The COSMIC Functional Size Measurement Method

Version 4.0.1

Guideline for sizing Data Warehouse Application Software

Version 1.1

April 2015

Acknowledgements

Reviewers of v1.1 (alphabetical order)		
Diana Baklizky TI Metricas Brazil	Arlan Lesterhuis Netherlands*	Luca Santillo Agile Metrics Italy
Charles Symons UK*	Monica K. Villavicencio Cabezas Espol Ecuador	Frank Vogelezang Ordina Netherlands
Chris Woodward Chris Woodward Associates Ltd. United Kingdom		

* Authors and co-editors of this Guideline

This COSMIC guideline is derived from papers by Harold van Heeringen of Sogeti Netherlands [1] and by Luca Santillo of Agile Metrics [2].

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
April 2015	COSMIC Measurement Practices Committee	Public version 1.1 updated to comply with the Measurement Manual version 4.0.1. Please see Appendix B for a list of the main changes made for version 1.1 of this Guideline.

TABLE OF CONTENTS

1

INTRODUCTION

1.1 Data warehouse systems

Organizations create a sustainable competitive advantage by creating large data warehouses with which they can assess the current status of their operations at any moment, and with which they can analyze trends and connections using up-to-date data.

A data warehouse system is a special type of business application software system designed to hold and/or to present both detailed and aggregated data to support business analysis and decision making. However, experience shows that it has been very hard to estimate the effort required to build such a system.

The starting point for making an estimate of the effort to develop a data warehouse system is to estimate its functional size. As a '2nd generation' FSM method, the COSMIC method is designed entirely on basic software engineering principles and is suited to measure all types of software, including data warehouse software.

There are two prevailing views on data warehousing, the one defined by W.H. Inmon, the other by R. Kimball. These views have consequences both for size measurement and for effort estimating:

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' [3] and to be familiar with the 'Guideline for Sizing Business Application Software' [4].

This Guideline:

- introduces models of comprehensive data warehouse systems, according to the views of both Inmon and Kimball, and defines their various sub-systems;
- shows how the functional size of these data warehouse systems may be measured using the COSMIC method.

This version1.1 of this Guideline differs from v1.0 primarily by including the Kimball model and updating the text in line with versions 4.0 and 4.0.1 of the COSMIC method. A few errors in v1.0 have also been corrected. For a full list of changes from v1.0, please see Appendix B.

1.2 Terminology: Functional user requirements

For COSMIC method terminology, see the Measurement Manual [3]. For terms specific to this Guideline, see Appendix A.

The COSMIC method, like all FSM methods, aims to measure the Functional User Requirements (FUR) of software. For the definition of FUR, see the Measurement Manual.

In line with version 4.0.1 of the COSMIC method, we restrict the use of the term 'FUR' to mean functional requirements that contain all the information needed for a precise COSMIC functional size measurement.

2

WHAT IS A DATA WAREHOUSE?

2.1 An overview

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 them. 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, organizations want to exploit 'decision support systems' and 'big data' so that they can make sound decisions based on the data within the organization. Implementing a good decision support system starts with clarifying the most important aspects of the operational business of an organization and gathering data about them 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. 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' [5]. Another, commonly-used definition is: 'a collection of decision support technologies, aimed at enabling the knowledge worker to make better and faster decisions' [6].

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.

	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
Risk of duplicated data	Database is usually normalized	Database will be non-normalized if it contains aggregated data
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 monolithic transaction-processing systems, these differences (notably the multiple sub-systems of a

[1] 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 fact of the data being updated in real-time is a consequence of a 'Non-Functional Requirement'. This 'NFR' will not affect the functional size unless the NFR first affects the FUR.

data warehouse and the multidimensionality of the databases) make it hard to apply those methods to size the data warehouse type of software.

A data warehouse system loads data from various operational databases, cleans this data, transforms the data attributes each to a common definition and then stores the data in the data warehouse. In this process, some aggregated data may be formed 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 aggregations are part of the 'metadata' management system of the application. Users can extract the data from the data warehouse using, for instance, query tools.

Figure 2.1 shows an example of the architecture of a comprehensive data warehouse system of the Inmon type, Figure 2.2 shows the Kimball type.

In Inmon's view, a data warehouse should be designed to include all enterprise data and may be developed in a single project. Kimball suggests to design an overall structure of the data warehouse (commonly denoted as 'enterprise data warehouse') and then create the designed data marts in separate projects. The difference is that in a Kimball data warehouse the relational data warehouse area is absent and data from the data staging area is directly loaded into the data marts. In general, not all data warehouse systems will have all these elements, nor will they always interact in the way shown here.

'ETL' are sub-systems of a data warehouse system that Extract, Transform and Load data from one stage to the next. ETL sub-systems 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 sub-systems are also used to generate and maintain a central metadata repository.

In the COSMIC method terminology [3], all the sub-systems of a data warehouse system reside in the same application 'layer' and are thus 'peers'. The ETL sub-systems do not use each other's services; they do not even interact with each other directly, but via their shared databases. In the COSMIC model, persistent storage is available to any software that needs it, regardless of the layer structure.

The data warehouse architecture may define that the ETL sub-systems reside in different sub-layers. Such a view makes no difference to the COSMIC measurement process in what follows.

