|  |
| --- |
| **Module IO** |
| Document Technique |
|  |
| Version 1.0 |
|  |
| **AUGUEY Thomas** |
| **05/04/2013** |
|  |



Contenu

[Introduction 2](#_Toc348617853)

[Termes 3](#_Toc348617854)

[Environnement logiciel 4](#_Toc348617855)

[Arborescence 6](#_Toc348617856)

[Modèle de données 7](#_Toc348617857)

[Entités 7](#_Toc348617858)

[Formats 7](#_Toc348617859)

[Operations système 8](#_Toc348617860)

[Acteurs 8](#_Toc348617861)

[Administrateur 8](#_Toc348617862)

[Système 8](#_Toc348617863)

[Résultats d’opération 9](#_Toc348617864)

[Détail des codes 9](#_Toc348617865)

[Détail des dialogues 9](#_Toc348617866)

[Cas d’utilisation 10](#_Toc348617867)

[Envoyer un message 11](#_Toc348617868)

[Développement 12](#_Toc348617869)

[Point d’entrée de l’application 12](#_Toc348617870)

[Modèle MVC 12](#_Toc348617871)

[API Références 12](#_Toc348617872)

[Modèle objet et Classes Managers 12](#_Toc348617873)

[Paramètres de configuration 13](#_Toc348617874)

[Notes d’implémentations 14](#_Toc348617875)

[Dépannage 15](#_Toc348617876)

[Problème de connexion au server SMTP 15](#_Toc348617877)

# Introduction

**Module IO** est utilisé pour la gestion des échanges de données (upload/download).

Fonctionnalités :

* Envoi (Upload) / Téléchargement (Download) de données par fragment

# Termes

Webframework

Projet principal de ce module.

# Environnement logiciel

Webframework

Ce module est développé pour le projet **Webframework**.

Votre application web doit au minimum implémenter l’interface IApplication et définir la variable globale $app pour fonctionner normalement (voir le modèle **/wfw/minimal** du projet **Webframework**).

PHP

Ce module est programmé avec **PHP version 5.4.3**.

Les modules suivants doivent être disponibles :

* **php\_fileinfo**
* **php\_pgsql**

Base de données

Le **système de base de données** utilisé est de type **PostgreSQL 8.3**.

Environnement de développement intégré (IDE)

Le module est développé sous l’environnement **NetBeans 7.2.1** avec les langages **PHP/Javascript/XHTML et SQL.**

Suivit de version

L’Utilitaire **GIT** est utilisé pour le suivit de version. La branche master est utilisée pour contenir la dernière version stable du projet.

**Utile** : Le script archive.sh permet de créer une archive compressée du projet ne contenant que les fichiers importants. L’archive est stockée dans le dossier « **../archives »** et nommée automatiquement avec la version du projet en cours.

Modèle de données

La conception de la **base de données** et du **modèle orienté objet** à été réalisé en **UML** avec le logiciel **PowerAMC 15.1**.

**Important** :

Pour générer correctement le code **PHP depuis PowerAMC**, veuillez copier le fichier « documents/php.xol » dans le répertoire d’installation de votre programme **PowerAMC** (ex : C:\Program Files (x86)\Sybase\PowerAMC 15\Fichiers de ressources\Langages objet\php.xol).

Ce fichier permet de générer le modèle orienté objet et les classes **managers** permettant l’interaction avec la base de données.

# Arborescence

Voici une vue d’ensemble des dossiers du projet.

* DOCUMENTS : Documents divers relatif à ce projet
* MCD : Fichiers du Modèle conceptuel de données (**PowerAMC**)
* nbproject : Fichiers du projet **NetBeans**
* SQL : Fichiers SQL de création de la base de données
* wfw-XXX : Librairies et sources du projet à lier avec votre application

# Modèle de données

Le modèle des données persistantes est conçu sur le modèle Merise.

Le document « **mcd/io.mcd** » propose une version schématisé du modèle conceptuel de données, ce modèle est utilisé pour générer le script d’installation SQL. [Vous pouvez ouvrir ce fichier avec le logiciel **PowerAMC** de **Sybase**].

## Entités

Liste les tables ajoutées à la base de données.

