STRUCURING YOUR DATA WAREHOUSING PROJECT : MOVING FROM THE CONCEPT TO THE REALITY

Peter R. Welbrock, Consultant, Maxim Group, Wayne, PA

ABSTRACT

For every successful Data Warehousing project, there are many that fall by the wayside; ignored, forgotten or scrapped. Although just a continuation of the familiar I.T. pattern, these failures represent a fundamental misunderstanding of Data Warehousing and should be treated as such. This paper is intended to focus on the process of Data Warehousing, discussing specifically the importance of a structured transition from a specific concept to reality of implementation. This paper will purport the view that Data Warehousing projects often fail due to the lack of understanding of this transition, either due to it being ignored, or due to ignorance of its existence.

INTRODUCTION

It has been written by nearly every author who broaches the subject of Data Warehousing that it should be treated not as a 'thing' but as a 'process'. This means that, unlike the claims of a large portion of the software industry, it is not possible to buy a Data Warehouse.

A Data Warehouse Analogy

Imagine a situation where you have to produce a meal for a set of disparate people that are coming to your home. Each of those people will expect a full meal, but might not want exactly the same dishes (what if one were a baby?). To produce the meal, you will have to go through a very complex process, from buying your food (*extracting the data*), through to the preparation *(cleansing and transforming the data)*, serving *(moving the data to where it is needed)* and presentation *(front-end tools to use the data)*. This simplified process is in some ways very similar to that involved in the successful implementation of a Data Warehouse.

Software and hardware are the tools you will use to get the 'meal onto the table'. The refrigerator *(the Operational System)*, the oven *(one of many data transformation engines)*, the serving platter *(an OLAP tool)* might all be necessary to create your meal, but will not help you decide what food should be bought *(what data elements are required to make up the Data Warehouse)*, from which shop the ingredients should be bought *(the Operational System where the data resides)*, whether the potatoes should be scrubbed or peeled *(what the business rules are)*, or whether the people eating the meal need to know all the ingredients *(should the users of the Warehouse have to understand where the data comes from)*.

This analogy can be taken to extreme levels (does the waiter whipping up zabaglione at your table represent true Client/Server catering?), but the point it represents is that in designing a Data Warehouse, do not think about it as a Database, or an OLAP tool, but simply as a means *(a process)* of 'getting food on the table' *(supplying information to those that need it)*.

Once this concept of 'building' a Data Warehouse is understood, it naturally leads to a 'de-cluttering' of the design process. Abstraction of this design process *(the concept)* from the implementation process *(the reality)* leads to a more successful Data Warehouse since it allows for a more objective and less restrictive approach that will ultimately pay off in terms of universal (enterprise-wide) functionality and flexibility.

This approach to creating a Data Warehouse, however, inherently involves another step : moving from the concept to the reality. It is no good if the 'conceptual warehouse' bears no resemblance to the 'implemented warehouse' and we must therefore also plan how we are going to mirror our concept in reality.

WHY DO DATA WAREHOUSING PROJECTS FAIL?

There are currently many successful Data Warehouses, but as mentioned above, many more failures. How many times have we heard the plaintive I.S. voice bemoaning the fact that they have bought an expensive server, installed the world's best Relational Database, populated numerous (usually normalized) tables with data, supplied an ODBC driver to the users and lo and behold, the ungrateful user community isn't even using it!

This is the classic case of 'reality' without the 'concept'. This is a very common mistake in Data Warehousing : enterprises create what they think is a Data Warehouse using what has been described as the 'Big Bang'[a] approach, where the entire enterprise is addressed all at once, find it isn't used (i.e. the project is a failure) and then they go back to the drawing board. Even at this stage, the same mistake is often duplicated, but less apparent, since the common pattern is for the Data Warehouse to be scaled back, so the problem is only seen on a more iterative level.

WHAT CAN BE DONE?

One very common misconception is that because Data Warehousing projects deal with computers, that they should therefore be technical in instigation. Indeed, if one accepts the 'concept' and 'reality' approach, then the former should be developed with only modest help from I.S. Like the French philosopher Compte, who stopped reading months before writing because he believed in 'cerebral hygiene', the 'concept' of the Data Warehouse should not be cluttered with extraneous and irrelevant issues (like technology), but should be based purely on business needs.

