Migration SEO et redirections Apache : ce qu'il faut savoir

Par Aymeric Bouillat

Domaine :	Recherche	**Référencement**
Niveau :	Pour tous	**Avancé**

Une migration de site avec modification des URL est toujours un projet sensible, notamment en termes de SEO, afin de garder l'acquis de référencement des anciens intitulés. Pour ce faire, il est nécessaire de mettre en place des différentes règles techniques de réécriture et de redirections, adaptées à vos problématiques. Voici un certain nombre de conseils à suivre pour effectuer une migration en évitant un maximum de stress lors du changement des intitulés d'URL en configurant au mieux votre serveur Apache.

Lors d'une refonte de site avec modification de la structure des URL, le SEO ne doit pas être négligé, bien au contraire. Une migration mal préparée peut s'avérer catastrophique en termes de trafic, si des redirections 301 sur des URL avec du potentiel SEO ont été omises par exemple.

En termes de performance, une migration SEO peut également avoir un impact non négligeable en cas de fort volume d'URL à traiter : cet article va vous détailler les différents points à analyser pour que votre migration soit la plus transparente possible, pour l'internaute, mais également pour Googlebot et les autres crawlers .

Avec plusieurs dizaines de milliers (ou millions) d'URL à migrer, de nombreux challenges sont à gérer :

- Identifier l'anatomie des URL de l'ancien / nouveau site, afin de créer les règles de redirection les plus génériques via des expressions régulières quand cela est possible ;
- Organiser les règles par ordre d'importance ;
- Générer les tables de correspondance entre les anciennes et les nouvelles références produits/articles ;
- Décider des URL à migrer *versus* celles à ne pas migrer (historique négatif pour Google, URL avec netlinking discutable) ;
- Tester en pré production la bonne configuration des règles de redirection sur un fort volume d'URL.

L'organisation des règles Apache et des différentes directives vous permettront de préparer au mieux votre migration, tout en sensibilisant votre hébergeur ou DSI sur les optimisations possibles grâce aux directives Apache. Et ce, afin de gagner en rapidité et en souplesse, tout en limitant le duplicate content et la conservation des URL non pertinentes.

Migration SEO ? Le même principe qu'un déménagement

Une migration SEO pourrait être comparée à un déménagement dans la vie de tous les jours. Quand vous changez d'habitation, il vous faut retrouver toutes vos affaires, et cela passe souvent par du tri. Il s'agit de la partie la plus délicate de la migration, pour ne rien perdre mais également pour ne garder que ce qui est nécessaire.

On déballe rarement la totalité des cartons en une journée après le déménagement. Si vous n'avez pas noté dans un cahier ce que contient chaque carton, vous risquez de passer beaucoup de temps à retrouver les objets que vous utilisez régulièrement : c'est pour cela que les redirections 301 sont utiles : elles permettent à Google de retrouver vos contenus les plus pertinents. Classer vos différentes affaires (URL) vous permettra également d'y accéder plus vite mais pour cela, il faut de l'organisation : classer vos URL sera un gain de performance et de gestion de ces dernières pour la suite.

Un déménagement est également l'occasion de faire du tri : les objets qui vous encombrent et que vous n'utilisez jamais, autant s'en débarrasser, cela fera plus d'espace pour le reste et vous y verrez plus clair dans votre nouveau logement : une migration SEO ne nécessite pas de rediriger l'intégralité des URL, au contraire. C'est parfois l'occasion de faire du tri, pour rendre plus accessible ce qui vous sert le plus. Google Webmaster Tools sera votre meilleur allié pour effectuer ce tri d'URL.

Enfin, vous pouvez également retrouver des objets ayant exactement la même utilité. Ces « objets en double » (URL dupliquées) prennent de l'espace et il est parfois préférable de garder uniquement ceux qui fonctionnent le mieux : il est important de limiter au maximum le contenu dupliqué qui peut être généré avec les paramètres d'URL d'une ancienne structure de site vers une nouvelle. C'est ce que nous verrons avec l'analyse de logs et les conditions de réécriture.

