Choisir son Framework CRUD : retours d'expériences

Présentation du framework web open-source FuelPHP

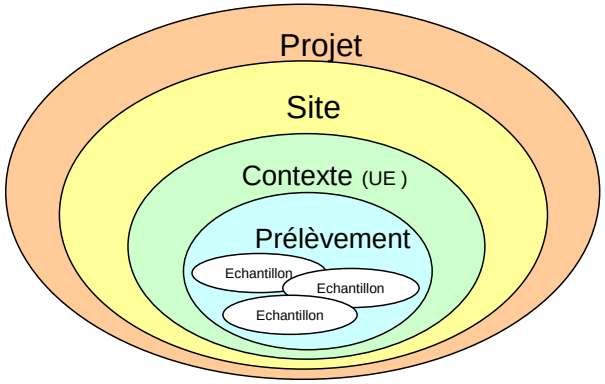
Mercredi 20 novembre 2013

jean-baptiste.barreau@univ-rennes1.fr

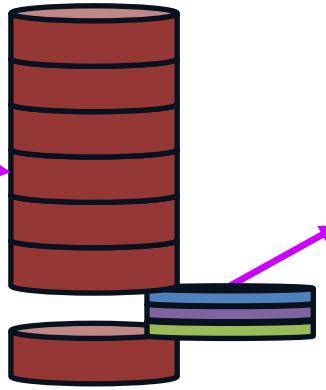
I Présentation/Contexte du projet

1. Problématiques

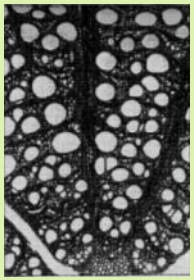
Projet ABCData (Archéologie, Biodiversité, Chronologie Data): Bibliothèque de données paléoenvironnementales.



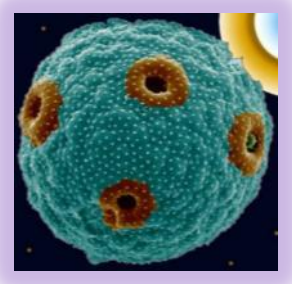
Compatible avec la base BIOARCHEODAT (UMR7209): recensement des données archéozoologiques et archéobotaniques de l'Holocène, en France Métropolotaine.



