

THE DATA DICTIONARY

What is A Data Dictionary?

A Data Dictionary is a repository of data about data (James Martin).

The main purpose of a Data Dictionary is to provide a source of reference in which the analyst, user and designer (i.e. the project team) can look up an identifying name and find out the content.

The Data Dictionary makes the Structured Specification rigorous.

The Data Dictionary survives the analysis phase and acts as the source of reference during the Design phase and depending on the form in which it is held, into the Development phase. It can also serve as a source of information for future projects.

Contents of Data Dictionary

The Data Dictionary (DD) contains entries for:

- Data flows
- Data stores
- Process names
- Entity names
- Entity Attributes
- Mini specs

A Data Dictionary should be non-redundant i.e. it should have only one entry for each occurrence of the objects listed above and the entries should not contain information which is obtainable elsewhere in the Structured Specification.

Data Flows and Data Stores

Figure 1 shows a data description hierarchy.

The simplest feature - the primitive - is the data element. A data element cannot usefully be further decomposed e.g. invoice date, total amount, item price.

A data structure is a related set of data elements, e.g. invoice.

Data flows and data stores are usually data structures consisting of several data elements, although it is possible to have a data flow or store which consists of just one element.

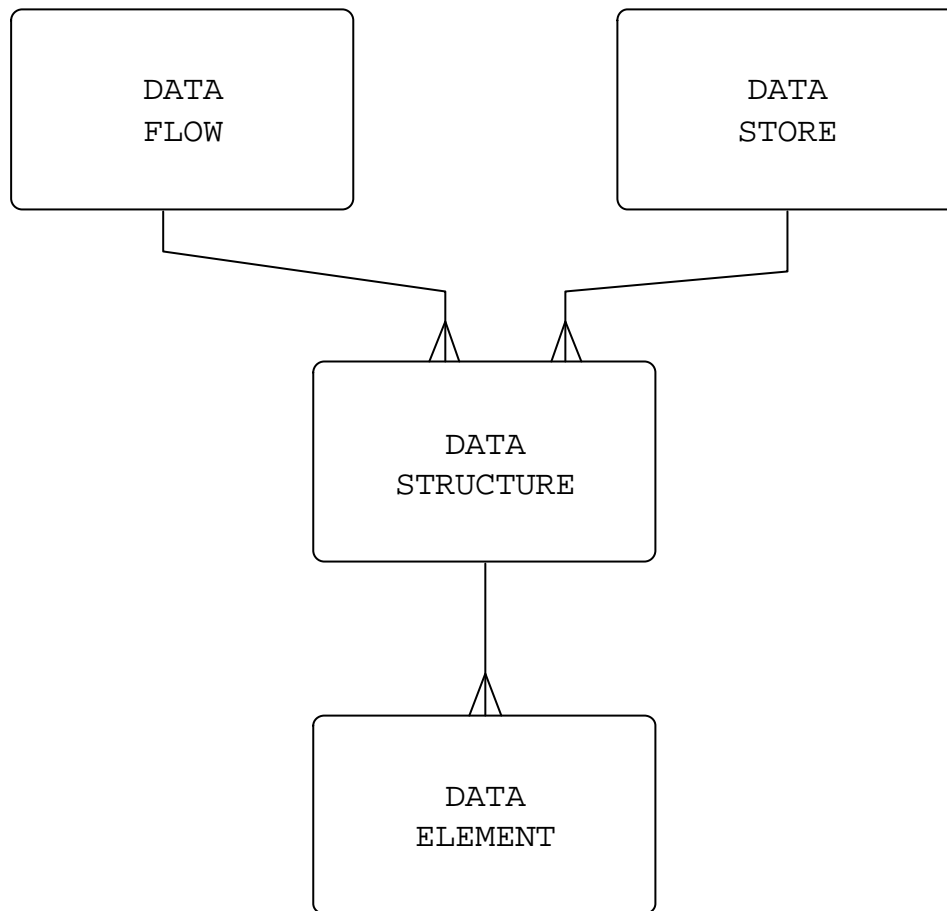


Figure 1

Data Dictionary Entry for A Data Element

The entry for an element should include:

- Name
- Size
- Possible values or range

It may also include the author's name and the date of entry and comments. The comment should be any useful piece of information which is not contained elsewhere in the Structured Specification e.g. 'this data item is no longer used'.

Figure 2 shows examples of data element definitions.

<i>DATA ELEMENT DEFINITION EXAMPLES</i>			
Data Element	Name	:	Price
	Size	:	99999.99
	Values	:	Current Minimum £15.50 Current Maximum £2050.00
Data Element	Name	:	Customer Name
	Size	:	X (40)
	**	:	Free Format
Data Element	Name	:	Applications Status
	Size	:	X
	Values	:	S = Successful H = Held R = Rejected

Figure 2

Data Dictionary Entry for A Data Structure

The entry for a structure consists of the definition and other useful information such as:

- * Frequency
- * Volume
- * Peaks and valleys
- * Security considerations

This other information may not always be available during the analysis phase.

Defining a Structure

A structure may be defined initially in terms of sub-structures which are then themselves defined, or in terms of data elements. The analyst must select the approach appropriate to each structure.

If there are many data elements in a structure it may be hard to read and absorb the definition if the structure is defined only in terms of data elements.

On the other hand, a structure defined as several substructures means that several accesses to the Data Dictionary are required before the actual content of the structure - the data elements - are reached.

When a structure has been defined in terms of its substructures and data elements, each data element must be defined.

