Bonjour à tous !

Un système de gestion de contenu (SGC), plus connu sous le nom anglais de CMS (pour Content Management System) est un logiciel ayant pour but de faciliter ou d'automatiser les tâches de conception et de mise à jour d'un site Internet.

> *Intéressant, mais moi j'aimerais faire un site web… une définition, ça ne m'aide pas vraiment !*

Cela tombe bien car l'objectif de ce cours est de répondre le plus concrètement possible aux questions qui se posent assez rapidement lorsque l'on s'intéresse aux CMS :

- Est-ce difficile à mettre en place ?
- Dois-je savoir programmer ?
- Quel CMS choisir ?

Allez, c'est parti ! Nous allons commencer par un rappel des notions de base à connaître sur le Web quand on parle du fonctionnement de CMS, puis nous enchaînerons sur leurs différentes caractéristiques.

Prérequis : aucun

Objectifs pédagogiques :

- Comprendre le fonctionnement des CMS
- Explorer leurs différentes fonctionnalités
- Découvrir leurs avantages et leurs inconvénients
- Choisir un CMS adapté à vos usages

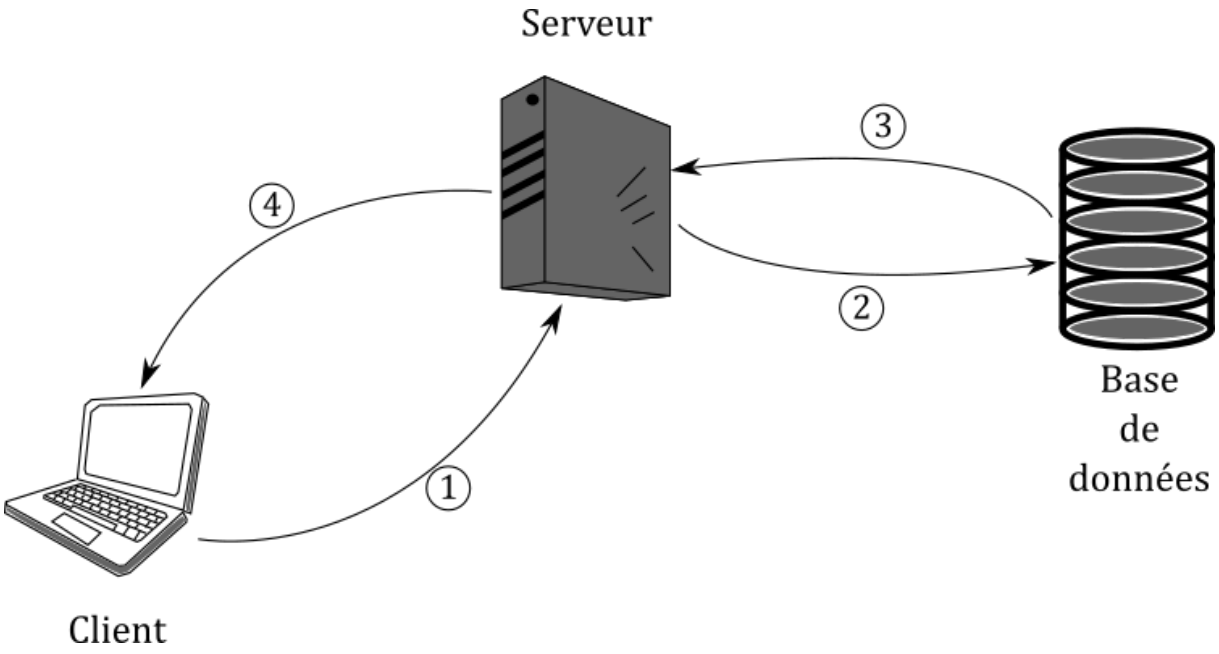
Qu'est-ce qu'un CMS ?

Les notions clés

Avant de parler des CMS en eux-mêmes, je vais vous expliquer ou vous réexpliquer quelques bases sur le fonctionnement des sites web en général. Cela vous sera utile pour bien suivre le cours.

Client et serveur

Je vais commencer par vous présenter le fonctionnement général d'un site web. Représenté sous le forme d'un schéma, un site web fonctionne comme cela :



L'architecture d'une page web

Ce que l'on appelle "client", c'est l'ordinateur (ou téléphone/tablette) qui vous sert à vous connecter à Internet. Le client effectue une requête (lorsque vous cliquez sur un lien ou quand vous tapez une adresse directement dans votre navigateur) pour obtenir une page web.

La base de données est l'espace où sont stockées les informations nécessaires à afficher la page web demandée (textes, images...).

Enfin, le serveur est la machine qui permet de faire le lien entre les deux.

