

FAST for ENV 6.x
Framework for Application and System Testing for ENV 6.x

Edition 1.400 - 6 February 2002

1

Overview

- Purpose
- Test Case Specification
- Test Automation Requirements
- Framework Design
- Framework Implementation
- Test Execution
- Capturing Results

February 6, 2002 FAST for ENV 6.x 1.500

© Eurocontrol - Stefan Steurs 2

Framework Purpose (1)

- System Testing:
 - behavioural and use case based
 - syntax and semantics (but not all)
 - endurance and application reliability
 - multi-user (threading, dead-locks, interaction)
- In an efficient and maintainable way

Framework Purpose (2)

- Automate Test Specification, Execution, and Follow-up
 - by Building WinRunner Scripts to:
 - allow data-driven or keyword driven tests
 - in a modular and framework based way
 - using re-usable functions
 - by integrating with TestDirector
 - one click test execution
 - collecting test results and monitoring progress

Test Specification Requirements

Test Case Structure
Test Case Template

5

Test Specification

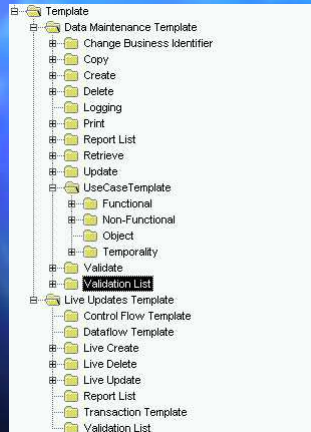
- Create a Template to build Test Specifications from Use Cases
- Create a Structure of Tests

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

6

Template Structure



- The structure repeats generic elements
- The structure is copy/pasted for each concept that requires testing

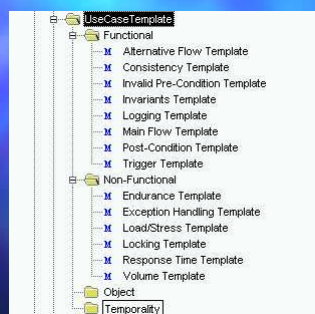
February 6, 2002

FAST for ENV 6.x 1.500

7

© Eurocontrol - Stefan Steurs

Use Case Template



- Based on Standard Use Case Template
- Grouped in
 - 4 Categories
 - with Basic Elements

February 6, 2002

FAST for ENV 6.x 1.500

8

© Eurocontrol - Stefan Steurs

Documentation

- The template is fully documented and self-explanatory
- Test Documentation is generated from the tool

Future Extensions

- If we can afford it:
 - generate test cases from use cases, a simple WORD import is not sufficient
 - integrate with modelling tools
 - integrate with Configuration Management (was not supported)

Pro's and Con's

- Requires a spec
- Requires discipline
- Up-front investment
- Investment in tool is expensive
- Imposes Structure
- Provides Overview
- Saves some intellectual work
- Uniform across participants
- Allows Test Planning and Follow-up

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

11

Test Automation Requirements

Test Execution:
Data Driven
Keyword Driven

12

Data Driven

- Separate Data from Scripts
- Data contained in separate files
- Data can be manually entered or generated/extracted

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

13

Data Driven Approach

- Data is read from Data Tables which are contained in Excel .xls files

Order of columns is not important = easy to adapt and maintain

Test can be expected to pass or fail. Fail produces an error message!

TestNr	Control	Predicate	AbId	ReferenceDate	PeriodIndex	AbName	AbNote	PointIndex	AbPoints	TestResult	Comment	DmrId
1	Name	Ab Name	220EG	2001/03/01	1	InTest Name				Pass	Modify Name	009
2	Note	Ab Note	220EG	2001/03/01	1	InTest Note				Pass	Modify Note	009
3	ModifyPoint	Ab Polygon	220EG	2001/03/01	1			1	000000N/0000000E	Fail	Start and End	009
4	ModifyPoint	Ab Polygon	220EG	2001/03/01	1			2	900000N/3600000E	Fail	Invalid Co-	009
5	ModifyPoint	Ab Polygon	220EG	2001/03/01	1			2	/	Fail	Empty Co-	009
6	ModifyPoint	Ab Polygon	220EG	2001/03/01	2	InTest Name				Pass	Modify second	009
7	ModifyPoint	Ab Polygon	101EV	2001/03/01	1			7	561600N/0240000E	Pass	Modify Point	009
8	AddPoint	Ab Polygon	101EV	2001/03/01	1			after 8	562100N/0237000E	Pass	Insert Point	009

