恩門 BILLINMON.COM

Building the Data Warehouse: Getting Started

BY

W. H. Inmon

"The journey of a thousand miles begins with a single step"
old Chinese proverb

"Perfection is spelled P-A-R-A-L-Y-S-I-S"
Winston Churchill

In any large undertaking, getting off the mark to a good start is the first step toward success. And getting off to a good start is certainly at the heart of the success of the data warehouse environment. Nothing delays success and destroys momentum in the data warehouse environment more than undue deliberation at the outset. The hallmark of success in the data warehouse environment is iterative analysis, development, and processing, and it is impossible to do iterative processing without first starting.

The entire mindset of the classical operational system and application developer is dominated by the need for perfection of design and the need to gather ALL requirements before design and development ensues. Such an attitude may well be proper for the operational application environment where processes are run repetitively, and where requirements can be divined before a system is built. But the data warehouse environment is one where many requirements CANNOT be discerned until the data warehouse is built and the data in the warehouse is available for analysis. Simply stated, only SOME of the informational requirements of a corporation can be known before the data warehouse is built.

For this reason, the data warehouse is built iteratively, in small, fast bursts of development. And the first small, fast burst of development is not achieved by massively studying and gathering large amounts of requirements before the first of development starts. It is imperative that the data warehouse developer have the attitude of moving quickly through the steps of design and development, even though it is known that the processing and data requirements are not complete at the outset of development.

The data warehouse environment is built in an entirely different way than the classical operational environment has been built. The classical operational environment has been built on the system development life cycle - the SDLC - that requires that requirements be identified, that analysis proceed, followed by design and programming. Next testing occurs, and is completed by implementation. The SDLC is fine where requirements can be known in advance. But the DSS analyst - who is the ultimate beneficiary of the data warehouse - does not and cannot know his/her requirements. DSS analysts' requirements are discovered in a trial and error manner - in a mode of discovery. There is an entirely different mode of development that is required for data warehouse development.

PREREQUISITES - THE DATA MODEL

The starting point for the design and development of the data warehouse environment is the data model. Without the data model, it is difficult to contemplate building the data warehouse. The data model acts as the roadmap for development.

The data warehouse data model is built from the corporate data model. The Tech Topic - Creating The Data Warehouse Data Model From The Corporate Data Model - describes this activity in depth and will not be repeated here. In addition to the corporate data model being factored into the data warehouse data model, the known and obvious informational requirements of the organization are also factored into the data warehouse data model. Usually these requirements are gathered by means of a "time box". A time box is a limitation of time - from one week to six months - predetermined such that all known and obvious informational requirements are gathered in that period of time. Once the deadline occurs, the informational requirements are entered into the data warehouse data model.

The reason why time boxing is important is that informational requirements gathering can continue for as much time as is allowed. If there is no finite limit to the amount of time that is allowed, then the informational requirements gathering process continues indefinitely and the first iteration of the data warehouse never gets built.

The data warehouse data model consists of at least two major components - a high level model and a mid level model. Figure 1 graphically depicts the data warehouse data model.

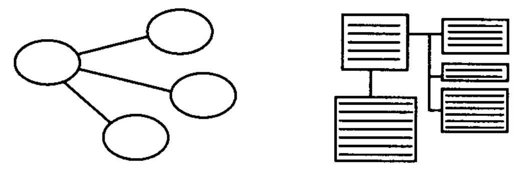
Figure 1: The data model is done.

The high-level data model consists of the major subject areas (or "entities") of the corporation. In addition the major relationships between the subject areas are also included. The high-level data model is complete prior to the first iteration of data warehouse development. The task of completing the high-level data model is not a difficult or time consuming task. The mid level data model data model consists of keys and attributes, and their grouping together. The mid level data model needs to be complete only insofar as the mid level model must exist for those portions of the data warehouse that will be built for the first iteration of design.

PREREQUISITES - TECHNOLOGY SELECTION

In order to build the first iteration of warehouse development it is necessary to have selected the technology on which to build the data warehouse. Figure 2 shows that technology must have been selected.

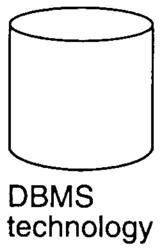

Figure 2: Both hardware and software DBMS technology is selected.

The selection of data warehouse technology - both hardware and software - depends on many factors, such as:

- the volume of data to be accommodated,
- the speed with which data is needed,
- the history of the organization,
- which level of data is being built,
- how many users there will be,
- what kind of analysis is to be performed,
- the cost of technology, etc.

Both hardware and software must be selected. The hardware is typically mainframe, parallel, or client/server hardware. The software that must be selected is for the basic data base manipulation of the data as it resides on the hardware. Typically the software is either full function dbms or specialized data base software that has been optimized for the data warehouse.

