L3 CMI Informatique

HLIN601 – TERL3

CourtCircuit

RAPPORT #2

Bachar Rima
Othmane Farajallah
Wissem Soussi

Responsable CMI Informatique :
Anne-Elisabeth Baert

Encadrant :
Eric Bourreau

5 mai 2018

Table des matières

1 Introduction **1**
1.1 Contexte du projet 1
1.2 Présentation du LIRMM 2

2 Problème, Méthodologie, et Outils **3**
2.1 Problème 3
2.2 Méthodologie 3
2.3 Outils de conception 4
2.4 Outils d'implémentation 4

3 Implémentation et Résultats **7**
3.1 Application web monopage 7
3.2 Front-end 7
3.2.1 React.js 7
3.2.2 Bootstrap et Font-Awesome 7
3.2.3 Charte graphique 8
3.3 Back-end 9
3.3.1 Node.js 9
3.3.2 Express 10
3.3.3 MariaBD 10
3.4 Résultats 12

4 Conclusion **13**
4.1 Difficultés survenues 13
4.2 Perspectives 13
4.3 Apports personnels 14

Annexes **15**

Bibliographie **16**

Chapitre 1

Introduction

1.1 Contexte du projet

Dans ce rapport, nous nous consacrons à la description détaillée de la phase d'implémentation du projet intitulé `CourCircuit`[1] à effectuer au sein du **LIRMM** (**L**aboratoire d'**I**nformatique, de **R**obotique et de **M**icroélectronique de **M**ontpellier) dans le cadre du module **HLIN601** – **TER** de la 3e année de licence en informatique **CMI** (**C**ursus **M**aster **I**ngénierie).

Le projet se déroule sous l'encadrement de Mme Anne-Elisabeth Baert, enseignante/chercheuse au sein du LIRMM dans l'équipe **MAORE** (**M**éthodes **A**lgorithmes pour l'**O**rdonnancement et les **Ré**seaux), en tant que responsable de la formation CMI informatique et M. Eric Bourreau, enseignant/chercheur au sein du LIRMM dans la même équipe **MAORE**, en tant que responsable pédagogique et encadrant du projet.

Le sujet du projet consiste en la création d'un site web représentant une interface de communication entre fournisseurs de produits locaux et leurs clients. Il est inspiré du site « La Ruche Qui Dit Oui »traitant le même thème et répondant aux mêmes besoins, cependant adoptant une approche différente, surtout au niveau de la logistique et de l'architecture du site, afin de fournir une vision différente, voire plus optimisée de la gestion des interactions directes entre clients et fournisseurs.

Nous commencerons ce rapport en faisant un récapitulatif du contexte du projet, des problématiques traitées, et de la méthodologie et des outils de conception adoptés. De plus, nous proposerons une courte description des outils d'implémentation choisis pour la mise en œuvre. Après, nous procéderons à la description détaillée des outils d'implémentation et à la présentation des résultats obtenus. Enfin, nous conclurons en discutant les difficultés sur-

1. précédemment nommé **LaRuche**

venues lors du développement et les perspectives du projet.

1.2 Présentation du LIRMM

« Le [...] – LIRMM – est une unité mixte de recherche, dépendant conjointement de l'Université Montpellier et du Centre National de la Recherche Scientifique [(CNRS)]. Il est situé sur le Campus Saint-Priest de l'UM [(Figure 1.1)].

FIGURE 1.1 – bâtiment 3 du LIRMM, Campus St. Priest

Les travaux sont menés dans trois départements scientifiques de recherche, [(L'Informatique, La Robotique, et La Microélectronique)] eux-mêmes organisés en « équipes-projet ».

Les recherches menées au LIRMM trouvent généralement une finalisation dans des domaines applicatifs aussi divers que la biologie, la chimie, les télécommunications, la santé, l'environnement... et dans les domaines propres du laboratoire : l'informatique, l'électronique et l'automatique.

Ses activités de recherche [le] positionnent [...] pleinement au coeur des sciences et technologies de l'information, de la communication et des systèmes. [En particulier,] les thématiques du département Informatique s'étendent des frontières des mathématiques à la recherche appliquée : algorithmique des graphes, bioinformatique, cryptographie, réseaux, bases de données et systèmes d'information [...], génie logiciel [...], intelligence artificielle [...], interaction homme-machine [...]. » (LIRMM n.d.)