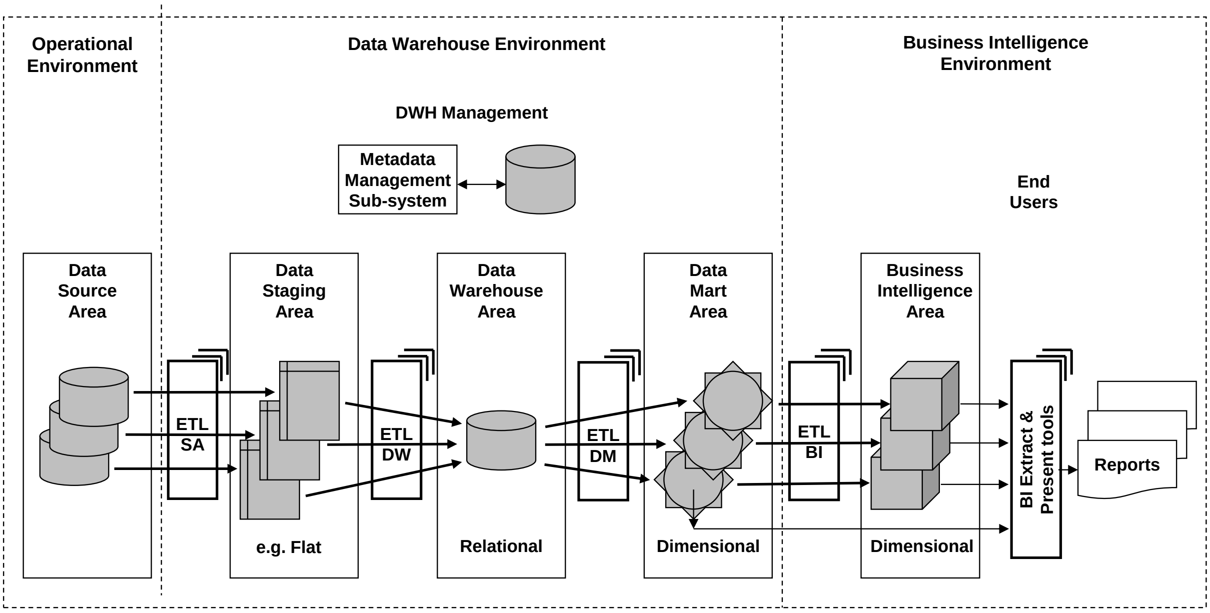

Figure 2.1 - Visualization of a data warehouse, Inmon type

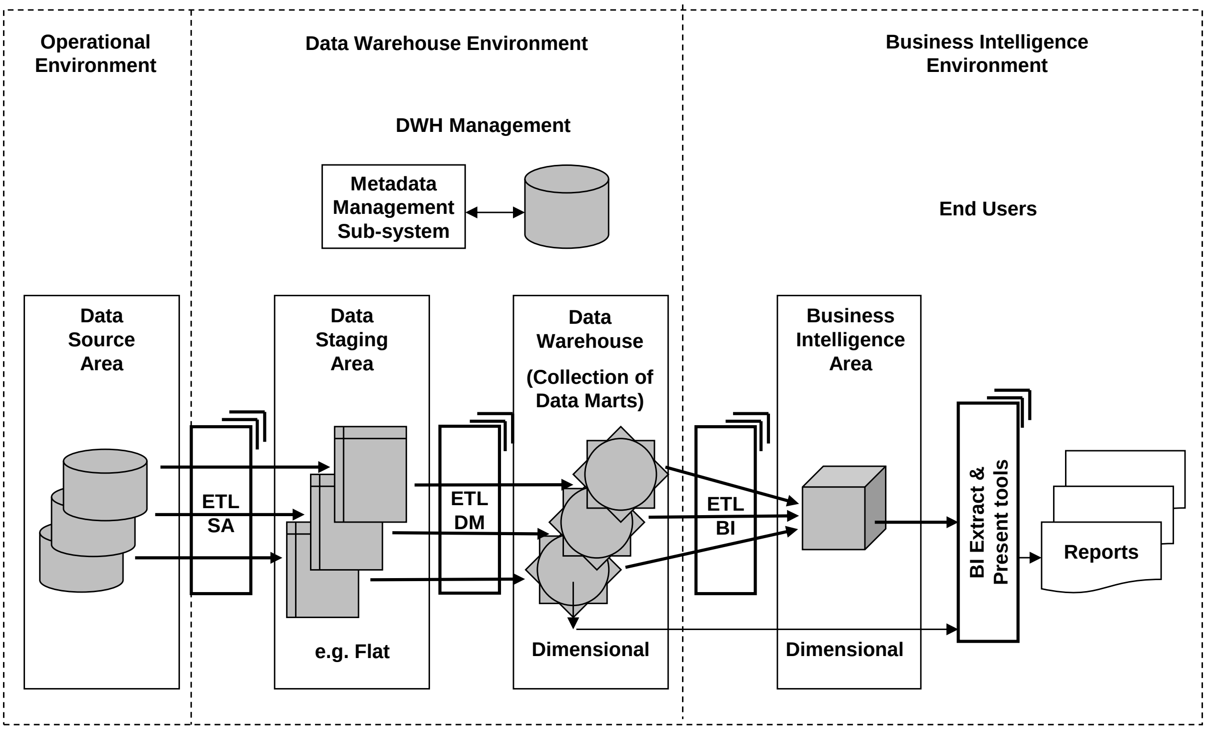

Figure 2.2 - Visualization of a data warehouse, Kimball type

2.2 Data stores and sub-systems of a data warehouse system

The main data stores and sub-systems of a data warehouse system are:

ETL sub-systems

The data warehouse system consists of four sub-systems that fill the data stores successively. Each will shortly be described under the data store it fills. The sub-systems are the ETL Staging area sub-systems, the ETL Data warehouse sub-systems, the ETL Data mart sub-systems and the ETL Business intelligence sub-systems.

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 are not part of 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 data store that is commonly loaded by 'ETL SA' sub-systems. 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 sub-systems.)

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 sub-systems may be used to process data into the staging area, for example to resolve name and address misspellings and the like.

Data warehouse database

In an Inmon-type data warehouse the data warehouse database is a relational data structure that is optimized for subsequent distribution. 'ETL DW' sub-systems 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 for all shared data. In a Kimball data warehouse the relational data warehouse is absent; data from the data staging area is directly loaded into the data marts

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'. In an Inmon type data warehouse, 'ETL DM' sub-systems 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 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. In a Kimball type data warehouse 'ETL DW' sub-systems create data marts derived from the data stores (flat files) in the data staging area. 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 both Figures 2.1 and 2.2, 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, an ETL-BI tool may be needed to re-configure and store the data in a Business Intelligence area, as also shown in both Figures 2.1 and 2.2.

BI databases can be visualized as cubes or, more generally as multi-dimensional stores (see below). 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; rather it is information that addresses a number of characteristics of data attributes 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 attributes 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 [7], as shown in the Figures 2.1 and 2.2. However, it is also possible that each ETL sub-system or tool may incorporate and manage the metadata it needs for its specific purpose.

All data warehouse systems must have this 'technical 'metadata which give the rules by which data are cleaned and transformed and structured into the data warehouse and marts. In addition, there can be 'business metadata' and 'process metadata' [8] and there are very many ways in which metadata can be organized, maintained and used by the ETL sub-systems and end-user tools. In this Guideline, for simplicity we will only describe basic examples of the functionality needed to access technical metadata. The Measurer is alerted to the fact that the functionality needed to use and maintain metadata may be much more complex that is described here.

2.3 (Multi-) Dimensional data storage

The Figures 2.1 and 2.2 show 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. This is the case in data marts and 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 2.3. 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.

