

Testimony - Testers' Guide

2.21 — Last update: 2020/03/24

Basis Technologies

Table of Contents

1. Introduction	2
2. Glossary	3
3. Defect Management.....	6
3.1. What is a Testimony defect?	7
3.1.1. The defect header	8
3.1.2. Defect comments	10
3.1.3. Defect history	11
3.1.4. Execution queue steps	12
3.2. Reviewing and prioritising defects	13
3.2.1. The defect overview	14
3.2.2. Prioritising defects	16
3.3. Investigating defects	18
3.3.1. Common areas of the Investigate Screen.....	20
3.3.1.1. Header Section	21
3.3.1.2. Script Steps	23
3.3.1.3. Related Linkages	24
3.3.1.4. Expected vs Actual	26
3.3.2. Object type screens	28
3.3.2.1. Dialog Transactions	29
3.3.2.2. Batch Jobs.....	30
3.3.2.3. Inbound RFC	32
3.3.2.4. Inbound Web-Services.....	33
3.3.3. Linkage Type Screens	34
3.3.3.1. Change Documents	35
3.3.3.2. SAP Script Forms	36
3.3.3.3. Inbound / Outbound IDocs	38
3.3.3.4. Number Range	40
3.3.3.5. Front-end Actions	41
3.3.3.6. Local Files	42
3.3.3.7. Application Server Files	43
3.3.3.8. SET/GET parameters	44
3.3.3.9. User Preferences	45
3.3.3.10. Dynamic ID	46
3.3.3.11. Dynamic ID (User Preferences)	48
3.3.3.12. Spool Requests	49

3.3.3.13. Clipboard Imports	50
3.4. Types of defects and how to identify them.....	51
3.4.1. Defects caused by changes in the release	52
3.4.1.1. Example 1: A change to a message severity resulting in different output in a batch job	53
3.4.1.2. Example 2: Authorisation failures.....	56
3.4.2. Defects arising from data issues	58
3.4.2.1. Testimony playback sequencing	59
3.4.2.2. Previously failed transactions	61
3.4.2.3. Example 3: Batch job with different output	62
3.4.2.4. Example 4: Inbound RFC with different output	66
3.4.2.5. Example 5: Double-clicking on a list item calls up a different document	69
3.4.3. Defects arising from sequencing issues	75
3.4.3.1. Locking issues	76
3.4.3.2. Example 6: A lock in the recording not seen in the playback	77
3.4.4. Defects arising from bot configuration issues	79
3.4.4.1. Microsoft Office installation	80
3.4.4.2. Example 7: Excel button not available on screen	81
3.4.4.3. Example 8: No word processing program available.....	83
3.4.5. Defects arising from file download virtualisation	85
3.4.5.1. Example 9: No message is displayed.....	86
3.4.5.2. Example 10: “Setting was applied” message is displayed	88
3.4.6. Defects caused by functionality not supported by Testimony.....	89
3.4.6.1. Example 11: Drag & Drop	90
3.4.6.2. Example 12: Batch Input.....	92
3.5. Defect Suppression	94
3.5.1. Step-level suppression.....	95
3.5.1.1. Example of a step-level suppression.....	96
3.5.1.2. Creating a step-level suppression	97
3.5.2. Script-level suppression	101
3.5.2.1. Example of a script-level suppression	102
3.5.2.2. Creating a script-level suppression	103

1. Introduction

Welcome to the Testimony Testers' Guide.

This guide details the defect management aspect of Testimony, giving you hints, tips and best practices for analysing the results of a Testimony playback. It is aimed at testers as well as technical and functional experts who will be involved in this analysis. The guide contains the following sections:

- [What is a Testimony Defect?](#)
- [Reviewing and prioritising defects](#)
- [Investigating defects](#)
- [Types of defects and how to identify them](#)
- [Defect suppression](#)

This Guide should be used in conjunction with the other available Testimony documentation and templates.

Remote support is also available from Basis Technologies if required via the contact details outlined at the end of this guide.

2. Glossary

Bot

The bot is an executable which resides on a windows machine (normally a virtual machine). During playback the bot logs on as the recorded user and executes that user's transactions. The requirements for bot setup can be found [here](#).

Central System

This is the SAP system into which Testimony is installed and from which it is operated. Testimony users log on to this system to set up Testimony, start recordings and playbacks and analyse results.

Check Steps

These are run manually before a recording or playback and the operator needs to check the results of these to see if they should continue with the recording or playback.

Dynamic IDs

These are used to link scripts which use the same data, for example a purchase order number. If the creation of a purchase order fails during the playback then Testimony recognises that there is no point running a subsequent script that approves this purchase order. Testimony will therefore cancel the execution of the order approval script.

Enhancements

To record and playback Testimony has enhancements on the source or target system to enable the recording or playback to operate correctly. These are switched on before recording or playback and are automatically deactivated at the end of the recording or playback in the "Post-Processing Steps".

Execution Queue

The execution queue is built from the repository and contains the scripts to be played back.

Filtered Recording

A filtered recording is used when you want to record a small subset of users or transactions on the source system. It is typically used for testing purposes to ensure that the setup from central to source system has been completed correctly.

Filter Sets

Filter sets have two main uses: to exclude certain objects (transactions, batch jobs, etc.) from a recording; and to provide special handling of error cases during a playback. For example, if you want Testimony to ignore all occurrences of transaction SM21 from the recording, then adding this transaction to the recording

filter set will achieve this. If you want to ignore occurrences of message E123 from a particular screen, you can set this message as an exclusion in the comparison filter set. Filter sets are also available for the transfer to repository and transfer to execution queue processes, although these are less frequently used. This topic should be further studied via the Filter Sets section here. **THIS MUST BE LINKED**

Linkages

Testimony records deeper than just the UI so that objects such as change documents and number ranges are also recorded. These can then be checked at playback to ensure that these match providing a deeper level of testing.

Playback

The playback is the playing back of the scripts in the execution queue, via the bots, on the target system.

Post-Processing Steps

These are run automatically after a recording or playback is completed. Any errors in post-processing cause a hard stop preventing the status moving to complete. If errors are found then the operator should check the errors and see if they need to manually resolve these.

Preparation Steps

These are run automatically before a recording or playback starts. Any errors cause a hard stop preventing the recording or playback starting. Errors should be resolved before restarting the recording or playback

Recording

A recording (either Filtered or Standard) is the process by which actions on the source system are captured by Testimony for later playback.

Repository

The repository is a staging post for recorded transactions. Once all recorded transactions have been stored in the Central System, they are transferred to the repository (potentially with some filtering) before being transferred to the execution queue for playback.

Source System

This is the system that is recorded and therefore acts as the source for the recording. In BAU operation of Testimony, this is usually the production system.

Standard Recording

A standard recording records everything except a small number of users or objects as defined in the "Filter Sets"

Target System

This is the regression test system into which recorded scripts are played back via the Bots.

3. Defect Management

In this section of the guide we will discuss the management of defects within Testimony.

We start by looking at what a Testimony defect is and how they are created, before moving on to the tools available for defect analysis, investigation and management within Testimony. We then look at certain types of defects commonly seen in Testimony, and how they can be diagnosed and processed.

3.1. What is a Testimony defect?

During a playback, Testimony is looking for differences between outputs seen in the recording and outputs seen in the playback. For example, in a dialog transaction it will be looking for unexpected messages (a message seen in the playback that was not seen in the recording) or an unexpected next screen (where the playback navigated to a different screen than was seen in the recording). Whenever a different output is detected during the playback, Testimony marks the step as having failed, and the script as either completed (if the failure occurs on the last step) or partially complete (if the failure occurs on an earlier step).

Once the playback is completed, a defect proposal is run which groups identical failures into defects. For example, if a transaction has failed 25 times with the same error in the playback, then all 25 occurrences of this failure will be grouped into one defect.

The defect record contains lots of useful information about the defect which can be used to aid diagnosis and perform defect investigation and management functions. These are discussed in the following sections.

3.1.1. The defect header

Defect Header

Defect ID	2961	Ext	Assigned Team Role	
Current Status	Assigned	Assigned to		
Short Description	FAGLB03 : No message received. Expected - S024(MSITEM)			
		Task Priority	Unclassified	

The defect header shows you key information about the defect and the status of the record itself, including:

- Defect ID
- Assignment details
- Current status
- Defect priority
- The short description of the defect

Header | Comments | History | Execution Queue steps

Key information

Test Plan	RP3-RZ3: 2020-01	Software Component	Financial Accounting
Test Phase	0001	Sub-Component	(FI-GL) General Ledger Accounting
System	RP3 - RP3 Source system	Object Type	Dialog transaction
Task Source	Testimony	Object Name	FAGLB03
Failure Reason	Unexpected message	Task Type	Defect

SAP GUI Dialog Transaction details

Transaction Code	FAGLB03
Program Name	SAPLSVC_FULLSCREEN
Screen Number	0500
Fieldname	BDC_OKCODE


Within the Header tab, you can see information on:

- Test plan and system information
- Software component information
- The object type and name
- The headline failure reason

- Technical information (e.g., program name and screen number for dialog transactions)

3.1.2. Defect comments





Free-text comments can be entered against a defect record and displayed.

Header			
Comments			
History			
Execution Queue steps			
Comments			
Date	Time	Comment by	Comment preview
11.02.2020	11:28:38		 Need to investigate in more detail.

3.1.3. Defect history

Testimony keeps an audit trail of actions taken within a defect which can be displayed in the History tab.

Header Comments History Execution Queue steps

Testimony Event History				
Event Date	Event Time	Icon	User Name	Action
06.02.2020	07:08:56		XXXXXXXXXX	Status changed from 'Proposed' to 'Assigned'
06.02.2020	07:08:56		XXXXXXXXXX	Assigned User changed from "" to 'XXXXXXXXXX'
06.02.2020	07:08:56		XXXXXXXXXX	Source changed from "" to '01'
30.01.2020	10:15:29		XXXXXXXXXX	Defect 2961 created

3.1.4. Execution queue steps

This tab will show you the individual execution queue steps that were affected by this defect. There can be one or many steps affected by a defect.

Header Comments History Execution Queue steps								
Execution Steps for Defect								
Phase	Executi...	EQ step	Res	Result	Reason	Display Step	Investigate Test Script	Sys
0001	914	686138		Failed		GUI dialog step	FAGLB03	RP3

Defect that only affected one step

Header Comments History Execution Queue steps								
Execution Steps for Defect								
Phase	Executi...	EQ step	Res	Result	Reason	Display Step	Investigate Test Script	Sys
0001	1007	686816		Failed		GUI dialog step	ML81N	RP3
0001	3921	708252		Failed		GUI dialog step	ML81N	RP3
0001	5499	723126		Failed		GUI dialog step	ML81N	RP3
0001	5521	723215		Failed		GUI dialog step	ML81N	RP3
0001	10116	750743		Failed		GUI dialog step	ML81N	RP3
0001	13871	786712		Failed		GUI dialog step	ML81N	RP3

Defect that affected several steps

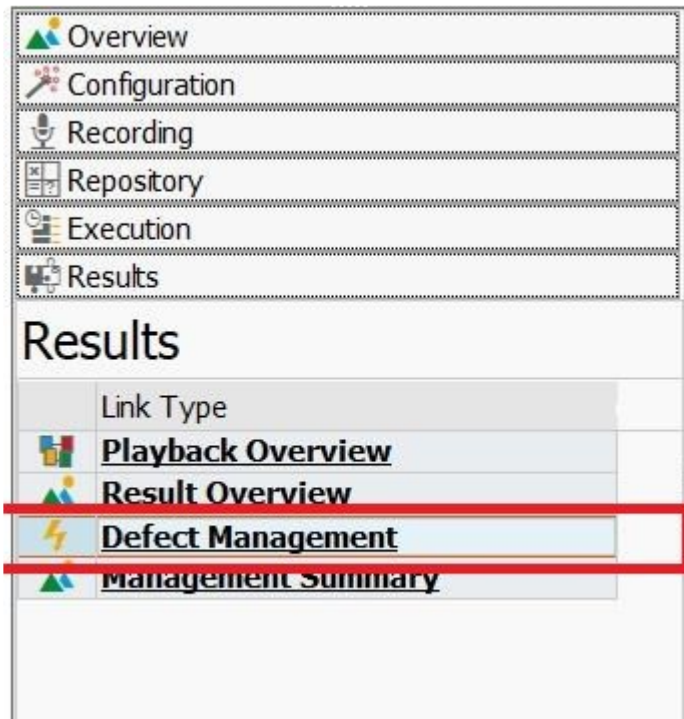
3.2. Reviewing and prioritising defects

Once defects have been created by executing the defect proposal, the first step is to review the defects at a high level and prioritise them for investigation.

The following sections introduce you to the Defect Overview (the entry point for defect management in Testimony) and discusses ways in which you may wish to prioritise your defects for investigation.

3.2.1. The defect overview

The Defect Management screen can be found in the Results section of the Testimony Context menu



Once selected, it shows you a list of all the defects generated as a result of your playback.

Defect Management													
Business Task Defect Management (28)													
Defect ID	Defect Description	Object Name	Component	Pri	Priority	Scripts	Sts	Status	Role (team)	Curr. Assignment	Object Type	Failure reason	RCA, RCA Fl.
6539	MD07 : Unexpected next screen : SAPMSSY0.0	MD07	PP-MRP-BD	Unknown	Unknown	1	Proposed				Dialog Trans	Unexpected next screen	●
6540	Change document missing in Dialog transaction	ME21N	MM-PUR	Critical	Critical	1	Proposed				Dialog Trans	Change document missing	●
6541	MD04 : Unexpected next screen : SAPMV45A.4	MD04	PP-MRP-BD	Unknown	Unknown	1	Proposed				Dialog Trans	Unexpected next screen	●
6542	IP10 : Unexpected next screen : SAPMSDYP.0	IP10	PM-PRM-TL	Unknown	Unknown	1	Proposed				Dialog Trans	Unexpected next screen	●
6543	ME29N : Unexpected next screen : SAPMSSY0.0	ME29N	MM-PUR	Unknown	Unknown	1	Proposed				Dialog Trans	Unexpected next screen	●
6544	IE03 : Unexpected next screen : SAPLSDI4.02	IE03	PM-EQM-EQ	Unknown	Unknown	1	Proposed				Dialog Trans	Unexpected next screen	●
6545	Change document missing in Dialog transaction	XX02	FI-AP-AP	Unknown	Unknown	1	Proposed				Dialog Trans	Change document missing	●
6546	EBLSN : Unexpected message : S00Z(MSITEM)	FBL5N	FI-GL-15	High	High	2	Assigned				Dialog Trans	Unexpected message	●
6547	IDoc is different in Dialog transaction WE19	WE19	BC-MID-ALE	Unknown	Unknown	1	In Progress				Dialog Trans	IDoc is different	●
6548	Change document missing in Dialog transaction	FBL1N	FI-GL-15	High	High	1	Proposed				Dialog Trans	Change document missing	●
6549	CJ20N : Unexpected message : S221(CJ)	CJ20N	PS-ST-OPR-PB	Unknown	Unknown	1	Proposed				Dialog Trans	Unexpected message	●
6550	UDM_SUPERVISOR : Unexpected message : EQ	UDM_SUPERVISOR	FIN-FSCM-COL	Unknown	Unknown	1	Cancelled				Dialog Trans	Unexpected message	●

Useful information contained here includes:

- Defect description: This contains the affected object (transaction code, batch job name, etc.) and a summary of the issue (e.g., “Unexpected message” or “Unexpected next screen”).
- Priority: This is set to the default of “Unknown” by the defect proposal run, but can be changed to indicate prioritisation
- Status: The current status of the defect, which can be one of the following:

- Proposed: This is the status given to all defects by the defect proposal run. All other statuses need to be manually set during defect investigation
- New: Can be used indicated that a defect is under initial review (for example during the Basis triage)
- Assigned: Has been assigned to someone but has not yet been picked up
- In Progress: Is being investigated
- Complete: Has been resolved
- Cancelled: Has been cancelled as this is not a genuine defect
- Curr. Assignment: The Testimony user to whom the defect has been assigned for investigation.
- RCA: Root Cause Analysis of Defect (if enabled)

It is possible to filter the list so that you only see, for example, the defects that are assigned to you, or those for a particular transaction code.

Defect Management			
Business Taxent (28)			
Defect ID	Defect	Object Name	Object Name
6539	MD0	APMSSY0 0	MD07
6540	Char	transaction	ME21N
6541	MD0	APMV45A 4	MD04
6542	IP10	APMSDYP 0	IP10
6543	ME2	SAPMSSY0	ME29N
6544	IE03	PLSDH4 02	IE03
6545	Char	transaction	XK02
6546	FBL5	07(MSITEM)	FBL5N
6547	IDoc	ion WE19	WE19
6548	Char	transaction	FBL1N
6549	CJ20	21(CJ)	CJ20N
6550	UDM	SUPERVISOR : unexpected message : E0	UDM_SUP

3.2.2. Prioritising defects

Analysing defects based on some kind of priority is a useful way of ensuring that you begin your investigations by focussing on the most important, or most heavily-impacted, transactions. There are various different ways of prioritising defects.

Prioritisation by number of affected scripts

From Testimony v 2.21 onwards it is possible to see the number of scripts affected by each defect from within the Defect Management screen.

Business Task Defect Management (7)

Defect ID	Defect Description	Object Name	Component	Pri	Priority	Scripts	Sts	Status
5802	Difference in output values in RFC BAP...	BAPI_BUPA_ROLES_GET	AP-MD-BP	⊗	Unknown	541	🚩	Proposed
5803	Difference in output values in RFC BAP...	BAPI_USER_GETLIST	BC-SEC-USR-ADM	⊗	Unknown	541	🚩	Proposed
5804	SU01D : Unexpected next screen : SAP...	SU01D	BC-SEC-USR-ADM	⊗	Unknown	2	🚩	Proposed
5805	VA03 : No message received. Expected...	VA03	SD-SLS	🚩	Critical	1	🚩	Proposed
5806	BP : Unexpected message : S202(R1)	BP	AP-MD-BP-UI	⊗	Unknown	1	🚩	Proposed
5807	FB01 : Unexpected message : E205(F5)	FB01	FI	🚩	High	1	🚩	Proposed
5808	SE38 : Unexpected next screen : /BTI/...	SE38	BC-DWB-TOO-ABA	⊗	Unknown	1	🚩	Proposed

Displaying defects ordered by number of affected scripts

Sorting by the Scripts column allows you to prioritise your work so that you're focussing on the defects which have had the biggest impact on the execution of the playback.

Prioritisation by success rate

Within the **Result Overview** it is possible to see, on the Most Frequent Transactions tab, each object (dialog transaction, batch job, etc.) and the success rate of each one.

Most Frequent Transactions

Type	Object Type	Object name	Object Text	Sys	App Component	Priority	Script	Pass	Fail	Suppr. No r...	Unpr	Succ %
🚩	GUI dialog step	FB01	Post Document	HD4	Financial Accounting	High	1	0	1	0	0	0
🚩	RFC	BAPI_BUPA_RO...	SAP BP, BAPI: Determine All Roles (O...	HD4	SAP Business Partner	Unknown	541	0	541	0	0	0
🚩	RFC	BAPI_USER_GET...	Search for Users	HD4	User and Authorization ...	Unknown	541	0	541	0	0	0
🚩	GUI dialog step	VA03	Display Sales Order	HD4	Sales	Critical	2	1	1	0	0	50
🚩	GUI dialog step	SE38	ABAP Editor	HD4	ABAP Editor	Unknown	54	51	3	0	0	94
🚩	GUI dialog step	SU01D	User Display	HD4	User and Authorization ...	Unknown	68	66	2	0	0	97
🚩	GUI dialog step	BP	Maintain Business Partner	HD4	Dialog SAP-GUI	Unknown	53	52	1	0	0	98
🚩	GUI dialog step	VA05	List of Sales Orders	HD4	Basic Functions	Unknown	1	1	0	0	0	100
🚩	GUI dialog step	SU01	User Maintenance	HD4	User and Authorization ...	Unknown	1	1	0	0	0	100
🚩	Batch job	SAP_REORG_NO...		HD4		Unknown	1	1	0	0	0	100

Displaying transactions ordered by the least successful

Sorting by the "Succ %" column will allow you to focus your defect analysis on the least successful

transactions.

Prioritisation by object priority

Also within the Most Frequent Transactions display, you can see the priority of each transaction as determined by the Coverage Analysis. It is possible to filter by Priority so that you are working on the objects that have been deemed the most important to the business (starting, for example, by only looking at Critical or High priority transactions).

Most Frequent Transactions

Type	Object Type	Object name	Object Text	Sys	App Component	Priority	Script	Pass	Fail	Suppr. No r...	Unpr	Succ %
	GUI dialog step	FB01	Post Document	HD4	Financial Accounting	High	1	0	1	0	0	0
	GUI dialog step	VA03	Display Sales Order	HD4	Sales	Critical	2	1	1	0	0	50
	GUI dialog step	FBL5N	Customer Line Items	HD4	Information System	High	1	1	0	0	0	100

Showing only the most important objects

Prioritising by object type

You may also want to consider looking at the non-dialog transactions first, as it is often the case that defects with batch jobs or RFCs can be quickly dealt with. You can filter the defect overview screen so that you only see non-dialog objects.