Chapitre 2

Problème, Méthodologie, et Outils

2.1 Problème

Les consommateurs cherchent de *plus en plus* à acheter des produits frais minimisant les étapes de *processing*, alors que les producteurs cherchent à se libérer des centres d'achat et des intermédiaires de distribution.

Dans l'esprit du site francais LaRucheQuiDitOui, on souhaite implémenter une interface sous forme d'un **site web** d'*e-commerce* permettant aux producteurs de vendre des sélections diversifiées de produits aux consommateurs en se regroupant dans des endroits précis de collecte.

Le site web offrira ainsi **tout ce dont il est nécessaire** aux consommateurs pour effectuer des commandes prépayées et précisera ensuite les points de collecte de produits les plus proches. D'autre part, il mettra à la disposition des fournisseurs la possibilité d'organiser ces points, de gérer la mise à jour des stocks et la mise en vente/prise de commandes par les clients, en se basant sur des algorithmes d'optimisation se portant aussi bien sur la gestion de la logistique, la préparation/facturation des commandes, que sur la redistribution/partage des produits entre les différents fournisseurs voisins.

2.2 Méthodologie

Dans le but d'assurer une bonne gestion des ressources et d'offrir le plus de fonctionnalités possibles aux utilisateurs, en les implémentant itérativement dans des versions fonctionnelles et testées du site, nous avons opté pour une approche basée sur les **méthodes agiles** de développement, en particulier

sur **XP** [1].

En effet, la méthodologie de développement proposée par les méthodes agiles étant de plus en plus prépondérante en génie logiciel, nous avons décidé de l'utiliser pour la modélisation et l'implémentation de notre site web afin d'assurer le plus d'extensibilité et de flexibilité possibles.

2.3 Outils de conception

Pour garantir une bonne modélisation du projet, en cohérence avec l'approche de la méthodologie discutée précédemment, nous avons explicité les spécifications fonctionnelles et organisationnelles de notre projet en se servant des outils suivants :

user stories des requis fournis par les utilisateurs décrivant en langage naturel les fonctionnalités qu'ils souhaitent avoir dans le site.

diagrammes de cas d'usage des diagrammes dynamiques, souvent utilisés en **UML** pour décrire en haut niveau les fonctionnalités d'un système, en se servant de notions telles que **acteurs**, **cas d'usage**, **systèmes** et les **relations** entre chacune de ces entités.

modèle EA un modèle **conceptuel** utilisé pour décrire les entités du projet ainsi que les associations décrivant leurs relations et comportements.

schéma de base de données schéma en modèle **relationnel** composé des schémas des relations et des contraintes d'intégrité sur l'ensemble des relations, traduit généralement à partir du **modèle EA** et servant comme **modèle logique** lors de l'implémentation de la **base de données**.

mockup storyboard document de haut niveau offrant un moyen pour schématiser l'utilisation d'un projet, en positionant les différents éléments le composant, sans rentrer dans les détails de leur fonctionnement.

2.4 Outils d'implémentation

Afin de réaliser les impératifs de la partie conception du projet, nous avons prévu, pendant la partie de modélisation, d'utiliser principalement *React.js*, *JQuery*, *Boostrap*, *PHP* et *MySQL*, ainsi que d'autres **API** pour l'implémentation de certaines parties du projet. Toutefois, après avoir compléter

1. *eXtreme Programming*

notre soutenance et suite à des recherches supplémentaires, nous avons décidé d'abandonner quelques technologies et de les remplacer par d'autres plus pertinentes afin d'obtenir un écosystème de travail plus cohérent et plus efficace.

Ce choix fût alimenté par plusieurs facteurs que nous discuterons plus en détail dans le chapitre 3 de ce rapport. Pour cette section, nous nous contentons simplement de fournir une courte description de l'ensemble de l'écosystème qui, en l'occurrence, est composé de :

Front-end

React.js : une bibliothèque **JavaScript** créée par *Facebook* et sortie en 2013 pour la création des interfaces graphiques.

JSX : (*JavaScript XML*) une extension syntaxique du **JavaScript** permettant de simplifier la création des composants *React* à travers une syntaxe à la **XML**.

Bootstrap : un *framework* **CSS** permettant la gestion du *responsive web* du site.