Organisation des URL

Ordre des règles

Comme nous l'avons vu dans la Lettre Recherche & Référencement du mois dernier (Avril 2015), les fichiers .htaccess sont gourmands en ressources, puisqu'ils sont appelés et interprétés à chaque requête de fichier sur le serveur. Dans la mesure du possible, il vaut mieux placer l'ensemble des directives liées à une migration dans la configuration du serveur Apache (avec l'ajout de la directive AllowOverride None pour désactiver l'utilisation de fichiers .htaccess). Cela vous permettra de gagner en performances.

Par ailleurs, l'ordre des règles a également son importance : sur plus de 10 000 règles de redirections post – migration par exemple, si la règle dynamique qui concerne les pages produits / catégories (pages souvent les plus visitées par les internautes) se trouvent en fin de fichier de configuration, l'interprétation des règles qui la précède sera effectuée sur chaque requête, ce qui est loin d'être optimal en terme de rapidité d'exécution pour le serveur Apache.

Il est préférable de placer les règles les plus utilisées en début de fichier de redirections, afin de limiter l'interprétation d'un trop grand nombre de règles, correspondants aux URL les moins sollicitées. Afin d'organiser les règles par nombre de requêtes mensuelles dans un contexte d'optimisation, vous pouvez vous référer à plusieurs données :

- Clics Google Webmaster Tools ;
- Nombre de hits serveur (analyse de logs) ;
- Nombre d'URL différente / format d'URL.

Cela permettra de limiter la charge serveur, pour que les règles les plus fréquentes se trouvent en tête du fichier de redirections.

Dans le cas de règles dynamiques, il est conseillé de travailler par format d'URL (*pattern*). La figure 1 ci-dessous montre le type de tri qui a été effectué pour un grand site e-commerce afin de mettre en place les règles de redirections par nombre de hits. Dans cet exemple, les règles seront ordonnées de la façon suivante :

1. Règle dynamique pour les catégories de niveau 2 et 3 (clics Google Webmasters Tools élevé) ;
2. Règle dynamique pour les fiches produit (hits Googlebot élevé) ;
3. Pages édito (clics + crawl inférieur au format d'URL des fiches produits) ;
4. Page inscription (nombre de visites Utilisateur + nombre de hits Googlebot inférieur à celui des pages éditoriales).

Segmentation des règles par répertoire

Dans le cadre de redirections avec une segmentation des URL par répertoire, prenons comme exemple une segmentation par langue : /fr/, /en/, /de/. Il est possible regrouper les directives par dossier, pour que les règles concernant la langue FR ne soient exécutées que quand le répertoire /fr/ ou un fichier contenu dans ce répertoire est appelé. Pour cela, il faut placer les règles dans des sections <Directory « chemin système du repertoire »>Directives</Directory>.

Exemple :
http://www.monsite.com/fr/category/23 redirigera vers http://www.monsite.fr/bougies

http://www.monsite.com/en/category/24 redirigera vers http://www.monsite.co.uk/perfumes

Type de page	Clics GWT	Impressions GWT	Nombre d'URL différentes	Hits Crawl	URL Exemple
Fiche produit	108439	996784	260697	995198	http:/
Inscription	1545	12400	1	80783	http:/
Catégorie de niveau 2, catégorie de niveau 3	1837233	24764833	14144	76786	http:/
Editorial	63634	642650	321	53054	http:/

Fig.1. Tri des URL pour un site e-commerce.

```
<Directory /var/www/monsite/fr/>
RewriteRule ^category/23$
http://www.monsite.fr/bougies/
[R=301,L]
RewriteRule ^category/24$
http://www.monsite.fr/parfums/
[R=301,L]
#autres règles…
</Directory>
<Directory /var/www/monsite/en/>
RewriteRule ^category/23$
http://www.monsite.co.uk/candles/
[R=301,L]
RewriteRule ^category/24$
http://www.monsite.co.uk/perfumes/
[R=301,L]
#autres règles…
</Directory>
```

Cela facilitera également la gestion des règles par la suite, grâce à une meilleure lisibilité dans les fichiers de configuration (configuration Apache ou fichier d'hôte virtuel).