Business Task Defect Management (2)

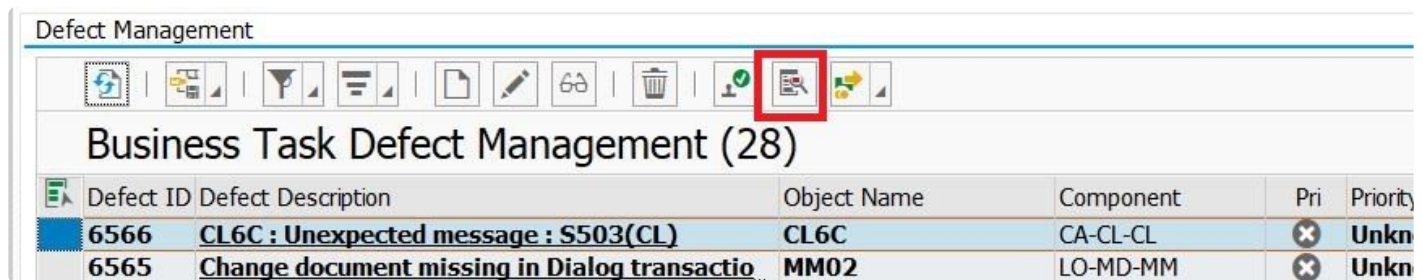
Defect ID	Defect Description	Object Name	Component	Pri	Priority	Scripts	Sts	Status
5802	Difference in output values in RFC BAPI_BUPA_ROLES_GET	BAPI_BUPA_ROLES_GET	AP-MD-BP		Unknown	541		Proposed
5803	Difference in output values in RFC BAPI_USER_GETLIST	BAPI_USER_GETLIST	BC-SEC-USR-ADM		Unknown	541		Proposed

Displaying only non-dialog objects

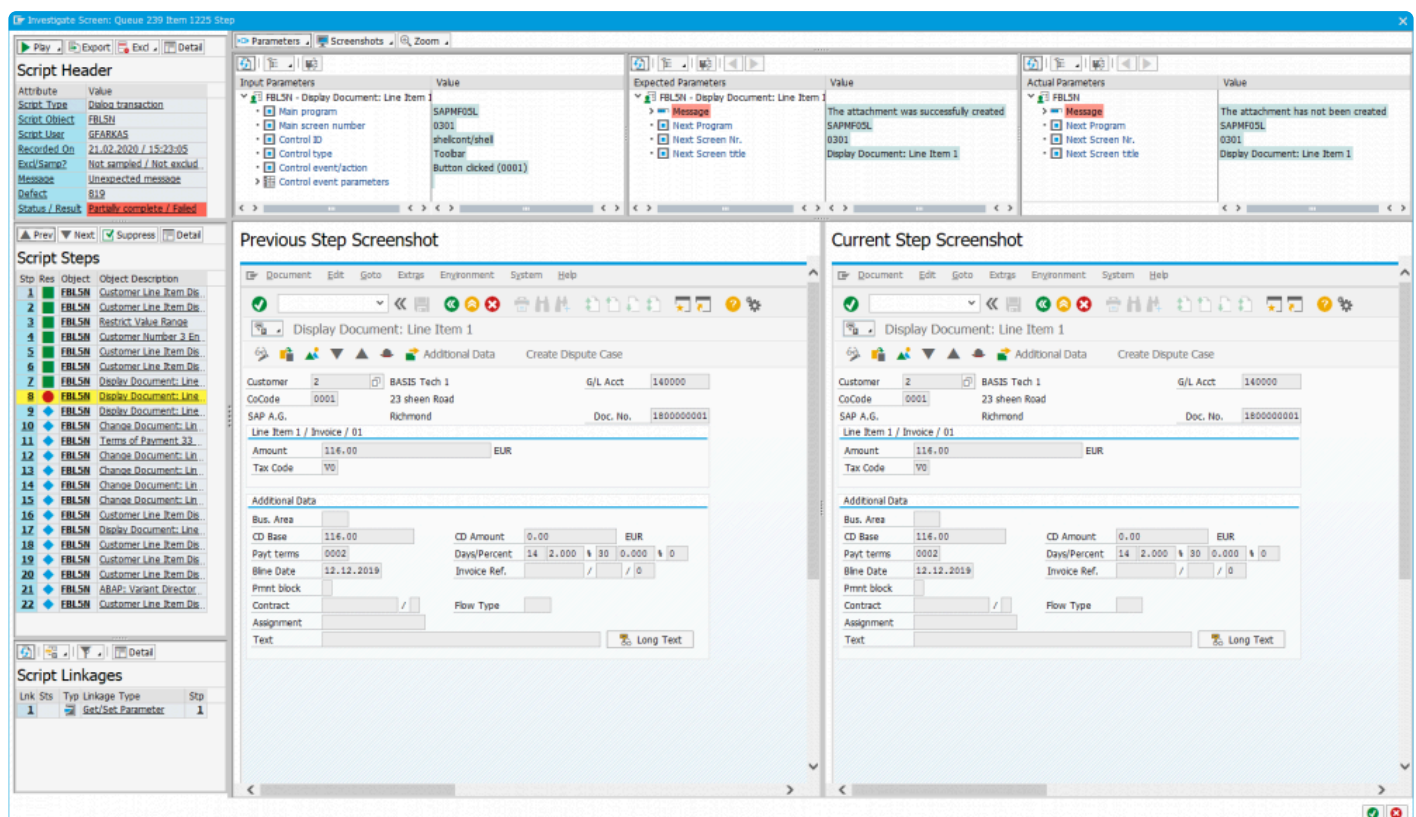
3.3. Investigating defects

The Investigate Screen is the most important tool for diagnosing the causes of failures during the playback, and has been significantly enhanced in Testimony v2.21.

From the Defect Management screen, highlight the Defect you want to investigate and click the “Investigate Script” button.



Here is an example of the Investigate Screen



Sample new investigate screen in v2.21

The Investigate Screen has different areas, some of which are common, and some of which change depending on the type of object (dialog transaction, batch job, etc.) that you are investigating. These are explained in detail in the following sections.

3.3.1. Common areas of the Investigate Screen

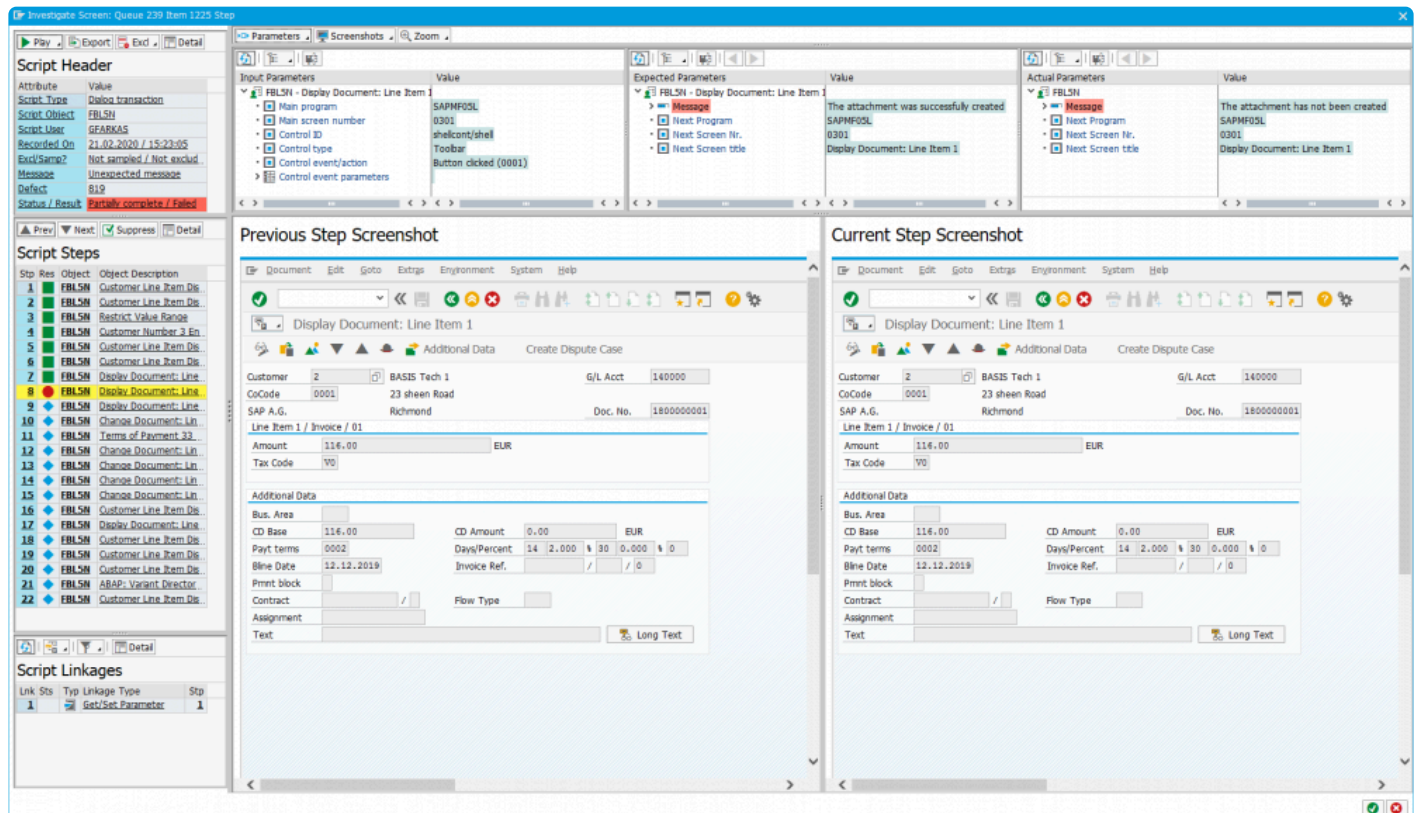
The following sections describe the areas of the Investigate Screen that are available for all object types.

- [Header Section](#)
- [Script Steps](#)
- [Script Linkages](#)
- [Inputs, expected outputs & actual outputs](#)

3.3.1.1. Header Section

The following changes have now been implemented in v2.21 of Testimony to show significantly more details to the user when investigating a script failure.

An example of the new investigate screen is shown here:



Sample new investigate screen in v2.21

Queue Information

At the top of the investigate screen you can now see the execution queue ID and the item ID within that queue (within the window title). The format of the title is “**Investigate Screen: Queue X Item Y**”.

Header Information

The grid in the top left representing the Script Header contains key information relating to the script that is being investigated. These include:

- **Script Type** – The type of script that is being investigated (e.g. Dialog transaction)

- **Script Object** – The name of the object representing the script (e.g. transaction VA01)
- **Script User** – The user who executed the script in the playback system
- **Recorded On** – The date and time that the script was recorded
- **Excl/Samp** – Information on whether this object is excluded or sampled in any way in the filter set configuration
- **Message** – If there is a message relating to the script (e.g. the failure reason) then this is shown here
- **Defect** – If a defect has been generated for this script then the defect number is displayed here
- **Status / Result** – Display the overall status of the script and whether it passed or failed

Additionally, you can perform the following:

- **Display the related defect** – If a defect number is shown, by selecting the hot-spot the defect details will be displayed
- **Display the test script header** – By selecting any other hot-spot area in the grid, the test script header will be displayed

Toolbar Actions

Within the toolbar of the header, you can perform the following actions:

- **Export the data** – This requires you switching to the old investigate screen and then exporting the script data from there
- **Setup Exclusions** – If this object should be excluded or sampled going forward, you can add it directly from here
- **Details of test script** – In the same way that you can click on various hot-spots in the grid, you can also click here to display the test script header
- **Trigger playback remote control** – This lets you re-run the script, step by step, with a bot that needs to be started.



Don't forget you need to activate the playback enhancements before using the playback remote control!

3.3.1.2. Script Steps

The script steps section of the new investigate screen is similar to the previous version. There are some noticeable differences which are noted below:

Grid display

The icon for displaying whether a script step passed or failed is now the standard “LED” icon.

- **Green** – means that the step executed and passed
- **Red** – means that the step has failed (or was a technical error)
- **Yellow** – means that the step was cancelled
- **Inactive** – means that the step did not run





✱ A new key piece of functionality is that the currently selected step is now highlighted in yellow. The user is therefore given immediate context on which step is currently selected during their investigation.

Action toolbar










- **Prev/Next** – Navigates to the previous or next step from the currently selected one
- **Suppress** – Immediately allows you to configure suppressions for the currently selected step
- **Detail** – Displays the test script step popup window

3.3.1.3. Related Linkages

The previous version of the investigate screen did not provide visibility of the linkages for the script/steps that were being investigated by the user. This is now resolved in Testimony v2.21 and linkages are now shown in the lower left part of the screen, as per the screen-shot below:

Detail

Script Linkages

Lnk	Sts	Typ	Linkage Type	Stp
<u>1</u>			<u>Get/Set Parameter</u>	<u>1</u>
<u>2</u>			<u>Number Range</u>	<u>4</u>
<u>3</u>			<u>Outbound IDoc - FLC..</u>	<u>4</u>
<u>4</u>			<u>Number Range</u>	<u>9</u>
<u>5</u>			<u>Number Range</u>	<u>9</u>
<u>6</u>			<u>Number Range</u>	<u>9</u>
<u>7</u>			<u>Inbound IDoc - FLCUS..</u>	<u>9</u>

List of linkges relating to the script

Linkage list

The following fields are shown in the grid:

- **Linkage Number** – A sequential number starting from 1 for each linkage in the script
- **Linkage Status** – If the linkage has been validated in any way (only relevant to change documents, IDoc's and SAP Script Forms), then this will display the status of that linkage. If the linkage has passed, then this icon will be green. If it failed, it will show as red. If no validation has happened then this will be empty.
- **Linkage Type** – Shows the type of linkage that it is
- **Related Step** – Linkages are stored at the step level. When using the investigate screen, this is

shown at the script level. All linkages relating to all steps of the script will be shown and this field lets the user know which step the linkage specifically relates to.

Toolbar actions

- **Filtering** – The filter options lets the user restrict the list of linkages shown by type or sub-type
- **Detail** – Internal details about the linkage can be shown by selecting the linkage first and the clicking “detail”
- **Linkage Selection** – If you select the hot-spot for a particular linkage, the linkage will be shown on the right hand side of the screen

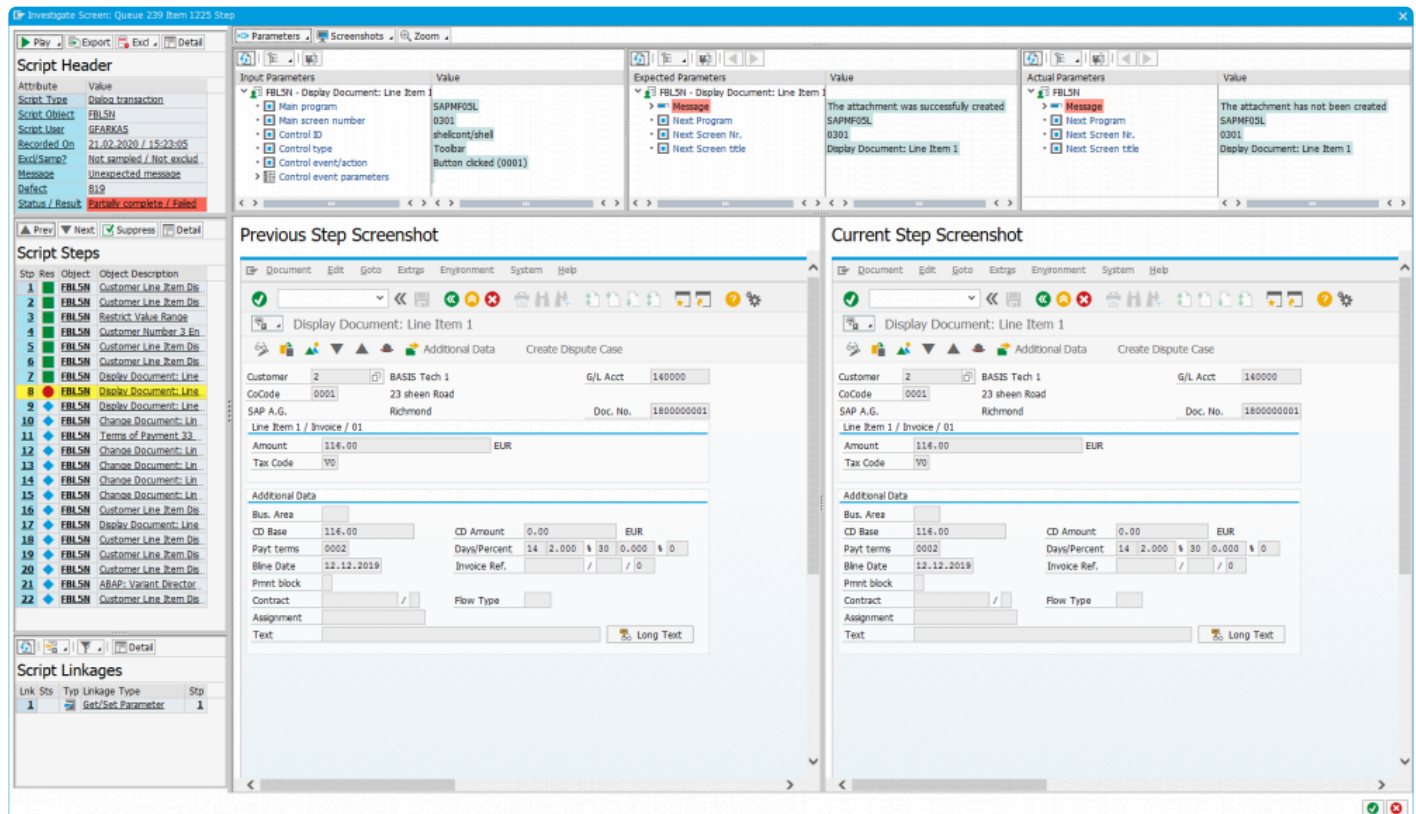
p(banner tip).



Like script step selection, if you select a linkage the row in the list will be high-lighted in yellow. This let's you immediately know which linkage you are currently investigating on the right hand side of the screen.

3.3.1.4. Expected vs Actual

The display of the inputs, expected outputs and the actual outputs was always available in the previous investigate screen, additional options are now available for your preference on how you would like these displayed. As per the previous version, the inputs to the step, expected outputs and actual outputs are displayed at the top of the screen, as seen in the screen-shot below.



Sample new investigate screen in v2.21

You are now able to change which of the 3 options are displayed and in what combination, as per the options listed below:

- **None** – No inputs/outputs are displayed at all
- **Inputs / Expected** – Only the inputs and expected outputs are displayed
- **Inputs / Actual** – Only the inputs and the actual outputs are displayed
- **Expected / Actual** – No inputs are displayed and only the expected and actual outputs are shown
- **Inputs / Expected / Actual** – This is the default setting – show all 3 inputs/outputs



If you change these windows, your preference is immediately saved against your user and the object type you are currently displaying. This means you can have different layouts for the different object types (e.g. one layout for dialog transactions and another for batch jobs).

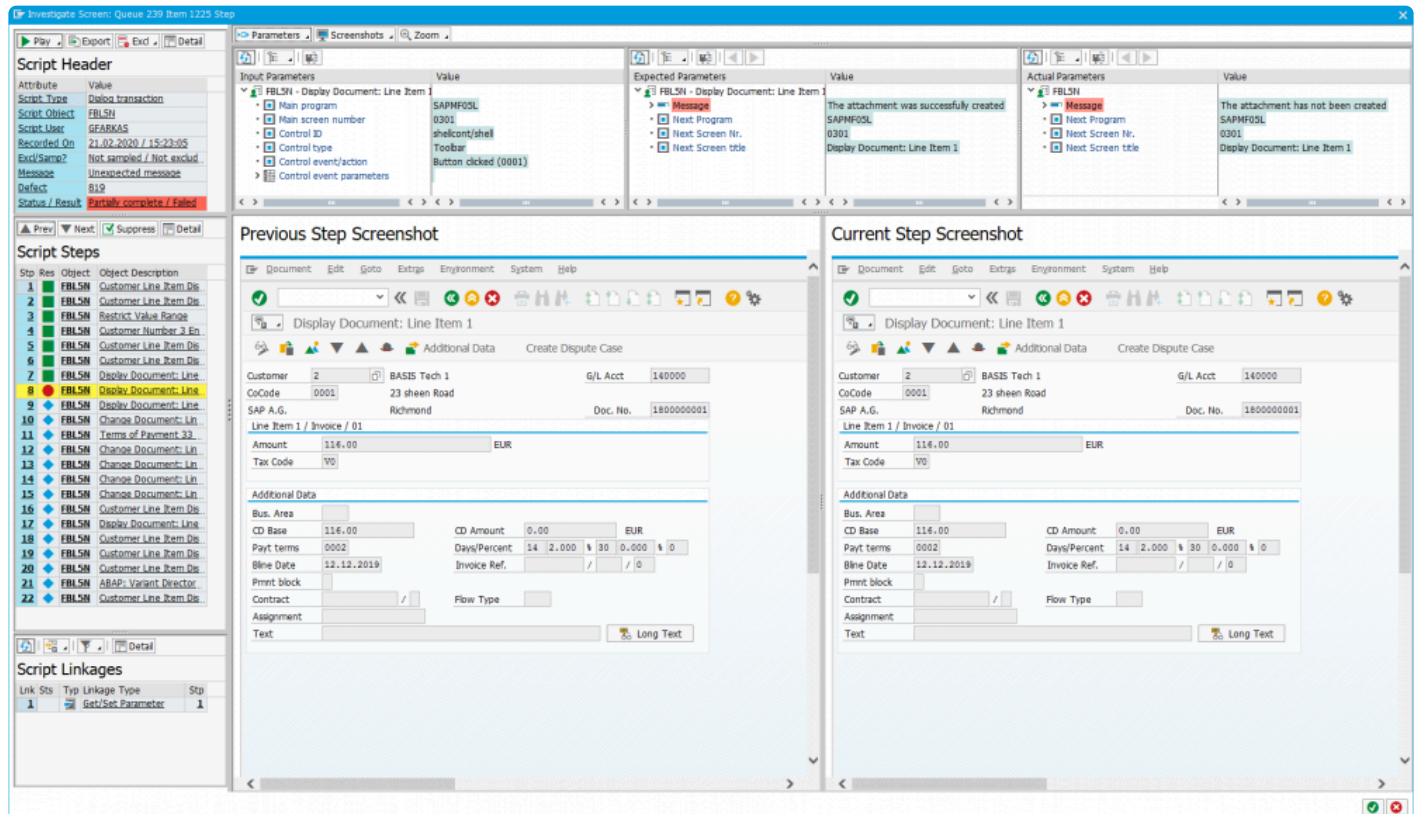
3.3.2. Object type screens

Some areas of the Investigate Screen – especially the large area reserved for screen shots in dialog transactions – changes depending on the type of script you are investigating. The following sections cover these areas for different script types.

