

# Eurêka! Science News

## Une étude de cas Drupal

Par Michael Imbeault ([Fireang31](#))

[Eurêka! Science News](#) vient d'être lancé. C'est un site dédié à l'actualité scientifique, mais avec une particularité : celle d'être entièrement automatique. Le site n'emploie aucun rédacteur ou éditeur ; il identifie les relations entre les informations brutes provenant des grands sites existants d'information scientifique, et les regroupe, les classe, les ordonne, les labélise, trouve les communiqués de presse qui s'y rapportent, et les publie directement sur le site. Le résultat est une vue générale efficace de tout ce qui se passe dans le monde scientifique, en prise directe sur l'actualité. Voici comment.



### Historique



Tout d'abord, quelques notes sur ma découverte de Drupal. Il y a quatre ans, j'ai lancé [Biology News Net](#) sur la base de [Movable Type](#) ; pour moi la biologie est la science de référence, et j'avais trouvé anormal qu'il n'y ait aucun site dédié à l'actualité de la biologie. Le site est rapidement devenu très populaire (numéro 1 sur Google pour « Biology News »), ce qui était inattendu, dans la mesure où j'avais lancé le site comme un projet / blogue amateur, et j'ai rapidement été confronté aux limitations de Movable Type : l'ajout de nouvelles fonctions étaient compliqué, les performances n'étaient pas extraordinaires, même sur un serveur dédié, et la personnalisation n'était pas vraiment possible. Un seul exemple : pour le forum, j'avais en fait été contraint d'utiliser une instance de phobie avec un lien entre ses sessions et celles de Movable Type, une solution hasardeuse s'il en est, et

un cauchemar pour la maintenance. Comme je ne pouvais de toute façon pas attendre plus de la part d'un moteur de blogues - rôle pour lequel Movable Type m'a rendu de bons et loyaux services - je me suis mis à la recherche d'une solution plus adaptée, et c'est alors que j'ai découvert Drupal (il y a environ deux ans) et que j'en suis tombé amoureux. Sa courbe d'apprentissage est raide, mais il est si puissant que le temps investi à le maîtriser est largement rentabilisé à long terme. Bien que je n'aie pas le temps de contribuer largement au développement de Drupal lui-même, j'aide lorsque je le peux, et je maintiens un module contribué ([quickstats.module](#), codé par [chx](#) avec quelques améliorations de mon cru).

## Objectifs

---

Je voulais construire quelque chose qui soit non seulement plus gros que Biology News Net, mais aussi différent : un site qui publierait les nouvelles lors de leur révélation, et pour la science dans son ensemble. Je mets à jour Biology News manuellement et, étant passablement occupé, cela signifie généralement une seule fois par jour. J'ai ressenti le besoin d'automatiser ce processus tout en maintenant la qualité à haut niveau ; il existe de nombreux supports de diffusion de l'information scientifique au quotidien et, pour tout dire, tout n'est pas du plus grand intérêt. L'inspiration m'est venue de sites comme [Techmeme](#) et [Google News](#).

## La construction du moteur Eurêka

---

J'ai commencé par identifier les principaux composants d'un agrégateur intelligent :

- une source d'informations, telle qu'un agrégateur RSS
- un moteur de regroupement, pour relier les nouvelles entre elles en amas cohérents
- un moteur de classification, pour catégoriser les nouvelles (Biologie, Physique, Médecine, Astronomie)
- un moyen d'affecter des scores aux amas, pour déterminer l'ordre d'affichage des nouvelles



Pour commencer, il me fallait donc un agrégateur RSS de qualité; celui fourni par défaut dans la distribution de Drupal ne convenait pas, dans la mesure où il ne créait pas de nœuds à partir des éléments RSS importés. Les autres agrégateurs disponibles à l'époque étaient [Leech](#) et [Feedparser](#) mais l'un comme l'autre incluait de nombreuses fonctionnalités dont je n'avais pas l'usage et qui n'avaient pas encore atteint un degré de maturité suffisant. Fort heureusement, Ted Serbinsky ([m3avrck](#)) est venu à ma rescousse, en publiant [Simplefeed](#) au moment où j'en avais le besoin. Simple, rapide, et efficace : que demander de plus ?