Prélèvement par carottage



Echantillon anthracologique

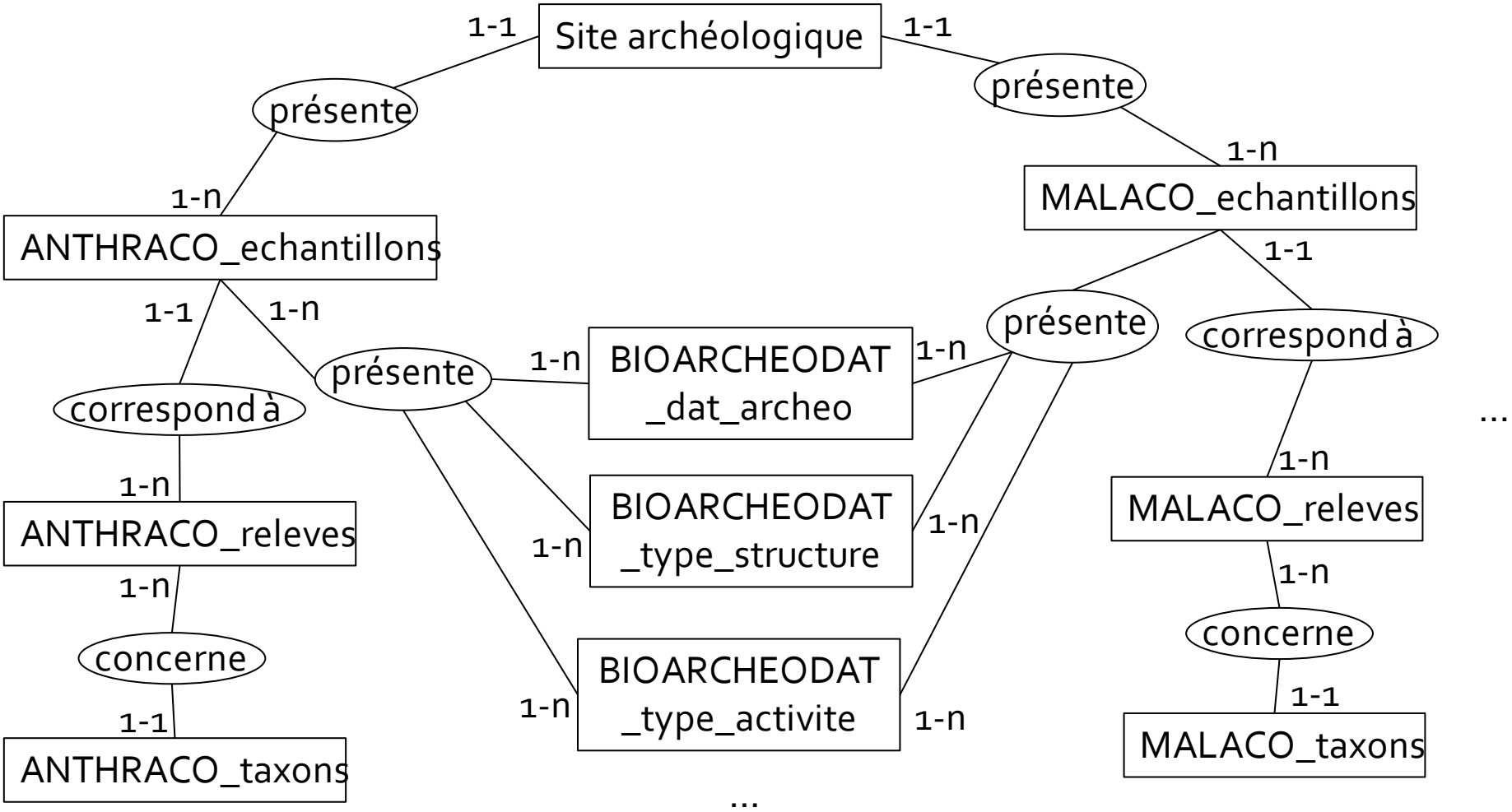


Echantillon palynologique

Echantillon malacologique

I Présentation/Contexte du projet

2. La base de données



Plusieurs centaines d'enregistrements / vingtaines de tables (pour l'instant)

I Présentation/Contexte du projet

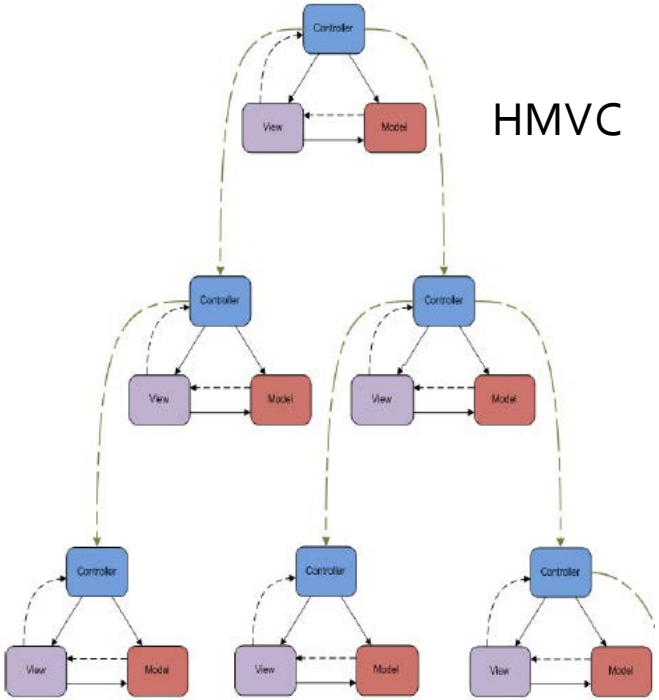
3. Besoins en termes d'interface web

- Nombre d'accès utilisateurs: 1 dizaine d'utilisateurs actifs.
- Gestion des droits d'accès: L'utilisateur doit pouvoir, après authentification login/mdp, lire-éditer-créer-supprimer.
- Administration de la base: par application Web classique de gestion pour les systèmes de gestion de base de données.
- Consultation: Les tableaux **à entêtes fixes** de données et éléments graphiques associés (ex cartes interactives) doivent être accessibles à n'importe qui.
- Saisie / Modification: Zones de texte, listes déroulantes (1-n) + lien création d'entrées dans ces listes, listes à choix multiples (n-n)…

II Présentation de FuelPHP

1. La technologie

- est écrit en PHP 5.3
- se définit comme une version enrichie de CodeIgniter
- SGBD compatibles « sûrs »: MySQL, PostgreSQL, MSSQL et SQLite
- utilise l'extension-interface PHP Data Objects (PDO)
- respecte l'architecture HMVC (Hierarchical MVC)
- moins riche que Symfony mais plus simple



HMVC

II Présentation de FuelPHP

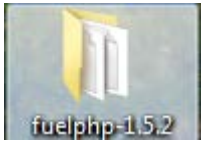
2. ***Model_Crud*** *vs Orm\Model*

-Effectue simplement les opérations CRUD
-Pour des opérations simples sur les tables
-Très rapide à mettre en place
-Génération de code par Scaffold
-Utilisation: *class Model_... extends \Model_Crud*

-**Pas de gestion des relations, pour ça: ORM (object-relational mapping)**
-Model_Crud et ORM utilisent une syntaxe semblable: **possibilités de migrations**...