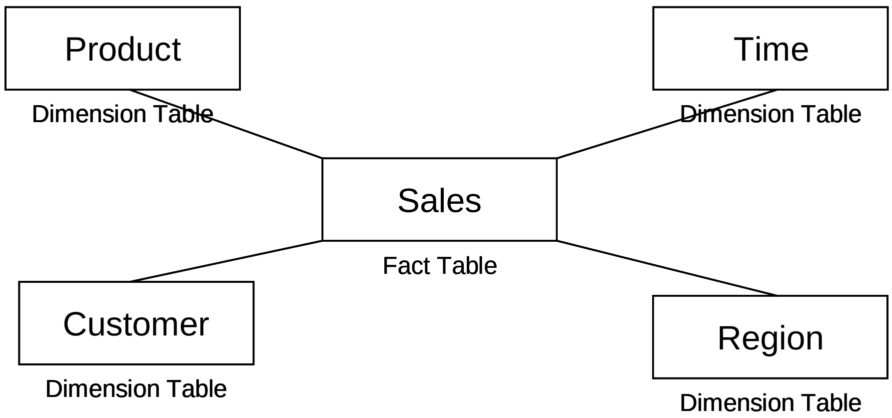

Figure 2.3 - Star schema

Dimension tables often consist of hierarchies of attributes. In consequence, it is possible for end users to view the data of their fact table by rolling up a dimension, or drilling 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 2.3, the name (or topic) of the fact table is 'Sales'. Assuming the lowest level of aggregation is 'monthly sales':

- the Sales fact table might have attributes such as 'current-year budgeted sales', 'actual monthly sales' (by month), etc.,
- all describing the object of interest (OOI) 'Sales of a given Product to a given Customer in a given Region in a given month'.

Examples of star schemas for other businesses might be:

- For a retailer, as in Figure 2.3, but with a 'shop' dimension in addition;
- For a manufacturer, the fact table might be named 'purchasing', with dimensions for 'supplier', 'material', 'factory', and 'time'.

Another equivalent form of storing data is the 'snowflake' schema shown in Figure 2.4. This is needed when the dimensions must have their own branches and they must be made explicit. 'Snowflake' schemas are implemented using relational databases and have technical advantages and disadvantages from star schemas which do not concern us here.

In the following we will only discuss star schemas since from a measurement point of view, the same concepts apply to both schema types.

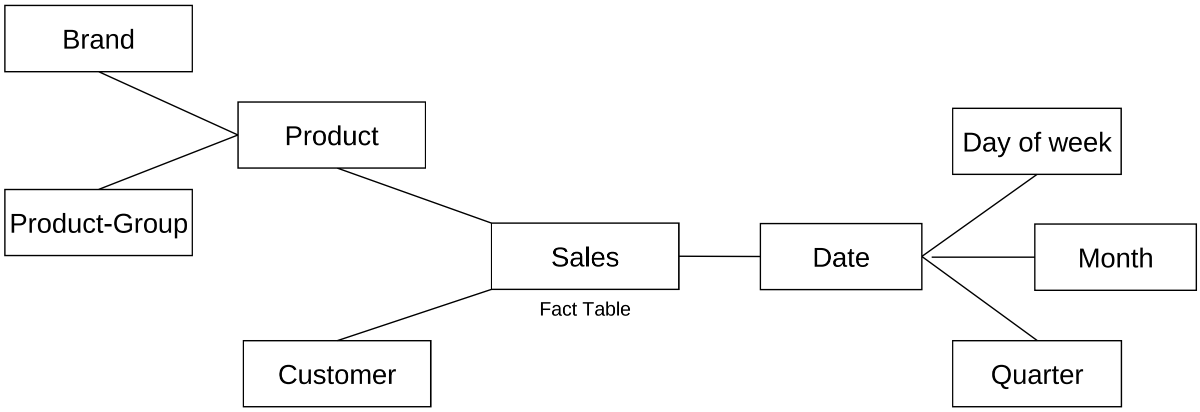

Figure 2.4 Snowflake schema

In the business intelligence area, data that has three dimensions can also be viewed as a cube. An example is shown in Figure 2.5. When more than three dimensions are involved, the term 'cube' is still used though technically it is a 'hypercube'. A partial graphical representation of a hypercube can be made by selecting any three of their N dimensions.

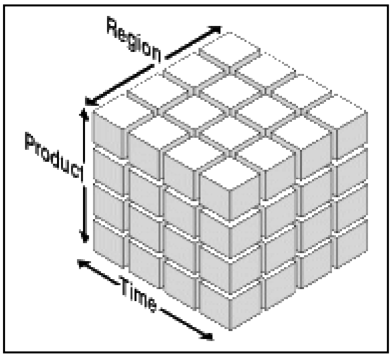

Figure 2.5 - Cubic schema

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 [3].

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, the level of decomposition of the software and the level of granularity of the FUR should be determined.

The model assumed in this guideline is of a data warehouse which is loaded by periodic batch updates and which is available for on-line enquiries, as outlined in chapter 2. As will be shown, all batch processes access persistently stored data, which can be accessed or requested via other software.

The purpose. We assume that the purpose is to measure the functional size of the data warehouse software, as input to a process for estimating development effort.

The scope. The 'overall scope' of the measurement is assumed to be the FUR of the data warehouse software, as per the example architectures in Figures 2.1 or 2.2., excluding the 'extraction and presentation tools' that allow end users to access data warehouse, data mart or business intelligence databases. For sake of simplicity we assume that these are externally-supplied software utilities that do not need to be sized, so are not within the overall scope of the measurement, however, if needed, the functionality of customized applications of such tools can be measured using the COSMIC method.

Each separate sub-system or tool in the data warehouse system, i.e. the ETL sub-systems and the metadata management sub-system may be executed autonomously and may be developed separately. It follows given the purpose of the measurement that we define each sub-system as having its own separate measurement scope. These are:

- ETL Staging area sub-systems;
- ETL Data warehouse sub-system (in an Inmon type data warehouse only);
- ETL Data mart sub-systems;
- ETL Business intelligence sub-systems;
- Metadata management sub-system.

Layers. As already described in section 2.1, in COSMIC terminology, all sub-systems of the data warehouse system reside in the same application 'layer', including any software that is used to maintain the metadata.

The functional users. Since no ETL sub-system communicates directly with any other ETL sub-system, no ETL sub-system is a functional user of any other sub-system. (These sub-systems pass data to each other via the persistent data storage that they use in common.) If however, in the architectures of Figures 2.1 and 2.2, all ETL sub-systems need to pass a request to access the metadata management sub-system, each ETL sub-system is a functional user of the metadata management sub-system, and vice versa.

Other functional users of the ETL sub-systems will include any DB administrators or ETL procedure administrators that control the processing of an ETL sub-system, or who receive output (e.g. error reports) from the sub-system. Likewise, another functional user of the metadata management sub-system will be a metadata management administrator.

The functional users of the Business Intelligence tools are the ‘end users’ who make the enquiries.

The level of decomposition. As each sub-system is autonomous and we have defined its scope so that it must be measured separately, each sub-system may be regarded as a ‘whole application’, i.e. there is no need to consider decomposition.

The level of granularity. We assume that the FUR of each sub-system is at the functional process level of granularity, so that the individual functional processes and data movements can be identified.

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 (FUR) are examined and the various functional processes, objects of interest, data groups, and the data movements are identified. This section treats the Inmon type data warehouse (see Figure 2.1). Section 3.2.6 discusses where the Kimball type (Figure 2.2) differs.

3.2.1 ETL Staging Area sub-systems

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

The first functional processes we encounter are those of the ETL-SA sub-systems that move data from operational data sources into the staging area files, as shown in Figure 3.1.

