The COSMIC Functional Size Measurement Method

Version 3.0

Guideline for sizing Data Warehouse Application Software

Version 1.0, May 2009

Acknowledgements

This COSMIC Guideline is derived from a paper originally presented at the SMEF Conference in May 2006 in Rome, Italy by Harold van Heeringen of Sogeti Netherlands [1]. The paper was entitled 'Measuring the functional size of a data warehouse application using the COSMIC-FFP method. That paper was written in response to a paper by Luca Santillo [2] who suggested that the COSMIC method would provide a better way of measuring a functional size of data warehouse software than earlier '1st Generation' Functional Size Measurement methods.

Reviewers (alphabetical order)		
Harold van Heeringen, Sogeti Netherlands*	Peter Fagg, Pentad Ltd., UK	Arlan Lesterhuis, Sogeti, Netherlands*
Marie O'Neill, Software Management Methods, Ireland *	Luca Santillo, Agile Metrics, Italy*	Charles Symons, UK*
Frank Vogelezang, Sogeti Netherlands		

* Authors and co-editors of this Guideline

Version Control

The following table summarizes the changes to this 'Data Warehouse Guideline'.

DATE	REVIEWER(S)	Modifications / Additions
May 2009	COSMIC Measurement Practices Committee	First public version 1.0

TABLE OF CONTENTS

1

INTRODUCTION

Nowadays, more and more organizations hope to create a sustainable competitive advantage by creating a system with which they can assess the current status of their operations at any moment, and with which they can analyze trends and connections using truly up-to-date data. Therefore, one of the current trends in the ICT market is the growing interest in the development of large data warehouses.

A data warehouse system is a special type of business application software system designed to hold and/or to present both detailed and summarized data to support business analysis and decision making. However, experience shows that data warehouse projects tend to fail even more often than 'normal' ICT projects. (A search through the ISBSG data [3] shows only two data warehouse system projects.). This failure rate may be due to the fact that traditionally, it has been very hard to estimate the effort required to build a data warehouse system.

The starting point for making an estimate of the effort to develop a data warehouse system is to estimate its functional size. But certain'1st Generation' Functional Size Measurement (FSM) methods are difficult to apply to size a data warehouse because the latter typically have a complex architecture and data structures, and require large, complex transactions to load and retrieve the data. In the literature, a number of ways are proposed to measure data warehouse systems using '1st Generation' FSM methods.

This Guideline:

- introduces a model of a comprehensive data warehouse system, and defines its various components;
- shows how the functional size of a data warehouse system may be measured using the COSMIC FSM method.

COSMIC Guidelines aim to supplement the method's standard documentation by examining specific types of software to demonstrate how they can be sized using the COSMIC method. The reader is assumed to have mastered the basic concepts of the COSMIC method as defined in the 'Measurement Manual' [4] and to be familiar with the 'Guideline for Sizing Business Application Software' [5].

2

WHAT IS A DATA WAREHOUSE?

Today, databases are the engines in almost every data-driven organization. Over the years, databases have been optimized to support the operational business processes within these organizations. However, as the number of different databases increases within an organization, it becomes more and more difficult to extract meaningful data from across these databases. This arises because the data in different databases in various parts of an organization have often not been defined and named consistently. Standard database management systems are generally not equipped to merge data so as to produce reports from multiple databases.

At the same time, more and more organizations want to exploit 'decision support systems' so that they can make sound decisions based on the data within the organization. Implementing a good decision support system starts with the clarification of the most important aspects of the operational business of an organization. Information about these aspects must be gathered in a full and consistent way. Then it is possible to generate reports upon this data.

A data warehouse system supports all of the above. Bill Inmon, a pioneer in this field, defined a data warehouse to be: "a subject-oriented, integrated, time-variant and non-volatile[1] collection of data in support of management's decision making process" [6]. Another, commonly-used definition is: "a collection of decision support technologies, aimed at enabling the knowledge worker to make better and faster decisions" [7].

A data warehouse system loads data from various operational databases, cleans this data, transforms the data attributes to a commonly defined attribute and then stores the data in the data warehouse. In this process, a number of aggregations and calculations with the data are performed to speed up the performance of the warehouse for its users. The rules that must be used to clean and transform the data and to perform the calculations are part of the 'metadata' management system of the application. Users can extract the data from the data warehouse using, for instance, query tools. The components of a typical data warehouse system are shown in Figure 1.

There is a large number of differences between traditional transaction-oriented business application systems, which involve a lot of user interaction with the database (sometimes referred to as CRUD functionality, where 'CRUD' stands for Create, Read, Update, Delete)), and data warehouses. The main differences are shown in Table 1.

[1] Recently, so-called real-time data warehouses have been introduced in which the warehouse is updated in real-time whenever the operational data change. Their aim is to enable decision-making based on the latest available data. The model assumed here is of a data warehouse which is loaded by periodic batch updates and which is available for on-line enquiries. From a functional size measurement viewpoint, the principles are the same, regardless of the frequency and mode of data updating.

	Transaction Processing	Data Warehouse
Purpose	Run day-to-day operations	Information retrieval and analysis
Structure	RDBMS optimized for Transaction Processing	RDBMS optimized for Query Processing
Data Model	Normalized	Multi-dimensional
Access	SQL	SQL, plus Advanced Analytical tools.
Type of Data	Data that runs the business	Data to analyze the business
Nature of Data	Detailed	Detailed and summarized
Data Indexes	Few	Many
Data Joins	Many	Some
Duplicated Data	Normalized DBMS	Non-normalized DBMS
Derived and Aggregated Data	Rare	Common

Table 1: Differences between a transaction-processing system and a data warehouse system.