II Présentation de FuelPHP

3. Création des fichiers du CRUD



public	24/01/2013 17:51	Dossier de fichiers	
.gitignore	24/01/2013 17:51	Fichier GITIGNORE	1 Ko
.gitmodules	08/02/2013 12:58	Fichier GITMODU...	1 Ko
build.xml	24/01/2013 17:51	Document XML	3 Ko
CHANGELOG.md	08/02/2013 12:58	Fichier MD	40 Ko
CONTRIBUTING.md	24/01/2013 17:51	Fichier MD	7 Ko
oil	24/01/2013 17:51	Fichier	2 Ko

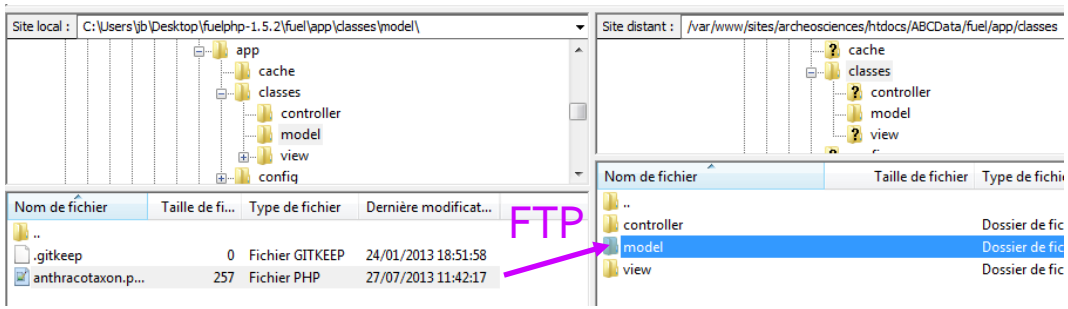
Colonnes | Index? | Contraintes? | Triggers? | Règles? | Info

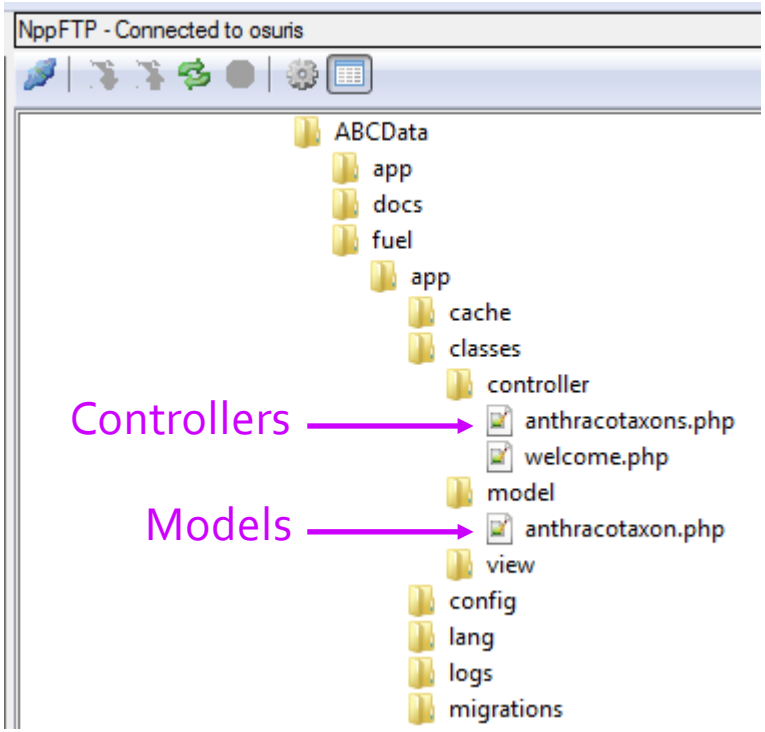
Colonne	Type	NOT NULL	Défaut	Contraintes	Actions			Commentaire
id	integer	NOT NULL	nextval('taxons_id_seq'::regclass)		Parcourir	Modifier	Supprimer	
nom	character(32)				Parcourir	Modifier	Supprimer	

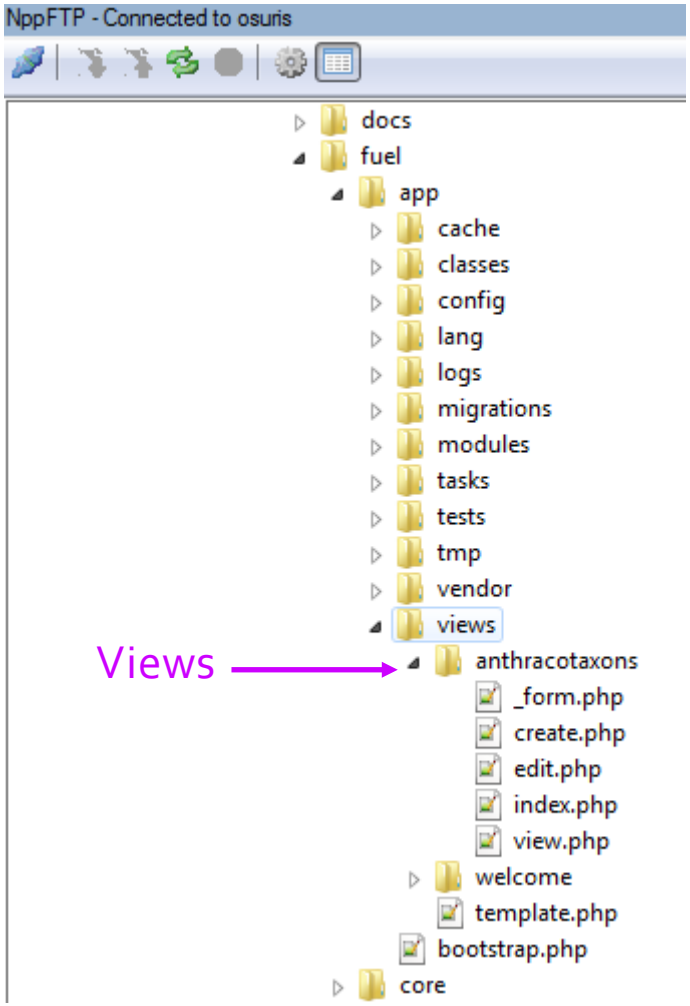
```
C:\Users\jb\Desktop\fuelphp-1.5.2>php oil generate scaffold/crud anthracotaxons
nom:character(32)
        Creating migration: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\migration
s/002_create_anthracotaxons.php
        Creating model: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\classes/model
/anthracotaxon.php
        Creating controller: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\classes/
controller/anthracotaxons.php
        Creating view: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\views/anthraco
taxons/index.php
        Creating view: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\views/anthraco
taxons/view.php
        Creating view: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\views/anthraco
taxons/create.php
        Creating view: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\views/anthraco
taxons/edit.php
        Creating view: C:\Users\jb\Desktop\fuelphp-1.5.2\fuel\app\views/anthraco
taxons/_form.php
```