Back-end

Node.js : un environnement d'exécution **JavaScript** côté serveur gratuit et *open-source* utilisant le moteur **JavaScript V8** de *Google Chrome*.

Express : un framework minimaliste gratuit et open-source, devenu standard pour construire des applications web et des **API**s côte serveur sur **Node.js**.

MariaDB : un serveur de base de données gratuit et open-source remplaçant **MySQL**[2], introduit par les créateurs de **MySQL** suite à l'achat de ce dernier par **Oracle**.

Editeur de texte

Pour la rédaction du code de notre projet, nous avons choisi unanimement d'utiliser l'éditeur de texte gratuit, open-source et multiplateforme **Atom** pour son architecture basée sur **Node.js**. En effet, cette architecture permet de personnaliser son utilisation et de le rendre plus souple via des modules téléchargeables par le biais de son gestionnaire de packages **APM** (*Atom Package Manager*).

2. en restant compatible avec **MySQL**

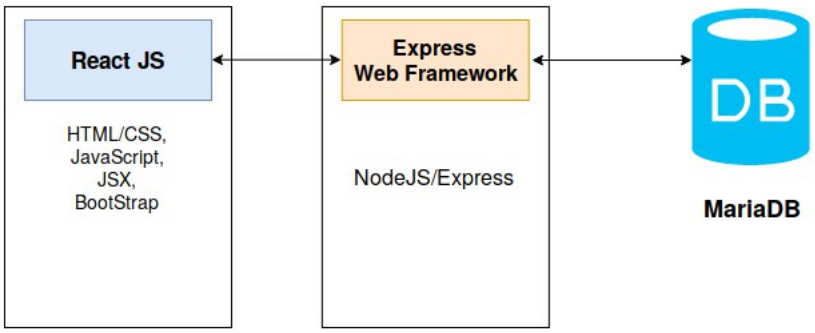

FIGURE 2.1 – L'architecture des technologies utilisées en CourtCircuit

Par conséquent, l'architecture des technologies choisies pour l'implémentation de notre projet est illustrée dans la figure 2.1.

Chapitre 3

Implémentation et Résultats

3.1 Application web monopage

Une application web est dite monopage quand elle est concu pour ne pas avoir plusieurs fichiers HTML relatives aux differentes pages du site, mais plutot une unique page HTML avec les rélatives ressources Javascript et CSS permettant d'afficher les differentes pages dinamiquement. Avec un site web classique le navigateur affiche simplement les pages envoyées par le serveur et lui transmet les actions de l'utilisateur. Le navigateur prend donc un rôle plus important que dans un site web classique, puisqu'il ce charge de toute la logique applicative, en reagissant aux actions de l'utilisateur. Le serveur devient donc responsable que des ressource a fournir en effectuant de son coté que les verifications de securité necessaire avant qu'il n'envoie quelque ce soit. C'est pour cela que la page web unique deviens une veritable application, qui tourne coté client pour fluidifier l'experience de l'utilisateur : en evite de charger a chaque requette de l'utilisateur une page entiere, et on change la page en fonction du serveur partiellement.

3.2 Front-end

3.2.1 React.js

3.2.2 Bootstrap et Font-Awesome

Bootstrap est un framework pour le developpement front-end, donc coté client d'un site web. Il est pricipalement un framework CSS mais embarque aussi des composants Javascript et HTML. En utilisant un systeme de grille pour gerer le Layout d'une page Bootstrap est capable de changer dynami-

quement ce Layout par rapport à la taille de la fenetre du navigateur permettant d'avoir un site responsive utilisable avec praticité soit sur les petits ecrans tel que smartphone ou tablet, comme sur les grands comme un moniteur ou un téléviseur. Grace à l'ensemble de classes css, Bootstrap permet un developpement du site-web rapide tout en permettant de personnaliser le design en ajoutant peut de lignes de code. On a par exemple des boutons, des formulaires, des navbars, des accordilons, des carousels, tous esthetiques et personalisables avec l'ajout d'animations javascript qui est de base fait avec JQuery. Sur notre site on a donc utilisé Bootstrap pour tout ce qu'on a

FIGURE 3.1 – navbar du site fait avec bootstrap

mentionné, en evitant de reinventer la roue et definir l'eshtetique de chaque element HTML à partir de zerò. **Font-awesome** a été crée pour etre utilisé avec Bootstrap et c'est un outils qui permets d'utiliser les icons comme des polices d'écritures qui peuvent subir des changement de style avec CSS, SASS ou LESS. Avec ces deux frameworks, chaque composante REACT a été esthétiquement personnalisée, nottament :

— L'entete du site, qui est un Navbar (barre de navigation) responsive et "collapsable" pour les écran de petite taille ;
— Le footer du site, en utilisant des icones Font-awesome pour les liens vers d'autres pages.

