Plast’Prod

Projet Komodo

Spécifications techniques livrable 2 Version 0.1

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Modifié | Par | Action |
| 0.1 | 18/09/2012 | Cora Cox | Initialisation |
| 0.2 | 19/09/2012 | Sébastien Lavie | Livrable 1 & 2 |

[1. MCD et traitements 4](#_Toc190606034)

[1.1 MCD 5](#_Toc190606035)

[1.2 Traitements 6](#_Toc190606036)

[2. Architecture technique 7](#_Toc190606037)

[2.1 Architecture applicative 8](#_Toc190606038)

[2.2 Architecture logicielle 9](#_Toc190606039)

[2.3 Architecture matérielle 10](#_Toc190606040)

[2.4 Environnement de développement 11](#_Toc190606041)

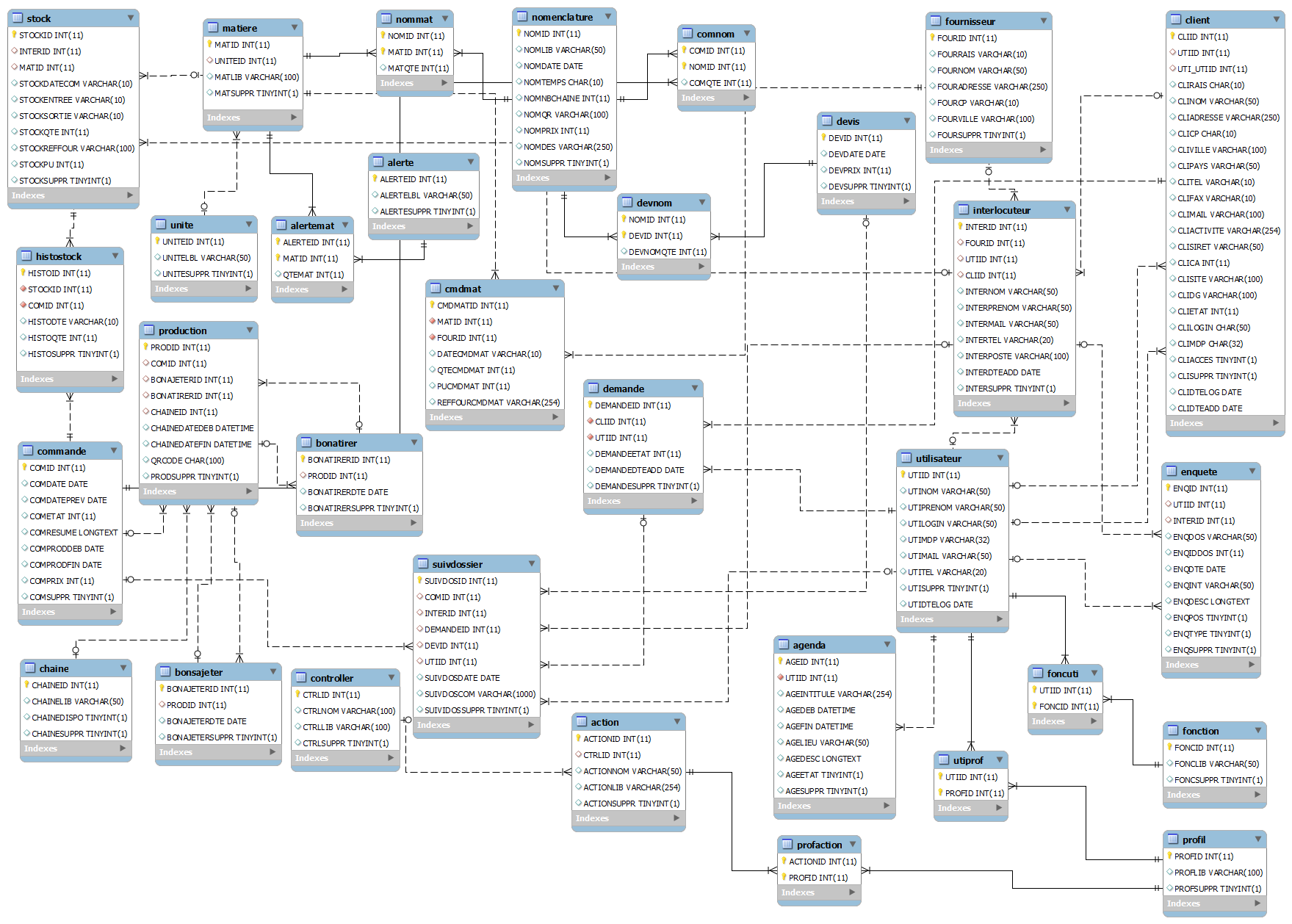
[3. Annexes 12](#_Toc190606042)