II Présentation de FuelPHP

4. Upload des fichiers du CRUD







II Présentation de FuelPHP

5. Résultat interface CRUD

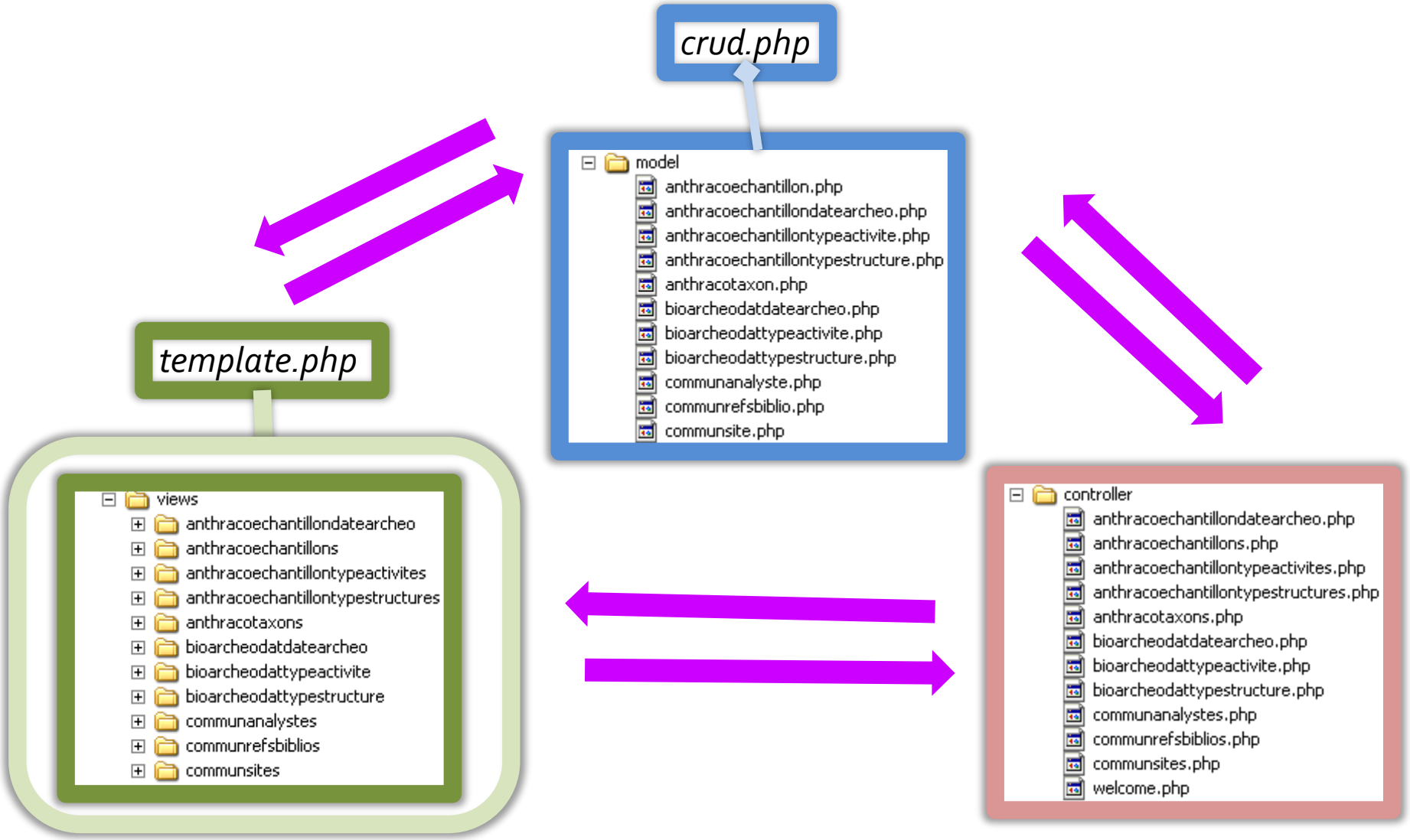
```
class Model_Anthracotaxon extends Model_Crud
{
    protected static $_table_name = 'ANTHRACO_taxons';

    public static function validate($factory)
    {
        $val = Validation::forge($factory);
        $val->add_field('nom', 'Nom', 'required');

        return $val;
    }

}
```