Since defining a structure is not simply a matter of listing all the data elements a means of showing the relationship between them must be available.

Figure 3 shows the symbols we will use to define the constructs in data structures and Figure 4 shows examples of their use.

<i>DATA DICTIONARY NOTATION</i>	
=	IS COMPOSED OF
+	AND
()	OPTIONAL or OCCURS n TIMES
{ }	ITERATION - REPEATED ITEMS
[]	SELECT ONE OF
/	SEPARATES CHOICES
**	COMMENT

Figure 3

<i>EXAMPLES OF USE</i>	
Course	= <u>Course No</u> + Course Title + No. of Students + Tutor Name + (Basic Course Text)
Order	= Order Header + {Order Line} + Order Trailer
Employee ID.	= [Employee Name/Personnel No./Social Insurance No]

** All three may become mandatory in near future.

Figure 4

- The basic constructs are:
- Sequence - the order in which data elements appear.
 - Selection - either/or (alternatives) or select one of (where there are more than two possible choices).
 - Iteration - Where a data item or group of data items is repeated a number of times.

Note: that the upper and lower limits of iteration are shown outside the iteration brackets.

To allow for greater precision we have an additional symbol to define selection where the alternatives are zero or one - the optional symbol '()'.
(Note: The original image contains a typo '()' which has been corrected to '()' in this transcription.)

Figure 5 shows a typical invoice form and Figure 6 shows Data Dictionary definition for it.

ANY COMPANY INVOICE									
INVOICE NO: [][][][][][]		Date:							
Customer Name: _____		Delivery Address: _____							
Address: _____		(if different) _____							
_____		_____							
*ITEM	Quantity	Price	Cost						
Totals									
* Item Code if for associated company Item Description if for outside customer Customer Discount (if any)									
		<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 33%; padding: 5px;">% age</th> <th style="width: 33%; padding: 5px;">Discount Amt</th> <th style="width: 33%; padding: 5px;">Discount Total</th> </tr> </thead> <tbody> <tr> <td> </td> <td> </td> <td> </td> </tr> </tbody> </table>		% age	Discount Amt	Discount Total			
% age	Discount Amt	Discount Total							

Figure 5

Invoice = Invoice No. +
 Invoice Date +
 Customer Name +
 Customer Address +
 (Delivery Address)+
 { [Item Description / Item Code] +
 Item Quantity +
 Item Price +
 Item Cost }+
 Total Quantity +
 Total Cost +
 (Customer Discount % +
 Discount Amount +
 Discount Total)

Figure 6

Figure 7 shows the same structure defined initially in terms of sub-structures.

DATA DICTIONARY DEFINITION USING SUB-STRUCTURES

Invoice = Invoice Header + {invoiced item} + Invoice Trailer

Figure 7

The key data element in a structure should always be underlined. Figure 8 shows definitions for a data store. When making an entry for a data store information about the organisation of the file should be given.

DATA DICTIONARY DEFINITIONS FOR DATA STORE

Customer File = {Customer Records} + Total No. of Customers

Customer Record = Customer Name + {Customer Address Line} +
 Telephone No. + Telex No. + Credit Rating +
 Discount %

** Data Store is maintained in Customer Name Alphabetic order.

Figure 8

Many organisations design a form specifically for defining data structures, which may make the use of some of the operators unnecessary. Figure 9 shows an example of such a form.

Data Dictionary	Title	System	Document	Name	Sheet
	Type		Name		
	Alias				
	Definition				
	Occurrence				
	Picture				
	Notes				
Author	Date				

Figure 09 - Typical Data Dictionary Form

Bi-directional Relationships

In our Data Dictionary we define the relationships in one direction only - top to bottom e.g. we define an invoice line as consisting of a number of data elements. When defining those data elements we do not say to which structure(s) they belong.

However, some Data Dictionary processors (see Maintenance of a Data Dictionary below) have a facility for reporting bi-directional relationships.

Aliases

Aliases arise when a data structure or element is known by more than one name. They occur quite often in business applications because different users and different areas in a business have various names for the same thing.

Aliases should be entered in the Data Dictionary so that the Data Dictionary will be a complete reference set, even though aliases are, strictly speaking, redundant.

When making an entry for an alias, simply give a reference to the name under which the object is already entered. Do not define it again.

Maintenance of A Data Dictionary

Maintaining a Data Dictionary manually is a substantial overhead. There are several possible approaches to automating the Data Dictionary:-

- a) use a text editor: The use of a text editor (such as in WORDPAD) is the simplest approach. It makes updating easier and listings of the entries can be obtained.
- b) use a standalone Data Dictionary package: There are several Data Dictionary packages on the market such as DATA MANAGER.

These packages usually provide facilities such as:-

- * non redundant input
- * listings in an order specified by the user
- * cross reference listings
- * including reports on bi-directional relationships mentioned in 'Bi-directional Relationships' above.

However, such packages do not necessarily relate at all to the techniques of structured analysis and it may be difficult to define structures and elements as described above.

c) use a Data Dictionary manager integrated with a structured analysis support tool:

The automated Data Dictionaries which are of most interest are those which are an integral part of a structured analysis support tool such as YOURDON TOOLKIT or EXCELERATOR.

The facilities offered by such packages include

- * population of the Data Dictionary from the DFDs and mini-specs
- * input screens for defining structures and elements appropriate to structured analysis
- * consistency checking
- * non-redundant input

as well as the reporting facilities described under b) above.

In addition, with some of these packages the Data Dictionary may be exported from the PC based workbench to a mainframe for use during the development phase.