_Toc515660542)

[Introduction 20](#_Toc515660543)

[4.1. Langage de programmation et environnement de développement 20](#_Toc515660544)

[4.1.1. Langage de programmation 20](#_Toc515660545)

[Bibliographie 24](#_Toc515660546)

[Annexes 26](#_Toc515660547)

[Table des matières 28](#_Toc515660548)

|  |  |
| --- | --- |
| Liste des tableaux |  |

|  |  |
| --- | --- |
| Liste des figures |  |

|  |  |
| --- | --- |
| Acronymes et abréviations |  |

|  |  |
| --- | --- |
| Résumé |  |

|  |  |
| --- | --- |
| Introduction générale |  |

Il se dégage aujourd’hui un consensus quant aux possibilités ouvertes par les technologies de l’information et de la communication (TIC) qui se développent rapidement dans tous les domaines de l’entreprise et plus largement de la société́. Elles permettent de manipuler de l’information pour la stocker, la convertir, la gérer, la transmettre et la retrouver.

L’Internet est considèré comme étant un moyen idéal de communication, d’échange de données ou encore d’apprentissage, et est également un outil efficace pour avoir des informations sur un service ou un pro- duit. L’apprentissage est l’un des services privilégiés qu’Internet offre aux visiteurs. Dans cette optique, de nombreuses applications et sites Web dynamiques ont vu le jour.

Les institutions publiques gouvernementales sont parmi les établissements qui ont besoin d’un système informatique pour bien conduire leur travail en évitant la perte de temps. Ces institutions dépendent de plus en plus de l’informatique pour réaliser leurs objectifs, elles sont donc plus sensibles à la qualité́ des services informatiques fournis aux différentes catégories d’utilisateurs et sont à la recherche de moyens et des ressources pour améliorer leurs services. Ainsi, restaurer le plus rapidement possible le fonctionnement normal des services afin de minimiser l’impact négatif de celui-ci sur les activités métiers, et s’assurer que les meilleurs niveaux de qualité́ de service et de disponibilité́ sont maintenus, seront des atouts pour chaque entreprise.

L’objectif de notre travail est de concevoir et mettre en œuvre une application web de suivi du projet, du courrier et d’archivage numérique au sein **CARPA** dans le but de résoudre les problèmes liés au suivi des projets et à la gestion, le suivi et l’archivage du courrier.

Le présent mémoire s’articule autour de cinq chapitres donc le premier est intitulé contexte et problématique dans lequel nous allons vous présenter le **CARPA** ou nous avons effectué́ notre stage et vous présenter le contexte dans lequel nous avons ressortis la problématique, le second chapitre intitulé les généralités qui présenteront quelques techniques et outils utilisés, le troisième chapitre intitulé analyse et conception consistera à établir les différents besoins de l’application aussi fonctionnels que non fonctionnels, et de faire des modélisations pour mieux comprendre les scenarios de notre application, le quatrième intitulé implémentation présentera la mise en œuvre de la plateforme proprement dite, elle consiste en l’établissement des meilleures méthodes, technologies et outils nécessaires au développement de l’application, ainsi que les dis- positions architecturales de cette dernière. Enfin le dernier chapitre portera sur les résultats et commentaires qui consistera à présenter les résultats obtenus.

|  |  |
| --- | --- |
| : CONTEXTE ET PROBLEMATIQUE |  |

|  |  |
| --- | --- |
| : GENERALITES |  |

## Introduction

Dans l’accomplissement d’un travail scientifique, on s’appuie « toujours » sur des résultats d’autres recherches. Il devient donc très indispensable, dans un tel projet, de présenter les outils et techniques qui ont été́ utilisés pour parvenir à une bonne fin.

Dans ce chapitre des généralités, nous allons présenter la technique Rapid Application Développent, puis l’outil Ruby On Rails qui la met en valeur. Dans un souci de respect des principes du génie logiciel, pour éviter des erreurs de régression et pour mieux gérer les versions, nous utiliserons l’outil Git que nous présentons également dans ce chapitre. L’architecture MVC, le langage UML et la méthode de développement Agile feront l’objet des dernières présentations.

## Rapid Application Development

Le Rapid Application Development (RAD), francisé en « Robot Automatique de Développement », est une méthode de développement rapide de logiciels. Il se fait généralement au moyen des outils tels que des AGL (Atelier de Génie Logiciel) ou des L4G (Langages de Quatrième Génération).

Partant du résultat de la modélisation, ils doivent pouvoir générer une application avec des fonctionnalités basiques telles que l’ajout, la modification et la suppression d’une entité́ ; également l’affichage des données par liste. On parle généralement de CRUD (Create, Read, Update, Delete).

Ces outils sont des ensembles de programmes permettant la conception de programmes, d’applications ou de systèmes parfois très complexes. Ils sont généralement formés par des langages puissants et évolués, accompagnés d’utilitaires de création d’interfaces graphiques des programmes générés. Exemple : Ruby On Rails, Sinatra, Visual Basic, Django.

Avec un L4G, on programme vite et c’est simple, mais le code généré́ est souvent lourd et très lent, sans véritable optimisation. De plus, on n’a que rarement accès aux entrailles de son programme, et s’il ne fonctionne pas, on peut mettre beaucoup de temps à diagnostiquer.

Il est donc nécessaire pour le programmeur de trouver l’outil qui répondra à ses attentes. Un outil qui soit compatible au langage utilisé, qui soit le plus « open source » possible et qui procure une architecture familière. Ruby On Rails et Sinatra sont des outils destinés à la plateforme Ruby et qui utilisent des architectures en MVC. Ruby On Rails a particulièrement attiré notre attention.

