How-To Guide
SAP Active Global Support
Document Version: 20.0 as of October 2014

SAP Custom Code Management
Usage & Procedure Logging
Implementation Guide for Managers, Architects and Developers
Disclaimer

SAP assumes no responsibility for errors or omissions in this development guideline. SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained herein. This document is provided without warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. Where this document contains information about intended SAP strategies, developments and/or functionalities of SAP products, such information is not intended to be binding upon SAP to any particular course of business, product strategy and/or development.

Any software coding or code lines / strings („Code“) provided are only examples and are not intended for use in a productive system environment. The Code is only intended to better explain and visualize the syntax and phrasing rules for certain SAP coding. SAP does not warrant the correctness or completeness of the Code provided herein and SAP shall not be liable for errors or damages cause by use of the Code, except where such damages were caused by SAP with intent or with gross negligence.

Please note that this document is subject to change and may be changed by SAP at any time without notice.
# Table of Contents

1. **SAP Usage & Procedure Logging** .................................................................................................................. 5

2. **Differentiator and Relations to other solutions** ............................................................................................... 5
   - 2.1 Workload Statistics (transaction ST03N) .................................................................................................... 5
   - 2.2 User Interface Logging for SAP GUI for Windows ......................................................................................... 5
   - 2.3 ABAP Usage Log ............................................................................................................................................... 6
   - 2.4 SQL Monitor ...................................................................................................................................................... 6
   - 2.5 Overview .......................................................................................................................................................... 6
   - 2.6 Missing functionality ........................................................................................................................................... 6

3. **What is UPL?** ...................................................................................................................................................... 7

4. **Possible Use Cases supported and solved with UPL** ......................................................................................... 7
   - 4.1 Requirement phase ............................................................................................................................................... 8
   - 4.2 Design phase ....................................................................................................................................................... 8
   - 4.3 Build & Test phase ............................................................................................................................................... 8
   - 4.4 Transport ............................................................................................................................................................. 9
   - 4.5 Maintain ............................................................................................................................................................. 9
   - 4.6 Optimize ............................................................................................................................................................ 10

5. **UPL - How it works** .......................................................................................................................................... 11
   - 5.1 What is the difference to ST03N workload statistic? ...................................................................................... 12
   - 5.2 UPL preconditions ............................................................................................................................................. 12
   - 5.3 Notes & Bug fixes .............................................................................................................................................. 13
       - 5.3.1 Optional notes ........................................................................................................................................ 13
   - 5.4 How to activate the UPL tracking .................................................................................................................... 13
       - 5.4.1 Prework ................................................................................................................................................ 14
       - 5.4.2 Needed authorization ............................................................................................................................ 14
       - 5.4.3 Activation of UPL ................................................................................................................................ 14
   - 5.5 How to read the UPL data in the managed system? ....................................................................................... 17
   - 5.6 How to access data in the Solution Manager System? .................................................................................. 18
       - 5.6.1 BW Access .............................................................................................................................................. 18
       - 5.6.2 Access preconfigured UPL BW queries ................................................................................................. 18
       - 5.6.3 Define BW Queries on UPL data ........................................................................................................... 19
       - 5.6.4 Preconfigured function modules: .......................................................................................................... 19
   - 5.7 Integrations and tools using UPL data ............................................................................................................ 19
       - 5.7.1 Solution Documentation Assistant .................................................................................................. 19
       - 5.7.2 Custom Code Lifecycle Management .................................................................................................. 19
       - 5.7.3 Custom Development Management Cockpit (CDMC) ....................................................................... 20
       - 5.7.4 Business Process Change Analyzer .................................................................................................. 20
       - 5.7.5 System Recommendation .................................................................................................................. 21
       - 5.7.6 Custom code Apps - Clonefinder ......................................................................................................... 21
       - 5.7.7 SAP Worklist Tool (SWLT) .................................................................................................................. 21
       - 5.7.8 Custom Code HANA effort estimator (precessor of SWLT) ................................................................. 21
       - 5.7.9 Scope & Effort Analyzer (SEA) ........................................................................................................... 21
       - 5.7.10 SAP Code Inspector (SCI) .................................................................................................................. 21
   - 5.8 How to evaluate UPL results ........................................................................................................................... 21
       - 5.8.1 How to find out which database tables where displayed with TCODE SE16 .................................. 23
       - 5.8.2 How to find executed function modules and some security risks ...................................................... 23
       - 5.8.3 Find all of used programs ...................................................................................................................... 24
       - 5.8.4 Evaluation of procedure usage - enhanced evaluation ...................................................................... 25
       - 5.8.5 Find used cloned programs .................................................................................................................. 26
       - 5.8.6 Find all used function modules ........................................................................................................... 27