Other software that needs to be considered is the interface software that provides transformation and metadata capability such as PRISM Solutions Warehouse Manager.

A final piece of software that is important is the software needed for changed data capture, such as that provided by PRISM Solutions.

PREREQUISITES - SIZING THE DATA WAREHOUSE

Another prerequisite to the first iteration of data warehouse design and population being performed is that of the sizing of the data warehouse, as depicted by Figure 3.

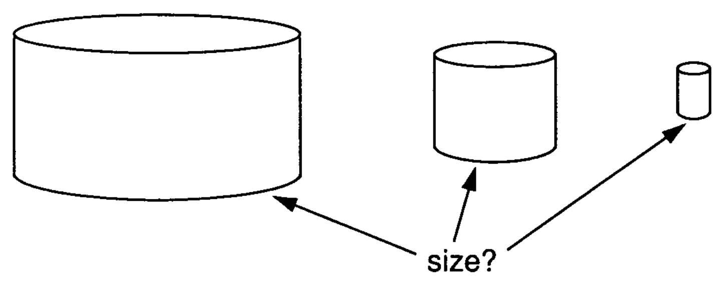

Figure 3: The rough dimensions of the data warehouse are sized.

Figure 3 shows that a rough sizing of data needs to be done to determine the fitness of the hardware and software platforms. If the hardware and software platforms are either much too big or are much too little for the amount of data that will reside in the data warehouse, then no iterative development should occur until the fit is properly made.

If the hardware and dbms software are much too large for the data warehouse, the costs of building and running the data warehouse will be exorbitant. Even though performance will be no problem, development and operational costs and finances will be a problem. Conversely, if the hardware and dbms software are much too small for the size of the data warehouse, then performance of operations and the ultimate end user satisfaction with the data warehouse will suffer. At the outset it is important that there be a comfortable fit between the data warehouse and the hardware and dbms software that will house and manipulate the warehouse.

In order to determine what the fit is like, the data needs to be sized. The sizing does not need to be precise. If anything, the sizing needs to err on the size of being too large, rather than too small. But a rough sizing of the data to be housed in the data warehouse at this point can save much grief at a later point in time if in fact there is not a comfortable fit between the data warehouse data and the environment it is built in. The estimate of the data warehouse data should be in terms of order of magnitude.

Of course, one of the issues that relates to the volume of data in the data warehouse is that of the level of granularity of data. If too much data is likely to be built into the data warehouse, the level of granularity can always be adjusted so that the physical volume of data is reduced.

PREREQUISITES - COLLECTING INFORMATIONAL REQUIREMENTS

The informational requirements of the organization need to be collected by means of a time box, as previously discussed. Figure 4 shows the typical means by which those informational requirements are identified and collected.

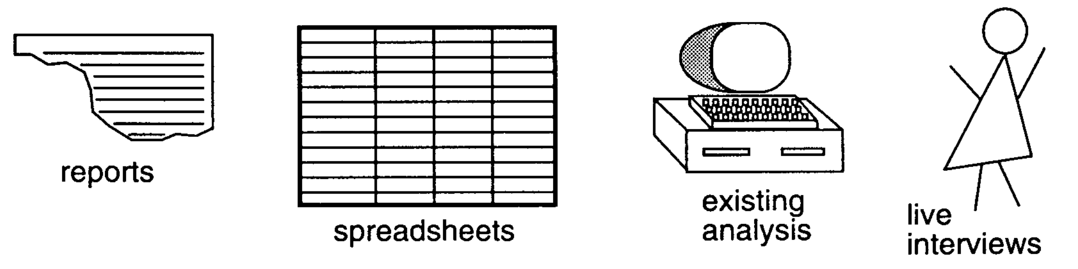

Figure 4: Existing DSS analysis is collected.

Typically informational requirements are collected by looking at:

- reports. Existing reports can usually be gathered quickly and inexpensively. In most cases the information displayed on these reports is easily discerned. But old reports represent yesterday's requirements and the underlying calculation of information may not be obvious ay all.
- spreadsheets. Spreadsheets are able to be easily gathered by asking the DSS analyst community. Like standard reports, the information on spreadsheets is able to be discerned easily. The problem with spreadsheets is that:
- they are very fluid. Important spreadsheets may have been created several months ago that are not now available,
- they change with no documentation whatsoever,
- they may not be able to be easily gathered unless the analyst creating them wants them to be gathered, and
- their structure and usage of data may be obtuse.
- other existing analysis. Through EIS and other channels there is usually quite a bit of other useful information analysis that has been created by the organization. This information is usually very unstructured and very informal (although in many cases it is still very valuable information.)
- live interviews. Typically through interviews or JAD sessions, the end user can tell what the informational needs of the organization are. Unfortunately JAD sessions require an enormous amount of energy to conduct and assimilate. Furthermore, the effectiveness of JAD sessions depends in no small part on the imagination and spontaneity of the end user participating in the session.