- D'abord, le serveur reçoit une demande de la part du client (qui pourrait se traduire par : "Envoie moi la page xxx"). **(1)**
- Il récupère les informations nécessaires dont il a besoin pour construire la page sur la base de données. **(2)** et **(3)**
- Il envoie (on dit aussi qu'il sert) la page demandée au client **(4)** et le client interprète ce qu'il reçoit pour l'afficher. À partir de là, la page s'affiche sur votre écran ! :)

> Les mots client et serveur désignent tout aussi bien la machine (l'ordinateur) que le logiciel qui tourne sur cette machine. Dans la suite du cours, nous nous intéresserons uniquement à la partie logicielle.

Maintenant, voyons un peu plus en détails les différentes étapes.

Côté client

Comme vous le savez peut-être, une page web est généralement construite avec du HTML/CSS/JavaScript :

1. HTML est une norme qui permet de structurer un document (titres, parties, header, footer...). C'est le seul élément indispensable à la création d'une page web.
2. CSS est un formalisme qui permet de mettre en forme un document (couleur, police, taille de caractère...).
3. JavaScript est un langage de programmation permettant d'animer le contenu d'une page web (réagir à une action de l'utilisateur, afficher un élément évolutif, etc.).

Lorsque vous demandez à afficher une page web, vous demandez au serveur de vous envoyer un ou plusieurs fichier(s) HTML, JavaScript et CSS. Une fois reçus, ces fichiers seront utilisés par votre navigateur (Chrome, Firefox, Opéra...) pour vous afficher la page demandée.

Côté serveur

À la différence du HTML qui est standard côté client, il existe de très nombreux langages pour programmer un serveur web (PHP, JSP, ASP, Java...).

En fait, presque tous les langages permettent de faire un serveur web ! Cependant certains, tels que PHP, sont dédiés à la création de serveur web.

> La plupart des CMS sont écrits en PHP.

Quel que soit le langage utilisé, l'objectif du serveur reste le même lors d'une requête d'affichage de page.

1. D'abord, il va interpréter le message envoyé par le client (sous la forme d'un lien, d'une adresse, d'un formulaire...).
2. Il va ensuite récupérer les informations sur la base de données.
3. Cela va lui permettre de construire la page en assemblant le HTML, le CSS et le JavaScript.
4. Enfin, il pourra envoyer la page au client.

Côté base de données

Sans rentrer trop dans le détail, la plupart des bases de données utilisées dans le Web aujourd'hui sont des bases de données relationnelles utilisant le langage SQL (Structured Query Language, soit langage de requête structurée).

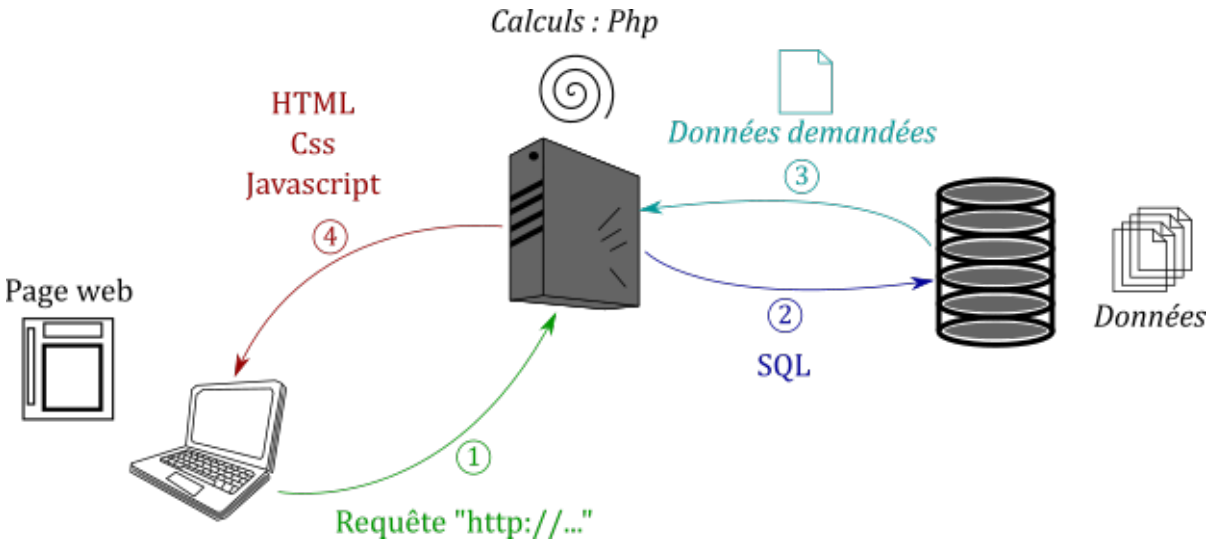
Comme son nom l'indique, SQL est un langage dédié au stockage et à la récupération d'informations sur (ou à partir) d'une base de données.