Each test has a unique number (or label if more useful)

Control is an action word which is used to decide what kind of test to do

Column Header = Parameter Name for WinRunner. Each parameter has its own column

Reading Data

```
#####
#
# Initialise Airblock Data File
#
ret_value=ddt_open( ActFileHandle, DDT_MODE_READ );
if ( ret_value == E_FILE_OPEN )
    {
    ddt_close( ActFileHandle );# file already open
    ret_value=ddt_open( ActFileHandle, DDT_MODE_READ );
    }
ret_value+=ddt_get_row_count( ActFileHandle, ActMaxRow);

if ( ret_value==E_OK )
    {
    ActMessage="AirBlock Test Data File=" & ActFileHandle;
    Log_Activity(LogFile, ActMessage);
    }
else
    {
    ...
    }

#####
#
# Perform Tests from Start to Finish
# Read the data
# Call Interaction Functions with AB Chmi.Env
# Log Results
#
...
for ( ActRow=1; ActRow <= ActMaxRow; ActRow++ )
    {
    # Read Data
    ActTestNr=ddt_val_by_row( ActFileHandle, ActRow, "TestNr");
    ActControl=ddt_val_by_row( ActFileHandle, ActRow, "Control");
    ActPredicate=ddt_val_by_row( ActFileHandle, ActRow, "Predicate");
    ActAbId=ddt_val_by_row( ActFileHandle, ActRow, "AbId");
    #
    ActRefDate=ddt_val_by_row( ActFileHandle, ActRow, "ReferenceDate");
    ActAbValPeriodStart=ddt_val_by_row( ActFileHandle, ActRow, "AbValPeriodStart");
    ActAbValPeriodEnd=ddt_val_by_row( ActFileHandle, ActRow, "AbValPeriodEnd");
    ActAbName=ddt_val_by_row( ActFileHandle, ActRow, "AbName");
    ActAbNote=ddt_val_by_row( ActFileHandle, ActRow, "AbNote");
    ActAbPolygonSeries=ddt_val_by_row( ActFileHandle, ActRow, "AbPolygonSeries");
    ActExpectedResult=ddt_val_by_row( ActFileHandle, ActRow, "TestResult");
    ActComment=ddt_val_by_row( ActFileHandle, ActRow, "Comment");
    ActEnvArea=ddt_val_by_row( ActFileHandle, ActRow, "EnvArea");
    ActDmrId=ddt_val_by_row( ActFileHandle, ActRow, "DmrId");
    }
```

February 6, 2002

FAST for ENV 6.x 1.500

15

© Eurocontrol - Stefan Steurs

Data Driven means:

- The dialogue flow is encoded
- The variation is limited to the data
- Need many scripts to cover all UC flows
- Many columns = difficult to maintain
- + Scripts can be re-used for all data combinations
- + Easy to add tests (copy/paste/adapt)
- + Tests **can** re-use output from other tests
- + Test Data is not hard-coded

February 6, 2002

FAST for ENV 6.x 1.500

16

© Eurocontrol - Stefan Steurs

Keyword Driven

- Separate Data and Navigation from Script
- Data contained in separate files
- Data can be generated (but not easily extracted)

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

17

Keyword Driven Approach

Concept	Action	Parameter Value	ExpResult	Comment
AB	clear		Pass	
AB	get	Type Criteria Date AIRAC Idp \$AbExists\$ EnvArea OPS	Pass	Query AB by criteria
AB	clear		Pass	
AB	create	Date AIRAC Id 001EB EnvArea Preparation	Pass	Create AB
AB	selectdmr	Dmrid DMR-1	Pass	Select DMR
AB	define	PeriodStar AIRAC PeriodEnd AIRAC Points 504700N/0030900E;504100N/0032600E;503 Name IST 001EB Note AB Created by Independent Testing Id 001EB	Pass	Enter Details
AB	save	Id 001EB	Pass	Save the Airblock
AB	clear		Pass	

The concept identifies which suite of functions to access

Each action is considered a test. Action words should be generic over concepts.

Each action accepts a set of parameters, not all parameters are needed, some are mandatory others are optional, order is not important.

Use Global Parameters to manage Test Data dependencies

Each test is expected to pass or fail. Results are logged in correspondence to expected outcome.

