V. How CMS? Design Your Content Management System

The author of this dissertation is a Taiwanese Journalist and consultant in the Media and IT industries. At the same time, he is also an active Researcher at Soochow University, Taipei, Taiwan. Since 2003, sponsored by the German Academic Exchange Service (Deutscher Akademischer Austausch Dienst, DAAD[33]) **(APPENDIX C.)** he works in Department of Information Science, Institute for Media Studies and Communication Science, at Free University Berlin, advised by the Dean of the Institute, Univ- Prof. Dr. Gernot Wersig. (For detailed information, please see the attached CV of the author **(APPENDIX H.)**.)

In this dissertation, we are not trying to pretend Prophet in the CMS field. The judgment criteria and evaluations concerning CMS concepts and products are all based on the literature reviews, market surveys, interviews with IT professionals and the researching conclusions of this dissertation.

However, “Reality is in the eyes of beholders!” Only when one jumps into a real situation, then he/she will have more understandings of the problems domain and find his/her own way to overcome the difficulties!

Therefore, when trying to find out how to design a CMS for the company and, most important of all, how to make it works in the enterprise. We take a very practice oriented approach in this dissertation. That is, we create a REAL CASE to build up a CMS based web-site.

In this REAL CASE, we cooperate with two German CMS developers, INFOPARK AG (http://www.infopark.de) and STRUKTUR AG (http://www.struktur.de), and a Taiwanese business intelligence service vendor, the Creative Network Solution Co., Ltd. (http://www.creative-lab.net)[34] **(for details of the trilateral cooperation, please see the attached photocopies of the contracts in APPENDIX D/E/F.)**, to make an experiment on CMS with the following three goals:

1. Trying to build the first Knowledge Portal of CMS and the related document related technologies (to some degree, also refers to Enterprise Content Management, ECM) (http://www.dcms.org.tw) (http://www.dcms.org.cn)

2. Trying to build a CMS based cooperation platform and information exchange system for this trilateral project. And this international cooperation aims to push the CMS knowledge and technology developments in the Chinese speaking areas and, of course, to open the CMS market there.

3. Obviously, in this three parties’ cooperation project, there will be surely many problems which we might be able to resolve by implementing a CMS based cooperation platform and information exchange system.

By starting the international CMS business cooperation and by building up the mentioned CMS Knowledge Portal in the Chinese speaking areas, just like all other companies, we have to consider the whole project from the business value perspectives and facing exactly same technical and organizational challenges by choosing, evaluating, designing and implementing an “appropriate” CMS system.

[33] The certificate of sponsorship form DAAD could be found in Appendix C of the dissertation.

[34] In Appendix D/E/F of the dissertation, we offer the scanned copies of the business contracts of the above mentioned international cooperation project for CMS in the Chinese speaking areas. For further details and information of the cooperation, please directly contact the author: Mr. Vincent, Chung-Wei Lin at Vincent@creative-lab.net

So, the above mentioned initial of an international cooperation for CMS makes us exactly in the similar and "quasi-enterprise" situations and conditions like all other companies. We start this project since March 2003 and spent more than 2000 man/womanpower/hours to put it into practice. Based on the lessons learned and the experiences gained, mostly from our working diary and developing notes, we try to offer a CMS Implementation Compass to the companies which are interested for such an application. And at the same time, these practices are also good materials for further communication researches in the CMS field.

5.1 Design and Implementing CMS – Business Project perspectives

There is no "King's Way" to select the right CMS package or to organize the whole project well! You should take the same approach as you would in selecting any major server software. One difference, though, may be the breadth of internal users and external visitors who will interact with your CMS. The more you can efficiently involve those disparate players, the greater your chances of success.

In fact, the steps below essentially outline how to implement your entire CMS project. New and packages CMS tools, products and solutions can play an important role in your efforts, but in order to succeed, technology must serve the business master.

5.1.01 Identify Stakeholders

Inventory the range of stakeholders for your CMS project. Then identify "internal champions", people with line responsibilities who can advocate for building your CMS. Internal champions should not necessarily be the most technical representatives, just the most in pain under the present system or the most to gain from the new one. Typically, those users with the most at stake are editors, authors, product/service managers, and other communicators.

But don't forget about your IT group. In our CMS experiment, we found out that the selection teams trying to work around their internal systems resources because they don't like the direction the latter is heading. Hence, circumventing your IT group is a grave mistake. They are going to have to support and enhance whichever package you chose, and none of these products are completely "out-of-the box".

In any enterprise software selection process, if your Marketing, IT, and other teams have strong disagreements, it's always best to surface and address the conflicts as early as possible.

Note also that you may need to involve different IT groups, if you company splits responsibility for application development, network and server administration, and security. They are all important to a content management project.

Likewise, CMS technology choices and system design should never be left solely to technical stakeholders.

One of the other key players is your Chief Financial Officer (CFO) – or at least their stand-in. You must try to involve him or her early and often, and make sure someone from your Finance group remains on the team for the duration. Among other benefits, they can help you build a business case and advocate for a reasonable budget.

5.1.02 Build and Prioritize your Initial Requirements

First of all, use the check list for the content management problem health in your enterprise in Chapter IV (4.1). Take a thorough inventory of content areas and owners (you'll need to do this sooner or later), and base-level corporate infrastructure (human, knowledge, physical) for the project. A complete enterprise content audit may be overkill – focus instead on the major content components going to your key web properties. But look deeply at those components, because what you find may surprise you.

The natural inclination is to develop expansive requirements. This is OK as a starting point. But at each step in the process, you'll want to ask your team, "How important is it really that this aspect be automated?" Remember you are not building the perfect system, just a great one. No single platform presently on the market can solve the entire web content management problems seamlessly. Remember that complete requirements are nice, but prioritized requirements are gold. Knowing which features or attributes are more important than others will make it easier to contrast the vendors, and will also help you keep costs in budgets.

At this point, you may wish sound out either your favored integrator or a handful of vendors who would appear to be a suitable match, especially on broad-brush cost estimates, to help inform the next step. This can be done by issuing a "Request for Information" (RFI). Nonetheless, avoid a vendor dog-and-pony show at this point. It is premature and can waste your time (and theirs).

5.1.03 Develop User Cases or Scenarios

Sometimes called "scenarios," use cases can provide a much richer way of describing your needs and connecting them to business benefits. Use cases can also vastly simplify and speed up the vendor selection process, by giving everyone a target to shoot for. Finally, selection team members can understand vendor offerings much better if they discussed and demonstrated using scenarios that team members face every day.

We suggest you to develop use cases that start with an "as-is" and then describe an ideal "to be." That gives vendors. The more detailed your use cases are, the more you will be able to differentiate solutions. But if time is short, even simple use cases may suffice. Consider this:

AS-IS

Jenkel Marketing e-mails some important text changes for a webpage to the IT department, who, for security reasons, control all access to the web servers. The request goes into a helpdesk queue. Two days later, Müller-prothman Engineer makes the requested changes in a text editor, and e-mails the HTML file for Jenkel to review. Jenkel e-mails back that everything is fine. The next day, Müller-prothmann FTPs the modified HTML file to a Unix staging web server.

That night, a sync process synchronizes the staging and production web servers. Jenkel checks each day to see when the new material is in production; when the modified page shows up as live on the Germany site, Jenkel e-mails Lin, Manager in Taiwan. Lin manages the company's Chinese web presence in Taiwan. Jenkel includes a copy of her original message to Müller-prothmann, to indicate what has been changed. Lin, who runs Windows XP plus Office 2003, checks among the 130 Chinese sites to see which ones contain this content (only 42 of them do) and therefore need to be updated. He looks up the relevant translators from her Outlook contacts directory and forwards Jenkel's e-mail (which is now several levels deep) to them.

Upon receiving that message, Schuck-Wesig, the translator for the English site, saves the relevant HTML page to her local hard drive, opens it up in Dreamweaver, and then checks against the e-mail she received from LIN, and makes the relevant modifications to the English version of the page. She then e-mails the HTML file back to Lin, who forwards it to Müller-prothmann Engineer. One day later, Müller-prothmann FTPs the HTML file to the staging docroot of the Englsih site. That night, a sync process pushes the new content into production. Schuck-Wesig checks the live site daily and informs Lin when new changes have gone live.

TO-BE

Jenkel Marketing logs into the CMS from using her Mac and changes text on an existing static HTML page, modifying some text and adding several new paragraphs. She has sufficient rights to push the change into production and does so. The page goes live on the USA site. Since this page is localized on other sites, this triggers a translation workflow.

Lin Manager receives notification via e-mail that this particular page has been revised on the English page. The CMS tells him that this text has been localized on several sites within his Chinese region.

Within the CMS, Lin directs the update task to several different translators, representing those sites. Schuck-Wesig, the translator for the English site, receives an email notification and logs into the CMS, where she is only allowed to read and modify pages in tasks assigned to her. She is given a visual cue as to what has changed in the latest German and Chinese version of the page and modifies the relevant English page.

Upon submit, the task returns to Lin, who must approve all final translated pages. He can preview all modified pages first. When Lin approves a page, it goes into a queue for an automated nightly promotion to the live server. On a daily basis within the CMS, Lin checks the completion status of the various tasks he has doled out to translators[35].

What you have done here is signaled to the designing team of your CMS that:

- Your IT team cares very much about security, but has become a bottleneck; significant time savings could be achieved.
- There is a dearth of communication in your system; there are significant risks for the wrong material to be published.
- Pages have relationships across sites; work could run more smoothly if those relationships were systematically maintained.
- Your staff uses Macs and Windows machines, albeit in different places.
- You have a spectrum of very specific rights you wish to assign, and different content deployment routines based on those rights and/or content types.

You could have skipped the "as-is" piece, though bidders will find it very useful. More importantly, you've given the designing team of your CMS something very concrete to propose against and demonstrate when they meet you.

[35] Similar situation-simulating approaches and techniques are deeply discussed in Creech ML: Author-oriented link management, COMPUTER NETWORKS AND ISDN SYSTEMS 1996, Vol 28, Iss 7-11, pp 1015-1025, ELSEVIER SCIENCE BV

5.1.04 Solidify your Business Case

Before you move on, make sure you can still connect a CMS to your broader corporate objectives. Having even a simple case statement that does this will facilitate all the choices you make going forward.

Is this a good time to set a project budget? Two schools of thought predominate here[36]. One school recommends waiting to see what potential opportunities lay ahead and setting a budget later.

However, it may be more practical to set a budget now to discipline the rest of the requirements gathering and vendor-winnowing, while remaining flexible depending on what arises down the road. In either event, don't forget about customization and integration resources. Account for all of the services you may need, and only set aside 25-35% of your budget for software.

5.1.05 Design the Outlines of the System

There are several tasks you can accomplish here to scope out the likely size of the project.

Hopefully, the use-case building exercise has compelled you to model your existing workflows and review what aspects can be better automated. Now take a stab at building taxonomies and vocabularies. Diagram how you anticipate different systems will work together. Create the information architecture for your new site, along with wire framed page component diagrams.

Develop a creative brief as necessary, and perhaps user-interface prototypes.

Many well-intentioned project teams skip or downplay this step. You should not. A deep understanding of your own content is an essential precondition to managing it more effectively.

You may need some outside help here. The key in selecting external consultants is their solution neutrality.

An integrator may well define the solution in a way that makes it most susceptible to resolution through their favored technology, and may receive a bounty from the CMS software vendor for recommending their product.

5.1.06 Review Technology Alternatives

Use your requirements and the outlines of a new system to develop a work plan, which will simplify matters for you and vendor alike. The work plan, together with your use-cases and the deliverables described above, can serve as the basis for a perfect ending decision making.

Surely, from the mid-market and up most (but not all) CMS vendors will come in to present their products firsthand, which may serve as a learning experience and could help sharpen your requirements and shorten your list. But chances are that if you have designed your system carefully and digested this report, you might not need to conduct a wide cattle call. Have them demo your use cases rather then their canned routines. A browser with a good Internet connection is much more useful here than PowerPoint slides.

[36] Miller R: Content management - Case studies, ECONTENT 2003, Vol 26, Iss 5, pp 23-26, ONLINE INC

Be sure to ask what optional modules the CMS vendor would not include in your solution and is not including in its price. Unfortunately, "sandbagging" remains rampant: to fully realize a solution after you are underway, the CMS vendor informs you that you must purchase optional modules. This potential for extra cost is especially important to consider for "deployment" modules and application server or portal connectors.

5.1.07 Perform Due Diligence

You probably already know the importance of performing due diligence on server software, but in any case, here's a brief primer. For starters, you'll want to talk to companies like yours who have implemented the package.

CMS Vendors will total the number of licenses they have issued in the past year, but try to find out how many actually implemented and then kept the product. If you are suspect about a particular feature or module, ask to see it in production on a live server from among the vendor's clients. This is especially critical when your implementation depends on integrating two or more products – the fact that the vendors are "partners" means little. How many sites currently use those two products in tandem now? Fair or not, it's the early adopters who provide integration lessons for everyone else.

If you are going to procure professional services from the CMS vendor or an integrator (and you probably will), be sure to meet the team who would work with you. Every company has experienced CMS hands. Many of them are extraordinarily knowledgeable, but unless you are the SUPERBUYER (or even if you are), they are likely to be preoccupied with the next sales call as soon as your contract is signed.

So, meet the actual project managers, architects, and developers who will be implementing your system or training you how to run it. If you can't work successfully with them, what makes you think you can work with their product? Note that this is especially important with CMS vendors who are enjoying substantial success and momentum in the marketplace, and are therefore growing (and hiring) a lot. That's right: financial health and the quality of human resources at hand can be competing dimensions.

Review any vendor's financial performance, but with grain of salt. By our product selection processes for the CMS Knowledge Portal in the Chinese speaking areas, we have seen big, successful companies jettison CMS offerings, and niche vendors survive and thrive. Note that: most of the major CMS software players have just gone through the stress of flat sales and are only now beginning to climb out of the tech recession. Nevertheless, there remain key metrics to evaluate, like available cash (and burn-rates for cash-flow negative companies) and revenue momentum. Of the two, cash on hand is the best predictor of whether a company will survive. Use your experience – and instincts – here. For better or worse, positive "buzz" around a particular vendor may be a better predictor of long term viability than the quality of the technology.

5.1.08 Download Trial Packages When Possible

You can sometimes download trial versions of CMS packages from vendor web-sites. Do this for your shortlist (but skip the CD-delivered demos). Note that this trend is new, and many of the trial versions are "light" implementations that can be buggy and not always well-supported. But they will give you a sense for the product. Of course, with an open-source package, you can always download, install, and play with the complete production version. Use trial versions as a basis for asking questions, rather than as your final decision point.

Remember you may need more than one package as part of a complete solution. Your requirements may push you to consider XML-handling tools, Search engines, a separate delivery platform, automated classifications products, and perhaps other software, in conjunction with a CMS package.

5.1.09 Evaluate Security Implications

By now, your security staff should have a good understanding of how the different products will fit into your network environment and overall protection schemes.

However, the CMS package itself should be scrutinized. Be sure your vendor can answer these six key security questions1:

- Who runs security tests on the CMS? "No one" is a bad answer.
- Are there regular security notifications? When breaches are found, you'll want to be the first o know.
- Which third party products are included within the CMS? Search engines and other applications have their own security considerations.
- Are there security specific guidelines for the CMS? If not, they probably haven't look deeply enough.
- Can the supplier provide staff with recognized security qualifications?
- Have you established adequate internal controls? Remember that most security breaches are internal.

5.1.10 Consider Requiring a Proof-of-Concept

If you are about to drop huge investment on CMS licenses alone (with integration on top), it's reasonable to ask your final list of CMS vendor finalists to come in to your company, install their packages on your hardware, and develop some sample templates and workflows that make sense to your team.

You should define this exercise in advance, using your content, rather than using canned vendor demo sites. Note that: depending on whether the proofs can be done concurrently, a proof of- concept step could add a month or more to your selection cycle. Nevertheless, this kind of "show-me" evidence can be a powerful predictor of which offering will work best for you.