This proffers the obvious question of what happens if the 'concept' is completely out of balance with the potential of the 'reality'? What happens, for instance, if the 'concept' contains the necessity for the Warehouse to contain atomic data that would result in single requests needing multiple reads of 1 billion record files, when the current platform for the organization is Windows 3.1? The answer to this is that is doesn't matter so long as the process of moving from 'concept to reality' is correctly understood and managed.

*The **concept** of what the Data Warehouse should be, as compared to how it is actually realized, need not be the same so long as the process that manages the 'concept to reality' is correctly **understood** and **managed.***

One of the major weaknesses of Information Systems is its understandable preoccupation with implementation. This is grounded in the historical significance of Operational Systems, where the rules (specifications) are more finite and consistent than in a Data Warehousing environment.[b] This preoccupation is less damaging (although by no means innocuous) in the Operational System environment, but is one of the major reasons that Data Warehousing projects are not as successful as they should be.

A very popular method created to ensure that an enterprise is ready for a Data Warehousing project is the ubiquitous 'checklist', which although of use, misses the point. Data Warehousing is a process by which an enterprise can affect its future by looking at its past. It therefore gives the enterprise an element of control without which, at worst will mean the enterprise will perish and at best will mean potential under performance. Every enterprise should be encouraged to undergo the process of looking at a Data Warehouse at the conceptual level, and the process to move this concept to reality. Only then can the real costs (to the enterprise, not of implementing the Warehouse) be ascertained. By and large, all checklists do is to give an enterprise an indication whether they are ready to implement the Warehouse, before they know the business costs of not doing so. The checklist therefore puts the cart before the proverbial horse and encourages the I.S. syndrome of being implementation centric.

SO WHAT IS THE DATA WAREHOUSING PROCESS?

Up until this point, we have been looking at a technique to develop a Data Warehousing process that differs from the standard approach. The real basis of this approach is to move away from being 'implementation centric' by dividing the process into three distinct parts:

- **Conceptual/Business Phase**
- **Concept to Reality Phase**
- **Implementation Phase**

The final phase is discussed at length in virtually all industry publications related to Data Warehousing, and will therefore not be covered in this paper, but the previous two phases require a slightly different approach to Data Warehousing and will be discussed at some length below.

CONCEPTUAL/BUSINESS PHASE

It is generally accepted that a Data Warehouse process is not only specific to industries, but to separate enterprises within those industries. It is arguable, but believable, that this process is also unique to departments within each enterprise, although this then gets into the Data Warehouse / Data Mart discussion. Given that the above is true, then obviously this conceptual phase of Data Warehouse development, by nature, will also differ. This is true to a point, but the intent of this phase will be consistent: the Conceptual Warehouse is the non-technological rubric for the Implemented Warehouse.

The progress and evolution of the Implemented Warehouse is based upon comparisons to the Conceptual Warehouse.

This conceptual phase is the ultimate opportunity for the users to gather all of their requirements based upon business need justification, as opposed to technological cost versus business need justification. This is one of the very many confusing aspects of most current methods of developing a Data Warehousing process : justification is often seen by looking at the ratio of business benefit to technological cost. This confuses two entirely different issues and should be avoided at this stage. Technological cost should not be a factor in deciding whether a requirement should be in the Conceptual Warehouse. This decision comes at a later stage.

The inclusion of an enterprise requirement into the Conceptual Warehouse should not have to be technologically justified. The basis for inclusion should be purely based upon the belief that the realization of that requirement will allow the business more control over its future.

The above point can best be explained by an example: let's suppose that the Marketing Department prophesies that it could increase sales of their line of squeaky gadgets if they could send direct mailings into the zip codes where their furry widgets sold well on Saturday mornings. In other words, they would like a breakdown of sales by product by zip code by time period. This is a perfectly valid option to include in the Conceptual Warehouse, although unfortunately current sales are not collected by time period, and to do so would cost the company $1 million. This is a situation where there would be a difference between the Conceptual and Implemented Warehouse.

Remember, although it would be ideal if the two (Conceptual and Implemented) matched exactly, just because they don't does not mean the requirement should be excluded. It should remain there as a constant reminder that opportunities might well be lost because of a shortfall in information. The Warehouse process is therefore acting as it should do: it is giving the business control over its future, in this case by pointing out that something is not currently being done.