Je suis donc passé à l'étape suivante, la construction des trois modules spécifiques destinés à mettre en œuvre les autres fonctionnalités. Le processus a été facilité par l'infinie extensibilité de Drupal : le mécanisme des « hooks » est une merveille ! Le résultat est ce que j'appelle Eureka Engine, le moteur Eurêka. Voici quelques détails à propos de ces trois modules.

### **Clusterer.module**

La première étape consiste à rassembler les articles similaires en amas (« clusters »). Deux mécanismes sont nécessaires pour rassembler des articles : une métrique de similitude, et un algorithme d'amassage. Pour ce qui est du texte, les métriques de similitude peuvent s'appuyer sur la présence de mots dans deux textes : un document peut être représenté sous forme d'un vecteur (cf [modèle vectoriel](#)).

Fort heureusement, c'est exactement ce que fait le moteur de recherche en texte intégral de MySQL : en utilisant un article complet comme critère de recherche sur l'ensemble des autres articles, il est possible d'obtenir un score de similitude entre chaque paire d'articles. J'utilise une technique de fenêtre glissante pour limiter le nombre d'articles d'articles que clusterer.module doit comparer. Les articles étroitement corrélés ont tendance à paraître dans un intervalle de temps relativement court, de sorte qu'il suffit de limiter les recherches à une fenêtre de quelques jours parmi l'ensemble des actualités. La complexité des algorithmes d'amassage hiérarchiques est en  $O(n^2)$ , voire pire, de sorte que le coût de calcul des amas augmente de manière exponentielle en fonction du nombre d'articles. Pour procéder à ce regroupement, j'ai utilisé l'[interface PERL d'une bibliothèque en C](#) implémentant l'amassage hiérarchique : c'est un code très rapide, capable de rassembler un millier d'articles en quelques secondes. J'avais initialement tenté d'écrire mon

propre algorithme d'amassage en PHP, mais le résultat était trop lent et inefficace en matière d'occupation mémoire : ne réinventez jamais la roue si ce n'est pas nécessaire ! Des réglages fins ont été nécessaires pour identifier les paramètres d'amassage idéaux, qui combinaient l'exactitude avec une grande précision. Être trop laxiste produit des amas trop vastes d'informations non reliées, tandis qu'être trop strict empêche les informations pourtant reliées de se rassembler dans les amas. Mais au bout du compte, le résultat s'est révélé satisfaisant.

## **Categorizer.module**

Il existe un grand nombre d'algorithmes de classification ; et il m'en fallait un qui soit précis, mais à un degré encore plus élevé, qui soit très rapide : les articles issus des fils d'agrégation RSS sont nombreux. J'ai choisi un [classifieur bayésien naïf](#) - probablement similaire à celui qu'utilise votre messagerie pour déterminer si le courrier entrant est du spam ou non. Dans le cas d'Eureka Science News, l'algorithme a pour rôle de répartir les nouvelles entrantes dans l'une des huit catégories prédéfinies : Astronomie, Biologie, Climatologie, Santé, Mathématiques, Paléontologie, Physique, et Psychologie. A cet effet, j'ai retravaillé et porté vers Drupal une [bibliothèque PHP d'inférence bayésienne naïve](#). Elle est raisonnablement rapide, et catégorise un article en un dixième de seconde. L'exactitude du système est surprenante une fois qu'il a été convenablement entraîné. Bien sûr, il lui arrive de commettre des erreurs de temps à autre, en particulier lorsqu'il rencontre un article caractérisé par des mots qu'il n'a encore jamais rencontrés, mais je suis satisfait de ses performances à ce stade. Dans l'avenir, le système pourra probablement être amélioré en passant à une [analyse sémantique latente](#).

## **Publisher.module**

Pour en finir, j'ai construit un module pour ordonner les amas de nouvelles et pour localiser et analyser les communiqués de presse. Ce module ordonne les amas en fonction du nombre d'articles qu'ils contiennent, de la pertinence chronologique de chacun de ces articles, et de quelques autres facteurs comme la popularité. Le score est modifié par une décroissance temporelle au moyen d'une formule similaire à celle de la décroissance radioactive à demi-vie, de sorte que seules les informations fraîches ou populaires restent en haut de la page d'accueil.