© 2015 SAP SE or an SAP affiliate company. All rights reserved.
1 SAP Usage & Procedure Logging.

It is always a challenge for every SAP system owner to know what is really going on in their installed systems. What kind of code procedures are executed and how often and if there is a relation between the time frame of execution and the overall number of executions. Besides existing technologies to track and log runtime executions there is always the issue of additional performance losses because the logging technology needs additional resources. And the level of details might be different and will never fit to the requirements. What we are looking for is a technology without system performance impact, with a high level of accuracy and the capability to track also at runtime dynamically generated and executed code language elements. SAP Usage & Procedure Logging (UPL) will give you all these capabilities directly build in your existing SAP solution without installation of additional software packages of difficult activation processes.

2 Differentiator and Relations to other solutions

In this section, we first provide an overview of the previously existing tools which provide insight regarding what is used in the system.

2.1 Workload Statistics (transaction ST03N)

UPL is not a replacement of the workload statistics (ST03N) but can be seen as an additional data source.

- Includes also runtime and DB time informations
- Tracks only spent dialog steps instead of real executions
- Supports only transaction codes, programs and RFC function modules
- Is not able to cover dynamically called routines or SUBMIT programs

2.2 User Interface Logging for SAP GUI for Windows

SAP Custom Development is offering a custom solution to track and log SAP GUI for Windows screens, input, values and user access and security audits

- Full transparency that is using which SAP GUI for Windows screens and all UI elements
- Seamless and impact free implementation
- Minimal impact on system resources
- not part of SAP standard - additional license necessary

SAP GUI for Windows on SAP NetWeaver 7.00, 7.01, 7.02 is supported
2.3 ABAP Usage Log

New capability of SAP Netweaver to determine the dependency and usage of ABAP related objects. It can be seen as a potential successor of the ABAP where-used capabilities of the SAP ABAP workbench with a more precise analysis capability. Available in releases SAP_BASIS 7.02 SP12, 7.31 SP5 and 7.40.

- Takes advantages of the ABAP package concept
- Supports all ABAP object types including DDIC elements
- Integrated with the ABAP workbench
- Dynamic usages/references are not supported
- No usage logging information about real-time executions

2.4 SQL Monitor

New capability of SAP Netweaver to track usage of executed SQL statements at runtime.

- No manual trace ST05 has to be started
- Runtime and record logging of SQL statement
- Transparency about used database tables
- Dynamically called ABAP routines with SQL statement will be catch
- Call stack information of initial caller will be logged
- Focus on SQL usage only

Release SAP_BASIS 7.02 SP14 and following releases

Further information

2.5 Overview

<table>
<thead>
<tr>
<th>Workload Statistics</th>
<th>UI Logging</th>
<th>ABAP Usage Log</th>
<th>SQL Monitor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Includes Runtime Info</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Etc.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2.6 Missing functionality

What is the goal with UPL? Which gaps to close?
3 What is UPL?

Usage and Procedure Logging (UPL) is a new functionality available in any ABAP based system based on the core functionality of SAP Coverage Analyzer. For details regarding release info, refer to section 1.4.3 UPL preconditions. It will be used to log all called and executed ABAP units (procedures) like programs, function modules down to classes, methods and subroutines. On top you can evaluate the usage of Smartform, Adobe print form and SAPscripts. This new enhanced SAP Netweaver capability will have no performance impact on your system and will catch usage information of ABAP routines directly when they happen. UPL will give you 100% coverage of usage without estimations or evaluation of ABAP call stacks. This includes also the detection of dynamically called ABAP elements. UPL is the one and only technology to close the existing gaps in the SAP workload statistic. With the secured access to the UPL data, your usage information is protected against 3rd party eyes. The full reporting capabilities with enriched information in BW of the Solution Manager will give you the flexibility to analyze ABAP usage on your demands.

