TDQM Versus the Edifice Complex in Data Warehousing

MAX C. HANSEN
MAX HANSEN & ASSOCIATES

1. Introduction

Data warehousing, once practiced only by a few leading-edge corporations, is fast entering the mainstream. Since this is a fairly recent development, there are a few notable success stories, and a larger number of partial or complete failures. The owners of the failures may be expected to be less talkative than the owners of the successes, but we don't need to dig too deeply to uncover both kinds of stories.

Data warehousing is a term that actually covers many activities, processes, tools and technologies. Several of these may be new to a firm, so that there may be many possible reasons for difficulty; an organization trying to master several new technologies at once may well be expected not to master all of them perfectly.

This paper discusses some possible reasons for failure and success in achieving the goals of data warehousing. It does so, not by discussing each of the many tools and technologies involved, but by taking a very high-level conceptual view of the whole undertaking, trying to arrive at a way of thinking about the problem that will allow us to not get lost in the technological details. It is concerned not with implementation or design, but with a few principles which, if allowed to govern the architecture and planning of a project, should greatly increase its chances of success.

We begin by understanding that "warehousing," like so many terms we use in discussing information technology, is a metaphor. Metaphors are powerful but blunt-edged tools — we use them because they convey a lot of meaning, but we cannot always be sure just what that meaning is. Understanding this, the paper tries to expand our understanding of the task at hand by looking closely at what we can learn from the warehouse metaphor, as well as ways in which the metaphor may mislead us.

These thought experiments lead us toward a focus not on the warehouse, but on its product, information. From there it is a short step to where we focus on quality information products and processes.

This conclusion will already have been reached by many of the paper's readers. The paper is not likely to teach those much about data quality that they don't already know. By starting from another point, though, and arriving at quality through a philosophical exploration of the data warehousing problem, it may help them to think about how to carry on the work of evangelism. This is needed because data quality practitioners too often face the problem that they are called in after a Data

Warehouse project has fallen on the rocks. The budget is already spent, and now the client reluctantly deals out a few dollars for work on what should have come first: data quality.

2. The Search for Smart Metaphors

The building metaphor has outlived its usefulness.

Frederick P. Brooks, Jr. [Brooks p 201]

It seems inescapable that we use metaphors in order to discuss the world of information technology. It is a world entirely new to us, and in order to understand and convey to others the concepts we work with, we relate them, as far as we can, to concepts that are familiar.

Our metaphors give endless grist to the mills of serious thinkers, as well as the droves of curmudgeons who write for the computer press. To take a single example, each of several dozen columnists have met at least one of their deadlines by pointing out their own good reasons why "Superhighway" is a poor metaphor to describe the future toward which the Internet is leading us. Now and then an even more radical voice pipes up urging us to abandon metaphors altogether.

Metaphors truly are a handy method for conveying a good deal of meaning in a very few words. But they have their limitations. One is that, precisely because they convey so much meaning, we cannot be sure what meaning is being conveyed to a particular hearer. I may talk about the Information Superhighway to an audience of three, for example, and each of them will hear something different, depending on their own experiences of highways. One of them may be an overland truck driver who loves his work, who drives from San Francisco to New York in three days and back in two, several times a year, and always looks forward to the next trip. The second hearer may nourish fond dreams of loafing down historic Route 66, soaking up the nostalgia and delighting in the many hues of local color. The third may think of a superhighway as a scene of fiery crashes, a bedlam where lunatics habitually endanger themselves and others by speeding and weaving, a place to venture onto only in the direst necessity. Each of them will obviously hear something different in my metaphor.

Another limitation of is that, when a single metaphor gains dominance, it comes to be viewed as more true and complete than it can possibly be. It narrows our thinking, closing our minds to other ways of viewing the subject at hand. This is the primary reason, and a good one, for those pundits asking us to do without a particular metaphor, or even metaphors in general.

