DAMAN
CONSULTING

Planning for a Data Warehouse

By Michael Haisten

Few publications on the data warehouse deal with the substance of the planning process. They almost make it seem like you can pick a subject area, make some technology decisions, hire a data modeler, and hit the ground running. I have worked with numerous data warehouse projects and I can say, without qualification, that the quality of the forethought and understanding of your development team has a far greater impact on success than anything else. It isn't the technology. Few of you are reading your first data warehouse article. I kept this firmly in mind from the outset. This article does not cover the basics of data warehouse 101 and the chaotic history of information management. What it does cover, instead, is an outline of the pivotal issues to consider before committing to action.

Clarify Your Intentions

Moral: *Rationale is the foundation of your plan and a determinant of your potential*.
So you plan to build a data warehouse. Why? The concept has become so popular that it is treated as a self-justifying proposition. As in the hit film *Field of Dreams*, normally sound-mined IS professionals firmly believe: *If we build it, they will come!*

One factor shared by all successful data warehouse efforts is pragmatic rationale. Articulation of the potential benefits is a minimum starting point. The ability to paint a rosy picture of future possibilities is a valuable skill. But, real success comes from effectively associating your warehouse proposal with the daily reality of the enterprise.

Don't confuse vision with rationale. Vision defines the path. Your rationale pays for the journey.
The most powerful justifications for a data warehouse investment are qualitative goals. Typical objectives culled from actual executive presentations are:

- Improving information access
- Bringing the user in touch with their data
- Enhancing the quality of decisions
- Providing cross-function integration

Statements like these help illuminate your vision. A common failing, though, is to present these statements as if they are, in fact, the rationale.

What does it mean to *improve information access*? How do we ensure this has the same meaning for everyone? Every goal should not simply point out a direction. Each should tell a consistent story. Each must carry specific deliverables and must, somehow and in some form, be measurable.

A data warehouse can rarely be justified based (solely, primarily) on operational cost reduction.
Where do you turn to add substance to these fuzzy statements of direction? The normal tendency of IS professionals is to revert to a tactical pitch involving cost reduction. *Reduce reporting costs by 30%* is typical of these cost-centered arguments. It sounds pretty specific, doesn't it, and it comes equipped with a number. What more could you ask for? Well, for one, it helps if it is based on even a shred of truth.

Let's follow this chain of logic: Give me some money now and I promise to eliminate a batch of operational reports at some time in the future. I will accomplish this by creating a pool of data and training the users to wade in on their own to siphon off what they need.

This is only an elaborate shell game which management will see through, sooner or later. In the best possible outcome, you simply transfer costs from the very visible IS budget into a user cost center. If lucky, you produce a slim IS-only return on

investment. In a more typical case, the saving from eliminating an operational expense does not even cover the development costs. And in any scenario, the business as a whole experiences a negative return.

Ironically, a data warehouse can return *real* benefits that go unnoticed when the promised objective is not achieved. Good work can not rise above bad rationale. Help your project build on a solid foundation.

The best-cost objective: Achieve a permanent reduction in the full lifecycle cost of delivering information.
A data warehouse solution can become a highly efficient process for responding to new information needs. Traditional means of extracting information from our data systems are highly labor intensive and, at best, linear in cost. Dealing with older legacy systems can result in exponential costs; it costs more and more every time you go back to the well. The objective of data warehouse implementation is to *initiate a data acquisition and delivery process that offers lower marginal cost with each new user over time.*

The best overall objective: Align your goals explicitly with a strategic business initiative.
Link your data warehouse to the strategic plan of your enterprise. Not the IS business plan - I am talking about the business business plan. I use the motto *AIM Higher* to remind me to search these plans for opportunities in three areas:

- *Analysis* – Capture 20% market share by increasing volume and reducing the price on high margin products.

- *Integration* – Decentralize by elevating the divisions to autonomous profit centers while retaining corporate-wide visibility of product movement and customer interactions.

- *Migration* – Re-engineer the order process to ship 90% of orders the day they were taken.

These strategic goals often suggest the right subject areas and business context to pursue. For instance, the analysis example implies a market subject area that commingles internal data on sales volume, pricing, and profitability with external data on market share. The integration example explicitly cites a product and customer subject area. Indirectly, it implies a financial data warehouse spanning the newly independent divisions.