[3.1 Annexe 1 13](#_Toc190606043)

# MCD et MOO

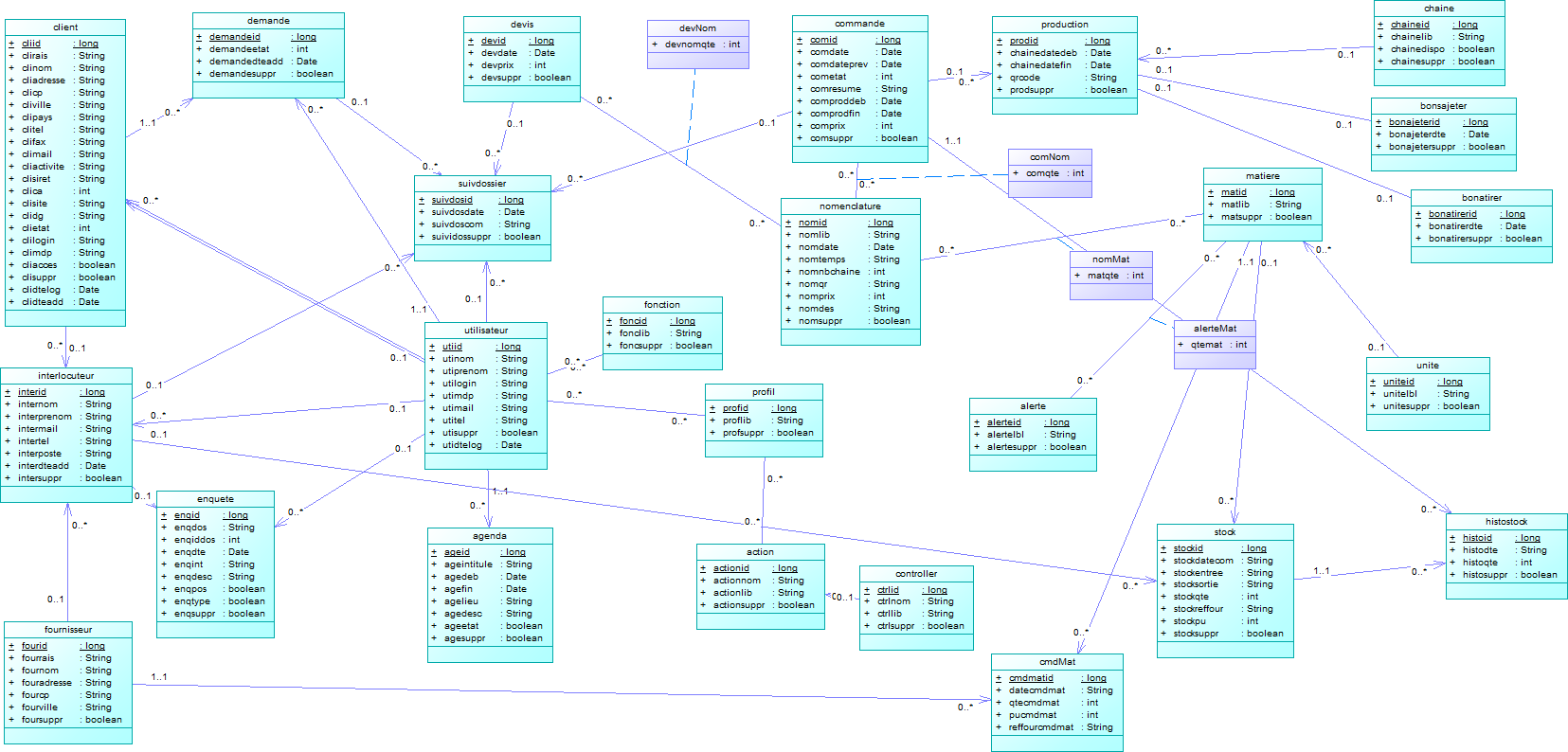
## 

## MODELE CONCEPTUEL DE DONNEES



## 

## MODELE ORIENTE OBJET



# 

# Architecture technique

## Architecture applicative

APPLICATION WEB :

L'application est réalisée en utilisant le modèle **M.V.C (Modèle, Vue et Controller)**.