Dans l'ensemble, le système a dépassé mes attentes : il lui est arrivé de se montrer plus pertinent que moi à quelques occasions, lorsqu'il a trouvé des liens entre des articles que je ne pensais pas être reliés ! *Mais pourquoi a-t-il regroupé ces quatre articles ? Ils ne sont pourtant pas fortement liés ! Oh, mais... ils sont vaguement liés, mais ils ont été présentés lors de la même conférence ! Cool !*

Vous pourrez probablement enrichir sous peu vos propres sites avec des mécanismes similaires, grâce au module [memetracker](#) de Kyle Mathews, un étudiant du Google Summer of Code 2008 ! Il projette de construire un framework générique permettant l'amassage et la classification automatiques des nœuds Drupal. Je suis impatient de voir quelle implémentation il va choisir : un tel framework à usage générique est plus complexe que la réalisation d'un module spécifique à un site comme celui d'Eureka Science News. Si tu lis ceci, Kyle, écris-moi si tu veux en parler !

## **Conception visuelle**

Advantages of grid design and typography : Constant vertical rythm	
World first discovery -- genes from extinct Tasmanian tiger function in a mouse	Sleep-deprived brains alternate between normal activity and 'power failure'
Researchers from the University of Melbourne, Australia, and the University of Texas, USA, have extracted genes from the extinct Tasmanian tiger (thylacine), inserted it into a mouse and observed a...	New imaging research shows that brain activity differs in sleep-deprived and well-rested people. The study, in the May 21 issue of <i>The Journal of Neuroscience</i> , shows that individuals who are...

Disposant de [plus de temps que de moyens financiers](#), j'ai décidé de réaliser moi-même la conception visuelle du site : les designers professionnels sont chers ! Pour ce faire, j'ai employé un framework CSS: [Blueprint CSS](#). C'est un outil idéal pour une conception de type « grille », qui se trouve être précisément bien adaptée aux sites

d'actualités. Avec Blueprint, il est réellement très facile de construire n'importe quel type de conception visuelle en grille, mais uniquement en largeur fixe, au mois pour l'instant. Pour ma part, je pense qu'il vaut mieux utiliser un tel framework plutôt que de tâtonner dans la mise au point d'une conception visuelle maison semi-fluide qui aura des défauts sur presque tous les navigateurs, et à chaque fois pour une raison différente et largement obscure. Les résultats obtenus avec Blueprint sont compatibles avec les principaux navigateurs, ce qui a représenté un gain de temps considérable. Le framework inclut une réinitialisation CSS<sup>1</sup> complète, et une typographie élémentaire agréable, qui maintient un rythme vertical de 18px. En pratique, ceci signifie que chaque ligne de texte, même dans des régions différentes, sera sur la même ligne virtuelle, créant un rythme virtuel plaisant à l'œil.

Couplé avec [Panels.module](#), ce choix nous a permis de créer la conception visuelle des principales pages du site en un temps record : la simplicité d'utilisation du framework permet de tester des dispositions différentes très simplement. Mais même avec un framework, CSS recèle des pièges : j'ai été victime de bogues obscurs dans divers navigateurs, parmi lesquels le bogue guillotine, ou la disparition complète d'un menu avec IE6. J'ai largement utilisé [Browsershots.org](#), qui permet d'obtenir gratuitement des aperçus de son site dans n'importe quel navigateur.

## Drupal – les modules

---

J'ai construit le site avec Drupal 5.x, certains modules et composants critiques n'étant pas encore portés sur Drupal 6.x à l'époque. Il est parfois frustrant de voir à quel point la nouvelle version Drupal est meilleure que celle que l'on déploie, mais c'est une situation inévitable. J'ai prévu de mettre le site à niveau prochainement, mais dans l'intervalle, j'ai porté sur Drupal 5 quelques-unes des fonctionnalités les plus intéressantes pour ce projet, comme l'agrégation Javascript.