Be aware, however, that you risk ruling out a vendor who elects not to compete in this way. For an enterprise installation, this is an unreasonable position for a vendor to take, but in the mid-market, the cost of participating in a proof-of-concept could far outweigh any margin the vendor would like to achieve.

One way to avoid this is to offer to compensate the vendor an agreed-upon sum for their time (but not their software) if they invest in a proof-of-concept and not emerge the winner.

5.1.11 Choose a Platform

With solid requirements and a firm budget the choice should be fairly clear by now. Negotiate your best terms. If you go the proof-of-concept route, you probably want to negotiate those terms before the final tests, when you will still have maximum leverage, but hold off on completing the contract, as issues may arrive during that phase that you might wish to address in the final agreement.

No software is perfect. Conduct a group de-brief after the competition phase and make a list what don't like about the package you are going to select. This helps you set internal expectations and account for necessary work-around as you finalize the implementation plan. It can also give you more to go back to request of the vendor. Contributing resources towards resolving some of shortcomings your team finds could be part of the final deal.

5.1.12 Prototype as Early as Possible

If you are purchasing a large system, consider confirming that you made the right choice before getting the full spate of licenses you will ultimately require. This means quickly implementing even a partial solution for part of your online platform – perhaps a single section of your site, or within a single department. That way, you – and your vendor(s) – can learn critical lessons while the stakes remain reasonably low. It also allows you to show project momentum. And if it all blows up, hopefully you have hedged some of your investment.

That sounds awful, but what is worse than admitting a mistake and switching to a different package? Not admitting your mistake and being stuck justifying a system that doesn't you're your basic business needs. On several occasions by interviewing CIO of big companies in Taiwan, we have heard: "Yes, this software stinks, but we paid $X hundred thousand for it, so we have to use it…"

5.1.13 Recognize When You Need Help and Seek Outside Assistance

Through the whole process above, you may wish you had gotten help. Perhaps the project has stalled or you have not been able to reconcile differences among the team. The right consultant can add value from the earliest stages of the project.

A knowledgeable and experienced outsider can help by:

- identifying key internal stakeholders,
- facilitating among disparate factions,
- applying proven requirements templates,
- helping to clarify the confusing array of technical and design choices,
- assisting in developing the RFP and evaluating vendors and
- generally pushing the process along (this doesn't have to be a long process).

The right company can also serve as a resource (or outsource) for implementation by a team invested in the overall plan. It is also the trends of IT services industry. As always, picking the right consultant will present a key challenge. You'll be able to choose from among independent contractors to the Big 5, and everything in-between, including public web agencies, regional integrators, and niche design and information architecture firms.

However, Note that most professional services firms have pre-existing relationships with specific CMS vendors. This is not necessarily a bad thing – for example, you want an integrator to have pre-existing experience with any particular package they implement for you – but be sure to surface and understand potential conflicts before getting started.

In any case, be sure the specialists working on your system have worked with the relevant CMS software package(s) at least 3 times before, and the current version at least once. Deep knowledge of complex software is an acquired talent. The difference between an experienced and novice developer can make or break your project schedule and budget.

5.2 Design and Implementing CMS – The "Collaborative Function Model"

As what have been mentioned in Chapter III, by literature review, we found Nakano's theory **(Nakano 2002)** for designing a collaborative web site to support our ideas[37]. In Nakano's prototype, he describes two concepts for applying web content management: "Collaboration operations" and "work cycle development".

According to Nakano, "***Collaboration operations***" include the following five operations:

(1) **Submit** — when you move web assets from a work area to a staging area;

(2) **Compare** — identifies whether the assets that were submitted were new, modified or deleted assets to the staging area;

(3) **Update** — copies new, modified or deleted assets from staging area to the given work area;

(4) **Merge** — resolves conflicts between the work area and staging area and

(5) **Publish** — is a snapshot of the staging area.

And "***Work cycle development***" is important to people who are ready to start a web project. They must follow these basic rules:

- **Update** — get the most recent assets from the staging area;
- **Edit** — make changes to the assets,
- **Test** — make sure that the edits work properly;
- **OK** — if the edits work, then submit to staging area (then goes back again to the beginning of the Collaboration Operations).

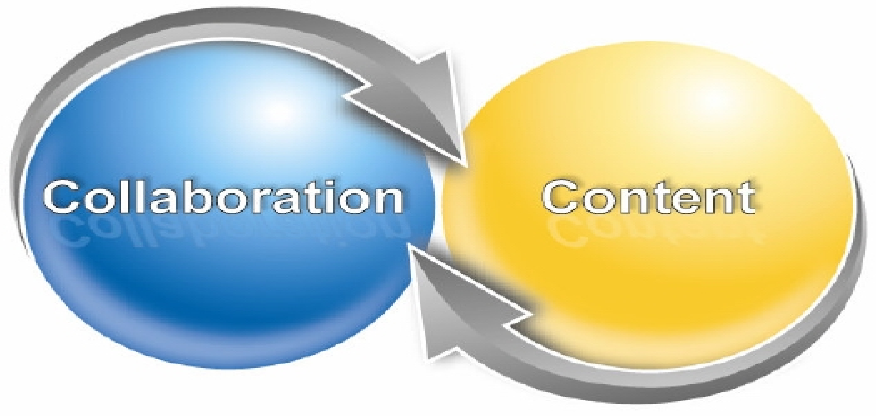
Nakano's prototype is very constructive, theoretically. It provides a fundamental, well-organized, and easy to follow observing and designing perspective for CMS. When working with many web developers, it is very important to follow these two concepts. If they are not followed, the web content can become unusable and starting over may be unavoidable.

However, Nakano's prototype provides only a linear and process oriented perspective. He mentioned only the five important ideas (**SUBMIT, COMPARE, UPDAE, MERGE, PUBLISH**) for CMS designing and the basic work cycle procedure (**UPDATE, EDIT, TEST, OK**). It's, in the essence, a simple CMS prototype, and there's no further ideas and implications from other

[37] Nakano, R: Web content management: a collaborative approach , ADDISON WESLEY PROFESSIONAL 2002, BOSTON

perspectives. And most important of all, in this prototype, "content" is the "result"/"product" from a linear production line! Hence, by really engaged to design, set up and launch the purposed web site, most important of all, a collaborative platform (web-site) for our international cooperation project we need a more function oriented methodology to help us.

For an international cooperation project like what we are attempt to establish, the CMS, which serves as the collaboration platform, should provides more functional mechanisms to improve the interactions among workers, speaking three different languages and locating themselves in t countries and 5 cities, to be easier and straight forward to create value-full contents together. That is, from a communication perspective, we believe the relationship between "content" and "collaboration" (now, it means the mechanism which helps people to manage content) should be more dynamic and interactive, just like the chart that we illustrate below **(same as CHART 3.04).**



Dimension	Content Production	Content Delivery

(Chart 5.01) The Conceptualization of the relationship between "content" and"collaboration" with dimension division "content production" and "content delivery"

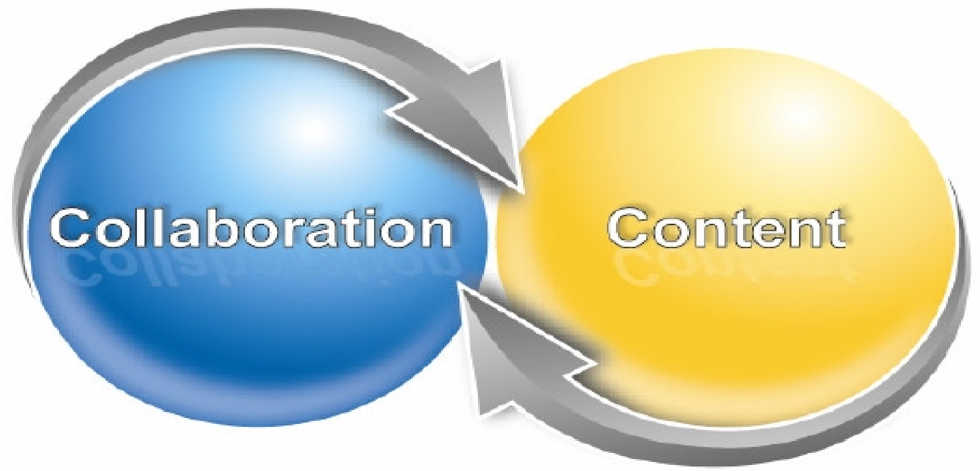
Following this concept and based a Nakano theory for designing a collaborative web site, we established a function oriented model on our won as the critical consideration and judgment criteria to design and implement a CMS based web site for our international cooperation web site. We call it "**Collaborative Function Model**" of CMS, 'cause it stresses the importance of understanding the internal/external communication needs and processes in any CMS Designing Project.

In this model, for international business collaboration/ communication, a (perfect) CMS should be firstly divided in two dimensions:

1. DIMENSION I. **Production**, **where content goes "from thought to click".**

2. DIMENSION II. **Delivery**, **where content actually gets "consumed by end-users"**.

Nakano hasn't mentioned the dimension II "Delivery" or he did by take this part simply as "publish". However, as the boundaries between information and communication, or let's say, among the industries which are illustrated in the two charts in 3.1 **(CHART 3.01/3.02)**, vanish and melt gradually, it is necessary to expand discussions in this dimension. So, we conclude the "Collaborative Function Model" in the following chart below **(CHART 5.02)**. Both "content production" and "content delivery" dimensions contain its specific attributes that must be carefully considered in any CMS project:



Dimension	Content Production		Content Delivery	
ATTRIBUTES The sub-attributes are the function oriented mechanisms that we think most essential for improving the interactive effectiveness and the business value achievements of a CMS platform	Role Management	**(5.2.01)**	Page Generation	**(5.2.10)**
	User Interfaces	**(5.2.02)**	Searching	**(5.2.11)**
	Author System	**(5.2.03)**	Personalizing	**(5.2.12)**
	Integration	**(5.2.04)**	Privileging	**(5.2.13)**
	Metadata	**(5.2.05)**	Caching	**(5.2.14)**
	Workflow	**(5.2.06)**	Syndication	**(5.2.15)**
	Templating	**(5.2.07)**	Cross Media Publishing	**(5.2.16)**
	Versioning	**(5.2.08)**		
	Globalizing	**(5.2.09)**		

(Chart 5.02) The Structure and Attributes of the "Collaborative Function Model"

In this "Collaborative Function Model", the interactive and dynamic relationships between "content" and "collaboration" is on the bottom of the prototype to remind all the designers the symbiosis of these two elements. That is, CMS serves a "place" of series of communications and interactions.

The in table listed 9 attributes in the dimension of "content production" and 7 in the dimension of "content delivery" could be regarded as the expansion of Nakano's simple CMS prototype. Based on our lessons learned and the experiences by designing and implementing the CMS Knowledge Portal in the Chinese-speaking areas, we found the attributes, mostly the function-oriented mechanisms, essential for improving the interactive effectiveness and the business value achievements of a CMS platform. That is also the reason why we name our prototype the "Collaborative Function Model", because the attributes are functions needed to enhance the "collaboration", in other words, to improve the communications in CMS.

In the following sections of chapter V, we will have further discussion on "**Collaborative Function Model**". Based on our lessons learned and the experiences by designing and implementing the CMS Knowledge Portal in the Chinese-speaking areas, each attribute in the table above will be discussed in details, both technically and practically. These discussions will help you to get clear on the latest technological developments and the factors that you must think about if your web site has the needs in the attribute related criterion.

Besides, we also make an "**Author's Checklist**" at the end of each discussion. This "Author's Checklist" is both a review to the discussion and additional remarks to the related discussion. It is a questionnaire aimed to help you focus on your own analysis for planning the appropriate CMS platform for your unique needs.

In short, "**Collaborative Function Model**" concludes the central concerns, considerations and the decision making processes, when we attempt to select our foreign CMS solution partners, to evaluate the CMS packages, products and solutions (seem to be anonym?) in the market, to propose our business strategies, and finally to design and to implement the unique and the first CMS Knowledge Portal in the Chinese speaking areas.

Surely, "**Collaborative Function Model**" is also very useful and helpful for you to design and implement your own CMS project or to select the proper CMS package or solutions from the vendors in the booming CMS market.

==

The Dimension of Content Production

In the "Collaborative Function Model", we define "Content Production" as everything that happens to content before the end-user sees it. Although first in chronology, this dimension is typically last in line to be automated by web teams. Indeed, by selecting a CMS package or by designing a CMS project, most companies tend to focus first on delivery systems – be it through portals, application servers, or other forms of dynamic publishing.

However, from our experience in implementing the CMS based web site, we find out that: the Production dimension is actually where many substantial cost, control, and efficiency gains can be realized. It is, from both functional and organizational, the core elements of CMS. And according to the results of a research with similar approach[38], if the CMS designer could put more attention on the production dimension, it is estimated that the efficiency of the whole software system may have a 35% increase. And, we would also like to point out that, only when the project leader of the CMS has carefully consider this dimension, can the effectiveness and efficiency of the CMS which will be implemented and launched in your own company reach your expectation.

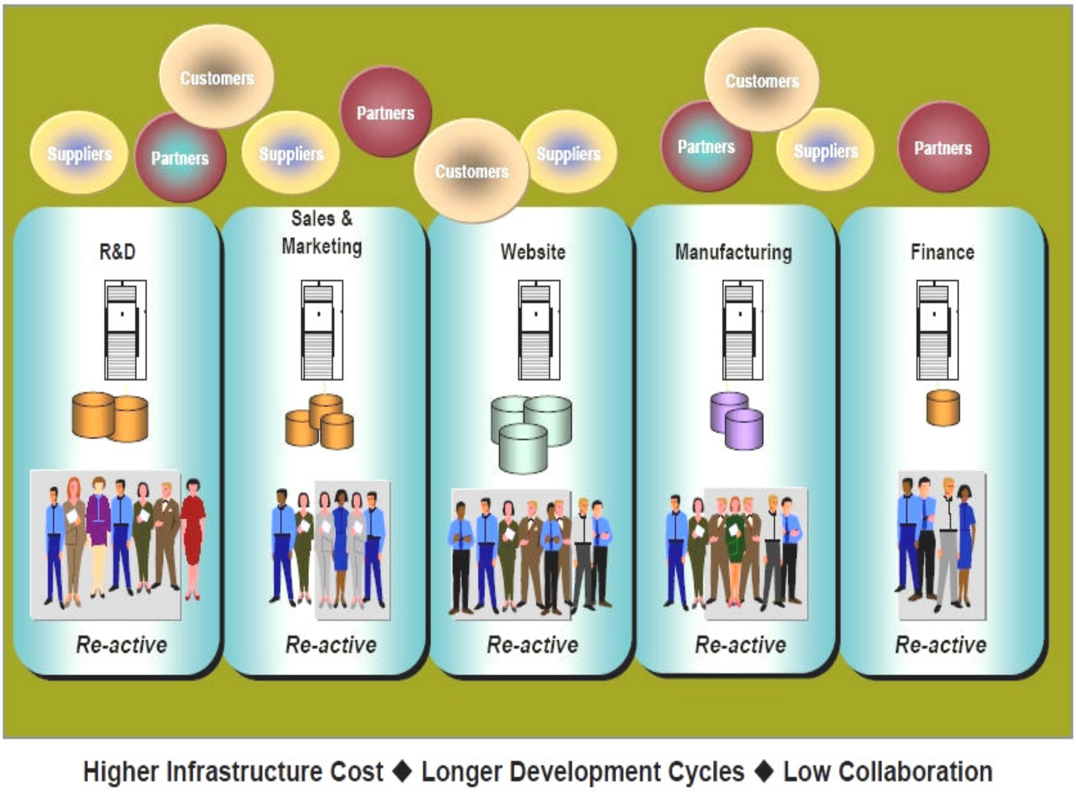
Let's start our discussion from the groundwork for all CMSs today: the Role Management:

5.2.01 Role Management: The Groundwork of Your CMS

With the continuous developments of information and communication technologies, modern enterprises start, more or less, sooner or later, their processes of informationalization and digitalization. However, the adoption of new technologies in the early stage doesn't really guarantee the competitive advantages. As you can see in the following chart **(CHART 5.03).**, for many companies, which have implemented many different internal/external IT projects, they are now suffering greatly from the problems of too many Information systems get involved and, most important of all, the chaos caused by too many related users get involved[39]. Trying to find out a way to integrate these systems for the synergy is then the biggest concern of all the IT managers in the world.