Il existe diverses variantes de ce langage en fonction de la base de données utilisée (MySQL, PostgreSQL, Oracle...) mais elles sont relativement minimes et nous ne les considérerons pas dans ce cours.

Pour faire simple, la base de données a deux utilités :

- sauvegarder des informations envoyées par le serveur ;
- envoyer au serveur des informations préalablement enregistrées.

En reprenant le schéma initial, cela donne :



Architecture d'une page web

Site dynamique et site statique

> *Tous les sites ont-ils besoin d'une base de données et d'un serveur ?*

Non. Il est tout à fait possible de se passer de base de données et/ou de serveur. Comme je l'ai dit, seul le HTML est indispensable à la création d'une page web.

Sans serveur ni base de données, la page sera **statique** : c'est-à-dire qu'elle sera toujours la même quel que soit le moment où vous la consultez.

Avec une base de données et un serveur, il est possible de créer une page web **dynamique** : une page web qui peut évoluer avec les utilisateurs (comme une page Facebook) ou au cours du temps (comme un blog par exemple).

Front-office et back-office

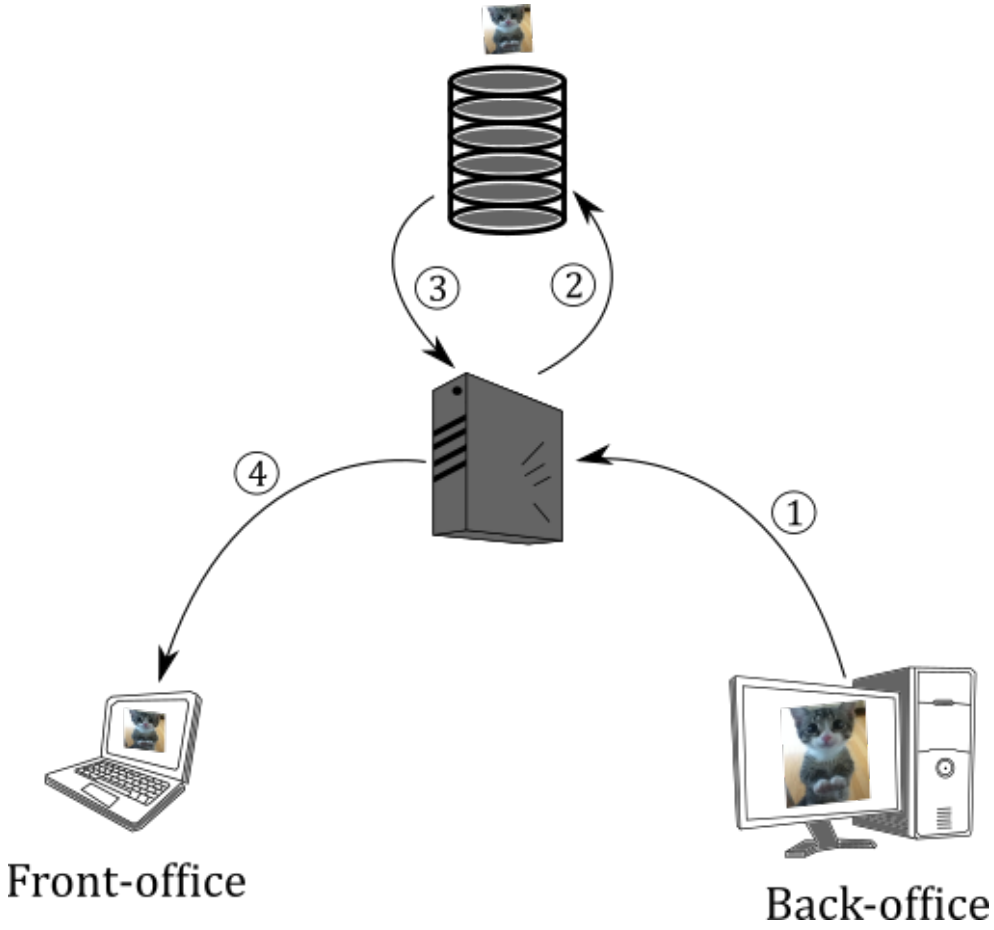
Dernier point de connaissance à aborder : la différence entre ce que l'on appelle le front-office et le back-office ("boutique" et "arrière-boutique" en français).

Comme nous venons de le voir, le rôle du serveur, lorsque l'on demande à afficher une page, est d'aller chercher des informations sur la base de données. D'où la question légitime : d'où viennent ces informations qui sont présentes dans la base de données ?