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

18

Reading Keywords/Data

- Look at the code
 - Read Concepts
 - Read Actions
 - Read Parameters
 - Execute Interactions
- Script above script - says what to do not how

Keyword Driven means:

- + still data driven
- + flexible dialogue flow
- more can go wrong if script is wrong
- not so easy to re-use output from other tests (or SQL queries)
- requires more robust test scripting

Example in Excel

Concept	Action	Parameter	Value	ExpResult	Comment
AB	clear			Pass	
AB	create	Date Id EnvArea	AIRAC 819EB Ops	Pass	Create AB
AB	selectdmr	Dmrid	DMR-1	Pass	Select DMR
AB	define	PeriodSta PeriodEnc Points Name Note Id	AIRAC+1 AIRAC+1 504700N/0030900E;504100N/0032600E;503110N/0032905E;502942N/0032236E;503051N/0032137E;503530N/0031630E IST 819EB AB Created by Independent Testing 819EB	Pass	Create Ab - OPS Area not allowed
AB	save	Id	819EB	Fail	Save the Airblock

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

21

Test Framework Design

Framework Based Implementation

22

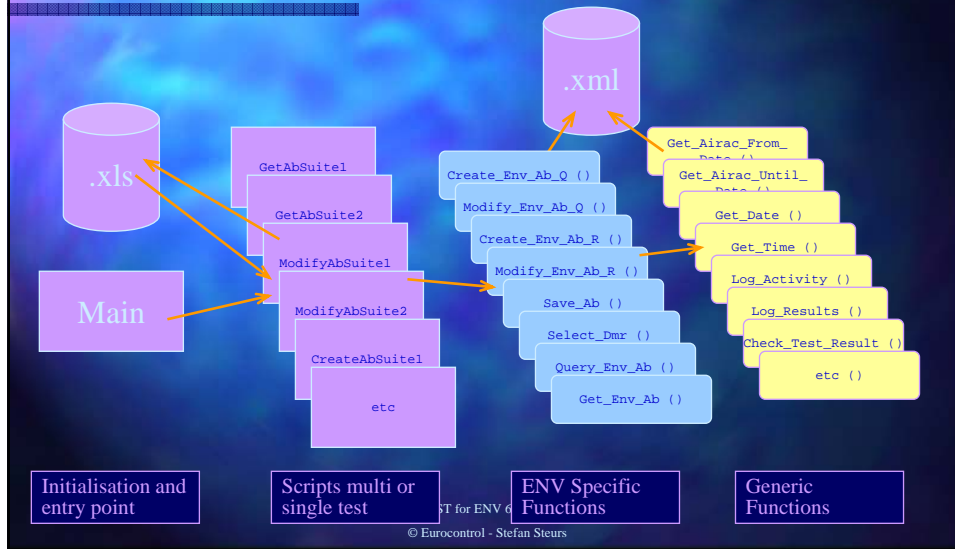
Framework Based Approach

- Modular and re-usable functions
- Localise interactions to limit coupling to AUT
- Data from tables has to be passed to functions in a “maintainable” way

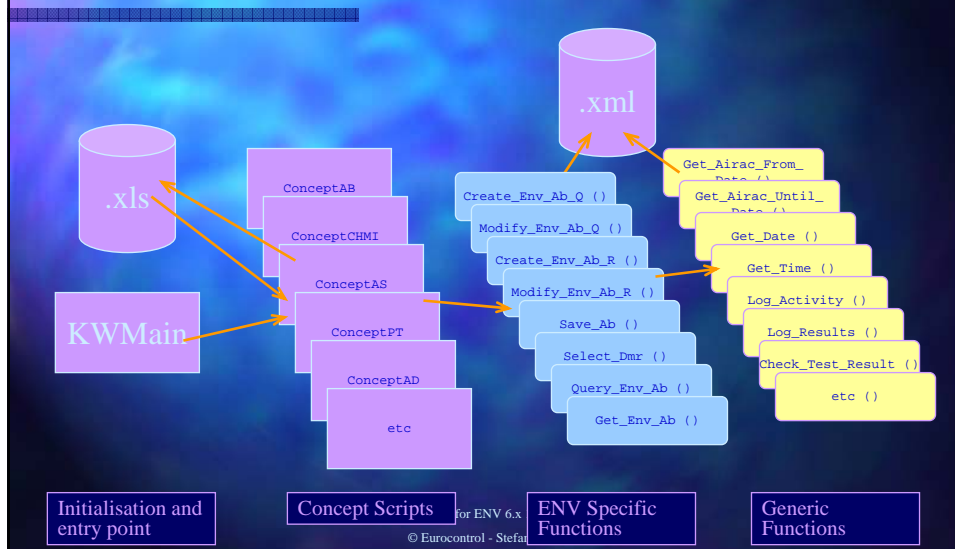
Framework Based Functions

- Accept parameter list
- Parse and Process parameters
- Check pre-condition if any
- Interact with GUI (NOT test specific)
- Check for result if any
- Return result of function execution