Benefits:

- No measurable performance impact neither CPU nor runtime
- Supports all ABAP based systems with the technical preconditions
- Catch also any dynamically called ABAP routines
- Supports all ABAP procedure types down to subroutines

Limitations:

- No user related information - Performance and work council reasons
- No runtime information about spent processing time - performance reasons and not in focus
- No call stack information - planned to be implemented in an upcoming release
- No logging for used DDIC like tables, structures, etc
- No logging of transaction codes
- No logging of ABAP development environment language elements for simplified coding (Includes, module pools,...)

4 Possible Use Cases supported and solved with UPL

The following use cases are structured with the SAP Application Lifecycle Management (ALM) phases.
4.1 Requirement phase

Tracking of successful custom code projects - Focus on value creating custom code

Custom developments are an investment into the future to adopt customer systems by implementing business requirements, closing functional gaps or even better implement coding to gain a competitive advantage. New created custom code will generate only business value as soon as it is really and often used. Based on fact custom code is not always an executable program but in most cases minor improvements like modifications, function modules or methods, the classic measurement techniques will not help. UPL will give you the possibility to log the usage on a daily basis and give you the needed information if your custom development was worth to be developed. An evaluation of the real usage after the custom code was requested will give you a possibility to distinct between good or bad investments and will help you to focus in the upcoming development cycles of custom code on the right requirements. UPL key figures can be used to define KPIs to measure the success of a development project and a KPI of efficient the change request was driven by business. This use case is supported by SAP Custom Code Lifecycle Management (CCLM).

Tracking of unsuccessful custom code projects - Validation of the expected business case

Unsuccessful custom code projects are not used, but will increase the maintenance costs. All custom code objects have to be maintained, changed and upgraded. Of course as long as unused custom code objects are not used, the risk having issues is low. But unused custom code will not have a direct impact on your TCO but the indirect impact must be paid as soon as the overall number of custom code objects will be taken into account. UPL will help you to avoid spending development efforts for business requirements from departments with a low usage adaption. IT department focus on their customers and have to define priorities. UPL will help you to put the historical usage facts on the table and will influence the decision process for a new custom code development. Learn from the past; evaluate former business cases based on real usage. This use case is supported by SAP Custom Code Lifecycle Management (CCLM).

4.2 Design phase

No use case available at the moment

4.3 Build & Test phase

Efficient and effective test coverage - Ensure everything was tested before going to production

After the creation of custom code, it is necessary to do technical tests and functional tests of the business processes issued by end-users from the business departments. Typically Test case descriptions are
available and testers use these test cases to test the impacted business processes. Many customers do not have an easy and efficient way to determine the test scope. UPL will help you to identify the test scope by showing how often developed custom code objects were executed during the testing. The main objective of this use case is defining thresholds for every custom code objects when a test was done or not. UPL can also be used in one of the preparation steps of BPCA (Business Process Change Analyzer) which help customers to further reduce the test scope.

Test case quality - Validate the reality of test cases

Effective testing requires that the existing test cases are covering all business scenarios which truly represent the reality of business process usage in the productive environment. Reverse engineering of productive environments with help of SAPs Reverse Business Process Documentation (RBPD) will give you an overview what is going on in your system. Business process usage in combination with UPL data will give you the possibility to detect usage of business processes or process variants. The business blueprint which is created based on such a reverse engineering processes can then be the basis on which tests can be assigned or organized-- independent whether they are manual or automated tests. Thus UPL can help you to validate coverage of test cases with daily, weekly, monthly or yearly aggregated UPL data so that the test cases are covering all processes executed in the productive environment.

Test case requirements - Where are the gaps in my testing plans - ensure business continuity

You can do an ABC analysis of the UPL data to get a list of most used objects (A segment). You can then use BPCA to see which business processes are using these most used objects and ensure these business processes are covered with test cases. This will help you secure your business continuity as the most used objects are well covered by the test cases. For doing such an impact analysis you can use the new “object list analysis” functionality of BPCA which was introduced with SAP Solution Manager 7.1 SP05.