In any case, gathering the obvious and easily accessed informational needs of the organization should be done and should be factored into the data warehouse data model prior to the development of the first iteration of the data warehouse.

THE FIRST ITERATION - HOW MUCH DATA TO LOAD?

The first issue of design and planning for the first iteration of the data warehouse to be developed is exactly how much data is to be loaded and what variety of data is to be loaded. It is almost never the case that huge amounts of data are loaded as a result of the first iteration.

A general guideline for the loading of the first iteration is that:

THE FIRST ITERATION SHOULD CONTAIN DATA THAT IS LARGE ENOUGH TO BE MEANINGFUL AND SMALL ENOUGH TO BE QUICKLY DOABLE.

There are many different ways to cut down the size of data without losing its effectiveness. Figure 5 illustrates a few of those ways.

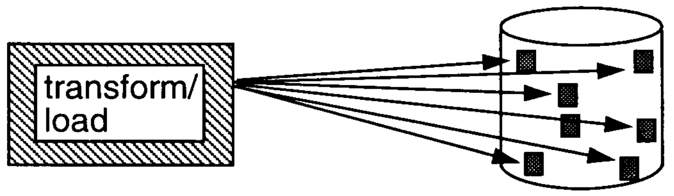

Figure 5: How much of the data warehouse is to be loaded?
— by unit of time?
— by geographical area?
— by product line?
— by customer class?
— by activity type?

There are practically speaking, an infinite number of ways to subset data. What is a meaningful subset depends entirely on who the first user of the first iteration of the data warehouse will be. The data architect needs to ask the question - who will be the first user of the first iteration of data. Now, knowing whom the first user is, what data would be meaningful to them?

There is a careful line to be walked here. On the one hand the data architect must take care not to put too much data in the first iteration of development. On the other hand the data architect must take care not to include so little data that the spontaneity of discovery by the DSS analyst is limited.

THE FIRST ITERATION - FISHING IN THE RIGHT POND

There are many functional arenas that informational processing may bear fruit in. But there are some classical functional areas that over the years have borne more fruit than others. Figure 6 depicts those functional arenas.

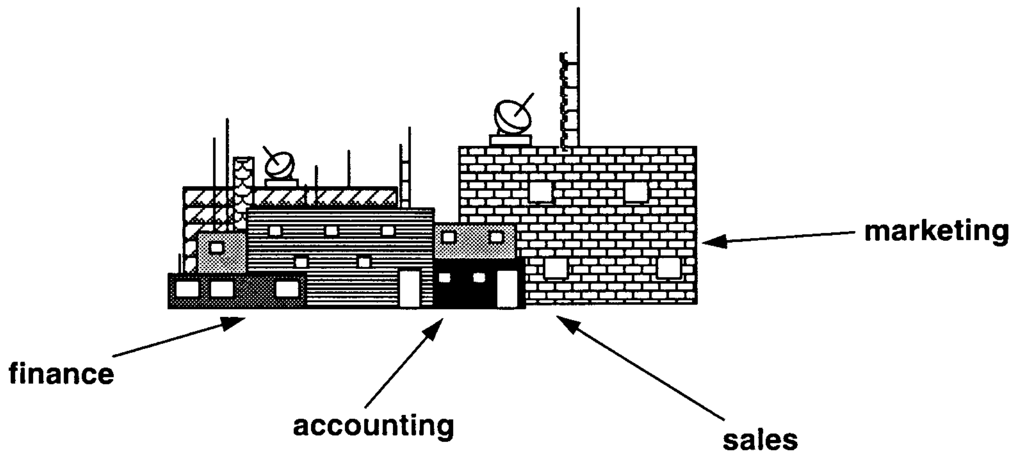

Figure 6: What functional area will be the first for the first iteration of the data warehouse?

The classical functional arenas that data warehousing has proven to be effective in are:

- finance. Finance data tends to speak directly to the executive. It tends to get right to the heart of the corporation. In addition, finance data is relatively smaller, in terms of volume, than other types of data for most corporations. For these reasons, finance is a popular arena in which to build the data warehouse,
- accounting. Accounting is similar in most regards to finance data,
- sales. Sales data is always important to the corporation. Sales data is like finance data in that it typically is compact, compared to other data. (Of course, for some industries this is not the case,)
- marketing data. Marketing and sales data are closely related and both are central to the health and success of the corporation.

These then are the classical arenas where data warehouse has found fruitful ground. But, depending on the industry or the circumstances of the company there are some other fruitful grounds as well. Some of these other frequently fruitful grounds include:

- actuarial processing (for the insurance and insurance related industries),
- process control (for the manufacturing industry),
- human resources, and so forth.