Anthracotaxons **Listing Anthracotaxons**

Nom	
Quercus sp.	View | Edit | Delete
Quercus/Castanea	View | Edit | Delete
Cytisus sp.	View | Edit | Delete
Ulex sp.	View | Edit | Delete
Fagus sylvatica	View | Edit | Delete
Juglans sp	View | Edit | Delete
Fraxinus sp.	View | Edit | Delete
Populus sp.	View | Edit | Delete
Salix sp.	View | Edit | Delete
Acer sp.	View | Edit | Delete
Alnus sp.	View | Edit | Delete
Castanea sativa	View | Edit | Delete
Ulmus sp.	View | Edit | Delete

II Présentation de FuelPHP

6. Organisation des fichiers



II Présentation de FuelPHP

7. Relations 1-n: index.php

crud.php | anthracoechantillons.php | index.php

```php
//retourn tableau complet pour relation n-1
public static function getAllTable($table1, $champ1, $champ2, $table2, $champ3)
{
    $modif1=$table1.'.'.$champ1;
    $modif2=$table2.'.'.$champ3;
    $query=DB::select($modif1, $champ2)->from($table1)->join($table2)->on($modif1, '=', $modif2)->execute()->
    as_array($champ1,$champ2);
    return $query;

}
```

crud.php

```php
<?php
class Controller_Anthracoechantillons extends Controller_Template
{

    public function action_index()
    {
        $data['getAllSite'] = Model_Anthracoechantillon::getAllTable( 'COMMUN_sites', 'id', 'nom_site',
        'ANTHRACO_echantillons', 'id_site');
        $data['anthracoechantillons'] = Model_Anthracoechantillon::find_all();
        $this->template->title = "Anthraco";
        $this->template->content = View::forge('anthracoechantillons/index', $data);

    }
```

anthracoechantillons.php

id_site	
1	Site archéologique
1	
1	Champ du Château

```php
<?php foreach ($anthracoechantillons as $anthracoechantillon): ?>        <tr>

            <td><?php echo $anthracoechantillon->nom; ?></td>
            <td>
            <?php if (($anthracoechantillon->id_site != null)) { print_r($getAllSite[$anthracoechantillon->
            id_site]); } else { echo $anthracoechantillon->id_site = null; }
            ?>
```

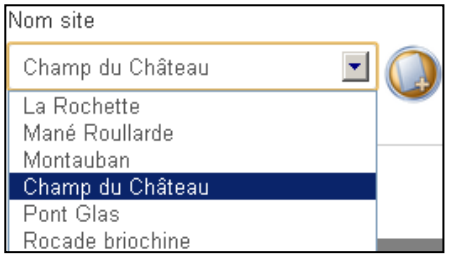
index.php

II Présentation de FuelPHP

8. Relations 1-n: form.php

anthracoechantillon.php

```php
<?php
class Model_Anthracoechantillon extends Model_Crud
{
    protected static $_table_name = 'ANTHRACO_echantillons';
    public static $_relation_params = Array
                                        (
                                            Array
                                                (
                                                    'table_name' => 'COMMUN_sites',
                                                    'id_field' => 'id_site',
                                                    'name_field' => 'nom_site',
                                                    'all_var' => 'getAllSite',
                                                    'one_var' => 'getOneSite',
                                                    'query_var' => 'EditSiteQuery'
```



anthracoechantillons.php

```php
public function action_edit($id = null)
{
    //Relations 1-n
    $data = Model_Anthracoechantillon::getRelationDatas($id,'id',query_var,'ANTHRACO_echantillons',Model_Anthracoechantillon::$_relation_param
    $data = array_merge($data, Model_Anthracoechantillon::getRelationDatas($id,'id',one_var,'ANTHRACO_echantillons',Model_Anthracoechantillon
```

form.php

```php
<div class="input">
    <?php
        $id = "";
        if(isset($getOneSite)) { $id = array_search($getOneSite, $EditSiteQuery); }
        echo Form::select('id_site', $id, $EditSiteQuery);
        echo " " . Html::anchor('communsites/create', Asset::img('add.png', array('width' => '35')));
```