This is precisely the case with data warehousing. It is a universally accepted metaphor, so well-accepted, in fact, that it may take a few moments' thought for us to notice that it is a metaphor at all. Of course it is; a data warehouse and a tire warehouse have many features *not* in common. But because metaphors have strengths as well as limitations, rather than abandoning the warehouse metaphor outright, why not see what we can learn by making ourselves aware of its meaning?

3. The Edifice Complex

In the past few months I have read five books on data warehousing. Some are better than others. I wished that any of them had been available to me in the late 1980's, when I was feeling my way through dozens of data deployment projects. Of all my recent reading, though, the book that gave me the greatest insight into data warehousing was Stewart Brand's iconoclastic book about buildings.

How Buildings Learn, subtitled *What Happens After They're Built,* is based on the premise that what is most interesting and valuable to study about buildings is their histories in use — occupied, worked in, played in, and adapted to their occupants' needs. Brand indicts the architecture profession for largely ignoring this vast and fascinating area. All too commonly, he says, an architect's relationship to his or her creation ends when, immediately after the exterior is finished, a photographer is summoned to take the pictures that will be sent to the juries that dispense architectural awards. Although the profession has methods for performing Post-Occupancy Evaluations, Brand is disappointed that so few architects use them. Too often, the focus is not on the building as dwelling or as tool, but as lifeless edifice; if the viewer is awed when the plaster has dried, the architect feels the project was a success.

There are many unfortunate results of this point of view. It tends to cause a myopia not only towards user needs, but also toward fitting the building to its infrastructure and site. It causes its proponents to borrow design elements which fulfilled very real functions in some other context, but which cannot possibly fulfill them in this one. By turning a blind eye to the fact of adaptation, it creates buildings hard to adapt. It causes highly inflated fees to be paid for impressive but partial products. And so on — Brand builds his case against this syndrome broadly.

He states its counterpoint succinctly: "A building isn't something you finish. It's something you start." He is not so succinct in naming the syndrome he decries, though, and so I have taken the liberty of dubbing it "The Edifice Complex." I felt it was worth giving a short and memorable name to this syndrome because I find it to be useful in explaining why data warehousing is so difficult for many organizations to bring off successfully. If the edifice complex is inappropriate and destructive in creating buildings, how much more so might it be in a discipline in which warehouses are built only metaphorically?

The craft of data warehousing has several common dangers into which practitioners fall despite repeated warnings. Among these dangers are:

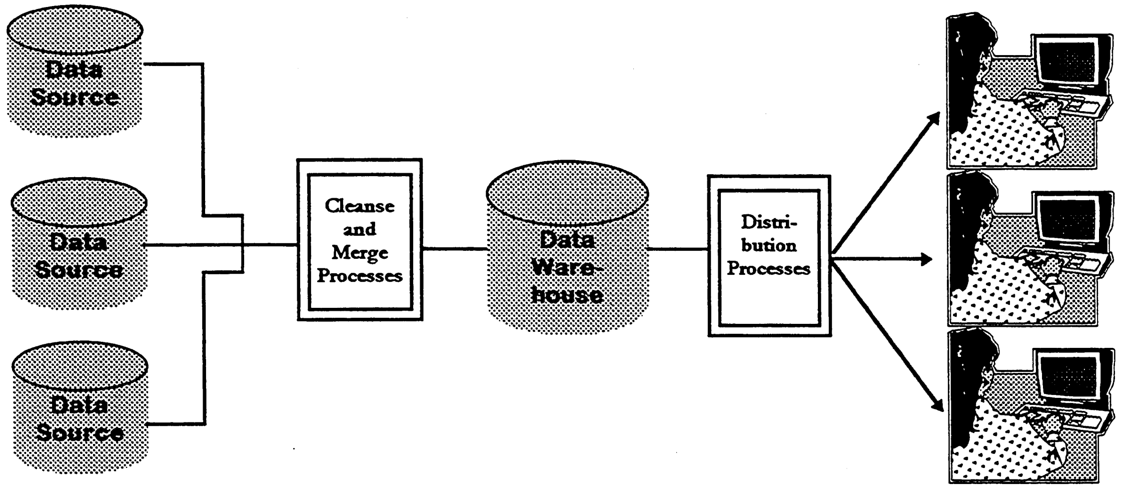
- Underestimating the time required to locate, extract, cleanse and rationalize data to be placed in the warehouse.
- Overestimating the ease with which end users will become satisfied and frequent users.
- Underestimating the resources required for maintenance.