[38] Goodwin S; Vidgen R: Content, content, everywhere... ... time to stop and think? The process of web content management, COMPUTING & CONTROL ENGINEERING JOURNAL 2002, Vol 13, Iss 2, pp 66-70, IEE-INST ELEC ENG

[39] Kerer C; Kirda E; Kurmanowytsch R: A generic content-management tool for Web databases, IEEE INTERNET COMPUTING 2002, Vol 6, Iss 4, pp 38-42, IEEE COMPUTER SOC

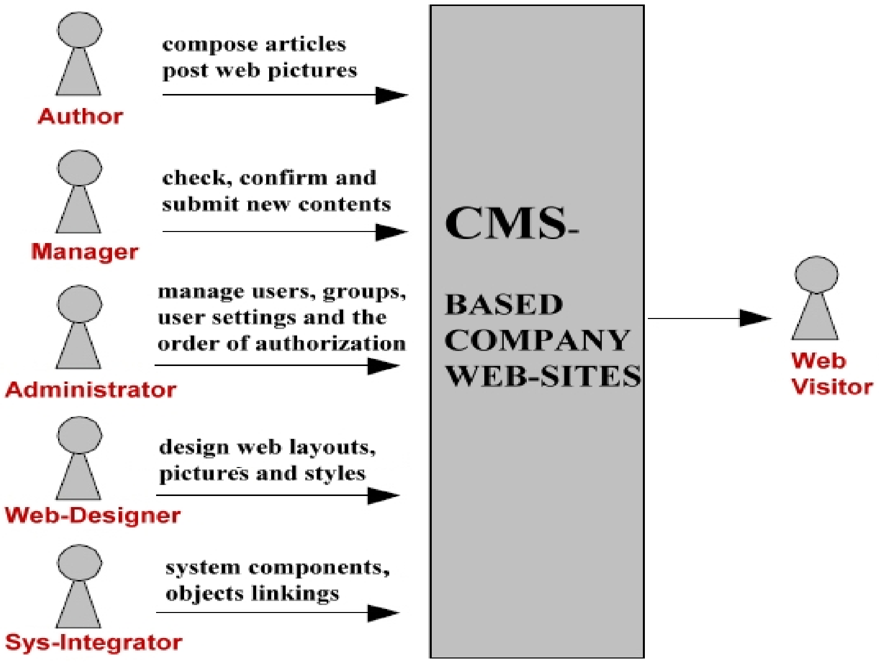


(Chart 5.03) The isolated and chaotic information system deployment in modern enterprise, BP of CMS vendor "Stellent Inc." The chart could be found in http:///www.stellent.com under Online Library

The above mentioned situations and similar problems could also be found by the web content management today. Many people can be involved in the production of even a departmental web-site. And since one of the key advantages of CMS is to distribute content maintenance capabilities directly to content owners, implementing a CMS leads to even more people becoming involved in this process.

In that case, if the CMS is the technology to rescue, that is, it must be a system that can enable you to manage internal access and permissions (access and privileges for users outside the corporation are addressed later) that is much more robust than that required to support only one or two webmasters updating an entire site. Users can be assigned privileges based on the role they play (the types of things they can do), or group to which they belong, which circumscribes their authority – and typically the scope of the content areas they can edit. It is possible (and indeed quite common) for a user to make up a group of one.

How will CMS make it? Mostly, CMS uses the author system concept (we'll have further discussion on this in 5.2.03 of this dissertation) as its kernel and takes the role management as its running principals and schema. Just like the chart below **(CHART 5.04)**., it is clear that the users are categorized into different groups and then differently authorized for their tasks and privileges. By doing so, CMS solves the chaotic information collaboration problems in modern enterprises, a least, to some degrees.



(Chart 5.04) The Role Management Structure in CMS ICOYA. This chart are "exclusive authorized" by the Struktur AG to privileged academic uses. Pleas see the appendix for more Info.

Today, almost all CMS packages come out of the box with generic roles already configured for your use. Except on the very low end, these products then enable you to modify those roles as necessary. However, not all CMS packages allow you to create completely new roles, and among those that do offer this capability, they may not be able to circumscribe functions in exactly the way you would like.

For example, you may want your Interns to add and modify metadata, but have no other privileges, or for Managers to initiate workflow tasks, but not be able to author content. Hence, it is important to ask CMS vendors to show you just how o make the roles and groups you think you need. Most CMS packages will tie into existing corporate directory systems (such as LDAP servers), but note that the way they do so will vary markedly among competing products. For example, some products will access an LDAP repository in real-time, whereas others require that the LDAP server sync up with the product's own access control lists on some sort of regular schedule. In the former case, you need to make sure that the network between your CMS and your directory server is completely reliable. In the latter case, there can be periods where a user whose rights have been expunged in the corporate repository may still have access to CMS privileges, or conversely, has been added to the corporate directory but is not yet visible to the CMS until the next synchronization.

Author's Checklist for Role Management

1. What level of granularity do you need now, and anticipate needing in the future?

Some CMS packages proscribe certain roles or limit the number of groups you can define. You can theoretically extend these to create new roles, but the level of effort and potential for problems at upgrade time might make it more worthwhile to simply select a package that could support the umber of different roles you need from the outset.

2. Does your enterprise use a particular authentication mechanism?

Internal users can be authenticated by the operating system, the web server software, through internal application logic, or even by a content database. Each has its advantages and disadvantages, but nearly universal experience reveals that you will trade off potency of security with ease of use and maintenance. Don’t underestimate maintenance overhead here.

3. Do you require or prefer a certain operating system?

Obviously, this is a global consideration, so let’s address it right up front. If the CMS uses Microsoft Active Directory permissions to manage roles and authentication, then you are going to have to invest in some potentially tricky integration to be able to run it on the new Sun Solaris server you just bought. Operating system choices present the first of many chicken-or-egg predicaments in selecting a CMS package. In an ideal world, you would be free to select the most suitable product regardless of any platform limitations; in the real world however, the IT team that will have to support the application going forward may be much better versed in one operating system over another.

4. Do you need to integrate with other identity management systems?

If your users can access the CMS through an existing corporate log-on they will be more inclined to participate actively. If you use LDAP or Microsoft’s Active Directory for corporate directory services, look for a CMS package that supports those protocols. Then ask the vendor how they support the protocol. Or perhaps you are using your corporate Portal to support single sign-on across the enterprise; clearly you’d want your CMS package to tap into those permissions.

(Chart 5.05) Author’s Checklist for “Collaborative Function Model’s” System Attribute: “Role Management”

5.2.02 User Interfaces: The Usability of Your CMS

Debates about the usability of content management systems have gone on for as long as there have been CMS packages. Many early CMS products – some of which represented quick migrations from client-server systems – were notable in their difficulty to use. Over time, some common norms have emerged and are beginning to propagate throughout the industry, and we’ll highlight some of them below. However, the most important thing to take away from this section is that the most usable interface is that which your content managers find most useful. What is logical and simple to one person or company can be non-intuitive for another.

Some UI trends are emerging and have spread rapidly across the CMS vendor landscape. Nearly all vendors now provide multiple ways of entering content into the system (e.g. Word, Dreamweaver, browser forms of various types, etc.). After popping new windows became fashionable in the late 1990s, most vendors now keep users to one, perhaps 2 screens to complete a simple task, using sophisticated DHTML controls and or frames.

Many usability issues – such as authoring and workflow interfaces are addressed separately in those sections below. Some other, more global interface topics are addressed here, including:

a.) User Interface Customization
b.) Work Queues
c.) Help Subsystem
d.) Management Reporting

And finally, to be clear, in this section we are talking about user interfaces for content contributors, editors, and managers, not content consumers. For your site visitors, you control the usability through your templates and navigational structures; although some CMS vendors will circumscribe your flexibility here more than others.

a.) User Interface Customization

If the intuitiveness of a user interface remains at least somewhat in the eye of the beholder, then vendors would need to be extraordinarily prescient (or lucky) to know what would work well for your team. Clearly, the UI assessments that your content managers make of competing CMS vendors is an important consideration. But in the long run, perhaps equally important is your ability to modify the user interfaces to your particular needs.

CMS Vendors vary substantially here in whether and how they expose user-screens to your developers. In some cases, user screens are constructed from XML config files or templates, which are useful for large sites where you may need variants of the same interface but want to maintain some hierarchical integrity among them, but could be limiting due to a lack of a scripting interface.

In other cases, CMS vendors allow you to create and modify system screens using the same templating approach and methods that you use to create output templates for content – often some sort of scripting language. In general, we tend to prefer this approach because it makes maximum use of the system's existing capabilities (e.g. versioning) and leverages developer skills (i.e. learning one templating paradigm, not two).

And still in some other cases, vendors allow you to modify input screens via a forms interface; this is the easiest to use for businesspeople to make simple changes, but likely to be inflexible in the long run and doesn't always allow for simple UI enhancements, like form field validation. Finally, some vendors – especially at the lower end of the marketplace – do not allow you to modify user interfaces at all.

Note that the return of thick clients (e.g. Morello from Mediasurface, or various custom interfaces for Mac OS X from other vendors[40]) presents a new dimension to this problem. Many of these interfaces are far richer than their browser-based predecessors, but nearly all of them are modifiable only through a very basic configuration interface, if that.

Before you set off to modify any interfaces, however, get in writing from the CMS vendor some sort of certification about what changes you can make (and where) that won't expose you to having your work overwritten in any new version of the product. Vendors are increasingly sensitive to this problem and will often put templates in a separate, untouchable directory for just this reason. But not all of them work that way.

b.) Work Queues

Work Queues is also called a "task list" or "user in-box". In the work queue process, some works are pending some action from that user. Busy users who interact with the system frequently will appreciate many of the following capabilities:

- view and sort tasks according to priority and other variables;
- view ad-hoc tasks as well as content-approval workflows;

[40] Bartenstein O: Software components for Internet based self-service consulting systems, WEB KNOWLEDGE MANAGEMENT AND DECISION SUPPORTS 2003, Vol 2543, pp 23-35, SPRINGER-VERLAG BERLIN

- see a generous description of the task, including a meaningful title and where that task stands in its overall workflow;
- who originated and/or submitted the task
- the ability to take some action on the task without opening up the content item and/or task detail description.

With the above described features of work queues, a content manager can filter on “overdue tasks” and still could have also looked at “future,” “past” and other combinations of tasks due.

Separately, a good CMS should also show which content objects have been “checked-out” by that user, so that they have a quick idea about their own work in progress, as well as know what content items they may be keeping “off-limits” to other users.

c.) Help Subsystem

Help screens are not useful for everyone. Ideally, your content managers would be so well versed in the interfaces they employ – carefully customized just for their needs – that they would never have to seek any help.

Yet, in practice, users ask for a lot of help, especially in super-distributed management models characterized by an abundance of casual contributors who may use the CMS infrequently. In those cases, “help” often ends up becoming an IT function, in the form of a person on the end of a phone line – even though part of the purpose of implementing a CMS in the first place was to disintermediate IT staff in the publishing process.

A good help subsystem can provide support on a self-service basis. Ideally your CMS would have instructions next to each field in a form, or contextual help on each screen in the system. In practice, most CMS packages that offer help content do not offer context-specific help, reducing its usefulness.

So, who’s job should it be to customize help screens? According to our experiences, we would suggest: it is the job of those who provide end-user training to supplement that work with contextual help materials. Keep that information relevant and updated is everybody’s job. In more forward-looking CMS packages, help blurbs for form fields can be edited on the fly by authorized users.

d.) Management Reporting

It is a measure of the maturity of the CMS marketplace that buyers are starting to look at the software not just for editorial automation, but for real *management* as well. Management begs metrics, and metrics require reports. Ironically, many content management systems don’t provide such reports, and some of those don’t even create logs upon which such reports can be built.

Look back over your business objectives and figure out how your CMS could help you measure how you’re doing. You should decide which ones are most important for your business, but a small sampling of reports that could help your managers manage better might include:

- speed of workflow clearance;
- typical bottleneck points;

- duration of content within particular stages;
- who is logging into the system, how often, and who does not use it at all;
- what volumes and types of content have been classified according to specific nodes in your taxonomy;
- do you have any orphaned content;
- do you have live content of a particular lifespan that has gone unreviewed for accuracy and freshness;
- and so on.

Many CMS vendors will tell you that since they provide all the audit data, all you have to do is turn a report-writer (like Chrystal Reports) against it. That's a cop-out. You'll want reports right within the system itself, so that you can click on content items and tasks to act upon the data you find. You might also want to be able to generate ad-hoc reports within the system. Note that many of these reports can be created by canned search-engine queries in the system.

Author's Checklist for User Interface

1. **What does "easy-to-use" and "intuitive" mean to your content contributors?**

   If you can't answer that, don't request it in your RFP or tender. Be prepared for blank stares from users when you ask, however; most people aren't used to being queried for details here and are likely to respond, "I'll know it when I see it." That's normal. But it then behooves you to organize hands-on tests of the different systems you consider.

2. **How many users will there be in the system, and how diverse are the roles?**

   The sheer number of users – along with the diversity of roles – is going to place greater emphasis on your ability to customize the user interfaces, including any help subsystem. If you have more than 100 users, make sure there is context-specific help and some sort of robust scripting interface into all he system UI screens. If you have more than 1000 users, make sure that the help subsystem is editable, and that any internal templates can undergo the same careful "build" and deployment process that new web-site content areas undergo – since with that many users, you will do well to roll out "editions" of the CMS. The more roles you have in the system, like the greater demand you'll see or custom interfaces; some of these can be addressed in the rights-management subsystem, which ill circumscribe what different roles can see and do, but that is not the same thing as building a sable interface for, say, your corporate librarian.

3. **How much workflow do you anticipate?**

   If you anticipate only a simple author-approve process for most of your contents, and you expect he overall throughput to remain light, then very simple work queues will suffice. But if you have multi-step workflows, with many actors, and/or high-volumes and potentially narrow choke-points, then you will rue the day that you foisted a plain-vanilla task in-box on your users, as they will likely respond by processing tasks offline, likely using some of the rich cueing and messaging in Outlook and Exchange. Power editors in particular want highly customized and customizable task lists.

4. **What reports will your managers (technical, business, and editorial) require on a regular basis?**

List the reports you will want in any RFP and ask the vendors to show you them in their system. Again, start with your business objectives and go from there. If the goal was to reduce your content time-to-market, then figure out what metrics you'll need to show that. In addition to business managers, be sure to consult with editorial and technical managers – they have reports they'll want to run as well.

(Chart 5.06) Author's Checklist for "Collaborative Function Model's" System Attribute: "User Interface"

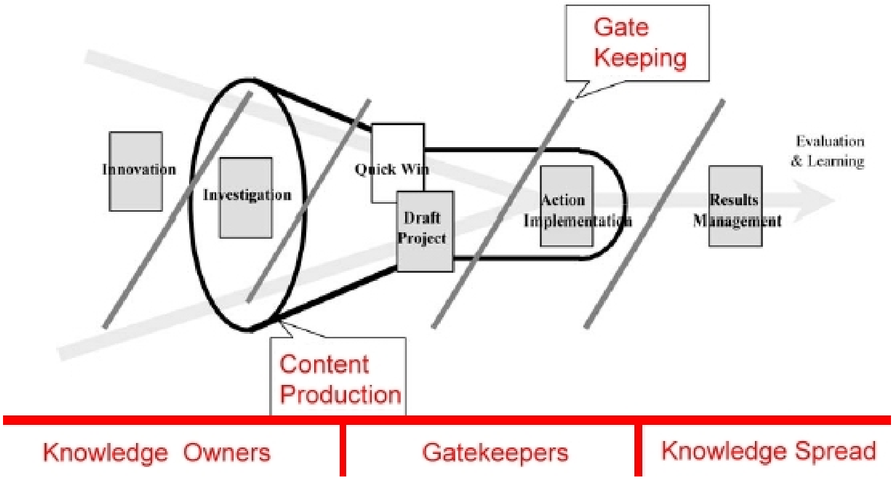
5.2.03 Author System: The Core Function of Your CMS

By interviews with IT managers, we find out that: most internal CMS users will spend most of their time here, so it's important to get this right.