Si vous avez lu le titre de cette section, vous devez vous douter de la réponse : du back-office !

On appelle back-office la partie du site web qui permet de modifier/ajouter/retirer des informations sur notre site. La partie d'un site web où sont affichées ces informations, c'est le front-office. :)

Un schéma pour visualiser ça :



Front-office et back-office d'un site web

- Le back-office envoie au serveur une information et le serveur stocke cette information dans le base de données. **(1)** et **(2)**
- Le front-office demande l'affichage de la page au serveur, qui va chercher dans la base les données nécessaires. **(3)** et **(4)**

> *On ne peut pas mettre d'informations dans la base de données directement depuis le front-office ou sans back-office ?*

Si.

Par exemple, lorsque vous laissez un commentaire sur un blog, ce commentaire est stocké dans la base de données pour pouvoir être affiché aux autres internautes. Pourtant, vous n'êtes pas passé par le back-office !... mais vous avez bel et bien modifié le contenu de la base de données.

En fait, la différence entre front-office et back-office n'est pas une différence technique, c'est une différence d'usage. Le back-office est la partie du site (généralement privée et sécurisée par un mot de passe) qui permet à l'administrateur du site d'administrer son site web : ajouter des pages, modifier du contenu, gérer les utilisateurs, etc. Le front-office, lui, est à destination des usagers du site : il est visible par les internautes.

Bref historique du Web

Un petit historique pour vous expliquer l'apparition et la démocratisation des CMS, et on passe au cœur du sujet ! :)

Avant les CMS... le HTML !

Le Web est apparu aux alentours des années 1990 avec l'invention du HTML, par Tim Berners-Lee qui travaillait au CERN (Conseil européen pour la recherche nucléaire) à l'époque. L'objectif était de permettre la diffusion de documents sur le réseau mondial (Internet).

> Ne confondez pas le Web avec Internet, ce ne sont pas des synonymes ! :) Internet est un projet initialement militaire, datant des années 1960. Pour plus d'informations à ce sujet, vous pouvez consulter le cours Comprendre le Web.

Le format HTML a évolué au cours des années. En 1990, on pouvait seulement formater des documents (titres, tableaux...) et créer des liens entre documents grâce aux liens hypertextes. Depuis 2014, nous en sommes au HTML5 et le format est beaucoup plus riche !

L'arrivée du CSS et des langages web

Le CSS a été ajouté en 1995 lors de la spécification du HTML 3 : il permet de mettre en forme les documents HTML.

La même année, le JavaScript a été introduit par le navigateur Netscape (l'un des tout premiers navigateurs).

Côté serveur, c'est en 1994 qu'est apparu le langage PHP (diffusé en 1995). Il a été inventé par Rasmus Lerdorf pour sauvegarder des informations sur son site web en le connectant à une base de données.

> *Et, du coup, les bases de données ?*

Les bases de données existent en fait depuis bien plus longtemps que le Web ! Elles ont commencé à apparaître vers le milieu des années 1960 pour évoluer et se perfectionner au fil des années.

Les derniers avancements majeurs dans le domaine des bases de données datent de 2010 avec l'apparition du NoSQL... mais on ne va pas s'étendre là-dessus dans ce cours. Pour l'instant, très peu de CMS utilisent cette technologie (à ma connaissance, il y en a principalement... un !).

> *Et les CMS dans tout ça ?*

Difficile de dire quel était le premier CMS à apparaître, mais on estime que c'était peu avant l'an 2000. Depuis, le nombre de CMS a fortement augmenté. Il en existe aujourd'hui plusieurs centaines, voire plusieurs milliers.

Les caractéristiques d'un CMS

Vous vous souvenez de la définition initiale d'un CMS ?

> Un CMS est un logiciel ayant pour but de faciliter ou d'automatiser les tâches de conception et de mise à jour d'un site Internet.

À partir de ce que vous avez appris dans les deux premiers chapitres, on peut maintenant préciser cette définition !

Un CMS est un logiciel – qui tourne sur un serveur web – permettant de créer, de façon simple, des pages web dynamiques. Le CMS gère automatiquement les tâches d'affichage des pages (front office) et de mise à jour du contenu (back-office et lien avec la base de données).

Autrement dit, les CMS gèrent à votre place tout ce qui permet de relier le client au serveur et le serveur à la base de données.

> *Donc, si j'ai bien compris, un CMS est un site web dynamique pré-construit qui dispose d'un back- office et qui automatise pour nous la gestion de l'affichage des pages et la relation avec la base de données.*
>
> *Il y a d'autres caractéristiques ?*

Oui ! La plupart des CMS disposent des autres caractéristiques ci-dessous.