- Overestimating the completeness of requirements as gathered by pre-design needs analysis.

Might it be that these pitfalls result from the edifice complex? It might indeed.

4. The Data Warehouse in Context

A number of the books on data warehousing agree in surrounding the data warehouse in a good deal of context. There are several variations on the plan shown in this drawing:



In this simple diagram, data moves from its sources into a data warehouse through a complex of extracting, cleaning and loading procedures. The data warehouse is where the data resides when it is fully ready for distribution to end users. The box on the right, labeled Distribution Processes, is also a complex of activities. These may include On-Line Analytic Processing (OLAP), placement of data into departmental data warehouses or data marts, data mining, and many other tasks. In this box, in fact, belongs everything that must be done to the warehoused data in order to turn it into information, which the user in turn assimilates into knowledge through the normal perceptive processes of viewing and reading.

The boxes on either side of the data warehouse are where the action is. The data warehouse itself is merely the resting place for the data between its arduous journey through collection processes, and its simpler, but still complex, delivery to users.

Yet when we talk about this entire effort, we call it data warehousing, as if the storage in the middle were the main event, the crowning achievement, the whole show.

It is not. In fact, it is precisely through excessive focus on the edifice in the middle, that we so often set ourselves up for failure in achieving the overall task.

Consider:

- At Larry Greenfield's web site, on the page titled "Data Warehousing Gotchas," Greenfield states "You are going to spend much time extracting, cleaning, and loading data...W.H. Inmon, in his data warehousing books, estimates that, on average, 80% of the time building a data warehouse will be spent on this type of work."
  Inmon is widely considered the foremost authority on data warehousing, in fact, the father of the practice. His *Building the Data Warehouse* may well be the best-selling book on the subject. Yet Greenfield believes that, despite the wide circulation of Inmon's books, this point is "not discussed enough in the literature." In other words, despite the widely published word of Inmon, underestimating this aspect of data warehousing work is a trap that practitioners are still all too likely to fall into. My experience confirms this.
- At the other end of our process picture, there is a great deal of agreement by leaders in the field that data warehousing project scope is likely to increase after the project is under way. This follows from the observation, about which the authors of the best books agree, that the normal SDLC (Systems Development Life Cycle) is inappropriate for data warehousing. There are several reasons for this, but foremost among them is that the users cannot know what they want from the data until they have had their first glimpses of what the data looks like. There are questions they can't even formulate until they have had their first round of questions answered. It is in answering these follow-on questions that the data deployment process delivers its greatest value. And yet, as several experts suggest, and my experience confirms, data warehouse planners are still all too likely to use the SDLC in its most inappropriate form, the classic waterfall model, a model which is focused not on the continuing meeting of evolving user needs, but on the unveiling of a finished product, a "rollout" all too closely akin to the architect's date with the photographer.

There are close parallels between the edifice complex in data warehousing and in building. The data warehouser's tendency to underestimate the tasks of preparing data parallels the edifice-obsessed architect's failure to fit the building to site, to infrastructure, and to available materials. The failure to account for changes in user needs parallels the architecture profession's widespread failure to perform Post-Occupancy Evaluations, and to build their buildings for change.

5. Cleaning up Our Thinking

Since accepting the warehouse metaphor exposes us to such dangers, perhaps we should now abandon it. Or then again, maybe we should continue to learn from the metaphor by learning about the architecture of buildings.

There is a third alternative: we can do both in turn.

If Brand is on the mark, then the problems which plague our data deployment projects may not be caused by the poor fit of the building metaphor, but rather, to the extent that the metaphor *is* valid, by the professional hazards of architecture which have real parallels in our work. So, we might first learn all we can by exploring the metaphor.