II Présentation de FuelPHP

9. Relations n-n : index.php

```php
public static function getManyManyRelationArray($_ExternalTable,$_SelfIdField,$_ExternalIdField,$_ReturnNames=true)
{
    $l_cur_ech = -1;
    $l_Array = self::find(array('order_by' => array($_SelfIdField => 'asc')));
    if($l_Array)
    {
        foreach ( $l_Array as $value )
        {
            if( $l_cur_ech != $value[$_SelfIdField] )
            {
                $l_cur_ech = $value[$_SelfIdField];
                $l_result[$value[$_SelfIdField]] = array();
            }
            if($_ReturnNames)
            {
                $l_result[$value[$_SelfIdField]][] = $_ExternalTable[$value[$_Exte
            }
```

crud.php

id	echantillon	type_structure	Types de structure
1	1	16	-Fosse (étudiée dans un complexe) -Ensemble de fosses/Trous (précision: calage/dalle posée à plat au fond)
2	1	23	
3	2	16	
4	2	23	
5	3	16	
6	3	23	

```php
function showManyToManyList($_ManyToManyArray)
{
    if(isset($_ManyToManyArray))
    {
        foreach ($_ManyToManyArray as $l_value):
            echo "-" . $l_value . "<br>";
        endforeach;
    }
```

template.php

```php
public function action_view($id = null)
{
    //Relations 1-n
    $data = Model_Anthracoechantillon::getRelationDatas($id,'id','one_var','ANTHRACO_echantillons',Model_Anthracoechantillon::
    $_relation_params);

    //Relations n-n
    $data['anthracoechantillontypeactivite'] = Model_Anthracoechantillontypeactivite::getManyManyRelationArray(Model_Anthracoechantillon
    ::getTabAssoc('BIOARCHEODAT_type_activite', 'id', 'description'),'ech_id','activite_id');
    $data['anthracoechantillondatearcheo'] = Model_Anthracoechantillondatearcheo::getManyManyRelationArray(Model_Anthracoechantillon::
    getTabAssoc('BIOARCHEODAT_dat_archeo', 'id', 'description'),'echantillon','dat_arch');
    $data['anthracoechantillontypestructure'] = Model_Anthracoechantillontypestructure::getManyManyRelationArray(
    Model_Anthracoechantillon::getTabAssoc('BIOARCHEODAT_type_structure', 'id', 'description'),'echantillon','type_structure');
```

anthracoechantillons.php

```php
if (isset($anthracoechantillondatearcheo[$anthracoechantillon->id])) { showManyToManyList($anthracoechantillondatearcheo
$anthracoechantillon->id]); }
?>
```

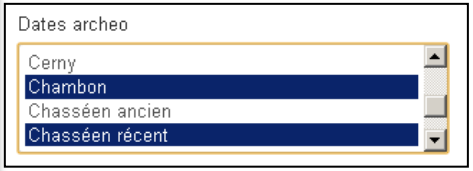
index.php

II Présentation de FuelPHP

10. Relations n-n : form.php

```php
public static function saveManyManyRelation($_selfId,$_SelfFieldId,$_ExternalFieldId)
{
    $l_result = true;
    if(Input::post($_ExternalFieldId))
    {
        foreach (Input::post($_ExternalFieldId) as $l_value):
            $l_Model = self::forge(array(
                $_SelfFieldId => $_selfId,
                $_ExternalFieldId => $l_value,
            ));
            $l_result = $l_result && $l_Model->save();
```

crud.php




```php
$data['bioarcheodat_date_archeo'] = Model_Anthracoechantillon::getTabAssoc('BIOARCHEODAT_dat_archeo', 'id', 'description'

if ($anthracoechantillon and $anthracoechantillon->save())
{
    $l_LastId = Model_Anthracoechantillon::getLastId();
    if( Model_Anthracoechantillontypeactivite::saveManyManyRelation($l_LastId,'ech_id','activite_id') &&
        Model_Anthracoechantillondatearcheo::saveManyManyRelation($l_LastId,'echantillon','dat_arch') &&
        Model_Anthracoechantillontypestructure::saveManyManyRelation($l_LastId,'echantillon','type_structure')
    )
    {
```

anthracoechantillons.php

```php
<?php
$l_SelectedTypeDateArcheo = getManyToManySelectedList( isset($anthracoechantillondatearcheo) ?
$anthracoechantillondatearcheo : null, isset($anthracoechantillon) ? $anthracoechantillon->id : null );
echo Form::select('dat_arch[]', $l_SelectedTypeDateArcheo, $bioarcheodat_date_archeo, array('class' => 'span4',
'multiple'));
?>
```

form.php

II Présentation de FuelPHP

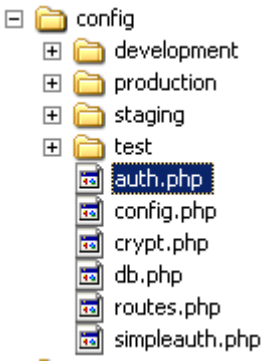
11. Sécurité

- L'intégrité référentielle est gérée au niveau de **Postgresql** (NULL/NO NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK) et de la logique des interfaces.
- FuelPHP privilégie **l'encodage en sortie** plutôt que le filtrage en entrée: les données sont stockées *brutes* et envoyées inoffensives au client.
- Protection CSRF: un **jeton de sécurité** dans le formulaire contre les attaques CSRF.
- **Filtrage XSS** contre les attaques XSS.
- Filtrage des $_GET, $_POST et $_COOKIE possibles
- Protection contre l'injection SQL grâce au **Query Builder** qui génère son propre SQL: pas de requête SQL directement dans le code.