Simply creating an inventory of your existing different digital content assets can be eye-opening, and is a good place to start your efforts. We can almost guarantee that there are more web pages on your present site than you think right now, although much of it may be duplicative (A beneficial side-effect of such an inventory is that you will find all sorts of content that you will want to delete, easing your migration effort later). Tracking down these assets will likely lead you to *people,* since putting thought to keyboard to create and update content is a labor-intensive effort. Your first task when building requirements is to identify these people – the content owners[41].

Note that domain experts often reside outside of Marketing and IT departments, the two traditional institutional homes for corporate web-sites. Content owners can be found in Sales, HR, Customer Service, Logistics, and any other important area of your firm.

Content owners can be split into two general categories: knowledge-owners and gatekeepers. And the relationship of the content production, authoring and transformation could be illustrated in the following chart (somehow from a communication perspective) **(CHART 5.07)**:



(Chart 5.07) The Roles and Processes in CMS (Our analysis)

[41] See also: Goodwin S; Vidgen R: Content, content, everywhere... ... time to stop and think? The process of web content management, COMPUTING & CONTROL ENGINEERING JOURNAL 2002, Vol 13, Iss 2, pp 66-70, IEE-INST ELEC ENG

Knowledge-owners typically create content; gatekeepers validate and approve it. Systems designers sometimes focus inordinately on one group or the other, but a successful workflow should take into account the needs of both user groups. Otherwise, the gate keeping process will easily cause the bottle-neck problems and the effectiveness of the whole content production/communication. In correlation with the design of CMS mechanisms, there are principally the following seven factors that you have take into consideration:

- **a.) Content Retrieval**
- **b.) WYSIWYG Authoring and Editing**
- **c.) Transforming Desktop Content**
- **d.) Content Elements and Reuse (and the case for XML)**
- **e.) Re-use versus Re-Purposing**
- **f.) In-Context Authoring and Editing**
- **g.) WebDAV**

Let's take a closer look at each of these factors of consideration:

a.) Content Retrieval

Within a CMS, content contributors need to be able to navigate quickly to the area where they want to add or modify content. Another term for this is "content retrieval," and it can become an easily overlooked source of frustration.

If contributors can't easily find content on their own site that they wish to modify or update, you risk obliging them to recreate it[42]. Systems that offer "in-context" editing – where contributors browse to an area of the site, log in to the CMS and start editing – sometimes substitute this approach in lieu of a more robust retrieval mechanism. Browsing through rendered content is convenient, but may not show you all the content you need to see. In general, your users will likely want to be able to list content by owner, type, and workflow status. On large sites they may need some form of internal fielded search mechanism that may not be well suited for a simple search engine, for example, to find all the pages modified by a particular author and/or within a particular date range. Understand how this works in any product you consider purchasing.

In some cases, the CMS vendor will rely on their own – typically SQL-based – query tools, while in other cases, they will employ a lightweight version of a 3rd-party search engine (such as Verity or Autonomy) that they bundle with their product.

If it is the latter case, make sure to find out if there is a limit on the number of content items that can be indexed, and whether the search interfaces and scripting have been pre-packaged (sometimes, sadly, that is left to you to code, and it may not be trivial). In both cases, you'll want to know how you can make searchable any custom fields that you will be adding to your repository.

b.) WYSIWYG Authoring and Editing

Gatekeepers require an online queue (ideally with email notification) telling them what tasks

[42] Gordon DH; Whitehead EJ: Containment modeling of content management systems, METAINFORMATICS 2002, Vol 2641, pp 76-89, SPRINGER-VERLAG BERLIN

await them, but otherwise should expect to edit content in an identical or very similar interface to that used by original contributors.

For web-based entry screens, users generally feel more comfortable in a familiar environment that provides both formatting tools, spell-check, and other editorial functions, such as preview. Naked forms-based entry can be easy or hard. In the example below, the author can take advantage of a WYSIWYG editor attached to the text area box[43].

Don't underestimate the amount of training that will be needed, though, for users who don't know HTML. Even such tasks as adding an inline link requires more than basic word-processing skills and you will probably need to set some corporate policies here, e.g. can authors even add in-line links at all, or should they simply enter "related links" into a separate field.

There are three technical approaches to applying WYSIWYG tools to browser-based text area boxes. Most vendors that enable forms-based entry will use one or two types concurrently:

- **ActiveX controls.** These tend to be the most feature-rich widgets (and are frequently programmable). Since they work in the client, they can include spell-checkers with custom dictionaries as well. Their chief drawback is that they can only work in a Windows desktop environment, and frequently require additional DHTML for advanced functionality. Also, you'll need to ensure that your enterprise security policy allows ActiveX controls to be installed and run (they are usually a one-time download).

- **Java Applets.** Like ActiveX controls, these run in the client, and can offer spell-checkers and multi-lingual features. They will run in any Java-capable browser, but tend to be less feature-rich than the ActiveX variety, and because of the plethora of platforms and browsers they must support, can sometimes be a bit buggy.

- **DHTML code.** This is the most light-weight approach, and if your needs are simple, it will suffice. Beware of under tested code, though (it can crash your browser). Also, complex DHTML (usually combined with JavaScript) can increase load times – not always a lot, but often enough to be perceptible to users. The key here is that typically the interface doesn't work until all the code has loaded, and in fact can throw an error if a user tries to access a function whose accompanying behaviors have not yet loaded. Test before you buy.

Note that with all three approaches, you are auto-generating HTML tags, although a savvy user an always view the source. Experienced interface developers know that this brings advantages and drawbacks.

The principal advantages are ease-of-use for non-technical contributors, as well as kind of enforced coding standard. The main drawbacks revolve around the quality of the code, and the sometime need to make adjustments at the code level. All of these widgets have accessibility problems and likewise the codes they generate often fail the more strict accessibility screens. Like all WYSIWYG systems, if you make changes to the same text both through the editor and at the code level – and switch back and forth – unexpected results can appear.

Finally, it is always a good idea – and essential when using an XML-based CMS – to make sure hat the system enforces XHTML compliance on any formatted content that gets written to

[43] Miller R: Content management - Case studies, ECONTENT 2003, Vol 26, Iss 5, pp 23-26, ONLINE INC

the repository. There are various ways of doing this, usually entailing the use of HTML Tidy (an open source tool) or some other filter on the server.

c.) Transforming Desktop Content

If contributors use a desktop package – like MS Word or Quark – to develop and edit authoritative versions of source content, the CMS needs to be able to convert those documents quickly, ideally without manual copy and pasting. As a practical matter, though, many CMS users do end up copy pasting, because the content in the Word file doesn't match the structure that's expected, or because they can control the formatting better than any automated tool does, or both.

Converting documents demands some sort of automated file transformation (to HTML, XML, or plain text) and import into a content repository. CMS vendors typically deal with this in one of two ways:

- **Via plug-in to the specific desktop application.** These plug-ins are more mature in HTML/XML editors right now, but are expanding into word processing and desktop publishing (DTP) programs.
- **Via file explorer.** Users drag and drop files into the CMS from their computer's desktop, as in the screen, below, from Zope.

To convert users to templates or style sheets within Word or any other desktop application presents significant training challenges, at best, and severe change-management problems, at worst. (Vendors usually don't tell you that in their demos, but experience suggests it is quite true.) This is because most content owners are not used to thinking about their content as structured, typically because they were never required to before. But structured content is where much of the value of a CMS comes from, so let's look at it in more detail.

d.) Content Elements and Reuse (and the case for XML)

Most authors will format the appearance documents for their own needs, and desktop programs give them easy means to do so. Also, companies outside of the publishing industry typically do not enforce strict style sheet conventions. This problem merits a minor digression. Many documents are intrinsically quite structured – yet content owners have not usually had to think of them that way (except perhaps for presentation consistency).

But exposing a document's structure extends the value of its content and saves users time – and that is where the real benefit lies. Consider the typical press release drafted in a word processor. It contains:

- A logo
- Headline
- Subhead
- Date
- Contact Information
- First Paragraph
- Rest of the Body

Press releases are fairly uniform and familiar documents, and this example is no exception.

Note that its structure is fundamentally hierarchical, rather than relational; that is, content pieces flow from a central "element" and have parent-child relationships. For example, the Subhead is a child of the Headline element.
In this instance, "Press Release" is a content "type." Other content types on your site could be Case Studies, Staff Lists, or simply, Articles. It really doesn't matter which terms your organization uses, so long as you come up with a common vernacular to communicate document components and structure.

Breaking apart and applying rules against a document's intrinsic structure enables context-specific display and just-in-time content strategies based on personalization. Imagine that the content types which we've mentioned has been transformed to XML and imported into your CMS along with other company press releases. Your publishing system can now automatically display the press release headlines on your home page in bold type.

On your media index page, the server displays head (hyperlinked to the actual release) and subhead in plain text. On the HTML release page itself, the subhead is italicized and of course the whole body appears. Wireless visitors see only the date, headline, and first paragraph. Syndication partners see only the Headline and First Paragraph. And what if the contact person changes? No problem. Change it once in the system and the new information cascades to all releases (assuming you are reassembling dynamic pages – this is a caching issue addressed below).
Of course, this underscores the value of XML and helps explain why XML has become tied closely to web content management. XML provides a way of describing the structure of content, or more accurately, a way for the structure to describe itself. It is particularly well-suited to hierarchical documents. And because the model is inherently extensible, XML documents can typically handle additions and changes to documents much more easily than database structures (where a new subsection in a document might require an additional field to be added to a database table). The example above, however, could also be accomplished in a database-driven CM without XML[44].

e.) Re-use versus Re-Purposing

Thus far we have been talking about re-use in a very general way, but actually we should be more precise. Strictly speaking, re-use means taking authoritative information fragments that live outside of any specific visual context and publishing them out in different combinations to different locales. This might include, for example, product content that might be published in one form on the Web, but then married up with specific case studies for a print brochure.

Another, far more simpler, way to obtain more value from content is by repurposing it. Here you convert a single document to multiple formats, possibly stripping and/or converting some elements for certain media types. At the most simplest level, most content management packages can repurpose Word content to HTML. At a more complex level, you may have XML-based documents that you wish to transform via XSL into more than one output: e.g. one rendition for your web-site, another for wireless devices, and another for an syndication feed. In this case, the transformation is happening on the entire document, and there is no notion of assembly of atomic units. All renditions off the source are visual renditions, as opposed to structural renditions.

The distinction is important, because nearly all content management vendors suggest that they support re-use, but many of them, including many if not most of the XML-based web content management players actually just support re-purposing. And indeed, for most of

[44] Detailed discussions could also be found: Rothfuß Gunther, Ried Christian: Content Management mit XML – Grundlagen und Anwendungen. SRINGER 2001, p.18~36

their clients, repurposing is the goal, and can bring value to an enterprise with multiple delivery challenges. But another reason why you see more content repurposing in the Web arena than content re-use is because the latter is actually quite hard. A solid re-use strategy requires substantial up-front analysis, a more complex content management system, but perhaps most importantly, it requires a major cultural change among content authors and managers to work on content snippets, rather than entire pages and documents[45].

Some functions, such as technical documentation, and some industries, such as pharmaceuticals, can boast of a long history of re-use going back to SGML. The cost-savings were well worth all the effort and organizational change required. For other sectors, the case for re-use is typically justified on a more ad-hoc basis.

f.) In-Context Authoring and Editing

Most CMS vendors now offer "in-context" editing, where, after logging in, you browse through a version of your site in the system until you get the page you wish to edit. Click on the "edit" button, and either a forms-based editing window pops up or some cases you can edit the material right there in the same window. This kind of approach is useful for casual contributors who just need to make quick changes – e.g. the archetypal administrative assistant who has to keep the Intranet phone list up-to-date – but who doesn't need and doesn't want to do more complex tasks in a CMS.

In-context editing is very useful. In more powerful CM systems, you can even rearrange elements on the page or add new elements – if your content model allows. But for content that is being repurposed, you might want to insist on some sort of preview in all the different renditions scheduled for that document. For content being re-used, in-context editing presents an even greater challenge, because the editor is working on text in a specific presentation environment when in fact that content is supposed to be presentation-neutral. This doesn't have to be a major problem, so long as your CMS package can report on dependencies in the system in a useful way (many can't), and content updaters have been trained properly to recognize the downstream implications of their changes.

g.) WebDAV

WebDAV stands for "Web-based Distributed Authoring and Versioning." It is a set of extensions to the HTTP protocols that allow users to collaboratively edit and manage files on remote web servers without overwriting each others' work.

WebDAV-compliant CMS products can enable contributors using similarly-compliant desktop tools to check files and in and out of server repositories directly from the desktop, avoiding proprietary plug-ins and messy (and likely insecure and unsynchronized) file transfers. Most HTML editors, for example, are WebDAV-compliant. Or you can use Windows Explorer to simply drag files into a repository. Also, WebDAV enables an associated XML-based metadata payload that can ensure that key attributes (like author, expiration, etc.) will be maintained and updated along the way.

WebDAV compliance can vary from implementation to implementation, so test before you plunge, but the protocol can be extremely helpful, especially for image management (where the atomic unit for web publishing purposes tends to be a discrete file).

[45] Also see Zarnekow R; Brenner W: Content-management in service on demand-systems, WIRTSCHAFTSINFORMATIK 1997, Vol 39, Iss 5, pp 451-&, VIEWEG. The authors have a CMS project implemented. And some time later, they found themselves tricked by the CMS vendors because the found our that the data volume explode in their systems.

Author's Checklist for Author System

1. Will you require purely browser-based administration and editing?

Most CMSs enable internal users to manage content via a web browser. Administration of the publishing system as a whole (as opposed to just editorial access) may require a thin Windows client or Java applet. For spell-check or WYSIWYG HTML editor, several CMS packages embed a Java applet or ActiveX control in the browser interface. Will your corporate network and firewall allow this?

2. Do your contributors all know HTML (or XML)?

Typically, the answer to this is "no," so most CMS packages include WYSIWYG content formatting tools that support forms-based entry and editing. A question here, though is whether you actually want your contributors formatting their text. Check to see if the interface supports style sheets. This way you can control the look and feel of your presentation globally and override the marketing intern who likes to add in-line emphasis through red, 16-point Times New Roman Narrow type.

3. Will you need to convert documents? Including batch import?

If you are regularly converting large documents, you will want to look into batch import functionality. This is less important if you principally need to perform a one-time load of legacy documents.

4. Are you making an investment in XML?

Some enterprises are aggressively adopting XML; others are taking a wait-and-see approach. The latter strategy is not unreasonable: XML should not be adopted for its own sake and is not the right language for every project. However, if your company falls closer to the former, you will want to carefully examine how your WCM package will leverage your XML investment. Many packages can store content as XML, but much of the power of the language resides in its content transformation capabilities. You may want (or even need) to use XML to simplify getting content into and out of your CMS repository. If your content does not yet reside in a normalized database, then the extra effort of converting legacy content to XML may well be worth it.

5. How much content validation will you need?

Validation is an oft-overlooked requirement when companies transition to a more structured content entry environment. Certain fields required, others need specific types of content. JavaScript can trap for some of this in form fields, but more robust options return to XML and formal validation against DTD or a schema.

6. How much link validation will you need?

Validating that your internal links all still work is a tricky undertaking. Some systems address it at production time, others through periodic sweeps, others at runtime, and still others not at all.

7. Do you require a particular database package?

This is another global consideration, but it crops up early in the requirements process. Your favorite CMS package stores metadata only in Oracle? If you weren't one already, you just became an Oracle shop.

(Chart 5.08) Author's Checklist for "Collaborative Function Model's" System Attribute: "Author System"

5.2.04 Content Integration: The Puzzling Game of Your CMS

First- and second-generation web-sites were primarily collections of static, freeform HTML pages. The HTML mark-up provided presentation logic to the text, but did not define any underlying structure.

