



Working with eCATT (Extended Computer Aided Test Tool)

Hello friends my name is Prasad Sri K I would like to share some of my

experiences with eCATT, in which am currently working on.

Description:

In this article first I will introduce some of the basic concepts regarding eCATT and then we will see how to develop a test script to upload a test data file with a practical example.

So in precise in this article you will see what eCATT is, how we can use it; how we can load a test data from a file in 4.7X and it end with an example.

After reading this article you will have a clear idea about eCATT tool. And you will be able to write a test script in which test data can be loaded from a file.

This article is divided into following sections,

1. What is eCATT?
2. How to load test data from a file in 4.7X with an example?

What is eCATT?

eCATT stands for extended Computer Aided Test Tool (eCATT) which is built is testing tool to test SAP system. By using testing tool we can test the entire business process, and we can also use this tool with a third party testing tool (I am not covering this topic). Execution of every test script ends with a log, which explains the results of the test script.

By using eCATT we can do following operations,

- Test transactions, reports, and scenarios
- Call BAPIs and function modules
- Test remote systems
- Check authorizations (user profiles)
- Test updates (database, applications, GUI)
- Test the effect of changes to customizing settings
- Check system messages

For more information go to: [eCATT::Extended Computer Aided Test Tool \(sdn.com\)](http://sdn.com)

Here I am discussing fundamentals about eCATT in detail. You can find very good documentation in sdn.com.

To develop a test script in eCATT we need to follow the following steps,

1. Creating Test Scripts.
2. Creating Test Data Containers .
3. Understanding System Data Containers .
4. Executing Test Configurations.

There is a very good web blog on eCATT in sdn.com which explains eCATT with necessary screen shots. To read document click here [Blog on eCATT in sdn.com](http://sdn.com)



SCAN ME


Loading test data from a file in 4.7X with example:

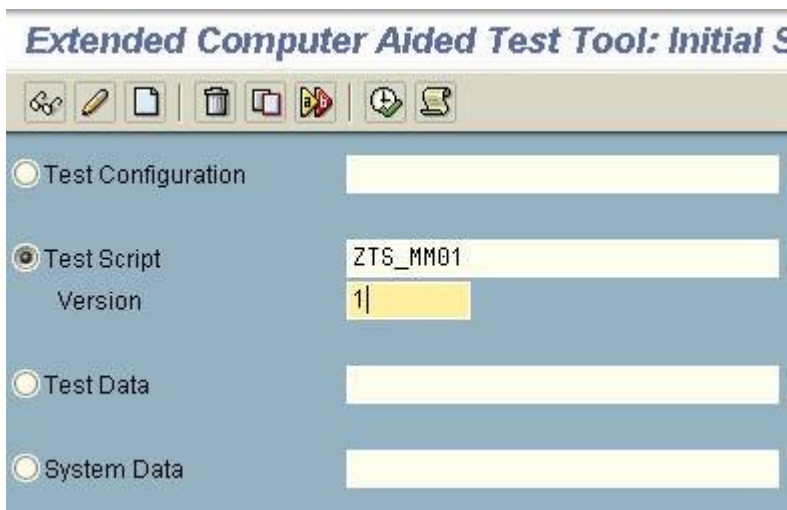
We can load test data from file using the ABAP...ENDABAP statements in eCATT.
In this article I have given an example on transaction code MM01 with sample code.
To upload the test data file follow the steps given,

1. Record the transaction

Open eCATT tool. This can be done using 'SECATT' tcode.



Give the script name and version number example 'ZTS_MM01'.
Version number can be used to maintain different program constructs under same
program name. And choose create  button.



SCAN ME

Give description and component name as 'BC-TWB-TST-ECA'.

The screenshot shows the 'Editor Attributes' dialog box with the 'General Data' tab selected. The 'Header Data' section contains the following fields:

Header Data	
Title	sample test script using eCATT tool in SAP by KVR Prasad Babu
Package	\$TMP
Person Responsible	HARI
Component	BC-TWB-TST-ECA
Type	B
eCATT Extended Computer Aided T...	

Select editor tab panel. Then click on 'pattern' button. Or go to Edit->Pattern or Press Ctrl+F6. This opens a 'insert statement' dialog box.

In that command dropdown box choose TDC (Record) option. Then press enter. Enter the transaction name as 'mm01' then MM01_1 interface automatically created. Then press enter.

The screenshot shows the 'Insert statement' dialog box with the following fields:

Insert statement	
Command	TCD (Record)
Transaction	MM01
Interface	MM01_1
Target System	

At the bottom, there are two buttons: a green checkmark (OK) and a red X (Cancel).

Then 'Create Material: Initial screen' will appear. Enter the necessary fields.