- [Dialog Transactions](#)
- [Batch Jobs](#)
- [Inbound RFC](#)
- [Inbound Web-Services](#)

3.3.2.1. Dialog Transactions

An example of the dialog transaction investigate screen is shown below:



Sample new investigate screen in v2.21

Screen-shots

Testimony takes screen shots for every dialog step that is executed in the playback. The investigate screen shows the screen shots for the previous step and the step you are investigating, allowing you to see exactly what happened during the playback. It is possible to change your preferences for this screen so that only one screen shot is shown. This can be either the current step, or the previous step.

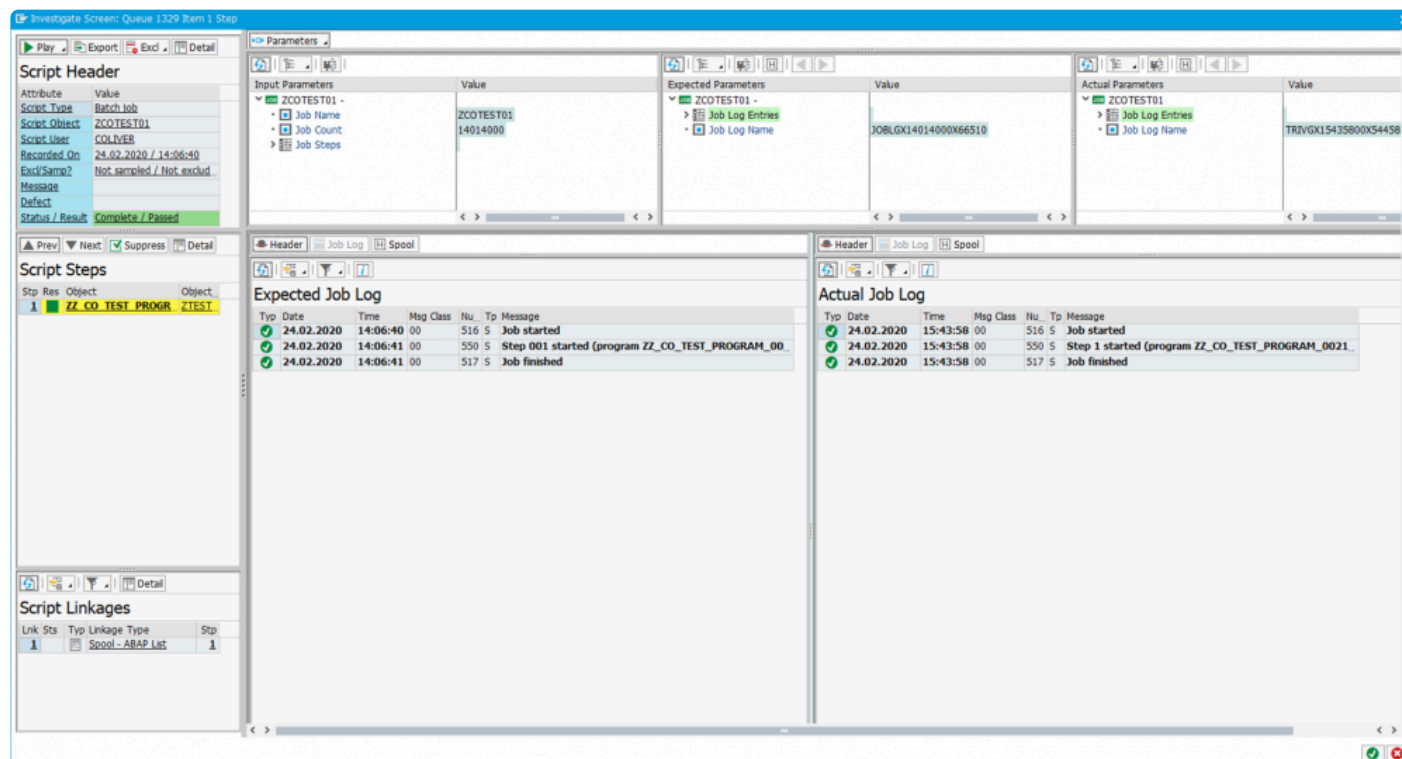
It is also possible to zoom in or zoom out dynamically for the screen-shot (or screen-shots) by a % factor. The options are 25%, 50%, 75%, 100% or 200%.



Zooming requires your central system to be on a certain kernel / support pack level. It also requires the RFC destination **IGS_RFC_DEST** to be configured correctly. Ensure the registered server program ID is set correctly – “IGS.SID”.

3.3.2.2. Batch Jobs

An example of the batch job investigate screen is shown below:



Batch job investigate screen

Key areas of the screen

The key areas of the investigate screen for batch jobs are:

Multi-step batch jobs

If a batch job has multiple steps defined, then each step will be shown in the “Script Steps” grid and each step can be selected.

Batch job log

The logs of the batch job are now displayed in a grid which can be filtered by message class and message type. Note that batch job logs are common for all steps if the batch job has more than one step in its definition.

Batch job header

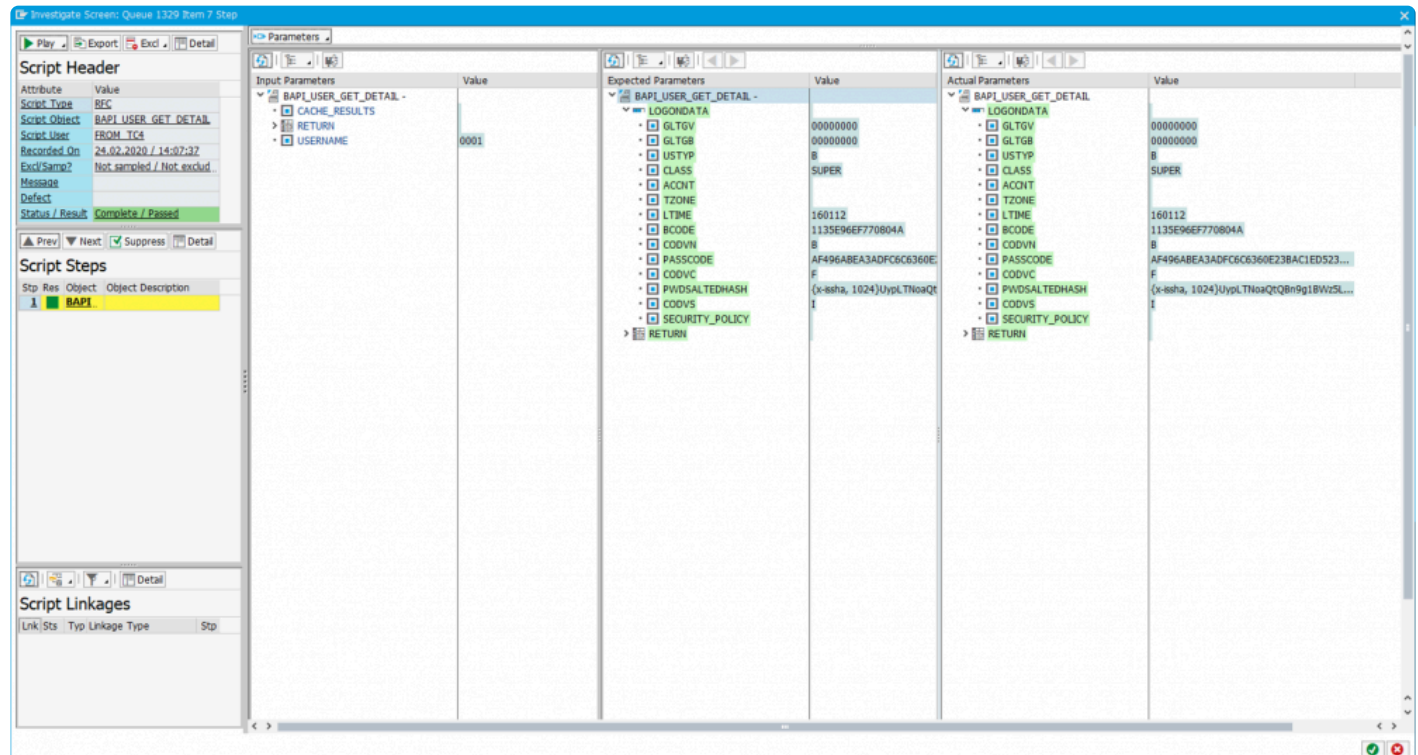
The header definition of the batch job step is available in the “Header”. This contains information such as when the batch job was scheduled to run in the recording and when it ran in the playback. For multi-step batch jobs, you can select each step and the header is specific to that step.

Batch job spool

Version 2.21 now captures the spool during the recording and the playback. You will not be able to see the spool output until the playback is finished completely as the spool contents are retrieved during a post-processing step. Similarly, the batch job spools are not retrieved until the recording is switched off. Spool content is currently not compared between the recording and playback. They are there to support the user in their investigation of batch job failures.

3.3.2.3. Inbound RFC

An example of the inbound RFC investigate screen is shown below:

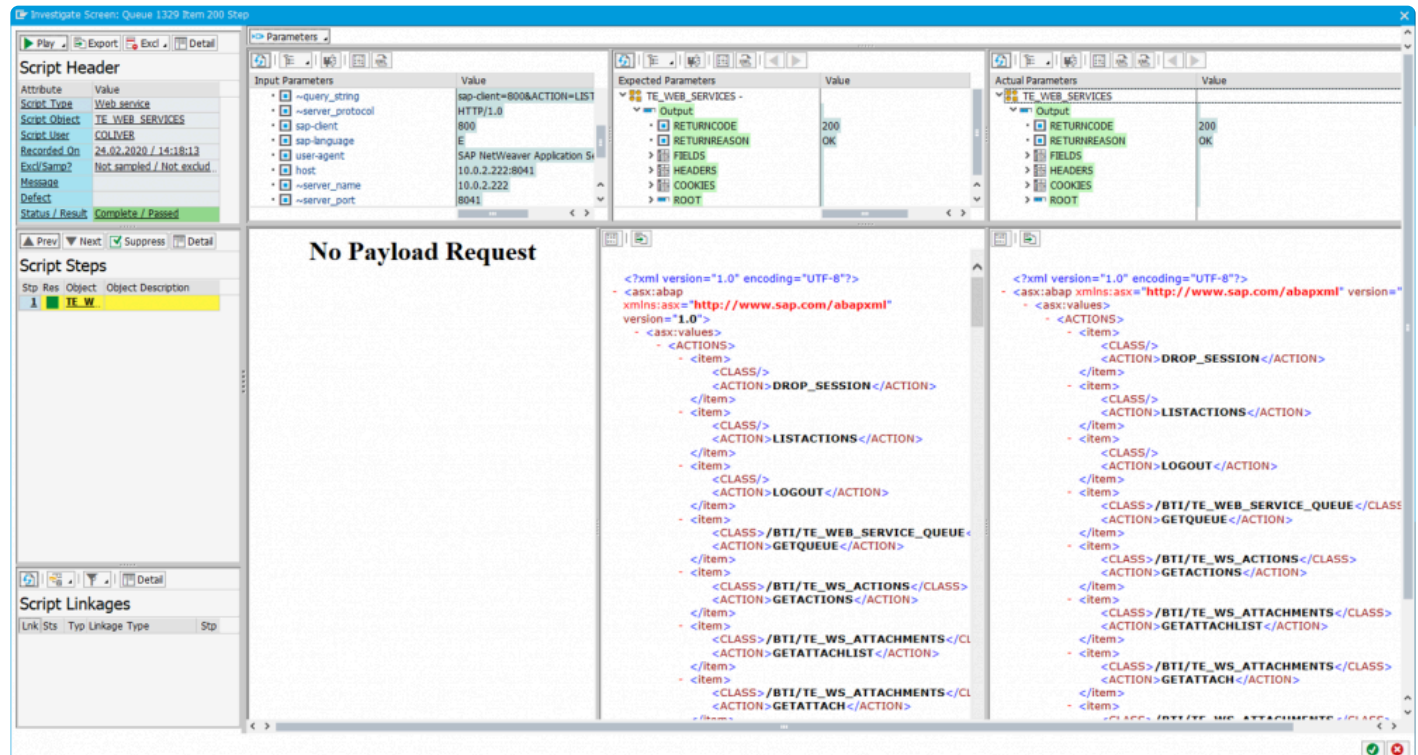


Inbound RFC investigate screen

Note that the area of the screen used for dialog step screen shots or batch job logs is not used in this screen.

3.3.2.4. Inbound Web-Services

An example of the inbound web-service investigate screen is shown below:



Investigate screen for inbound web-services

The expected and actual output from the web service call are displayed as XML documents in the lower section of this screen.

3.3.3. Linkage Type Screens

There are 13 linkage types which all now have their own specific investigation screen.

There are presently 3 types that are compared between the recording and the playback. This means that if the playback is different to the recording (unless suppressed) they will fail the script during playback:

- **Change Documents**
- **SAP Script Forms**
- **Inbound / Outbound IDoc's**

The next 6 types are used for service virtualization during the playback in order to mimic what the system does or what the user does by the bot:

- **Front-end Actions**
- **Local Files**
- **Application Server Files**
- **SET/GET Parameters**
- **User Preferences**
- **Clipboard Imports**

And finally, the following linkage types are used to obtain key information on business processes so that Testimony can understand links between scripts/steps or provide additional information to the user investigating the playback:

- **Number Ranges**
- **Dynamic ID**
- **Dynamic ID (User Prefs)**
- **Spool Requests**

Each linkage type can be visually seen in the investigate screen by selecting the linkage in the lower left side of the screen. Each of these are now described in the following sub-sections.

3.3.3.1. Change Documents



This linkage type is used for validating that the same change document (or change documents) were also generated in the playback system for the same script/step. There are two failure reasons (1) Change document missing or (2) Change document is different.

Linkage Type - Change Document

There are two failure reasons for change documents:

1. **Change Document Missing** – this failure reason means that when the expected change document is validated, a corresponding change document from the playback was not found.
2. **Change Document Different** – the corresponding change document was found, but when the header or line items were compared, they were found to be different



Each change document that is captured in the recording is compared with the change documents that were captured in the playback. The result of this comparison is shown in the lower left linkage list. A green indicator shows that the comparison was the same. If it is red it means the validation failed.

3.3.3.2. SAP Script Forms



This linkage type is used for validating that the same SAP Script Form were also generated in the playback system for the same script/step. There are two failure reasons (1) SAP Script Form missing or (2) SAP Script Form is different.

Script Header

Attribute	Value
Script Type	Sales transaction
Script Object	SEZI
Script User	SEZI
Recorded On	16.02.2020 / 17:39:11
End/Start?	Not started / Not ended
Message	SAP script is different
Default	
Status / Result	Partially complete / failed

Script Steps

Step	Res	Object	Object Description
1	SEZI	SAPScript Form Mani...	
2	SEZI	Form Painter: Request	
3	SEZI	Print	
4	SEZI	Print Preview of IP01	
5	SEZI	Form Painter: Request	

Script Linkages

Link	Sts	Typ	Linkage Type	Step
1			Set/Spec Parameter	1
2			SAP Script Form - ZC	2

Expected SAP Script Form

Fly & Smile
Zeppelinstr. 4
10101 Fliegen
XX.XX.XXXX

Fa. Turnaround
Mittlerer Ring 145
75001 Neustadt

/E CLOSING_REMARK
Yours faithfully,
Your Fly & Smile Team

/E INTRODUCTION
Dear Sir or Madam,

We would like to thank you for your order and confirm the following flight bookings. Please note that a discount of 12 percent has already been included for flight AZ 0792.

Flight	Date	Departure	Price
AA 0017	20.07.1998	13:15	799,00 USD
AA 2019	23.07.1998	21:55	1.799,00 USD
AZ 0790	11.11.1998	6:55	1.644,00 USD
LN 0454	22.08.1998	10:10	1.151,40 USD
LN 1999	31.12.1998	23:59	666,66 USD

Page 1

Actual SAP Script Form

Fly & Smile
Zeppelinstr. 4
10101 Fliegen
XX.XX.XXXX

Fa. Turnaround
Mittlerer Ring 145
75001 Neustadt

/E CLOSING_REMARK
Yours faithfully,
Your Fly & Smile Team

/E INTRODUCTION
Dear Sir or Madam,

We would like to thank you for your order and confirm the following flight bookings. Please note that a discount of 12 percent has already been included for flight AZ 0793.

Flight	Date	Departure	Price
AA 0017	20.07.1998	13:15	799,00 USD
AA 2019	23.07.1998	21:55	1.799,00 USD
AZ 0790	11.11.1998	6:55	1.644,00 USD
LN 0454	22.08.1998	10:10	1.151,40 USD
LN 1999	31.12.1998	23:59	666,66 USD

Page 1

Linkage Type - SAP Script Form

There are two failure reasons for SAP Script Forms:

1. **SAP Script Form Missing** – this failure reason means that when the expected SAP Script Form is validated, a corresponding form from the playback was not found
2. **SAP Script Form Different** – the corresponding SAP Script Form from the playback was found, but when the header or OTF data was compared, they were found to be different



You can additionally view the header of the SAP Script Form or the raw OTF data of the form by selecting the appropriate toolbar option.

- ✿ Each SAP Script Form that is captured in the recording is compared with the forms that were captured in the playback. The result of this comparison is shown in the lower left linkage list. A green indicator shows that the comparison was the same. If it is red it means the validation failed.

Additional information is available in the Key Enhancements for Testimony v2.21 on [SAP Script Forms](#).

3.3.3.3. Inbound / Outbound IDocs



This linkage type is used for validating that the same IDoc was also generated in the playback system for the same script/step. There are two failure reasons (1) IDoc missing or (2) IDoc is different.

Script Header

Attribute	Value
Script Type	Release transaction
Script Object	WE19
Script User	GETEST
Recorded On	16.02.2020 / 17:38:41
End Same?	Not sampled / Not needed
Message	
Default	
Status / Result	Error / No result

Script Steps

Step	Res	Object	Object Description
1	WE19	Test Tool for IDoc Pin	
2	WE19	Test Tool for IDoc Pin	
3	WE19	Test Tool for IDoc Pin	
4	WE19	Outbound Processing	
5	WE19	Outbound Processing	
6	WE19	Test Tool for IDoc Pin	
7	WE19	Test Tool for IDoc Pin	
8	WE19	Test Tool for IDoc Pin	
9	WE19	Test Inbound IDoc Use	
10	WE19	Test Inbound IDoc Use	
11	WE19	Test Tool for IDoc Pin	
12	WE19	Test Tool for IDoc Pin	

Script Linkages

Link	Step	Linkage Type	Step
1	1	Set/Set Parameter	1
2	1	Number Range	4
3	1	Outbound IDoc - FLC	4
4	1	Number Range	9
5	1	Number Range	9
6	1	Number Range	9

Expected IDoc

Expected result	Value
Outbound IDoc	0000000000000077
Control Record	FLCUSTOMER_CREATEFROMDATA
Data Segments	3 segments
E1SCU_CRE	1 field (2 sub-segments)
Switch to Simulation Mode	
E1BPSUNNEW	15 fields (No sub-segments)
Customer name	FOO
Form of address	FRMA
Street	DIETHAR-HOPP-ALLEE 999
PO Box	
Postal Code	69190
City	WALLDORF
Country Indic.	DE
ISO country code	DE
Region	
Tel. no.	06227-34-0
E-Mail Address	INFO@BASISTECHNOLOGIES.COM
BIP customer	9
Discount (%)	10
Language Key	D
Lang. (ISO 639)	DE
E1BPPAREX	5 fields (No sub-segments)
BAPI table extension struc	
Data	
Data	
Data	
Data	

Actual IDoc

Actual result	Value
Outbound IDoc	0000000000000089
Control Record	FLCUSTOMER_CREATEFROMDATA
Data Segments	3 segments
E1SCU_CRE	1 field (2 sub-segments)
Switch to Simulation Mode	
E1BPSUNNEW	15 fields (No sub-segments)
Customer name	FOO
Form of address	FRMA
Street	DIETHAR-HOPP-ALLEE 999
PO Box	
Postal Code	69190
City	WALLDORF
Country Indic.	DE
ISO country code	DE
Region	
Tel. no.	06227-34-0
E-Mail Address	INFO@BASISTECHNOLOGIES.COM
BIP customer	9
Discount (%)	10
Language Key	D
Lang. (ISO 639)	DE
E1BPPAREX	5 fields (No sub-segments)
BAPI table extension struc	
Data	
Data	
Data	
Data	

Linkage Type - Inbound / Outbound IDoc

There are two failure reasons for inbound and outbound IDocs:

- IDoc Missing** – this failure reason means that when the expected IDoc was validated, a corresponding IDoc from the playback was not found
- IDoc is Different** – the corresponding IDoc from the playback was found, but when the control record or the IDoc segment data was compared, they were found to be different



Each IDoc that is captured in the recording is compared with the IDoc's that were captured in the playback. The result of this comparison is shown in the lower left linkage list. A green indicator shows that the comparison was the same. If it is red it means the validation failed.