THE FIRST ITERATION - SELECTING A SUBJECT AREA

Another important decision that needs to be made for the building of the first iteration of the data warehouse environment is that of the selection of the first subject area to be implemented. Figure 7 shows this selection.

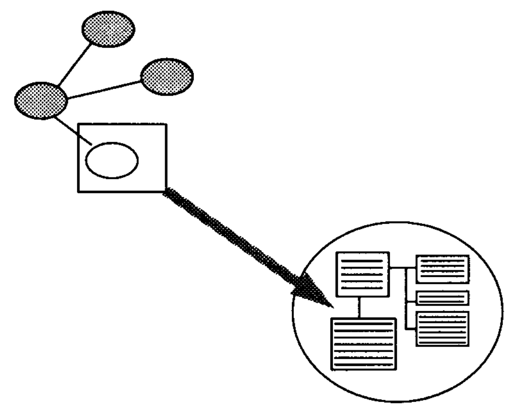

Figure 7: Selecting a subject area.

The high level data model is useful in the selection of the major subject area that will entail the first iteration of data warehouse development. The first, best option is to select an entire subject area for implementation. But on occasion the major subject area that is to be implemented is too large. So the next best option is to select a subset of a single subject area for implementation.

In some industries (such as banking and insurance) it often times is not possible to select a single subject area for first iteration implementation. In these cases it is necessary to select a series of subject for implementation. Figure 8 shows this option.

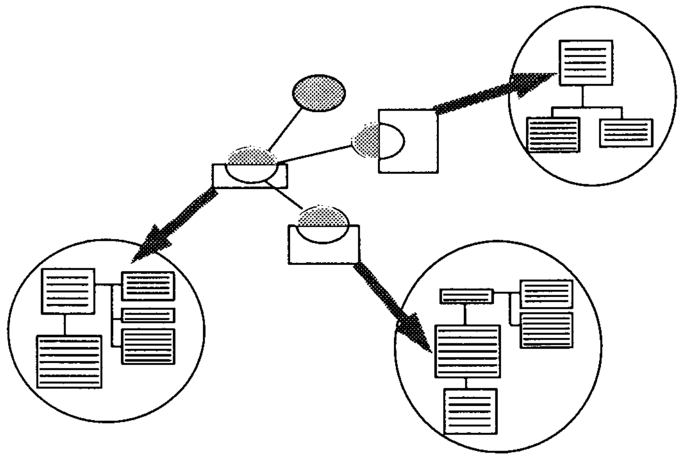

Figure 8: Choosing a subset of multiple subject areas.

When it is necessary to select multiple subject areas for first iteration implementation, only a subset of those subject areas should be implemented. The importance of the data model becomes very apparent in this case as the data model shows what can be implemented without overlap or conflict.

The different options that apply to the building of the first iteration of the data warehouse are:

- implement a single subject area,
- implement a subset of a single subject area, or
- implement a subset of multiple subject areas.

After this point the steps of design for the data warehouse will be described. It should be recognized that these steps are done for very small amounts of data - there is nothing sweeping or grand in design that is implied here. Typically there are from five to fifteen physical tables that will result from the design efforts that are being described. If there are more than that, then the whole point of doing iterative design and development has been missed.

ADDING THE PHYSICAL ATTRIBUTES

After the subject area decision is made, the next task is to go from the data warehouse data model into physical data base design. The first step in this direction is the specification of physical attributes from the data model, as shown in Figure 9.

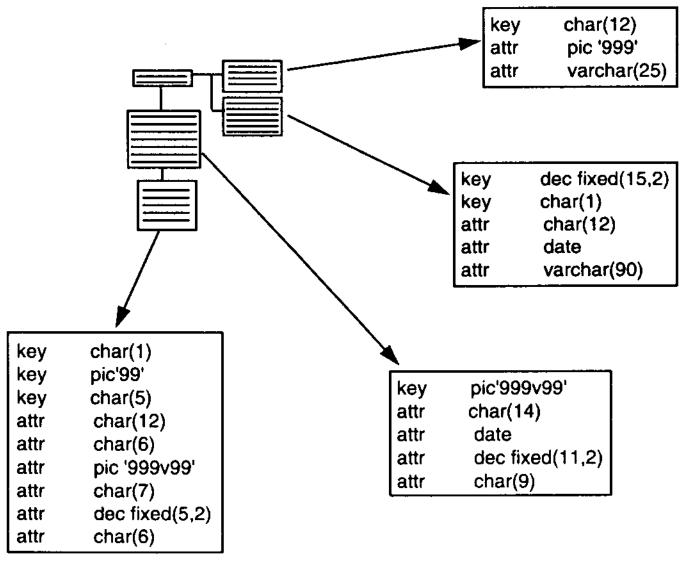

Figure 9: Next, the physical attributes are added.

The specification of physical attributes includes the task of identifying keys and attributes. In addition the physical specification for each key and each attribute is created. At first glance the physical specification of the keys and attributes appears to be an almost mechanical task; it is not.