Dans le cas où votre accès à la configuration du serveur est limité, et pour éviter trop d'aller-retour avec votre DSI, l'utilisation d'un fichier .htaccess est recommandée. Mais celui-ci peut vite devenir conséquent et lourd à gérer si le site à migrer contient un nombre trop important d'URL.

Plutôt que de créer un répertoire par langue à la racine, avec un fichier .htaccess pour chacun d'entre eux, il est possible d'utiliser des alias pour segmenter les redirections par répertoire : les alias évitent la création de trop nombreux répertoires à la racine, et permettront dans notre cas de regrouper l'ensemble des règles dans un répertoire dédié aux redirections :

Création d'Alias Apache vers un seul répertoire dédié aux redirections, à placer dans la configuration du serveur :
Exemple avec un répertoire /redir/ à la racine du site :

```
Alias /fr/ /var/www/redirections/fr/
Alias /en/ /var/www/redirections/en/
Alias /de/ /var/www/redirections/de/
```

Etc.

Avec un fichier .htaccess par répertoire (ex : fichier .htaccess dans /var/www/redirections/fr/):

```
RewriteEngine On
RewriteBase /fr/
RewriteRule ^category/23/(.+)$
http://www.monsite.fr/bougies/$1
[R=301,L]
```

Cela vous permettra d'avoir un fichier .htacces allégé pour le nouveau site, tout en traitant vos redirections de l'ancien site vers le nouveau, et ce, en limitant les interprétations multiples des règles pour toutes les requêtes.

RewriteMap ou le plan de redirection amélioré

Lors d'une migration, une directive Apache est particulièrement utile : RewriteMap.

Plutôt que de répéter plusieurs RewriteRule les unes en dessous des autres, cette directive permet de mettre en correspondance des URL *via* un fichier texte (anciennes *versus* nouvelles) sous la forme de clé | valeur, qui seront gérées via une seule directive RewriteMap.

Cette directive peut donc vous permettre de gérer plusieurs milliers de redirections, via un seul fichier de correspondance : plusieurs formats sont

supportés dont le .txt (texte), .dbm (fichier avec index qui sera plus rapide qu'un simple fichier texte), voire même directement une base de données SQL. C'est dans ce cas que les RewriteMap peuvent s'avérer très pratiques car elles ne demanderont pas de mises à jour des fichiers de configuration, la directive allant directement se servir dans une base de données du CMS par exemple (A noter que l'installation et le paramétrage du module Apache mod_dbd sera nécessaire dans ce cas de figure). Attention toutefois à la vitesse d'exécution des requêtes SQL, pour lequelles une mise en cache sera vivement recommandée.

Définition d'une table de correspondance

Supposons que notre ancien format d'URL soit le suivant pour les fiches produits d'un site e-commerce : /fr/produit/identifiantproduit/infos.html
ex : /fr/produit/2435/infos.html

Le nouveau format d'URL SEO-friendly aura cette forme : /categorie/nomduproduit-identifiantduproduit.html, exemple : /bougies/bougies-parfum-vanille-2435.html

Il est alors possible de placer toutes vos correspondances dans un fichier texte (paire clé valeur séparée par un espace):
346 coiffure/peigne-a-cheveux
347 canape/canape-en-cuir-rouge
2435 bougies/bougies-parfum-vanille
etc.