La puissance de Drupal est la force de sa communauté de développement : quel que soit la caractéristique dont on a besoin, quelqu'un l'a probablement déjà réalisée et contribué au projet. Voici, sans classement particulier, une brève liste des modules contribués que j'ai utilisés :

- Simplefeed
- Views
- CCK
- Pathauto
- Global Redirect
- Imagecache
- Forward
- Panels (1.x, la version 2.x est sortie un peu trop tard pour ce projet)
- jLightbox
- Taxonomy Access Control
- Quickstats
- CAPTCHA
- Service Links

Certains seront peut-être surpris de voir une liste aussi courte. Drupal offre un accès très simple aux modules contribués, ce qui est pratique, mais qui peut être la source de ce que j'appellerai la « modulte » - l'excès de modules - qui se traduit par des performances limitées. La plupart des modules disponibles dans l'espace des contributions appliquent une approche de type « bon à tout faire » (« kitchen sink approach »), qui conduit à l'accumulation de fonctionnalités et parfois à des limitations de performances : il ne faut jamais oublier que chaque module doit être chargé en mémoire, de sorte que chaque fonctionnalité inutile induit un coût à

---

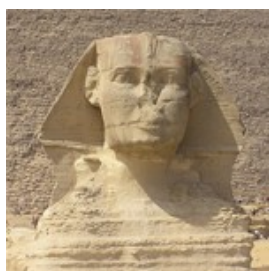
1 Une ré initialisation CSS est un ensemble de déclarations CSS définissant des valeurs par défaut pour l'intégralité des éléments utilisés par un site, afin de s'affranchir des différences de valeurs par défaut entre les divers navigateurs. Le concept a été formalisé et popularisé par Éric [Meyer](#)

chaque chargement de page. A titre d'exemple, le module AdSense, bien qu'il soit très utile dans certains cas, peut la plupart du temps être supprimé en collant simplement le code AdSense dans le fichier `node.tpl.php` ou tout autre fichier ad hoc. Il en va de même pour Google Analytics; bien qu'il existe un module pour l'intégrer, dans la plupart des cas il est plus efficace d'inclure le code dans le pied de page commun. Une direction intéressante serait celle de modules plus « modulaires »; les mainteneurs de modules ont souvent du mal à résister à la tentation de nouvelles fonctionnalités. La mise en place d'une notation des modules sur D.O. pourrait aussi aider au choix entre les cinq solutions en matière de module pour les images, les trois implémentations distinctes de Lightbox, et ainsi de suite.

En tant que développeur de site, c'est un piège dans lequel il est aussi facile de tomber soi-même : ajouter des tonnes de caractéristiques dont les utilisateurs n'ont ni besoin ni envie à un instant donné, et qui peuvent provoquer des limitations de performances. J'aurais ainsi pu ajouter des fonctions de vote, un forum, des blogues individuels par utilisateur, une messagerie instantanée, etc. J'ai choisi de lancer le site avec ce que les usagers recherchaient : les informations. Je pourrai toujours ajouter les autres fonctionnalités par la suite si une demande se manifeste.

## Moteur de recherche : découvrez le Sphinx

---



En première approche, j'ai utilisé [Google Site Search](#) comme moteur de recherche, en considérant que le module de recherche `search.module` intégré à Drupal ne serait pas en mesure de faire face au nombre important de nœuds du site : le site avait en effet produit près de 50 000 nœuds en un mois et demi de tests ; qu'en serait-il dans 5 ans, avec des sources plus nombreuses ? Avec la recherche Google, les mises à jour n'étaient pas instantanées : il subsiste un délai important entre la publication d'un article et son indexation par Google ; l'interface utilisateur était aussi très limitative : bien qu'il soit possible d'intégrer les résultats directement au site, il n'était pas possible d'en modifier la mise en page où les résultats. C'est en

lisant [un article publié par chx sur son blogue](#) que j'ai découvert [Sphinx Search](#), un moteur de recherche simple à configurer et extrêmement rapide : il indexe des centaines de milliers de nœuds en quelques secondes, et renvoie les résultats de recherche en une seconde ou moins. En configurant une indexation de type « principal + différences », il m'a permis d'indexer les informations dès leur publication.