There is much consideration of:

- the end users understanding of the data,
- the format of the data as it resides in the legacy systems environment,
- the different legacy systems formats that must be consolidated,
- the volume of data that is to be created, and so forth.

After the physical characteristics of the data that will reside in the data warehouse is determined, the next step is the identification of the unit of time that resides in each unit of data warehouse data. In most cases the unit of time - year, quarter, month, day, etc. - is appended to the key of the table. In a few cases, such as where full table snapshots of data are taken monthly, the unit of time that identifies each occurrence of data warehouse data is implicitly identified with the data warehouse.

Figure 10 shows that after the identification of the physical characteristics of data comes the specification of the unit of time for each data warehouse table.

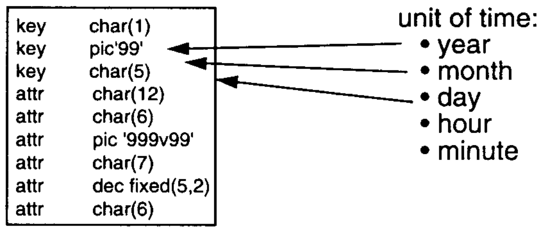

Figure 10: Each key structure will have some unit of time in the table.

There is one consideration of design that is especially relevant and important here and that is that the unit of time selected greatly influences the number of occurrences of data. Selecting the unit of time of day implies that there will be many more records in the data warehouse environment than if the unit of time of week were chosen, for example. The design of the granularity of data is directly related to the specification of the unit of time that resides in the data warehouse occurrences of data.

Once the unit of time has been specified, the next step in the creation of the first iteration of the development of the data warehouse is to identify the system of record. Figure 11 shows the identification of the system of record.

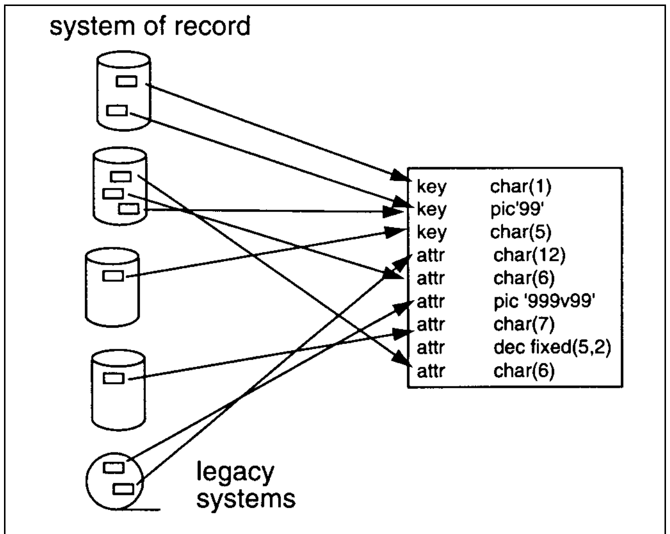

Figure 11: Defining the system of record.

Figure 11 shows that the system of record is the source data that resides in the operational, legacy systems environment. The system of record is the data that is the:

- most complete,
- most accurate,
- most timely,
- has the best structural conformance to the data model, and
- is nearest to the point of operational entry.

The identification of the system of record is the most complex activity in the building of the first iteration of data warehouse development. There are MANY circumstances that must be accounted for in the specification of the system of record:

- multiple sources of data,
- no sources of data under certain cases,
- reformatting of data,
- summarization of data,
- conversion of data,
- recalculation of data,
- alteration of key structure,
- restructuring of data attributes,
- conditional merging of data, and so forth.

The data architect attempts to identify the system of record as accurately as possible. But it is understood that if the system of record is incorrectly specified, that after the first iteration of development is complete, that adjustments can (and will!) be made.

One of the aspects of specifying the system of record is that recognizing that there are (at least!) two types of source of data - normal operational data bases and delta data (i.e., "changed data capture"). Figure 12 illustrates these choices.

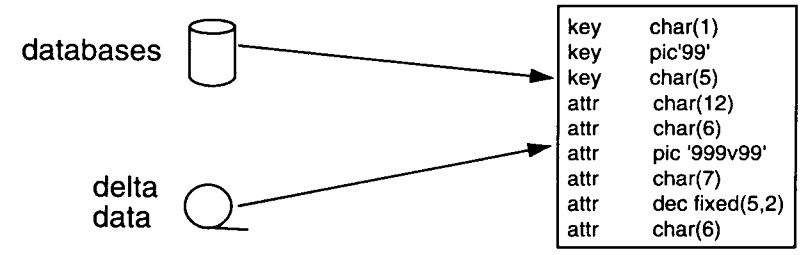

Figure 12: The system of record may be housed in actual databases or in delta data (changed data capture).