Additional information is available in the Key Enhancements for Testimony v2.21 on “[IDoc deep valuation](#)”:

3.3.3.4. Number Range

A number range retrieval is captured during the recording and used to find “Dynamic ID’s” between scripts / steps. These create dependencies so that earlier scripts can create data which can then be used subsequently in later scripts.

An example of a number range linkage displayed in the new investigate screen is shown below:

Investigate Screen: Queue 1329 Item 26 Step

Play

Export

Exc

Detail

Script Header

Attribute	Value
Script Type	Dialog transaction
Script Object	VA01
Script User	SCRIP.T.02
Recorded On	24.02.2020 / 14:08:32
Excl/Samp?	Not samp'd / Not excl'd
Message	
Defect	
Status / Result	Complete / Passed

Prev

Next

Suppress

Detail

Script Steps

Stp	Res	Object	Object Description
1	VA01		Create Sales Order; Int
2	VA01		Create Sales Order; Int
3	VA01		Create Sales Order; Int
4	VA01		Create Standard Order;
5	VA01		Create Standard Order;
6	VA01		Create Standard Order;
7	VA01		Create Standard Order;

Script Linkages

Link	Sts	Typ	Linkage Type	Stp
1		Get/Set	Parameter	1
2		Number Range		6
3		Number Range		6
4		Number Range		6
5		Number Range		6
6		Number Range		6

Number Range

Attribute	Value
Test Script ID	0005286235
Sequence Number	000001
Object	COPA_OB1
Sub Object	IDEA
Short Text	Object numbers
Long Text	Number range for object numbers (CO-PA)
Interval	01
Year	0000
From Number	000000000000000000000001
To Number	000000000000000000000001
Data Element	RKEOBJNR
External ID	
Generated Number	0000385127

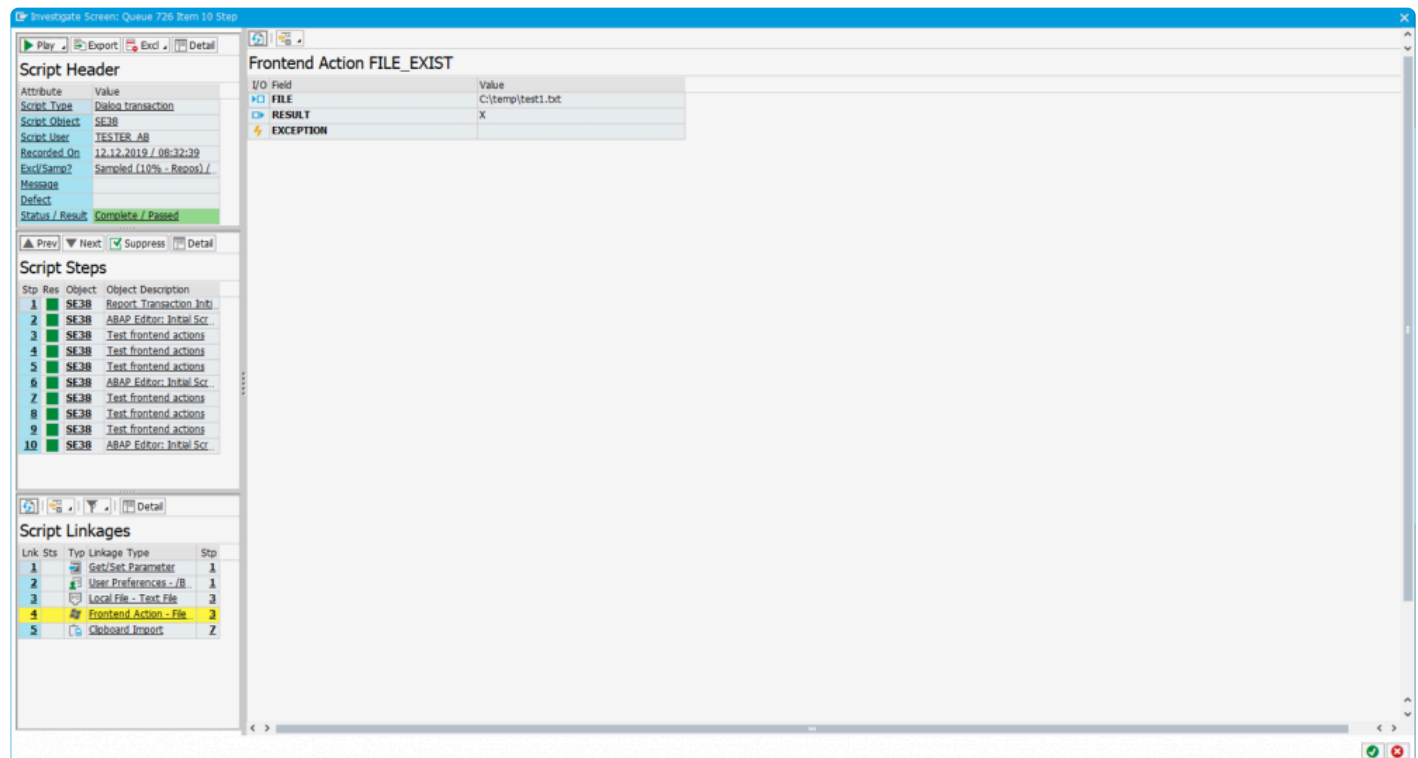
Linkage Type - Number Range

Key information on the number range retrieval include the number range object and the number itself that was retrieved in the recording.

3.3.3.5. Front-end Actions

Front-end actions are used to do service virtualization of the bots during the playback.

An example of a front-end action linkage displayed in the new investigate screen is shown below:



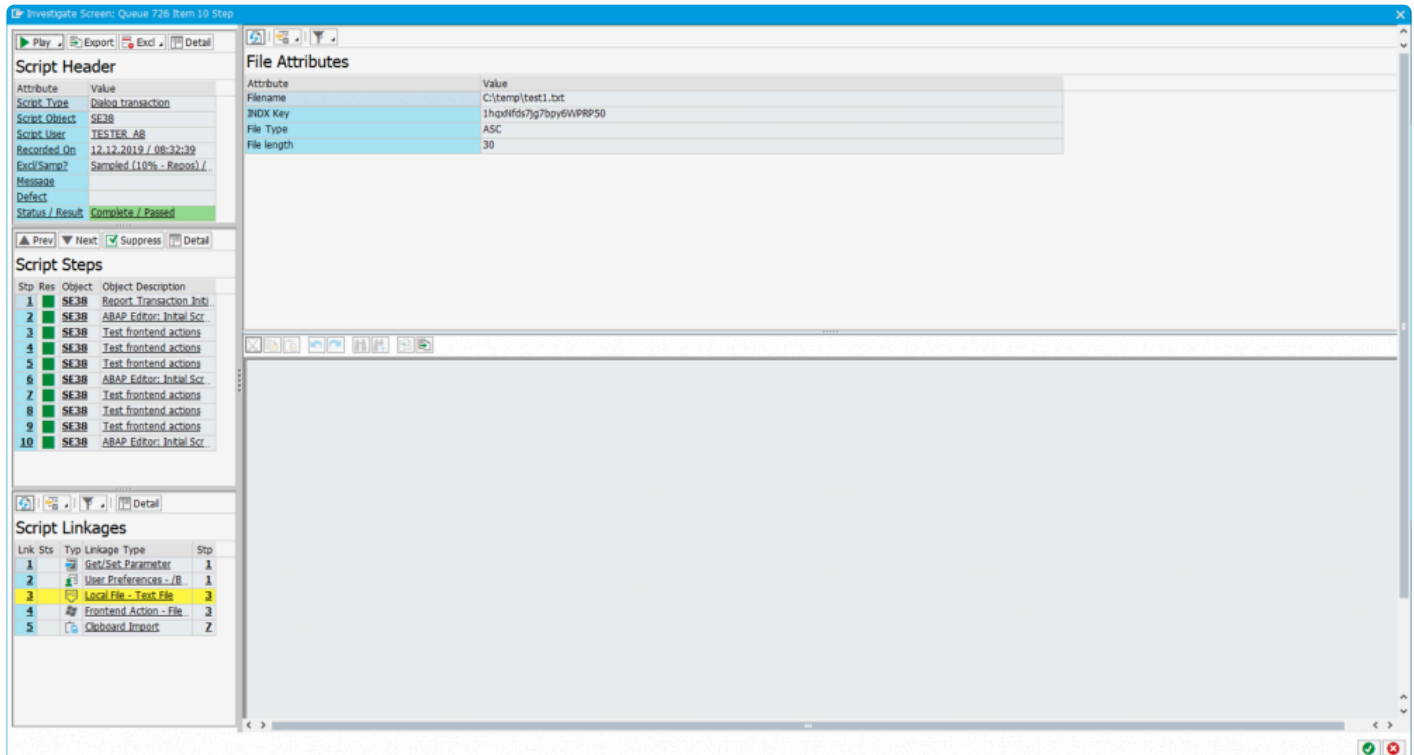
Linkage Type - Front-end Action

Key information on the front-end action includes the type of front-end action (e.g. checking if a file exists, a directory exists, creation of a local file etc) and the result of the front-end action during the recording.

3.3.3.6. Local Files

Local files are used for service virtualization of the bots during the playback.

An example of a local file that is captured during the recording and used for service virtualization in the playback is shown below.

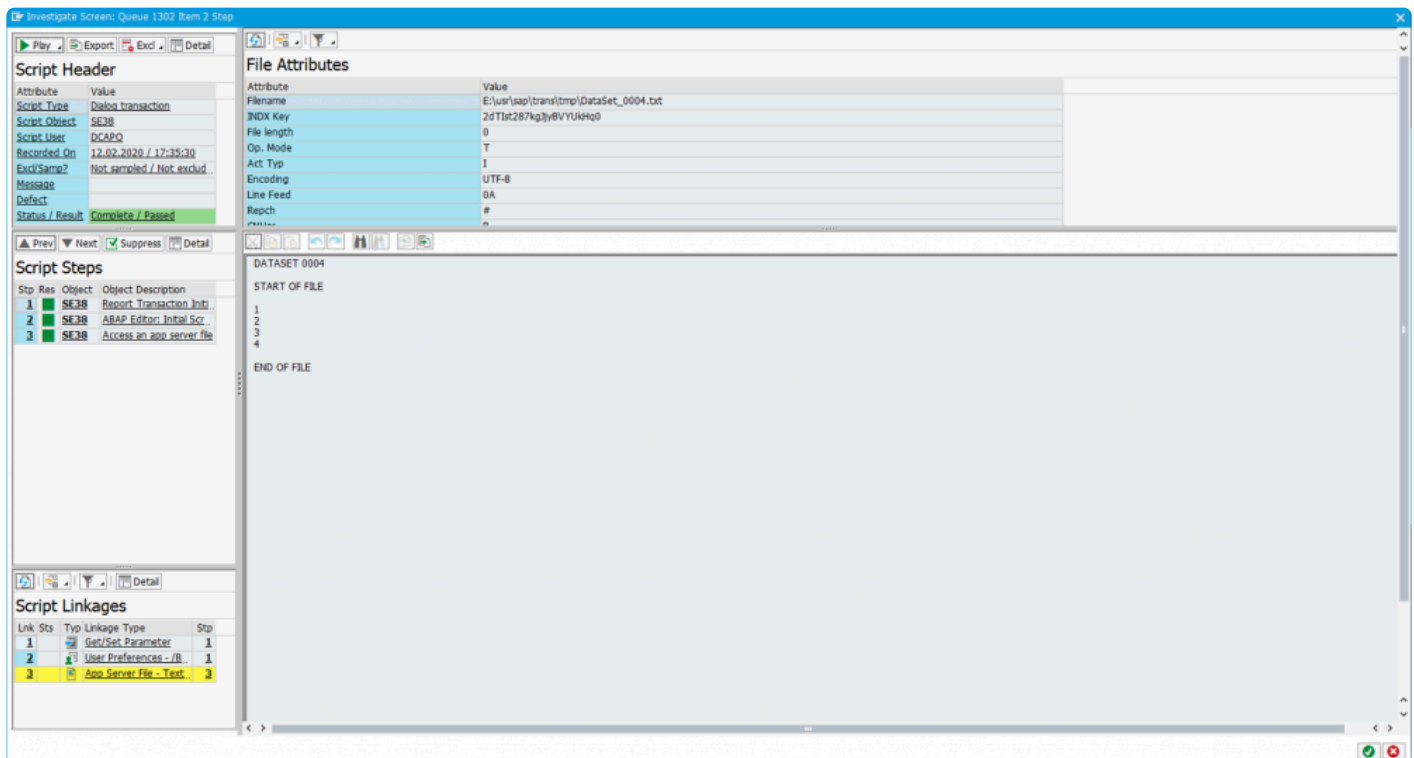


Linkage Type - Local file

Key information on the local file includes the filename and directory of the local file. The file contents can also be seen.

3.3.3.7. Application Server Files

Application server files are used to ensure that (typically) batch jobs process the same files during the playback as they did in the recording. The files are provided to the application server just prior to running the step that contains the linkage.



Linkage Type - Application Server File

When you click on the linkage, you can see the header properties of the file and the file contents itself.

3.3.3.8. SET/GET parameters

SET/GET parameters are used to ensure that dialog transactions run in the playback are run in the same manner as they were captured in the recording. The users SET/GET parameters are captured at the start of the transaction only (not at each step) and restored during the playback just before triggering the transaction for the user. Thus this linkage type acts as a service virtualization technique.

An example of the new investigate screen showing these SET/GET parameters is shown below.

Parameter ID	Description	Parameter Value
BAP		S
PAR		AG
POK	PO: Views (Key, Short Text, Validity, et	X
SCL	Upper and lower case in source code: 'X'	X
SPI	Spool ID	0000000000
SPR	Language	E
US2	Save time of last login	202002131656202020022414083520200213000
VTV		0

Linkage Type - SET/GET Parameters

When the user clicks on the linkage, each of the SET/GET parameters that will be set during the playback are shown along with their associated values.

3.3.3.9. User Preferences

User preferences are different to SET/GET parameters in that specific transactions contain settings for a particular user stored within one or more tables. These settings are configured in configuration tables within the central system in Testimony and delivered out of the box with Testimony. However, you can configure your own entries for specific transactions that you are aware of requiring these user settings to be captured during the recording.

When users run transactions during the recording, their user preferences are captured on the transaction being started. During the playback, these user preferences are restored prior to the transaction being started by the worker jobs and bots.

An example of the new investigate screen showing these user preferences is shown below.

The screenshot shows the 'Investigate Screen: Queue 1329 Item 27 Step' window. The main table is titled 'User Preferences for table ESDUS'. It contains the following data:

HANDT	USERNAME	ACTION	ELEMENT	Po_Process	ACTIVE
800	BGUYMAN	MEPO	Application	4500018456AX	
800	BGUYMAN	MEPO	Document	HELPWINDOW	X
800	BGUYMAN	MEPO	Document	SEARCHHELP	X
800	BGUYMAN	MEPO	Document	SHOWKEY	
800	BGUYMAN	MEPO	Environment	ExternAmodal	
800	BGUYMAN	MEPO	MessageHandler	Event10	
800	BGUYMAN	MEPO	MessageHandler	Event11	
800	BGUYMAN	MEPO	MessageHandler	Event12	
800	BGUYMAN	MEPO	MessageHandler	Event13	
800	BGUYMAN	MEPO	MessageHandler	EventsforDialog	10
800	BGUYMAN	MEPO	OwnDocsCust	Timeframe	-7
800	BGUYMAN	MEPO	Query Controller	Autoload	X
800	BGUYMAN	MEPO	Query Controller	Current Blade	0
800	BGUYMAN	MEPO	ROOT	version	1.1
800	BGUYMAN	MEPO	SAPLMEGJ/0000	Tree-On	
800	BGUYMAN	MEPO	SAPLMEGJ/0000	Tree-Width	132
800	BGUYMAN	MEPO	Tree-Control	Tree-On	
800	BGUYMAN	MEPO	Tree-Control	Tree-Width	132
800	BGUYMAN	PurchaseOrder	DYN_4000-BUTTON	TOGGLE_STATE	2
800	BGUYMAN	PurchaseOrder	DYN_4001-BUTTON	TOGGLE_STATE	2
800	BGUYMAN	PurchaseOrder	DYN_4002-BUTTON	TOGGLE_STATE	2
800	BGUYMAN	PurchaseOrder	HEADER-TABSTRIP	SUBVIEW_INDEX	9
800	BGUYMAN	PurchaseOrder	ItemTable	Online_Dispo_Dialog	X
800	BGUYMAN	PurchaseOrder	ItemTable	Online_Dispo_Prop	
800	BGUYMAN	PurchaseOrder	OrgData	View1222Listbox	
800	BGUYMAN	PurchaseOrder	POHeaderProposer	BSART	NB
800	BGUYMAN	PurchaseOrder	POHeaderProposer	BUKRS	
800	BGUYMAN	PurchaseOrder	POHeaderProposer	EKGRP	
800	BGUYMAN	PurchaseOrder	POHeaderProposer	EKORG	
800	BGUYMAN	PurchaseOrder	POItemProposer	AFNAM	
800	BGUYMAN	PurchaseOrder	POItemProposer	AKTHR	
800	BGUYMAN	PurchaseOrder	POItemProposer	BEDNR	
800	BGUYMAN	PurchaseOrder	POItemProposer	EEIND	
800	BGUYMAN	PurchaseOrder	POItemProposer	ELPEI	
800	BGUYMAN	PurchaseOrder	POItemProposer	KNHTP	
800	BGUYMAN	PurchaseOrder	POItemProposer	KZABS	
800	BGUYMAN	PurchaseOrder	POItemProposer	LGORT	
800	BGUYMAN	PurchaseOrder	POItemProposer	HATKL	
800	BGUYMAN	PurchaseOrder	POItemProposer	PSYTP	0
800	BGUYMAN	PurchaseOrder	POItemProposer	WERKS	

The 'Script Linkages' section on the left shows the following linkages:

Link	Sta	Typ	Linkage Type	Step
1			Get/Set Parameter	1
2			User Preferences - ESD	1
3			Dynamic ID - User Pr	1
8			Change Document - P	8

Linkage Type - User Preferences

When the user clicks on the linkage, each of the values that were captured in the user preference configuration tables is shown with their associated values. It is these entries that will be restored during the playback making sure that the transaction functions in precisely the same manner as was captured in the recording.

3.3.3.10. Dynamic ID

Dynamic ID's are used to capture relationships between scripts so that data dynamically generated in one script can be used in a later script.

Simple Example

A user create a business partner in the recording. The business partner number created is number 123. The same user (or another) at a later point in time creates a new sales order for the customer 123. During the playback (due to issues with sequencing and number range buffers), the same business partner 123 may not be created by the same script. Instead, in the playback, the business partner 124 is created. Dynamic ID's are leveraged to ensure that the second script to create a sales order is done for customer 124 rather than 123.

Dynamic ID Types

A dynamic ID can be of two types – (1) A producer or (2) a consumer. A **producer** is the step that generates the unique number / identifier. A **consumer** is a step that make use of the value from a previous producer. Various scenarios exist such as:

- A consumer cannot exist without a producer
- There can be many consumers of a producer
- A producer script (or step) can also be a consumer (but these will be separate linkages)
- Consumers can exist within the same script as a producer

An example of the new investigate screen showing these dynamic ID's is shown below:

The screenshot displays the 'Investigate Screen' for Queue 1329, Item 26. The interface is divided into several sections:

- Script Header:** Shows attributes like Script Type (Dialog transaction), Script Object (VA01), Script User (SCRIPT_02), Recorded On (24.02.2020 / 14:08:37), and Status / Result (Complete / Passed).
- Producer step - Step/Script details:** A table showing step details for item 6, which is a 'GUI dialog st. VA01' in a 'Complete' status.
- Consumer Steps/Scripts:** A table listing consumer steps for items 34, 52, and 75, all of which are 'Complete'.
- Script Steps:** A list of steps (1-7) for the script, all with a status of 'Complete'.
- Script Linkages:** A table showing linkages between script steps and objects, including 'Get/Set Parameter' and 'Number Range'.

Linkage Type - User Preferences

When investigating a dynamic ID linkage, keep in mind whether you are looking at a producer or a consumer (it is made clear in the linkage list). Additional grids in the right hand side of the screen show you the consumer(s) (if you are looking at a producer). It also shows you other producer or consumers in the same scripts or later ones. You are able to see the status of the producer scripts (did they complete successfully or not).



Critically, you are able to see the values that were generated in the recording (for the producer) and the values that were generated in the playback. Thus you can investigate your script failure knowing the values that were used.

3.3.3.11. Dynamic ID (User Preferences)

Sometimes, user preferences contain identifiers that are dynamically generated during the playback (by Dynamic ID producers). Hence, during the playback, these dynamic variables need to be substituted prior to running common transactions so that they operate in the same during the playback as they did during the recording.