Then we can leave the word "warehouse" behind, and consider data deployment in the light of alternative metaphors.

5.1. Lessons From Architecture

The special beauty of Brand's book is that it goes far beyond naming the problem. *How Buildings Learn* is quite helpful in suggesting solutions. Not all of the remedies Brand suggests are tested notions, and some don't transfer to information technology as well as others. Yet there is enough good sense in his approach to the problem that it offers considerable insight into the problems of data warehousing. (For this exercise, we will continue to use the warehousing metaphor.)

Among Brand's suggestions are these:

- **Scenario Planning:** Use visionary scenario planning to imagine both likely and less likely futures for the building.
- **Slippage:** Understand that various layers of a building have their own characteristic rates of change. For example, a building's site is almost never changed, and its structure not often, while its space plan (defined by windows, doors, interior walls) changes several times in the life of the building, especially in and around kitchens and bathrooms. Knowing this, plan the building to maximize the slippage between these layers. Don't, for example, make every wall around the kitchen a load-bearing, structural wall, since this tying of structure to space plan will add trouble and expense to the remodeling that the kitchen is likely to undergo.
- **Starting Small:** Build small and plan to expand. This way the full extent of the building project will not be undertaken before the occupants' needs are understood.
- **Accessibility of problems:** Don't hide problems. Aluminum and vinyl siding both do this. They give the appearance of a low-maintenance building while underneath, the structure may rot quickly and unnoticed.
- **"The Book":** Follow the lead of an innovative builder in Massachusetts, who supplies a house's occupants with unusually rich documentation of how the building went

together. This includes extensive suites of photographs of the house after the services (plumbing and wiring and such) are in and before the walls are covered. The book of plans, photographs, and other documentation greatly ease maintenance and remodeling.

- **Post-Occupancy Evaluations:** Keep your eye on how occupants use the building and your ear open to how they wish they *could* use it. Adapt your design technique based on what you learn.
- **Continuing Contracts:** A corollary to "Starting Small", this idea is that the architect should plan on having a continuing relationship with the building and its owners, creating the first small structure, and then adding to it in response to user needs.

Some of these have obvious parallels to data warehousing projects. Others may transfer in less obvious but very valuable ways.

- **Scenario Planning:** This one is obvious: consider how the technology, the regulatory environment, even the nature of the firm's business, might change over various time horizons. Think of how the warehouse can be architected to accommodate the broadest range of outcomes.
- **Slippage:** Identify the layers of your data warehouse environment which may have different rates of change. To do this, try to grasp the history of change in your industry and your organization. Also, allow possible layers to emerge in your scenario planning. If identifying the layers is hard—and it may be—then bow to this difficulty by allowing loose fit and high modularity in as many areas as budget and complexity will allow. Consider the possible rates of technological change in various areas of the project. For example, on the distribution side of the warehouse, web-based, skinny-client tools are coming along quickly, but are not yet mature, robust, or full-featured. This suggests that your architecture should support but not mandate a skinny-client delivery system.
- **Starting Small:** Just as with a building, build small and plan to expand. Data marts are based on this idea. Be careful, though: as Inmon points out, one of the purposes of an architected data warehouse environment is to overcome the proliferation and anarchy of data sets, and data-marting, as practiced by many organizations, leaves this problem untouched or even exacerbates it.
- **Accessibility of problems:** As vinyl siding hides problems, so does eliminating all redundancy in data, and we may fall into this trap if we equate redundancy with anarchy. Quality problems may hide in a single set of data but may show up if multiple versions of the data could be rubbed together to show discrepancies. Therefore, the selecting of a "system of record" and eliminating all other sources of similar data should

be done with the greatest of care, making certain that other data quality measurements adequately replace the opportunity for comparison that will vanish.