Re-engineering, renewal, or re-whatever often involves significant application migration activity. Migration spawns many productive justifications for a data warehouse. One can be implemented in advance to shield information consumers from disruption due to the transition. A data warehouse can simultaneously serve a *forward conversion* role as well as its normal information access function. A data warehouse can also supplement information access and analysis deficiencies in new applications.

The worst (and frighteningly frequent) objective: a technical proof of concept for data warehousing.
This is the goal of way too many pilot projects. Suffice it to say data warehouse projects justified exclusively on technical grounds have the highest failure rate.

Create a Program

Moral: *Offer a bold, comprehensive, and rigorous program with a pragmatic bias.*
With a strong, credible rationale, you are off to a solid start. The next step is to frame what you intend to do.

Spread the vision – outline a multi-year, multi-phase plan, and propose a pilot supporting this direction.
When you articulate what the data warehouse will become, always position it as evolving incrementally to match the changing needs of the enterprise. In early releases, you will be perfecting techniques and building technical infrastructure. Make sure management realizes that each early release will tackle a new business subject area and will offer additional information deliverables. Then prepare to fulfill these promises.

Vision background; not just another database, not just a project, more than infrastructure.
A data warehouse is not just another database. Most database designs are derivative – intended to serve the processing requirements of a specific application. A data warehouse database is both primitive and primary – intended as the definitive

source for a broad spectrum of basic business facts. The prime directive of traditional database development is to *optimize efficient handling of a known set of transactions.* The data warehouse prime directive is to *optimize availability of cross-functional data in support of future needs.*

A data warehouse is not just a project. A project is the *managed organization of resources to produce a defined set of deliverables in a specified timeframe.* A data warehouse effort does involve several time-sequenced deliverables: raw atomic data acquired from operational systems, collections of data prepared for use, and executable procedures to perform this processing. However, these deliverables are produced using exploratory techniques that are radically different from traditional methodologies. And, though there is a definite beginning, there should be no end to the data warehouse. Sourcing, packaging, and delivering data will continue and evolve as the business changes.

But the differences go deeper than this. When you initiate a data warehouse implementation, you are investing in a whole array of new services supported by the deployment of new technical infrastructure. You select and introduce analysis tools. You plan and implement training programs. Support procedures need to be designed and support mechanisms must be put into place. This effort takes place amidst an evolving set of new information management practices.

A data warehouse is a program to manage sharable information acquisition and delivery universally.
A data warehouse, like your neighborhood library, is both a resource and a service. The value of library *resources* is determined by the breadth and depth of the collection. The value of library *services* is based on how quickly and easily they can assist you in finding and using what you need. Unlike a library, a data warehouse must take on the role of *manufacturer* and *distributor* as well. As in a factory, raw materials are collected from operational systems and packaged for use by information consumers. Similar to a public utility, a data warehouse uses a common distribution network to deliver products to the point of use. The value of a factory is determined by how well it produces the finished goods that consumers most need today and into the future. The value of a utility is measured by the efficiency and timeliness of delivery.

Avoid the false promises of the "just do it" advocates.
Whew, this sounds like you must propose an expensive mega-project. Don't all the other data warehouse advisors say that you can - in fact, that you should - start on a small-scale, limited-exposure, pilot project?

If you follow the advice of the *quick start* or *just do it* advocates literally, you are signing up for a quick trip to nowhere. Many people are duped into thinking that a pilot project is the limit of their planning horizon. There is nothing wrong with moving out fast *if* you know where you are going. Starting with a real plan need not take forever, only two or three months. It will be time well spent.

Take the time to look for strategic opportunities. Create the outline of a long-term plan. Define potential future destinations and alternate paths to get there. Pitch your pilot as the logical first step.

Data warehouse containers

CORE DATA: Raw data acquired and lightly refined form the operational system.

- **Base –** basic elemental business data at the most granular level (ideally atomic)
- **Reference –** descriptive or identifying information for based data (i.e., master files)
- **Pre-derived –** calculated or summarized values retained to ensure auditability of results

COLLECTIONS: Packaged sets of information prepared to satisfy consumer needs.