The Conceptual Warehouse therefore, by necessity, contains high level information. It will contain enterprise requirements, but will not state where this information resides, or in what form, or how it should be captured. That is entirely irrelevant to the Conceptual Warehouse.

High level business rules also need to be included in the Conceptual Warehouse. In the above example, what exactly is a time period as the business (not the Computer Systems) require it to be? In this case, a time period might be defined as either before or after 12 noon. If the Operational Systems were to currently collect time periods, the likelihood would be that the transaction would be time stamped, so one of the data transformations required would be a conversion from time to either morning or afternoon. These business rules will directly translate to the Implemented Warehouse if required (not so in the case of time period, since it is not currently collected). It is important to note that these business rules would also have an owner attached, although this will be discussed more later.

So as a quick summary, the Conceptual Warehouse can be seen as the enterprise's wish list of requirements based not upon the narrow restrictions of current systems or resources, but entirely upon what is needed for the enterprise to control its future (as much as is feasible from the use of data).

The Conceptual Warehouse is a wish list of what is needed on an information level for the enterprise to have maximum control over its own destiny.

As mentioned above, not only are high-level information requirements included in the Conceptual Warehouse, but also high-level business rules. A further piece of information should also be collected: the owner of the requirement. How many times in Data Warehouse development is a vast amount of time and money spent to make available certain pieces of information, only for them not to be used? A level of accountability should also therefore be built into the Conceptual Warehouse which will flow through to the Implemented Warehouse, if applicable.

One of the major issues in Data Warehousing is data ownership, but this does not go far enough. It is also necessary to have requirement ownership, and therefore business rule ownership. This naturally brings us to another classic problem with Data Warehouses, the inconsistency of business rules : should the total sales of furry gadgets include those sold by the stores belonging to the company recently taken over, or should these be tracked separately? Business rules become more important the more summarized the data becomes, yet these rules must be made and who owns these rules should be obvious, not part of some complex and devious decision made by a mythical person who left the enterprise three years ago.

One more piece of information that should be recognized and documented is the reporting period that the enterprise requirement requires. For instance, in the example that we have been using, where sales is reported information, would we want to see this on a daily basis, weekly basis, or monthly basis. This is a key piece of information since it will help in the design of the Data Warehouse Implementation by determining the level of data collected.

Finally, the Conceptual Warehouse should contain information that gives a priority to different requirements. This is always a difficult topic to approach, since everyone is under the illusion that their requirements are of the highest importance. Very often, this assignment of priorities has to be made, not by those that requested them, but by someone at a higher level who is in a position to see them from an enterprise wide perspective. There will, of course, be disagreements, and each enterprise might even formulate some form of algorithm (scoring technique) to assign relative importance. Without this assignment, however, the creation of the Implemented Warehouse might well be developed in a random fashion, with little regard to broad enterprise needs.

'Conceptual Warehouses' in the real world

Like the traditional Data Warehouse, the Conceptual Warehouse has to be dynamic, including as many enterprise requirements as possible to theoretically give the enterprise the best chance of control over its future. This is no small task and has to be structured and organized. Up to this point, all discussions have been theoretical, and it might be a good time to demonstrate an example of how to work a Conceptual Warehouse in the real world.

One of the continual fights to get any type of system based on computers successfully implemented is to gain support from the users. They are the most vital source of information, whether building a Data Warehouse or an Operational System. Like many of us, however, the users are nearly always willing to talk about themselves and this is the 'Achilles heel' that must be used to develop the Conceptual Warehouse. As with any new development within a company, a project team has to be formed and support needs to be garnered at the very highest level to ensure continued and consistent support. This has been documented elsewhere and will not be covered in this paper. We are really more interested in the form the Conceptual Warehouse will take.

The Conceptual Warehouse is no more than metadata about the business. It contains no data itself. The contents of the Conceptual Warehouse can therefore be treated and stored as metadata.