Le tableau des entités est définit ainsi :

* Le **type** de donnée est écrit de façon générique (VARCHAR, CHAR, INTEGER, REAL, …)
* La **contrainte** est appliquée sur les champs par la base de données (Domaines, Clés, Check, …)
* Le **format**, définit plus bas, définit la syntaxe pour les chaines de caractères

MAIL\_MESSAGE

Stock les messages

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attribut | Description | Type | Contrainte | Format |
| MAIL\_MESSAGE\_ID | Identifiant | VARCHAR(64) | PRIMARY KEY | Identifier |

MAIL\_CONTACT

Stock les contacts

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attribut | Description | Type | Contrainte | Format |
| MAIL\_CONTACT\_ID | Identifiant | INT | PRIMARY KEY | Identifier |

## Formats

Tous les formats utilisés hérites du projet **Webframework**, reportez-vous au document technique lié au projet pour plus d’informations.

# Operations système

Les opérations systèmes sont conçues sur le modèle UML.

Le document « **mcd/operations.moo** » propose une version schématisé des cas d’opération. [Vous pouvez ouvrir ce fichier avec le logiciel **PowerAMC** de **Sybase**].

## Acteurs

### Administrateur

Super-Utilisateur de l’interface de gestion.

## Système

Définition du système avec ses différentes couches de dépendances.

**Système Langage**

1. Application HTML/Javascript
2. Interface Serveur PHP
   1. Système embarqué (système de fichier) Console UNIX
3. Interface de Base de données PG/SQL PL/pgSQL
   1. Base de données SQL SQL

## Résultats d’opération

Les résultats d’opération sont basés sur le modèle du projet **WebframeWork**, pour plus de détails à ce sujet reportez-vous au document technique concerné.

Notez que ces codes d’erreurs viennent s’ajouter aux codes existants du projet **Webframework**. Certains d’entre eux peuvent être réutilisés dans ce projet.

### Détail des codes

**Contexte Code Description**

ERR\_OK IO\_FILE\_UPLOADED Fichier reçu

ERR\_FAILED IO\_FILE\_TO\_BIG Le fichier excède la taille maximale autorisée

ERR\_FAILED IO\_ZERO\_FILE\_SIZE Le fichier possède une taille nulle

ERR\_SYSTEM IO\_CANT\_CREATE\_UPLOAD\_DIR Le dossier d’envoie n’existe pas et ne peut pas être créé

ERR\_SYSTEM IO\_DUMMY\_UPLOAD\_FILE\_CREATE Le fichier de réception ne peut pas être créé

ERR\_SYSTEM IO\_UPLOAD\_DIR\_NOT\_EXISTS Le dossier d’envoie n’existe pas

### Détail des messages

|  |  |  |
| --- | --- | --- |
| **Contexte** | **Code** | **Description** |
|  |  |  |
|  |  |  |

## Cas d’utilisation

Les cas d’utilisations (ou use cases) liste les procédures et leurs interactions avec le système.

Une procédure est soit validée, soit annulée dans sa totalité. Dans le cas contraire et pour maintenir un système cohérant, aucunes des actions précédentes n’est conservées.

Etat de l’implémentation

Cette liste informe sur l’avancement du développement. Un statut OK indique que le cas d’utilisation est rédigé et codé dans l’application, au contraire un statut KO indique un développement en cours

|  |  |  |  |
| --- | --- | --- | --- |
| **Cas d’utilisation** | **Statut** | **Rôle** | **Priorité** |
| mail\_send\_message | KO | Administrateur | Requis |
| mail\_create\_contact | KO | Administrateur | Optionnel |
| mail\_remove\_contact | KO | Administrateur | Optionnel |

### Débuter un envoie de données

Initialise un envoie de données.

Informations

Identifiant : create\_upload

Acteur : Visiteur

Entrée

Type Description

* Mail Adresse du nouvel utilisateur
* Identifier Nom d’utilisateur
* Password Mot de passe
* Name Identifiant client (module Client)

Sortie

Type Description

Procédure

1. Vérifie si l’utilisateur est connecté
2. Récupère le compte utilisateur associé à la connexion (USER\_ACCOUNT**)**
3. Vérifie si un profil existe déjà en base de données pour ce compte utilisateur (APP\_PROFIL\_EXISTS)
4. Initialise les entrées dans les tables IO\_UPLOAD\_PROCESS

### 

# Développement

## Point d’entrée de l’application

Conformément au projet **Webframework**, ce projet implémente l’interface IModule.