These are captured in a linkage type referred to as **Dynamic ID User Preferences**.

An example screen-shot of the investigate screen for one of these types is shown below:

The screenshot displays the 'Investigate Screen' with a queue of 1309 items at step 12. The interface is divided into several panels:

- Script Header:** Shows attributes like Script Type (Dialog transaction), Script Object (ME21N), Script User (BOLYMAN), Recorded On (13.02.2020 / 14:13:37), and Status / Result (Complete / Passed).
- Producer step - Step/Script details:** A table showing step 5, GUI dialog st, ME21N, Complete status, Passed result, and a recorded value of 4500018446.
- Consumer Steps/Scripts:** A table showing item 101, GUI dialog st, ME21N, Complete status, Passed result, and a playback value of 4500018446, linked to the user preference table ESDUS.
- Script Steps:** A list of 6 steps, all of type ME21N, including 'Start MEPO' and multiple 'Create Purchase Order' and 'Save Document' actions.
- Script Linkages:** A table showing 11 linkages. Linkage 11 is highlighted in yellow and is of type 'Dynamic ID - User Pr', linking step 5 to step 5.

Linkage Type - Dynamic ID User Preferences

The key information in the view is what the dynamic ID value is from the recording (versus that generated in the playback) and which user settings table this dynamic identifier will be substituted into during the playback.

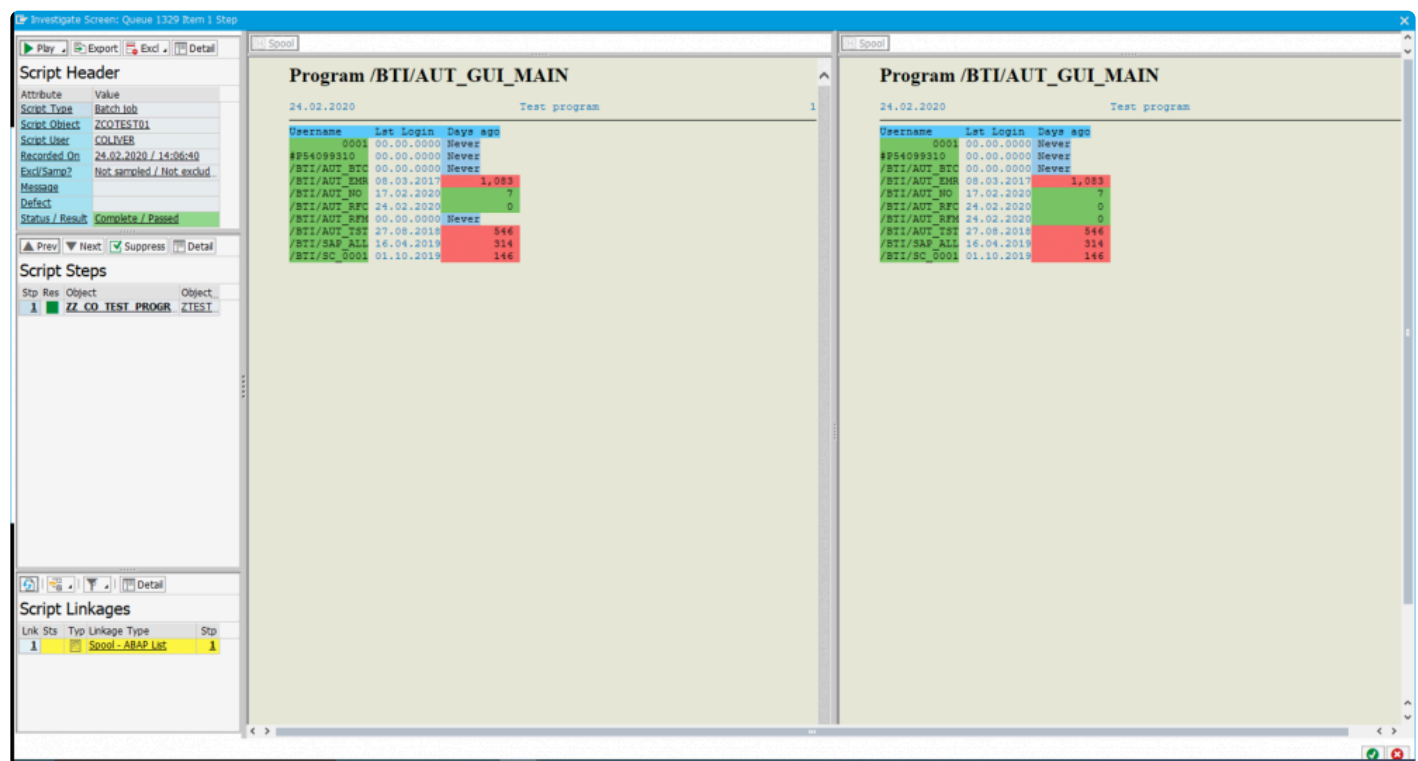
3.3.3.12. Spool Requests

Batch job steps typically generate spool requests when they run. This is not for all batch jobs but for most. The output in these spool requests often provide information as to how the program functioned during the recording and also during the playback.

Spool requests generated by batch jobs captured in the recording are now created as new linkage types at the end of the recording process (during the data transfer back to the central system). For the playback, these same spool requests generated by the batch jobs run during the playback are retrieved once the playback is completed (in a post-processing step).

While the spool requests from the recording and the playback are not currently compared in any way, the spool requests provide critical information to the user when investigating failures during the playback.

An example of the new investigate screen for the linkage type spool request is shown below.



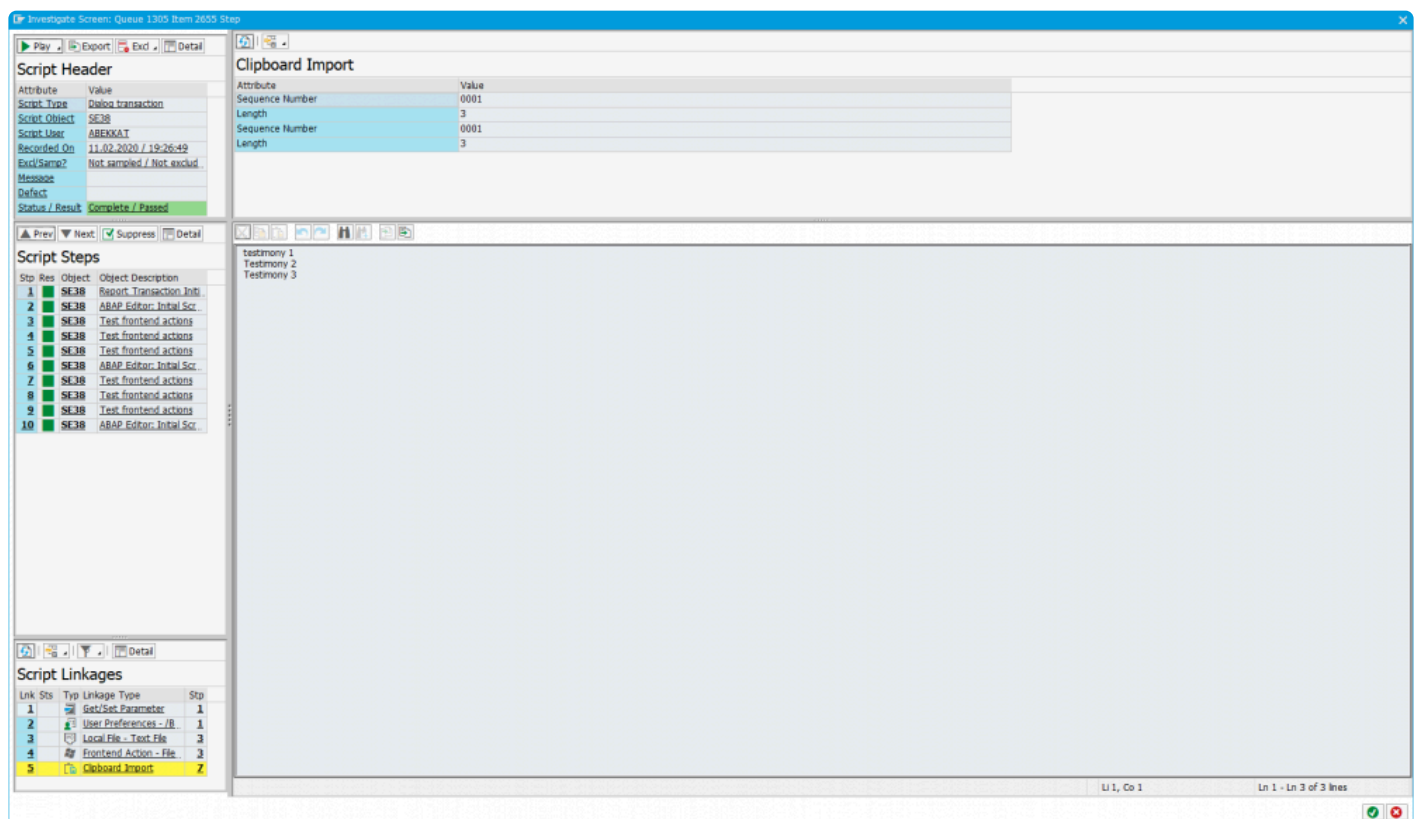
Linkage Type - Spool Request

As you can see, both the spool request generated by the batch job in the recording is shown on the left and the spool request generated by the same batch job in the playback is shown on the right. You can visually compare the two spools to determine if they functioned in a similar manner.

3.3.3.13. Clipboard Imports

During the recording, users operating SAP transactions are able to paste data from their local machine into the SAP application from their clipboard. Since the bot does not have the same content in its clipboard, it must populate this prior to executing the same step during the playback. Hence, the purpose of this linkage type is to do service virtualization during the playback to ensure the transaction operates in the same manner.

An example of a clipboard import linkage displayed in the new investigate screen is shown below:



Linkage Type - Clipboard Import

The most important piece of information for the user here is the content of the clipboard that will be set in the bot during the playback.

3.4. Types of defects and how to identify them

This section takes you through some common types of Testimony defects and discusses how to categorise them and how to deal with them within your defect analysis process. Examples of the different defect types are shown.

3.4.1. Defects caused by changes in the release

Since the whole purpose of using Testimony is to perform a full regression test of your SAP system before a new release is deployed to production, it is quite likely that some defects will be detected which are expected: i.e., something has changed in the release being tested meaning that a transaction, batch job, RFC, etc., will not behave in the same way. Testimony will spot these and highlight them as failures, leading to defects being raised.

Note that from Testimony v2.21, Testimony's Root Cause Analysis (RCA) feature will be able, by analysing the release transports, to suggest which defects might be as a result of changes introduced by the release.

3.4.1.1. Example 1: A change to a message severity resulting in different output in a batch job

This defect has been raised because an unexpected message has been raised in a batch job.

Display Defect

Defect Header

Defect ID: 3275 new 123 Ext:
Assigned Team Role:
Assigned to: CHITJO
Current Status: Assigned
Task Priority: Unclassified
Short Description: Unexpected message in job in Batch job RNM_ZFI_JOURNAL

Header Comments History Execution Queue steps

Execution Steps for Defect

Phase	Executi...	EQ step	Res	Result	Reason	Display Step	Investigate Test Script	Sys
0001	53	683681		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	166	683794		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	288	683916		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	393	684069		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	534	684312		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	733	685124		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	1015	686916		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	1370	688941		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	1759	691369		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	2346	695131		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	2975	701342		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	3741	706956		Failed		Batch job	RNM_ZFI_JOURNAL	RP3
0001	8903	742538		Failed		Batch job	RNM_ZFI_JOURNAL	RP3

In the investigate screen, we can see that rows 3 and 4 in the job log are being highlighted as differences.

Expected Parameters	Value	Actual Parameters	Value
Job RNM_ZFI_JOURNAL - <ul style="list-style-type: none"> Job Log Entries <ul style="list-style-type: none"> Row 1 Row 2 Row 3 Row 4 Job Log Name 		Job RNM_ZFI_JOURNAL <ul style="list-style-type: none"> Job Log Entries <ul style="list-style-type: none"> Row 1 Row 2 Row 3 Row 4 Job Log Name 	
	DB7 dY/+ba5cIp\8QByQ		DB2 5RIM0UO-WfzwrQ#2

The last row of a job log is always the "Job cancelled" or "Job finished" message.

Expected Parameters	Value	Actual Parameters	Value
<ul style="list-style-type: none"> Job RNM_ZFI_JOURNAL - <ul style="list-style-type: none"> Job Log Entries <ul style="list-style-type: none"> Row 1 Row 2 Row 3 Row 4 <ul style="list-style-type: none"> ENTERTIME: 020001 ENTERDATE: 20191202 MSGID: 00 MSGNO: 564 MSGTYPE: A TEXT: Job canceled after system exception ERROR_MESSAGE RABAXKEY: 0 RABAXKEYLN: 0 MSGV1 MSGV2 MSGV3 MSGV4 PROGRAM PFKEY DYNPRO 		<ul style="list-style-type: none"> Job RNM_ZFI_JOURNAL <ul style="list-style-type: none"> Job Log Entries <ul style="list-style-type: none"> Row 1 Row 2 Row 3 Row 4 <ul style="list-style-type: none"> ENTERTIME: 151115 ENTERDATE: 20200129 MSGID: 00 MSGNO: 517 MSGTYPE: S TEXT: Job finished RABAXKEY: 0 RABAXKEYLN: 0 MSGV1 MSGV2 MSGV3 MSGV4 PROGRAM PFKEY DYNPRO 	
Job Log Name	DB7 dY/+ba5cIp\8QByQ	Job Log Name	DB2 SRIM0UO-WfzwrQ#2

We can see here that in the recording the job was cancelled because it received an error message. However, in the playback the job finished normally. We now need to look at the 3rd row in the job log to see if we can find out more detail about what went on.

Expected Parameters	Value	Actual Parameters	Value
<ul style="list-style-type: none"> Job RNM_ZFI_JOURNAL - <ul style="list-style-type: none"> Job Log Entries <ul style="list-style-type: none"> Row 1 Row 2 Row 3 <ul style="list-style-type: none"> ENTERTIME: 020001 ENTERDATE: 20191202 MSGID: 00 MSGNO: 001 MSGTYPE: E TEXT: No data exists RABAXKEY: 0 RABAXKEYLN: 0 MSGV1: No data exists MSGV2 MSGV3 MSGV4 PROGRAM PFKEY DYNPRO 		<ul style="list-style-type: none"> Job RNM_ZFI_JOURNAL <ul style="list-style-type: none"> Job Log Entries <ul style="list-style-type: none"> Row 1 Row 2 Row 3 <ul style="list-style-type: none"> ENTERTIME: 151115 ENTERDATE: 20200129 MSGID: 00 MSGNO: 001 MSGTYPE: S TEXT: No data exists RABAXKEY: 0 RABAXKEYLN: 0 MSGV1: No data exists MSGV2 MSGV3 MSGV4 PROGRAM PFKEY DYNPRO 	

Here it seems that the only difference is that the MSGTYPE has changed from an E in the recording to an S in the playback. MSGTYPE denotes the severity of a message, and batch jobs will terminate if they encounter a message with MSGTYPE=E (unless the message is otherwise handled in the code). We can also see that the message text "No data exists" is the same in both the recording and the playback.

What has happened here is that in the release, the message being generated when the batch job finds no data to work on has changed from type E to type S. This job is a periodic job, running every 15 minutes to

upload data into SAP from files on the application server. Since there will not always be files to process every time the job runs, a “No data exists” condition is actually normal. The Basis team requested this change to the message output when there is no data to process to prevent multiple unnecessary job cancellations, as they are required to investigate these.

Since this defect was as an expected result of a change being introduced in the new release, it can be ignored.

It should also be noted that any future recordings (and playbacks based on them) that are taken after this change has been imported into production will no longer raise this defect.

3.4.1.2. Example 2: Authorisation failures

The following defect was raised because of an unexpected message in transaction FEB_BSPROC.

The screenshot shows the 'Display Defect' window with the following details:

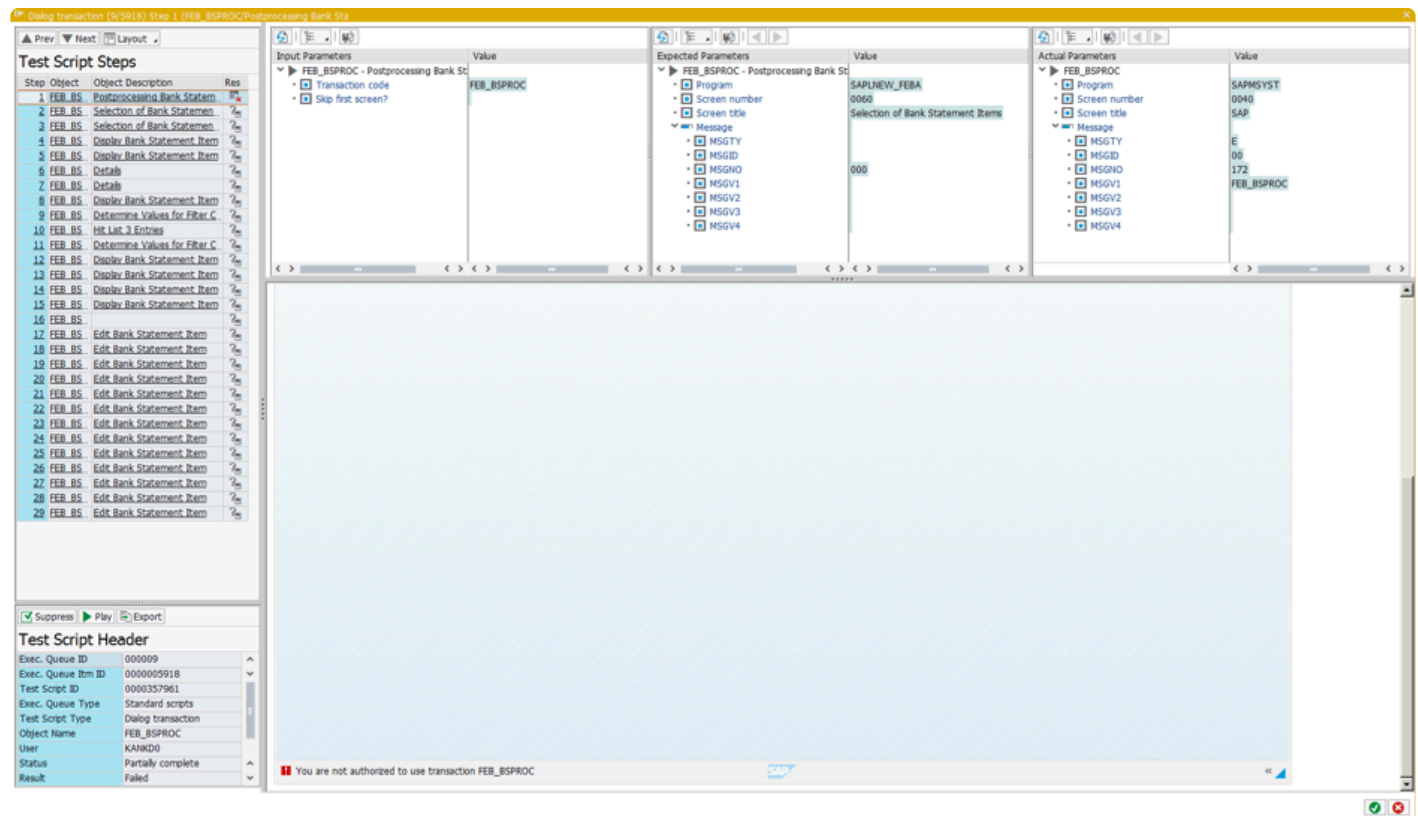
Defect Header	
Defect ID	3311
Current Status	Assigned
Short Description	FEB_BSPROC : Unexpected message : S(APMSYST)
Assigned Team Role	
Assigned to	KUMAU0
Task Priority	Unclassified

Navigation tabs: Header, Comments, History, Execution Queue steps

Key information	
Test Plan	RP3-RZ3: 2020-01
Test Phase	0001
System	RP3 - RP3 Source system
Task Source	Testimony
Software Component	Financial Accounting
Sub-Component	(FI-BL-PT-BS-EL) Electronic Bank Sta...
Object Type	Dialog transaction
Object Name	FEB_BSPROC
Failure Reason	Unexpected message
Task Type	Defect

SAP GUI Dialog Transaction details	
Transaction Code	FEB_BSPROC
Program Name	
Screen Number	
Fieldname	

Looking at the Investigate Screen, we can see that the message was displayed at the start of the transaction, and that the user was left on the SAP Easy Access screen. The screen shot shows us that the message was "You are not authorised to use transaction FEB_SPROC".



This is another example of Testimony highlighting exactly the type of change in behaviour that you want to test for in a regression test. Since Testimony uses the real production userids with their exact production roles and authorisations, any changes to their authorisations that will have been introduced by the release will be seen during the Testimony playback. It may be that a change to one of the roles assigned to this user has deliberately removed their authorisation to execute this transaction, in which case this is an expected failure. However, it may be that a role change (or perhaps an organisational structure change) done for some other reason has had the unintended consequence of removing this user's authorisation to perform part of their job. This will need to be investigated with the security team. (Again, from v.2.21 the Root Cause Analysis functionality will be able to tie the role change to a specific transport, making it easier to determine whether or not this was an intended or unintended consequence of the change.)

3.4.2. Defects arising from data issues

A Testimony playback is started on a system which is a copy of production taken at the time of the start of the recording. Because of this, we can be confident that all of the data that was in the system when the recording started is exactly replicated in the playback system at the start of the playback.