Because of the fact that '1st Generation' FSM methods were primarily developed to measure the size of transaction-processing systems, these differences (notably the multidimensionality and the multiple indexes) make it hard to apply those methods to size the data warehouse type of software.

Figure 1 shows an example of the architecture of a comprehensive data warehouse system. Not all data warehouse systems will have all these elements, nor will they always interact in the way shown here, as will be explained.

'ETL' are programs that Extract, Transform and Load data from one stage to the next. ETL programs are used to extract data from data sources, cleanse the data, perform data transformations, and load the target data warehouse and then again to load the data marts. ETL programs are also used to generate and maintain a central metadata repository.

N.B. In the terminology of the COSMIC method [4], all the programs of a data warehouse software system reside in the same application 'layer'[2] and are thus 'peers'. The ETL programs do not interact with each other directly, but via their shared databases

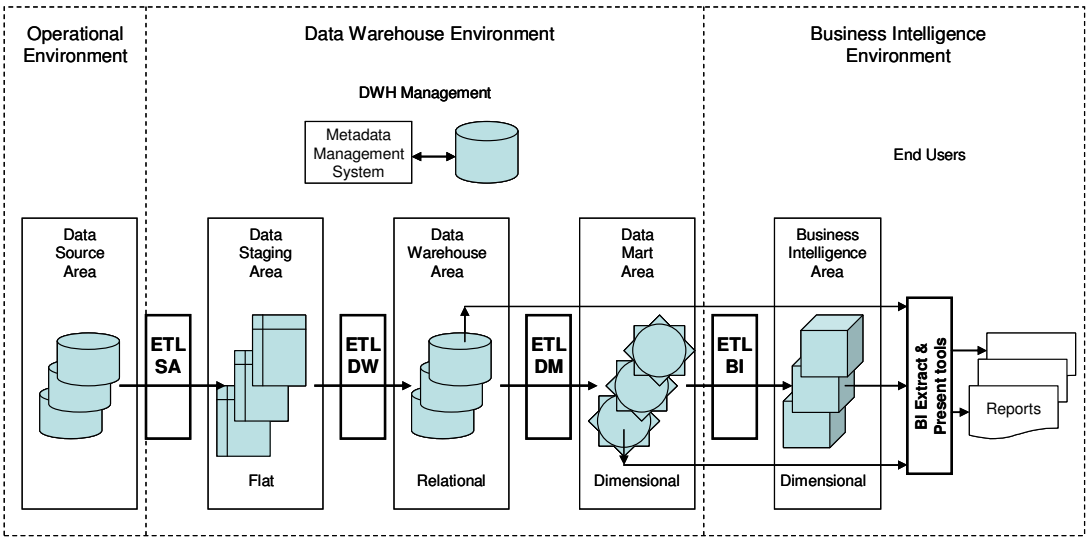

Figure 1: Visualization of a data warehouse.

[2] It is common in data warehouse terminology to describe the different components of the warehouse system as being different 'layers'. This is NOT the case in COSMIC terminology.

The main data stores and software components of a data warehouse system are as follows:

Operational data sources

An operational data source is the database of an operational system in which transactions of the organization are captured. The source systems and their databases should be thought of as being outside the data warehouse, since the data warehouse system has no control over the content and format of the data. The data in these systems can be in many formats from flat files to hierarchical and relational databases.

Data staging area

The data staging area is the portion of the data warehouse that (we assume) is loaded by 'ETL SA' tools. These extract, clean, match and load data from multiple operational source systems into (usually) flat files. (An alternative that we have not assumed here but which would make very little difference to what follows is that applications of the operational environment send data to ETL-SA tools.)

The data staging area is explicitly off-limits to the end users of the data warehouse; it does not support query or presentation services. One or more data-cleansing tools may be used to process data into the staging area, for example to resolve name and address misspellings and the like. They may also resolve other data cleansing issues by use of fuzzy logic.

Data warehouse database

The data warehouse database is a relational data structure that is optimized for distribution. 'ETL DW' tools collect and integrate sets of data from multiple operational systems that have been cleaned in the data staging area, and store them in the data warehouse. The latter then becomes the one source of the truth for all shared data.

Data marts

The easiest way to envisage a Data Mart (DM) is that it is an extension of the data warehouse for a specific user 'department'. 'ETL DM' tools create data marts derived from the central data warehouse source. The theory is that no matter how many data marts are created, all the data is drawn from the one and only one version of the truth, which is the data contained in the data warehouse database. Distribution of the data from the data warehouse to the data marts provides the opportunity to build new summaries containing subject-specific information to fit a particular department's need. Data marts can provide a rapid response to end-user requests if most queries are directed to pre-computed, aggregated data stored in the data mart.

Business intelligence databases and End user functionality

Using the definitions of Wikipedia, 'Business Intelligence' (or 'BI') tools 'provide historical, current, and predictive views of business operations, most often using data that has been gathered into a data warehouse or a data mart'. The term encompasses Decision Support Systems (DSS), On-Line Analytical Processing (OLAP) tools, Executive Information Systems, data mining tools, and custom-built tools that enable end-users to make standard pre-defined or ad hoc queries.

As shown in Fig. 1, BI tools that enable end-users to extract and present data may be able to access data warehouse or data mart databases directly, or they may require those data to be configured in a particular way. If the latter, a ETL-BI tool may be needed to re-configure and store the data in a Business Intelligence area, as also shown in Fig. 1.

BI databases can be visualized as cubes or, more generally as multi-dimensional. Wikipedia states that, 'databases configured for OLAP use a multi-dimensional data model, allowing for complex analytical and ad-hoc queries with a rapid execution time'.