## L’outil Ruby On Rails

Dans le monde Ruby, ROR, c’est-à-dire Ruby On Rails, apporte une réponse à la concurrence des fra- meworks de développement rapide tels que Groovy on rails ou Django. Il permet de créer rapidement des applications WEB. Il a été́ initié en 2005 par David Heinemeier Hansson c’est un Framework web open source écrit en Ruby sous licence MIT. Il suit le modèle -vue-contrôleur(MVC). Il propose une structure qui permet de développer rapidement et intuitivement (reference : wikipedia.org/wiki/RubyOnRails). Il s’installe d’une manière très particulière notamment en ligne de commande après avoir installé́ Ruby. On peut l’utiliser en ligne de commande ou via des IDE qui l’intègrent. Sa commande principale est **Rails**. L’exécution de cette commande dans une invite de commandes Mac nous donne l’aperçu suivant :

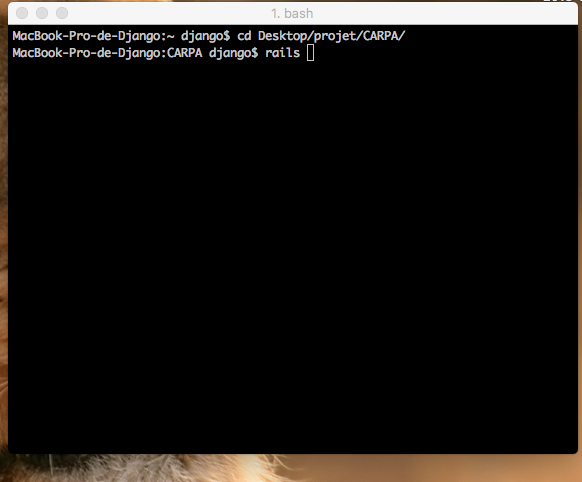


Figure 1 Terminal

* **Les modèles** sont des classes servant à modéliser les données et à établir les relations entre elles. Cela permet d’établir le mapping entre les Objets et la base de données, grâce aux outils tels que **Active Record**. Il est possible d’effectuer des requêtes de base sur les modèles. On trouve sur chaque modèles des méthodes save (), create (), find\_by\_name(),find() etc... Ces méthodes n’ont jamais été́ définies par le développeur, mais elles existent grâce au framework. Il est également possible de contrôler la validation des formulaires depuis le fichier qui définit le domaine.
* **Les Contrôleurs**. Un contrôleur est une classe qui reçoit la requête de l’utilisateur et qui, en fonction de l’action demandée, va effectuer le traitement. Pour gérer quelle est l’action et quel contrôleur est appelé́, Rails se base sur le formatage de l’URL : **: http ://<. . . >/controller/action/**.
* **Les Vues** sont représentées par un moteur de template **erb**, ou du **Haml**, on peut insérer du code Ruby. On constate dans l’arborescence qu’il y’a une vue dédiée à chaque contrôleur et une vue dédiée à une action du contrôleur (Voir Annexe).

## Le langage Ruby

Le langage Ruby a été́ conçu, au milieu des années 90, par Yukihiro Matsumoto, un programmeur Japonais. Son objectif était d’avoir un langage qui soit « plaisant » à utiliser : Ruby is **“made for developer happiness” !** (reference).C’est un Langage Orienté Objet. La figure ci-dessous présente un arbre généalogique de Ruby. Quant au tableau, il représente les ancêtre de Ruby, avec les principales caractéristiques héritées de ces ancêtres.

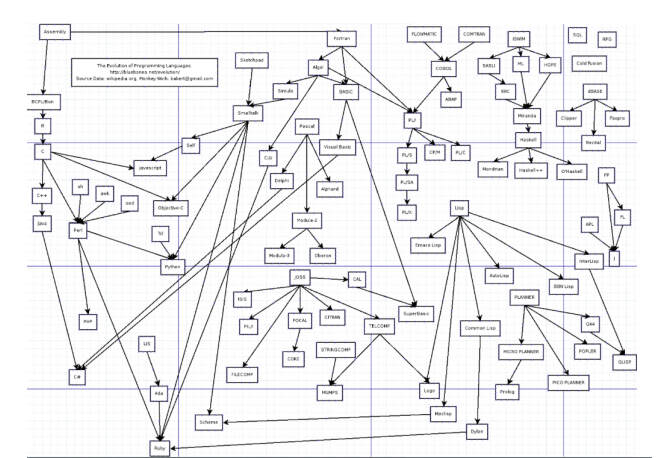


Figure 2 Arbre généalogique de Ruby

|  |  |  |
| --- | --- | --- |
| Langage | Année | Caractéristiques |
| LISP | 1958 | Approche fonctionnelle métaprogrammation |
| CLU | 1974 | Littérateurs |
| Smalltalk | 1980 | Langage objet pur, blocs de code GUI, sUnit |
| Eiffel | 1986 | Uniform Access Principle |
| Perl | 1987 | Expressions régulières et pattern matching |
| Ruby | 1993 |  |

Tableau 1 Les ancêtres de Ruby

La syntaxe de Ruby est faite pour apporter plus de flexibilité́ au framework Ruby On Rails.

## Architecture et création d’un projet avec Ruby On Rails

La création d’un projet ROR se fait en trois étapes :

* Création des classes et modèles ;
* Scaffolding ;
* Running

### Création des classes et modèles

Les « migrations » sont définitions du modèle. C’est la première étape dans la création d’un projet Rails. Nous avons utilisé dans ce projet la version **5.1.6.** La commande rails new nomProjet permet de créer un projet nommé nomProjet. Rails va construire un nomProjet avec un contenu tel que représenté́ par la figure ci-après.