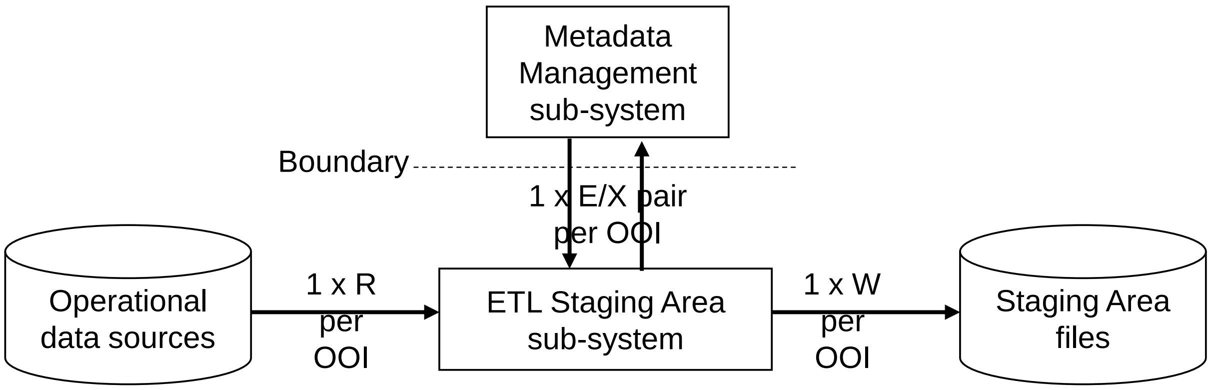

Figure 3.1 – ETL Staging area sub-system

To identify functional processes, first we should identify the objects of interest (OOI’s) in the operational data sources about which data must be moved to the staging area.

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,
- transform the data attributes as described by the metadata management system and
- 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, i.e. 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 section 4.2.3 of the COSMIC ‘Guideline for Sizing Business Application Software’ [3], especially example 6. In the context of an

ETL-SA sub-system, 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 sub-system, the subject of the code table will be an OOI.

In this and the following examples we assume that the ETL sub-systems can access the data of the data stores within their own boundary and that for access of the metadata a request must be passed to the metadata management sub-system. In the latter case an Exit and an Entry to move the request to and receive the requested data from the metadata sub-system must be identified. If, in contrast, the metadata store is within the boundary a single Read suffices to obtain these data.

EXAMPLE 1: An ETL SA sub-system

For a simple functional process in an ETL Staging Area (SA) sub-system 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 sub-system to obtain the transformation rules for this OOI*

E *from the metadata management sub-system 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 3.2 below shows the functional process of the ETL-SA sub-system of Example 1 and its interaction with the functional process of the Metadata Management sub-system in the form of a Message Sequence Diagram.

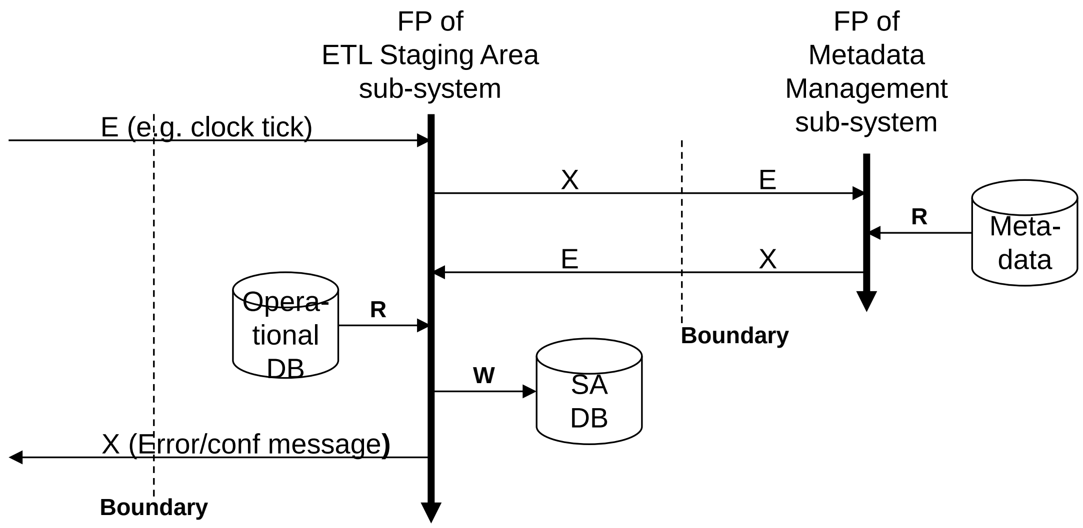

Figure 3.2 – Interaction of the Staging Area sub-system and the Metadata Management sub-system

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

a) If the functional process of the ETL-SA sub-system must move a logical grouping of data about, say, two OOI's, then the above example of moving multi-item orders 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 is no need for 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.) The total size of this functional process would then be 8 CFP.

b) If the data are supplied to the ETL-SA sub-system by an application of the operational environment (i.e. the ETL-SA sub-system is 'passive', awaiting the arrival of operational data) then the triggering Entry of the above functional process of Example 1 would not be a clock tick, but would move data describing one OOI from the operational application and that had to be loaded to the staging area. Consequently, a Read of the operational 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 sub-system is maintained and stored by this sub-system (rather than by a separate metadata management system as shown in Figure 2.1), then the Exit/Entry pair in the above Example 1 needed to retrieve the metadata should be replaced by a single Read. This is also true for the other ETL sub-systems that are analyzed below. The total size of this functional process would then be 5 CFP.

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.

3.2.2 ETL Data Warehouse sub-system (Inmon)

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

In the ETL data warehouse sub-systems 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 3.3.

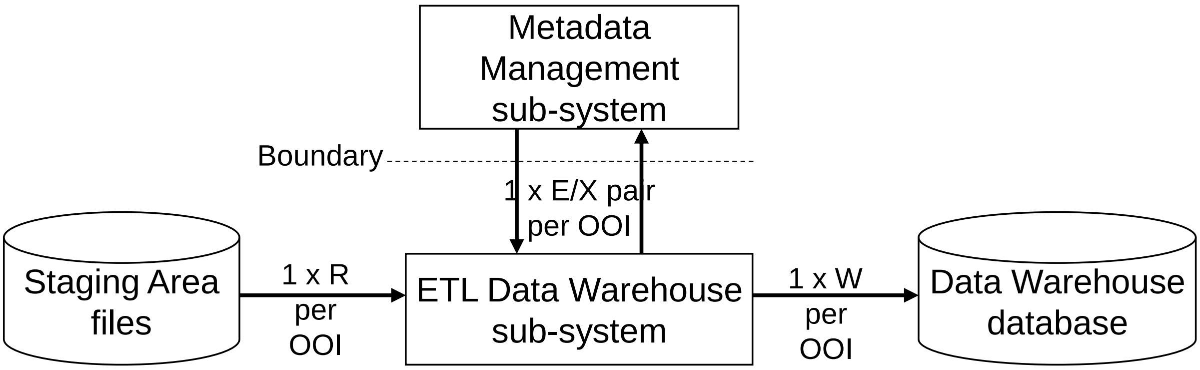

Figure 3.3 – ETL Data Warehouse sub-system (Inmon)

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 sub-system have the same structure as those of the ETL staging area sub-system (assuming the data in the data warehouse database are not stored multi-dimensionally. For a case where multiple fact tables are maintained, see section 3.2.3).