Vous trouverez le code de la classe MailModule dans le dossier « wfw-1.x/lib » avec l’ensemble des classes du modèle, vous devez inclure le fichier « lib/mail/mail.php » à votre application pour pouvoir utiliser les fonctionnalités du module **Mailing**.

## Modèle MVC

Le projet est construit sur le modèle <**Modèle-Vue-Contrôleur**>.

* Les classes C++ du modèles sont stockées dans le dossier « wfw-1.x/lib **»**.
* Les classes de vues sont stockées dans le dossier « wfw-1.x/view »
* Les classes contrôleurs sont stockées à la racine du dossier « wfw-1.x/ctrl**».**

## API Références

La documentation du code source est disponible dans le dossier « doc », il contient une arborescence web (HTML) généré à partir du code source.

Les fichiers de projet **Doxygen** est situé dans le dossier «  doxygen ».

## Modèle objet et Classes Managers

Le modèle de données a son équivalent en classes objets sous **PHP**.

## Paramètres de configuration

Voici les différentes constantes de configuration influant sur le module :

|  |  |  |  |
| --- | --- | --- | --- |
| [io\_module] | Requis | Définition | Exemple |
| ctrl\_path | **OUI** | Chemin d’accès aux contrôleurs | ${io\_module\_path}/ctrl/io |
| upload\_dir | **OUI** | Dossier de stockage temporaire des données (si mode=="file" seulement) | bin/public |
| public\_output\_dir | **OUI** | Dossier de stockage des fichiers publiques (envoies initiés avec le UseCase: begin\_upload) | bin/upload |
| max\_upload\_size | **OUI** | Taille maximal accepté pour les envois publiques (initiés avec le UseCase: begin\_upload)  [Spécifié en bytes] | 52428800 ; 50Mo |
| packet\_size | **OUI** | Taille des paquets accepté pour les envois publiques (initiés avec le UseCase: begin\_upload).  [Spécifié en bytes] | 524288 ; 512Ko |
| storage\_mode | **OUI** | Mode de stockage des données.  Si la valeur file est spécifiée le fichier est construit parallèlement dans le dossier local (upload\_dir) et la **BDD**. | file |

## Procédure de stockage

**Il existe 2 méthodes pour stocker un fichier sur le serveur :**

1. Avec un accès publique via le contrôler begin\_upload :

Dans ce cas, l’autorisation de stockage est créée automatiquement pour tous les utilisateurs. Une entrée est insérée dans la table IO\_UPLOAD avec les restrictions suivantes :

* Le dossier de destination est prédéfinit par la variable de configuration [io\_module:public\_output\_dir]
* La taille des données est limité par la variable de configuration [io\_module:max\_upload\_size]
* La taille des paquets est définit par la variable de configuration [io\_module:packet\_size]

L’utilisateur final peut ensuite utiliser les contrôleurs de flux packet\_upload et finalize\_upload pour initialiser les données.

1. Avec un accès privé via le contrôler create\_upload :

Dans ce cas, l’autorisation de stockage est créée via l’administrateur dans la table IO\_UPLOAD avec des paramètres de configurations spécifiques. L’utilisateur final peut ensuite utiliser les contrôleurs de flux packet\_upload et finalize\_upload pour initialiser les données.

# Dépannage

## Problème de connexion au server SMTP

Il arrive souvent que des problèmes de connexions se produisent lors de l’envoi de courriers par un serveur distant. La plus part du temps il s’agit d’un problème de configuration.

Vérifiez les points suivant :

* Vérifiez les paramètres de configuration (adresse du serveur et numéro de port)
* Si vous utilisez une Box (BBox, NeufBox, …), vérifiez que celle-ci ne bloque pas les envois des messages pour les expéditeurs utilisant un autre nom de domaine que celui du FAI (…@bbox.fr, …@sfr.fr,…). Une option est généralement présente dans l’interface de configuration de votre Box.