On déclare ensuite ce fichier texte avec la directive :
RewriteMap produits txt:/var/www/monsite/produits.txt

Pour générer un fichier .dbm qui sera plus rapide à lire par le serveur Apache (index), voici la commande à utiliser :
httxt2dbm -i produits.txt -o produits.map

puis déclaration du fichier das la configuration d'Apache :
RewriteMap produits dbm:/var/www/monsite/produits.map

Règle de redirections de la table de correspondance

L'intérêt de gérer l'ensemble des redirections *via* un fichier texte indépendant est de pouvoir mettre à jour rapidement les correspondances (disponibilité produit par exemple) *via* une simple requête vers une base de données, sans aucune intervention dans le fichier de configuration Apache (attention toutefois, la déclaration des RewriteMap ne fonctionne pas dans les fichiers .htaccess).

La règle permettant de gérer les correspondances entre les anciennes et nouvelles URL, avec des correspondances établies au préalable par l'équipe technique en charge du projet, sera la suivante :
RewriteRule ^fr/produit/([0-9]+)/infos\.html$ /${produits:$1}-$1.html [R=301,L]

Explications : Dans cette règle, on récupère l'ID du produit sur les URL commençant par fr/produit/ et se terminant par infos.html (format d'URL produit), en définissant un ensemble de caractères numérique ([0-9]+) (chiffres de 0 à 9 répétés au moins une fois), avec des parenthèses pour isoler l'identifiant produit.

On extrait ensuite la valeur qui correspondant à cet ID Produit (${produits:$1}) dans le fichier RewriteMap déclaré précédemment (RewriteMap « produits » définit dans le fichier de configuration Apache).

L'URL /fr/produit/346/infos.html sera donc redirigée en 301 vers /coiffure/peigne-a-cheveux-346.html

Il est également possible de définir une valeur par défaut quand aucune correspondance n'est trouvée (Ex : Identifiant de produit inconnu). Il faudra dans ce cas traiter l'affichage côté applicatif, en fonction de la valeur par défaut qui peut être renvoyée (non-trouve dans cet exemple) :

```
RewriteRule ^fr/produit/([0-9]+)/infos\.html$ /${produits:$1|non-trouve}-$1.html [R=301,L]
```

Gestion des chaînes de paramètres lors d'une migration

Comportement par défaut d'Apache

Par défaut, les règles de redirection Redirect ou RedirectMatch conservent les chaînes de paramètres (partie de l'URL se trouvant derrière le caractère « ? »). Pour intervenir sur cette chaîne et modifier ou supprimer son contenu d'une URL, il faut passer par des règles de réécriture RewriteRule (et RewriteCond pour en analyser leur contenu).

Malheureusement, cette étape n'est pas assez approfondie lors des migrations SEO : on retrouve souvent d'anciens paramètres d'URL présents sur les URL du nouveau site, paramètres qui ne sont parfois plus utiles. Bien que l'élément canonical <link rel="canonical" ..> permette de limiter le contenu dupliqué qui serait généré par ces URL, cela posera d'autres problèmes de crawl à terme, car Google conservera dans son index une trace de ces URL orphelines.
Les actions possibles afin de supprimer ces chaînes de paramètres sont détaillées dans la Lettre Recherche & Référencement du mois dernier (Avril 2015).

Au sujet de la détection des paramètres / valeurs que cette chaîne peut contenir, il faut utiliser une condition de réécriture pour chaque règle de redirection afin d'en déterminer le contenu. Cela peut rapidement compliquer la gestion des directives :

```
#Si la chaîne de paramètres est égale à "parametre1=valeur", on effectue des redirections spécifiques en fonction de l'URL demandée
RewriteCond %{QUERY_STRING} ^parametre1=valeur$
RewriteRule ^ancienneurl /nouvelleurl? [L,R=301]
RewriteCond %{QUERY_STRING} ^parametre1=valeur$
RewriteRule ^ancienneurlbis /nouvelleurlbis? [L,R=301]
RewriteCond %{QUERY_STRING} ^parametre1=valeur$
RewriteRule ^ancienneurlter /nouvelleurlter? [L,R=301]
```

Afin d'éviter une répétition des conditions de réécriture en fonction de la chaîne de paramètres pour un ensemble de règles, une fonction d'Apache va permettre une simplification des règles associées à la même condition.