- **Aggregate –** accumulation of related data into a single table to improve ease of access (i.e., pre-joined)
- **Summary –** consolidated data summarized on one or more dimensions from base tables
- **Prepack –** packaged data in the unique structure (e.g., starjoin, hypercube) required by a specific tool
- **Export –** unloaded flat file data created for a specific use and requiring delivery to another location

ARCHIVE: Active extension of a warehouse that retains enterprise history for on-demand retrieval.

- **Atomic history –** atomic level data placed in an archive immediately after base tables acquisition
- **Time fragment –** dataset containing data for a defined timeframe and optimized for selective retrieval
- **Context history –** periodically stored snapshots of reference, definition, and navigation data
- **Strategic results –** saved set of any collection deemed strategic for "corporate memory"

METADATA: Data about data to facilitate access, integrated tools and utilities, and automated processing.

- **Definition –** data names, formats, meanings, source, usage, and other defining characteristics
- **Navigation –** catalog of available data, cross-reference summary, and help/support information
- **Administration –** control data for process management, monitoring, and integration across the data warehouse

Outline Information
Deliverables

Moral: *Learn the differences about data warehousing since commonly held beliefs will cause you to fail.*
Now let's turn our attention to the outputs of a data warehouse. First, it helps to understand the various types of data that should be stored on your warehouse shelves. Then we can discuss how you decide what to put into these containers.

A data warehouse contains a far wider variety of containers than traditional applications.
Expect to design four major types of containers: *core data, collections, archive, and metadata.* (see box)

Core data is the raw material from which all other data tables are built. This data should be brought in with minor cleanup and conversions to make it more useful. Avoid selective acquisition and take all rows from the source. Avoid summarization on the way in and keep the detail. Avoid creating new calculated values at this point, but it is okay to import derived data when it is essential to maintain auditability. New derived data is created later when the collections are produced. Core data is typically stored in a normalized form.

Collections are manufactured in the warehouse environment to satisfy known and anticipated needs. Aggregations are produced to support repetitive access of related data drawn from multiple base tables. Summaries are created based on usage experience, not by *a priori* analysis. Prepacks and exports satisfy specific needs of your consumers or the tools they use. By definition, all collections are denormalized.

Though the terms used here may be unfamiliar, virtually every warehouse effort builds tables containing the kind of data described above. Unfortunately this is not the case for archive and metadata. Though extensive lip service is given to the need to maintain history, few projects go beyond adding timestamps to online data warehouse tables. A virtual avalanche of words is lavished on the essential role of metadata. Yet in the daily grind, this is always something that can be addressed in the next release.

The central role of the archive and comprehensive use of metadata are two features that clearly set the data warehouse concept apart from prior information access initiative. Build them into your plan at an early stage. As with everything else, you can start small, but do include these features.

The **Archive** contains data packaged for easy retrieval and reintegration with the online warehouse data. Raw data at the most granular level (atomic) is stored to ensure no loss of information over time. In some cases, the archive is the only place where this level of detail is available. Context history is stored to ensure that you can recreate the data environment that existed at any period.

Metadata is the most discussed but least understood. Most people think of a data dictionary first, but this is the least valuable form of definition metadata. Within this subcategory, knowing the ultimate source for a data element, meanings rather than descriptions, and references to how the data is commonly used are more important than the basic data dictionary-style information. Navigation data is intended to help the consumer find what exists and how to use it. The card catalog analogy suggests some of the possibilities of navigation. The least understood form of metadata, administrative, is possibly the most powerful in the long run. It plays an essential role in process integration and automation.

A data warehouse must be designed to support unknown future needs of unidentified consumers.
When you turn your attention to selecting the data for your core tables, you immediately run into a wall. How can we possibly plan for the unknown? The most difficult hurdle is to realize you must try.

There are three classic ways to fail when an information consumer searches for data: 1) not having the data in the form they desire, 2) not having the required detail, or 3) not having the data element they need at all. The first problem is a matter of design that can be fixed on the fly. A collection can be built to satisfy this need from core data already in the warehouse. The latter two, however, are a factor of advance planning. It takes considerable time to acquire missing data - time the con-

sumer does not have.

Traditional requirements gathering methods must be abandoned or extensively modified.
Traditional practices and commonly held beliefs lead us to:

- **Focus** too concretely on what existing users know they need and give them only what they ask for;
- **Eliminate** data not known to be required in order to control the scope of the project or size of the table;
- **Consider** users closest to the source systems to be the primary information consumers;
- **Fixate** on response time and storage issues and other operational concerns.