3.2.3 Charte graphique

Pour le site web, etant principalement un site de commerce de fruits et legumes, nous avons pensé d'utiliser des couleurs comme le vert et l'orange, qui donnent aux images des produits à vendre une embience de nature et de fraicheur. De plus les fonts, couleurs et logos sont sombres et simples.

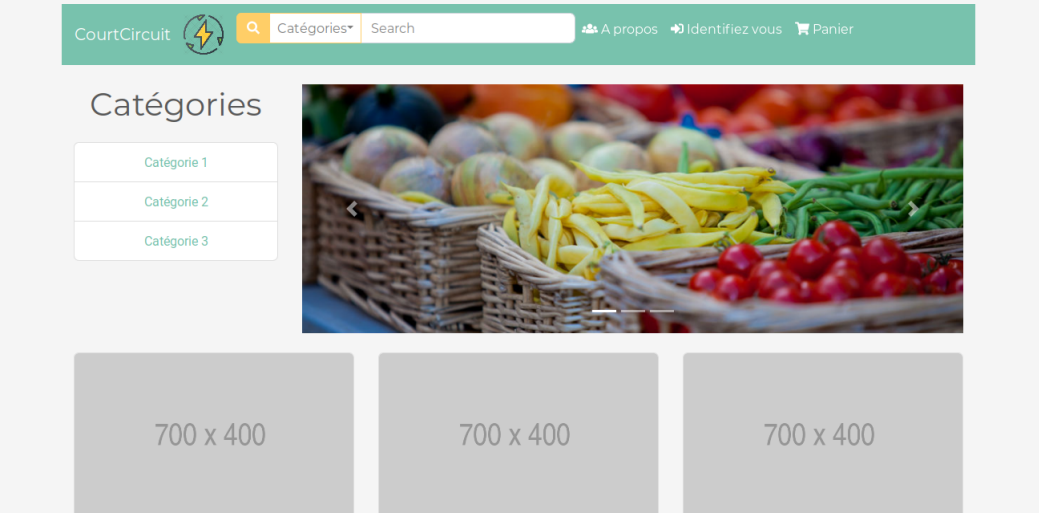

FIGURE 3.2 – Page d'accueil du site web

3.3 Back-end

Pour la partie back-end, nous avons décidé d'utiliser **Node.js**/**Express.js** pour le serveur et **MariaDB** pour la base de données. Nous détaillerons les raisons et les résultats d'utilisation

3.3.1 Node.js

FIGURE 3.3 – le logo de **Node.js**

Node.js est un environnement d'exécution **JavaScript** côté serveur utilisant le moteur **JavaScript V8** de *Google Chrome*. Il est libre, gratuit et doté d'une efficacité et d'une légèreté provenant de sa modélisation événementielle, monothread et non-bloquante et de son architecture modulaire. Sa facilité d'usage et son extensibilité est dûe aussi à son gestionnaire de paquets **NPM** (*Node Package Manager*) considéré comme l'un des plus larges écosystèmes de librairies open-source au monde.

Node.js est une technologie récente qui est de plus en plus utilisée, surtout en développement web avec les architectures **MERN** (*MongoDB Express React Node*) et **MEAN** (*MongoDB Express Angular Node*). Par suite, ceci nous a motivé à l'intégrer dans notre projet au vu de ses nombreuses avantages.

En premier, notre choix de **Node.js** découle du fait qu'il permet d'écrire du code **JavaScript** du côté serveur. Cette particularité nous a permis d'unifier la rédaction du code en un seul langage, pour les côtés client et serveur. En effet, l'unification du code a rendu possible à l'ensemble du groupe de participer au débogage du projet entier sans aucune restriction liée à la maîtrise d'un langage ou l'autre.

Outre l'unification du code, étant donné que nous aurons besoin de gérer un ensemble important de données dans le cadre de notre projet, l'aspect monothread et non-bloquant de **Node.js** assure une performance fluide que nous ne trouverons pas forcément avec d'autres technologies back-end, utilisant généralement des modèles multi-processus ou multithread.