Delta data is data that has been gleaned from the updates that have occurred to a file since the last time a refreshment of the data warehouse was accomplished. Delta data is used where there are very large files that undergo very little change from one refreshment of the data warehouse to the next. Customer files are typically in this category.

Once the transformations have been specified, they are transferred to the metadata infrastructure that sits above the data warehouse. Figure 13 shows this transformation.

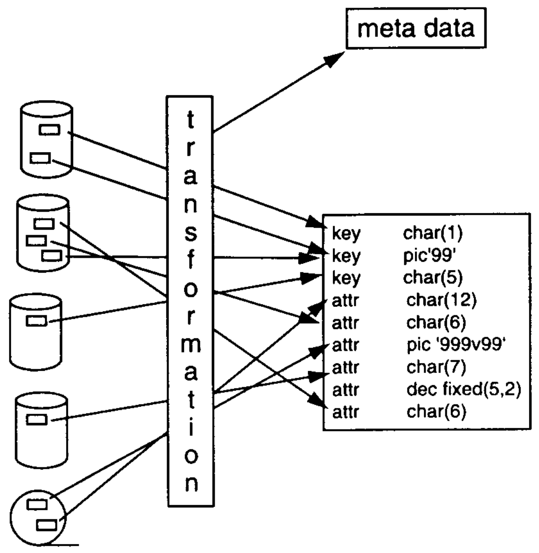

Figure 13: Moving transformation information to meta data store.

The metadata contains a record of what the source data is, what transformations and summarizations have occurred, and what the description of the data warehouse looks like. The metadata then becomes the DSS analyst guide to what is in the data warehouse and how it got there.

Subsequent to the specification of the metadata from the transformations that occur is the activity of specifying the frequency of transformation. Figure 14 shows this specification.

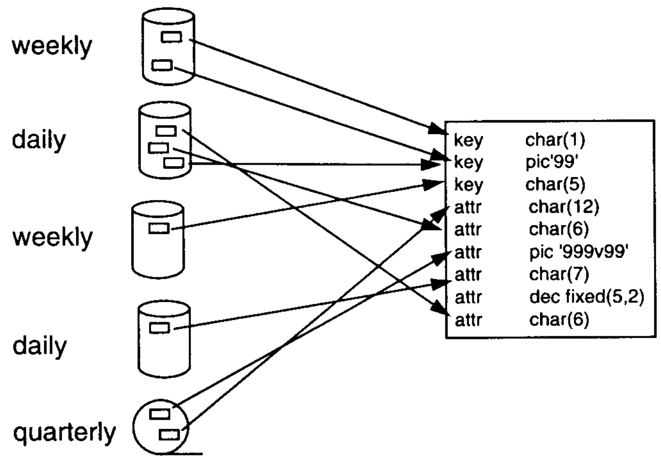

Figure 14: Defining the frequency of execution of the transformation.

The frequency of execution of transformation depends on many things, such as:

- how much data will be transformed,
- whether a delta file is being used,
- the complexity of the transformation,
- the business need to see the updated data warehouse,
- the availability of the operational data,
- how frequently system checkpoints are being done,
- the amount of summarization and aggregation that needs to be done, and so forth.

At this point the DDL that does the actual definition of the data into the dbms is executed, as seen in Figure 15.

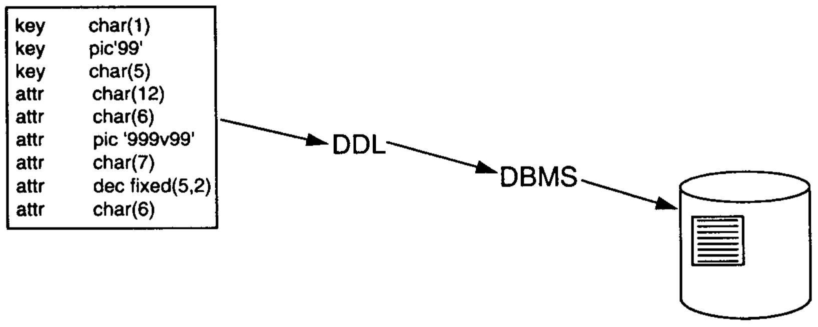

Figure 15: Defining the DDL.

The DDL of course specifies to the dbms the exact content and structure of data.

In some cases the DDL may have already been specified. If such is the case, then it is a worthwhile activity to check that the DDL being generated by the first iteration of processing is identical to the DDL that has already been produced. Figure 16 illustrates this simple but very important check.