Only getting what is a *known requirement* leads to repeated, costly trips to the well. Try, instead, to:

- **Anticipate** data needs based on business trends, all potential usages, intuition, etc.
- **Include** data not known to be worthless in order to eliminate future costs which will be greater;
- **Seek** potential consumers widely, particularly those not close to the source in space or function;
- **Concentrate** on availability and flexibility while creatively managing performance and volume issues.

Start from the premise that you will drain every potentially usable nibble of data out of the source systems. Fight at all turns the tendency to accept summarized data rather than the ultimate atomic level of detail. Make your associates justify not including data rather than the other way around. You will have to back off, to some degree, but you are coming from a more defensible position for the long run.

Outline the Technology

Moral: *How you build it is at least as important as what you build.*
Much attention has been devoted to Information Architecture in data warehouse articles and conferences. The vast majority of data warehouse speakers and writers are (former) database analysts, data modelers, data analysts or other practitioners of data-centric fields. Almost no attention is given to technical architecture or the practice of defining the optimal way to build a warehouse.

Process components are the building blocks of a data warehouse.
A data warehouse is more than a computer, a DBMS, a network, and a client/server query tool. These things are the infrastructure services that form the backbone upon which you build a data warehouse. But what is the essence of the data warehouse itself?

From the perspective of technical architecture, a data warehouse is a set of processing components that interact to make enterprise information available to information consumers:

- *Acquisition* – bringing data into the warehouse form the operational environment.
- *Preparation* – packaging the data for a variety of uses by a diverse set of consumers.
- *Operations* – controlling repetitive tasks and managing warehouse inventory for optimal processing.
- *Navigation* – assisting the information consumer in finding and using warehouse resources.
- *Access* – supporting direct interaction with data warehouse data by consumers using a variety of tools.

- *Delivery* - providing subscription delivery of data as an alternative to interactivity.

To build a complete data warehouse solution you need to build, buy or manually perform these services.

Create a technical plan for your data warehouse that is technology-free.
A well-defined technical plan allows you to specify how your data warehouse will function and what services it will provide. The plan can define what technical components you intend to offer initially, and with subsequent releases, thus providing a common framework defining growth over time.

A technical plan that does not list any vendor-specific products by name (technology-free) provides an excellent foundation for making the decision of what to buy, build or perform manually. It can be used to frame the requirements for product selection and, later, to provide purchase justification for essential components.

Determine Automation Targets

Moral: *Failure to automate a data warehouse leads to failure.*
The basic development tasks are so deceptively simple that most people resort to our normal manual practices. This includes writing all the code by hand and writing a separate custom procedure for each activity.

To continue to add value, a data warehouse must adapt to changing needs. New data must be continually acquired. Available raw data must continually be packaged and repackaged to support new demands, tools, and forms of analysis. The key to survival is to remain dynamic. The key to remaining dynamic is to automate repetitive tasks and reuse the prior deliverables whenever and wherever you can.

Go from design to implementation in one step.
The holy grail of Computer-Aided Systems Engineering is to eliminate all coding. For the most part, CASE has been a monumental failure. Sophisticated upper CASE modeling tools and methods failed to reduce development time, and resource requirements often increased. Now in the specialized field of data warehouse development, CASE techniques are finally beginning to pay off. I will give you just two examples.

First, it is now possible to use graphical modeling tools to design, implement and maintain a data warehouse database. In fact, this is the preferred method. Maintaining synchronization between the graphical image and the physical database is essential. You accomplish this by eliminating all manual manipulation of the warehouse database data definition language statements. This can and does work for a data warehouse.

The second high productivity area involves the new breed of *extract and load* code generators. Using very different approaches, these products allow you to define the data source(s), the warehouse target table, and the mappings and transformations to get from one to the other. They then generate code to operate on the specific platforms in your environment. Productivity is increased by a factor of two to five times for initial development, and is considerably greater for revisions to existing feeds. Given the number of data feeds, the rapid change, and the ability to reuse prior definitions, these products pay for themselves very quickly.

Buy generalized utilities whenever and wherever possible. Otherwise, consider building them.
Though the current concept of a data warehouse is new, many tasks necessary to develop and operate one are well understood. Existing tools and utilities can be harnessed for data warehouse use. New products are regularly introduced to simplify data design or management tasks. Where a repetitive activity exists, there is an opportunity to develop a custom utility. Examples abound. Here are just a few:

- A reverse engineering utility can be used to create a logical model of the source application.