Framework Structure Data-driven



Framework Structure Keyword Driven



Framework Properties

- Test Scripts don't know about window details
- Interactions with ENV are isolated in functions for all generic interactions = easier to maintain
- Some specific Interactions are still part of the test script but may be modularised when needed

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

27

Framework Robustness

- Anticipate failure
- Cannot recover from all failures
- When test fails it is abandoned but next test is attempted
- Some failures stop tests altogether (abort)
- Tests have to run unattended

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

28

Framework Sensitivity

- Only mandatory elements can be asserted "safely" without too much intelligence - Test Oracle problem
- E.g. Response functions may assume:
 - there is at least 1 validity period
 - there is a polygon definition
- Comparisons with reference XML objects? For a future version

Using Assertions

- Assertions are powerful ways to make in-line observations
 - condition true before you start a test? (pre-condition)
 - condition true after you did a test? (post-condition)
 - condition true before and after you did a test (invariant)

Test Automation Framework Implementation

Writing Test Functions

31

What are we after

- Re-usable approaches
- Re-usable software
- Lower Maintenance Cost

- Sounds familiar?

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

32

Re-Usable Approaches

- See requirements for Automation
- Split up in Concepts and Functions
 - How do we group concepts?
 - How do we group functions?
- What do we start from?
 - SRD / Use Cases / Stories
 - Conceptual Models

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

33

Re-usable

- "Main", "KWMain", and test selection
- Parsing of Data and Keyword files
- Initialisation of the tool
- Initialisation and Start-up of AUT
- Loading of functions
- Logging of Activities and Results
- Terminating AUT and clean-up

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

34

Maintainability

- In a keyword driven approach - only 1 script "KWMain" to link to Test Management Tool
- Concepts and Functions isolate test scripts from "flow" in AUT
- GUI Files (WinRunner) isolate test scripts from "MMI design" in AUT

Typical Structure

- Initialise (AUT, Logs, WinRunner)
- Open data file and Parse Data
- Partition in Concepts
 - make assertions
 - Perform Interactions and Call Test Functions
 - make more assertions
- Validate Interactions for "pass" or "fail"
- Close data file and Clean up (AUT, Logs, WinRunner)

Initialise

- Set Test Tool Runtime Configuration
- Load the Functions that the test requires
- Start-up the AUT
- Set the initial state of AUT and other apps that are required
- Open Logs

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

37

Parse and Pass Data

- Open Datafile
- [validate data - not yet implemented but could]
- Parse the data
- Pass data to functions
- Loop until end of script is reached

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

38

Parameter Passing

```
public function Modify_Env_Ab_Q ( inout Parameters[] )
{
# Parse parameters
Index=0;
do
{
split( Parameters[Index], KeyValue, " "
switch ( tolower(KeyValue[1]) )
{
case "id":
ActAbId=KeyValue[2];
break;
. Etc .
}
Index++;
} while (Parameters[Index]!="$*");
```

```
Parameters[0]="Id=" & ActAbId;
Parameters[1]="Date=" & ActDate;
Parameters[2]="$*";
ReturnValue=Modify_Env_Ab_Q( Parameters );
```

I am not a great programmer
;-)

Make sure we stop parsing
parameters

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

Global Parameter Substitution

- Use a separate file that contains parameter value pairs
- When you find a \$parameter\$ look in the file and replace it by the value

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

40

Concepts and Functions

- Use concepts to group related functions
- Functions separate test script from GUI by:
 - using logical names i.s.o. physical
 - creating 1 function per interaction => single point of maintenance
- Functions do assertions and validations and report success, failure, or abort

Test Results Processing

- Assertions and Validations allow to conclude Pass / Fail / Fatal
- Exceptions - can you recover or not
- In-line vs. Off-line results comparison