The Conceptual Warehouse contains no more than a set of high-level requirements, business rules and owners. This is, in effect, metadata. It is metadata about the business : what information does it, the enterprise, need to maximize control, what are the rules that determine that this information will remain consistent, who defines the rules and who needs this information. Because this is simply metadata, not about the data. but about enterprise requirements, it is therefore possible to structure it as such. The tools that can be used to store this information can be as simple as a SAS® data set, or it could be incorporated into a tool like the SAS Data Warehouse Administrator®. It can be structured as a Star Join Schema[c], where a fact table containing the requirements themselves will be linked to dimension tables that contain business rules, owners and where it is addressed in the Implemented Warehouse (if at all). This final dimension, relating the enterprise requirement to the Implemented Warehouse, is very important, because it is the link that will allow for the check on the progress and evolution of the Implemented Warehouse based upon comparisons to the Conceptual Warehouse.

One of the advantages of using the Conceptual approach is that requirements can be collected on a Departmental basis, rather than having to look at the entire enterprise at one time. It is even possible to start with one Department (or one high-level executive) and then begin to move to others. This allows the size of the project to remain manageable, without decreasing its overall effectiveness.

The Conceptual Warehouse will be a work-in-progress, as will the Implemented Warehouse. There are many less obvious uses that will make it a very important tool. If the Conceptual Warehouse is structured carefully then the following benefits should be achieved:

- Reports will be available to let people know (in business parlance) what is available within the Implemented Warehouse, what is being worked on and what people deem as desirable to the business, with no restriction based on technical justification.
- It can be used to encourage an 'open door' policy to the Data Warehouse, since suggestions can be incorporated without the need for lengthy justifications and will be available for all to see.
- It should, if structured correctly, create a healthy breeding ground for discussions about the business.
- It acts as a continual reminder of what the business is not doing to control its own destiny, represented by the enterprise requirements that have not been included in the Implemented Warehouse.

These benefits are over and above those that will be directly related to the creation of the Implemented Warehouse, which will be discussed below.

CONCEPT TO REALITY PHASE

At this stage, we have what we have termed a Conceptual Warehouse. It contains no data as such, but metadata pertaining to what the enterprise needs, to retain maximum control over its destiny. This metadata is in the form of business requirements that have associated business rules and owners of both the requirements and the rules. The metadata is stored in a structure that is self-contained (it does not pull information from other sources) and is linked to the Implemented Warehouse, inasmuch as it is possible to find out any enterprise requirements in the Conceptual Warehouse that have actually been included in the Implemented Warehouse.

Up to this point, there has been little, if any, direct involvement from a technological standpoint. There has been no technological cost justification for including requirements in the Conceptual Warehouse into the Implemented Warehouse, so therefore no confusion over the issue of what the enterprise needs as opposed to what can be sensibly supplied by current Operational Systems. As mentioned before, the two are entirely separate issues that should not be muddled when designing the Data Warehouse process.

It is essential that requirements for information that the enterprise needs to gain as much control as possible over its future, are not confused with either the availability or cost effectiveness of obtaining that information

Indeed, this comes back to the classic problem that enterprises come up against time and time again : should software determine how the business is run, or vice-versa? There is no universal right or wrong answer to this question, since it depends on an enormous number of factors, but whatever the situation, in looking at the enterprise from a Decision Support basis, as opposed to an Operational basis, isolating requirements from reality will open up far more options and therefore allow far more control.

So assuming we have reached the stage where at least part of the Conceptual Warehouse has been built, the project has to be broadened to allow for the involvement of those people who will actually be building the Implemented Warehouse. Very often, this will be an I.S. department, although this is not necessarily essential for a successful Warehouse process. For the sake of this paper, let's assume that it will be the I.S. department. Along with the I.S. department, there will have to be a Conceptual Warehouse liaison; the link to the enterprise, which is the single most important role. since this translates and interprets the enterprise requirements to the I.S. Department.

The 'Concept to Reality' phase involves the following steps:

- Breaking down each of the enterprise requirements into information needs.
- Documenting what data elements actually make up the information needs, and the Operational System (if any) from which these will be obtained.
- Deciding which of the enterprise requirements should be included in the Implemented Data Warehouse and in what order.

To successfully approach each of these three steps involves a great deal of work. The entire point of approaching the Data Warehousing Process in this way is to ensure that it matches the real requirements of the enterprise and also to streamline the Implementation phase of the process.

'Concept to Reality' in the real world

The first step is to take the Conceptual Warehouse and begin to decompose it from high-level requirement into actual information needs from a computer system perspective. An example of this has already been given above : if a user requires a report that will show the sales of furry widgets by zip code on a Saturday morning, then this could be changed to sales by product by geographic region by time period.