- **"The Book":** The Massachusetts contractor's book equates to both project documentation and rich metadata. Most of Brand's arguments in favor of "the book" can easily be translated to support for the "faked rational design process" proposed by Parnas and Clements [Parnas & Clements]. And the book also gives valuable information about the quality and sources of the materials that went into the building. To the extent that source data are considered the raw material that make up the warehouse, this parallels a rich store of metadata kept in an accessible repository.
- **Post-Occupancy Evaluations:** Again, keep rich metadata. In this case, the metaphor suggests keeping metadata not only on the data sources, but on the uses to which they are put. Also, the needs that aren't being met are an important realm of metadata, guiding (we hope!) the future adaptation of the warehouse.
- **Continuing Contracts:** View the needed human capabilities as an essential part of the plan. This allows for a system that has an organic, growing relationship with its owners. Keeping this point of view will also open our eyes to a richer set of capabilities than are usually thought of, which are often limited to administrative capabilities and "training on the tools."

The last lesson may be the most powerful one we get from Brand. It is that the most important element in a learning architecture is a marrying of human capabilities to the building, or, in our case, to the data deployment system, whether we call it a warehouse or not.

5.2. Lessons From Other Metaphors

Having milked the warehouse metaphor for meaning, let us now see what we can learn by doing without it. First, we can view the issue through the lens of other metaphors. Remembering that our endeavor is to create a product called "information," let us apply three metaphors to this product and see what they tell us about the process:

Information is:

- A weapon.
- Lifeblood.
- Medicine.

5.2.1. INFORMATION AS COMPETITIVE WEAPON

For our first experimental metaphor, we view business as war, and information as a weapon which gives essential advantage in that war. Some implications of this way of thinking are:

- Those who take to this metaphor will probably be the first to adopt the term deployment, since this term is commonly used in military actions: soldiers and materiel are deployed to the various battlefronts.
- The central database, the part which had been known as a data warehouse, might then be called a "data armory." We thus have some of the good and bad results of a building metaphor, but calling it an armory is consistent with the broader picture; every general knows that an armory is a very small part of a very large apparatus. That apparatus is the whole war machine, and remembering this may call to mind a few facts about the weapons housed in the armory:
    - They are valuable only insofar as soldiers know how to use them.
    - Mission planners must know how to use the weapons as well, and in a larger sense than the soldiers, understanding which ordnance is appropriate for which target, which weaponry for which battlefront.
- Those responsible for deploying real weapons know that redundancy is a defense not only against losses, but also against the inaccuracy, imprecision and unreliability that are inherent in any weapon. These unfortunate qualities must be measured in order to determine the level of redundancy that will give reasonable assurance of a successful mission.
- Weapons are manufactured products, whose design and overall quality may make a great difference, as witness the showdown of SCUD vs. Patriot missiles in the Gulf War. That they must also be well maintained in the armory is true, but is less important than their original quality.

The weapon metaphor may yield even more important insights, but we've given ourselves a start, and can move on to another metaphor.

5.2.2. INFORMATION AS LIFEBLOOD

In this metaphor, the firm is viewed as an organism, and information keeps it going, as blood keeps the human body going. Some implications:

- The former warehouse is now a heart, through which the lifeblood must pass in order to get to where it is needed. In giving us a more active picture than that of simple storage, which is implied in the name "warehouse", this metaphor conveys more truth about our endeavor.
- Like the data armory, the heart gives a more balanced picture of the role of the central database. Yes, the heart is very important, but other organs have equally important functions. The lungs do the essential work of refreshing the blood with oxygen, and of filtering and discarding certain impurities. The large size of the lungs as compared with

the heart does some degree of justice to the complexity of the intake and cleansing functions on the left-hand side of our diagram. This, remember, is where Inmon says eighty percent of the work goes on.

- Both heart and lungs are subject to the monitoring and direction of a highly developed control system. And while the brain's control of the heart is normally autonomic and unconscious, the lungs are easily made subject to conscious control. The organism realizes important benefits from exercising this control, including, in humans, a sense of well-being as well as clear speech and pleasant song.

5.2.3. INFORMATION AS MEDICINE

In our final metaphor, we see organizations as subject to entropy, decay and disease, and information as medicine to guard against these ills. One beauty of this metaphor is that, more than that of lifeblood, it reflects the diversity of information needs within the organization. Just as different people in a city need different medicines, so do different regions, divisions, and managers within the firm need different kinds of information to maintain their health.