However, throughout the course of the playback it is possible that the data state slowly drifts away from the production state. The following sections explain some common causes of this.

3.4.2.1. Testimony playback sequencing

In order to be able to perform a playback in less time than it took to record the data, Testimony doesn't necessarily play back in the strict sequence of recorded events, except in certain specific cases:

- Batch jobs are always played back in strict sequence, so if Testimony recorded Job A, followed by Job B, followed by Job C, then the playback will play these jobs back in the order A, B, C.
- Transactions within a user session are always played back in strict sequence. If a user logged on and then executed transactions VA01, VA02 and VA01 again in that order, then Testimony will play back that user's session in the order VA01, VA02, VA01.
- "Dynamic ID" dependencies are always maintained. Testimony has functionality which can generate "Dynamic IDs" between transactions, batch jobs, RFCs, etc., and these are always enforced. For example, if UserA creates order 1234, and then UserB changes order 1234, and then UserC displays order 1234, then Testimony will ensure that these transactions are played back in the correct order.
 - A further piece of functionality related to Dynamic IDs handles what happens when, for example, the order creation fails. So in our example above if the creation of order 1234 fails, then Testimony can use the Dynamic ID for this order, and its link to other transactions, to recognise that there is no point in trying to either change or display this order as it doesn't exist in the playback system. Testimony will therefore cancel the dependent change and display transactions.
 - Dynamic ID also handles the possibility of differences in document numbers between the recording and the playback. For example, let's say that document 1234 was created during the recording and then it was changed later on. Now, when we come to play back this transaction, we may find that the document created in the playback system actually has the number 1233. (This might be because a previous document creation failed, meaning that the document number range has not been incremented.) Testimony knows that the change document transaction for the original document 1234 is dependent on this create document transaction. Since we now have a different document number (1233) Testimony will change the "change document" script to use this document number rather than the original.

This enables Testimony to "compress" the time taken to perform the playback to sometimes much less than the recording time. For example, if two batch jobs are executed 30 minutes apart, then Testimony is likely to start one as soon as the previous one has finished. Likewise, if a user executes VA01 and then, 10 minutes later, executes VA02, then Testimony is likely to start VA02 as soon as VA01 has finished.

An example of where Testimony might not play back in strict sequence might be where a batch job changes the status of a record, which is later acted upon by a dialog transaction. For example, TransactionA creates

a new order; BatchJobB processes this order and updates its status (to, for example, “ready for shipping”); and TransactionC performs the shipping of this order.

In this case, we have a mixture of dependent and independent processes happening: TransactionC is dependent on TransactionA (because of the Dynamic ID link for the order), but BatchJobB is independent of either. It is therefore possible in this example that instead of the recording sequence:

TransactionA → BatchJobB → TransactionC

We instead play back in the following sequence:

TransactionA → TransactionC → BatchJobB

Or:

BatchJobB → TransactionA → TransactionC

In the first example of an “incorrect” sequence, we would expect TransactionC to fail, as the order created by TransactionA is not in the correct status for shipping (as BatchJobB has not yet run). In the second example, we might expect TransactionC to fail for the same reason as before, but we may also see that BatchJobB fails because it doesn’t have a record to process (as TransactionA has not yet created the order), leading to a different output for the job.

3.4.2.2. Previously failed transactions

As mentioned above, the Dynamic ID process does a good job of ensuring that if, for example, a “Create Order” transaction fails during the playback, then subsequent Display or Change transactions for that same order are cancelled. However, some interdependencies between transactions are not so straightforward and this can lead to transaction failures during the playback which are not covered by the Dynamic ID process.

A good example of this is list processing, where a user calls up a list of available orders of a particular status (e.g., orders ready for fulfilment). It may be that when the user executed this transaction in production there were 15 orders in the list; however, during the playback one of the previous Create Order transactions failed, meaning that there are now only 14 orders in the list. If we try to double-click on the 15th order in the playback, we will get an error.

Batch jobs also provide a good example of where previously failed transactions may result in a playback error. Again, it may be that when the batch job ran in production there were 15 orders for it to process, but during the playback only 14 orders were processed because a previous transaction failed. This will be flagged by Testimony as a difference – and hence a script failure – as the batch job log will only contain entries for 14 records rather than the 15 that were expected.

3.4.2.3. Example 3: Batch job with different output

This defect has been raised because Testimony detected a different output in the batch job log.

Display Defect

Defect Header

Defect ID

3281

new 123

Ext

Assigned Team Role

Assigned to

KUMAU0

Current Status

Cancelled

Task Priority

Unclassified

Short Description

Unexpected message in job in Batch job ECC_CP_PO_ALL_PO_ZLV LOWV_IS

Header

Comments

History

Execution Queue steps

Execution Steps for Defect

Phase	Executi..	EQ step	Res	Result	Reason	Display Step	Investigate Test Script	Sys
0001	3312	703586		Failed		Batch job	ECC_CP_PO_ALL_PO_ZLV LOWV_IS	RP3

Defect record for a batch job failure

If we look at the investigate screen, we can see that there were 7 rows in the job log in production, whereas there were only 4 rows in the log during the playback.

Expected Parameters	Value	Actual Parameters	Value
<ul style="list-style-type: none"> Job Log Entries <ul style="list-style-type: none"> Row 1 Row 2 Row 3 Row 4 Row 5 Row 6 Row 7 Job Log Name 	DBS TMfPuwcDXCgEmEsz	<ul style="list-style-type: none"> Job Log Entries <ul style="list-style-type: none"> Row 1 Row 2 Row 3 Row 4 Job Log Name 	DB6 y0vWeZx(69XChb)n

Expected & Actual parameters (the job log)

The first two rows in a job log are always the Job Started and Step Started entries, so these can be ignored in our analysis (and indeed, these are identical in both the recording and the playback).

Expected Parameters	Value	Actual Parameters	Value
<ul style="list-style-type: none"> Job Log Entries <ul style="list-style-type: none"> Row 1 <ul style="list-style-type: none"> ENTERTIME: 122541 ENTERDATE: 20191202 MSGID: 00 MSGNO: 516 MSGTYPE: S TEXT: Job ECC_CP_PO_ALL_PO_ZLV LOWV_IS RABAXKEY: 0 RABAXKEYLN: 0 MSGV1: ECC_CP_PO_ALL_PO_ZLV LOWV_IS MSGV2: 12104100 MSGV3: MSGV4: PROGRAM: PFKEY: DYNPRO: Row 2 <ul style="list-style-type: none"> ENTERTIME: 122541 ENTERDATE: 20191202 MSGID: 00 MSGNO: 550 MSGTYPE: S TEXT: Step 001 started (program RM06BB30, v RABAXKEY: 0 RABAXKEYLN: 0 MSGV1: 001 MSGV2: RM06BB30 MSGV3: ZLV_IS MSGV4: FIPO_X PROGRAM: PFKEY: DYNPRO: 		<ul style="list-style-type: none"> Job Log Entries <ul style="list-style-type: none"> Row 1 <ul style="list-style-type: none"> ENTERTIME: 161546 ENTERDATE: 20200129 MSGID: 00 MSGNO: 516 MSGTYPE: S TEXT: Job ECC_CP_PO_ALL_PO_ZLV LOWV_IS RABAXKEY: 0 RABAXKEYLN: 0 MSGV1: ECC_CP_PO_ALL_PO_ZLV LOWV_IS MSGV2: 16154600 MSGV3: MSGV4: PROGRAM: PFKEY: DYNPRO: Row 2 <ul style="list-style-type: none"> ENTERTIME: 161546 ENTERDATE: 20200129 MSGID: 00 MSGNO: 550 MSGTYPE: S TEXT: Step 001 started (program RM06BB30, v RABAXKEY: 0 RABAXKEYLN: 0 MSGV1: 001 MSGV2: RM06BB30 MSGV3: ZLV_IS MSGV4: FIPO_X PROGRAM: PFKEY: DYNPRO: 	

Looking at rows 3 and 4, however, we can see that in the recording the batch job output some messages relating to purchase orders that it was processing, whereas in the playback the job output a “No suitable

purchase requisitions found” message and then finished.

Expected Parameters	Value	Actual Parameters	Value
Job ECC_CP_PO_ALL_PO_ZLV LOWV_IS		Job ECC_CP_PO_ALL_PO_ZLV LOWV_IS	
Job Log Entries		Job Log Entries	
Row 1		Row 1	
Row 2		Row 2	
Row 3		Row 3	
ENTERTIME	122542	ENTERTIME	161546
ENTERDATE	20191202	ENTERDATE	20200129
MSGID	06	MSGID	ME
MSGNO	218	MSGNO	261
MSGTYPE	E	MSGTYPE	S
TEXT	Net price must be greater than 0	TEXT	No suitable purchase requisitions found
RABAXKEY		RABAXKEY	
RABAXKEYLN	0	RABAXKEYLN	0
MSGV1		MSGV1	
MSGV2		MSGV2	
MSGV3		MSGV3	
MSGV4		MSGV4	
PROGRAM		PROGRAM	
PFKEY		PFKEY	
DYNPRO		DYNPRO	
Row 4		Row 4	
ENTERTIME	122542	ENTERTIME	161546
ENTERDATE	20191202	ENTERDATE	20200129
MSGID	06	MSGID	00
MSGNO	218	MSGNO	517
MSGTYPE	E	MSGTYPE	S
TEXT	Net price must be greater than 0	TEXT	Job finished
RABAXKEY		RABAXKEY	
RABAXKEYLN	0	RABAXKEYLN	0
MSGV1		MSGV1	
MSGV2		MSGV2	
MSGV3		MSGV3	
MSGV4		MSGV4	
PROGRAM		PROGRAM	
PFKEY		PFKEY	
DYNPRO		DYNPRO	

The “Job finished” message in the recording (which is always the last entry in the log for a job that ran successfully) can actually be found at row 7:

▼	Row 7	
▪	ENTERTIME	122542
▪	ENTERDATE	20191202
▪	MSGID	00
▪	MSGNO	517
▪	MSGTYPE	S
▪	TEXT	Job finished
▪	RABAXKEY	
▪	RABAXKEYLN	0
▪	MSGV1	
▪	MSGV2	
▪	MSGV3	
▪	MSGV4	
▪	PROGRAM	
▪	PFKEY	
▪	DYNPRO	

The "Job finished" message

This is a classic example of a data-related failure. In this case, either because of script sequencing or the failure of one or more previous transactions, when this job ran during the playback there were no records that met the selection criteria for the job.

3.4.2.4. Example 4: Inbound RFC with different output

This defect has been raised because Testimony detected a different output in the RFC.

Defect Header

Defect ID	2942	Ext	123	Assigned Team Role	
Current Status	Cancelled	Assigned to	KUMAU0	Task Priority	Unclassified
Short Description	Difference in output values in RFC /OTX/PF01_IF_GET_NEW_DOC_LIST				

Header | Comments | History | **Execution Queue steps**

Execution Steps for Defect

Phase	Executi...	EQ step	Res	Result	Reason	Display Step	Investigate Test Script	Sys
0001	14895	800413	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	14941	801169	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	14970	801268	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15422	803916	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15500	805521	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15573	805774	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15686	806107	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15809	807113	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15822	807138	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15833	807167	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15854	807286	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15892	807616	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15907	807690	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15922	807726	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	15941	807784	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3
0001	16002	808002	✗	Failed		RFC	/OTX/PF01_IF_GET_NEW_DOC_LIST	RP3

Defect record for an inbound RFC failure

Looking at the investigate screen, we can see that Testimony has highlighted differences in the PT_JOB_LIST and PT_JOB_METADATA elements of the output.

Expected Parameters	Value	Actual Parameters	Value
<div> <div>OTX/PF01_IF_GET_NEW_DOC_LIST</div> <div> <div>PE_RC</div> <div>PT_JOB_LIST</div> <div>PT_JOB_METADATA</div> <div>PT_JOB_QUERY</div> <div>PT_RETURN</div> </div> </div>	0	<div> <div>OTX/PF01_IF_GET_NEW_DOC_LIST</div> <div> <div>PE_RC</div> <div>PT_JOB_LIST</div> <div>PT_JOB_METADATA</div> <div>PT_JOB_QUERY</div> <div>PT_RETURN</div> </div> </div>	0

In this case, the PT_JOB_LIST output area is the most important to look at. (The PT_JOB_METADATA area just defines the fields within the PT_JOB_LIST, so if there are differences in PT_JOB_LIST then there will always be differences in PT_JOB_METADATA.)

Expected Parameters	Value	Actual Parameters	Value
▼ /OTX/PF01_IF_GET_NEW_DOC_LIST		▼ /OTX/PF01_IF_GET_NEW_DOC_LIST	
▪ PE_RC	0	▪ PE_RC	0
▼ PT_JOB_LIST		▼ PT_JOB_LIST	
> Row 1		> Row 1	
> Row 2		> Row 2	
> PT_JOB_METADATA		> PT_JOB_METADATA	
> PT_JOB_QUERY		> PT_JOB_QUERY	
> PT_RETURN		> PT_RETURN	

Here we can see that in the recording the RFC had two entries in PT_JOB_LIST, whereas in the playback there were 6.

Looking at what's contained within these rows, we can see that each row relates to a particular document (specified by the DOCID field).

Expected Parameters	Value	Actual Parameters	Value
<ul style="list-style-type: none"> OTX/PF01_IF_GET_NEW_DOC_LIST <ul style="list-style-type: none"> PE_RC <ul style="list-style-type: none"> PT_JOB_LIST <ul style="list-style-type: none"> Row 1 <ul style="list-style-type: none"> DOCID: 000000008861 RETRY_NO: 0001 IMAGE_URL: http://eunatan01.com:808... IMAGE_DOCTYPE: ZV_LO_ICCP DOC_URL: http://eunatan01.com:808... DOC_DOCTYPE Row 2 <ul style="list-style-type: none"> DOCID: 000000008862 RETRY_NO: 0001 IMAGE_URL: http://eunatan01.com:808... IMAGE_DOCTYPE: ZV_LO_ICCP DOC_URL: http://eunatan01.com:808... DOC_DOCTYPE PT_JOB_METADATA PT_JOB_QUERY PT_RETURN 		<ul style="list-style-type: none"> OTX/PF01_IF_GET_NEW_DOC_LIST <ul style="list-style-type: none"> PE_RC <ul style="list-style-type: none"> PT_JOB_LIST <ul style="list-style-type: none"> Row 1 <ul style="list-style-type: none"> DOCID: 000000004163 RETRY_NO: 0001 IMAGE_URL: http://eunatan01.com:808... IMAGE_DOCTYPE: ZV_GH_ICCP DOC_URL: http://eunatan01.com:808... DOC_DOCTYPE Row 2 <ul style="list-style-type: none"> DOCID: 000000004164 RETRY_NO: 0001 IMAGE_URL: http://eunatan01.com:808... IMAGE_DOCTYPE: ZV_GH_ICCP DOC_URL: http://eunatan01.com:808... DOC_DOCTYPE Row 3 <ul style="list-style-type: none"> DOCID: 000000006099 RETRY_NO: 0001 IMAGE_URL: http://eunatan01.com:808... IMAGE_DOCTYPE: ZV_GH_ICCP DOC_URL: http://eunatan01.com:808... DOC_DOCTYPE Row 4 <ul style="list-style-type: none"> DOCID: 000000007964 RETRY_NO: 0001 IMAGE_URL: http://eunatan01.com:808... IMAGE_DOCTYPE: ZV_GH_ICCP DOC_URL: http://eunatan01.com:808... DOC_DOCTYPE Row 5 <ul style="list-style-type: none"> DOCID: 000000008494 RETRY_NO: 0001 IMAGE_URL: http://eunatan01.com:808... IMAGE_DOCTYPE: ZV_CT_ICCP DOC_URL: http://eunatan01.com:808... DOC_DOCTYPE Row 6 <ul style="list-style-type: none"> DOCID: 000000008579 RETRY_NO: 0001 IMAGE_URL: http://eunatan01.com:808... IMAGE_DOCTYPE: ZV_GH_ICCP DOC_URL: http://eunatan01.com:808... DOC_DOCTYPE 	

This defect, therefore, has been raised because, either due to sequencing issues or a previously failed transaction, there was a difference in the number of available orders that could be processed by this RFC.

3.4.2.5. Example 5: Double-clicking on a list item calls up a different document

This defect has been raised because a different screen was shown in the playback than was seen in the recording.

The screenshot shows the 'Display Defect' window with the following details:

- Defect ID:** 2972
- Ext:** 123
- Assigned Team Role:** (empty)
- Assigned to:** (empty)
- Task Priority:** Unclassified
- Current Status:** Assigned
- Short Description:** SBWP : Unexpected next screen : /OPT/SAPLVIM_IDX_UI 1000

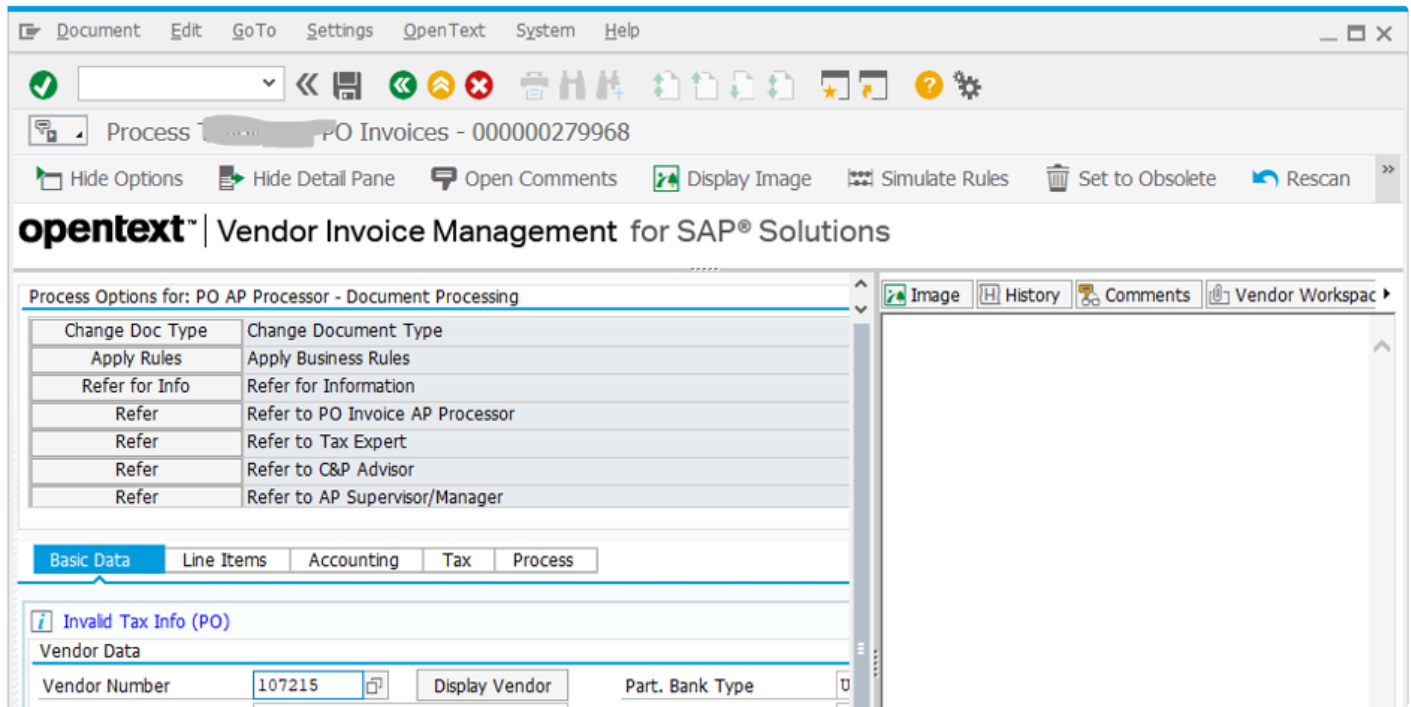
The 'Execution Queue steps' tab is selected, showing a table of execution steps:

Phase	Executi..	EQ step	Res	Result	Reason	Display Step	Investigate Test Script	Sys
0001	1139	687882	✖	Failed		GUI dialog step	SBWP	RP3
0001	1665	690669	✖	Failed		GUI dialog step	SBWP	RP3
0001	1942	692438	✖	Failed		GUI dialog step	SBWP	RP3
0001	2458	696161	✖	Failed		GUI dialog step	SBWP	RP3
0001	2503	696395	✖	Failed		GUI dialog step	SBWP	RP3
0001	2959	701221	✖	Failed		GUI dialog step	SBWP	RP3
0001	3169	702510	✖	Failed		GUI dialog step	SBWP	RP3
0001	3275	703189	✖	Failed		GUI dialog step	SBWP	RP3
0001	3784	707182	✖	Failed		GUI dialog step	SBWP	RP3
0001	4169	710855	✖	Failed		GUI dialog step	SBWP	RP3
0001	4608	714213	✖	Failed		GUI dialog step	SBWP	RP3
0001	5116	720043	✖	Failed		GUI dialog step	SBWP	RP3
0001	5875	726434	✖	Failed		GUI dialog step	SBWP	RP3
0001	5997	727518	✖	Failed		GUI dialog step	SBWP	RP3
0001	6630	733246	✖	Failed		GUI dialog step	SBWP	RP3
0001	7117	736047	✖	Failed		GUI dialog step	SBWP	RP3

Looking at the Investigate Screen, we can see that instead of going to a “Decision step in workflow” screen, we went to a “Process PO Invoices” screen.