4.4 Transport

No use case available at the moment

4.5 Maintain

Efficient Upgrades - Minimized effort for adaptations and tests

Within an upgrade project you have to take care about impacted custom code objects. This will need up to 50% of the project costs by adapting and testing custom code to ensure these objects will work after upgrade or support package implementation. SAP is offering CDMC tool to solve this issue and part of this evaluation is a usage analysis based on workload statistic. UPL is now able to improve the usage result of custom code and supports the calculation of the impact. UPL will help you to minimize the overall number of code objects to be adapted and finally tested.
Support of Audits, Security-Audits

Create security usage transparency of high risk programs. Some SAP programs, but also some custom code programs should be executed only on specific dates. With current available techniques the usage of ABAP procedures will be traced on a very high level. Sometimes special subroutines have a high influence on security like execution of printing forms or special execution of support applications. UPL will help you to evaluate a pre-defined blacklist of procedures and will avoid any manipulations of the logging data. UPL data are not persisted in the productive environment. The encrypted data container will be transferred to the SAP Solution Manager and is available in a read-only mode only. UPL architecture avoids manipulations of usage date and displays what's really going on.

4.6 Optimize

Improve system quality

Custom Code clearing strategy! Get rid of unused custom code and follow a lean system strategy. UPL will help you to get an overview about used custom code objects combined with the usage distribution. These indicators will give you hints where are very often executed and used custom code objects to be optimized with performance services; where hotspots can be defined for detailed runtime analysis. Rarely used custom code program in best case aggregated and summarized on a monthly or yearly view are real candidates to be removed from the system. If your custom code object is not part of the UPL data for a long period of time then remove it immediately from your system. Avoid maintenance, adoption and test efforts for unused custom code objects.

Improve code quality

UPL and Code Inspector are now integrated. With the latest version of /SDF/SHOW_UPL you will be able to export an object set of ABAP code objects of the productive system. In the corresponding development system you can import this file with help of an object collector "Import from file". The reason to make this 2 step approach was to ensure there is no need to have a RFC connection between production and development.

Improve services for Custom Code Management

UPL is a central usage data source used in SAP Custom Code Lifecycle Management. Without any additional effort, as soon as UPL base functionality is available in managed system, CCLM will take over the usage information and automatically enrich the usage on daily basis.

Solution transitions for Landscape Optimizations - Merge/Acquisitions & Consolidations

Within a split solution landscape with either different main applications or a country specific template structure of productive environments there is always the wish to make a system consolidation. This "single instance" approach needs a detailed analysis which system will be the leading system in the future and which parts of the retired systems must be taken over. This included besides the pure database content (master & transactional data) also the element of custom code. In most cases we can
assume that custom code will fulfill the same business purposes in the different systems. With minimal adaptations for country specifics there is a challenge: Which custom code and modifications are used? Which custom code programs are used? Where are similar custom code objects with different business behavior? Should be follow a greenfield or brownfield approach?

For the overall decision we recommend a SAP Solution Transition Evaluation Service. This service helps you to understand the different landscape options including a rating about benefits, efforts and costs. If the decision was made, we recommend using SAP UPL technology to get full transparency about the usage of custom code. The SAP Clonefinder can also be used to support the code merge process. To analyze custom code naming duplicates a name check is needed. SAP CCLM is doing this as an integrated option in your Solution Manager.

Before you start a conversion project, you should activate the data collection for at least 2 weeks to cover all important main core business processes. All additional monthly, quarterly or yearly end closing reports should be handled in parallel and should be already documented.

Solution Transition - System Split

Company merge is also an important use case UPL can support the analysis phase. Before a merge or split will be done, the corresponding custom code must be evaluated to ensure the IP will not be violated. UPL is the basis to create transparency about which custom code elements are in use, what are the logical dependencies and which elements needs to replicated to the new systems. The can lead also to code clones of custom code in the future.

5 UPL - How it works

The UPL data will be collected every day and saved in an own encrypted data persistency layer on the managed system. Within a predefined period a housekeeping job will purge the outdated UPL data after 14(default) days. If activated from a Solution Manager system, the data will be extracted and collected in an UPL cube on the BW of the Solution Manager for an unlimited history and aggregated weeks and months to support different queries.
5.1 What is the difference to ST03N workload statistic?