Other implications are:

- The data deployment process is akin to the work of a pharmaceutical company. An important feature of such firms is that R&D is far more important to them than manufacturing. This in turn may imply that the more important knowledge is that which doesn't yet exist. Awareness of this may guard us from the hubris of architects, who relish the day the building is done. We will know that the work of data deployment is never done, that at no time have we created a cure-all, and that the next health threat to our firm may call for new insight, not just *from* the data, but *about* the data.
- Drugs have side effects. So might the decisions we make based on our data. A piece of information may tell us how to cure the symptom that today is foremost in our awareness, but this may not be the whole story. Our use of the information may have a side effect worse than the disease. For example, in the convenience food store industry, data may show that filling the windows with large promotional signs increases not only sales of the promotional items, but overall volume and average margin. This may lead us to cover every inch of glass, unaware or forgetful that doing so is also known to increase the incidence of armed robbery, since the covered windows give robbers a reassuring sense of invisibility. If we are too focused on the whiz-bang insights given us by data mining, we may be myopic toward side effects.

6. TDQM as Antidote to the Edifice Complex

The metaphors have been helpful to us in exploring various ways of looking at the data deployment problem. We may now set them aside and talk about the matter non-metaphorically.

First, we may do well to be aware that the three new metaphors we discussed began not with the question, "What is a data warehouse like?" but rather, "What is information like?" This choice was in itself a rebellion against the edifice complex, and was the reason the metaphors were so helpful.

In talking about our real goal, we might realize that, in the eyes of the end user, the information our systems produce may fit one metaphor quite well. Some executives regard their work as very much like war, others as a struggle to maintain strength, or to ward off disease. If we look at all the metaphors together, they have both similarities and differences. To arrive at a clear, non-metaphoric view of what data deployment is all about, we need only see what all the metaphors have in common: in every case, the information is a product, and one which must be of high quality in order to be effective.

Thus, we are not engaging in metaphor at all if we say that our task is to manufacture a product, information, and our challenge is to produce that product as needed, in a timely way, and of high quality.

The first order of business is to decide which skills to develop or enhance.

In his critique of "No Silver Bullet," Capers Jones faulted Brooks for focusing on productivity. "Focus on quality," wrote Jones, "and productivity will follow" [Brooks p 217]. I agree, and so I recommend that some form of TDQM be put in place as the antidote to the edifice complex.

There are many notions brewing of what TDQM is or should be. I will not argue the details. Since the focus of this paper is on the edifice complex, though, I will state what features of TDQM must exist and mature in order to combat the disease.

First, TDQM must maintain as a central tenet that it focuses on creating and honing essential skills within the IS organization. This will allow TDQM activities to provide essential knowledge regarding the organization's readiness to undertake new data deployment efforts.

TDQM must maintain a focus on business needs, and continue to increase its repertoire of methods for cost-justifying itself. The researchers in the field have seen the need to overcome resistance to TDQM, resistance which arises from the common prejudice which says that achieving perfect data quality could be infinitely costly. In response they have focused strongly on cost-justification, and I hope they will, continue to do so, since this is itself one of the essential skills for data deployment.

Remember that one of the themes that emerged in our study of metaphors was the importance of metadata. This is one of the areas of data deployment most fraught with pitfalls, and many of these pitfalls have to do with setting priorities and cost-justifying. On the one hand, the desire for good metadata can grow into extremely ambitious repository schemes. In the course of a recent architectural study for what will eventually be a huge data deployment project, I saw a report from a consultant, a specialist in repositories, which stated flatly that the completion of an enterprise metadata architecture and an enterprise data model were prerequisites to *starting* the "data warehouse." Fortunately, another repository expert was called in for a second opinion, so that the project may hope to get underway within the century.

