CMS（Content Management System）

隐藏在内容管理系统(CMS)之后的基本思想是分离内容的管理和设计。页面设计存储在模板里，而内容存储在数据库或独立的文件中。 当一个用户请求页面时，各部分联合生成一个标准的 HTML 页面。

一个内容管理系统通常有如下要素：

文档模板

脚本语言或标记语言

与数据库集成

内容的包含物由内嵌入页面的特殊标记控制。这些标记对于一个内容管理系统通常是唯一的。 这些系统, 或 Java 等。

内容管理系统对站点管理和创造编辑都有好处。这其中最大的好处是能够使用模板和通用的设计元素以确保整个网站的协调。 作者只需在他们的文档中采用少量的模板代码，然后即可把精力集中在设计之上的内容了。要改变网站的外观， 管理员只需修改模板而不是一个个单独的页面。

内容管理系统也简化了网站的内容供给和内容管理的责任委托。很多内容管理系统允许对网站的不同层面人员赋予不同等级的访问权限， 这使得他们不必研究操作系统级的权限设置，只需用浏览器接口即可完成。

其他的特性如：搜索引擎、日历、Web 邮件等也会内置于内容管理系统 CMS 内，或允许以第三方插件的形式集成进来。

如何开发CMS

内容管理系统是一个很泛的概念：从商业门户网站的新闻系统到个人的Weblog都可以称作发布系统。

框架型：本身不包含任何应用实现，只是提供了底层框架，具体应用需要一定的二次开发，比如Cocoon，Vignette；

应用型：本身是一个面向具体类型的应用实现，已经包含了新闻/评论管理，投票，论坛，WIKI等一些子系统。比如：postNuke xoops等；

但无论如何，在发布系统选型之前，首先了解自己的实际需求是最重要的：想根据现成系统将自己的需求硬往上照搬是非常不可取的。访问量，权限控制和各种功能需求。每个模块和功能自己都比较清晰一点以后，再去网上找找类似的实现：你会发现其实每个环节到目前上都有比较成熟的实现了，而且还在不断完善和发展中，如果没有：你的需求太特殊，或者可以尝试分解成更小的系统组合实现。

内容管理系统被分离成以下几个层面：各个层面优先考虑的需求不同

1，后台业务子系统管理（管理优先：内容管理）：新闻录入系统，BBS论坛子系统，全文检索子系统等，针对不同系统的方便管理者的内容录入：所见即所得的编辑管理界面等，清晰的业务逻辑：各种子系统的权限控制机制等；

2，Portal系统（表现优先：模板管理）：大部分最终的输出页面：网站首页，子频道/专题页，新闻详情页一般就是各种后台子系统模块的各种组合，这种发布组合逻辑是非常丰富的，Portal系统就是负责以上这些后台子系统的组合表现管理；

3，前台发布（效率优先：发布管理）：面向最终用户的缓存发布，和搜索引擎spider的URL设计等……

内容管理和表现的分离：很多成套的CMS系统没有把后台各种子系统和Portal分离开设计，以至于在Portal层的模板表现管理和新闻子系统的内容管理逻辑混合在一起，甚至和BBS等子系统的管理都耦合的非常高，整个系统会显得非常庞杂。而且这样的系统各个子系统捆绑的比较死，如果后台的模块很难改变。但是如果把后台各种子系统内容管理逻辑和前台的表现/发布分离后，Portal和后台各个子系统之间只是数据传递的关系：Portal只决定后台各个子系统数据的取舍和表现，而后台的各个子系统也都非常容易插拔。

内容管理和数据分发的分离：需要要Portal系统设计的时候注意可缓存性（Cache Friendly）性设计：CMS后台管理和发布机制，本身不要过多考虑"效率"问题，只要最终页面输出设计的比较Cacheable，效率问题可通过更前端专门的缓存服务器解决。

此外，就是除了面向最终浏览器用户外，还要注意面向搜索引擎友好(Search engine Friendly)的URL设计：通过 URL REWRITE转向或基于PATH\_INFO的参数解析使得动态网页在链接（URI）形式上更像静态的目录结构，方便网站内容被搜索引擎收录