

PROCEDURE SPECIFICATION (MINI-SPECS)

Procedure Specifications, or Mini-specs as they are generally called, are used to describe a function from a lower level DFD which cannot be decomposed further. For this reason the Mini-spec is often referred to as the 'Lowest level DFD'.

Mini-specs can take various forms depending on the most suitable method of expressing the basic process. Most of the methods used are probably already familiar to you, ie Flowchart, Structured English, Text (or narrative English) and Pseudo code. However, a very common method is the use of Decision Tables or Trees.

DECISION TABLES

Decision Tables are a useful and concise method of recording complex information in an easily readable format. They can be used at any appropriate point in a project.

A Decision Table is made up of four main parts.

CONDITION STUB									
ACTION STUB									

Figure 1

The **Condition Stub** is a list of all the conditions that apply.

The **Condition Rules** shows the combinations of conditions which must be satisfied.

The **Action Stub** is a list of all actions that are necessary.

The **Action Rules** shows the actions to be taken when particular conditions, or combinations of conditions are satisfied.

A title and any comments or notes required should also be shown. For completeness the date should also appear on the Decision Table.

A simple, but incomplete example of the technique is a traffic light table (see Figure 2). There are three colours and four combinations of the colours.

Red	Y			Y
Green			Y	
Amber		Y		Y
Stop	T			T
Proceed with caution		T		
Go			T	

Figure 2

The list of conditions in the CONDITION STUB can be in any order, but the list of actions in the ACTION STUB must be in the correct order in which they are to be carried out. The above table implies that 'stop' is the most important and is therefore put at the top.

Decision Tables are classified into three types:

- * LIMITED ENTRY
- * EXTENDED ENTRY
- * MIXED ENTRY

LIMITED ENTRY

Condition rules are shown as: Y = Yes

N = No

- = Immaterial (ie condition is not important)

Action rules are shown as:

T = Carry out action

(BLANK)= Do not do it

The still incomplete traffic light table (Figure 2) would become:

Red	Y	N	N	Y
Green	N	N	Y	N
Amber	N	Y	N	Y
Stop	T			T
Proceed with caution		T		
Go			T	

Figure 3

EXTENDED ENTRY

Shows the value of the Condition rule and describes the action to be taken in the Action rule.

The table in Figure 3 would now become:

Colour of Lights	Red	Amber	Green	Red & Amber
Action	Stop	Proceed with caution	Go	Stop

Figure 4

These tables are more compact and easier to read but much more difficult to check.

MIXED ENTRY

Mixed entry tables, as their name implies, use a combination of limited and extended entry conditions and actions.

The traffic light example does not readily lend itself to mixed entry but a hypothetical example would be:

Red and Amber	N	N	N	Y
Any other colour	Red	Amber	Green	
Stop	T			T
Other Action		Proceed with caution	Go	

Figure 5

Note that any one condition row or action row must be either limited entry or extended entry: not both, ie mixed on the same row.

CHECKING FOR COMPLETENESS

One of the advantages of Decision Tables is that they can be checked to ensure that every desired combination of conditions has been included. Two formulae are used.

LIMITED ENTRY -

$R = 2^C$ where:

R = Number of Rules

C = Number of Conditions

EXTENDED ENTRY

- $R = N1 \times N2 \times N3 \dots$ where:
- R = Number of Rules
- N1 = Number of possible values of Condition 1
- N2 = Number of possible values of Condition 2 and so on.

If formula 1 is applied to the traffic light table in Figure 3 it may well be seen that there are 3 conditions and therefore the number of rules should be $2^3 = 8$ and that is why the table was incomplete. The complete table should be:

	1	2	3	4	5	6	7	8
Red	Y	Y	Y	Y	N	N	N	N
Green	Y	Y	N	N	Y	Y	N	N
Amber	Y	N	Y	N	Y	N	Y	N
Stop			T	T				
Proceed with caution							T	
Go						T		

Figure 6

It is now apparent that there are problems with Rules 1, 2, 5 and 8. The lights are not working correctly and a decision has to be made as to what action to take. In real life most motorists would opt for 'Proceed with caution' but in systems terms this easy option could be disastrous. Let us assume that an error routine has been defined and the correct action to take is 'Stop' and 'Go to error routine'. This additional action must now be added to the table (Figure 7).