Mise en place simplifié

Les CMS permettent de mettre en place un site web sans connaissances en programmation. Toutes les actions se font à travers des interfaces graphiques, et non avec des lignes de code.

Édition WYSIWYG de pages

> WYSIWYG est l'acronyme de What You See Is What You Get, ce qui veut dire en français "ce que vous voyez est ce que vous obtenez".

De la même manière que le logiciel Word vous permet de mettre en forme un texte de façon visuelle, les CMS vous proposent de mettre en forme vos pages web grâce à une interface graphique : vous n'avez pas à écrire vous-même le HTML que vous voulez afficher aux utilisateurs, les CMS s'occupent de le générer.

Séparation entre contenu et design

Dans un même site web, de nombreuses pages, si ce n'est toutes, ont un design similaire. Les CMS intègrent des outils permettant de ne pas avoir à refaire ce design pour chaque page : une fois le design du site choisi, il s'appliquera automatiquement à toutes les pages.

Ceci a deux avantages :

- vous pouvez récupérer des designs "touts faits" (que l'on appelle des **thèmes** ou des **templates**) et les appliquer à votre site directement ;
- vous pouvez modifier le design du site sans avoir à vous soucier de son contenu. Le nouveau design sera appliqué automatiquement.

Gestion des droits d'accès

Les utilisateurs peuvent avoir des rôles différents au sein d'un site web, et donc avoir le droit, ou non, d'effectuer certaines actions ou d'accéder à certaines pages.

Prenons l'exemple d'un forum :

- les utilisateurs non enregistrés peuvent lire mais pas poster de messages ;
- les utilisateurs enregistrés peuvent lire et poster des messages ;
- les modérateurs peuvent lire et poster des messages, et ils ont en plus la possibilité de modifier et de supprimer les messages des autres utilisateurs ;
- etc.

La possibilité d'effectuer ou non une action est gérée par ce qu'on appelle des **droits** (avoir le droit, ou non, d'effectuer une action).

Dans la plupart des CMS, cette gestion des droits existe et est simple à mettre en œuvre : il suffit de créer des groupes d'utilisateurs et de leur donner, ou de leur interdire, l'accès aux différentes parties du site (par exemple l'accès au back-office, à une partie "membres"...).

Extensions

Même s'il ne s'agit pas à proprement parler d'une caractéristique, les extensions, aussi appelées par leur nom anglais **plugins**, sont désormais un incontournable de tous les CMS.

Les **extensions** ne sont pas directement incluses dans le CMS (même si certaines sont installées par défaut !) : il est possible de les installer séparément selon vos besoins. Ceci permet d'avoir un cœur de CMS contenant uniquement les fonctionnalités indispensables (celles listées ci-dessus notamment), tout en augmentant fortement la flexibilité de l'outil.

Il est souvent possible de développer vous-même une extension et l'utiliser sur votre site.

Avantages et inconvénients

Les CMS sont souvent avantageux, mais attention, ils ne sont pas une solution miracle ! Selon la nature de votre projet, choisir un CMS pourra ne pas être pertinent. À vous de peser le pour et le contre ! :)

Avantages

Comme nous venons de le voir, les CMS ont de nombreux avantages.

- Ils sont **accessibles** : pas besoin de connaître un langage de programmation pour pouvoir les utiliser ou les installer.
- Ils sont **rapides à mettre en place** : quelques heures pour les plus complexes, quelques minutes pour les plus simples.
- Ils font **gagner du temps** : développer tout un site web à partir de zéro – plutôt qu'utiliser des outils préconçus – est chronophage.
- Ils sont **soutenus par une communauté** : c'est le meilleur moyen d'avoir des outils performants et variés, et d'obtenir de l'aide. Nous reviendrons plus en détails là-dessus dans la deuxième partie de ce cours.
- Ils sont **faciles à maintenir** et à faire évoluer : le Web évolue, les CMS répercutent ces évolutions et permettent d'avoir un site qui évolue également (usage, technologie, design, ergonomie, etc.).

Inconvénients

Cependant, comme toute solution technique, les CMS ont également des inconvénients intrinsèques.

Voici les plus notables et les plus répandus.

- Ils **manquent de flexibilité** : la grande majorité des CMS proposent énormément de fonctionnalités, néanmoins il est souvent complexe et coûteux d'ajouter celles qui ne sont pas initialement prévues.
- Ils sont **moins performants** : la généricité et la complexité des CMS les rend, à qualité égale, moins performants qu'un site construit sans CMS.
- Ils sont **plus susceptibles d'être attaqués** : comme tous les sites utilisant le même CMS partagent un code source commun, il est nettement plus aisé de pirater un CMS, surtout s'il est mal protégé ou implémenté. Par ailleurs, le temps investi par un pirate pour attaquer un site web peut être rentabilisé sur d'autres sites construits avec le même CMS.
- Ils sont **difficiles à migrer** : changer de CMS est souvent beaucoup plus long et complexe que de faire évoluer un site web construit sans CMS.