- There is a growing category of products that scan the data itself to help you define one-time cleanup operations or ongoing conversions necessary to repair integrity or consistency error.

- Transformation products exist for most platforms and format combinations to simplify data movement.

- Some analysis tools require data to be structured in a very explicit fashion. Rather than write a unique piece of code for each occurrence, write a generic utility to prepare data for these products.

Provide the glue to integrate off-the shelf products into a seamless process.
Many labor-intensive data warehouse tasks can be streamlined by simply stitching together existing products and procedures. For instance, the dynamic nature of a data warehouse means there are many frequent changes in the batch processing cycle. Wouldn't it be nice if you could update a diagram of the batch dependency plan to change the schedule? How about automating the process of error escalation? Think how much easier it would be if all interested parties were automatically notified by e-mail when a data load fails.

One solution incorporates a PERT charting applications, the DBMS of your choice, your existing schedule management software, one or more mail packages and a mail gateway along with a client/sever development tool and a bit of code. All this functionality, and more, can be made available by creative use of building blocks.

Plan for automation and reuse in your first project and expand from there.
No data warehouse program can operate cost-effectively without introducing a high degree of automation and reusability. Many opportunities exist. Just search for repetitive tasks that can be reduced or eliminated. Don't wait. Begin in the pilot.

Select the Tools

Moral: *Abandon completely even the desire to find a silver bullet.*
Many data warehouse project teams waste enormous amounts of time searching in vain for a silver bullet. They believe their mission is to find the one access tool that will handle all the needs of all their users. Don't even try. It can't be done. One size does not fit all.

There is a wide and ever-growing diversity of tools for data access, exploration, analysis, and presentation. The appropriate mission of a data warehouse or decision support team is to understand these diverse alternatives and properly match the tool to the intended usage.

The names typically applied to tools are ambiguous and confusing. Generally every new name only obscures the issue more. What is the definitive definition of *ad hoc query tool*, *decision support system*, *executive information system* or *online analytic processing*? Some terms, like EIS, carry the burden of years of overzealous selling and underwhelming results.

To evaluate tools, you need to slot the alternative into categories that allow for meaningful comparison. Since the traditional terms add little discriminatory power, where do you turn? The first part of the answer is to create purely functional categories. For example:

- *Browser* – simple direct access to DBMS tables or views without join or business semantic layer support.

- *Query* – two-level client/server tool providing complex direct access to data with full SQL support.

- *Multi-dimensional analysis (MDA)* – provides unhindered data explosion, drill-down, and surfing capabilities.

- *Syntax* – language-based (4GL) direct access to data with full SQL support.

- *Fixed* – pre-planned data access with generally pre-computed options (EIS tools often fit here.).

- *Custom* – custom-developed data access exploration, analysis or presentation application. These applications may be developed using a specialized DSS toolkit, 3GL, or 4GL.

One category missing from this list includes tools such as PC spreadsheets, charting and presentation tools. In most enterprises today, they are the final packaging vehicles for management presentation. However, they are rarely the tools of choice for direct access to the data warehouse. Also, this taxonomy only includes interactive tools. Many mature technologies exist for batch reporting. A new class of non-interactive tools will come on the market over the next several years. They will use agent-processing techniques to scan remote databases and pass result data back to the client asynchronously.

Match the tool to the consumer, context, and usage.
The next step in selecting access tools is to survey your potential consumers. You need to collect information in three broad interrelated areas: 1) who they are and what their role is, and how they operate, 2) what the business context is, and 3) their typical usage of data. This information will help you select the appropriate tool category(s) for each consumer. Then rank order your potential consumers according to who will be the most involved, productive, and powerful participants. The priority tool categories should become apparent. Several examples follow:

- A *product manager*, who interacts extensively with data, identifying patterns of purchasing behavior to plan new marketing campaigns, will probably be best served by a multi-dimensional analysis tool.

- A *marketing analyst* studying store site demographics to profile differences between selected geographic areas is more likely to need a query tool. Analysts who are very business savvy, but not very technically inclined, will be better off with a query tool that provides a business-like, semantic interpretation layer.