Expected Parameters	Value	Actual Parameters	Value
SBWP - Business Workplace of		SBWP	
Message		Message	
Next Program	SAPLSWU2	Next Program	/OPT/SAPLVIM_IDX_UI
Next Screen Nr.	0300	Next Screen Nr.	1000
Next Screen title	Decision Step in Workflow	Next Screen title	Process PO Invoices - 0000002..

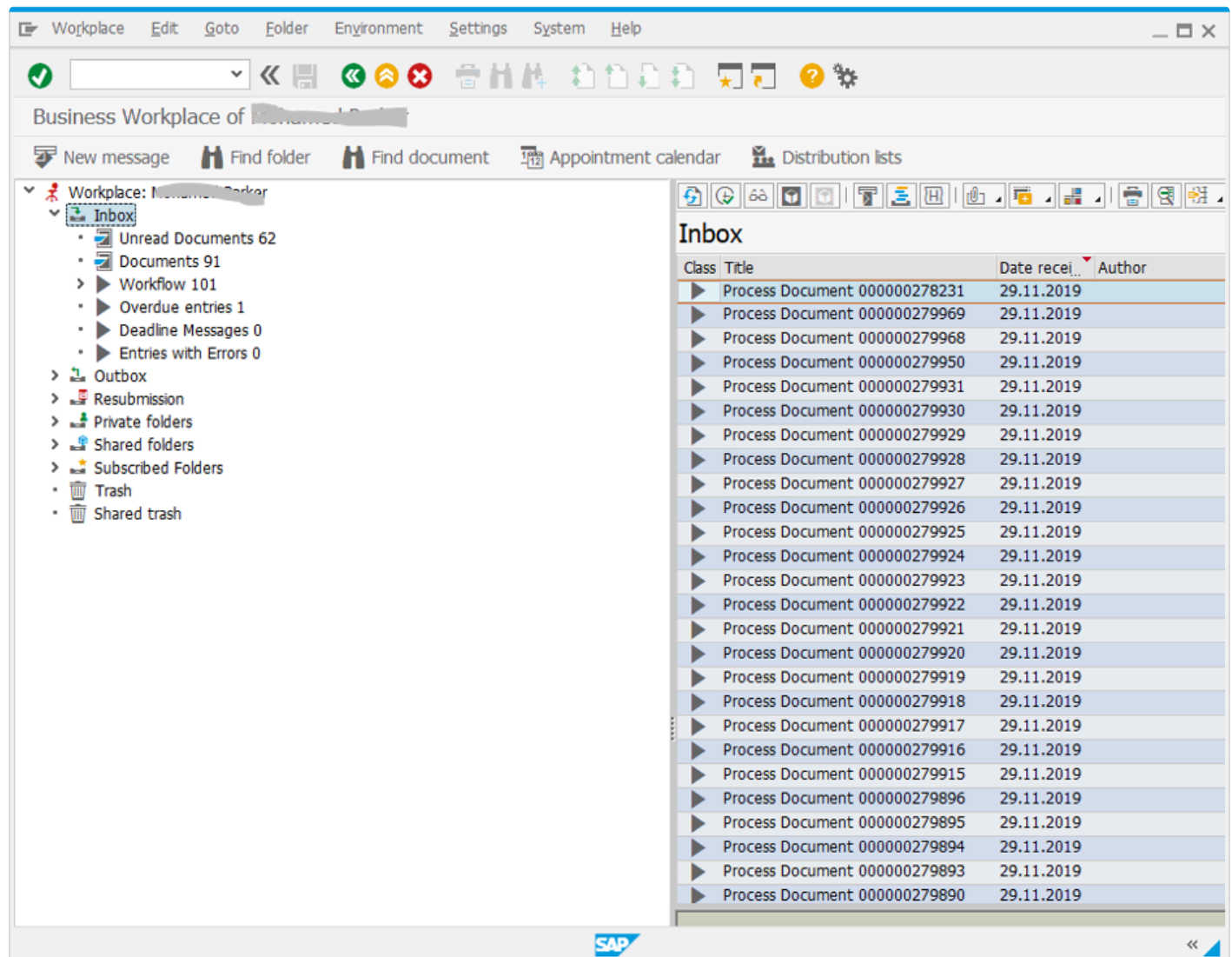
The screen that we saw in the playback looked like this.



If we look at the Input Parameters for this step, we can see that the user double-clicked on a field on the previous screen (as shown in the "Control event/action" parameter).






















Input Parameters	Value
SBWP - Business Workplace of	
▪ Main program	SAPLSINWP
▪ Main screen number	1000
▪ Control ID	cntlSINWP_CONTAINER/shellcont/shell/shellcont[1]
▪ Control type	GridView
▪ Control event/action	Double click (0003)
> Control event parameters	
> Control Properties	

And looking at the screen shot from the previous screen, we can see that the user was working with a list of items from their Business Workplace inbox.



In order to figure out exactly what's happened here, we'll need to drill into the information in the Investigate Screen in a bit more detail (and gain a better understanding of Control processing). It is also helpful to be able to recreate this issue in the playback system.

The first thing we should try to do is work out which item in the list the user double-clicked on. We can do this by looking at the Control Properties on the Input Parameters for the failed step.

Input Parameters	Value
 SBWP - Business Workplace of I	
▪  Main program	SAPLSINWP
▪  Main screen number	1000
▪  Control ID	cntlSINWP_CONTAINER/shellcont/shell/shellc
▪  Control type	GridView
▪  Control event/action	Double click (0003)
>  Control event parameters	
 Control Properties	
 Row 1	
▪  CONTROL_ID	cntlSINWP_CONTAINER/shellcont/shell/shellc
▪  CONTROL_CLASS	GridView
 PROPERTIES	
 Row 1	
▪  NAME	CurrentCellCol
▪  VALUE	2
 Row 2	
▪  NAME	CurrentCellRow
▪  VALUE	3
 Row 3	
▪  NAME	CurrentCellText
▪  VALUE	Approval needed: PR No. 10043952 for ZAR

Here we can see the following information.

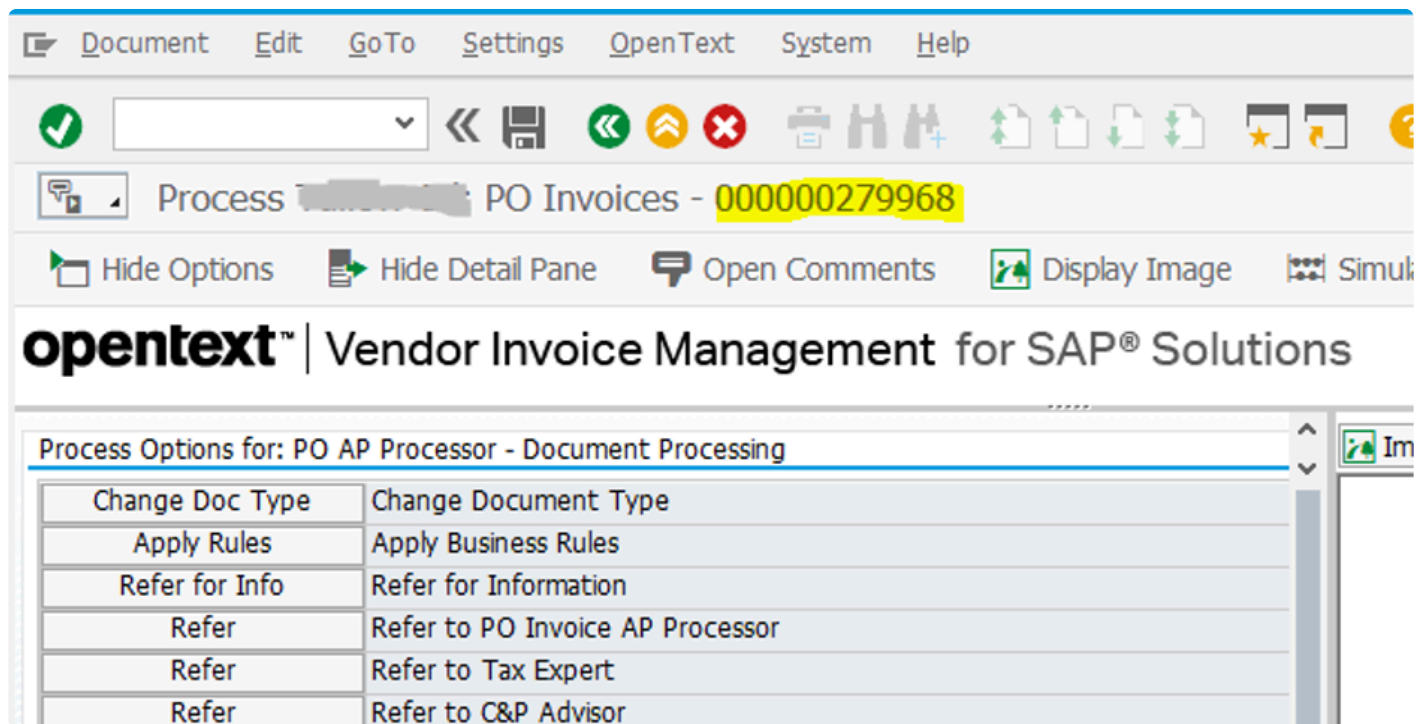
- Row 1 shows us that the user double-clicked on a cell in column 2 of the list
- Row 2 shows us that the user double-clicked on a cell in row 3 of the list
- Row 3 shows us that the cell the user double-clicked on had the text “Approval needed: PR No. 10043952 ...”

We now need to compare this with the list we saw displayed in screen shot from the previous step:

Inbox			
Class	Title	Date recei...	Author
▶	Process Document 000000278231	29.11.2019	
▶	Process Document 000000279969	29.11.2019	
▶	Process Document 000000279968	29.11.2019	
▶	Process Document 000000279950	29.11.2019	
▶	Process Document 000000279931	29.11.2019	
▶	Process Document 000000279930	29.11.2019	
▶	Process Document 000000279929	29.11.2019	
▶	Process Document 000000279928	29.11.2019	
▶	Process Document 000000279927	29.11.2019	
▶	Process Document 000000279926	29.11.2019	
▶	Process Document 000000279925	29.11.2019	
▶	Process Document 000000279924	29.11.2019	
▶	Process Document 000000279923	29.11.2019	
▶	Process Document 000000279922	29.11.2019	
▶	Process Document 000000279921	29.11.2019	
▶	Process Document 000000279920	29.11.2019	
▶	Process Document 000000279919	29.11.2019	
▶	Process Document 000000279918	29.11.2019	
▶	Process Document 000000279917	29.11.2019	
▶	Process Document 000000279916	29.11.2019	
▶	Process Document 000000279915	29.11.2019	
▶	Process Document 000000279896	29.11.2019	
▶	Process Document 000000279895	29.11.2019	
▶	Process Document 000000279894	29.11.2019	
▶	Process Document 000000279893	29.11.2019	
▶	Process Document 000000279890	29.11.2019	

The highlighted entry is the cell at column 2, row 3, and as you can see this has the text “Process document 000000279968”. It looks, therefore, as though the list from which we were working in the playback was different to the list the user was working from in the recording.

If we now look at the screen shot of the step that failed, we can see that we are actually working on document 279968.



We can therefore conclude that the playback bot was doing what it was being asked to do: double-click on the cell in column 2, row 3 in the list. It's just that the entry in the list was not the one that we were expecting.

It's now worthwhile doing some investigation in the playback system to try to pinpoint what went on. Looking again at the list displayed in the user's SBWP inbox, we can see that all of the “Process document” messages were generated on the same date. Drilling down into the workflow logs for these entries on the playback system, we can see that they were all created (i.e., added to the user's inbox) in a batch, with only a second or two between each entry. This suggests that a batch job has performed some processing to add these documents to the user's inbox. Further investigation revealed that a workflow deadline monitoring job ran at the time these entries were added to the user's inbox.

We therefore have a classic example of a data issue being caused by the way that Testimony sequences batch jobs and online transactions. This resulted in a list from which we were working not being the same in the playback as it was in the recording.

3.4.3. Defects arising from sequencing issues

As well as data issues arising from the sequencing of scripts in the playback (discussed above), other issues relating to sequencing may arise.

3.4.3.1. Locking issues

Object locks (using SAP's enqueue methods) are taken whenever a user goes to change an object (a purchase order, for example). Locks can be seen in both the recording and the playback, but because of the way that Testimony sequences the scripts in the playback, it is possible that:

- A lock seen in the recording is not seen in the playback, or
- A lock is seen in the playback that was not seen in the recording

Because of Testimony's Dynamic ID functionality, the second type is much less likely to occur than the first.

3.4.3.2. Example 6: A lock in the recording not seen in the playback

This defect was raised because a message seen in the recording was not seen in the playback.

Display Defect

Defect ID3187123Ext

Assigned Team Role

Assigned to

Task PriorityUnclassified

Current StatusAssigned

Short DescriptionZCP_REQ_TRACK : No message received. Expected - S006(ME)

Header

Comments

History

Execution Queue steps

Key information

Test PlanRP3-RZ3: 2020-01

Test Phase0001

SystemRP3 - RP3 Source system

Task SourceTestimony

Software Component

Sub-Component

Object TypeDialog transaction

Object NameZCP_REQ_TRACK

Failure ReasonUnexpected message

Task TypeDefect

SAP GUI Dialog Transaction details

Transaction CodeZCP_REQ_TRACK

Program NameSAPIMEGUI

Screen Number0014

FieldnameBDC_OKCODE

As can be seen from the Investigate Screen, the message in the recording was caused by the fact that a user was already processing the purchase requisition.

Expected Parameters	Value	Actual Parameters	Value
<div><div>ZCP_REQ_TRACK - Display Purchase Re</div><div><div>Message</div><div>Next Program</div><div>Next Screen Nr.</div><div>Next Screen title</div></div></div>	<div>User ADMIN0 already processing Purch. Requisition 10043631</div> <div>SAPLMEGUI</div> <div>0014</div> <div>Display Purchase Req. 10043631</div>	<div><div>ZCP_REQ_TRACK</div><div><div>Message</div><div>Next Program</div><div>Next Screen Nr.</div><div>Next Screen title</div></div></div>	<div>SAPLMEGUI</div> <div>0014</div> <div>Change Purchase Req. 10043631</div>

Because the playback processed this transaction in a different sequence than in the recording, the lock that

had occurred in the recording wasn't seen in the playback.

3.4.4. Defects arising from bot configuration issues

The bots (Windows applications) that are used to perform the playback require a certain minimum specification as well as several bits of recommended configuration in order to be fully able to execute all types of transactions. However, it is not necessarily desirable for all customers to set up the bots in ways that will allow the playback to always run successfully in all cases. In this section we will discuss common types of bot set-up issues that may be seen during your playbacks.

3.4.4.1. Microsoft Office installation

Almost all SAP users will have MS Office installed and configured on their local machines. This enables them to download data from SAP directly into MS Excel or even, in some transactions, display an embedded MS Excel spreadsheet or edit directly in MS Word from within their SAP screens. Some SAP transactions also have different buttons or menu options available depending on whether or not MS Office is installed on the local machine.

Ideally, MS Office will be installed and configured on all of the Windows machines (physical machines or VMs) that are used to host the Testimony bots. However, because of the cost involved in this some customers – especially those that need a large number of bots – will choose not to license MS Office on the bots. This can cause some particular types of failure during a playback which are discussed here.

3.4.4.2. Example 7: Excel button not available on screen

This defect has been raised because of an unexpected message:

Display Defect

Defect Header

Defect ID: 2951 NEW 123 Ext:

Assigned Team Role:

Assigned to: kevin callaghan

Task Priority: Unclassified

Current Status: Cancelled

Short Description: F.01 : Unexpected message : S255(00)

Header | Comments | History | **Execution Queue steps**

Execution Steps for Defect

Phase	Executi...	EQ step	Res	Result	Reason	Display Step	Investigate Test Script	Sys
0001	638	684696		Failed		GUI dialog step	F.01	RP3
0001	2468	696222		Failed		GUI dialog step	F.01	RP3
0001	6603	733111		Failed		GUI dialog step	F.01	RP3
0001	8830	742327		Failed		GUI dialog step	F.01	RP3
0001	10393	755526		Failed		GUI dialog step	F.01	RP3
0001	13101	780124		Failed		GUI dialog step	F.01	RP3
0001	13684	785317		Failed		GUI dialog step	F.01	RP3
0001	14437	795103		Failed		GUI dialog step	F.01	RP3
0001	14990	801572		Failed		GUI dialog step	F.01	RP3

Looking at the Investigate Screen, we can see that in the playback we received the message “Function code cannot be selected”.

Expected Parameters	Value	Actual Parameters	Value
<ul style="list-style-type: none"> F.01 - Financial Statements <ul style="list-style-type: none"> Message Next Program Next Screen Nr. Next Screen title 	SAPLSLVC_FULLSCREEN 0500 Financial Statements	<ul style="list-style-type: none"> F.01 <ul style="list-style-type: none"> Message Next Program Next Screen Nr. Next Screen title 	Function code cannot be selected SAPLSLVC_FULLSCREEN 0500 Financial Statements

If we look at the input parameters for the step, we can see that the function code that the bot was trying to execute was “Microsoft Excel (= &VEXCEL)”.

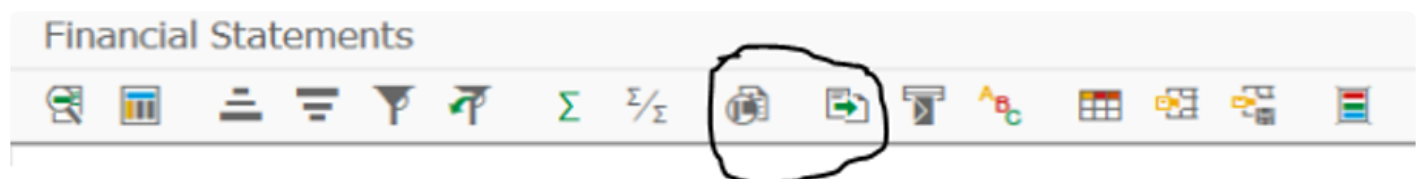
Input Parameters	Value
▼ F.01 - Financial Statements	
▪ Main program	SAPLSLVC_FULLSCREEN
▪ Main screen number	0500
▪ Function	Microsoft Excel (= &VEXCEL)
▪ Current window width	0237
▪ Current window height	0038
> Control Properties	

So, the bot was trying to click on a button to display the list in MS Excel, but that function was not available. This is because MS Office is not installed on the bot that executed this transaction. Because MS Office was not installed, two buttons on this screen are not available.

If we look at a screenshot of this screen run on a machine which has MS Office installed, we can see that there are two Office-related buttons



The first of the highlighted buttons is the Microsoft Excel button; the second is the Word Processing button. However, when we run this same transaction on the bot that doesn't have MS Office installed, you can see that these two buttons are missing:



The MS Office buttons should be between the two buttons circled above.

3.4.4.3. Example 8: No word processing program available

This defect was raised because the playback encountered an unexpected next screen.

Display Defect

Defect ID

3259

Ext

123

Assigned Team Role

Assigned to

KHANFO

Current Status

Assigned

Task Priority

Unclassified

Short Description

ME33K : Unexpected next screen : SAPLSTXX 1100

Header

Comments

History

Execution Queue steps

Key information

Test Plan

RP3-RZ3: 2020-01

Test Phase

0001

System

RP3 - RP3 Source system

Task Source

Testimony

Software Component

Materials Management

Sub-Component

(MM-PUR) Purchasing

Object Type

Dialog transaction

Object Name

ME33K

Failure Reason

Unexpected next screen

Task Type

Defect

SAP GUI Dialog Transaction details

Transaction Code

ME33K

Program Name

SAPMM06E

Screen Number

0103

Fieldname

BDC_OKCODE

Looking at the Investigate Screen, we can see that we saw a different screen and also received a message that wasn't seen in the recording.

Expected Parameters	Value	Actual Parameters	Value
ME33K - Display Contract : Header Text		ME33K	
Message		Message	No program available for processing document
Next Program	SAPLSTXX	Next Program	SAPLSTXX
Next Screen Nr.	2102	Next Screen Nr.	1100
Next Screen title	Display Release Order Text: 4700001256 L...	Next Screen title	Display Release Order Text: 4700001256 Language EN

The screen that was displayed during the playback was the standard SAP text editor.

Display Release Order Text: 4700001256 Language EN			
F..	L	Row Text	R
	1.....2.....3.....4.....5.....6.....7..	^
*		27/11/2019:	v
*		FWA amended to increase value by US\$11.014.000 and extend validity end	

This is another example of a defect being raised because MS Office was not installed on the bot that executed this script. In this case, a word processing program (MS Word) should have been called.

3.4.5. Defects arising from file download virtualisation

During a recording, Testimony will capture any user transactions that download files to a user's local desktop. In order to avoid potentially flooding the bot machines with downloaded files, Testimony virtualises these file downloads during the playback, intercepting the calls to the download functions in SAP and instead "masking" them to make SAP think that the file has been successfully downloaded.

In most cases, when a file is downloaded to the desktop a message of message class FES is displayed. (For example, message FES028 "200 bytes passed"). Because we know that we will not actually be downloading a file (and therefore that no FES message will be displayed) we use part of the message exclusions functionality to automatically pass steps that, in the recording, displayed one of these messages.

Unfortunately, SAP is not always consistent in its use of FES messages. In some transactions a generic message (00001) is used. This is a "catch-all" message that can be passed up to eight variables, but has no fixed text. Although it would be technically possible within Testimony to bypass this message, it is not desirable to do this as we cannot be certain under what circumstances it will be displayed.

