

Content Management
20/20 Foresight Through 20/20 Hindsight

by Doug Gorman

Ignore history at your own peril.

Those who ignore the past are condemned to repeat it.

If I have seen farther than others it is because I have stood on the shoulders of giants.

How does this help us get content management systems to work? Because currently they don't, you know!

According to recent surveys by Forrester Research, fewer than half, perhaps fewer than 20% of content management systems are achieving the kinds of ROI's that were planned for them.

I am (and astute analysts at Forrester, Gartner, Giga, etc. may be) 100% certain that most of the problems with these systems lie, not in the software functionality, but rather with the fundamental structure and architecture of the content (information) itself.

Learning From My Past

I admit it. I may be older than you. I remember Ronald Reagan in the 80's and Richard Nixon in the 70's and, well lets just stop there.

In my first job after college, I programmed a small application for the Federal Reserve Bank of Boston. It wasn't much, by today's standards, but, like all programs, it had a file structure. The file structure consisted of big long files where the program would have to access a piece of a file called a "record."

At the time, IBM mainframe computers handled most of the big software applications. The files were big long "flat" files that got accessed by VSAM (Virtual Sequential Access Method), ISAM, and these file structures were used by most large organizations including USAM (that's Uncle Sam.)

Large organizations were ready for a technological revolution. It took forever to create those flat files. Even worse, if there were 10 programs that accessed the "Last Name" part of a record, that record was repeated 10 times, perhaps in slightly different ways. And if a last name changed, 10 files would need to be updated from 10 different programs and maybe some people who controlled the programs would not know that the name had changed. So, you can see that we really had a mess on our hands.

What if there were a way to....store data once and access it from many programs. One would only have to create a single record. And, when a name changed, one would only

have to change it once. That was essentially the idea behind Database Management Systems (DBMS's).

Along the way, the following learnings happened.....

We're not in Kansas any more

At first, people bought DBMS's like IMS, IDMS, and Model 204 and they did the easy thing. They loaded their flat files into the database structure that could be a network structure, hierarchical structure, relational structure or inverted list. But here's the beginning of the learning – they didn't change the data architecture of the old flat files and the CPU's worked very, very slowly.

“See,” the pundits cried, “these DBMS's don't work!”

The learning was that to be most efficient, one had to separate the physical storage of the data in the database from the logical view (accessed by programs) of the database. One needed architecture for data and people like James Martin made a bundle of money helping organizations do just that.

What's in a name? A rose by any other name would smell as sweet.

So programs started groping around and they found the database file name. Oops – I only wanted the last name. This one contains the whole name... but this one is my “mother's maiden name.” Do you see the problem? When multiple programs needed to access the name, the programmer had to know what “name,” meant.

Shakespeare was wrong – a rose by any other name in a database does not smell as sweet. A rose is a rose is a rose.

Enter Data Dictionaries.

Organizations found that they needed better descriptions of what was in the database, so data dictionaries were created to describe the data architecture. The idea of metadata or “data about data” would let a programmer know that a field was “Last Name”, and it was 14 characters long (important back then). In the scheme of things, metadata was just as important to the application developer as the data itself.

Organizations had to create many standards to help with the creation, access and maintenance of DBMS's. This involved leadership, management, focus, and money. The money was allocated to re-writing the applications that would have the best payback, and this is important, most new applications were subsequently designed from scratch using the DBMS's and new data architecture models.

Fast Forward to 2003

Today, data and data architectures are everywhere -- been there, done that. Organizations are now trying to put their content (manuals, reports, presentations, pictures, spreadsheets, etc.) into content management systems. Most of these content objects look a lot like “flat” files. For example, the policy and procedure manual for a 6-sigma quality program might be 300 pages long. But most users don’t want to access 300 pages. They only want the page or two that they need to do what they have to do. And, by the way, much of the information in that manual is used in other manuals and, wouldn’t it be nice if we could “single source” the use of the information?

How do we fix the problem? Maybe we could learn from the past experience with DBMS’s.

1. People want to access specific pieces of information in the content management system, not a wall of words.
2. Organizations will need the equivalent of data dictionaries and useful metadata to help readers figure out what they want to use and re-use in the knowledge base.
3. We need information developers (writers) to be concerned with the architecture of content, itself.
4. Organizations are going to need standards governing the creation, access and maintenance of content
5. We should start by re-structuring the information in organizational hot-spots where huge costs are being incurred.
6. Organizations should develop all new information using a structure compatible with content management systems.

The Pill you don’t want to Swallow (but must)

The bottom line is that people in your organization are going to need to write differently. Right now they write page after page of “gobbledy gook” with little purpose or precision. These people, lets call them “writers” are going to have to structure their information so that users, lets call them “readers” can actually find the information they need to make decisions and take effective action. Furthermore, they need to structure information so that useful metadata can be attached. Some call this “structured writing,” We have a methodology called Information Mapping™.

Using an easily learned, replicable standard or methodology, writers can learn to

- Develop information with the reader’s needs in mind
- Separate types of information based on the purpose for the reader
- Assemble large knowledge bases so that creation, access and maintenance is simplified, and
- Develop information for paper-based and web-based delivery.

Some call this “structured writing,” We have a methodology for doing this called Information Mapping™.

Paying the Piper

The bad news is that you didn't budget for this. The MIS guys who developed your homegrown content management system didn't think about this problem. Your friendly content management system vendor might not have known it, but more likely "forgot" to mention that information architecture was critical attaining the promised ROI. To get your content management system to work, you are going to need to change the way that content is developed. You need a robust, replicable content architecture. You need to take your medicine, which is a pill, but to you it looks like it's the size of an elephant.

How Do You Eat an Elephant?

Ask your kids. Or any kids. They will tell you that you eat an elephant one bite at a time. In a moment of trepidation at a business meeting in Japan, I was presented with a plate of sushi (bait to us Cape Codders). A wise associate told me that you can eat almost anything if you "cut it small, eat it quickly, and pretend its chicken."

Here's how to cut your content management problem into small pieces for quick consumption. In the context of your organization's strategic plan and key operating initiatives you need to

1. Examine your major processes to determine the extent to which information (content) positively or negatively impacts that process.
2. Pick the processes where information has the most negative impact and fix that information.
3. Train your developers of the most critical information to create labeled, reusable content. Develop all your important paper-based and web-based content this way.
4. Repeat until your plate is clean

.....and call me in the morning. You'll be feeling better in no time.