EXAMPLE: A simple functional process of the ETL data warehouse sub-system 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 sub-system to obtain the transformation rules for this OOI*
E	*from the metadata management sub-system 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 Figure 3.2.

3.2.3 ETL Data Mart sub-system (Inmon)

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

In the ETL data mart sub-systems we find the functional processes that feed the data marts from the data describing the OOI's that are stored in the data warehouse sub-system, as shown in Figure 3.4.

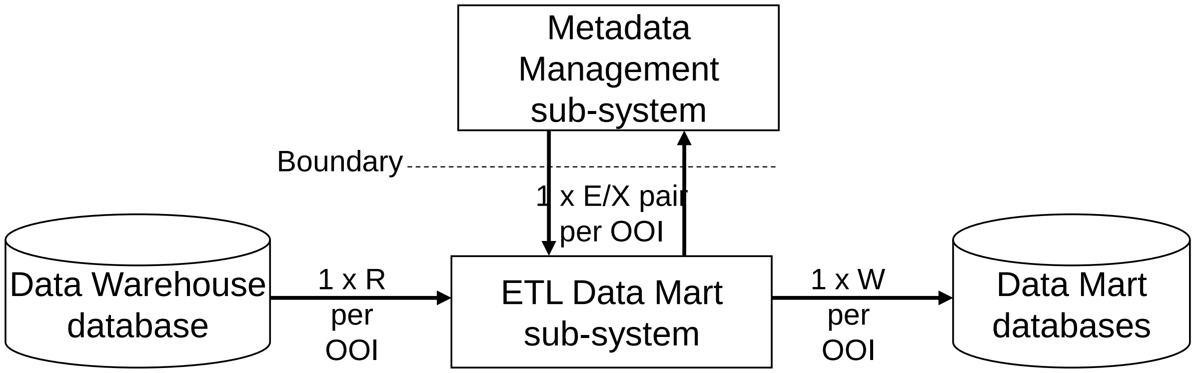

Figure 3.4 – ETL Data Mart sub-system

In the data mart databases, the data may be stored in a dimensional way, as in the star schema of Figure 2.3, which shows both 'dimension tables' and the 'fact table' (or tables). It is important to distinguish the creation of the fact table(s), 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 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 sub-systems, it is easiest to illustrate the process with an example star schema, as in Figure 3.5.

EXAMPLE. We assume functional user requirements for a data mart to store sales data in fact tables at the lowest (i.e. not aggregated) level and also for each level of aggregation resulting from the various possible combinations of the levels of all four dimensions[2]. 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 of the one country in which the company operates. The data warehouse owner therefore wishes to aggregate sales data along both the organizational and geographical axes independently.

[2] Data could be aggregated to several higher levels by various combinations of all the possible levels on *three* of the four dimensions, i.e. Product, Time and either Customer or Region/Country. But the Functional User Requirement is to store fact tables for all combinations of the levels of all four dimensions. If data is needed at higher levels of aggregation, this could be obtained by using Business Intelligence tools to enquire on the data mart. The highest possible level of aggregation of data derivable from this star schema is the sales of the total product range for all years to the whole country.

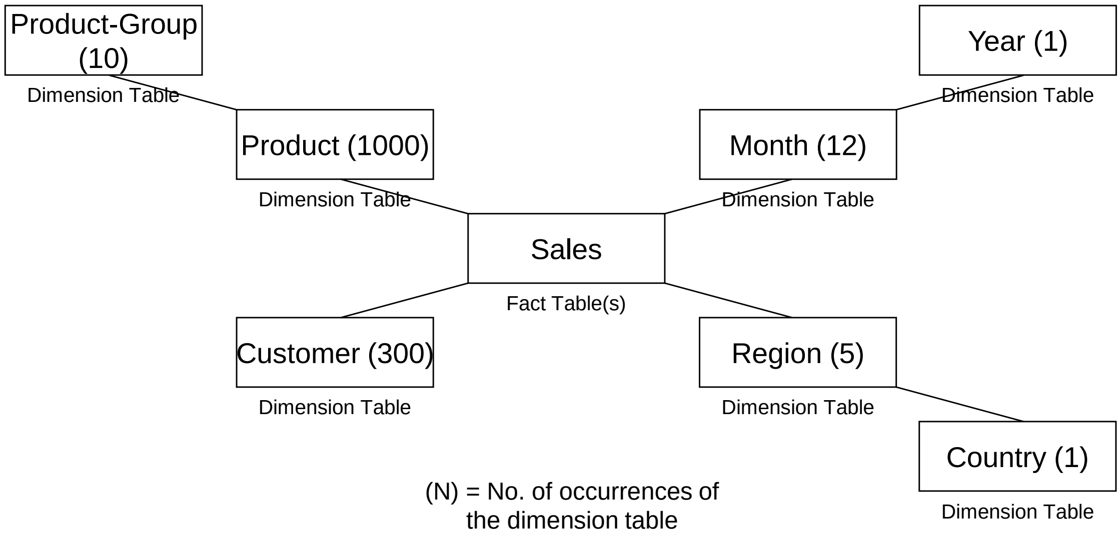

Figure 3.5 – Example of a star schema for a 'Sales' data mart

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 orders that have been shipped and paid for in the given month. The attributes of this 'sales' 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 (Product, Customer, Region, Month)'. The functional process of an ETL data mart sub-system 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 (Product, Customer, Region, Month) data*

X *error/confirmation* messages

Total: 6 CFP

(We assume here that for this functional process there is no further need to reference the Metadata Management sub-system. If reference to this sub-system is needed, an additional X/E pair must be added to the above functional process to obtain the metadata, as for the earlier ETL sub-systems.)

However, as the FUR for this example state that 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 given product to a given customer in a given region in a given month (the lowest level)*
- *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 (Product, Customer, Region, Month) *[the lowest level]*

Sales (Product, Customer, Country, Month)

Sales (Product, Customer, Region, Year)

Sales (Product, Customer, Country, Year)

Sales (Product-group, Customer, Region, Month)

Sales (Product-group, Customer, Country, Month)

Sales (Product-group, Customer, Region, Year)

Sales (Product-group, Customer, Country, Year)

So in practice, if there is one functional process of the ETL data mart sub-system that reads orders, etc. in the data warehouse and computes and loads sales data to fact tables for 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)*

8 x W's *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 <u>input</u> data. This assumption implies for example:

- the product-group to which a product belongs would have to be obtained for each product from the order items. (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.):
- 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;
- the date of each transaction will be imported, from which the month and year are computed.