Dans certains cas, les CMS peuvent pâtir d'autres inconvénients.

- Ils sont **moins faciles à référencer** : la structure des pages et les redondances limitent souvent l'adéquation avec les règles du référencement.
- Ils sont **moins stables** : la complexité des CMS les rend parfois moins stables – toujours à qualité égale – qu'un site construit sans CMS.

Bilan

Avant de passer à la suite, voici un tableau récapitulatif des avantages et inconvénients de CMS.

Avantages	Défauts
Accessibilité	Sécurité
Mise en place	Flexibilité
Gain de temps	Performances
Communauté	Stabilité*
Maintenabilité	Référencement*
Évolvabilité	

Les défauts marqués d'un astérisque (*) sont corrigés ou limités si vous utilisez un bon CMS mis à jour récemment, de base ou en ayant recours à des plugins.

Comment choisir son CMS ?

Open source ou propriétaire ?

Parmi les nombreux CMS disponibles, vous trouverez des CMS open source et des CMS propriétaires. Chaque solution a ses qualités et ses défauts : en fonction de vos besoins, le choix pourra être différent.

Open source

Un CMS, et c'est vrai pour n'importe quel logiciel, est **open source** si :

- son code source est disponible d'accès – sans frais supplémentaires – à tout le monde sans restriction ;
- tout le monde peut le modifier et le redistribuer librement ;
- il ne limite pas le contenu qui peut être présenté.

Par définition, ces CMS sont gratuits et sont développés de façon communautaire. Pour ceux qui veulent des détails sur les critères des logiciels open source, vous pouvez trouver les dix critères à cette adresse : http://opensource.org/osd.html

Opter pour un CMS open source a différents avantages.

- Ils disposent généralement d'une communauté plus large.
- Ils sont gratuits et le support, fait par la communauté, l'est également.
- Ils possèdent le plus souvent de nombreux plugins, développés par la communauté, qui peuvent pallier certains manques ou faiblesses.
- Il est toujours possible de les faire évoluer (vous avez accès au code source).

Si vous choisissez un logiciel open source, vous risquer de rencontrer principalement deux inconvénients.

- La publication du code source rend le CMS plus vulnérable aux attaques.
- Le support, s'il est gratuit, est plus hasardeux puisqu'il repose sur le bonne volonté de la communauté.

Propriétaire

Un CMS, et là encore c'est vrai pour n'importe quel autre logiciel, est dit "propriétaire" si ses conditions d'utilisation limitent son utilisation, son étude, sa modification, sa duplication ou sa diffusion.

Ces CMS peuvent être payants, mais pas forcément. En revanche, le code source n'est pas accessible et ils sont produits par une société qui maîtrise tous les aspects du développement et de la distribution.

Les solutions propriétaires ont également des avantages.

- À qualité égale, elles sont moins vulnérables aux attaques.
- Le support est assuré par des professionnels du CMS, il est donc généralement plus ciblé et précis.

Elles ont également des inconvénients.

- L'utilisateur est dépendant de l'entreprise et de ses choix.
- Il est généralement plus difficile de migrer (changer de CMS) à partir d'un CMS propriétaire.
- Le CMS peut "mourir" : si la société cesse de le développer, il n'évoluera plus.
- Il est parfois payant, même si c'est assez rare pour les CMS.

Pour découvrir les CMS, ou pour un site personnel, il est largement préférable d'opter pour une solution open source : moins cher, moins contraignant, plus flexible...

Dans le cadre d'un projet de plus grande envergure, le choix d'un CMS propriétaire est parfois intéressant – même s'il doit être mûrement réfléchi. Cela dépend principalement des fonctionnalités que l'on recherche. C'est justement l'objet du chapitre suivant !

Un CMS, pour quel usage ?

Avant de vous lancer dans la création d'un site web, il faut premièrement et principalement vous poser la question de **l'usage que vous voulez en faire**. Pour choisir un CMS, ou au contraire pour choisir de ne pas en utiliser un, il est nécessaire d'avoir une idée très précise de ce que vous souhaitez obtenir et des contraintes que vous subissez.

Fonctionnalités

Les CMS offrent un ensemble de fonctionnalités – certaines basiques, d'autres plus avancées – mais s'ils ne proposent pas une fonctionnalité, il est généralement plus coûteux de l'ajouter au CMS que la construire à partir de rien.

