Petit traité de performances à l'usage de tous

adyaX commerce guys {EPITECH. L'EXPERTISE INFORMATIQUE} G-nova

MAKINA CORPUS OSInet radio france sigma L'INFORMATIQUE AU SERVICE DE L'HOMME

Drupalcamp Nantes - 29 et 30 janvier 2011

DoLink Everywhere - Anytime ChipWay La communication multimédia interactive Bloggy Business

Drupal
Petit traité de performances à l'usage de tous

Intégrateur, architecte, développeur, tous concernés

Pierre Rineau

Makina Corpus
pierre.rineau@makina-corpus.com

Janvier 2011

Synopsis

L'API grandissante de Drupal et la multiplication de nouvelles fonctionnalités génèrent de plus en plus de sites "éléphantesques". La réactivité d'un site, dans le monde du web, est un domaine critique qui concerne tous les intervenants. Travail préparé en amont par le développeur, cet aspect nécéssite d'être présent dans l'esprit de l'intégrateur et doit être maîtrisé par les administrateurs système. Cette session à pour but de présenter la toile d'araignée et les réactions en chaîne provoquées dans un contexte ou ces trois rôles s'ignorent.

Sommaire

1. Petite introduction
2. Les causes évidentes
3. Détails et solutions
4. Et pour finir

Qu'est ce que Drupal ?

Qu'est ce que Drupal ?

Sur http://drupal.org :

> *Drupal is an open source content management platform powering millions of websites and applications.*

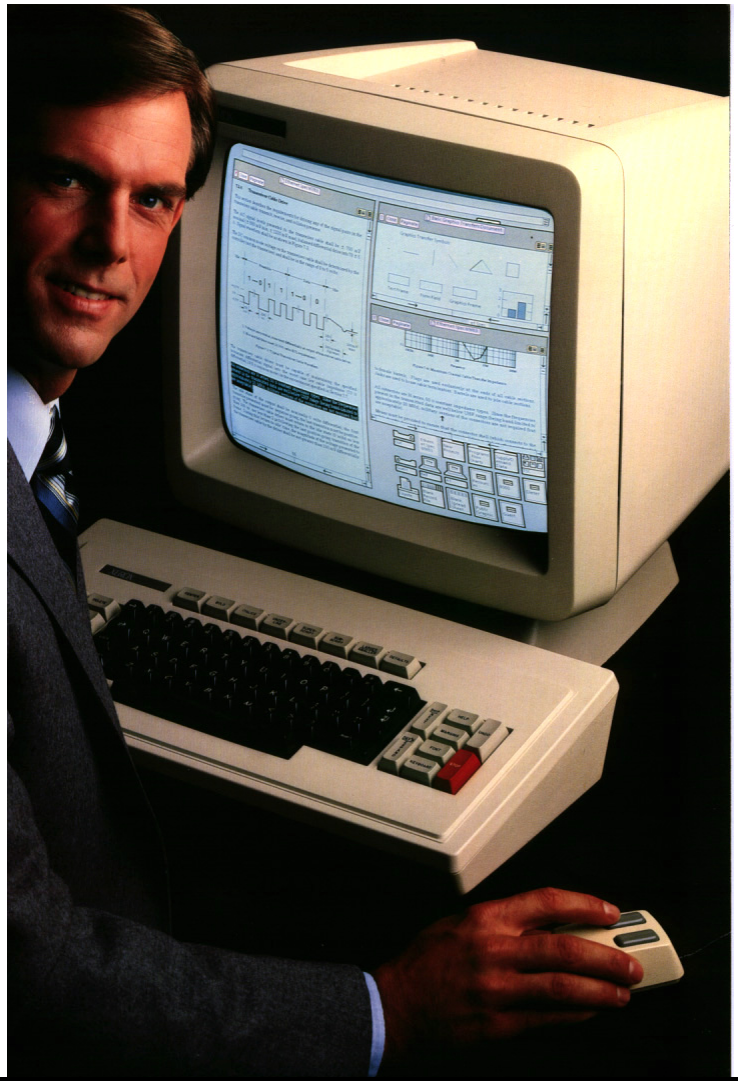
Sur http://acquia.com/what-is-drupal :

> *Tens of thousands of people and organizations have used Drupal to power scores of different web sites, including community web portals, corporate web sites, social networking sites, personal web sites or blogs, and much more.*

Forces mais faiblesses

- Très fort **dynamisme**
- Le **hook** : altération VS extention
- Un métier **délaissé**, récupéré par les modules
- Des utilisation **hétéroclites** : *blog, e-commerce, social networking*

Le développeur



- Développe souvent pour **un seul cas d'utilisation**
- Ne **connaît pas** qui va utiliser son produit
- Documente peu

L'intégrateur

- Utilise le travail **de multiple** développeurs
- Ne voit que la **partie émergée de l'iceberg**
- On lui demande souvent d'être **architecte logiciel**