Today, companies push much more diverse content to the web, including data based information, binary files of all types, structured documents, and freeform text content. This information frequently comes from multiple databases and varied content repositories. At the same time, web publishers are making more sophisticated use of images and multimedia. The job of a CMS is to make it all come together. At the moment, there are two main issues in this field:

a.) Content Integration

b.) Managing Images

a.) Content Integration

There are typically 4 broad types of content that go into a CMS:

- **Databased information.** For most companies, this means relational data from an relational database management system (RDBMS), which is highly structured, but in a particular way that features rows and columns of data.

- **Structured documents** (such as press releases). These are typically hierarchical in nature and can be split into constituent elements. Today, they may be sitting as unstructured binary word-processing files on your organization's servers or desktops, but the information is quite structured.

- **Unstructured documents.** These include things like e-mail messages, or freeform documents like a one-off case study or your copyright statement.

- **Media Assets.** These may be structured or unstructured. Lower-end CMS packages tend to see them as unstructured, although DAM (Digital Asset Management) vendors helpfully point out that media files can be highly structured indeed. In any case the associated metadata is critical to extending the value of this type of content in your system, and media files typically have unique formats and management needs that need to be addressed in any overall CMS strategy. Some CMS vendors offer browser-based image editing, usually via an ActiveX control or Java applet.

Before you decide for a CMS package, complete an inventory of your own source content types and make sure the system you envisage adequately accounts for them.

If you already have content residing in other active Web-based subsystems (CRM, ERP, enterprise planning and knowledge management) that you want to publish on a web-site or sites, then you will have some critical integration decisions to make.

Each subsystem typically incorporates functionality for updating its own internal content, but it is unlikely to be as robust as you will find within your CMS. If your CMS is your primary e-business platform, then you probably want to run all your content through it. Different CMS packages will force an architectural choice here. Some will want to aggregate that content in the CMS repository where it can be properly versioned, re-chunked, and so forth. Other packages will offer "virtual access" to the content, but leave it resident in a remote repository.

In our experiences by surveying and selecting the most CMS packages on the market, we find that: third party application servers and portals – with their "portlet" interfaces – typically make this type of integration cleaner, and you may wish to aggregate content at the delivery tier rather than within your CMS package. Consider this approach if you do not need to massage the content (e.g. version it, put it through a workflow, re-edit it) coming from another system, but only need to display it on a web page in tandem with the information coming from your CMS. Our fair warning is: this looks nice in diagrams but it is never easy in practice, and relies on careful attention to metadata, which will serve as the glue putting different content items together at the delivery tier.

The rise of Web Services might simplify the process of adding proper content management facilities to other e-business efforts, and vice-versa, but broad adoption of Web Services remains some years off. Most slogans from web solution providers are still in the experiment phase.

b.) Managing Images

In our experiences by surveying and selecting the most CMS packages on the market, we also find that: image management has been long the Achilles heel of all the available CMS packages nowadays. Many web content management systems treat images (and other media) as generic binary files, such as PDFs, when in fact images and other media represents an increasingly important part of web content – and therefore need to be managed as such. Aggregating image and text content is therefore a key challenge facing any CMS project.

To manage images on web pages, editors usually need (at a bare minimum) access granular image libraries, with images sorted by type or other attributed, all according to appropriate access rights.

Most CMS packages today will let contributors peruse thumbnails and manually add images to pages. A CMS would ideally store key metadata about each image (such as alt tags, height and width, perhaps even origins and rights), but as a practical matter few do. Some CMS vendors will point you to the new DAM product they acquired, but be careful here; you may risk paying US $100-200,000 more in licensing fees for the privilege of managing your media content in a browser.

Many CMS vendors now provide capabilities for editing images on the fly in the browser. New widgets are popping up, including a slick ActiveX control, and a variety of Java applets that some CMS vendors are incorporating.

However, this capability is a double-edged sword. You may want to allow a marketing person to crop an image or modify its size, and perhaps even overlay some caption text. There are other capabilities – such as blurring/sharpening images or changing color depth – that might be best left to a designer. But the good news about these widgets is that they offer the ability to annotate images and put them through a meaningful workflow, thereby automating a process that previously only happened offline.

Finally, these rich media editor widgets are new, and may prove to be a bit buggier than the rich text editors they are

Author's Checklist for Content Integration

1. Do you anticipate tapping multiple content repositories?

Authoritative versions of digital content may reside in different places. These locations could include different countries within a multinational enterprise, different places vertically within the company (e.g. back-office ERP databases and front-office file systems), or different places functionally within the company (e.g. HR and Marketing). A single system that can put them all together for web presentation is difficult, but may be necessary.

2. Do you need to aggregate content at the CMS tier of the Delivery tier?

If you have a requirement to publish content from remote repositories into your web-site(s), you'll need to decide whether to put that content under management or publish it directly to a web server, portal, or application server. If you need to add value to that content in some way before it hits the web – put it through a workflow, add metadata, strip away in-line content elements, build an audit trail, etc. – then you'll want to aggregate that content via your CMS, and will need to select a tool capable of tapping those repositories. If, on the other hand, content from your other repositories can be published out to your web-sites as is, there may be no reason to complicate things by running it through your CMS first.

3. Will you need to relate database information to documents and vice-versa?

Do your present databases hold isolated content, or will you need to make associations between database content (such as customer records) and other structured and unstructured text content?

Customers don't care about internal walls between your data.

4. Do you anticipate incorporating e-mail into your site content?

Only a handful of WCM products actively integrate e-mail messages into workflows and content repositories. It can be deceptively difficult, so consider it carefully before proceeding. If the target site is an Intranet, ask yourself whether you aren't really creating an enterprise portal, and therefore might be better off with a Knowledge Management product. KM offerings typically encompass e-mail repositories, but usually lack workflow and other content management functions out of the box.

5. Will your site be delivering substantial numbers of formatted files, such as spreadsheet files and media assets?

If the answer is yes, you need to make sure that your CMS understands and can treat these files as the digital assets that they are, as opposed to text chunks that can be easily data based and recalled with simple SQL queries. Systems with a background in document or asset management are likely to have more robust support.

6. Will you be importing syndicated content?

If so, then you will almost surely require a WCM package that can read in and parse XML, the de facto standard for syndication repositories. Meta group estimates that by 2002, 85% of corporate web-sites will make use of syndicated content (although only an estimated 20% will offer syndicated content externally, a number that could rise as B2B content requirements expand).

7. Will you allow non-designers to edit images?

If so, then seek out CMS packages that offer browser-based image-editing utilities. But remember you will want to carefully control here who is allowed to edit what. Look for widgets that allow you to configure (i.e. turn off and on) specific features against particular roles.

(Chart 5.09) Author's Checklist for "Collaborative Function Model's" System Attribute: "Content Integration"

5.2.05 Metadata: The Value-Adds of Your CMS

Sometimes lumped under the general category of “tagging,” this technically concerns building, managing, and applying content classification systems. When users – or systems – apply metadata, they are classifying content. Classifying content is another key factor in liberating the underlying value of your information.

The first major benefit to tagging content is that it helps people find things. Metadata enables your search facilities to become more targeted and efficient. How so? Imagine your library with no catalog system, no labels on the shelves, and books without title and author info on their spines.

Finding the specific information you seek would be extraordinarily difficult. Today’s full-text search engines are only slightly better than going into the library above. Metadata enables you to provide greater meaning and cues, without which content is just text. Tagged data enables dynamic generation of tables of contents, and vastly improves search engine results relevancy.

A second major benefit to tagging is establishing relationships. Content element or documents can relate to each other in time (“sort by date”), subject (“show me both the 3-D image and detailed specs on the product line I’m viewing right now”), and other attributes. Tagging enables this. Tagging is especially important for establishing ad-hoc relationships among hierarchical data; relational databases were specifically designed to accomplish this using “key” fields that relate data in different tables to each other.

As noted previously, metadata is almost always the key to automated site navigation. That is, content doesn’t appear somewhere because an author put it in a particular place – it’s there because the document put itself there.

The automation here can be quite powerful and has obvious benefits for site owners and site visitors alike. Recognize however, that like all automation, you are giving up some level of granular control over layout and navigational patterns. Much the same way you standardized around a core set of templates and document archetypes, so are you standardizing around a set of categories and relationships. A good system, however, will allow you to make exceptions.

In any case*,* if making your content more intelligent and mobile is an important part of your CMS strategy, then pay close attention to how your vendor deals with metadata and classification. In particular, take careful note of how metadata values map you’re your content management system to your content delivery system. If your portal software can’t make use of subject metadata to build dynamic navigation structures, then your author’s work to complete “subject” fields in content entry forms went largely for naught.

Assigning metadata is often seen as the job of content contributors and editors – who else can better apply meaning to their content? Of course, some metadata can be applied implicitly, or transferred from a source document. This includes such attributes as date, file type (where relevant) and size, author, approver, and so forth. The availability of this new implicit metadata is a powerful reason to implement a CMS in the first place.

However, because entering explicit metadata requires extra manual effort, people generally don’t like to do it. If tagging requirements are particularly onerous, contributors will rebel and either put in bad data or find work-arounds to avoid this work altogether. Most CMS packages will enable you to apply rigid technical controls to prevent this, but a better strategy revolves around getting buy-in from contributors for the extra work.

When we tried to persuade our Taiwanese employees to do this extra job, we did had great internal pressures. However, the heart winning secret is: You must point out to your contributors that good tagging likely means that more site visitors will see their content, and that since the contributor herself will find the text more readily in the future, updates will come easier. Explain that metadata is critical to whatever business objectives you are pursuing by linking content to core products and services that pay the rent. Then in the contributors' tagging interface, be sure to employ intuitive, easy pull-down menus with pre-set options. The industry phrase for this is "controlled vocabularies."

Besides, a very important lesson which we have learned by implementations of the METADATA schema in our research project is: to avoid making contributors do too much tagging at the start, and keep your classifications schemes simple.

Yes, metadata provides an important value-add within the system, but like workflow, it can be overdone. Keep in mind that you will need to update your vocabularies persistently as content changes, and good contributor feedback mechanisms will be essential to keeping taxonomies relevant. The important thing, though, is to just do it in the first place, especially if your web-site exceeds, or is going to exceed 1000 pages. Without metadata, you won't be able to keep track of all that content.

Author's Checklist for METADATA

1. Can you live with limited metadata sets?

Some basic CMS packages limit metadata fields to keywords and description fields. This won't work for you if you need to implement a custom scheme with your own vocabulary.

2. Who and what can have access to metadata and vocabularies?

Your search engine should certainly be tuned to leverage any metadata. Will you need your internal reporting systems to tap into it too? For example, do you need to know how many times a given tag has been applied? How easy is it to update vocabularies in the package you are considering? With iteration and learning comes revision, so changing the overall taxonomy and individual vocabularies should be simple as possible – ideally using the CMS itself to manage versions and approvals.

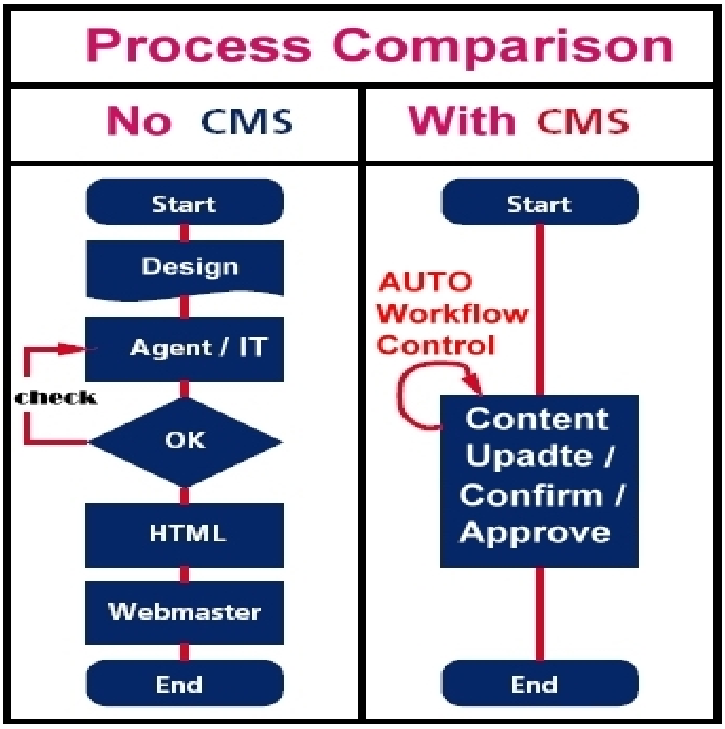
3. At what level do you want to tag content?

It is theoretically possible to apply metadata to content at any level, including folders, documents, content elements, even individual words. As with everything else, the more granular your system, the more complexity you introduce. Multiple levels of metadata can also have performance implications for the system at large. Your business objectives and workflow should guide where you fall on the spectrum here.

(Chart 5.10) Author's Checklist for "Collaborative Function Model's" System Attribute: "METADATA"

5.2.06 Workflow: The Gate-Setting of Your CMS

Workflow is how content gets "from thought to end-user" through a set of interdependent tasks that occur in a specific sequence. A workflow process can enhance systemic stability and reliability by allowing managers to align content paths with core business and editorial rules.



(Chart 5.11) CMS: The process comparison. This chart is "exclusive authorized" by the Infopark AG to privileged academic uses. Please see the appendix for more Info.

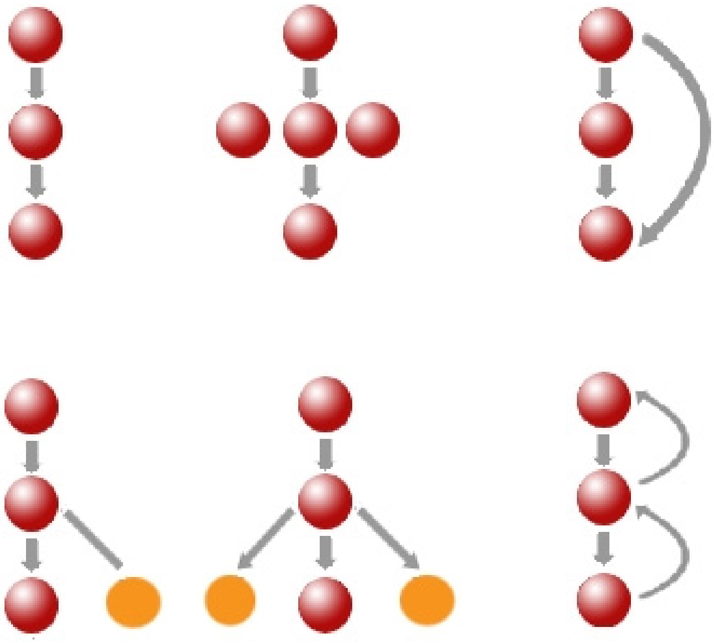
Done right, workflow systems can improve time-to-adapt within companies of all sizes. Executed poorly, a workflow system can gum up an otherwise perfectly good publishing system. If you haven't already done so, take the time to model your existing web-site workflow process – what you find may surprise you. It's exactly just like the work flow which is shown the left column of the chart above **(CHART 5.11)**..

In reviewing CMS packages, look for rules flexibility, including support for ad-hoc workflow and rerouting mechanisms. A package may contain 12 preset status identifiers, but what if one you need isn't on the list? You may well need this flexibility, because web content workflows tend to become more collaborative than other business processes that you might be automating. There are several different ways to model content. We always recommend, however, starting out with different content types as the base element. Some of the best models simply use tabular charts, with each row representing a different content object, each column a particular state that the content resides in, with each relevant cell describing what is supposed to happen to that content in that state.

Most advanced CMS products contain visual workflow modelers built in. Others will import specially-filtered Visio flow diagrams. The keys are abstraction and flexibility[46]. Note, for example, that in many CMS packages, content flows through different, highly-generic states, in which developers and managers can apply certain specific rules. Others follow a process-flow approach where activity and particular steps are centered on particular tasks, rather than the content itself. In the chart below, we have some demos of the possible situations that you might encounter **(CHART 5.12)**.