## Performances

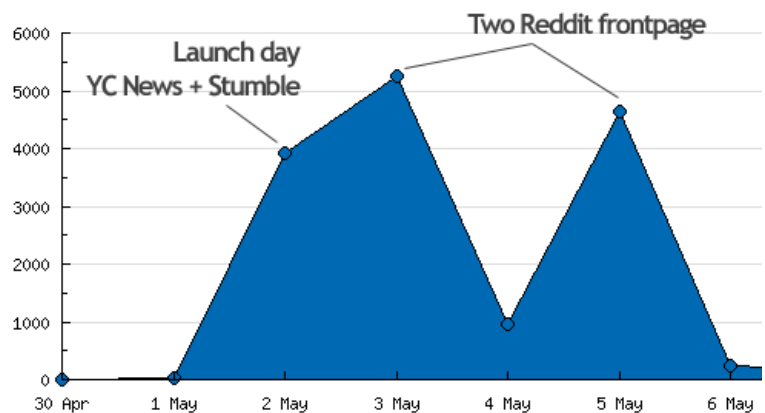
---

Les performances sont toujours la principale préoccupation lors de la création d'un site susceptible de contenir un très grand nombre de nœuds. Eurêka Science News est relativement complexe, ce qui m'a conduit à construire le SQL manuellement pour à peu près tout au lieu d'utiliser le module [Views](#), pourtant si pratique : cela m'a permis d'optimiser pratiquement toutes les requêtes pour leur permettre d'atteindre une bonne rapidité. Views n'est utilisé que pour la section « Archive ».

L'optimisation de MySQL est importante ; le mécanisme de cache intégré à Drupal et le cache de requêtes de MySQL ne sont pas une raison pour exploiter des requêtes mal optimisées. Qu'advient-il si dans l'avenir les utilisateurs doivent être inscrits ? Que se passe-t-il si un utilisateur visite une page non cachée qui provoque un délai de chargement de plusieurs secondes ?

[Servint](#) – J'ai installé le cache APC, optimisé les réglages d'Apache, mais n'ai pas à ce stade installé de mécanismes d'optimisation comme Memcache. En l'état, AB (le benchmark Apache) indique 500 requêtes par seconde sur une page en cache, ce qui est très satisfaisant.

J'ai appliqué point à point les principes de [YSlow](#) – et je rends grâce sur ce point à Wim Leers pour son explication détaillée sur [YSlow et Drupal](#). Bien que j'aie pour l'instant écarté l'idée d'avoir recours à un réseau de distribution de contenus dans la mesure où cela me semble surdimensionné, je conserve dans le collimateur son module d'[intégration CDN](#). Quelques-uns des points qui ont apporté une différence minimale mais mesurable sur les temps de chargement ont été :



- la minification de CSS et Javascript au moyen du [compresseur d'IU Yahoo](#)
- l'agrégation de CSS et Javascript
- L'utilisation de sprites CSS pour les petites icônes
- L'utilisation de la compression PNG et de [PNGslim](#) pour réduire la taille de toutes les images du site

J'ai également exploité les fonctions Drupal `cache_set` et `cache_get` pour les blocs riches en SQL présents sur chaque page, dont le bloc « contenu populaire ».

Je vous invite à vous méfier de l'utilisation de greffons Javascript multiples, car cela peut affecter la performance en ce qui concerne le temps de chargement des pages. A un stade du développement du site, le bloc des images récentes sur la page d'accueil contenait un carroussel Javascript, mais cela affectait de manière significative le temps de chargement des pages pour deux raisons : d'une part, les utilisateurs devaient télécharger des images que la plupart d'entre eux n'auraient pas le temps de voir, et d'autre part le temps d'interprétation et d'exécution du code Javascript était particulièrement élevé : 500 millisecondes pour ce seul carroussel. Bien sûr, c'était plus séduisant visuellement, mais ce n'était pas notre priorité, à la différence des performances.

Le cache de pages de Drupal est excellent pour les utilisateurs anonymes, mais il peut créer des problèmes ; ainsi, je voulais afficher l'intervalle de temps écoulé depuis la publication des articles sur le site amont (« publié il y a deux heures »), avec une précision à la minute près. Il n'était donc pas question que les utilisateurs voient « publié il y a une minute » pendant 10 minutes d'affilée. J'ai résolu ce problème en transposant la fonction `format_interval` de Drupal en Javascript.

Le retour sur investissement de ces optimisations ne s'est pas fait attendre, lorsque le site a été atteint par des pics de popularité provenant de [Reddit](#) (deux sujets différents), [YC News](#) et [Stumbleupon](#) le jour même du lancement et les jours suivants : le modeste VPS du site n'a même pas été mis en difficulté ; les chargements de pages sont demeurés instantanés, même avec 3000 requêtes de page en quelques heures, et environ 15 000 en quelques jours. Depuis, le site a été atteint par des pics [Slashdot](#) à plusieurs reprises, et a figuré en première page de [Mashable](#) sans problèmes de performances particuliers. Je suis convaincu qu'un seul VPS suffira à absorber la croissance du site pour quelque temps.