Again, just like the Conceptual Warehouse, we are just extending the metadata, but moving it from 'enterprise' terminology to 'computer system' terminology. This is an extremely important link, because if it is done correctly, then it will minimize the probability of a break down in communication between users and I.S. As with the Conceptual Warehouse, there are a variety of software tools that can be used to store this information : SAS® has number of options, from a straight forward SAS® dataset (using the broad definition), to the SAS Data Warehouse Administrator®, or, using ones imagination, an AF® object like the Organizational Chart.

The key to this part of the process is to standardize all the 'computer system' terminology. One of the major aims of this step is to ensure that in the future, as more enterprise requirements are considered, any potential overlaps with other requirements can be found, thereby reducing the amount of work that it will take to move the Implementation phase.

Standardizing terminology will allow for streamlining in the Data Warehousing Process by producing recognizable patterns that will reduce both the work needed to recognize what needs to be loaded into the Warehouse and to maximize its use once loaded.

As an aside, one of the major benefits of looking at the enterprise requirements in this way is that it allows for more structural thought from a systems standpoint. Instead of the classic case where data is loaded into the Warehouse and then front-end tools are designed or supplied to use this information (a 'data-implementation' approach), where the data and the use of the data are independent of each other, it should, if structured carefully, be possible to approach the Warehouse from an Object Oriented approach. One of the key parts of Object Oriented modeling is to look at your 'problem space'. This can be a very difficult concept to implement given the classic 'data-implementation' approach to designing a Data Warehousing process, but this is exactly what is being done in our 'concept to reality' approach. An enterprise requirement can almost be viewed as an object, and if desired, this should lead to a situation where all the benefits of Object Oriented techniques can be realized[d].

Once this step of taking each of the enterprise requirements and turning them into more generalized computer terminology is completed, then the more detailed step of taking each of these terms in turn and mapping them to data elements within a system (if possible) should be addressed.

Just to reiterate, in our example, we have an enterprise requirement : generating a report that shows the sales of a specific product (furry widgets), by zip code, by time period. We have turned this into more generalized computer terminology as follows : sales, by product, by geographic region, by time period. The next step is to take each of these four components and break them down even further : the breakdown must contain certain pieces of information including, but not necessary only, the following:

Using 'Sales' as an example:

The Concept to Reality approach to Data Warehousing will give you a working tool to not only control what is actually being loaded into your Implemented Warehouse, and for what purpose, but what is not loaded and therefore which enterprise requirements are not being addressed.

- **the source** : Accounts Receivable (note the importance of Business Rules being defined in the Conceptual Warehouse, since sales could potentially be obtained from Shipping if we define sales as shipped goods, or Billing if we have defined sales as what is billed, not collected. Note also that there is a timing issue, since the time the sales is made, billed, the goods shipped and the money collected could fall over months. It is essential to explicitly define the business rule for sales, although it could differ from enterprise requirement to enterprise requirement. This does not matter so long as we can distinguish which is which and who the owner of the rule actually is.).

- **the data element(s)** : this will include both the file and specific field within that file.
- **the level of data** : in this case, the sales amount will be obtained most probably from a line item on an invoice, so it could be collected on an invoice level (this is important, because although available on an invoice level, this might not be needed in the Implemented Data Warehouse).

Again, this information needs to be documented, since it should be possible to select any particular component of an enterprise requirement and find out each and every other requirement that component is contained within. Indeed, let's assume that at a later date, a new enterprise requirement is defined that requires a report on Salesperson profitability : one of the components of this requirement will have to be sales, and if we manage this section of the process correctly, we will know that this has already been defined as part of another enterprise requirement and is already in the Implemented Warehouse..

This process outlined above is nothing new. There are many tools (the SAS® Data Warehouse Administrator for example) that will help implement this process. What is different, however, in this approach, is that traditionally, only those elements that are going to be loaded into the Data Warehouse are documented. This concept to reality approach also includes those data elements that will be needed to fulfill an enterprise requirement, but might not yet be available, or might be so expensive to obtain, that they are not going to be loaded. This will then tell you what you don't have currently loaded in the Warehouse, but that you need, if a particular enterprise requirement is deemed important enough to address.