L'architecte système

- **Récupère le travail** de l'intégrateur
- Introduit des contraintes nouvelles, **souvent a posteriori**
- Doit maîtriser le *scaling* d'une plateforme **parmis tant d'autres**

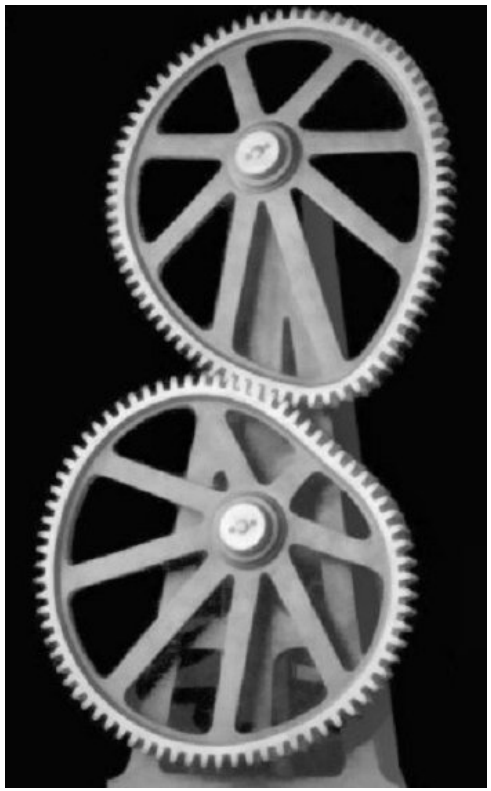
L'échelle d'un module

L'abstraction

Ce que conçoit le développeur :

L'abstraction

Ce qu'utilise l'intégrateur :



Le résultat !

Le SQL

- Drupal, **usine à requêtes**
- Schema **dispersé** et API **non centralisée**
- Des modules **contraints au hack**

Le SQL : des solutions

- Développeurs, **évitez les requêtes inutiles**
- Développeurs toujours, utilisez des **caches**
- Intégrateurs, **connaissez les modules**!
- Architectes, pensez à **activer les caches**

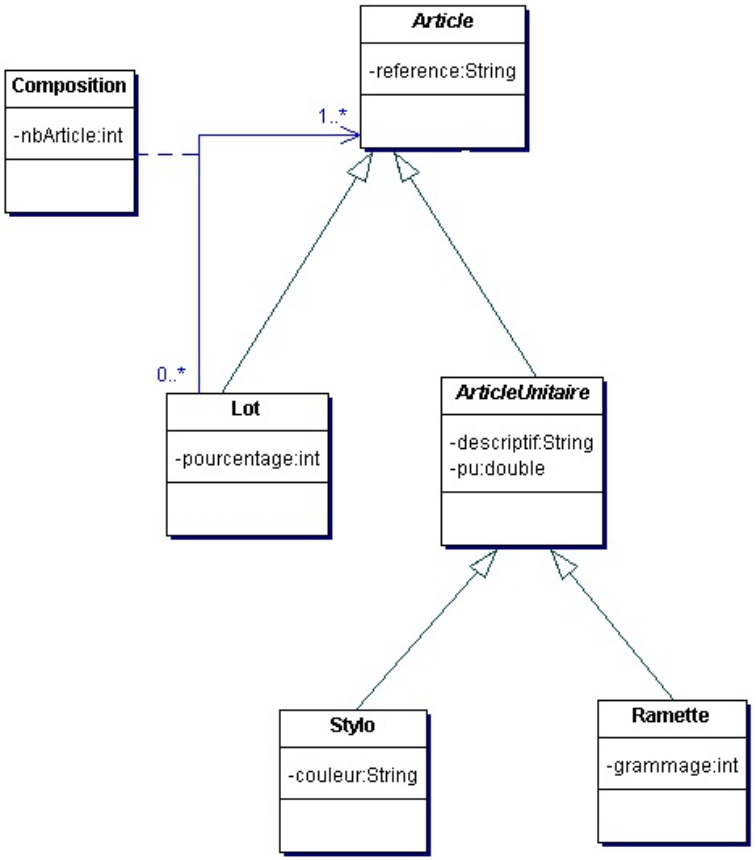
MySQL Vs. PostgreSQL

MySQL, un grand **gourmand**!

PostgreSQL, le poids de l'expérience.