- When you see repetitive activity involving the same selection criteria and domain, the power of query and MDA tools is overkill. A *financial analyst* who regularly produces a listing of the top and bottom performing products should use a browsing tool.

- When a *manufacturing quality assurance group* routinely studies defects using standard variables and analyses, a fixed mode tool or a custom application is suggested.

Build a best-in-class tool set of supported products.
Of course you can provide quality support for only a small, finite number of tools. After identifying the tool categories most appropriate for your consumers, you are prepared to evaluate what the market has to offer. The objective is to select the tools you intend to support in each category.

You will never be able to eliminate pirate activity. Some consumers will always find justifiable reason to pick a non-supported product. We have found that it is far more productive to explain the value of selecting a supported product than to wage a battle to enforce a *standard*. You will lose customers and goodwill.

Remember to plan regular surveys of the market. You need to remain aware of new features of existing products, new products in each category, and the advent of wholly new categories. Be ready to change your supported product mix when there is genuine need. Resist the tendency to move capriciously to the next best thing. You and your customers need time to become proficient and successful before moving on.

Lay Out the Project

Moral: *A data warehouse project is more like scientific research than anything in traditional IS!*
The normal IS approach places considerable emphasis on knowing what the expected results are before committing to action. In scientific research, the results are unknown up front, and emphasis is placed on developing a rigorous, step-by-step process to uncover the truth. The activities are hands-on, involving regular interaction between the scientist and the subject and among the participants on the project.

Adopt an exploratory, hands-on process involving cross-disciplinary participation.
The following situations indicate a project in trouble because these techniques are not being used:

- The project has proceeded for two months and nobody has touched the data.

- The future consumers are not involved hands-on from day one throughout the program.
- The team members doing data design (modelers and DBAs) have never used the access tools.
- The summary tables are defined before the raw atomic data is acquired and base tables have been built.
- The data design is finished before participants have experimented with the tools and live data.

Initiate multiple parallel activity tracks simultaneously.
A data warehouse project involves a wide range of deliverables that can be worked on in parallel. The track scheme we advocate is:

- *Management* – obtain sign-off, document milestone activity, and coordinate track deliverables.
- *Consumer* – discover the consumer needs and design the packaging requirements.
- *Model the database* – define and iteratively implement the database.
- *Source data* – define the technical architecture, select products, and perform the build and implementation tasks.
- *Tools* – evaluate, select and make available the access tool set.
- *Operations* – plan operational infrastructure and processing cycles.
- *Support* – develop documentation, and design and implement training programs and help-desk services.
- *Integration* – integrate the metadata and deliver navigation services.

This scheme offers a division of labor proven to work in practice. You do not need nine or more people to make this scheme work for you. But it is hard to gain the advantages of parallel activity with only a few participants.

Forcing closure on specific results too soon leads inexorably to disaster.
A general rule of thumb is to break the initial project roughly into thirds: *initiation, exploration,* and *implementation.* We assume that the project begins after the planning activity, when you assemble the bulk of the team for action.

Initiation Phase

This phase continues through delivery of the first trial database of live core data. On the consumer track, you outline the expected information deliverables. On the source data track, basic mapping of data from the source systems is defined. The model database track delivers a data warehouse data model containing only core tables and then proceeds to create and load the initial data warehouse database.

Ideally, the access tools are selected by the end of this phase. Optionally, you may identify finalists that are to be evaluated during the exploratory phase. Design deliverables are created for the other tracks (technical plan, support, operations, and integration). All the technology to support the development of the data model, database, and data acquisition procedures must be ready when needed during the initiation phase.

Exploration Phase

This phase involves iterative revision of the database and the preparation of procedures that maintain it. On the consumer track, a team of technical and consumer participants uses the access tools suite to develop representative samples of anticipated results. Based on the experience, the model database team members define aggregates and summaries that offer a good balance of user flexibility, production efficiency, and runtime performance. On the source track, the participants refine the

mapping and transformation logic based on hands-on experience with the data. They clean up the source data and define conversion logic to correct consistency or integrity problems. They modify the core tables based on tradeoffs between acquisition and preparation activity. All technical infrastructure and tools work concludes during the exploration phase. The operations track prepares for implementation and production migration. The support track prepares for initial consumer training in the next phase. Navigation services, the metadata tables, and the metadata maintenance procedures are developed at this time.