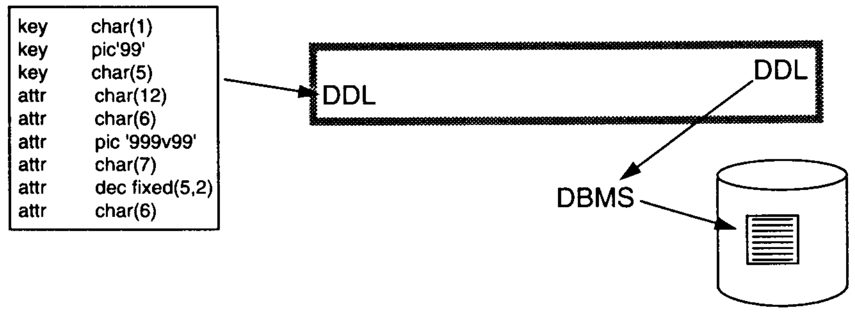

Figure 16: Verify the DDL if the DDL has been previously created.

If the DDL that the data architect has produced somehow does not match the DDL that has already been defined to the dbms, then there MUST BE a reconciliation before any other design and development ensues.

At this point in time, the transformations can be turned into code. Having an automatic code generator such as PRISM Solutions Warehouse Manager turns this task into an automatic and efficient exercise. Without PRISM Solutions Warehouse Manager this activity is done manually and is anything but fast and efficient. Furthermore, given that iterative development is the norm for data warehouse development, the automated approach means that future maintenance will be done automatically.

One of the aspects of automated code generation is that of automated gathering of legacy systems metadata. When legacy systems metadata can be gathered in an automated manner, the process of defining the metadata to the transformation program is streamlined, and the chance for error is greatly reduced than if the metadata were captured in a manual fashion.

Figure 17 shows the turning of transformation into code.

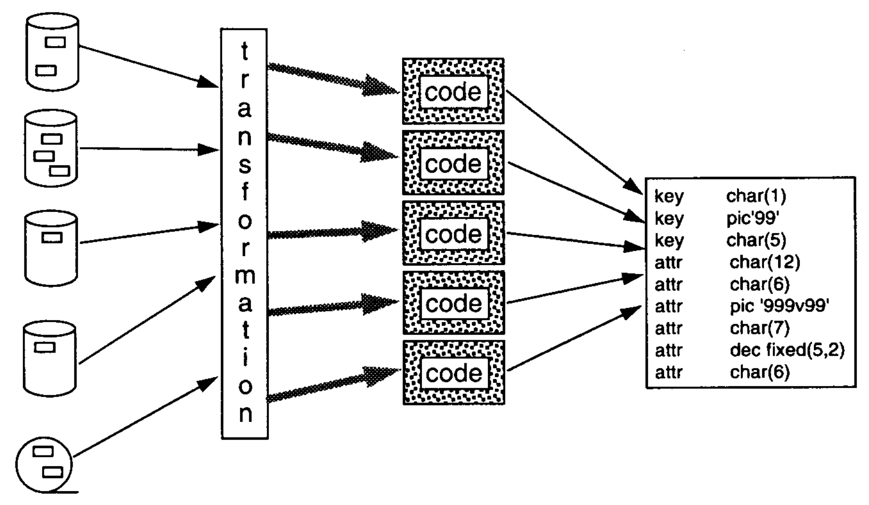

Figure 17: The transformation is turned into code.

Concurrent with the production of code is the task of allocating space for the data warehouse to reside on. Figure 18 depicts this task.

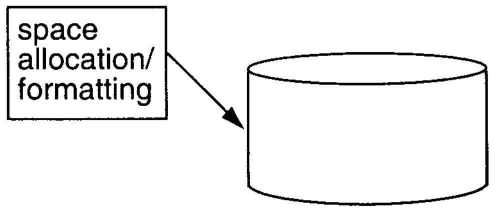

Figure 18: Allocating, formatting space.

In some technologies the allocation of space is very straightforward. But in other cases there are the considerations of secondary extents and the mapping of data across the parallel environment. When secondary extents will be allocated, care should be taken so that:

- the secondary allocation will not occur until well into the data warehouse development and population life cycle, and
- when secondary allocation occurs that the system programmer is alerted.

When mapping data to the parallel environment, care should be taken so that as much growth as can be anticipated can be accommodated. Trying to reorganize data after it has been physically loaded onto the parallel environment and mapped there is not a pleasant proposition and should be minimized as much as possible.

Once the allocation of space has been made, the programs that have been created can be put into execution and the data populated into the data warehouse. Figure 19 shows the population of data into the data warehouse.

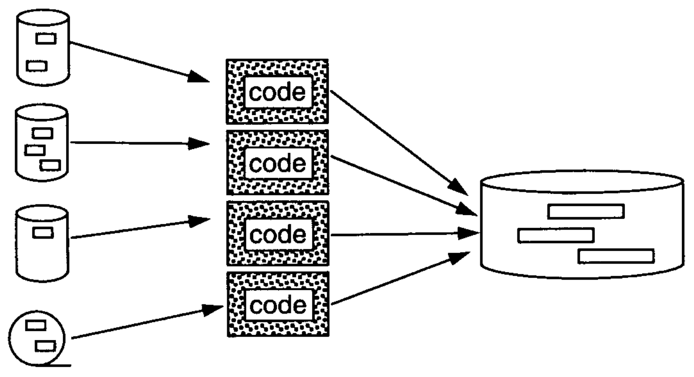

Figure 19: Actual loading of data into the data warehouse.