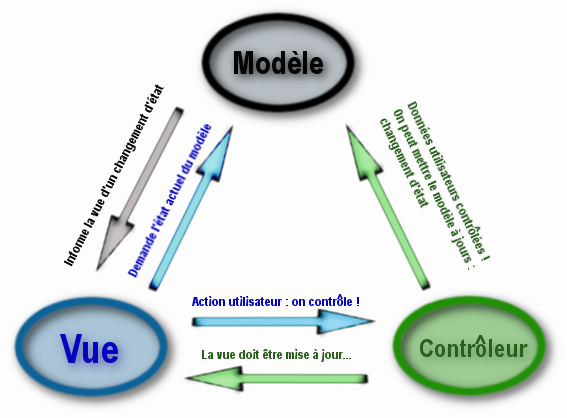
**Définition du modèle M.V.C :**

Modèle d'architecture qui cherche à séparer nettement les couches de présentation (UI : User Interface), métier (BLL : Business Logic Layer) et d'accès aux données (DAL : Data Access Layer).

Le but étant d'avoir une dépendance minimale entre les différentes couches de l'application. Ainsi les modifications effectuées sur n'importe quelle couche de l’application n'affectent pas les autres couches.

* Modèle : Encapsule le cœur fonctionnel de l'application, le domaine logique.
* Vue : Les données sont envoyées, par le modèle, à la vue qui les présente à l’utilisateur.
* Contrôleur : Reçoit les données et les transmets au modèle ou à la vue.

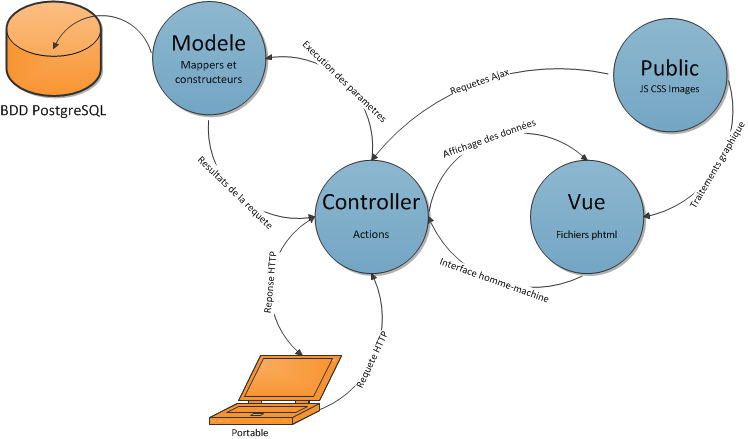
*Schéma graphique du modèle M.V.C :*



Mise en pratique du modèle M.V.C avec le Framework Zend :

* Le client fait une demande au contrôleur (actions). Ce contrôleur voit passer toutes les demandes des clients. C'est la porte d'entrée de l'application. C'est le C de MVC.
* le contrôleur traite cette demande. Pour ce faire, il peut avoir besoin de l'aide de la couche métier (mapper pour la liaison à la base de données et ACL pour la gestion des droits), ce qu'on appelle le modèle M dans la structure MVC.
* le contrôleur reçoit une réponse de la couche métier. La demande du client a été traitée. Celle-ci peut appeler plusieurs réponses possibles. Un exemple classique est :
  + une page d'erreurs si la demande n'a pu être traitée correctement
  + une page de confirmation, sinon :
* le contrôleur choisit la réponse (= vue fichier .phtml) à envoyer au client. Celle-ci est le plus souvent une page contenant des éléments dynamiques. Le contrôleur fournit ceux-ci à la vue.
* la vue est envoyée au client. C'est le V de MVC.

*Représentation graphique du modèle MVC avec le Framework Zend :*



## Architecture logicielle

### Briques utilisÉes

#### Modules standards

PDO PostgreSQL pour la connexion à la base de données

#### Modules développés sur mesure

ACL (Access Control List) pour gérer les accès des utilisateurs

Module "HELP" (php & javascript)

#### Framework additionnel (GUI, effets graphiques…)

Zend Framework (PHP)

Zend PDF (PHP)

PHP QR Code (PHP)

Table Filter (JavaScript)

EasyPaginate (JavaScript)

JQuery (JavaScript)

### Plateforme

#### Plateforme dédiée à l’intégration continue

JBoss

#### Base de données

PostgreSQL

#### Serveur http

Apache2

## Architecture matérielle

### MatÉriel

#### Serveurs

X

#### Switch

X

#### …

### OS

Version

Modules, extensions…

## Environnement de développement

### Convention de nommage

#### Nommage des programmes

X

#### Nommage des fichiers

X

### Gestion des versions

#### Application web :

Gestion des versions avec SVN installer sur un serveur virtualisé qui a pour O.S Debian.

#### Client lourd :

Gestion des versions avec GitHub en ligne.

### RÈgles de codage

#### Factorisation

X

#### Accès aux bases

X

#### JavaScript

X

#### ...

X

# Annexes

## Annexe 1