Second, such a functional process as analyzed above could 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 steps probably involving more Reads than shown above. (Remember, however, that the goal is to measure the FUR, not any particular implementation for processing efficiency.)

Third, 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;
- whether 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 Figure 3.5 above, the potential number of occurrences of Sales records at each level of aggregation in this data mart can be computed as follows

Sales (Product, Customer, Region, Month)	*1000 x 300 x 5 x 12 = 18,000,000 records*
Sales (Product, Customer, Country, Month)	*1000 x 300 x 1 x 12 = 3,600,000*
Sales (Product, Customer, Region, Year)	*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' before the Sales data can be loaded. 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 sub-system 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 ‘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 ETL Business Intelligence sub-system

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’ sub-systems 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 sub-system 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 Figure 3.6.

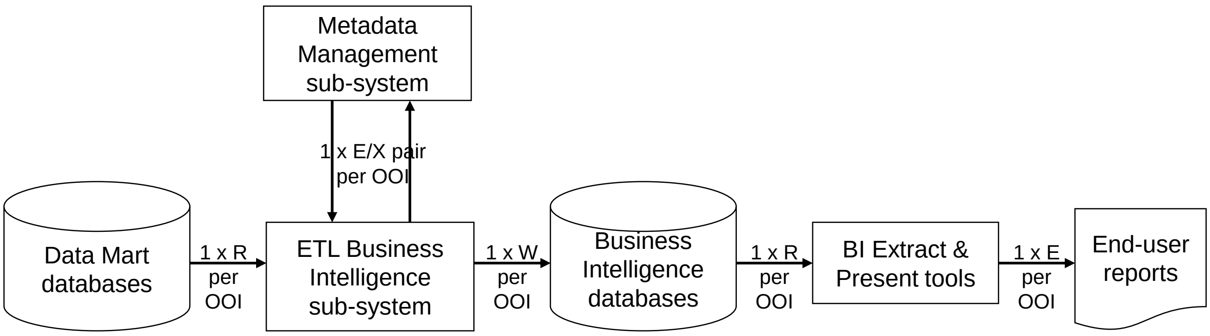

Figure 3.6 – ETL BI sub-system and extract and presentation tools

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

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 3.5 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 and assuming no need for an error/confirmation message)*
>
> *E* — *to start the enquiry, giving the month*
>
> *R* — *S (Product-group, Country, Month) – assuming data are stored at this level of aggregation*
>
> *X* — *S (Product-group, Country, Month) – which would have 10 occurrences, one for each Product-group*
>
> *Total size: 3 CFP.*

In this Guideline we assume 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). This is beyond the scope of this Guideline.

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 [3].

3.2.5 Metadata management sub-system

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

Regardless of whether this is a separate sub-system as in Figures 2.1 and 2.2, or whether metadata are integrated with each ETL sub-system, 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. Process metadata, like update frequency or system versioning, user profiles, access privilege files, data processing rules and use statistics can describe various possible OOI's for the metadata administrator [7]. Business metadata, like the content of data dictionaries, data on historical aspects, data on a data owner, will probably describe other objects of interests. For each OOI, there may 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 sub-system could therefore have many more data movements than the single 'E, X pair' shown in Figures 3.1, 3.2, 3.3, 3.4 and 3.6 (or the single Read data movement, if that applies).

3.2.6 ETL Data Mart sub-system (Kimball)

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

(This sub-section only for the Kimball type data warehouse).

In the ETL data mart sub-systems we find the functional processes that feed a data mart from the data describing the OOI's that are stored in the flat files in the staging area, as shown in Figure 3.7.

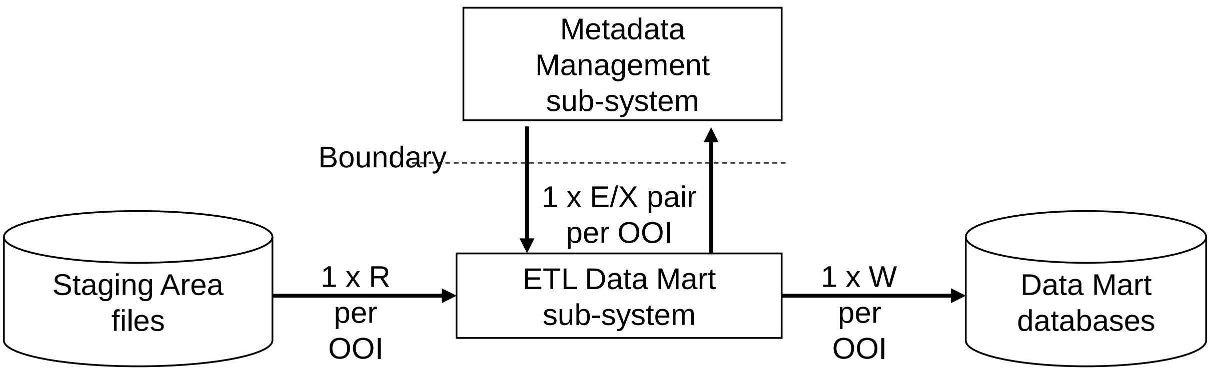

Figure 3.7 – ETL Data Mart sub-system

For the structure of a Kimball data mart, see the star scheme of Figure 3.8, which is Kimball's equivalent to Fig. 3.5 for how fact and dimension data should be *physically* stored.

Under the same assumptions of section 3.2.3, the functional processes to load the transformed data into a data mart database are identical to the functional processes described in that section.

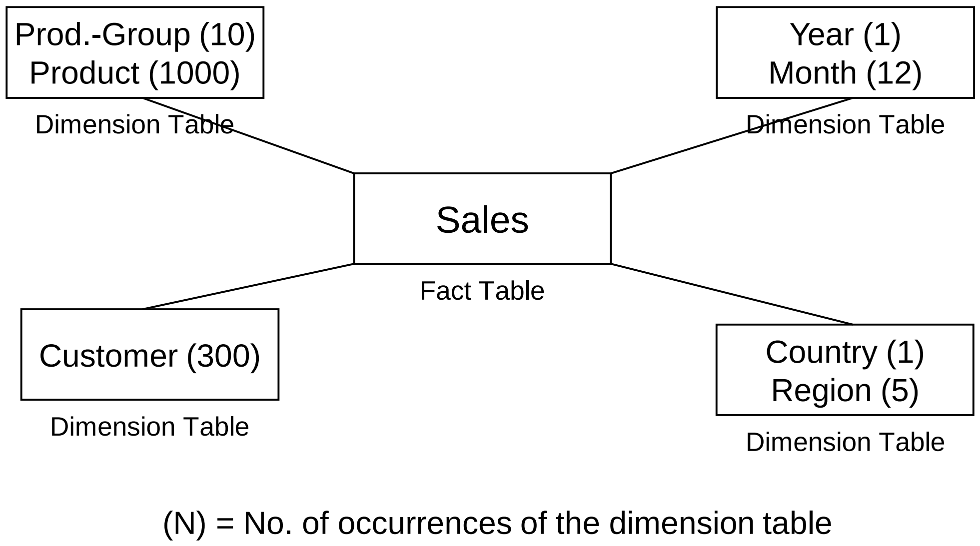

Figure 3.8 – Kimball view of a data mart star schema

According to the Kimball view[3] a fact table consists of

- references (foreign keys) to the dimension tables and
- 'measures' that are relevant to a business process

Measures are the attributes the business is interested in, such as order amount, extended cost, quantity ordered, gross margin.