## Considérations diverses

### Développer sur Windows

---

Il est bel et bien possible de faire du développement Web sur Windows ! J'ai intégralement construit le site sur Windows, principalement avec [Wamp](#) et [Notepad++](#). Je n'y serais pas parvenu sans le Firebug et ses extensions « [It's All Text](#) » pour Firefox.

Développer sur Windows présente évidemment des inconvénients : ainsi j'ai souffert d'un bogue étrange qui se manifestait par l'insertion d'un caractère invisible au début ou à la fin d'un fichier de module, ce qui causait des erreurs, notamment les écrans blancs de la mort. Une fois le problème identifié, éditer les fichiers sous Linux m'a permis de me débarrasser du problème. A propos de Linux, son utilisation est beaucoup plus simple que je ne le pensais précédemment; j'ai appris à l'utiliser le mois dernier pour optimiser le serveur ; c'est facile si vous avez pratiqué DOS (mais je déteste VI, quoi d'autre pour éditer des fichiers ?).

De plus, ma machine de développement s'est trouvée infectée par un publiciel embarrassant qui causait des pannes sur IE et Firefox en ouvrant des pop-ups à la chaîne. Fort heureusement, ceci m'est arrivé après le lancement, et j'avais pléthore de sauvegardes (sur le serveur, sur un disque externe, et sur un portable). Mais il m'a tout de même fallu 3 jours pour m'en débarrasser complètement.

## Leçons retenues / idées diverses

---

Voici, sans ordre particulier, quelques leçons que j'ai retenues de cette expérience :

- trouver un bon nom de domaine est difficile, et prend du temps (et/ou de l'argent) - commencez dès le début et n'arrêtez jamais de chercher, même lorsque vous en avez trouvé un ou deux bons ! Vous pourriez en trouver de meilleurs.
- Faites des sauvegardes fréquentes. Surtout sur Windows.
- Pensez simple ; « plus » signifie souvent « moins » : pour commencer, ne faites que ce que dont vos utilisateurs ont réellement besoin.
- N'ayez pas peur de reprendre quelque chose de zéro si la première version ne convient pas. J'ai reconstruit des composants critiques du système d'amassage à seulement quelques jours du lancement.
- Drupal est un outil de disruption du marché : il permet à un créateur isolé à temps partiel de construire quelque chose de qualité tout en l'apprenant ; imaginez ce qu'il permet d'accomplir entre les mains d'une équipe complète de professionnels !
- Je regrette de n'avoir pas connu [simpletest](#) il y a un an ; j'ai passé trop de temps à éliminer des bogues, et parfois le même bogue qui réapparaissait après avoir disparu. Certains aspects du code comme l'amassage et l'analyse à base d'expressions régulières auraient pu être nettement plus simples avec des tests appropriés.
- Publiez tôt; n'ayez pas peur de différer des fonctionnalités.
- Conservez une liste de choses à faire tout au long du processus, et tentez de les éliminer dès leur apparition. Ce n'est pas si simple que cela en a l'air !
- Vous saurez que vous avez construit quelque chose de valable lorsque vous visiterez votre propre site et que vous y trouverez quelque chose d'intéressante.

## L'avenir

---

En guise de conclusion, la plateforme que j'ai construite est hautement adaptable à d'autres types de sites d'actualités (sports, technologies, sources d'informations générales). Le seul problème est que les communiqués de presse ne sont pas disponibles pour ces domaines depuis une source centralisée comme c'est le cas pour l'information scientifique ; il n'est pas exclu que j'en vienne à rechercher du capital-risque pour souscrire une licence Associated Press afin de construire un site similaire à Newsvine (y a-t-il un VC dans la salle ?). J'évalue également l'idée d'autoriser les commentaires anonymes sur le site avec [Mollom](#). Et, comme de juste, j'ai d'autres projets à lancer sans rapport avec celui-ci, et mon doctorat à terminer ! J'espère que vous avez apprécié cette étude de cas, et que vous aimerez Eurêka Science News ! Pour toute question, contactez-moi ;- ) Et pour finir, mes remerciements tout particuliers à [funkyhat](#) qui m'a incroyablement aidé pour la rédaction de cette étude de cas.

© 2000-2008 Michael Imbeault ([Fireang3l](#)). Placé sous licence Creative Commons [CC-BY-SA 2.0](#)