	1	2	3	4	5	6	7	8
Red	Y	Y	Y	Y	N	N	N	N
Green	Y	Y	N	N	Y	Y	N	N
Amber	Y	N	Y	N	Y	N	Y	N
Stop	T	T	T	T	T			T
Proceed with caution							T	
Go						T		
Go to error routine	T	T			T			T

Figure 7

Rules 1, 2, 5, and 8 now have the same action and could be merged using the 'Else' rule.

Red	Y	Y	N	N	E
Green	N	N	Y	N	L
Amber	Y	N	N	Y	S
Stop	T	T			E
Proceed with caution				T	
Go			T		
Go to error routine					T

Figure 8

The Else rule can be used on any type of table, but is more common on Extended or Mixed Entry tables. It should be used with care as important combinations of conditions may be undetected.

Limited Entry tables are more usually rationalised, and redundant rules removed by replacing the 'Y' or 'N' with '-' denoting that particular condition is immaterial. These redundant rules can also be recognised because they have the same actions. A simple illustration is shown in Figs 9 and 10.

C1	Y	Y	N	N
C2	Y	N	Y	N
A1	T	T		
A2	T	T		T
A3			T	

Figure 9

C1	Y	N	N
C2	-	Y	N
A1	T		
A2	T		T
A3		T	

Figure 10

Decision Trees

Decision Trees are pictorial representation of Decision Tables. They tend to be more user-friendly but are not really a suitable medium for showing complex logic. Although they can be constructed independently they are usually derived from decision tables and used as a 'more acceptable' method of communication.

Using the traffic light example, the Decision Table in Figure 7 would be represented as follows:

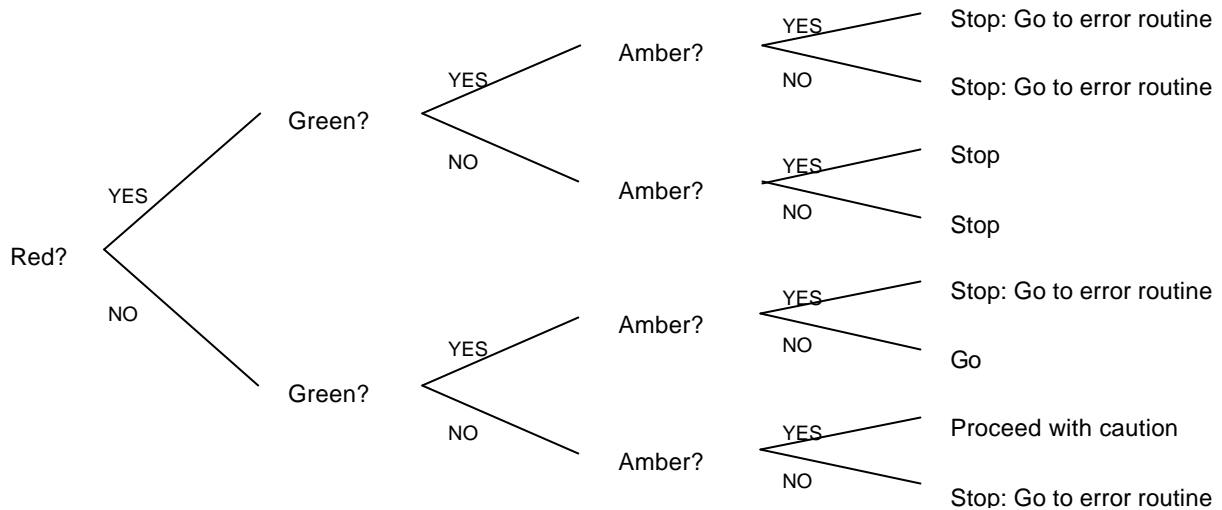


Figure 11

The basic rules are :

- * Condition entries are recorded from left to right (e.g. Red, Green, Amber).
- * The first condition entry is recorded once, the second twice, the third four times and so on.
- * Condition rules are known as Nodes
- * Like nodes should be aligned vertically and connected by branches clearly annotated Yes or No
- * The two branches leaving the last set of right-hand nodes point to the appropriate action.

Note :

With both Decision Tables and Decision Trees 'Go to' and 'Perform' statements can be used to break down large complex Tables or Trees into more comprehensible modules. For example, the last Action rules on Table 1 could be 'Go to Table 2', these are known as hierarchical.