Dimension tables supply the selection parameters for queries on the measures in the fact tables.

In practice dimension tables may contain hierarchies of attributes. In Kimball's view, dimension tables aren't (and to ease queries must not be) normalized.

Note that COSMIC size measurement requires that data be normalized in order to identify the OOI's. The same analysis shown in section 3.2.3 and the associated Fig. 3.5 therefore applies to this example, regardless of whether we take the Inmon or Kimball view of how data should be physically stored.

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.

[3] From 'Data Warehouse Design Solutions' by C. Adamson and M. Venerable, foreword by R. Kimball.

4

EFFORT ESTIMATION USING COSMIC SIZES OF DATA WAREHOUSE SUB-SYSTEMS

The various data warehouse software sub-systems 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 sub-systems. 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.

However, when deriving effort to develop or to enhance data warehouse (DWH) sub-systems from their COSMIC-measured functional sizes, especially when using benchmark data collected for non-DWH projects, certain points need special attention.

A different project productivity for the different sub-systems

When estimating the amount of time required for building the system, it could be advisable to consider using a different project productivity (size/effort) for the different sub-systems. For example, a distinction could be made between the amount of effort per CFP that is required to build the business intelligence sub-system (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 sub-systems. Therefore, it is important that effort data be defined, gathered and registered in such a way that this distinction can be made when analyzing the results of completed projects.

Complexity of DWH sub-systems

Measurers should be aware that the real functionality of DWH sub-systems may be much more complex than the examples given in this Guideline. Compared with these examples, ETL sub-systems may:

- have many more interactions with the metadata sub-system, or directly with the metadata database, than are shown in the examples;
- have many more data movements per functional process.

Data Manipulation

Data warehouse experts may be 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.

- The assumed project delivery rate (hours per function point) may be adjusted to compensate for the effort needed to deal with data manipulation.
- The COSMIC method offers the possibility to create local extensions of the method. For example, it has been suggested that it might be helpful to investigate in future studies the possibility of sizing the actual data manipulation sub-processes in the ETL processes, for instance by assigning functional sizes to the different data manipulation sub-process types that are normally considered to be accounted for by the data movement sub-process types.

Note however that 'a local extension' may imply 'only locally useful', i.e. it may prevent benchmarking.

Multiple levels of data aggregation

A consequence of measuring COSMIC sizes of multiple-dimensional databases where facts are stored at multiple levels of aggregation is that some functional processes may involve reading or writing data describing large numbers of objects of interest (OOI's), due to the multiple combinations of dimensions of the schemas. The effort to develop the software to handle these large numbers of OOI's may not be proportional to their number.

To understand this we will start with a simple example of a report that might arise from an enquiry on the fact tables of the example in section 3.2.3, which is based on the star schema of Figure 3.5, but which does not have a Region/Country dimension. The report illustrated below gives monthly sales figures for the customer 'John Doe & Co.' for each product sold, plus totals for each of the ten product groups, and also for the total sales for the year so far at the product and product-group levels.

The squares shaded yellow illustrate examples of the four different types of sales data for the four OOI-types about which data are shown on the report. (The report is mainly filled with repeated *occurrences* of these sales facts.)

Sales report January - March 2015 ($)								
Customer John Doe & Co								
Product	**Jan**	**Feb**	**Mar**	**Apr**	**May**	**(etc)**	**(etc)**	**Yearly Total**
A111	1951	2145	2137					6233
A112	95	130	240					465
A113	459	634	745					1838
(etc)								
Total PG 1	4591	4791	5342					14724
B111	495	495	520					1510
B112	321	432	543					1296
B113								0
(etc)								
Total PG 2	1037	1345	1599					3981
(etc)								0
(etc)								0
Total PG 10	49	44	43					136

In the notation of section 3.2.3, the four OOI's about which data are reported are as below (noting that this report does not have any data relating to the region/country dimension, and also that the report might be produced for just this one customer or repeated for all customers.)

Sales (Product, Customer, Month) [the lowest level]

Sales (Product, Customer, Year)

Sales (Product-group, Customer, Month)

Sales (Product-group, Customer, Year).

Now although the report shows data about four OOI's, there are really only two 'processes' needed to produce the aggregated data, assuming the Sales (Product, Customer, Month) data already exist at the lowest level. The two aggregation 'processes' are:

- The process of aggregating monthly sales data at the product level to the product-group level
- The process of aggregating monthly sales data to annual sales data, which is the same process whether applied to the monthly product sales or to the monthly product-group sales.

(Only two processes are needed if the alternative route to produce the aggregated data is taken which starts with aggregating from monthly to annual sales (for both products and product-groups) and then from annual product to annual product-group sales.)

The number of such ‘processes’ needed to aggregate data from the lowest level of a n-dimensional schema for all levels defined on the schema is the total number of aggregation steps summed over all dimensions. In the case of this report, the number of processes is = 2. In the case of the whole schema of Figure 3.5, the number of process steps is three, although the number of OOI’s of the fact tables of the whole schema is eight. It seems likely that this ‘process’ count should be more closely related to the effort to analyze, design, program and test the software to produce the data, than the count of OOI’s about which data must be created.

N.B. We must clearly distinguish:

- the COSMIC functional size of the reports where an Exit must be identified and counted for each OOI about which data is produced;
- the relative effort needed to produce the reports which might be more proportional to the number of ‘processes’ as illustrated above (or in other words require a lower effort per CFP, or higher productivity than normal).

REFERENCES

All the COSMIC documents listed below, including translations into other languages, can be obtained from the download section of www.cosmic-sizing.org .

[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] The COSMIC functional size measurement method, version 4.0/ 4.0.1: Measurement Manual.

[4] The COSMIC functional size measurement method, Guideline for sizing business application software, version 1.1.

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

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

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

[8] See for example, Wikipedia on the 'Common Warehouse Metamodel', 'Metadata Standards', etc.

APPENDIX A – GLOSSARY OF TERMS

The following terms are used in this guideline. For terms being used throughout the COSMIC functional size measurement method (the 'COSMIC method'), see the Measurement Manual [3].