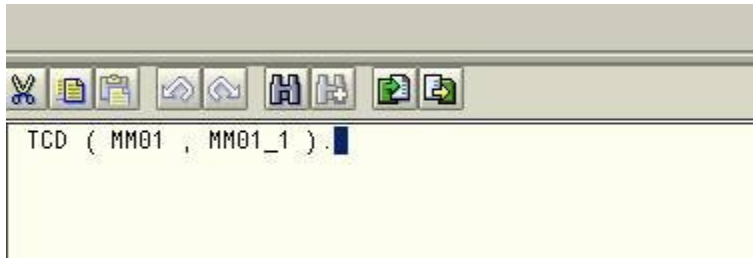
And be careful while recording, if not recording may fail. Note that while recording no error messages should pop up. If they happen restart the recording process. Here in this example I have considered a simple recording process. You can do any complex recording. If you have any doubts do feel free to mail me. I will reply to you. After completing the recording process recording ended dialog will appear. Choose 'Yes'.

The screenshot shows the 'Recording Ended' dialog box with the following text and buttons:

Do You Want to Transfer the Data?

Yes No Cancel

Then a TCD statement will appear in the editor area.



With this we have finished recording
Now let us see the variable declaration, assignment and programming part.
After developing a number of scripts I found one simple method to develop these test scripts. If you feel comfortable with this method you can also use it.


First note down the all screen fields in which you are entering values during recording process. Then create local variables in eCATT with the same name as the screen field technical name. (This method makes assignment easier).

Example:



In MM01 (material master) I have entered values for material, industry sector and material type. And their respective technical screen field values are,

RMMG1-MATNR
RMMG1-MBRSH
RMMG1-MTART

To find out technical value of the screen field select the field press F1, then click on technical information button. 

And now create the local variable as
V_MATNR,
V_MBRSH,
V_MTART,



To create local variables first click button, and then to create new variable



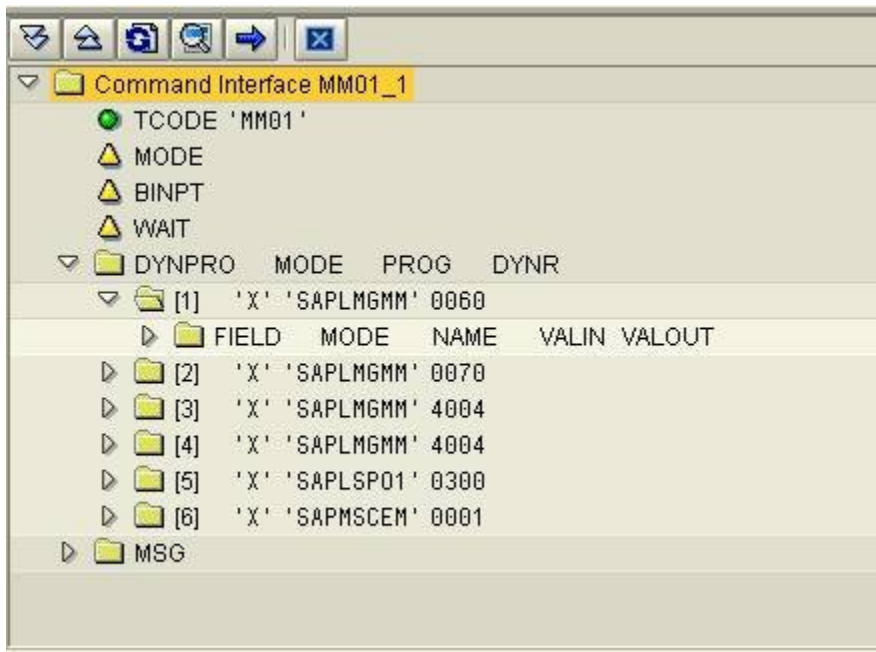
click button. And enter the variable name as (V_MATNR), Visibility of parameter as 'V' and finally parameter reference (name of the actual parameter). After declaring all the parameter it will look like this,

Parameter	Description	Value	IEV	Parameter Reference	Test System	ABAP T...	Length	Dec...	Group
V_MATNR	Material Number		V	MATNR		C	18		
V_MBRSH	Industry Sector		V	MBRSH		C	1		
V_MTART	Material Type		V	MTART		C	4		
V_MAKTX	Material Description		V	MAKTX		C	40		
V_MEINS	Base Unit of Measure		V	MEINS		C	3		
FILE_V	IBIP: Data transfer parameters	<INITIAL>	V	IBIPPARMS					
COUNT			V			I	10		
INT_LOC		1	V			I	10		