II Présentation de FuelPHP

12. Authentification

- **Simpleauth**:




```
'always_load' => array(
    'packages' => array(
        'auth',
    ),
),
```

config.php

```
public function action_login()
{
    $data = array();

    // If so, you pressed the submit button. Let's go over the steps.
    if (Input::post())
    {
        // Check the credentials. This assumes that you have the previous table created and
        // you have used the table definition and configuration as mentioned above.
        if (Auth::login())
        {
            // Credentials ok, go right in.
            Response::redirect('success_page');
        }
        else
        {
            // Oops, no soup for you. Try to login again. Set some values to
            // repopulate the username field and give some error text back to the view.
            $data['username']    = Input::post('username');
            $data['login_error'] = 'Wrong username/password combo. Try again';
        }
    }

    // Show the login form.
    echo View::forge('auth/login',$data);
}
```

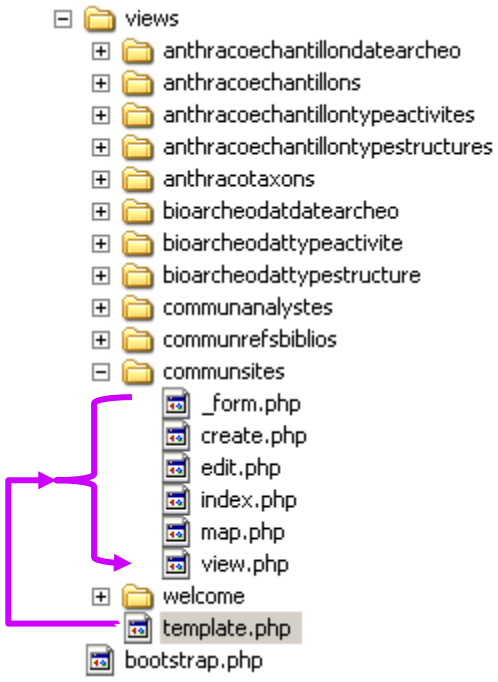
users.php

⇒ gestion des droits multi-utilisateurs possible (champs guest_login, Groups, rôles…)

- LDAP: *packages en cours de développement par la communauté…*

II Présentation de FuelPHP

13. Vues




```php
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title><?php echo $title; ?></title>
    <?php
    echo Asset::css('bootstrap.css');
    echo Asset::css('fixheadertable.css');
    echo Asset::css('jquery-ui-1.10.3.custom.css');
    echo Asset::js('https://ajax.googleapis.com/ajax/libs/jquery/1.5.2/jquery.min.js');
    echo Asset::js('jquery.fixheadertable.js');
    echo Asset::js('proj4js/lib/proj4js-compressed.js');
    echo Asset::js('proj4js/lib/defs/EPSG27572.js');
    echo Asset::js('http://leaflet.cloudmade.com/dist/leaflet.js');
    echo Asset::js('http://openlayers.org/api/OpenLayers.js');
    echo Asset::js('carte_open_layer.js');
    ?>
```

```php
<?php endif; ?>
            <!-- </div> -->
            <div class="span16">
<?php echo $content; ?>
            </div>
        </div>
```

template.php

```php
<?php foreach ($communsites as $communsite): ?>     <tr>

            <td><?php echo $communsite->nom_site; ?></td>
            <td><?php echo $communsite->commune; ?></td>
            <td>
            <?php if ($communsite->departement != null) { print_r($getAllDept[$communsite->departement]); }
            </td>
            <!-- <td> --><?php //echo $communsite->x_lambert; ?><!-- </td> -->
            <!-- <td> --><?php //echo $communsite->y_lambert; ?><!-- </td> -->
            <td><?php echo $communsite->code_unite_geographique; ?></td>
            <td><?php echo $communsite->precision_unite_geographique; ?></td>
            <td><?php echo $communsite->precision_type_site; ?></td>
```

index.php

II Présentation de FuelPHP

14. Internationalisation

```php
return array(
    'language'           => 'en', // Default language
    'language_fallback'  => 'en', // Fallback language when file isn't available for default language
    'locale'             => 'en_US', // PHP set_locale() setting, null to not set
    'locales'            => array(
        'en' => 'en_US',
        'fr' => 'fr_FR'
    ),
```

config.php

```php
return array(
    'test_page' => 'Page de test',
    'penguin' => 'Pingouin',
    'hello' => 'Salut !',
);
```

app/lang/fr/test.php

```php
return array(
    'test_page' => 'Test page',
    'penguin' => 'Penguin',
    'hello' => 'Hello!',
);
```

app/lang/en/test.php

```php
class Controller_Test extends Controller
{
    public function action_index()
    {
        Config::set('language', 'fr');
        Lang::load('test');

        $data = array();
        $data['title'] = Lang::get('test.test_page');
        $data['content1'] = Lang::get('hello');

        return View::forge('test/template', $data, 'UTF-8');
    }
}
```

fuel/app/classes/controller/test.php

```php
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title><?= $title ?></title>
</head>
<body>
    <div id="wrapper">
        <div id="content">
            <?= $content1 ?>
        </div>
    </div>
</body>
</html>
```

fuel/app/views/test/template.php

II Présentation de FuelPHP

15. Interopérabilité

La classe « Format » permet de convertir entre différents formats tels que XML, JSON, CSV…

```
$json_string = '{"foo":"bar","baz":"qux"}';
print_r(Format::forge($json_string, 'json')->to_csv());
```