Metadata management

'Metadata' literally means 'data about data'. This is a very important part of any data warehouse system. Metadata is not the actual data; but rather information that addresses a number of characteristics of data elements such as their names and definitions, where the data comes from, how it is collected, and how it must be transformed (if needed) before being stored in the data warehouse. Additionally, meaningful metadata identifies important relationships among data elements that are critical to using and interpreting the data available through the end user query tools. Metadata are typically maintained and are accessible via a Metadata Management System [8], as shown in Fig. 1. However, it is also possible that each ETL tool may incorporate and manage the metadata it needs for its specific purpose.

Dimensional Data Storage

Figure 1 shows how the data is transformed through the various areas of the data warehouse system. When data are loaded into the data warehouse, it is quite common to store the data in relational tables, although sometimes data in data warehouses can be stored dimensionally as well. In data marts, data is stored in a dimensional way, as also in the business intelligence area.

Dimensional data storage means the storage of all kinds of aggregations and computations in order to make the retrieval of the data faster. A common way to store the data in a dimensional way is the use of a star schema. In a star schema, a fact table forms the 'hub', and dimension tables form the 'spokes' of the schema. An example of this is presented in figure 2. Fact tables hold data about some major aspect of the business. The facts are computed and may be retrieved for any combination of the chosen values of the dimensions.

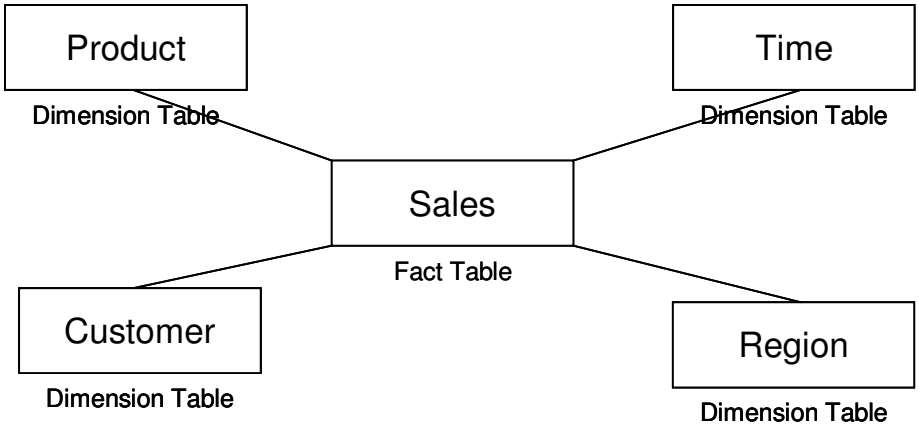

Figure 2: Star schema.

Dimension tables often consist of hierarchies. Typically, it is possible for end users to roll up a dimension, or drill down a dimension of the data. For instance, on the 'Time' dimension, days roll up to weeks, which roll up to months, which roll up to years; in the 'Product' dimension, products might roll up to product-group, to product-class and finally to 'All products' (or 'product-range').

In the example of figure 2, the 'base' subject of fact table is 'Sales'. It might have attributes such as 'Current year budgeted sales', 'Actual sales', 'Gross margin', etc.

Examples of star schemas for other businesses might be:

- For a retailer, as in figure 2, but with a 'Store' dimension and no 'Customer' dimension
- For a manufacturer, the hub might be named 'Purchasing', with dimensions for 'Supplier', 'Material', 'Factory', and 'Time'

In the business intelligence area, data that has three dimensions can also be viewed as a cube. An example is shown in figure 3. .

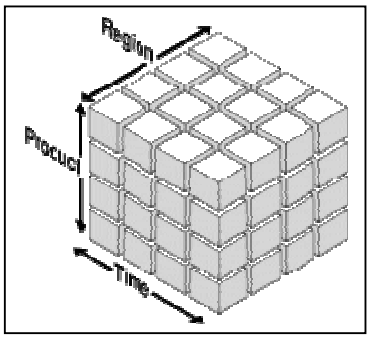

Figure 3: Cubic schema.

End users have tools at their disposal to enquire on data in these arrays at a particular point, or by a particular row or plane in any of the two, three or more dimensions, or for any part of the array, as they wish. This enquiry approach is often denoted as 'slice and dice' in data warehouse jargon.

3

SIZING A DATA WAREHOUSE SYSTEM WITH THE COSMIC METHOD

The reader is assumed to be familiar with the principles and rules of the COSMIC Method, as defined in its Measurement Manual [4].

The COSMIC method involves a three-phase process to perform a functional size measurement:

1. Measurement Strategy Phase
2. Mapping Phase
3. Measurement Phase

3.1 Phase 1: Measurement Strategy

In the Measurement Strategy phase, the purpose, scope, the functional users and the level of granularity of the analysis should be defined. We will assume that the purpose is to measure the functional size of the data warehouse software system, as input to a process for estimating development effort.

As we have already defined, all components of the data warehouse software system reside in the same application 'layer', including any software that is used to maintain the metadata.

The 'overall scope' of the measurement is assumed to be the entire data warehouse software system, as per the example architecture in Fig. 1. Each separate piece of software in the system, e.g. an ETL tool, may be considered as a (peer) 'component' of the system. It follows, given the purpose of the measurement and the fact that each component is an autonomous piece of software that may be developed separately, that we must define each component as having its own separate measurement scope.

The following components of the data warehouse software system are identified as having separate measurement scopes:

- ETL Staging area tools
- ETL Data warehouse tools
- ETL Data mart tools
- ETL Business intelligence tools
- Metadata management tool

We assume that the 'Extraction and Presentation tools' that allow end users to access data warehouse, data mart or business intelligence databases are externally-supplied software utilities that do not need to be sized, so are not within the overall scope of the measurement.