Pass/Fail/Fatal?

```
#      ExpResult  ActResult  OutResult
#      Pass      E_SUCCESS  Pass
#      Pass      E_FAIL     Fail
#      Pass      E_FAIL_NO_ERROR  Fail
#
#      Fail      E_SUCCESS  Fail
#      Fail      E_FAIL     Pass
#      Fail      E_FAIL_NO_ERROR  Pass
#
#      Don't Care  E_NOT_EXPECTED  Fatal
#      Don't Care  E_KEYWORD_NOT_EXIST  Fatal
#      Don't Care  E_FATAL        Fatal
#      Don't Care  E_NO_REPLY     Fatal
#      Don't Care  Invalid Value  Fatal
```

- Use a decision table
- Define exceptions within your context
- When "Fatal" abort the test
- [Example script](#)

Teardown

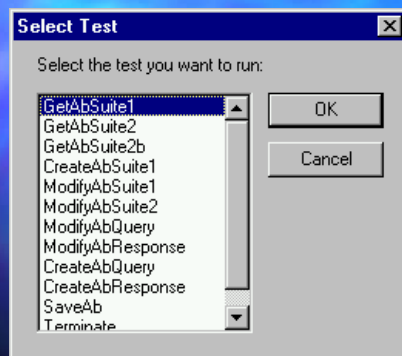
- Clean-up the state of the AUT
- Terminate the Logs
- Restart AUT if necessary (can also be in Setup or the script itself)

Test Execution

Stand-alone
or
Integrated with TestDirector

45

Main Test Start-Up (Data Driven)



- Test is defined by a string
- Can be a suite of tests or a single test
- Can be a single function for debugging
- Not integrated with TestDirector

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

46

KWMain Test Start-Up (Keyword Driven)



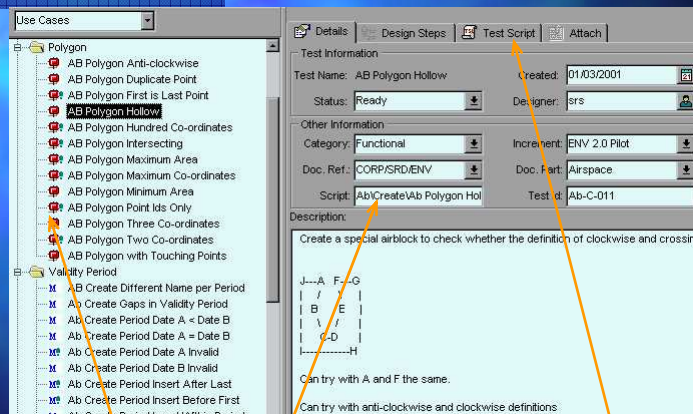
- "Open" file - browse list of .xls files
- No file selection = terminate test
- Filenames are meaningful
- Integrated with Test Director

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

47

TestDirector Integration (Keyword Driven)



Automated Test

Keyword Script

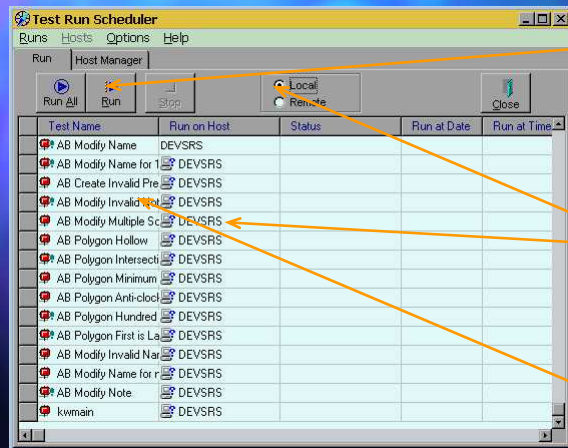
WinRunner KWMain script

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

48

TestDirector Test Execution (Keyword Driven)



Click and Run

Local / Remote

Automated Test

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

49

Capturing Results

Output Collection and
Comparison, Results Collection
and
Progress Monitoring

50

Test Output Collection

- Log Files to gather information about test script execution
- Results Files to gather summary information about pass/fail
- Examples in Internet Explorer:
 - [Ab Modify Inv Point Id_Log20011114120733.xml](#)
 - [Ab Modify Inv Point Id_Result20011114120521.xml](#)

Test Output Comparison

- Freeware tool: [ExamDiff](#)
 - Select the XML files -> textual comparison
- Compar from FUITE
 - compare differences in XML with a control file [tool exists but not yet integrated with test approach]