16. Gestion du cache

La classe « cache » permet de mettre en cache le résultat d'une opération lourde en ressources.

```
// Cache 'String to be cached' for 3 hours with identifier "test"
Cache::set('test', 'String to be cached.', 3600 * 3);

// try to retrieve the cache and save to $content var
try
{
    $content = Cache::get('test');
}
catch (\CacheNotFoundException $e)
{
```

II Présentation de FuelPHP

17. Développement de modules spécifiques

```
namespace Mymodule;

class Controller_Widget
{
    // ...
}
```

18. Debugger

Notice!

ErrorException [Notice]: Undefined variable: anthracoechantillons1

APPPATH/views/anthracoechantillons/index.php @ line 1

```
0
1 <?php if ($anthracoechantillons1):
2 $icons = createIconsHTMLCode('anthracoechantillons','24');
3 echo initFixHeaderTable(400);
4 ?>
5 <table class="fixme">
6         <thead>
```

II Présentation de FuelPHP

19. Communauté

- 6 développeurs principaux
- 300 développeurs – contributeurs
- Philosophie: développeur principal = développeur – contributeur
- Social coding: GitHub (répertoire « Fuel » pour v1.0 / « FuelPHP » pour v2.0)

20. Licence

Licence MIT: "La licence donne à toute personne recevant le logiciel le droit illimité de l'utiliser, le copier, le modifier, le fusionner, le publier, le distribuer, le vendre et de changer sa licence. La seule obligation est de mettre le nom des auteurs avec la notice de copyright." (Wikipedia)

II Présentation de FuelPHP

21. Historique

- Le projet est récent et a commencé officiellement en octobre 2010…

Version	Release date
1.0.1	August 23, 2011[6]
1.1	December 13, 2011[7]
1.2	June 3, 2012[8]
1.3	September 9, 2012[9]
1.4	November 9, 2012[10]
1.5	January 20, 2013[11]
1.6	May 3, 2013[12]
1.7	October 14, 2013[13]
2.0	No date set,[14][15]

22. Documentation (anglais)

III Retour d'expérience & Conclusion

1. Présentation d'une réalisation

1. Télécharger http://fuelphp.com/files/download/19
2. Concaténation avec excel à partir d'un copier-coller de phpPgAdmin: « *php oil generate scaffold/crud anthracorels id:integer id_echantillon:integer code_espece:integer nb_cernes:smallint largeur_tot:real courbure:character(255) rythme:character(255) combustion:character(255) hyphe:smallint insecte:smallint cambium:smallint thylle:smallint bois_reaction:smallint trace_travail:smallint moelle:smallint masse:smallint rayon:real age:character(255) remarque:text diam_mes:real type_branche:character(255) bi:character(255) bf:character(255) lr:character(255) prelevement:character(255) ppzp:smallint pfr:smallint* »
3. Lancement de la commande à partir de l'interpréteur de commandes de Windows
4. Upload par nppftp
5. Modification de la table $_table_name dans anthracorel.php
6. http://archeosciences-abcdata.osuris.org/anthracorels

III Retour d'expérience & Conclusion

2. Apprentissage

- Difficile de quantifier l'apprentissage des théories générales inhérentes aux frameworks MVC, à la programmation orientée objets, PHP etc… (personnellement 3 ans de manière épisodique Symfony, CakePHP).
- Pour FuelPHP: 1/2h pour créer la 1ère interface, 1 semaine de développement pour les relations 1-N, N-N…

3. Contraintes techniques

III Retour d'expérience & Conclusion

4. Problèmes rencontrés

- Le choix d'utiliser Model_Crud à la place du Orm\Model a imposé un peu de développement pour la gestion des relations (mais ça ne m'est personnellement pas désagréable…).
- Une page de 91 échantillons fois 9 variables au travers d'un tableau à entête fixe (http://archeosciences-abcdata.osuris.org/index.php/anthracoechantillons): résultats PageSpeed Insights = *57/100 Mobile*, *72/100 Ordinateur*
- Plus de temps passé sur les listes à choix multiples mais le code est prêt à être réutilisé.

III Retour d'expérience & Conclusion

5. Conclusion

- Adapté à mes besoins (générer des interfaces CRUD très rapidement) et à mon profil (un peu de développement mais pas trop…)
- M'a permis de me concentrer sur l'essentiel: conception de la base, interfaces de lecture/saisie.
- Excellent debugger
- J'ai la chance de « profiter » d'un serveur (OSURIS) sur lequel l'administration des bases et le développement du code est aisé et peu contraignant.
- Framework un peu jeune ce qui constitue indéniablement un risque.

Merci!
Des questions?