As regards the functional users of these components, since no one data warehouse system component communicates directly with any other component, no component is a functional user of any other component. (These components 'communicate' with each other via the data resources they use in common.) However, in the architecture of Fig. 1, all four ETL tools need to access the metadata management tool, so each ETL tool is a functional user of the metadata management tool, and vice versa.

Other functional users of the four ETL tools will include any DB administrators or ETL procedure administrators that control the processing of an ETL tool, or who receive output (e.g. error reports) from the tool. The functional users of the Business Intelligence tools are the 'end users' who make the enquiries.

The functional user of the metadata management tool will be a metadata management administrator.

This example COSMIC analysis will be made at the functional process level of granularity.

3.2 Phase 2: Mapping phase

The mapping phase is the most important phase in the measurement process. In this phase, the Functional User Requirements are examined and the various functional processes, objects of interest, data groups, and the data movements are identified.

3.2.1 ETL Staging Area tools

Identifying the functional processes (FP), objects of interests (OOI) and data groups

The first functional processes we encounter are those of the ETL-SA tools that move data from operational data sources into the staging area files, as shown in Figure 4.

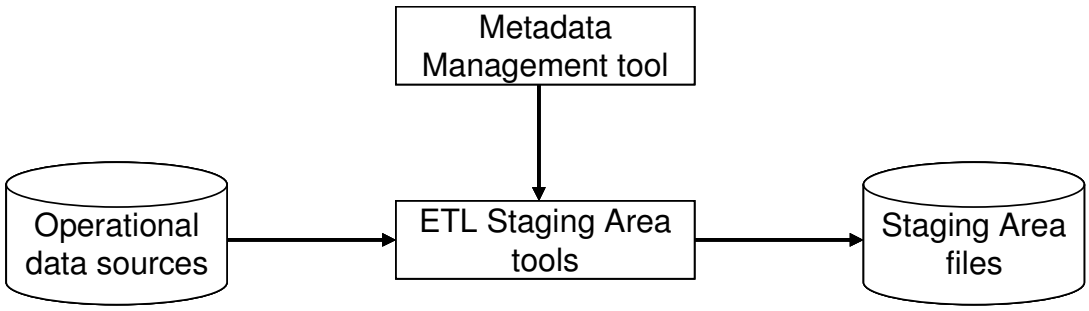

Figure 4: ETL Staging area tools.

To identify functional processes, first we should identify the objects of interest (OOI's) in the operational data sources.

A single operational data source can contain data describing numerous objects of interest. Depending on the FUR, one functional process must be identified for each of these separate OOI's, or for each 'logical grouping' of OOI's, to extract the data groups from the operational data source, to transform the data attributes as described by the metadata management system and to load the data groups into the staging area. An example of a 'logical grouping' could occur in a data source of multi-item orders where it is required to maintain the header/item relationships in the staging area. One functional process would be needed to extract, transform and load each order, comprising the header and the multiple items, that is data describing two OOI's.

Care must be taken when deciding if a 'code table' should be considered as representing an OOI or not. For a general discussion of code tables see the examples in the COSMIC 'Guideline for Sizing Business Application Software' [5]. In the context of an ETL-SA tool, where there is a requirement for a database administrator to be able to copy a code table from an operational database to the staging area by a functional process of the ETL-SA tool, the subject of the code table will be an OOI.

EXAMPLE 1: Identifying the data movements

For a simple functional process in an ETL Staging Area (SA) tool that must move data about a single OOI-type, the data movements would be typically as follows (where E = Entry, R = Read, W = Write and X = Exit)

E	to start the functional process (e.g. a clock tick, if a batch process)
X	to the metadata management tool to obtain the transformation rules for this OOI
E	from the metadata management tool with the required metadata
R	of the operational data source
W	of the transformed data to the staging area
X	error/confirmation messages

Total size: 6 CFP.

Figure 5 below shows the functional process of the ETL-SA tool and its interaction with the functional process of the Metadata Management tool in the form of a Message Sequence Diagram.

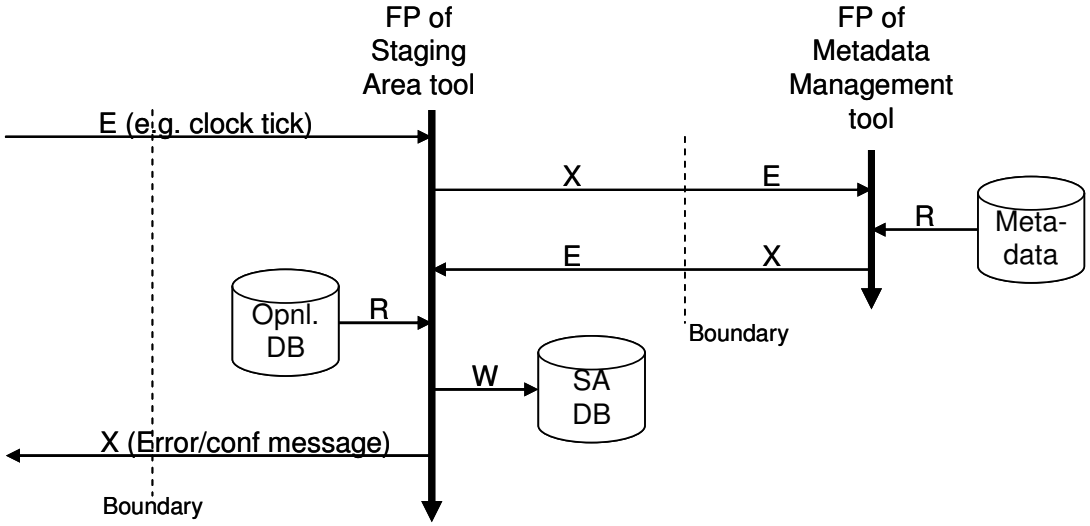

Figure 5: Interaction of the Staging Area tool and the Metadata Management tool

Some variations of this basic ETL-SA functional process are possible.