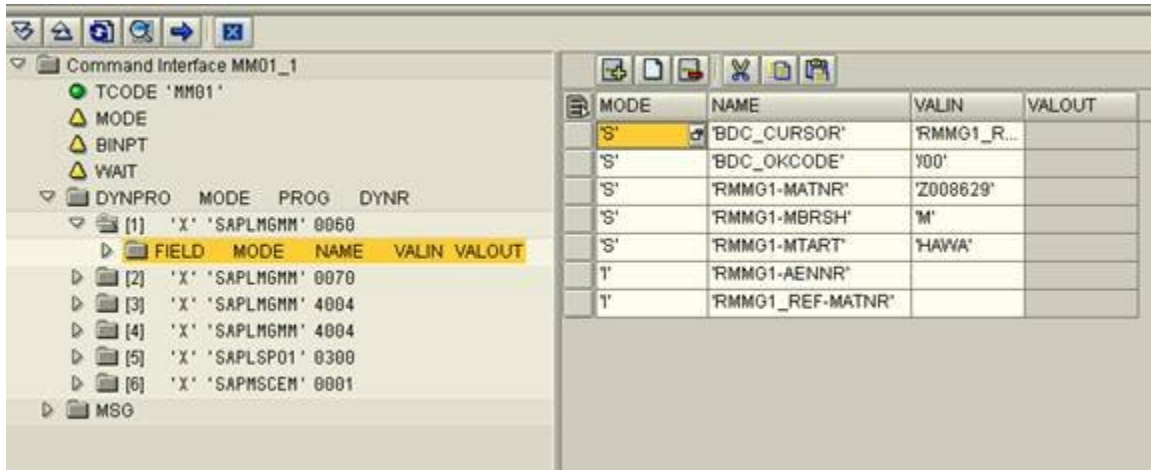
After declaring the local variables we need to assign them to screen field values.



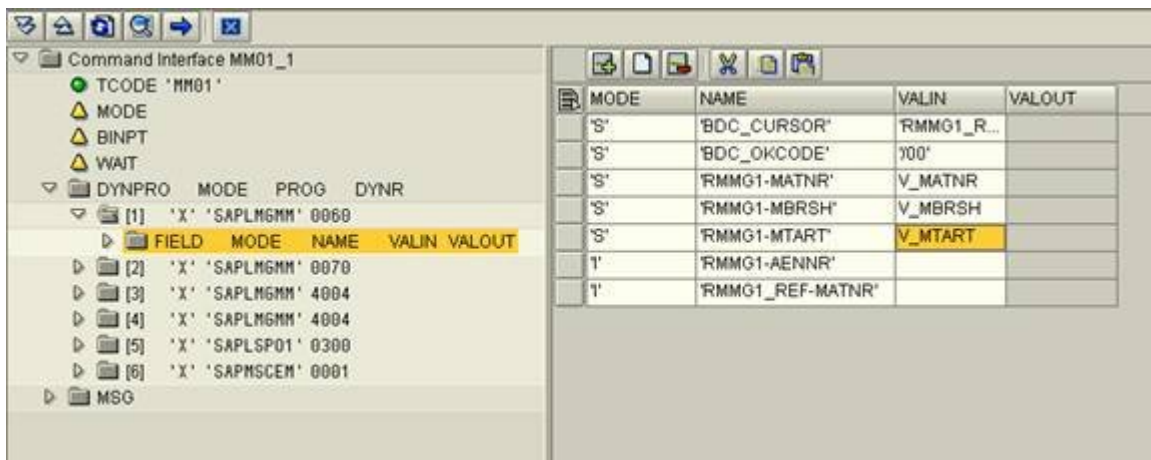
To do that again press or double click on MM01_1 in TCD (MM01, MM01_1) statement. This will take you to the command interface which look like,



To assign the screen field values double click on the 'FIELD'.



Then replace the screen **VALIN** values with the local variable names. After changing the values the interface look like this,



Repeat the above step with all the screen field values. With this we have finished the process of declaring and assigning variables.

Now we will see how to program and run the script successfully.

To write the program we need to follow two steps. They are,

1. Get the number (count) of records in file.
2. Loop through count number of times and load the data from the file pass to TCD through local variables of eCATT.

Let us see how to handle the things,

By using **ABAP...ENDABAP** statements we can do that.

Before we start writing the program we need to declare some local variables (to run the example program given) which help us to store values between two ABAP blocks. Because in eCATT each ABAP...ENDABAP block creates different internal function module with different contexts. So we need to declare some intermediate variables they are,

Parameter	Description	Value	IEV	Parameter Reference	Test System	ABAP T...	Length	Dec...	Group
COUNT			V			I	10		
FILE_V	IBIP: Data transfer parameters	<INITIAL>	V	IBIPPARMS					
INT_LOC		1	V			I	10		

COUNT : Holds the number of records.

FILE_V : Holds file path of the test data file.

INT_LOC : Index to read next from the file.

I am giving sample code to get the number of records from file in eCATT. Use this code and try for MM01 for Basic view. It will work fine.

This is very simple ABAP code. For understanding purpose necessary comments are provided.