ExamDiff

```
D:\AAA_DATA\MercuryTestSuite\EnvApp\KW\Scripts_VAb Create Invalid Precondition_Log20010831095138.xml D:\AAA_DATA\MercuryTestSuite\EnvApp\KW\Scripts_VAb Create Invalid Precondition_Log20010906081242.xml
51 <concept>AB</concept><action>define</action><parameters><pv>period</pv></action></message> 41 <concept>AB</concept><action>define</action><parameters><pv>period</pv></action></message>
52 </message> 42 </message>
53 <message> 43 <message>
54 <time>2001/08/31 09:52:31</time> 44 <time>2001/09/06 08:13:24</time>
55 <description><action>Define Details AB </action><result>Pass</result></message> 45 <description><action>Define Details AB </action><result>Pass</result></message>
56 </message> 46 </message>
57 <message>
58 <time>2001/08/31 09:52:32</time>
59 <description>Row 16 in File: D:\aaa_data\MercuryTestSuite\EnvApp
60 <concept>AB</concept><action>save</action><parameters><pv>id=819e
61 </message>
62 <exception>
63 <time>2001/08/31 09:52:36</time>
64 <details>
65 <test>D:\aaa_data\MercuryTestSuite\EnvApp\KW\EnvAbCreate</test>
66 <function>Save_Env_Ab </function>
67 <time>571</time>
68 <type>Info</type>
69 </details>
70 <description>Error Window found </description>
71 </exception>
72 </message>
```

Difference highlighted:
"Error window found"

Test Result Collection

- Status info at end of log
- Status info within TestDirector (progress monitoring is included)
- Tester activities in WORD
- Failure and Incident Reporting (in CM tool rather than TestDirector)

Status Info - Log

```

- <message>
  <time>2001/09/05 11:33:35</time>
  <description>Test Script Closed</description>
</message>
- <message>
  <time>2001/09/05 11:33:35</time>
  <description>
  - <teststatistics>
    <pass>4</pass>
    <fail>0</fail>
    <exceptionnotraised>0</exceptionnotraised>
    <exceptionnotexpected>0</exceptionnotexpected>
    <resultsmismatch>0</resultsmismatch>
    <noreply>0</noreply>
  </teststatistics>
  </description>
</message>
</log>

```

Pass/Fail
Type of Failure

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

55

Status Info - TestDirector

Test Name	Status
Ab Create Period Date A Invalid	Passed
Ab Create Invalid Note Syntax	No Run
Ab Create WR Test	Failed
Ab Create Period Shortest	Failed
Ab Create Period Longest	Passed
Ab Create Period Max Nr Of	Passed
Ab Create Period in the Past	Failed
Ab Create Period in the Future	Passed
Ab Polygon Two Co-ordinates	Passed
Ab Polygon Three Co-ordinate	Passed
Ab Polygon Point Ids Only	Passed
Ab Polygon Maximum Co-ordin	Failed
Ab Polygon Maximum Area	Passed
Ab Polygon with Touching Poi	Failed
Ab Polygon Duplicate Point	Passed
Ab Create Synonym	Failed
Ab Create Homonym	Passed
Ab Create Period Insert After	Failed
Ab Create Period Insert Before	Failed
Ab Create Period Insert Within	No Run
Ab Modify Period Delete First	Failed
Ab Modify Period Delete Last	Failed

- Pass: Actual = Expected Result
- Fail: Actual != Expected Result

Pass/Fail
What does Pass/Fail Mean?
Tools do not replace intellect.

- When "Pass" becomes "Failed"

February 6, 2002

FAST for ENV 6.x 1.500
© Eurocontrol - Stefan Steurs

56

Generic Logging Functions

Activity Log
Results Log

57

Activity Logging

- Activity log:
 - 1 per test or 1 per suite (time stamp in name)
 - XML style (do not use schema's or style sheets yet)
 - Recording activities
 - Log_Activity
 - Log_Exception
 - Closing activity log

```
ReturnValue=Get_Date(ActDate);  
ReturnValue=Get_Time(ActTime);  
LogFile=LogPath & LogFileName & "_" &  
substr(ActDate,1,4) & substr(ActDate,6,2) &  
substr(ActDate,9,2) &  
substr(ActTime,1,2) & substr(ActTime,4,2)  
& substr(ActTime,7,2) ".xml";  
ReturnValue=Init_Log( LogFile );
```