The UPL result will be more accurate, because it is able to detect also submitted programs or dynamic calls, can detect the usage of any ABAP based unit down to subroutines and will give you the transparency about used modifications, user exits, classes and also single method executions. Furthermore the execution of function modules or customer function exits will be the basis for a Business Function Prediction to leverage SAP latest innovations.

5.2 UPL preconditions

Managed or monitored system (System to be logged)

The main logic is located in the SAP_BASIS, ST-PI 2008_1_x software component layer and in the SAP kernel and needs at least the following SP level. UPL will become standard with SAP Netweaver 7.01 SP10 or

SAP Netweaver 7.02 SP9 + Kernel 720 Patch 94 or

SAP Netweaver 7.31 SP3 or 7.40

Recommended kernel patch level: 720 Patch level >430; 721 >120 or any higher

Not supported, outdated kernels, please upgrade as soon as possible: SAP NW 7.10, 7.11; 7.38

Please follow the detail in note 1828848

Also the software component ST-PI 2008_1_x SP4 is needed (SP7 is recommended)

Optional: Note 1683134

Solution Manager Release 7.1 with support package >=SP9 + ST-BCO 7xx

Recommended notes:
For improvement of usability and integration into the Custom Code Apps, we recommend to implement note 1808220.
5.3 Notes & Bug fixes

It is recommended to implement the latest available Kernel patch level and update the SAP_BASIS Netweaver stack on the latest support package stack.

Important notes

1955847 UPL: Activation Procedure and Authorization Handling in SAP Solution Manager (SP5 - SP11)

UPL Application logic improvements:

1808220. Bug fix for EFWK extraction in SP7 and minor UI changes

⚠️ The following notes will fix in the corresponding Kernel and SAP_BASIS components issues in performance. If the release will not fit to your release these fixes are not needed to be implemented.

1803420 valid only for SAP_BASIS 7.31 SP6 and 7.40 SP 1.
1721204 valid for SAP_BASIS 7.01.
1786251 valid for SAP_BASIS 7.01 in combination with kernel 720 AKK (important for counting)
1833798 valid for SAP_BASIS 7.01 in combination with kernel 720 AKK. Will fix issue in note 1721204!
1825837 all releases SAP BASIS (very important for performance improvement and data accuracy)
1833884 SCOV: Missing Coverage due to initial COVREF date
1845058 all releases in ST-PI
1862605 Kernel patch. Will improve the count accuracy and server processing with same hostname
1872630 ABAP change if note 1862605 is implemented. Will fix an issue with kernel 7.20 432.
1753038 UPL data collection: CX_SCV_LITE_TRUNCATE_FAILED

5.3.1 Optional notes

1779464 Optimization ins Error message handling
1789354 UI redesign
1822227 Minor bug fix for managed system UI
1857820 Improvement of analyzing tools
1895870 Activation of UPL extractor
1875325 Enhanced version of the /SDF/SHOW_UPL report for the managed system.

5.4 How to activate the UPL tracking

UPL is running on the managed system and Solution Manager is triggering the collection with the activation of the extractor framework job for UPL. All settings will be made automatically in the managed system, as long as the preconditions are ready.

UPL is collecting data from starting with the date and time of activation. Completed time slices will be saved in a local database and stay untouched. Stopping and restarting UPL at any time will lead to a reset of the counters for the current day of collection.
5.4.1 Prework

- Check if software component ST-BCO is installed on your Solution Manager System. Ensure Solution Manager Setup (SOLMAN_SETUP) was successfully completed for your managed system (Activity MANAGED SYSTEM SETUP) to be observed
- Assign the authorization to the defined users in the “Managed System Setup”

5.4.2 Needed authorization

In the managed system or system to be monitored.

- It is necessary to add these authorizations to the remote user coming from the Solution Manager system but on the managed system. This is an add-on to the Read profile or creates a customer specific profile for this purpose.