Cache binaire : concept

Normalisation VS **Sérialisation**



Cache binaire : utilité

SOULAGER LA BASE DE DONNÉE

Non sans constraintes, ne pas répetez les mêmes erreurs :

- Pensez aux **I/O**
- Pensez **latence**

Les caches et de le développeur

La responsabilité du développeur : mettre en place les caches

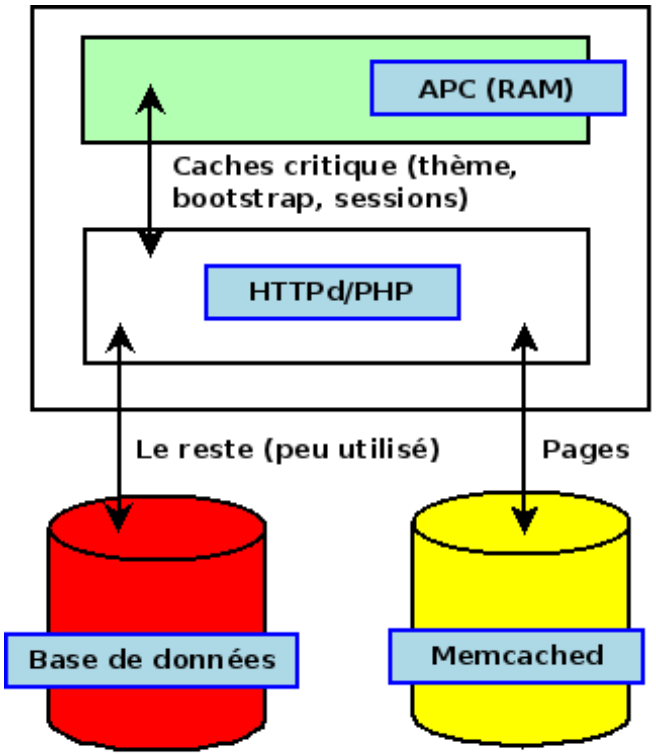
- De multiple niveaux : des **des caches différentiés**

Plusieurs facteurs de séparation logique des données :

- **Criticité** (fréquence d'accès)
- **Volumétrie** (taille des éléments)

Les caches et l'architecte

La résponsabilité de l'architecte : bien les utiliser



- Connaître les différents niveaux
- Et les **différents backends**
- **Critictié VS volumétrie**, ou bien choisir ses *backends*

Généralités et erreurs communes

- Parlons un peu des caches **statiques**
- Statiques ou persistents, **ne vous substituez jamais au compilateur!**
- Le cache doit être **au coeur du design de l'application**
- Dans certains cas il doit **orienter les API**
- Trop de niveaux de cache dans la même API soulève souvent un **problème de design**

L'ennemi de l'architecte : l'environnement

Linux / Apache / MySQL / PHP

- Peu importe l'OS, même problématiques
- Architecture **N-tiers** de fait
- Environnement **consommateur en I/O**

L'OS

- Son goulot d'étranglement, les **I/O**

Le HTTPd

- Son goulot d'étranglement, le HTTP (et donc les **I/O**)
- Mais aussi :
  - La **concurence**
  - La mémoire

Le SGBD

- Maître des données, de leur **intégrité et cohérence**
- Difficile à *clusteriser*
- Une force et faiblesse : le *query cache* (MySQL seulement)

PHP

- Langage **interpété** : compilation lors de chaque *hit*
- Non persistent, d'où la nécéssité des **caches binaires**

Comment rendre PHP *lightening fast*

Utilisez un **OPCode cache**

- **Enlève la phase de compilation et interprétation**
- Apporte de nombreuse **optimisations**

De manière générale, **divise le temps d'exécution approximativement par 2**

Un OPCode cache : APC

Voici un exemple des optimisations possibles. Les chiffres proviennent d'un environnement de développement, sur la home page d'un site D7 sans modules.

- PHP sans OPcode : **360ms**
- APC, sans configuration préalable : **107ms**
- Avec `apc.include_once_override` et `apc.canonicalize` et sans `apc.stat` : **103ms**
- Avec le `cache_bootstrap` binaire dans APC : **94ms**

L'ennemi du développeur : le manque de culture

- À quasiment chaque problème **il existe déja une solution**
- Un code grandissant **nécéssite de réfléchir**
- Un code trop grandissant **ne demande qu'à diminuer**
- Renseignez vous sur **comment la machine comprend votre code**
- N'ayez jamais peur lire celui des autres !

L'ennemi de l'intégrateur : trop de module tue le module

- Plus on est de fou, moins c'est rapide
- Pourquoi utiliser un module de 100 000 lignes de code quand 10 suffisent ?
- Pensez-vous à tester les performances durant le développement ?
- Savoir détecter les incohérences

Quelques idées en vrac

- **Profilez**-vous durant le développement et l'intégration (xdebug, analyse de code, benchmarks) ?
- **Analysez-vous l'impact de performance de chaque module ?** Si non, faites le ! Vous aurez des surprises..
- Le meilleur ami, de tous, à tous les niveaux : le **benchmark**.
- Mais avant tout, **quel est le métier de votre site ?**

Ne sachant pas où le temps allait nous amener

Zi Ende

En avant pour des sites toujours plus sexy!

Figure: ©Jean-Yves Lemoigne

adyaX commerce guys {EPITECH. L'EXPERTISE INFORMATIQUE} G-nova

MAKINA CORPUS OSInet radio france sigma L'INFORMATIQUE AU SERVICE DE L'HOMME

Drupalcamp Nantes - 29 et 30 janvier 2011

DoLink Everywhere - Anytime ChipWay La communication multimédia interactive Bloggy Business