In the definitions given below terms that are defined elsewhere in this glossary are under-lined, for ease of cross-reference.

Business intelligence. Historical, current, and predictive views of business operations.

Data mart. An extension of the data warehouse for specific use.

Data staging area. The data store of the data warehouse that is loaded from operational data sources, after having been cleaned and matched with existing data.

Data warehouse database. A data source that is optimized for distribution.

Data warehouse. A special type of business application designed to hold and/or to present both detailed and summarized data to support business analysis and decision making.

Dimension table. A table of attributes (types) with which the facts of the fact table can be viewed (Wikipedia).

Dimensional data storage. Storage of aggregations and computations in order to make retrieval of data faster.

ETL sub-system. A program that Extracts, Transforms and Loads data from one data store to the next.

Fact table. A table that contains the data concerning business events, transactions or the outcome of processes. Fact tables are defined as one of three types:

- Transaction fact tables record facts about a specific event (e.g., sales events)
- Snapshot fact tables record facts at a given point in time (e.g., account details at month end)
- Accumulating snapshot tables record aggregate facts at a given point in time (e.g., total month-to-date sales for a product) (Wikipedia).

Metadata. Data that define characteristics of data attributes.

Operational data source (- database). The database of an operational system in which transactions of the organization are captured.

Snowflake schema. A logical arrangement of tables in a multidimensional database such that the entity relationship diagram resembles a snowflake shape. "Snowflaking" is a method of normalizing the dimension tables in a star schema (Wikipedia).

Star schema. A schema that consists of one or more fact tables referencing any number of dimension tables (Wikipedia).

APPENDIX B – THE MAIN CHANGES FROM VERSION 1.0 TO VERSION 1.1

This appendix contains a summary of the principal changes made in the evolution of the COSMIC Data Warehouse guideline from version 1.0 to the present version 1.1.

A 'MUB' is a Method Update Bulletin, published between major releases of the Measurement Manual to announce and explain proposed changes.

V4.0 Ref	Change
-	Where needed text has been changed so as to comply with the Measurement Manual v4.0.
-	Figure numbering is now sequenced per chapter rather than increasing, in conformity with the numbering in other guidelines (e.g. Figure 5 is now Figure 3.2).
-	Some rationalization of terminology. ETL 'programs', 'tools' or 'tool components' are now referred to as ETL 'sub-systems'. The word 'tool' is used only for end-user software such as DSS, OLAP, EIS, data mining, etc., software that is normally procured to enquire on data warehouse or data mart databases.
1.1	Both views on data warehousing (Inmon and Kimball) introduced.
Ch. 2	Subtitles have been added to structure the chapter: 2.1 'An overview', 2.2 'Data stores and software components of a data warehouse system', 2.3 'Dimensional Data Storage'. The latter section doesn't describe stores or software components but describes a way of storing, hence a separate section.
2.1	Both last sentences of Note 1 have been moved to the Measurement strategy section 3.1 where they belong.
2.1	The text on layers has been expanded in light of the revised definition of a layer in v4.0 of the method.
2.1	A diagram of the Kimball view of a data warehouse has been added.
2.2	The sentence about fuzzy logic has been removed as the concept plays no further part in the Guideline. The text on Metadata has been expanded to point out that it can exist and be used in many more forms than are covered in this Guideline.
2.3	Diagram added showing a snowflake schema.
3.1	The text of this section has been more clearly structured.
3.2	Text on levels of granularity and decomposition has been added.
3.2	Figures 3.1, 3.3, 3.4, and 3.6 have been updated to show the data movements between components and databases.
3.2.1	The last sentence, on the PDR) removed as it duplicates the text of the first bullet of ch. 4.
3.2.3	The example of this section originally stated '*We assume functional user requirements for a data mart to store sales data in a fact table at the lowest (i.e. not aggregated) level and also at each level of aggregation resulting from the various possible combinations of the dimensions.* The second word 'at' is misleading. It should be 'for' (each level of aggregation). A footnote has been added for the example pointing out that higher levels of aggregation of data are possible than the eight levels at which data are required to be stored.

3.2.4	The examples in this section had several errors and contradicted the statement in section 3.2.3 that fact tables are stored for each level of aggregation. The text has been changed.
3.2.6	New section, describing the ETL Data Mart tools component of a Kimball type data warehouse and the Kimball (physical) view of a star schema.
4	This Chapter now focuses on factors to consider when estimating effort to develop data warehouse software from COSMIC-measured functional sizes. New text has been added on the processes involved in generating data at multiple levels of aggregation.
Glossary	A definition has been added for the term ‘Snowflake’, and a broader definiotion for a ‘Fact table’.

APPENDIX C - COSMIC CHANGE REQUEST AND COMMENT PROCEDURE

The COSMIC Measurement Practices Committee (MPC) is very eager to receive feedback, comments and, if needed, Change Requests for this guideline. This appendix sets out how to communicate with the COSMIC MPC.

All communications to the COSMIC MPC should be sent by e-mail to the following address:

mpc-chair@cosmic-sizing.org

Informal general feedback and comments

Informal comments and/or feedback concerning the guideline, such as any difficulties of understanding or applying the COSMIC method, suggestions for general improvement, etc should be sent by e-mail to the above address. Messages will be logged and will generally be acknowledged within two weeks of receipt. The MPC cannot guarantee to action such general comments.

Formal change requests

Where the reader of the guideline believes there is a defect in the text, a need for clarification, or that some text needs enhancing, a formal Change Request ('CR') may be submitted. Formal CR's will be logged and acknowledged within two weeks of receipt. Each CR will then be allocated a serial number and it will be circulated to members of the COSMIC MPC, a world wide group of experts in the COSMIC method. Their normal review cycle takes a minimum of one month and may take longer if the CR proves difficult to resolve. The outcome of the review may be that the CR will be accepted, or rejected, or 'held pending further discussion' (in the latter case, for example if there is a dependency on another CR), and the outcome will be communicated back to the Submitter as soon as practicable.

A formal CR will be accepted only if it is documented with all the following information.

- Name, position and organization of the person submitting the CR.
- Contact details for the person submitting the CR.
- Date of submission.
- General statement of the purpose of the CR (e.g. 'need to improve text…').
- Actual text that needs changing, replacing or deleting (or clear reference thereto).
- Proposed additional or replacement text.
- Full explanation of why the change is necessary.

A form for submitting a CR is available from the www.cosmic-sizing.org site.

The decision of the COSMIC MPC on the outcome of a CR review and, if accepted, on which version the CR will be applied to, is final.

Questions on the application of the COSMIC method

The COSMIC MPC regrets that it is unable to answer questions related to the use or application of the COSMIC method. Commercial organizations exist that can provide training and consultancy or tool support for the method. Please consult the www.cosmic-sizing.org web-site for further detail.