Step 1:

First ABAP...ENDABAP block, to get the number of records from the file and store the value in the COUNT local variable.

*** ABAP BLOCK TO GET THE NUMBER OF TEST CASES**

ABAP.

*** TOT : holds total number of records**

*** FILE: holds file path of test file**

DATA : TOT TYPE I VALUE 0,
FILE TYPE STRING.

*** ITAB TO HOLD TEST DATA FROM FILE**

DATA : BEGIN OF I_MARA OCCURS 0,
V_MATNR LIKE RMMG1-MATNR, " Material Number
V_MBRSH LIKE RMMG1-MBRSH, " Industry Sector
V_MTART LIKE RMMG1-MTART, " Material Type
*** Basic View**
V_MAKTX LIKE MAK1-MAKTX, " Material Description
V_MEINS LIKE MARA-MEINS, " Basic Unit of Measure
END OF I_MARA.

*** TO OPEN FILE DIALOG FOR TEST DATA FILE**

CALL FUNCTION 'F4_FILENAME '
EXPORTING
PROGRAM_NAME = SYST-CPROG
DYNPRO_NUMBER = SYST-DYNNR
FIELD_NAME = 'FILE'
IMPORTING
FILE_NAME = FILE_V-PATH.
FILE = FILE_V-PATH.

* LOADING DATA FROM THE FILE

```
CALL FUNCTION 'GUI_UPLOAD '  
  EXPORTING  
    FILENAME          = FILE  
    HAS_FIELD_SEPARATOR = 'X'  
  TABLES  
    DATA_TAB         = I_MARA.
```

* GETTING NUMBER OF RECORDS IN THE TABLE

```
DESCRIBE TABLE I_MARA LINES TOT.
```

* STORING NUMBER OF RECORDS IN LOCAL VARIABLE

```
COUNT = TOT.
```

* CLEARING INTERNAL TABLE

```
CLEAR I_MARA.  
ENDABAP.
```

Step 2:

Looping through the records count number of times and reading from the internal table and passing them to the screen field values.

This sample code explains how to read, and pass values to the screen.

* LOOPING THROUGH (COUNT) NUMBER OF RECORDS

```
DO (COUNT ).
```

```
ABAP.
```

* V_READINDX : holds index number to read the internal table

* FILE: holds file path of test file

```
DATA : V_READINDX TYPE I,  
       FILE TYPE STRING,  
       INDX TYPE I VALUE 0.
```

* ITAB TO HOLD TEST DATA FROM FILE

```
DATA : BEGIN OF I_MARA OCCURS 0,  
       V_MATNR LIKE RMMG1-MATNR, " Material Number  
       V_MBRSH LIKE RMMG1-MBRSH, " Industry Sector  
       V_MTART LIKE RMMG1-MTART, " Material Type  
       V_MAKTX LIKE MAKT-MAKTX, " Material Description  
       V_MEINS LIKE MARA-MEINS, " Basic Unit of Measure  
     END OF I_MARA.
```

* WORKAREA TO HOLD THE I_MARA DATA

```
DATA : WA LIKE I_MARA.  
FILE = FILE_V-PATH.
```


* LOADING MASTER DATA FROM THE FILE

```
CALL FUNCTION 'GUI_UPLOAD'
  EXPORTING
    FILENAME          = FILE
    HAS_FIELD_SEPARATOR = 'X'
  TABLES
    DATA_TAB         = I_MARA.
```

* INT_LOC : is a local variable hold the current index to read I_MARA
V_READINDX = INT_LOC.

* READING I_MARA USING ITS INDEX

```
READ TABLE I_MARA INDEX V_READINDX INTO WA.
```

* assigning work area values to the screen field values

```
V_MATNR = WA-V_MATNR. " Material Number
V_MBRSH = WA-V_MBRSH. " Industry Sector
V_MTART = WA-V_MTART. " Material Type
V_MAKTX = WA-V_MAKTX. " Material Description
V_MEINS = WA-V_MEINS. " Basic Unit of Measure
```

```
ENDABAP.
```

* TCD Transaction //////////////////////////////////

```
TCD ( MM01 , MM01_1 ).
```

* move index position by one

```
INT_LOC = INT_LOC + 1.
ENDDO.
```



SCAN ME

With this we have finished programming. Finally we need to prepare the test data file and execute the program either in Foreground or Background mode.

Please note that data in test file should resemble the order of the elements in the internal table. Other wise it won't work.

To execute the given test script, follow the steps, copy the code given and declare the necessary variables.

Disclaimer:

These programs are tested and running well on many, but not all SAP versions. The author is not responsible for any data loss or other kind of damage originating from the unintentional or intentional misuse of these programs. The person, who implements these programs, have to be sure that such damage does not happen.

Thankyou

Prasad Sri K