[46] Stein T: Intranet-organization - how to get benefits from connected enterprise networks by content management, WIRTSCHAFTSINFORMATIK 2000, Vol 42, Iss 4, pp 310, VIEWEG

In either case, you'll want to model your workflow using abstract roles, rather than saying "releases go from Bob to Lisa to Bill." The model is actually that releases are edited and then deployed. Today, Lisa may serve as the reviewer and Bill the deployer, but the people may change (or one of them may be out sick), while the workflow may not. This is absolutely the wring way to create a workflow that fit your needs.



(Chart 5.12) Some Simple but Practical Workflow Chart Portfolios. (Our Analysis)

Instead, thinking in terms of abstract titles, taking the chart in the left side for some illustrations, as well as content states and transitions helps you better understand the different situations your team is likely to encounter. In general, however, you want to avoid the creeping process-flow diagram. The simple process of modeling your workflow and introducing automation can tempt participants to insert new and potentially unnecessary steps into the flow. Yes, business rules need to be applied and enforced – if the CEO must sign off on all press releases, you'll want to prevent ad-hoc rerouting at an earlier point.

But if you overcomplicate a process or represent a workflow in way that doesn't actually work in the real world, users will rebel, often by circumventing the system. Stories of this are legion. We know of one public portal where content owners figured out how to FTP files directly to the staging server to avoid an onerous and buggy workflow process.

Moreover, one of the opportunities you have in implementing a new CMS is to streamline your processes – that's where time and cost savings lie.

One final caution on workflow modeling: make sure that your vendor can demonstrate using the same visual modeler in any CMS to make significant *changes* to an existing workflow process as well as create new ones. If you need to use underlying scripting to make changes to a particular workflow process, then you can be sure that the GUI editor is out the window going forward.

Author's Checklist for Workflow

1. Will your workflow include branching and looping (as opposed to simple linear tasking)?

When tasks and content begin to peel off in different directions, basic workflow systems come under stress. If your workflow requires branching and reconciliation, or complex looping chains, make sure your CMS package can support it.

2. Will you require monitoring and notification?

If contributors and editors sit more than 30 meters apart (the interpersonal universe of the typical cube dweller), chances are you will need automated mechanisms to inform actors in the system of the status of various tasks and/or content elements. Look for systems that generate alerts once certain deadlines pass. If an editor should suddenly take ill, how would others know to fill in or reroute?

3. Auditing and Reporting: what do you need to know?

And when do you need to know it? Identifying bottlenecks is a real benefit. You might also want to audit individual performance and other metrics.

4. Which is more important, speed or accuracy?

Like it or not, speed and accuracy often present a direct tradeoff. The fewer eyes that need to review a content element, the faster it gets to staging and production servers. This doesn't mean that companies with complex workflows are destined to be slow, nor that firms with simple approval mechanisms will inherently publish faster. Consider the time-to-market value on each of your documents as you examine a digital workflow.

5. The CEO wants an item published in five minutes; can you do it?

Your CEO needs to respond to an event in the stock markets and wants her piece online right away. It's times like these that a sclerotic workflow system can really get in the way. Make you're your system can handle unusual contingencies.

6. Are you considering Inter-enterprise contribution and workflow?

If you are, prepare yourself for substantial customization effort. There is a dearth of commercial tools and approaches to handle the complexity of inter-company workflows, and worse, no clear standards n the event two companies are using different WCM platforms. Look for rapid improvement here in he next year or so. Collaborative commerce – and the attendant complex content requirements – is forcing companies to manage content publishing jointly, and software vendors are sure to follow, most likely with approaches leveraging new Web Services standards.

7. Do you already employ a particular workflow modeling mechanism?

Some WCM packages will use Visio diagrams. For example, Stellent, Tridion, and Vignette offer realtime integration with Visio, allowing you to edit the properties of each step. Other products provide built-in WYSIWYG modeling tools of their own. In still other packages, workflow can only be modified through a forms interface or technical API.

(Chart 5.13) Author's Checklist for "Collaborative Function Model's" System Attribute: "Work Flow"

5.2.07 Templating: The Cinderella's Dresses of Your CMS

At their core, all CMS packages are templating engines. That is, they enable you to publish content out through preset visual and organizational models, or "templates." Note that when CMS vendors refer to templates, they sometimes conflate content input templates and content output templates.

We will concern ourselves more with the latter. Input templates are important, but are typically tied directly to the specific content type, or some intersection of the content type and contributor role.

In our experiences by selecting the most suitable CMS package and partner for our international cooperation project, we find that: CMS packages vary substantially in how they enable you to create templates. Some CMS packages provide their own WYSIWYG tools for template creation, but you will usually profit from using a real HTML editor to build these. Most CMS vendors have gotten wise to this and integrate closely with at least one standard editor like Dreamweaver, while allowing you to copy-paste from others.

Some CMS packages employ their own set of proprietary tags – like "<publish>","<include>", or "<hide>", and such – to extend HTML and give page designers direct access to some of the dynamic features within the system[47]. This is a blessing but also a curse: a blessing because non-technical designers can very quickly build more robust templates on their own; a curse because proprietary tags lock you in to a particular solution and increase any migration expense later, and may not be sophisticated enough for the sorts of template logic you want to express.

Therefore, template flexibility and component granularity are key considerations here. Some lower-end CMS packages simply dump the entire "content well" of a page into a single text area-editable element.

This may be too simple for sites that need a broader range of page display archetypes. Other midrange packages employ concepts of "inner" and "outer" (or "wrapper") templates, each made up of standard elements. Higher-end, more flexible packages will typically allow for very granular control of templates, with a hierarchical model for template development where you can create and manage multiple variants of single template models.

Nevertheless, resist the temptation to develop an unnecessary plethora of templates. The more templates you have, the more you have to maintain. Your contributors – if they have a choice – will vote with their keyboards and probably select a subset of the most useful templates anyway.

There is also the related question of where and how templates are stored. In systems where templating becomes primarily a technical task, packages tend to assume that you will store templates offline (as with Microsoft Content Management server – see screen below from VisualStudio.NET), perhaps in some sort of code repository like CVS. This has its benefits for developers, but is one more asset to sync up in your build processes. Other packages allow you to build and modify templates within the system as actual CMS assets, perhaps with an assist from Dreamweaver.

Also, keep an eye out for design and layout limitations. Does the template system presuppose a certain presentation model? In particular, look for problems with DHTML and Javascript, since some page generation engines aren't expecting to see formal objects outside what they

[47] Rothfuß Gunther, Ried Christian: Content Management mit XML – Grundlagen und Anwendungen. SRINGER 2001

would parse on the server. If you expect to use a complex, DHTML-driven design, ask your CMS vendor to show you other comparable sites using their technology.

Author's Checklist for Templating

1. Do you already use Dreamweaver Templates?

Many organizations successfully control standard layouts using Macromedia Dreamweaver templates (.dwt files). This is a nice interim approach before investing in a CMS, but doesn't always scale well and locks you into a file-based system. Nevertheless, some CMS packages can import and re-use DWT files, and even allow you to manage your templates on an ongoing basis using Dreamweaver utilities

2. How many templates do you really need?

To paraphrase Albert Einstein, you should build as many templates as you absolutely need – and not one more. More templates means more objects to manage, and more to modify when your design changes. If you publish a large site with many divergent templates, be sure to investigate how the CMS allows you manage those templates (and associated elements). Conversely, if your site layout consists of many variants of just one or two templates, you'll want to put careful consideration into the hierarchy of elements such that you understand the cascading layout effects of modifications to elements higher up in the tree.

3. Will your page bodies consist of multiple, reusable objects?

Some organizations want to deal with the page body, or "content well," as a single chunk, contained within an outer "wrapper template." Among other reasons, this gives authors great flexibility in applying unstructured content and, to some extent, the ability to define the layout of the well space. Nearly any CMS can accommodate this easily. If, on the other hand, you need to standardize elements in the page body, and/or your layout makes a less absolute distinction between "wrapper" and "content," then you will require a mid-range or higher CMS that can take a more granular approach to laying out content elements.

(Chart 5.14) Author's Checklist for "Collaborative Function Model's" System Attribute: "Templating"

5.2.08 Versioning: The Reliability of Your CMS

"Versioning" is shorthand for a system that keeps people from stomping on each other's work in collaborative environments.

This is typically enforced through document or element check-in / check-out facilities that prevent two or more people from working simultaneously and unknowingly on the same content. MS users in distributed publishing systems often find version control tremendously helpful, but it sometimes can come at the expense of collaboration. The idea that only one person can have a content item checked out at any one time an lead to very linear processes and sometimes added stress on workflow outlines (such as excessive looping). Some CMS vendors have responded with specialized capabilities for multiple editors or work on a content item simultaneously – typically by "saving" it but rather than submitting" – before promoting it to the next step in a workflow.

Versioning enables you to track changes, so you can audit and monitor what changes have been made to any element or asset. Some CMS packages have done this in a sophisticated way relative to desktop tools, while other packages perform change tracking in a simpler, side-by-side way that can also be useful.

"Versioning" has also become a major liability-related feature, and CMS vendors have horned in on this as a critical selling point. As a company you are responsible for what you put online, but whereas print content may be frozen in time, web content is dynamic, so how can you track what your site actually said at a particular date and time in the past?

For legal reasons, even companies in largely unregulated industries have to be able to recall exactly what their web-site was publishing at any given instant. The ability to "roll-back" a web-site to that time is therefore essential, and this is why a sexy new rationale for making a CMS investment has emerged: it's a necessary insurance policy.

A somewhat loftier benefit is the ability to perform date- and time-aware publishing. Digital publishing enables you to pre-schedule for content release *and* expiration, as well as any number of interim steps in-between (perhaps syndication or archiving). Expiration in particular represents an oft-overlooked benefit. You dilute the value of your current content by keeping outdated material in your repository, where, among other things, it can show up in search results.

Author's Checklist for Versioning

1. **Do you require version branching and reconciliation?**

   If you do, it will immediately push you to a mid-market package or better. Reconciliation can be tricky; if you anticipate doing a lot of it, take a look at the management interface of any WCM vendor that says they can handle it.

2. **What is your content lifecycle?**

   Is there an automated archiving or syndication date? And do you need to pre-produce content for publishing on a certain date or at a certain time? Are there interim steps where pieces of content have to be moved or revised while still "live" in production, but short of being archived?

3. **Do you anticipate needing to be able to roll-back the site? How far? What elements?**

   You don't require a complicated system if your principal concern is being able to return to a previous version of something after making a mistake. If you need to recreate your entire site at a given date and time, you will need a more powerful CMS package. Find out what is versioned: code fragments, content elements, documents, images?

(Chart 5.15) Author's Checklist for "Collaborative Function Model's" System Attribute: "Versioning"

5.2.09 Globalizing: The Multi-nationality of Your CMS

In a global publishing environment where less than half of Internet users understand English, globalization/localization is critical to the multinational enterprise. There are at least two major implications for your CMS effort:

- Your software may need to support multiple regions and countries, as well as multiple languages and dialects

- You'll require a process for converting or adding content to create region- and language specific pages, both with respect to the text, but also the graphics and all the other elements.

Most global companies have figured out that they do need to localize. They may centralize certain messages and materials, but they leave key decisions about the way that information is communicated – and in what language – to their local offices[48]. Indeed, localization represents far more than just text translation. It is about doing business in a way that makes sense for any particular country. Doing e-business is no exception.

However, these same companies generally have not automated the mechanics of globalization localization. According to a research in Germany, 84% of international companies with more than 20 web-sites still localize site components manually[49].

Localization may be the weakest area of CMS packages, though many are partnering with other specialized vendors and quickly trying to make up for lost time. What is worth to mention is that, in the process that we search for CMS package and business partners, we found out that European CMS vendors tend to provide better facilities for localizing interfaces as well as more sophisticated systems to manage multiple language versions of the same content. They have configured their version-compare systems to show the differences between a source text and the revised version, so that a translator can quickly make the same revisions in any local version.

In the meantime, a slew of professional services firms and companies with software add-ons have sprung up to help companies localize their CMS infrastructures. If globalization is important to your firm, you'll want to carefully review the partnerships your CMS vendor has in this space.

For a multinational site working off one platform, you'll want to look at the technical, organizational, and interface aspects of localization at every layer of your CMS, including:

- Database repositories, specifically how they handle character sets. Look especially for UNICODE 2.0 support. UNICODE is a standard that encompasses nearly fifty-thousand characters from all the world's major written languages.
- Administrative and content-contribution screens – especially the default ones that ship with a CMS package – as well any help screens.
- The public interfaces, including any search engine.

There are also critical procedural issues to consider. Some CMS packages, for example, will employ time zone synchronization to make sure that content is published at exactly the same local time on your different web-sites around the world.

Author's Checklist for Globalizing

1. Will you need to support non-English languages or non-Latin character sets?

If so, your CMS package, and any databases it uses, will need to support this. The most common international character set standard is UNICODE. Pay particular attention, though to languages that read right-to-left (such as Arabic), and test how your CMS will handle these.

[48] *Stein T: Intranet organization - how to get benefits from connected enterprise networks by content management (vol 42, pg 310, 2000), WIRTSCHAFTSINFORMATIK 2000, Vol 42, Iss 5, pp 476-476, VIEWEG

[49] Gersdorf R: Possibilities of content management, WIRTSCHAFTSINFORMATIK 2002, Vol 44, Iss 1, pp 75-78, VIEWEG

2. Is your workflow international?

Sending same piece of content to be localized on multiple sites requires branching. And if the source content changes, you will want to trigger a revision in its localized replicas as well. This raises a very practical issue for translators: how will they know what has changed.

3. What languages will you require in the editorial and administrative interfaces of the CMS?

Some CMS vendors have translated their default administrative, editorial and even help screens to non-English languages. Others have not. In most packages, these interfaces are primarily templates that are accessible through an API, so you could translate them yourself. But would you really want to? What happens when it comes time to upgrade to a new version?

4. Do your Images frequently contain text?

If so, then you will need to localize those as well, and build image changes into text-localization workflows. Some CMS vendors, such as Day, allow you to store a root image in a high-resolution format, then publish out different renditions (presumably with text in different languages) for distribution to local web-sites.

5. Will you and your users need to search in other languages?

Don't assume that the same search engine can function effectively in different languages. You'll want to test out any bundled search engine against a repository in languages you intend to employ.

(Chart 5.16) Author's Checklist for "Collaborative Function Model's" System Attribute: "Globalizing"

===

The Dimension of Content Delivery

In the "Collaborative Function Model", we define "Content Delivery", similar as the most popular term "Publishing" It encompasses what happens when a user visits your site and starts clicking. You can also think of it as the "consumption" analog to earlier production phase.

While you could consider the attributes that we are going to discuss below as "run-time" features, in fact, they all have critical design-time considerations. For example, how your users search your site has very much to do with how you indexed your content.

However, for most content managers, Delivery also includes what happens to content beyond the confines of your production web server, almost something like cross media publishing. You may not be thinking about this broader content distribution now, but you may be soon.

5.2.10 Page Generation: The Dynamics of Your CMS

Content has value only in the eye of the beholder, so this is where the investments you made in better content management need to pay off: when a user actually visits your site. Web-sites that serve static pages from file systems need only a web server.

Indeed, web server software is optimized to do just that: read a file off the disk and serve it up. If you are incorporating other dynamic logic, engaging in any customization, or checking for highly-changing content, pages must be assembled at the time a user calls up the page – that is, at "run-time." A common expression for generating pages dynamically at run-time is "frying" them.

The notion of dynamic assembly is not new; it dates to the earliest days of web with "server-side includes," where snippets of text from separate files would be inserted into master HTML pages.

The technology for frying pages is much more sophisticated now, enabling site owners to pull from different repositories, use conditional logic, and implement dynamic navigation and presentations. For this, you typically require an application server of some kind.

That application server does not need to be an expensive, Java-based product. At a basic level, Microsoft's Active Server Pages and Apache PHP (both scripting engines built into the web server itself) as well as Cold Fusion are sufficient lightweight application servers.

For things like transaction controls, persistent session management, database connection pooling, or more sophisticated forms of personalization, you probably want a heavier-duty application server, like BEA WebLogic, SunONE, or IBM WebSphere[50]. Note that some CMS products still have their own built-in application server, either homegrown or OEMed from another vendors, although the industry trend is certainly to use standard 3rd-party application servers.