Enfin, **Node.js** est le choix par défaut pour une architecture utilisant **React.js**[1], ce qui nous a encouragé davantage à s'en servir.

Dans le cadre de notre projet, nous avons principalement utilisé **Node.js** pour la création d'une **API** côté serveur permettant d'interfacer avec la base de données. Ceci fût concrétisé par la création des objets pour les différentes entités manipulés (*Client, Utilisateur, Produit,* ...), en s'appuyant sur des concepts de la programmation orienté-objet. Par conséquent, nous avons pu factoriser le code pour éviter les redondances et améliorer sa lisibilité lors de l'exécution des requêtes sur la base de données.

D'autre part, nous nous sommes également servi de **Node.js** pour l'utilisation du framework **Express** que nous détaillerons dans la section suivante.

3.3.2 Express

3.3.3 MariaBD

1. bien que pas le seul

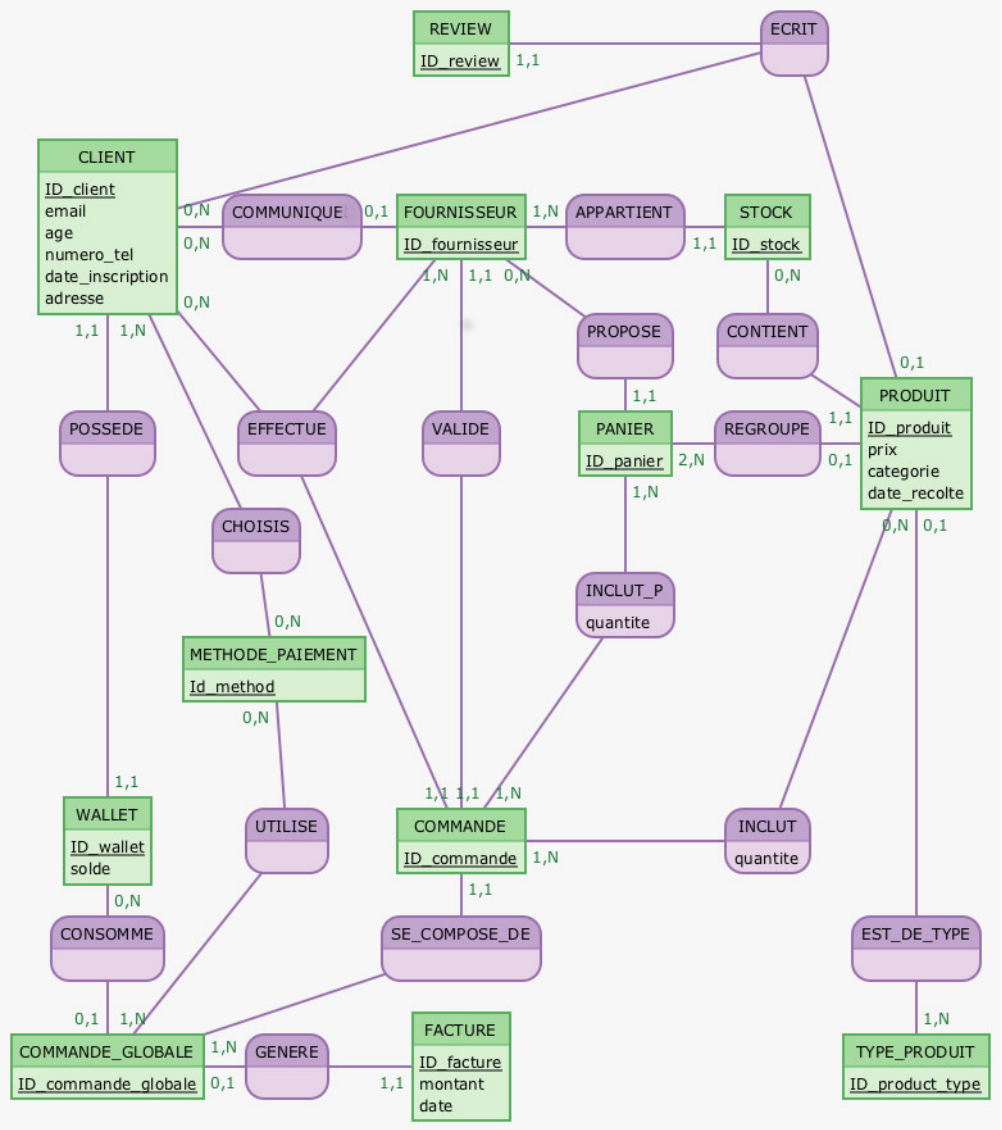

FIGURE 3.4 – Modèle Entité-Association

Le modèle relationnel obtenu par traduction du **modèle EA** des données est le suivant :