a) If the functional process of the ETL-SA tool must move a logical grouping of data about, say, two OOI's, then the above example should have one additional Read and, if the data are to maintained with the same structure in the staging area, one additional Write for the data about the additional OOI. (However, there should not need to be an additional Entry/Exit pair for an additional reference to the metadata for the transformation rules for the second OOI, assuming the second reference is a repeated occurrence of the first reference.)

b) If the data are supplied to the ETL-SA tool by an application of the operational environment (i.e. the ETL-SA tool is 'passive', waiting the arrival of operational data) then the triggering Entry of the above functional process would not be generated by a clock tick, but would come from the operational application and would include data describing one OOI that had to be loaded to the staging area. Consequently, a Read of this data would not be necessary and the total size of this functional process would be 5 CFP.

c) If the metadata needed by the ETL-SA tool is maintained and stored by this tool (rather than by a separate metadata management system as shown in Fig. 1), then the Exit/Entry pair in the above example needed to retrieve the metadata should be replaced by a single Read. This is also true for the other ETL tools that are analyzed below.

Note that the data manipulation involved in the transformation of a data group according to the metadata rules is assumed, in the COSMIC method, to be associated with the Write data movement in the above example. If the purpose of the measurement is to obtain a functional size as input to a process to estimate the amount of effort that the development of the ETL-SA software is going to take then, if the data manipulation of this transformation is particularly complex, this may need to be taken into account when deciding upon the most appropriate project delivery rate (PDR = number of hours per function point) to use in the estimating process.

3.2.2 ETL Data Warehouse tools

Identifying the functional processes (FP), objects of interests (OOI) and data groups

In the ETL data warehouse tools we find the functional processes that feed the data warehouse from the data describing the OOI's that are stored in the flat files in the staging area, as shown in Figure 6.

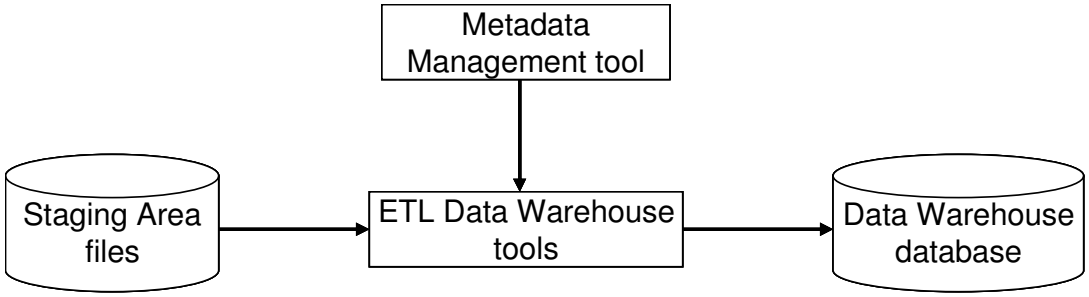

Figure 6: ETL Data Warehouse tools.

Again, for each OOI, or logical grouping of OOI's, a separate functional process should be identified, that extracts the data groups from the staging area file, transforms the data attributes according to rules obtained from the metadata management system, and then loads the transformed data into the data warehouse database.

Identifying the data movements

The functional processes of the ETL Data Warehouse tools have the same structure as those of the ETL staging area tools (assuming the data in the data warehouse database are not stored dimensionally).

EXAMPLE: A simple functional process of the ETL data warehouse tool that extracts, transforms and loads data describing a single OOI-type would have the following structure.

E	to start the functional process (e.g. a clock tick, if a batch process)
X	to the metadata management tool to obtain the transformation rules for this OOI
E	from the metadata management tool with the required metadata
R	of the staging area files
W	of the transformed data to the data warehouse database
X	error/confirmation messages

Total size: 6 CFP

The Message Sequence Diagram for the functional processes of this stage would have the same structure as that shown in Fig. 5.

3.2.3 ETL Data Mart tools

Identifying the functional processes (FP), objects of interests (OOI), data groups and data movements

In the ETL data mart tools we find the functional processes that feed the data marts from the data describing the OOI's that are stored in the data warehouse component, as shown in Figure 7.

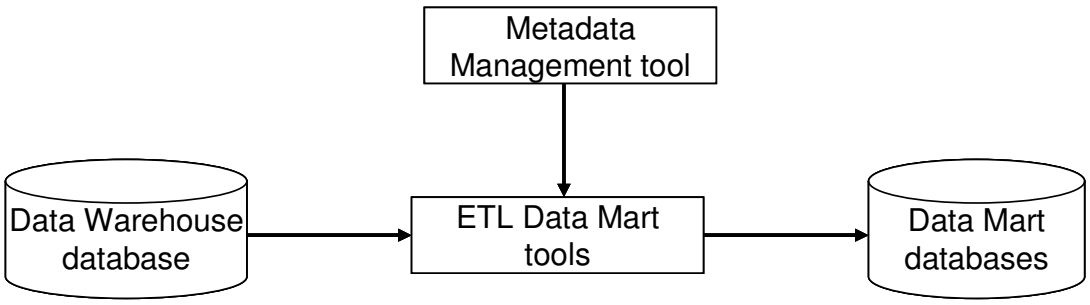

Figure 7: ETL Data Mart tools

In the data mart databases, the data will be stored in a dimensional way, thus in star schemas, as in Fig. 2, which shows both 'dimension tables' and 'fact tables'. It is important to distinguish the creation of the fact tables, i.e. the 'hub' of the star schema, from the creation of the dimension tables, i.e. the 'spokes' of the star schema.