On pourrait penser que si un CMS couvre 75% des fonctionnalités recherchées, il est nécessairement plus intéressant de l'utiliser que de développer entièrement un site : il y aurait seulement 25% à développer, contre 100% sans un CMS ! En réalité, ce n'est pas forcément le cas : il est important de bien calculer le coût nécessaire pour ajouter la fonctionnalité manquante. Souvent, cela vous prendra plus de temps que de développer tout un site sans CMS !

> Mais pourquoi ? Ce n'est pas logique !

Effectivement, intuitivement il est logique de penser que si 75% du travail a déjà été fait, il sera plus facile de faire les 25% restant que de tout refaire à partir de zéro. Il est difficile d'expliquer pourquoi sans rentrer dans les détails techniques... mais avec une analogie ça devrait pouvoir se faire ! :) Imaginez une maison. Celle que vous voulez, avec ses étages, sa forme, et toutes ses particularités.

À votre avis, est-il plus facile de la fabriquer à partir de rien ou de prendre une autre maison radicalement différente et de la transformer ? La plupart du temps, vous verrez que transformer l'existant est beaucoup plus chronophage !

C'est exactement la même chose pour les CMS. Si le CMS est proche, dans sa conception, de ce que vous voulez faire, il sera pertinent de l'utiliser pour développer les 25% manquants. Dans le cas contraire, il vaudra mieux repartir à zéro ou choisir un autre CMS plus adapté.

L'objectif dans le choix du CMS (ou celui de ne pas avoir recours à un CMS) réside dans la couverture entre les fonctionnalités que proposent le CMS et celle dont vous avez besoin.

Contraintes non fonctionnelles

Au-delà des fonctionnalités que vous souhaitez mettre en place sur votre site web, il faut aussi prendre en compte les contraintes non fonctionnelles (celles qui n'ont pas trait aux fonctionnalités mais à la façon dont elles sont exécutées).

Les contraintes de serveur

Il y a-t-il déjà un hébergement prévu ? Avez-vous des limites de coût pour l'hébergement ? Assurez-vous, avant de choisir un CMS, que celui-ci pourra tourner sur le serveur que vous devez utiliser, ou qu'il ne générera pas un surcoût rédhibitoire.

Les besoins de puissance et/ou de rapidité

Avez-vous beaucoup de données à traiter ? Existe-t-il des contraintes de temps réel (chat, jeux...) ? À qualité égale, les CMS sont généralement moins performants qu'un site web fait sans CMS. Si la performance est au cœur de votre projet, il faut être prudent quant à l'usage d'un CMS, quel qu'il soit.

Les délais de livraison

Combien de temps avez-vous pour terminer le site web ? Des délais courts favorisent souvent l'utilisation d'outils de haut niveau. Un CMS est souvent une arme efficace pour tenir un planning serré.

Existant et compatibilité

Existe-t-il déjà du contenu pour le site ? Si oui, sous quel format ? Devez-vous être interconnecté avec un autre logiciel (web ou non) ? Si oui, quel sera le format des échanges ?

Certains CMS permettent de mettre à jour le contenu du site via des formats d'échange (XML-RPC, JSONP...), d'autres non. Certains proposent d'importer certains types de données facilement, pour d'autres ce sera un véritable calvaire.

Pour vous donner une idée de la liste des fonctionnalités et des contraintes non fonctionnelles à vérifier avant de faire votre choix, vous pouvez consulter le site CMS Matrix.

> Attention, le site n'est malheureusement pas à jour et ne doit pas servir de base de choix. Je vous le conseille pour sa liste exhaustive des points à examiner.

Sélectionnez deux CMS ou plus dans la liste et cliquez sur "Compare" en bas de la page pour avoir la liste des fonctionnalités.

La communauté

Après mûre réflexion, vous avez choisi un CMS qui répond à toutes vos contraintes fonctionnelles et non fonctionnelles. Mais il y a encore un aspect auquel vous devez faire attention : la communauté du CMS est-elle active ?

Problèmes

Au cours de la vie de votre site web, vous risquez en effet d'être confronté à deux principaux soucis.

1. Être "bloqué" dans l'utilisation du CMS : un problème, un bug, une documentation peu claire ou mal traduite... les soucis peuvent être nombreux et parfois vous empêcher de déployer correctement votre site web.
2. Voir le CMS "mourir" : les personnes – ou la société – en charge du développement abandonnent le projet, qui n'évolue plus. Un site web qui ne correspond plus aux standards techniques et ergonomiques, c'est souvent très pénalisant.

Face à ces deux problèmes, il existe une solution unique : la communauté !