This leads to the final part of the 'Concept to Reality' Phase : this is in many ways the most important since it will determine what is actually loaded into the Implemented Warehouse. Just to return to basics, the aim of the Data Warehousing Process is give the enterprise a tool to control its own future. It therefore follows that those enterprise requirements that are going to meet the above goal are the ones that should receive precedence. This should be determined by the priority the particular enterprise requirement was given during the building of the Conceptual Warehouse. If only life were that simple! Deciding whether are particular enterprise requirement should be addressed within the Implemented Warehouse is also a factor of the mechanics of obtaining the information components that make up that requirement.

This decision is sometimes very easy to decide. A given enterprise only has so many resources and these should be used in the most effective manner possible to maximum benefit. Common sense comes into play as much as any complex algorithm, but the 'yes or no' decision is nowhere as difficult to manage as the expectations of the users. Although a particular requirement is often currently (a Data Warehouse process is dynamic) excluded from inclusion in the Implemented Data Warehouse on technical grounds (the Accounts Receivable system is still on an MVS system to which we have no link), the decision should be a joint effort between the user and the technical staff. If the users are included in this decision, then they will feel part of the Data Warehousing process and are more likely to remain in support. They will understand what the components of their enterprise requirements are and why at the current time it is not being incorporated in the Implemented Data Warehouse. There is nothing worse than a Data Warehouse process that has no end-user support.

Once these decisions have been made, then this phase of the Data Warehousing process is complete. As we all know, a process is never stagnant, so each and every enterprise requirement will be revisited (based upon new information or a standard review process to find out if the enterprise requirements are still valid or need updates) and in this way, the Implemented Warehouse will remain in line with enterprise requirements.

There are also other benefits that are associated with the 'concept to reality' phase of the Data Warehousing Process. These include the following:

- The structure of Data Marts will already, to a large degree, be formulated. Enterprise requirements will make it apparent the way that different users (departments?) will want the information summarized. On the same grounds, if a Multi-Dimensional structure is required, these will be largely defined already.
- It forces that the end-user community and the technical community can discuss, in terms that **both can understand**, the needs of the enterprise. This leads to less misunderstanding and therefore more realistic user expectations.
- All components of the Implemented Data Warehouse can be tied back to one or many specific enterprise requirements. This leads to

less redundancy and a more efficient data model (since the data can be modeled exactly to fit the need).

CONCLUSION

At this point, the implementation stage of the Data Warehouse process can proceed. Exactly what needs to be loaded, the business rules associated with the data, what the information will be needed for and a myriad of other information will already be known. There are, of course, many other decisions still to be made : the structure of the data itself, the tools the end-users will be given, what should be the client/server configuration (if any). These issues are usually the ones that most architects of Data Warehouse processes begin with and this is one of the major reasons that many projects fail.

The Data Warehousing process has been misunderstood due to the 'implementation centric' approach that has been foisted upon it by designers ill prepared to move away from Operational Systems. The Data Warehouse process must reflect the enterprise requirements designed to give the organization maximum control over its future. The physical implementation of this process should not take place without any checks and balances against the enterprise requirements. This is exactly what the 'concept to reality' approach discussed in this paper gives the enterprise.

---

[a] See Dan Lutter (Hewlett Packard Company, Cary, NC) : SAS® Data Warehousing : Open Designs for Enterprise Environments presented at the 1996 Midwest SAS Users Group Meeting.

[b] See Ralph Kimball's seminal work : The Data Warehousing Toolkit (John Wiley & Sons, Inc.), Chapter 1 for an excellent overview of the differences between On-line Transaction Processing (OLTP) systems and how they differ to Data Warehousing.

[c] See b. The Data Warehouse Toolkit is based upon based upon the design of Data Warehouses using a Star Join Schema.

[d] See Andy Norton : Object Interfaces, or Amy Turske-McNee : The Evolutionary Data Warehouse - An Object Oriented Approach in SAS® Users Group International Conference Proceedings, 1997 (SUGI 22). For those very interested in how to use SAS as a true Object Oriented tool, see anything Andy Norton has written on the subject.

TRADEMARKS

SAS is a registered trademark of SAS Institute Inc. in the U.S.A and other countries. ® indicates USA registration..

Other brand and product names are registered trademarks or trademarks of their respective companies.

AUTHOR INFORMATION

Peter Welbrock can be contacted on e-mail at :

peter_welbrock@juno.com