Normally in time sequence (depending of course on the FUR), the dimension tables would be loaded first, before loading the fact tables. However, for convenience we will start by describing the loading of the fact tables, assuming that the dimension tables have already been established

Given the need to calculate the number of types of OOI's and of functional processes of the ETL data mart tools, it is easiest to illustrate the process with an example star schema, as in Figure 8.

EXAMPLE. We assume a functional user requirement for a data mart to store Sales data in fact tables at the lowest (i.e. not aggregated) level and also at each level of aggregation resulting from the various possible combinations of the dimensions. There are both 'customer' (i.e. organizational) and 'region/country' (i.e. geographical) dimensions, since we assume that a customer has multiple offices in different regions. The data warehouse owner therefore wishes to aggregate sales data along both the organizational and geographical axes independently.

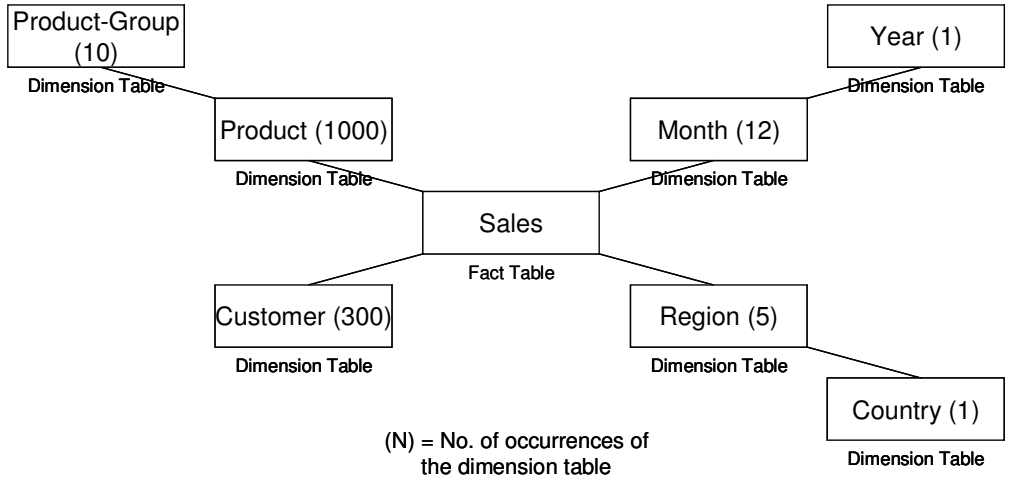

Figure. 8: Example of a star schema for a 'Sales' data mart

Let us suppose that the sales data at the lowest level (i.e. the data about the object of interest 'the business of any given customer buying a given product in any given region, in a given month') are obtained by reading all the multi-item orders that have been shipped and paid for in the given month. The attributes of this object of interest that could be created at this lowest level by reading the orders might include: 'actual sales', count of order-items', 'average price discount', etc.

We will denote the object of interest of these Sales data at this lowest level of aggregation as 'Sales (P, Cu, R, M)'. The functional process of a ETL data mart tool to create (only) this sales data fact table could have the following data movements:

E to start the functional process (perhaps including the dates that define the given month)

R of the Order-headers in the warehouse database (to obtain the Customer, etc)

R of the Order-items in the warehouse database (to obtain the product, quantity, selling price, etc)

R of the Customer file in the warehouse database (to obtain the Region from which the customer office placed the order)

W of the Sales (P, Cu, R, M) data

X error/confirmation messages

Total: 6 CFP

(We assume here that the data in the data warehouse has been fully cleaned and hence that there is no further need to reference the metadata. If this is not correct, an additional X/E pair must be added to the above functional process to obtain the metadata, as for the earlier ETL tools.)

However, we have said that in this example the Sales data mart holds fact tables for each possible combination of the various dimensions. There are, in total, eight possible combinations at which different levels of aggregated sales data may be stored for this particular data mart.

Each possible combination of the dimensions of this star schema, e.g.

- Sales of a here product to a given customer in a given region in a given month
- Sales of a given product to a given customer in the whole country in a given month
- Sales of a given product to a given customer in a given region in a given year
- Etc

could be envisaged as a separate pile of sold products; each (type of) pile of goods is a separate 'thing' in the real-world about which the users want to hold data, i.e. it is a separate OOI. Physically, each of the eight fact tables is keyed on a unique combination of the dimensions.

Hence the data mart fact tables store data about eight OOI's, which we can denote as:

Sales (P, Cu, R, M)	[the lowest level]
Sales (P, Cu, Co, M)	['Co' = Country]
Sales (P, Cu, R, Y)	['Y' = Year]
Sales (P, Cu, Co, Y)	
Sales (PG, Cu, R, M)	['PG' = product-group]
Sales (PG, Cu, Co, M)	
Sales (PG, Cu, R, Y)	
Sales (PG, Cu, Co, Y)	

So in practice, if there is one functional process of the ETL data mart tool that reads orders, etc. in the data warehouse and computes and loads sales data to fact tables at all eight levels of aggregation, then this functional process must have eight Write data movements, one for each OOI. The minimum number of data movements for such a functional process is then:

E to start the functional process (perhaps including the dates that define the given month)

R of the Order-headers in the warehouse database (to obtain the customer, etc)

R of the Order-items in the warehouse database (to obtain the product, quantity, selling price, etc)

R of the Customer file in the warehouse database (to obtain the Region from which the customer office placed the order)

Wx8 of the Sales data at all eight levels of aggregation

X error/confirmation messages

Total size: 13 CFP.