On the other hand, because setting limits on our ambition largely comes down to intelligent guesswork, which can be scary, we sometimes avoid it entirely by having no ambition at all. We could have responded to the over-ambitious aims of the first repository expert by starting the project with no repository at all, asking for no more metadata than would be embedded in the "warehouse" data structures, including triggers and query hints. This would have been just as disastrous as postponing the project forever in hopes of fitting the whole world into the repository.

TDQM should offer methods of evaluating just how close to perfection we should try to come. In some cases, this will not be close at all; we will have done enough just by gaining a clear picture of how reliable the data is. In other cases, our understanding of the analytical model in which the data will be used may tell us that the results may have very little robustness to poor accuracy. Then, if we know that the analysis is truly essential to the firm's mission, we can decide to expend great resources to get the data thoroughly cleansed.

7. Sources of Resistance

In exploring data deployment through both the old and the new metaphors, we have seen a few themes emerge.

1) Data deployment is a process, not an edifice. It is not a method of storing data, but a manufacturing process whose product is information.
2) Deployment is a process requiring not just up-front design expertise (although this *is* important), but a continuing application of human capability and judgment. This shows up especially in the weapon and medicine metaphors, in which it is clear that neither weapons nor drugs can provide value in the absence of skill and judgment.
3) Metadata is an essential tool for those who will maintain and continuously rebuild the system.

Although there are practitioners of data warehousing whose approaches acknowledge these three themes, there are too many instances where they are ignored. Among these ideas, the one that probably represents the most radical departure from common practice is the second, that of the continuing need for human judgment and capability. Yet it is supported by every one of our metaphors. Brand's insights into architecture support this idea. His ideas of starting small and of continuing contracts, for example, support this idea. So too does the notion that problems should not be hidden, since it presupposes that an intelligent monitor cares about and is looking for flaws.

This idea is not even absent from the experts who use the warehouse metaphor. Inmon, for example, favors incremental rather than "big-bang" development schedules, implying that analysis and design expertise are needed throughout the project, and that the project may in fact never end. The problem here is not that users of the metaphor are unaware of this issue, merely that the metaphor with its connotations of static completeness encourages us to forget the insight in practice.

Even where the metaphor is not at fault, there are forces that favor our forgetting it. In nearly all areas of information technology, many factors make it seem easier and more cost-effective to buy a tool rather than to build a capability. Among these are:

- Tool vendors have large stakes and large minimum scale, so they push their products avidly.
- Consultants may be able to increase a client firm's capabilities, but only the very highest fees are a good incentive to push a capability-enhancing practice.
- A widespread perception of consultants is that they exist to *steal* capability and create dependency. An old joke, for example, describes a consultant as a guy (this would read "person" if it weren't an old joke) who comes in to your office, borrows your watch, takes a large fee to tell you what time it is, and leaves with your watch. This is tempting to believe because it is grounded in truth.
- The computer press, such as for example Computerworld, tends to focus on large transactions and large companies. This means they write about large tool vendors and only the largest consulting firms, such as Andersen. A small consultancy offering seminars in statistical control of software processes is regarded as non-news, and so the wealth of what the press does print tends to drown out our awareness of the possibilities and benefits of capability-building.

Another insight into the neglect of organizational capabilities comes from a study I performed on the growth of the Software Engineering Institute's Capability Maturity Model (CMM). As I gathered anecdotal evidence as to why software development organizations resist the CMM, two themes emerged. First, the very notion that an organization undergoing a CMM assessment might be labeled "immature" is obviously threatening. The second reason is conveyed by the responses which are variants on a roll of the eyes and the statement, "Uh-oh, here come the bureaucrats."

If in the word "bureaucrats" we hear "those who stand in the way of progress, of creativity, of work," I believe we have understood this objection correctly. The objection itself, though, is wrong.

It is wrong first because a highly capable organization is one that is always monitoring its own health, like an athlete watching the scale, taking her pulse, adjusting her training to the goal as well as to her current condition. It is precisely this capability for self-monitoring and self-improvement that the

CMM tries to measure in a software organization. It is an essential skill, however threatening it may feel to be told that we don't possess it in the measure we would like.