Définir une condition qui s'applique à plusieurs règles de réécriture

La directive RewriteCond ne s'applique qu'à la règle RewriteRule qui la suit immédiatement. Ainsi, si vous souhaitez qu'une directive RewriteCond s'applique à plusieurs règles de type RewriteRule, il est possible d'utiliser une méthodologie qui consiste à inverser ces conditions et à ajouter une RewriteRule avec le drapeau [Skip] d'Apache.

Pour reprendre l'exemple ci-dessus, nous souhaitons que les règles soient exécutées uniquement quand la chaîne de paramètres contient « parametre1=valeur » (exemple : /ancienneurl?parametre1=valeur). Nous allons donc demander à Apache de sauter l'exécution de ces règles si les URL ne contiennent pas la chaîne demandée :

```
RewriteCond %{QUERY_STRING} !^parametre1=valeur$
RewriteRule . – [S=3]
RewriteRule ^ancienneurl /nouvelleurl? [L,R=301]
RewriteRule ^ancienneurlbis /nouvelleurlbis? [L,R=301]
RewriteRule ^ancienneurlter /nouvelleurlter? [L,R=301]
```

Si la chaîne de paramètres ne contient pas parametre1=valeur, alors on saute les 3 règles de redirections qui suivent (S=3). Les règles de réécriture ne seront donc exécutées que si la bonne chaîne est trouvée.

Cette fonctionnalité s'avère également pratique pour découper les règles en différents blocs afin d'accélérer l'interprétation de ces dernières.

Aller plus loin avec les paramètres d'URL

En fonction des différents liens internes et externes d'un site, on peut rapidement se retrouver avec de nombreuses combinaisons de paramètres dans les URL. Afin de pouvoir correctement cibler chacune de ces combinaisons, pour isoler les paramètres nécessaires sur le nouveau site lors d'une migration SEO, il est possible de lister l'ensemble de ces combinaisons de paramètres connues de Google grâce aux logs.
Voici un article à ce sujet : http://www.yapasdequoi.com/scripts/3560-combinaison-parametres-durl-les-logs-script.html

Cela vous permettra de mettre en place vos redirections 301 en fonction de vos paramètres d'URL connus de Google (le crawl d'un site n'étant pas nécessairement exhaustif)

Nettoyage d'anciennes URL avec les 410

Comme cela a été mentionné en début d'article, le fait de migrer un site n'implique pas nécessairement de rediriger l'intégralité de vos URL : certaines n'ont pas de poids (contenu de faible qualité, mauvais maillage interne) et n'ont aucun potentiel SEO. Plutôt que d'alourdir votre fichier de redirections 301 avec des redirections qui ne seront appelées que très rarement, allégez-le.

La migration est donc une bonne occasion de se séparer de ce type d'URL non pertinentes. Pour cela, l'utilisation du drapeau [G] d'Apache est la solution la plus adaptée. Cela

renverra pour les URL ciblées un status HTTP de type 410 : Gone.

Exemple :

```
#Supression des URL de type /search
RewriteRule ^search=.*$  - [G]
```

Google supprimera les URL qui commencent par /search de son index à terme. Attention, cela ne veut pas dire qu'il n'ira plus les crawler, il ralentira son crawl sur ces URL. Une restriction dans le fichier robots.txt s'imposera donc par la suite, quand la désindexation sera complète.

De la même manière, effectuer le ménage dans vos erreurs 404 avant la migration vous permettra d'y voir plus clair avec l'arrivée de nouvelle 404 post-migration, notamment dans l'interface de Google Webmaster Tools .

Bonne migration SEO !

Aymeric Bouillat, *consultant SEO Resoneo (http://twitter.com/aymerictwit ou http://www.yapasdequoi.com)*