Note that the above example has been very much simplified for the purpose of this text. First, for this functional process to produce the required aggregated sales data, we have assumed that all the data needed for all 8 aggregated sales fact tables would be obtained from the input data. For example, the product-group to which a product belongs would have to be obtained for each product from the order items. Similarly, the customer master file would have to include not only the region to which the customer office belongs but also the country to which the region belongs. (It is much more likely that the product-group for any given product and the country for any given region would be obtained by Reads of pre-loaded data in the data mart.). Also, imported will be the date of each transaction, from which the month and year are computed.

Second, such a functional process as analyzed above would be extraordinarily inefficient, since the sales figures at each level of aggregation would have to be computed from the input order data. Efficient processing would require, for example, that the sales data at the product-group level would be calculated by reading the (previously computed) sales data at the product level, rather than at the order-item level. As efficient processing could be achieved in various ways, we will not explore this particular aspect of the example further. Suffice it to say that in practice efficient execution would almost certainly dictate that the processing be split into a succession of phases involving more Reads than shown above.

Furthermore, in this example, for the sake of simplicity we have not considered various business issues such as:

- whether in practice the Sales at the 'Year' level of aggregation are actually 'current year-to-date', or a 'rolling-year';
- how the ETL data mart functional process determines which order-items have been invoiced and paid for in any given month (another Read may be necessary for this);
- the likelihood in practice that as well as the actual sales data at each level of aggregation there would be other data such as the budgeted or target sales at certain levels, etc.; these data will also need functional processes to load them ;
- if this data mart is re-created from 'empty' each month, or is updated by adding new sales to the previous month's version of the data mart; if the latter, then the functional process to update the data mart would have to read the previous month's data.

[It is important not to confuse data movement types and OOI types with their respective number of occurrences of the Sales fact table records in the data mart.

EXAMPLE: Using the occurrence data given in Fig. 8 above, the potential number of occurrences of Sales records at each level of aggregation in this data mart can be computed as follows

Sales (P, Cu, R, M) 1000 x 300 x 5 x 12 = 18,000,000 records

Sales (P, Cu, Co, M) 1000 x 300 x 1 x 12 = 3,600,000

Sales (P, Cu, R, Y) 1000 x 300 x 5 x 1 = 1,500,000

Etc.]

In addition to the functional process that loads or updates the Sales data, as we have already mentioned there must be functional processes that create the dimension tables and 'load master data'. These could exist in various forms.

EXAMPLE: Suppose the Product and Product Group master data are held in the data warehouse. An ETL data mart tool functional process to create the associated dimension tables would then have the following data movements;

E	to start the functional process
R	of the Product data in the warehouse database
W	of the Product data in the data mart
R	of the Product-group data in the warehouse database
W	of the Product-group data in the data mart
X	error/confirmation messages

Total size: 6 CFP.

Then there will be functional processes to maintain these dimension tables (probably only a 'delete' functional process in addition to the create functional process) and for the end-user to enquire on the dimension tables. A functional process to delete, say, a product could be quite complex, since it will need to check if there exist any Sales data for the given product.

(Note that for the functional processes that load and maintain the dimension tables, the subject of each dimension table is also an OOI, even if it has only e.g. a code and description as attributes. For this example star schema, therefore, there are eight OOI's for the fact tables and seven OOI's for the dimension tables.)

Finally, there will be several other types of functional processes for the end-user to read the Sales data at the various levels of aggregation (see next section) and for Data Administrators to delete or to archive these records.

3.2.4 Business Intelligence tools

Identifying objects of interests (OOI), data groups and functional processes (FP)

As already described above, in the Business Intelligence (BI) area there will be 'extract and present' tools that enable the end user to enquire directly on dimensional databases in the data warehouse or data mart databases, or in specifically-configured BI databases. The latter are often required by proprietary end-user tools such as OLAP or data mining tools. In these cases, an ETL-BI tool will be needed to extract data from a data warehouse or data mart database and to transform and load it into the required structure. See Fig. 9.

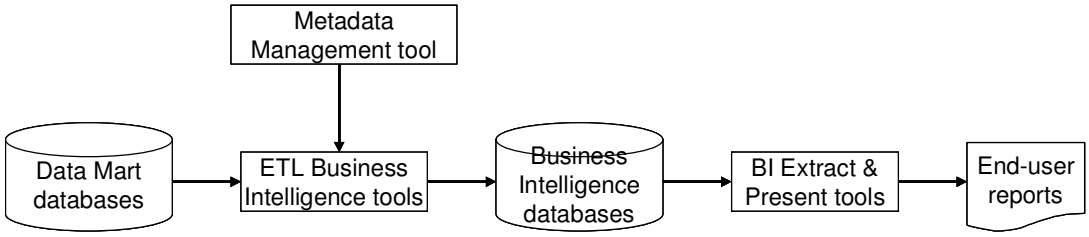

Figure 9: Functional processes within the Business Intelligence area.

The identification of the functional processes and their data movements for an ETL-BI tool involves exactly the same considerations as for the previous ETL data mart tools.

End user ‘extract and present’ functionality (‘slice and dice’ in data warehouse jargon) can be very varied. There can be all kinds of pre-defined queries, for each of which a functional process must be identified.