The second reason the objection is wrong is that it results in decisions which are perverse in that they push us into the very thing we think we are avoiding by shunning "bureaucracy."

Consider: When the top management of our software organization brings in the Software Engineering Institute or some other consultancy to provide us with a CMM evaluation, and if we have a knee-jerk objection to them as "bureaucrats", then no matter what they actually say or do, what we are likely to imagine they mean is "Hello, we're from Pittsburgh and we know what's best for you."

Naturally, we don't want to hear this. What we would prefer, if we know that our productivity or quality could use some improving, is to go out and buy the latest silver bullet, usually a software tool that promises to turn us into stars. And what we get is a piece of software that embodies, in its design, many dozens of decisions, made by developers we do not know, *about what is best for us*.

It is only because we have learned to relate to software in a certain way that we fail to notice this. Of course, we ourselves are software developers, and consider this so thoroughly a part of the game that we scarcely notice it happens. Also, the difference between a person versus a program knowing what's best for us is that the program avoids implying that we ourselves don't know, while a person, due to the nature of consultancy, can hardly avoid implying this. But, if we fall into this trap, we do so by failing to appreciate that the person, unlike the program, has at least the potential ability to listen to and heed our counter-arguments.

The history of the CMM bears this out. Early versions of the CMM had flaws, and the SEI heard about these flaws from many sides. They then responded, by making the model more flexible, by discouraging punitive uses of it, and by creating an adjunct model to measure people-management skills, as suggested by Capers Jones and several others.

8. Reasons for Hope

This paper has proposed a radical shift in our thinking about data deployment efforts, asking that we think of human architectures as much as of technological architectures. I believe that in doing so, we will create systems that are more adaptable, more responsive to user needs, and more capable of delivering a product of high quality. Because there is known resistance to expending resources on both capability-building and on data quality, it might be worth asking why I see any hope for this approach being adopted.

One reason for hope is that the effort we are discussing, whether we call it "data warehousing" or "data deployment," is in any case a paradigm-busting activity. It requires data modeling techniques which are foreign to many system developers. It turns the normal Systems Development Life Cycle on its ear. It is normally not mission-critical, yet the results it promises can sometimes be urgently needed, and it can present great stresses on systems performance, not because it requires fast response times, but because of the volumes of data that must be processed and the patterns of this processing in time.

Where a technology breaks so many paradigms, the managers responsible for using it will eventually open their minds to rethinking it from the ground up, finding at last a more useful paradigm.

The second reason for hope, and also for tying the new paradigm to a quality focus, is that as we turn more data into more information, and as more management decisions are based on the products we produce, demand for higher data quality is going to come. Successful quality initiatives, in manufacturing as well as in data, succeed because of top management support. This has been lacking in the case of information quality, because, in the absence of good data deployment tools, top managers have not been exposed to the harm caused by poor data. As we deliver improved tools, these managers will increasingly find themselves downstream from the pollution, and will start offering active support for getting it cleaned up.

This doesn't tell us data quality practitioners, though, how to solve our worst problem, which is that we are generally called in to look at data quality when there has already been a disaster. For this, there are two sources of hope: 1) We can evangelize, and 2) IS professionals *do* learn from each other.
If the latter is true, then even if Joe spends his whole data warehouse budget before thinking about data quality, we can hope he'll steer Mary clear of that mistake. Mary will then, ideally, call us in even before she makes a budget. Sometimes, by a simple data audit, we will be able to show Mary where her budget will come from — it's in the money that leaks out of her firm through the holes in data quality.

References

Brooks, Frederick P., Jr. The 20th anniversary edition of *The Mythical Man-Month.* 1995. Addison-Wesley Publishing. (The essay "No Silver Bullet: Essence and Accident in Software Engineering", originally published elsewhere, forms chapter ten of this edition.)

Parnas, David Lorge and Paul C. Clements. "A Rational Design Process: How and Why to Fake It" In *Software State-of-the-Art.* Tom DeMarco and Timothy Lister. New York, 1990. Dorset House Publishing. Pages 346-357.