Since frying can prohibitively slow down a web server, some CMSs are set up to "bake" dynamic pages to static files that then reside on the web server's hard disk. From then on, the web server spools them off on request, achieving very high performance. It also reduces risk of downtime, since static pages do not need to concern themselves with connectivity to a live database.
Typically, such page generation routines can be scheduled to run in batches, or individually as separate documents are updated. Obviously, batch-based updates mean that content on the file system may differ from that in the database until the next "baking" session synchronizes the two again.

If you take this approach, make sure that pages that are dependent at all on an asset that is baked are reassembled and re-baked as well. For example, if you change an image name or the location of page, other pages referencing those assets need to be updated and regenerated as well. "Baking" sounds simple, but in reality can be quite complex, especially with respect to deployment rules and integrity.

Some CMS packages support hybrid publishing systems that enable you to bake infrequently changing elements (such as navigation bars), while keeping other page elements (e.g. those used for personalization) dynamic. For example, the system may pre-assemble all the data based elements of a page – except the header – into a JSP file. The header contains some personalization logic that needs to be evaluated at run-time, via a snippet of JSP code. We call this approach "parbaking." These issues are addressed in more detail below under "Caching."

Author's Checklist for Role Management

[50] Cody WF; Kreulen JT; Krishna V; Spangler WS: The integration of business intelligence and knowledge management, IBM SYSTEMS JOURNAL 2002, Vol 41, Iss 4, pp 697-713, IBM CORP

1. Do you already use an application server?

If your enterprise has already invested in a particular flavor of application server – has grown deeply knowledgeable about its particular quirks and features and built various applications within its framework – then you may want to know which CMS packages can work with it.

2. What programming languages are you comfortable with? APIs?

This is another global consideration but let's address it here. You (or your consultants or integrators) may be more or less comfortable in certain programming environments. This is a critical consideration, since CMS packages are complex and almost always require writing custom code to achieve sufficient integration.

3. Will you need to incorporate logic from other subsystems? How so?

At a more complex level, this refers to what other major systems you may need to integrate, such as ERP, CRM, etc. At a more simplistic level, it could refer to particular business logic around the way certain pages are displayed. On weekends, for example you might generate lower bandwidth presentations for home users. Typically, this is done through application-server interfaces.

4. What are your performance (speed) requirements?

This is an easy one: everyone requires fast pages. But there is fast and faster. How screens get generated is a prime determinant for how fast your site will perform. Keep an eye out for knows performance hogs, like server-side "includes." This issue is discussed in more detail below under "Caching."

(Chart 5.17) Author's Checklist for "Collaborative Function Model's" System Attribute: "Role Management"

5.2.11 Searching: The Readability of Your CMS

If your site's content is complex enough for you to consider a CMS, imagine how it appears to your users. On content-heavy sites, most visitors will turn immediately to a site search engine if what they can't find what they seek after one or two clicks. Indeed, a search engine often best meets the primary need of site visitors: rapid access to just the information they want.

However, a lot of hidden complexity can lie behind search engine implementations. Don't make promises to users about relevancy rankings or the breadth of your indexes if you can't keep them. If your search implementation cannot simultaneously query your catalog data and your unstructured documents, then you should provide users a choice of repositories via a pulldown menu to set their expectations.

A Good CMS Can Improve Your Search Engine

As a matter of fact, search engine results will quickly expose the quality of some of your metadata, including page titles and descriptions tags[51]. Great content with bad or missing titles is unlikely to be found by end users.

[51] Hall WP: Maintenance procedures for a class of warships: Structured authoring and content management, TECHNICAL COMMUNICATION 2001, Vol 48, Iss 2, pp 235-248, SOC TECHNICAL COMMUNICATION

The good news here is that your CMS should provide a way for you to easily apply the all important title tag, description, keyword, and other custom metadata that will improve the quality of your users’ searches as well as the presentation of the results. The CMS will also capture other implicit metadata, such as how old a document is, that can be very helpful for search results.

A good CMS can help your search engine in other ways. Ideally, the CMS package can be used to store and manage your taxonomy, helping you keep it fresh and accurate. Your CMS can create more sophisticated rules for content elements (such as navigation or session elements) that you don’t want indexed, including the ad-hoc assignment of “don’t index” tags by authorized editors.

Besides, a good CMS can print a special, hidden sitemap of links just for your search-engine crawler. At the same time, the imposition of a CMS may make your search effort more complicated, and you will certainly want to re-tune any search engine after launching a CMS. It may also you’re your search efforts more costly, particularly is CMS user requirements – which can be fairly demanding – are taken into account. Indeed, some features within your CMS, such as reporting and task analysis may be taking advantage of a search engine in the background, and you may have some integration effort to expend to get it all to work, especially if you want to employ your existing search software, and not the package bundled with your CMS vendor.

A Good Search Engine Can Improve Your CMS

Vice versa, a good search engine that indexes your CMS content will also lead to happier authors: they can find that article they wanted to edit, more easily. Managers can become more powerful, because they can find the data they need to manage (e.g. how fast are workflows being cleared?).

However, effective search integration with your content management system often requires additional, specialized resources and expertise, and you may have neither close at hand. For example, you’ll need to figure out whether you want to index across versions of content, and across separate physical repositories (e.g. work-in-progress versus staged content). There is a case to be made for both (an author might want to search for all instances of a single content item, regardless of status).

In some cases, CMS vendors have built-in SQL or X-Path queries for this; in other cases they assume that you will have integrated a search engine to do the job. Many advanced CMS packages do come with a bundled, “light” version of a 3rd-party commercial search engine for the purposes of content retrieval for content managers working within the system. These will still require some tweaking against your retrieval needs. These packages generally cannot be used in a delivery environment, and usually come with a mixture of other limitations as well, including the:

- size of repository, and/or
- coverage (e.g. index PDFs?), and/or
- concurrent users or allowed environment.

In fact, although these search packages presumably hold the benefit that they are pre-integrated, they are supplied in the hopes that the customer will become incensed to purchase the fully functional version of the product from the search vendor. You MUST check your contracts carefully.

In addition, the newly-empowered content managers working in a CMS will start modifying content structures in way they couldn't before. That's generally a good thing and in fact represents an important reason to implement a CMS. However content managers may consistently put stress on your search engine configurations by:

- Modifying content types (e.g. adding new elements, changing element names that your search engine was looking for);
- Adding and removing directories;
- Creating new workflow states (that might go un-indexed);
- Creating a nearly infinite number of concurrent (e.g. sibling) versions of the same content item; and
- Opting out of irrelevant metadata vocabularies (by figuring out how to skip them or putting in bad data).

The good news is that you should be able to use that same search engine to find all these particularities, but you'll want to check often.

Author's Checklist for Searching

1. Have you already selected a search engine separate from your WCM package?

As what we've noted, while some CMS packages bundle their own Search Engine or OEM a separate product, increasingly, CMS vendors have begun to leave the choice up to their customers to select a "best-of breed" search product. If you are already set on a particular search engine, make sure that it can be integrated successfully with your favored CMS packages.

2. Will you require fielded as well as full-text searching?

There is a classic performance trade-off between fielded and full-text searching. Some search engines are optimized for fielded search, others for full-text, and other (more expensive) ones for both.

3. How often will you need to update your indexes?

When content is entered? Can it wait until an off-peak time? Indexing can be server intensive, although most search-engines have mechanisms for fairly lightweight updating of incremental additions. Perhaps more importantly, you'll want to configure your deployment mechanism to trigger this incremental update whenever you promote fresh content to your site(s).

(Chart 5.18) Author's Checklist for "Collaborative Function Model's" System Attribute: "Searching"

5.2.12 Personalizing: The Uniqueness of Your CMS

In the "Collaborative Function Model", we define "Personalizing" as generating custom content or presentation for web users. Since it fundamentally concerns itself with mixing and matching content elements – and seems on the surface to be a self-evidently very good idea – many CMS packages have wandered into the personalization space.

But many major CMS packages are now returning the personalization function to a separate application server or 3rd party personalization engine. Why? Because personalization has proven to be a highly specialized and quite complex domain.

Personalization can also be very resource-intensive from a design and publishing perspective. It takes a lot of effort to define and implement business rules against various use cases, then test carefully across various permutations. Then, serving up custom pages can add dramatically more server cycles to what may already be a very dynamic publishing process. Beware slower page loads. In short, the hidden costs of personalization can easily equal or exceed the initial licensing and development costs.

Note that there are several approaches to web personalization:

- **Pre-Defined Affinity Groups**

  Also called "roles-based" personalization In this model, site owners and editors pre-determine groups of users and create custom pages or content sets for those users. This approach may well be the easiest to automate and might not require software solutions.

- **Rules-based**

  In rules-based personalization, some level of artificial intelligence is employed. Site owners and editors establish business rules and conditions that, if met, affect the display of content on pages within a site. Example: an online gourmet store is overstocked in French Roast coffee; the site software recognizes this in the inventory database and automatically places promotions to users who have bought coffee before.

- **Preferences-based**

  In this model, users indicate their preferences, usually during an initial session, so that dynamic pages can be delivered on subsequent visits to the site that better match their interests. Users are typically assigned a username and password for authentication; cookies are often also enabled so users aren't required to login each time they revisit the site.

- **Collaborative filtering/community based**

  In this approach, site software uses complex statistical techniques and automated analysis to determine elements of a personalized page.

An example of "Personalizing" is Amazon's "people who bought this book also bought" service. This approach can also incorporate inference technology: learning user's behavior by monitoring where they go on the site and/or what they search for. An advantage of this approach is that users don't have to do any work to see a personalized view of the site.

On e-commerce sites, where a wide product offering is present, personalization can yield quite immediate returns by guiding shoppers toward that next purchase. You can present unique value propositions based on what you know about the customer and suitable incentives for him or her to provide still more profiling data to you.

The idea of personalization for pure information delivery is less appealing – although a case can be made if there is a tremendous amount of data on a site where the typical visitor is only interested in a limited subset. One example is the IBM tech support site, where users can identify the IBM equipment they own; upon subsequent visits, only material related to that equipment is shown, even in search results.

The development of a personalization system goes beyond designing the database and code to feed dynamic, personalized content to users. In addition, in many cases you will need to have a system for entering custom user information in the system or to override automatic processes. You will probably want user- and group-based reporting that typical web log analysis tools will not be able to provide. The effort to integrate and customize these administrative and reporting tools should not be underestimated.

A simpler – and generally less resource-intensive – approach is "customization," identifying distinct market segments and creating customized versions of your site for them. You may even want to consider "baking" those versions to yield the double-benefit of customization and performance. It's not 1-to-1 communication, but 1-to-many, which is still better than 1-to-all.

Above all, based on our experience by putting the CMS Knowledge Portal into practice, you must be sure to get input from all responsible parties, including editorial and technical staff before scoping out a personalization system.

Author's Checklist for Personalizing

1. Is there a pressing user-focused need that personalization could solve?

You should sense a palpable market need before proceeding with a complex personalization effort— you shouldn't just do it because you can or because your application server or CMS software supports it. If your answers are all focused on internal needs, then your investment may not provide a good return.

2. Do you require customization or true personalization?

Customization focuses on affinity groups, while personalization works at the individual user level. In many instances, limited customization can save time and money in developing a targeted information system. It is also a good way-station on the path to more advanced personalization.

3. What is the ROI you are expecting from a personalization effort?

Would your online sales benefit from a one-to-one marketing approach? Would you be able to increase customer satisfaction? Note that there are important reporting and privacy management issues to consider that may add overhead.

4. What personalization models will you employ?

As noted above, there are a variety of models, and they each have structural consequences. Do you have the database infrastructure to support a filtering or rules-based approach? More critically, which model or models do your target WCM packages support? A package that offers "personalization," but strictly on a preferences basis, cannot be expected to collaboratively filter your visitors.

(Chart 5.19) Author's Checklist for "Collaborative Function Model's" System Attribute: "Personalizing"

5.2.13 Privileging: The Access of Your CMS

Whether you manage an Intranet, Extranet, or public site, you may want to restrict access to certain content areas to authorized external visitors. See how your favored CMS packages approach this; some have built-in mechanisms for segmenting and authenticating external users.

Many CMS packages treat this as an extension of roles that internal actors play in the system, but it is really rather different. Unless they are actually contributing content themselves, consumers of information are likely to interact with your site very differently than internal authors and editors.

For example, external visitors have special usability needs with respect to authenticating, obtaining lost passwords, and understanding explicitly what areas may be off-limits to them.

Traditional access control lists for authenticating external users don’t always scale very well. But as with internal Role Management, many companies have found that LDAP provides a useful, universal infrastructure for managing access for partners and preferred visitors. Those firms will want to seek out LDAP-compliant CMS packages.

You can use the default authentication system built into your web server (such as working with “ht-access” files in Apache) to control logins. This presents a very simple and convenient approach, but often means working with crude interfaces, and that can bring management headaches at large numbers of users. Web server authentication systems typically do not offer more advanced security features, like preventing simultaneous logins with same credentials.

Author’s Checklist for Privileging

1. How granular will the permissions need to be?

You may have only one set of external users to authenticate, all with the same permissions. Or alternatively, you may need to account for multiple levels of users to multiple private content areas, or with different views to the same pages and content types depending on who they are. Make sure that your privileges system meshes with your approach to content management. If the latter entails burning static pages for publishing to a web server, then you will need page- or directory-based permissions.

2. Will you be selling online content or hosting a subscription service?

If so, then you may need a heavier-duty privileges-management system. Some DRM packages can manage complex permissions systems for subscribers. Of course, managing logins is not the same as managing subscriptions (renewals, grace periods, notices, payment, and so forth); you’ll probably need separate software for that.

(Chart 5.20) Author’s Checklist for “Collaborative Function Model’s” System Attribute: “Privileging”

5.2.14 Caching: The Speed of Your CMS

Caching describes a family of approaches to speeding up page delivery by keeping certain information “cached” in the server’s memory or some other readily-accessible repository (like the file system or even the database).

Caching is not just for busy portal sites. Since CMS packages tend to be surprisingly voracious resource hogs, caching has become a critical performance consideration even for low-traffic sites that employ a CMS platform. This is yet another reason why many CMS packages have turned over publishing to application servers that employ, among other things, state-of-the-art caching systems.

In considering a CMS package, find out what rules govern any caching it may do, in particular the always by zantine art of cache-invalidation rules. You'll want to be able to tinker with the settings, depending on how dynamic your content is, and what parts of your site you expect to be heaviest hit. In general, caching can get very complicated very quickly, and you ought to consider you're your CMS supplier may not have a core competency here.

Author's Checklist for Caching

1. How dynamic is your content?

There is a direct trade-off here between performance and freshness. If you seek to achieve maximum caching, it may result in the system serving content that has already been updated. Alternatively, if you always want to serve the latest content on an up-to-the-minute site, the page generation process will experience greater overhead looking for the latest version. Most site owners need something in-between.

2. What will you likely need to cache?

There are various levels that a caching system can operate: on files, web pages, objects, or elements. If the CMS uses file-based caching but you store and update your content as discrete elements outside the file system (e.g. in a database), you are not going to see the same performance improvements. The extent to which you are engaging in personalization comes into play here, since you may want to cache certain common page elements (such as images) to compensate for the extra processing involved in assembling a personalized page.

3. Do you anticipate needing to load-balance? At what level?

Load-balancing across two or more web servers in front of your content repository offers a simple solution, but does not protect you from network problems to the datacenter or a critical failure in your database. A more robust solution puts your site at multiple datacenters, but different WCM solutions will work within this topology in different ways.

4. How much hardware do you anticipate involving in the final architecture?

One of the reasons you may replicate in the first place is because CMS packages are hardware intensive. But keep in mind that most enterprise packages are charged by the CPU.

(Chart 5.21) Author's Checklist for "Collaborative Function Model's" System Attribute: "Caching"

5.2.15 Syndication: The Sharing of Your CMS

The case for syndication is simple: you can obtain more value from content by making it accessible in more places. But think quality in addition to quantity when considering the advantages of additional venues for your content. For example, if most consumers of your product enjoy closer relationships with your distributors, then you can make everyone's lives easier by syndicating content to your distributors' web-sites, where your materials can be meshed with theirs in a more compelling package.