EXAMPLE. Suppose there is a requirement for the case of Figure 8 to develop a pre-defined enquiry to display the total sales value summed over all customers for the whole country, by product-group, for a given month. This is a level of aggregation above any of the stored data levels. This enquiry would have the following size (using the same notation as above)

E to start the enquiry, giving the month M

R S (PG, Cu, Co, M) – this is the highest level of aggregation at which data are stored

X S (PG, Co, M) – which would have 10 occurrences, one for each PG

Total size: 3 CFP.

In this example, the generation of the Exit requires that the sales data that is read be summed over all customers, for each product group.)

Potentially, the users could require an extremely large number of enquiries for all possible levels of aggregation. In practice, therefore, rather than build pre-defined enquiries, it is more likely that data mining tools will be provided that include functionality with which the user can build his or her own queries. Therefore, for measuring the work-output of the developers, measure only functional processes that have been pre-defined (using such tools or any other means) by the development team - if required to be measured by the purpose and scope of the measurement task. For instance, if the purpose of the measurement is to help estimate development effort, any user-generated queries would not be measured.

We assume here that the end user ‘extract and present’ tool itself is bought-in software, so its functional size does not need to be measured. However, if the end user tool is to be developed in-house, then it can be measured by applying the COSMIC method to the functionality of the tool (not to the functionality of enquiries that the tool can be used to generate). For this, one needs to build a model of the OOI’s and to identify the functional processes of the tool itself (i.e. NOT the OOI’s and functional processes that apply when the tool is instantiated by the end user and applied to a particular database).

If a general-purpose enquiry is provided that enables the end-user to 'drill down' (or ‘drill up’) to obtain the ‘fact’ for any combination of the 'dimensions' of a star schema, this should be analyzed as one functional process. The selection criteria types are the same for all enquiries; only the specific selection criteria occurrences, i.e. the enquiry parameters, vary from one enquiry to another. The examples in the COSMIC Business Application Guideline can clarify how to do this [5].

3.2.5 Metadata management system component

Identifying objects of interests (OOI), data groups and functional processes (FP)

Regardless of whether this is a separate component as in Fig. 1, or whether metadata are integrated with each ETL tool, the metadata administrator will have a number of functional processes at his disposal. With these he can create new metadata rules, maintain existing rules or delete metadata rules. The number of objects of interest can vary widely between different data warehouse systems. Technical metadata, like update frequency or system versioning, user profiles, access privilege files, data processing rules and use statistics can be possible OOI’s for the metadata administrator [1]. Business metadata, like the content of data dictionaries, data on historical aspects, data on a data owner, will probably be objects of interests. For each OOI, at least one functional process should be defined, but more likely there will be functional processes to create, to update, to delete and to report the OOI.

Given all these possible OOI’s, a functional process that extracts metadata on request of a data warehouse component could therefore have many more data movements than the simple ‘E, R, X’ process shown in Figure 5.

3.3 Phase 3: Measurement phase

In the measurement phase, the number of COSMIC Function Points (CFP) is calculated, by adding the number of data movements per identified functional process. Each data movement equals 1 CFP. When estimating the amount of time required for building the system, it could be advisable to consider using a different project delivery rate for the different components. For example, a distinction could be made between the amount of effort per CFP that is required to build the business intelligence component (which is primarily GUI functionality) and the amount of effort per CFP that is required to build all the database Input/Output in the other components. Of course, the effort data should also be defined and gathered in a way that this distinction can be made when analyzing the results of completed projects.

4

COMMENTS & CONCLUSIONS

The various data warehouse software components can be separately measured using the COSMIC method. The measured sizes realistically account for the often large functional processes needed to load, update or enquire on complex data structures and for the interactions with and between the various components. This should mean that the COSMIC-measured sizes will be more accurate for purposes such as performance measurement and estimating than has been achievable using '1st Generation' FSM methods.

According to van Heeringen's original paper [1], data warehouse experts seem to prefer the COSMIC method for sizing data warehouse systems, but they are concerned that the actual data manipulation involved in the transformation of data, which they claim costs most of the effort, is not taken into account directly in any functional size measurement method. There are two ways of dealing with this when the purpose of the measurement is to help estimate development effort. Either, as stated above, the assumed project delivery rate (hours per function point) may be adjusted to compensate for the effort needed to deal with data manipulation. Alternatively, the COSMIC method offers the possibility to create local extensions of the method. Van Heeringen suggested that it might be helpful to investigate in future studies the possibility of valuing the actual data manipulation sub-processes in the ETL processes, for instance by assigning functional size values to the different data manipulation sub-process types that are normally considered to be accounted for by the data movement sub-process types.

REFERENCES

[1] Van Heeringen, H., 'Measuring the functional size of a data warehouse application using the COSMIC method', Software Measurement European Forum Conference, Rome, Italy, May 2006

[2] Santillo, L., "Size & Estimation of data warehouse systems", in FESMA DASMA 2001 conference proceedings, Heidelberg (Germany), May 2001

[3] International Software Benchmarking Standards Group database, version 9.

[4] "The COSMIC functional size measurement method, version 3.0: Measurement Manual", September 2007, www.gelog.etsmtl.ca/cosmic-ffp.

[5] "The COSMIC functional size measurement method, version 3.0: Guideline for sizing business application software", Version 1.1, May 2008 www.gelog.etsmtl.ca/cosmic-ffp.

[6] Inmon, W.H., "What is a Data Warehouse?", Prism, Volume 1, Number 1, 1995.

[7] Chaudhuri, S. and Dayal, U., "An Overview of Data Warehousing and OLAP Technology", ACM Sigmod record vol. 26 (1), 1997, pp. 65-74.

[8] Sachdeva, S., 'Meta data architecture for data warehousing', DM Review Magazine, April 1998, www.dmreview.com/issues/19980401/664-1.html.