1. `REVIEW` (ID_review, #*ID_client*, #*ID_produit*)
2. `CLIENT` (ID_client, *email*, *age*, *numero_tel*, *date_inscription*, *adresse*)
3. `FOURNISSEUR` (ID_fournisseur, #*ID_commande*, #*ID_client*)
4. `STOCK` (ID_stock, #*ID_fournisseur*)
5. `PANIER` (ID_panier, #*ID_fournisseur*)

6. `PRODUIT` (ID_produit, *prix*, *categorie*, *date_recolte*, #*ID_stock*, #*ID_panier*, #*ID_product_type*)
7. `CHOISIS` (#*ID_client*, #*Id_method*)
8. `INCLUT_P` (#*ID_commande*, #*ID_panier*, *quantite*)
9. `METHODE_PAIEMENT` (Id_method)
10. `WALLET` (ID_wallet, *solde*, #*ID_client*)
11. `UTILISE` (#*ID_commande_globale*, #*Id_method*)
12. `COMMANDE` (ID_commande, #*ID_commande_globale*, #*ID_client*, #*ID_fournisseur*)
13. `INCLUT` (#*ID_commande*, #*ID_produit*, *quantite*)
14. `COMMANDE_GLOBALE` (ID_commande_globale, #*ID_wallet*)
15. `FACTURE` (ID_facture, *montant*, *date*, #*ID_commande_globale*)
16. `TYPE_PRODUIT` (ID_product_type)

3.4 Résultats

Chapitre 4

Conclusion

4.1 Difficultés survenues

On a rencontré des nombreux difficultés pendant le développement du projet, notamment sur la maîtrise de nouveaux outils jamais utilisés auparavant : React, JSX, Express etc.. On a eu le sentiment d'avoir passé trop de temps dans la partie de la modélisation et conception en se retrouvant avec 3 mois (qui ne sont pas entièrement dédiés non plus) pour la réalisation du site. De plus, on a vu des concepts théoriques nouveaux comme le design pattern MVC, le concept d'une application web monopage, et tous les outils conventionnellement utilisés pour des projets javascript : Babel, Npm, Node, React, Jsx etc ... La compréhension et la pris en main de ces outils ont eu une partie importante du projet avant de commencer à les utiliser dans la pratique. Une autre limitation a été le serveur d'hébergement du département informatique, qui a mis en place des environnements node et npm non mis a jours, et qui nous n'a pas cree une base de données dediee a notre projet comme c'etait prevu sur le site SIF. En manque des permissions necessaires, nous n'avons pas reussi a le faire nous-meme non plus, bien que nous avions signale ceci au parties concernees.

4.2 Perspectives

Outre les fonctionnalités que nous avons jugées essentielles au projet, nous avons pensé à ajouter des fonctionnalités supplémentaires une fois celles de base établies. En effet, plusieurs extensions sont possibles afin de proposer aux utilisateurs un choix plus diversifié et plus complet. Dans ce cadre, nous avons prévu de donner aux clients la possibilité de régler des achats à l'aide d'une carte bancaire mais également à l'aide de services de paiements en ligne

tels que Paypal. En addition, nous pourrions également internationaliser le site et son concept en proposant des versions en plusieurs. Le projet nous interesse et nous avons envie de le continuer comme projet personnel au delà du parcours d'études.

4.3 Apports personnels

Les difficultés qu'on a eu pendant le projet sont les éléments qui nous a fait apprendre le plus, la nouveauté des outils et des concept, ainsi que les convention de la programmation web, qui vont être utiles pour notre parcours future, que ce soit dans une continuation d'études ou professionnel.

Annexes

Bibliographie

LIRMM (n.d.), 'Présentation'. Last updated : 2016-02-17.
URL: *https ://www.lirmm.fr/le-lirmm/presentation*