Implementation Phase

This phase starts with a freeze on source investigation and consumer experimentation. It concludes with formal release of the current version of the data warehouse. The final data definition and data manipulation and procedural changes are made. All component services (acquisition, preparation, operations, navigation, access, and delivery) are installed and unit tested. The production schedule is implemented and a full system test is conducted. Initial consumer orientation and training are offered during the production migration activity.

Continue to look over the horizon at the future possibilities.
To ensure that the data warehouse program evolves continuously, management must maintain focus on future opportunities.

By the beginning of the implementation phase of the current project, management must already be planning the subsequent project. What new subject area will be tackled? What new consumers will be recruited? What new techniques will be tried? What new technical components will be implemented? And very importantly, how has what you are offering been received and adopted?

Solicit a Team

Moral: *A cross-functional program must be inter-disciplinary. Use maximum leverage.*
The question I am most often asked is: " How many people are needed to build a data warehouse?" My typical answer is: "Anywhere from two and a third to a cast of hundreds." From project to project, there are substantial differences in objective and scale. But the resource question is also biased by how you define participation.

When is 2.33 equal to 4 or more?
The smallest project in my experience used two systems analysts and a part-time database programmer to build a data warehouse from scratch in eight month. Their subject area, R&D resources, was quite small. Their technical objectives were limited and they supported only one access tool. Yet, they are successful enough to move into other subject areas and add new consumer groups while expanding their direct staff to five. When you look closer, the 2.33 full-time equivalents significantly undercounts real participation. At least four full-time equivalents were expended on development alone. This includes developer on the source systems who implemented new and modified data feeds, reviews by a DBA, vendor contractors, an advisory consultant, and others. An unknown number of consumers, developers, and managers were involved before, during, and after development.

Consciously manage multiple spheres of involvement. This is the only real way to "get more from less."
Everyone wants to know the minimum number of people it will take to get the job done. When pressed for a single number, my answer is seven. This is roughly the critical mass needed to sustain a successful program for a medium to large business. But how this resource is structured varies too wildly to draw any general conclusions. The creative manager builds a small core team and borrows the rest. This is the key to delivering more with less. Our definition of levels of involvement is:

- *Core* – dedicated individuals whose sole mission is the data warehouse program.
- *Extended* – team members form developer or consumer groups with specific roles over the entire project.
- *Participant* – individuals who are assigned specific deliverables at one or more points in the project.
- *Involved* – people consulted for expertise, managers solicited for influence, etc.
- *Impacted* – consumers or developers whose work patterns will be changed by the introduction of the data warehouse.

- *Aware* – anyone who is relatively current in their knowledge of the data warehouse objectives and progress.

- *Oblivious* – those who know little or nothing about the data warehouse program.

Actively solicit extended team members and participants from other groups, departments, and cost centers. A data warehouse requires broad expertise from multiple disciplines. Sign on developers from the source systems to acquire the data. Enlist consumers and the IS support personnel. All the IS infrastructure groups need to be on your side. If not, they will be in your face.

Reaching out to other groups for expertise increases the management hassles, no doubt. But it can be immeasurably valuable in solidifying cross-functional support. Your data warehouse project should be *the* future direction for information access. Competition is wasteful and deadly. Don't even try to do it all yourself. Keep the core team small and recruit a cast of hundreds.

The talent you need on the core team is the hardest to find.
So, who should be on this small core team? Typically, the first talent hired is a data modeler and the second is a database analyst or DBA. While you need these data specialists, you don't need them as fulltime, permanent participants of the core team. The best individuals at the core of your team are people who intimately understand the broad issues of a complete data warehouse program. Ideally, they studied the field extensively and participated in at least one successful project. These folks are hard to come by.

Your next choice should be those with experience of packaging data for consumer use: a data analyst with DSS experience, a 4GL programmer with information center or end-user support background, a data–centric developer with real experience using spiral or interactive methods. You need data engineers to get started and to add new data resources to the warehouse. They create the potential. You will need data analysts or data usage specialists to stay in business. They deliver results.

Evangelize Endlessly

Moral: *Tell them what you are going to say, say it, then tell them what you said.*
The story of a successful data warehouse program is of a story often told. You need to keep the vision alive while delivering results. When you begin planning for your own data warehouse, plan to spend a considerable amount of your time on the campaign trail selling and reselling your concept.