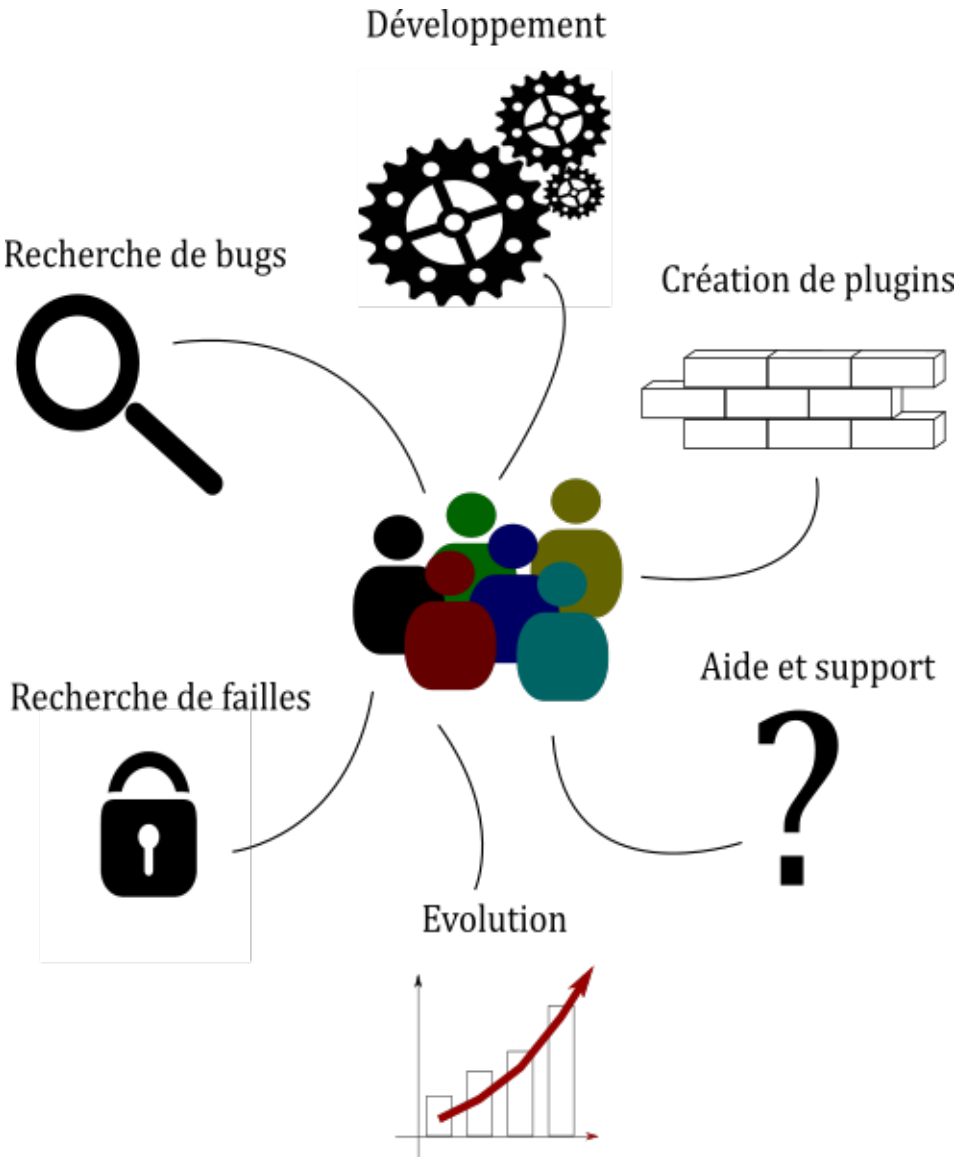
> Comme je le disais précédemment, la communauté est plus facile à trouver autour d'un CMS open source. C'est d'ailleurs là l'un de leurs avantages. :)

Solution : la communauté !

Une communauté large et active est un important critère de qualité pour un CMS.

Elle apporte :

- de la sécurité : en détectant et corrigeant les failles qui peuvent exister ;
- du support : en apportant de l'aide via les forums, des tutoriels, des cours...
- des fonctionnalités : en développant des plugins ;
- une durée de vie : en continuant à faire évoluer l'outil au fil des années.



Rôle d'une communauté dans le cycle de vie d'un CMS

L'importance et la réactivité d'une communauté sont primordiales. Si vous hésitez entre deux CMS apportant des fonctionnalités similaires, choisissez celui qui bénéficie de l'appui de la meilleure communauté !

Vous savez maintenant comment choisir un CMS ! Dans un prochain cours, je vais vous expliquer le fonctionnement de l'un d'entre eux, le CMS WordPress. Si ce CMS est adapté à votre projet, rejoignez-moi dans ce prochain cours. ^^