This example prompts us to address the difference between traditional media syndication and B2B syndication. Many people think of syndication solely in terms of media or publishing companies – that is, when content *is* product. Syndication may be even more powerful when content serves as collateral to the product or service. If content is a key differentiator for your firm, then the ability to get it out to prospects and customers wherever they may be found could be critical to maintaining your unique niche and identity in the online marketplace.

The actual mechanics of content syndication are fairly easy, especially if you already store content in XML or have the means to transform your data into that format. Syndication standards remain somewhat gelatinous right now, but all of them – including the commonly used ICE and RSS – are XML-based.
Although designed for feeding news headlines RSS is well-suited to any circumstance where you require a lightweight format and want to drive visitors back to your web-site. The feed contains a series of news items, each with an article headline, short blurb, and hyperlink back to the complete document that resides at the syndicator's web-site. Like most syndication formats, you store the article on your web server, where it can be retrieved, parsed, and transformed into suitable HTML by subscribing sites.

Fortunately, many CMS packages provide some basic syndication tools, if only the means to generate XML and HTML files for subscribers to pick up. Other packages facilitate integration with 3rd-party syndication infrastructure products or provide hooks into commercial syndication services.

Note that commercial syndication services and products provide a key value-add: subscriber management. To the extent you attract multiple subscribers, you are creating an additional channel that requires ongoing management (sign-up, renewal, cancellation, technical support). In short, you'll want consider the syndication function within a broader syndication system. RSS is a nice syndication mechanism, but it won't manage your subscribers.

Author's Checklist for Syndication

1. **Will you be sharing content with partners?**

   If so, syndication standards will make everyone's life easier. Even within closed ecosystems, syndication is an efficient and simple way to share content. In the absence of true inter-enterprise CMS functionality among the vendors reviewed in the report, in fact, syndication may be the only stable way to collaboratively manage content.

2. **What is the level of sophistication of the target sites?**

   Unless they are sites like industry Net Markets that are used to receiving diverse feeds, you can assume that their ability to do things like parse XML may be fairly low. In many syndication services report which we have surveyed[52], that their surveys show only 10-20% of their subscribers can do so today, but this number is growing

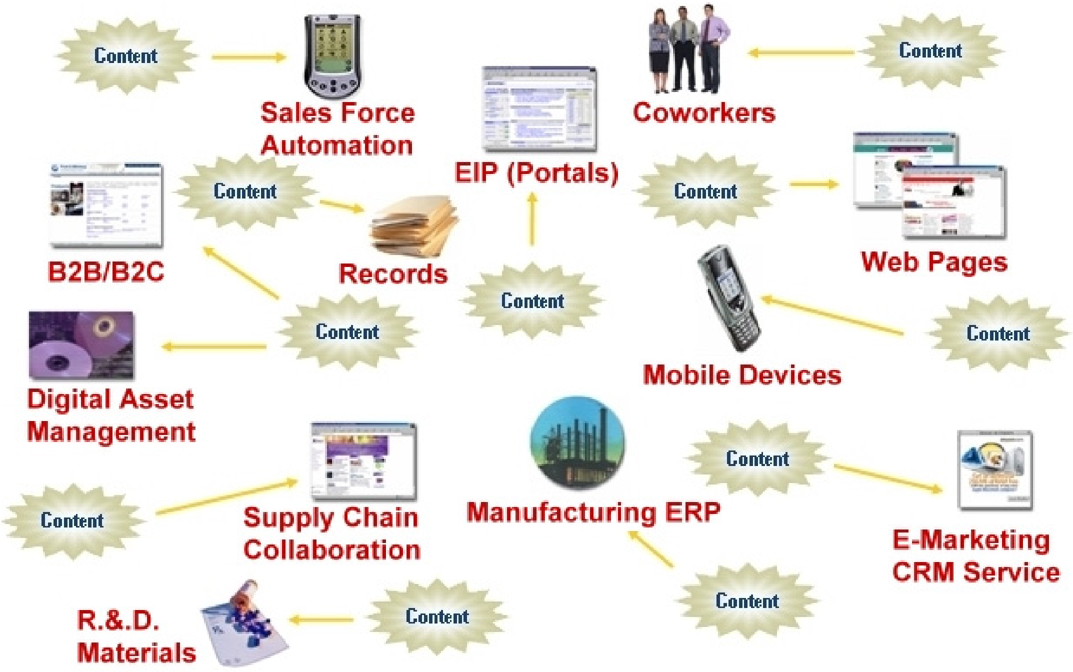
(Chart 5.22) Author's Checklist for "Collaborative Function Model's" System Attribute: "Syndication"

[52] In the research of Kerer C, he also points out see trend. Please see: Kerer C; Kirda E; Kurmanowytsch R: A generic content-management tool for Web databases, IEEE INTERNET COMPUTING 2002, Vol 6, Iss 4, pp 38-42, IEEE COMPUTER SOC

5.2.16 Cross Media Publishing: The Variety-Show of Your CMS

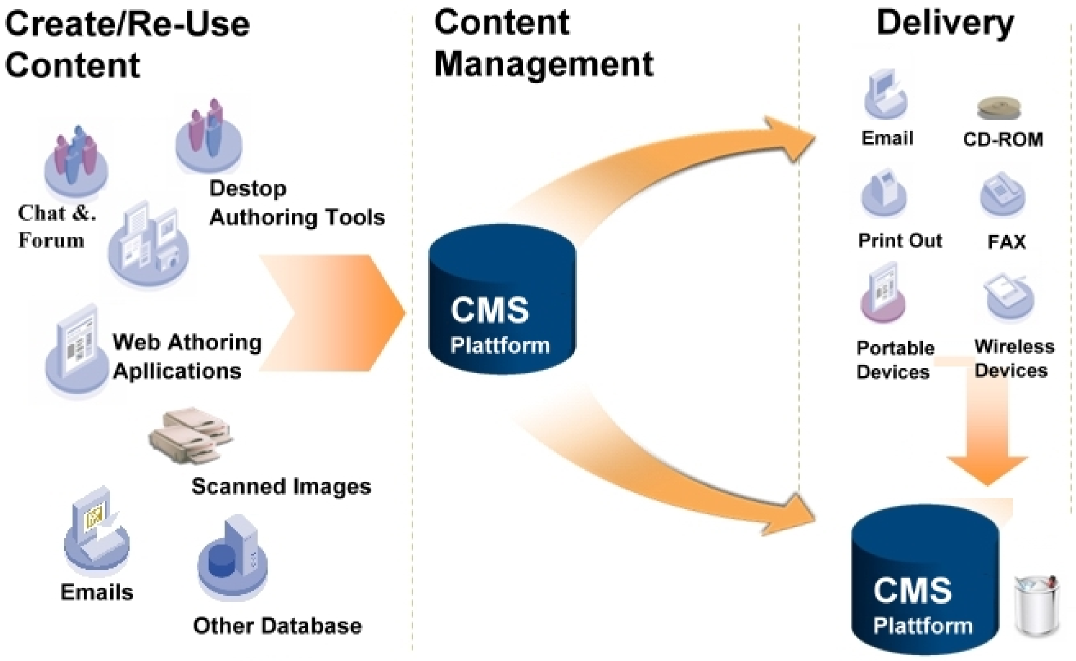
Whereas we addressed "Integration" of disparate content in the Production Dimension of the "Collaborative Function Model", you could consider this feature as "*Dis*-aggregation" or a more fancy and buzz word "Cross Media Publishing", that is, to break apart your web content for different delivery mediums beyond the web browser. These include wireless devices, e-mail, print, and CD-ROM.

The idea of "Cross Media Publishing" or let's say, the vision of "Cross Media Publishing" is to make the multiple uses of one single content/document possible, on the CMS platform. This idea comes originally from the developers for the Media Asset Management, MAM and it has both economic and business strategy meanings. The developers believe that, only in this way can the modern media (to some degree as "quassi enterprise") survive in this digitalized era and take full advantages of their "media assets", or in another word, to create the "synergy".

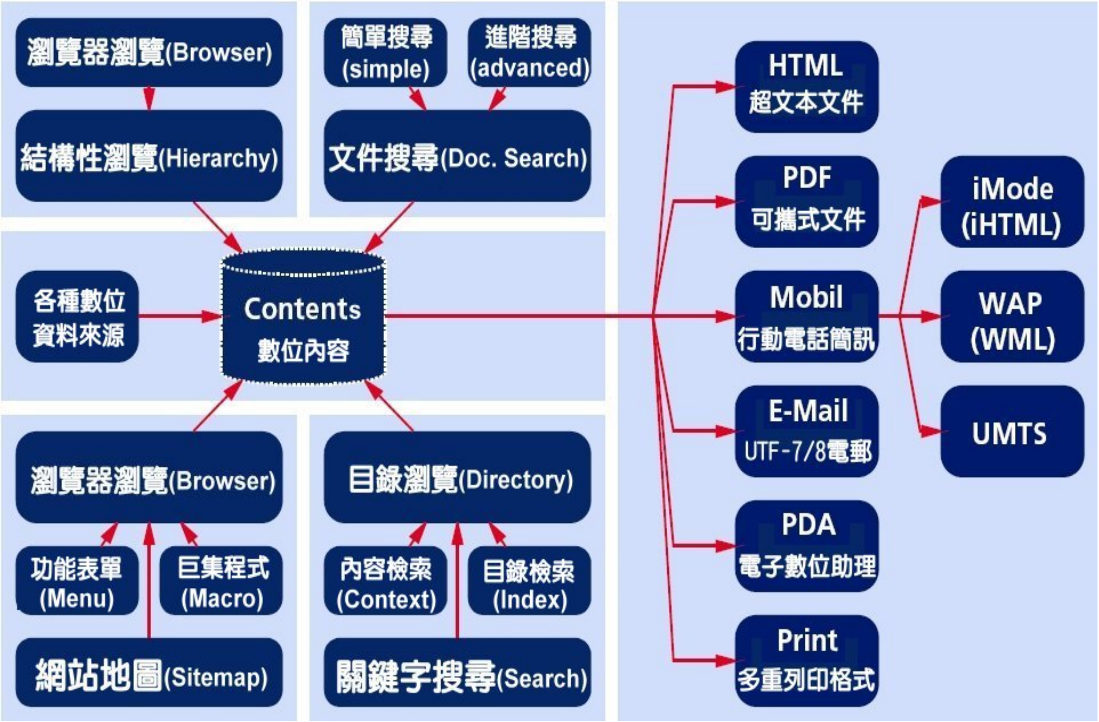


(Chart 5.23) The As-Is Situation of Content Delivery, BP of Documentum. You may find it in: http://www.documentum.com

As you can see in the chart which we illustrate **(CHART 5.23 / 5.24 / 5.25)**: Chart 5.23 shows the AS-IS situations of the content publishing/delivery situations, which could be easily observed in most of the companies today **(CHART 5.23)**. Through the cross media publishing based on the CMS platform, the American CMS vendor Documentum (since 2004, EMC Documentum) illustrates then in chart 5.24 the possibilities of content multi-delivery and publishing **(CHART 5.24)**. As far as how CMS achieve this "cross media publishing is concerned, in chart 5.25, our CMS partner for the international cooperation project, the Infopark AG in Germany, offers an overview on the software language structure of its CMS solution NPS5.5. In this chart **(CHART 5.25)**:, you may clearly identify the required software platforms, also from the programming perspectives, for cross media publishing based on CMS.



(Chart 5.24) The Vision of Cross Media based on CMS, BP of Documentum. You may find it in: http://www.documentum.com



(Chart 5.25) The Indexing Approaches and Delivery Platform of NPS5.5, both in Traditional Chinese and English
This chart is "exclusive authorized" by the Infopark AG to privileged academic uses. Please see the appendix for more Info.

However, the most important consideration with respects to cross media publishing is that technology will likely be the least of your challenges. Just like what has been showed in chart

5.25 above, techniques for exporting content to wireless formats, email broadcasts, or for print layouts is well-understood and fairly transparent, especially if your content resides in XML. The more difficult problems revolve around developing and maintaining a content model that will support repurposing information in logical chunks, while being able to evolve flexible processes for dealing with different media. Based on our surveys, "wireless" and "print" are the most important factors that must be carefully considered and evaluated in advance:

a.) Wireless

Wireless might seem the most complicated of non-HTML formats, but in some ways may be the easiest to execute technically. The most common presentation standard, Wireless Markup Language (WML), is simple to master, and XML-to-WML conversion in particular is quite straightforward. Consider the screens below1. The source code for the WML "deck" resembles a simple well-formed XML document (which in fact it is).

Regardless of the technical ease of development, some WCM packages helpfully include WYSIWYG tools for non-technical staff to control the conversion to WML.

Although the initial enthusiasm for wireless applications in the U.S. has ebbed, it is important to note that in Europe and Asia adoption rates continue to grow as new technologies rapidly come online, and therefore the ability to seamlessly translate content into suitable wireless formats becomes more a necessity than luxury for anyone publishing in those regions.

Japan uses a technology called "i-mode", which is similar, albeit more powerful, than the Wireless Access Protocol ("WAP") used in North America. Unlike European and US content, which is written in WML, Japanese content is written in another markup language called cHTML.

When you consider the following best practices, it will become clearer that the structure of wireless card decks are really quite different than HTML pages:

- In Europe and North America, wireless lives a low-bandwidth environment, and the standard mobile phone displays 96x64 pixels. Design accordingly.
- Your standard "pages" should be broken across multiple cards, with each card short enough to be read quickly.
- Because text entry is currently tedious on a phone keypad, successful sites allow users to select options from menus rather than typing long strings of text.
- You need to label all buttons, with short, clear words that instantly indicate what will happen when they are pressed.
- The back button is not automatically available and must (if desired) be coded as a soft button by the developer.
- Your site should be menu driven, rather than text-entry driven.

What has been pointed out above is all the more reason to carefully segment your content into wireless-friendly chunks, and carefully consider the different information architectures required of the different media formats. The technical transformation from WML to XML or cHTML may be trivial; the planning beforehand is not.

Integration with e-mail systems is increasingly popular. Some CMS packages offer shortcuts for selecting and formatting content for e-mail, but if you intend to do mass mailings, we recommend finding specialized software or ASP services designed specially to do that. Sending broadcast email can be hard on your servers, your network, and other IT systems. Not surprisingly, the hosted CMS vendors offer comparatively advanced capabilities for integrating CMS repositories and mass e-mail campaigns.

b.) Print

Integrating print publishing is more complicated. Native web content tends most often to find its way back into print via PDF files, perhaps because the tools to do this are fairly mature. In contrast, converting from XML to word processing files is still a clunky process.

One possible solution is to address this by completing the "single-source" loop with a unified authoring environment – almost surely XML-based – that will publish successfully to all your digital and print venues.

However, this is tougher than it sounds, partly because the print world is not completely XML-enabled (let alone completely digitized), and partly because it is often very difficult to "roundtrip" back changes that are made in the pre-press environment back into the original repository, to maintain that sole source of the truth. In short, there is no right way to go from Web to print; it depends on which alternative is most efficient for your people and processes.

In the end, the ability to publish to CD-ROM used to be a fairly common requirement, but today is an afterthought for most Web publishers, and software vendors have taken the hint. Any package that offers the prospect of publishing site "editions" can of course write static pages to directory that can be burned to a CD. For packaged multimedia presentations, CMS vendors may point you to Macromedia Director and the like.

Author's Checklist for Cross Media Publishing/Delivery

1. What wireless formats will you need to support?

There are a small handful of rendering standards for handheld devices, depending on the platform. Besides the widely-accepted WML, these include "Web Clipping" for the Palm line, cHTML for the Asian market, HDML for other devices, and more.

2. Will you be generating print materials from web-originated content?

If so, you'll probably want an XML version of your web documents that you can more easily import into desktop publishing systems. But first ask yourself: could the printout from a browser version of a printer-friendly page suffice? For some purposes, it might.

(Chart 5.26) Author's Checklist for "Collaborative Function Model's" System Attribute: "Cross Media Publishing/Delivery"