Once the population of the data warehouse occurs, the end user - the DSS analyst - is exposed to the data. Figure 20 illustrates the DSS analyst being allowed to access the data warehouse.

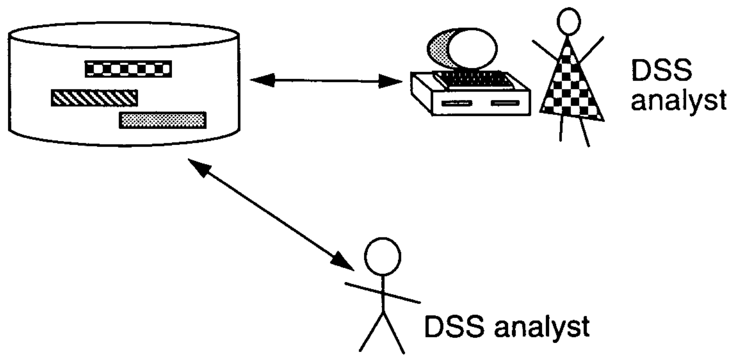

Figure 20: Turning the DSS analyst loose onto the data warehouse.

Of course the end user needs tools for the access of data. The access of data may be direct or may be indirect. In any case, the end user should be encouraged to explore the possibilities of the access and manipulation of data in the data warehouse.

At this point the first iteration of development of the data warehouse ends. While there have been many activities to do, there has been relatively little data and few types of data to do the activities on. Furthermore, the design and development activities have been "time boxed" so that progress is made, even if the progress is not entirely complete. And while every effort has been made to create a good and serviceable design, there is no intention of completeness or perfection. It is understood that errors in design will occur and that those errors will be corrected in the second iteration (and beyond) of data warehouse design and development.

THE SECOND ITERATION

The seeds of the second iteration of data warehouse development are sowed in the completion of the first phase of development. An important and essential aspect of the development life cycle is the monitoring of the end user for feedback. Figure 21 shows this feedback loop.

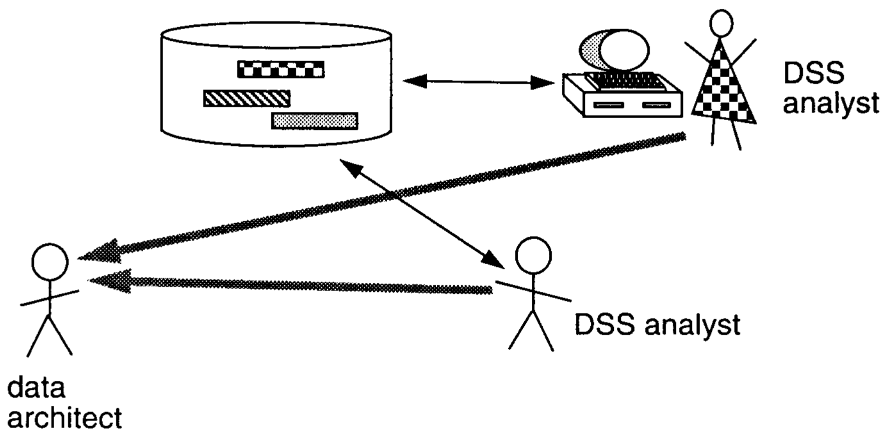

Figure 21: Carefully listening for feedback.

The data architect listens very carefully to the DSS analyst to determine where improvements to the data warehouse can be made. While the conversation between the data architect and the DSS analyst is an ongoing one, the conversation that occurs as the data warehouse is first populated is probably the most important. It is in the first few uses of the data warehouse that the most poignant and the most valuable feedback typically occurs.

SUMMARY

The data warehouse is built under a very different life cycle than the classical operational environment. The data warehouse is built iteratively - quickly and in small iterations.

The steps of development include such activities as:

- data modelling,
- selecting hardware and software platforms,
- sizing the data warehouse,
- collecting obvious and known informational requirements,
- determining meaningful subsets of data for initial loading into the data warehouse,
- selecting the most fruitful functional area for population,
- selecting a subject area for design,
- identifying physical characteristics of the subject area,
- identifying the unit of time for the data,
- identifying the system of record,

- determining whether delta data should be used,
- loading transformation data to metadata,
- specifying the frequency of transformation,
- executing DDL,
- creating code that embodies the transformations,
- allocating space,
- population of data into the warehouse, and
- turning the end user loose on the data.