February 6, 2002

FAST for ENV 6.x 1.500

58

© Eurocontrol - Stefan Steurs

Example Log_Activity()

```
ActMessage="Modify Ab Test Nr:" & ActRow &  
" for <abid>" & ActAbId & "</abid>" &  
"<refdate>" & ActDate & "</refdate>";  
Log_Activity(LogFile, ActMessage);
```

```
<message>  
<time>2001/05/11 09:15:07</time>  
<description>Modify Ab Test Nr:29 for  
<abid>004EB</abid><refdate>2001/05/11</refdate></description>  
</message>
```

Log_Exception ()

- Log_Exception(filehandle, type, exception):
 - See Log_Activity
 - Additional parameter - type - to classify exception as info, warning, or error
 - Function outputs calling function, test name of calling function, and linenumber (for debugging)

Example Log_Exception()

```
if ( ReturnValue != E_SUCCESS )
{
    ActMessage="Modify Ab Test Nr:" & ActRow &
    " for <abid>" & ActAbId & "</abid>" &
    "<refdate>" & ActDate & "</refdate>" &
    "<comment>note field changed only</comment>" &
    "could not be made.";
    ActType="Error";
    Log_Exception(LogFile, ActType, ActMessage );
}
```

XML tags

Each function returns:
E_SUCCESS or
E_FAIL

```
<exception>
<time>2001/05/11 09:15:20</time>
<details>
<test>D:\aaa_data\MercuryTestSuite\EnvApp\EnvAbCreate</test>
<function>Save_Env_Ab </function>
<line>956</line>
<type>Error</type>
</details>
<description>Expected a failure but no Error Dialogue window could be
detected.</description>
</exception>
```

February

© Eurocontrol - Stefan Steurs

Results Logging

- Open results log:
 - 1 per test or 1 per suite (with date time stamp)
 - XML style
 - Recording activities
 - Log_Results
 - Closing results log
 - Differs from activity log in that it only should contain test results

February 6, 2002

FAST for ENV 6.x 1.500

62

© Eurocontrol - Stefan Steurs

Example Log_Results()

```
SetFail++;
SetResultsMismatch++;
TestResults="<testresult>" & ActTestResult & "</testresult>" &
  "<abdetails>Details for <abid>" & ActAbId & "</abid>\n" &
  "<abname>" & GetAbName & "</abname>" &
  "<abnote>" & GetAbNote & "</abnote>" &
  "<nrvalperiods>" & GetAbValPeriodCount & "</nrvalperiods>" &
  "<nrpoints>" & GetAbPolygonCount & "</nrpoints>" &
  "</abdetails>";
ReturnValue=Log_Results(ResultsFile, TestResults);
```

```
<result>
<time>2001/04/30 07:42:04</time>
<details>
<testresult>Pass</testresult><abdetails>Details for <abid>003BI</abid>
<abname>DELEGATION TO EGPX</abname><abnote><datetime>2001/04/30
07:41:05</datetime>ModifyAbSuite1
Test</abnote><nrvalperiods>2</nrvalperiods><nrpoints>10</nrpoints></abdetails></details>
>
</result>
```

```
SetPass=0; # Nr of Passed Tests
SetFail=0; # Nr of Failed Tests
SetExceptionNotRaised=0;
# Nr of Tests which were expected to end in exception but no exception raised
SetExceptionNotExpected=0;
# Nr of Tests which were expected to pass but raised an exception
SetNoReply=0; # Nr of Tests for which we did not get any reply
SetResultsMismatch=0; # Nr of Tests for which the response does not conform to the expected result
```

February 6, 2002

FAST for ENV 6.x 1.500

63

© Eurocontrol - Stefan Steurs

Evolution

Mandatory Features

Optional Features

64

To do - Mandatory

- To be done:
 - Extend coverage to all ENV concepts
 - Add scripts and commands to allow testing all HMI features
 - Investigate how to automate “Graphical” testing
 - Configuration Management of scripts and test data

To do - Optional

- Nice to have:
 - Make installation easy and friendly
 - **Java method interaction?**
 - Clean-up of code

Future

- Use for regression testing by developers
- Use for acceptance testing by users
- Integration with models

Current Limitations of Use

- Java RunTime 1.3.0 (1.3.1 not supported yet)
- WinRunner 6.02
- TestDirector 6.02