Because of this, there will be cases where transactions that download files to the local desktop cannot be played back correctly.

3.4.5.1. Example 9: No message is displayed

The below defect has been raised because no message was received when one was expected.

Display Defect

Defect Header

Defect ID	2955	rev 123 Ext	Assigned Team Role	
Current Status	Cancelled		Assigned to	kevin callaghan
Short Description	CJ13 : No message received. Expected - S001(00)		Task Priority	Unclassified

Header | Comments | History | **Execution Queue steps**

Execution Steps for Defect

Phase	Executi...	EQ step	Res	Result	Reason	Display Step	Investigate Test Script	Sys
0001	781	685231		Failed		GUI dialog step	CJ13	RP3
0001	1905	692047		Failed		GUI dialog step	CJ13	RP3
0001	6906	735003		Failed		GUI dialog step	CJ13	RP3

Looking at the Investigate screen, we can see that the user was executing a download to a spreadsheet (Function = &XXL).

Input Parameters	Value
<ul style="list-style-type: none"> CJ13 - Display Actual Cost Line Items for <ul style="list-style-type: none"> <input type="checkbox"/> Main program <div>SAPLSLVC_FULLSCREEN</div> <input type="checkbox"/> Main screen number <div>0500</div> <input type="checkbox"/> Function <div> Spreadsheet... (= & XXL) </div> <input type="checkbox"/> Current window width <div>0146</div> <input type="checkbox"/> Current window height <div>0025</div> Control Properties <div></div> 	

And in the parameters, we can see that there was no message in the playback, where there was a message displayed in the recording.

Expected Parameters	Value	Actual Parameters	Value
▼ CJ13 - Display Actual Cost Line Items for Projects		▼ CJ13	
> Message	Download 2 MB SAPLSLVC_FULLSCREEN	> Message	
• Next Program	SAPLSLVC_FULLSCREEN	• Next Program	SAPLSLVC_FULLSCREEN
• Next Screen Nr.	0500	• Next Screen Nr.	0500
• Next Screen title	Display Actual Cost Line Items for Projects	• Next Screen title	Display Actual Cost Line Items for Projects

3.4.5.2. Example 10: “Setting was applied” message is displayed

In a similar vein to the previous example, in this defect we can see that a file download message was displayed in the recording, whilst a different message – “Setting was applied” – was shown in the playback.

Expected Parameters	Value	Actual Parameters	Value
▼ FBL5N - Select Spreadsheet		▼ FBL5N	
> Message	Download 191 KB N:\06. E	> Message	Setting was applied
▪ Next Program	SAPLSLVC_FULLSCREEN	▪ Next Program	SAPLSLVC_FULLSCREEN
▪ Next Screen Nr.	0500	▪ Next Screen Nr.	0500
▪ Next Screen title	Customer Line Item Display	▪ Next Screen title	Customer Line Item Display

3.4.6. Defects caused by functionality not supported by Testimony

Unfortunately, it is not (yet) possible for Testimony to support the recording or playback of all functionality within SAP. In most cases it is possible to exclude unsupported functions from a recording, either by not having a specific recording enhancement active (as is the case for outbound RFCs for example) or by specifically excluding some objects from a recording (for example, inbound RFCs generated by RPA tools).

However, there are some transactions within SAP which will often work during the playback *unless* the user chooses to use a particular piece of that transaction's functionality. An examples of this are given below.

3.4.6.1. Example 11: Drag & Drop

The following defect has been raised because the playback did not see a message that was shown in the recording.

Display Defect

Defect Header

Defect ID

2954

Ext

123

Assigned Team Role

Assigned to

kevin callaghan

Task Priority

Unclassified

Current Status

Cancelled

Short Description

OAWD : No message received. Expected - S170(OA)

Header

Comments

History

Execution Queue steps

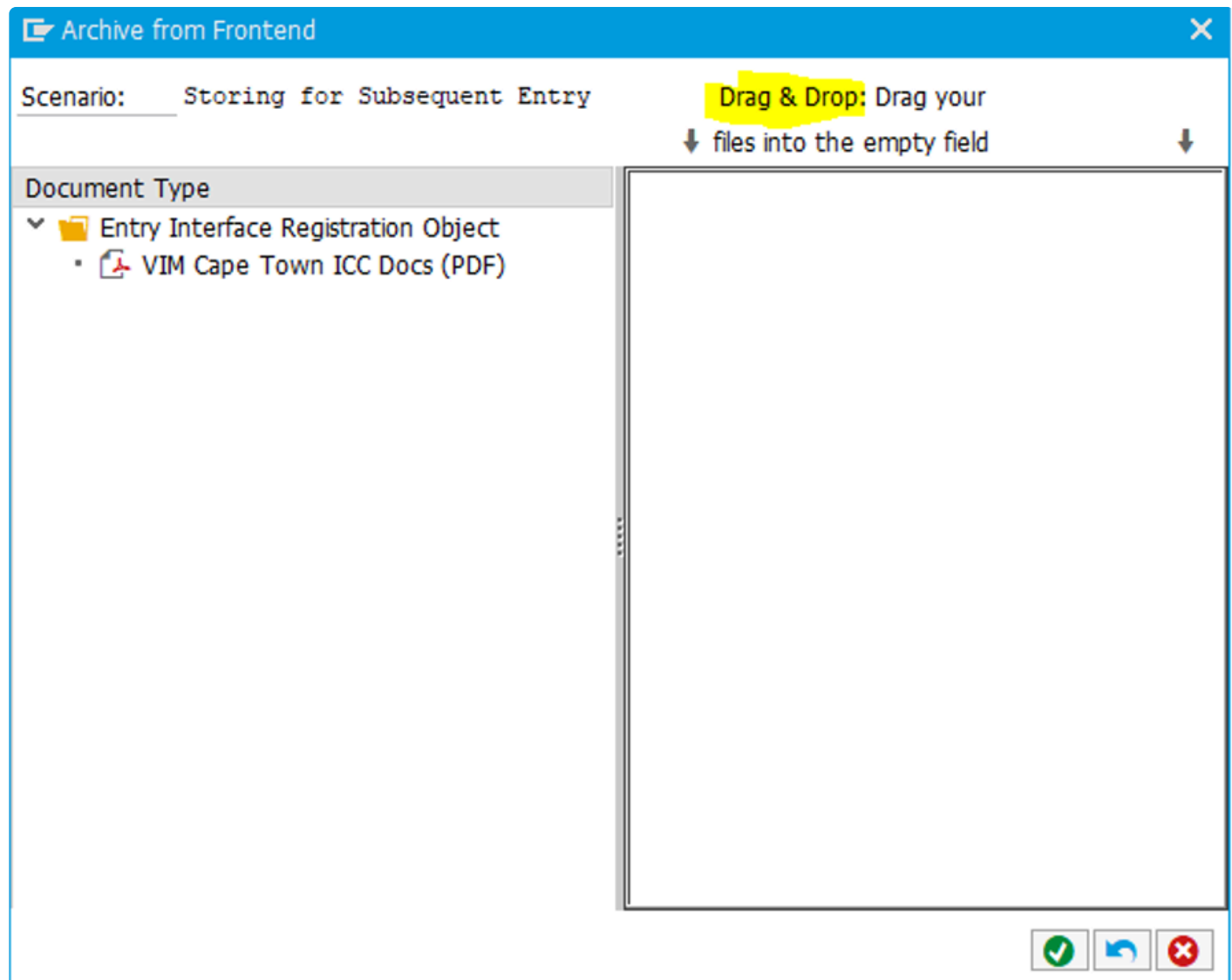
Execution Steps for Defect

Phase	Executi...	EQ step	Res	Result	Reason	Display Step	Investigate Test Script	Sys
0001	769	685194	✖	Failed		GUI dialog step	OAWD	RP3
0001	6969	735259	✖	Failed		GUI dialog step	OAWD	RP3
0001	11493	768517	✖	Failed		GUI dialog step	OAWD	RP3
0001	13802	786132	✖	Failed		GUI dialog step	OAWD	RP3
0001	14067	791557	✖	Failed		GUI dialog step	OAWD	RP3

Looking at the Investigate Screen, we can see that in the recording we captured a “Work item was created” message, whereas we didn’t see a message at all in the playback.

Expected Parameters	Value	Actual Parameters	Value
<div> <div>OAWD - Archive from Frontend</div> <div> <div>Message</div> <div>Next Program</div> <div>Next Screen Nr.</div> <div>Next Screen title</div> </div> </div>	<div>Work item 000007326434 was created</div> <div>SAPLALINK_DRAG_AND_DROP</div> <div>0100</div> <div>Archive from Frontend</div>	<div> <div>OAWD</div> <div> <div>Message</div> <div>Next Program</div> <div>Next Screen Nr.</div> <div>Next Screen title</div> </div> </div>	<div>SAPLALINK_DRAG_AND_DROP</div> <div>0100</div> <div>Archive from Frontend</div>

The screen shot in the Investigate Screen gives us a clue as to what happened here.



As you can see, this transaction, OAWD, has functionality to allow for the dragging and dropping of files into an archive. Drag & Drop is not currently supported by Testimony, so this defect can be ignored. Note, however, that OAWD has another option "Mass archiving" which allows for the upload of files from the front end via the standard Windows Explorer file selection screen. This would work in Testimony, so we can't simply exclude this whole transaction from subsequent recordings.

3.4.6.2. Example 12: Batch Input

The following defect was raised because the transaction FB05 failed to start properly.

Display Defect

Defect Header

Defect ID: 3312 123 Ext

Assigned Team Role:
Assigned to: kevin callaghan
Task Priority: Unclassified

Current Status: Assigned
Short Description: FB05 : Failed to start transaction

Header Comments History Execution Queue steps

Execution Steps for Defect

Phase	Executi...	EQ step	Res	Result	Reason	Display Step	Investigate Test Script	Sys
0001	7035	735571	✗	Failed		GUI dialog step	FB05	RP3
0001	7036	735572	✗	Failed		GUI dialog step	FB05	RP3
0001	7037	735573	✗	Failed		GUI dialog step	FB05	RP3
0001	7038	735574	✗	Failed		GUI dialog step	FB05	RP3
0001	7039	735575	✗	Failed		GUI dialog step	FB05	RP3
0001	7041	735577	✗	Failed		GUI dialog step	FB05	RP3
0001	7042	735578	✗	Failed		GUI dialog step	FB05	RP3
0001	7043	735579	✗	Failed		GUI dialog step	FB05	RP3
0001	7044	735580	✗	Failed		GUI dialog step	FB05	RP3
0001	7047	735583	✗	Failed		GUI dialog step	FB05	RP3
0001	7048	735584	✗	Failed		GUI dialog step	FB05	RP3
0001	7049	735585	✗	Failed		GUI dialog step	FB05	RP3
0001	7050	735586	✗	Failed		GUI dialog step	FB05	RP3
0001	8640	741651	✗	Failed		GUI dialog step	FB05	RP3
0001	8641	741652	✗	Failed		GUI dialog step	FB05	RP3
0001	8657	741702	✗	Failed		GUI dialog step	FB05	RP3

Looking at the Investigate Screen, we can see that we actually had an unexpected screen displayed at the start of the transaction.

Test Script Steps

Step	Object	Description	Res
1	FB05	Post with Clearing: Header Data	✗

Test Script Header

Exec. Queue ID: 000009
Exec. Queue Item ID: 0000007035
Test Script ID: 0000359221
Exec. Queue Type: Standard scripts
Test Script Type: Dialog transaction
Object Name: FB05
User: DUKCH0
Status: Complete
Result: Failed
Failure Reason: Unexpected next screen

Input Parameters

Parameter	Value
FB05 - Post with Clearing: Header Data	FB05

Expected Parameters

Parameter	Value
FB05 - Post with Clearing: Header Data	SAPMF05A
Program	0733
Screen number	Post with Clearing Enter selection crte
Screen title	
Message	

Actual Parameters

Parameter	Value
FB05	SAPMF05A
Program	0122
Screen number	Post with Clearing: Header Data
Screen title	
Message	

Some investigation into this showed that the screen that was displayed in the recording (screen 733 of SAPMF05A) is the special batch input screen for the FB05 transaction. At present Testimony doesn't support batch input as it currently isn't possible to tie the batch input parameters to the transaction execution in the playback. So, during the playback, when the bot received an instruction to start transaction FB05 it ran it as a normal dialog transaction, taking it to the standard screen 122. This caused the Unexpected Next Screen error.

3.5. Defect Suppression

As you use Testimony over time, you can “teach” it to suppress certain failures that are either expected or unavoidable, given the way that Testimony works (data-related defects for example) or the set-up of your environment (e.g., bot-related defects). By suppressing these failures, you can ensure that defects are not raised for them, meaning that the effort required for defect analysis will decrease and become more focussed over time.

There are two types of failure suppression in Testimony: step-level suppression, and script-level suppression. These are discussed below.

3.5.1. Step-level suppression

In some cases it may be possible to suppress a failure at the step level and carry on with the rest of the script. When you switch on step-level suppression for a particular failure, Testimony will mark the step as successful and attempt to continue with the rest of the script.

There are two steps involved in deciding whether or not a failure is suitable for step-level suppression. Firstly, of course, you need to have determined that this is a failure that should be ignored in subsequent Testimony playbacks. The examples given above should help you to determine this, based on some common types of defects.

The second step is to determine whether or not it would be possible for the script to continue, given the defect that has been raised.

3.5.1.1. Example of a step-level suppression

If we take another look at [one of the defects we saw earlier](#) relating to a file download failure, we can see that we had a message in the recording (“Download 191 KB N:\06. E”) and a different message (“Setting was applied”) in the playback.

Expected Parameters	Value	Actual Parameters	Value
<ul style="list-style-type: none"> FBL5N - Select Spreadsheet <ul style="list-style-type: none"> Message <ul style="list-style-type: none"> Next Program Next Screen Nr. Next Screen title 	Download 191 KB N:\06. E SAPLSLVC_FULLSCREEN 0500 Customer Line Item Display	<ul style="list-style-type: none"> FBL5N <ul style="list-style-type: none"> Message <ul style="list-style-type: none"> Next Program Next Screen Nr. Next Screen title 	Setting was applied SAPLSLVC_FULLSCREEN 0500 Customer Line Item Display

Note here that the Next Program and Next Screen Nr. are identical in the expected and actual parameters. This is important, as if we are on a different screen in the playback then it will not be possible to continue with the rest of the script.

We now need to take a look at the function that would have been executed next in the script. This can be done by selecting the step after the one that failed.

Dialog transaction (9/576) Step 8 (FBL5N/Customer Line Item Display)			
Test Script Steps			
Step	Object	Object Description	Res
1	FBL5N	Customer Line Item Display	✓
2	FBL5N	Customer Line Item Display	✓
3	FBL5N	Customer Line Item Display	✓
4	FBL5N	Customer Line Item Display	✓
5	FBL5N	Customer Line Item Display	✓
6	FBL5N	Customer Line Item Display	✓
7	FBL5N	Select Spreadsheet	✓
8	FBL5N	Customer Line Item Display	✗

Input Parameters	Value	Expected Parameters	Value
<ul style="list-style-type: none"> FBL5N - Customer Line Item Display <ul style="list-style-type: none"> Main program Main screen number Function Current window width Current window height Control Properties 	SAPLSLVC_FULLSCREEN 0500 Back (= &F03) 0237 0038	<ul style="list-style-type: none"> FBL5N - Customer Line Item Display <ul style="list-style-type: none"> Message <ul style="list-style-type: none"> Next Program Next Screen Nr. Next Screen title 	RFITEMAR 1000 Customer Line Item Display

We can see here that the user pressed the back button (Function Back (= &F03)). So in this case, since we have an identical screen on the step that failed, and the function being executed in the next step is a standard function, we can deduce that if the failed step (step 8) was suppressed by Testimony, then the rest of the script would be able to continue. This failure is therefore suitable for step-level suppression.

3.5.1.2. Creating a step-level suppression

You can create a step-level suppression from within the Investigate Screen by selecting the failed step and then clicking on the Suppress button.

Prev
 Next
 Layout

Test Script Steps

Step	Object	Object Description	Res
<u>1</u>	<u>FBL5N</u>	<u>Customer Line Item Display</u>	
<u>2</u>	<u>FBL5N</u>	<u>Customer Line Item Display</u>	
<u>3</u>	<u>FBL5N</u>	<u>Customer Line Item Display</u>	
<u>4</u>	<u>FBL5N</u>	<u>Customer Line Item Display</u>	
<u>5</u>	<u>FBL5N</u>	<u>Customer Line Item Display</u>	
<u>6</u>	<u>FBL5N</u>	<u>Customer Line Item Display</u>	
<u>7</u>	<u>FBL5N</u>	<u>Select Spreadsheet</u>	
<u>8</u>	<u>FBL5N</u>	<u>Customer Line Item Display</u>	

Suppress
 Play
 Export

Test Script Header

Exec. Queue ID	000009
Exec. Queue Itm ID	0000000576
Test Script ID	0000351880
Exec. Queue Type	Standard scripts
Test Script Type	Dialog transaction
Object Name	FBL5N
User	CAIRT0
Status	Partially complete
Result	Failed
Failure Reason	Unexpected message

The following screen is then displayed.

Create Suppression

Suppression Key

Filter Set

Testimony Comparison

Object Type

Dialog transaction

Object Name

FBL5N

Failure Reason

Unexpected Message

Message Type

S Success

Message Class

SALV_BS_MSG

Message Number

811

Suppression Type

☐ Step

Passes the Current Step and attempts to run the Subsequent Steps

☒ Script

Fails the Current Step and Suppresses the entire Script

Suppression Result

Step Status

Step Result

Failed

Script Status

Suppressed

Script Result

02

Create

Cancel

The Suppression Key contains information on the failure to be suppressed, including the object name, error type and the message to be suppressed.

Firstly, select your custom filter set from the first drop-down. By default, the option to create a script-level suppression is selected. You will therefore need to select the Step suppression type.



Don't use the default Testimony Comparison filter set, as this will be overwritten when you come to upgrade Testimony.

Your screen should now look like this.

Create Suppression

Suppression Key

Filter Set

1

▼

Object Type

Dialog transaction

▼

Object Name

FBL5N

Failure Reason

Unexpected Message

▼

Message Type

S Success

Message Class

SALV_BS_MSG

Message Number

811

Suppression Type

☒ Step

Passes the Current Step and attempts to run the Subsequent Steps

☐ Script

Fails the Current Step and Suppresses the entire Script

Suppression Result

Step Status

Suppressed

▼

Step Result

Failed

▼

Script Status

▼

Script Result

02

▼

Create

Cancel

Click on Create, and the suppression will be created. In future playbacks, if Testimony encounters this same error again it will mark the step as passed as attempt to execute the rest of the script.

3.5.2. Script-level suppression

Where there is an error you want to suppress, but it is not suitable for step-level suppression, then a script-level suppression is possible.

With a script-level suppression, the failed step is still marked as failed, and the script is terminated, but no defect is raised. It is also possible to determine how this is reported in the overall playback statistics.

3.5.2.1. Example of a script-level suppression

In the below defect, we can see that we have a [data-related defect](#): we received a “No stocks exist...” message on the selection screen.

Expected Parameters	Value	Actual Parameters	Value
▼ ZML_WHREPORT - T		▼ ZML_WHREPORT	
> Message		> Message	No stocks exist for specified data
▪ Next Program	SAPLSLVC_FULLSCREEN	▪ Next Program	ZR_ML_WAREHOUSE_INVENTORY_REP
▪ Next Screen Nr.	0500	▪ Next Screen Nr.	1000
▪ Next Screen title	Warehouse Stock	▪ Next Screen title	Warehouse Stock

In this case, notice that the screens are different in the expected and actual parameters. In the recording, the user was taken to a list of stock, whereas in the playback the bot remained on the selection screen. This failure, therefore, is not suitable for a step-level suppression.

However, since we know that this is a data-related defect caused by either sequencing or the failure of a previous transaction, we want to suppress the creation of a defect, which we can do at the script level.

3.5.2.2. Creating a script-level suppression

As before, we can create a script-level suppression by clicking on the Suppress button on the Investigate Screen. We again get a pop-up to enter the suppression details, and this time we select our custom filter set and keep the Script suppression type radio button selected.

Create Suppression

Suppression Key

Filter Set

Object Type

Dialog transaction

Object Name

ZML_WHREPORT

Failure Reason

Unexpected Message

Message Type

S Success

Message Class

ZML_MSG

Message Number

022

Suppression Type

☐ Step

Passes the Current Step and attempts to run the Subsequent Steps

☒ Script

Fails the Current Step and Suppresses the entire Script

Suppression Result

Step Status

Step Result

Failed

Script Status

Suppressed

Script Result

No result

Create



Cancel

In the Script Result drop-down, there are two possibilities for determining what the final status of the script should be.

- No result: this excludes this script from the calculation of passed or failed scripts. Instead, it is added

to the count of “Suppressed, No Result” scripts that can you can see in the Execution Queue status:

Execution Queue Playback Overview

Type	Queue type	Sts	Status	Tot	Run	Pass	Fail	Suppr. No res..	Error	Canc	Not Run
	Standard queue		Complete	<u>16314</u>	<u>16304</u>	<u>12074</u>	<u>1427</u>	<u>346</u>	<u>1059</u>	<u>1353</u>	<u>8</u>

- Passed: this passes the script, adding it to the total of passed scripts for the playback.

In general, it is best to select No Result, as this gives a more accurate representation of the status of the playback. We only want genuinely passed scripts to marked as such.