Ensure the user activating the UPL (defined in SOLMAN “Managed system setup” has at least the authorizations in the adapted profile, please add the following two authorization objects 'S_ADMI_FCD' ID 'S_ADMI_FCD' FIELD 'ST0R' 'S_COV_ADM' ID 'ACTVT' FIELD '02'

- In Solution Manager 7.1 to access to BW queries
  Authority object S_RS_COMP1 with ACTVT with values 02, 16
  Authority object S_RS_COMP1 with RSZCOMPID with value 0SM_CCL*
  Authority object S_RS_COMP1 with RSZCOMPTP with value REP*
  Authority object S_RS_COMP1 with RSZOWNER with value *

5.4.3 Activation of UPL

⚠ The activation of UPL has changed with Solution Manager 7.1 SP10.

5.4.3.1 Activation on Solution Manager 7.1 with SP > SP09:

The UPL activation is part of the configuration work center for Custom Code Management. Please follow the steps and read the embedded documentation on the top of each step:

Start the Transaction code SOLMAN_SETUP and select "Custom Code Management" on the left side.
5.4.3.1.1 Check the Extractor Framework

Start again the Tcode SOLMAN_SETUP and select "Custom Code Management" on the left side. Scope the system and follow the guided procedure until you reach step 5.1. Put a flag for UPL. If this field is not ready for input, then your managed system is not able to be UPL tracked. After activation of the checkbox 'UPL' you can validate the settings in the Solution Manager Extractor Framework (see link on the right side of the table).

5.4.3.2 Activation on Solution Manager 7.1 with SP <= SP09:

5.4.3.2.1 Activate the BW content (ST-BCO 7xx)

To ensure UPL is able to collect historical data from connected managed systems, it is necessary UPL data can be stored in BW cubes. Furthermore some Solution Manager applications will access UPL data via queries. To ensure this function, please activate the following BW queries.

The following queries have to be activated.

0SM_CCL_UPL_CUSTOB
0SM_CCL_UPL_CUSTOBJ
0SM_CCL_UPL_CUSTOBJ_AGGR
0SM_CCL_UPL_OBJM
0SM_CCL_UPL_MONTH
0SM_CCL_UPL

Please perform steps mentioned below.

Step 1 activate cube

1. Start transaction code 'RSA1' in the Solution Manager System
2. Click on the button 'BI Content' on the left side
3. Navigate to SAP Solution Manager in 'Infoprovers by Info Area' (in the middle of the screen)
4. Navigate (open the sub tree) to 'SAP Solution Manager - Custom Code Lifecycle Management'
5. Click (on the upper right screen) on button "Grouping and select "in Dataflow before and afterwards"

6. Drag and drop the green node "SAP Solution Manager - Custom Code Lifecycle Management" to the right side and confirm the next screens.

7. Wait

8. Select on the right side the entry and click on "Install" with sub selection "Install"

9. Wait

10. Check the status - then the cube is ready to be filled

**Step 2 activate queries**

1. Start transaction code RSA1 (BW Setup)
2. Click on "BI content"
3. Click on "Object Types"
4. Navigate to "Query Elements" - "Query" - "Select Objects" Double Click
5. Navigate to "SAP Solution Manager - Custom Code Lifecycle Management"
6. Navigate to "UPL information"
7. Press F7 in the window to see the technical names, if they are not displayed
8. Select all queries listed above, by clicking on "Transfer"
9. Flag on the right side all entries and click on "Install" with subselection "Install"
10. Wait
11. Check the status - then the query is activated
12. The queries can be executed and checked via the transaction "RSRT".

### 5.4.3.2.2 Activate the Extractor framework

Please look also into the [troubleshooting](#) section

1. Start transaction "Solman _Workcenter"
2. Select workcenter "Solution Manager Administration"
3. Select the view "Infrastructure"
4. Select the application "Extractor Framework"
5. Filter the name by "UPL"
6. Select entry "Custom Code UPL Extractor"
7. Change to edit mode
8. Activate the data collection

There are no further activities needed on the managed system, because it will trigger all needed activities in the managed system.

⚠️ **Remark:** Important to know is that the Extractor Framework is using the first working RFC connection pointing to the managed system. This could be also client 000 or any other non-productive client, if available. In this client, the job will be executed. So please ensure that the user in this client has sufficient authorizations like described below or ensure you are using the correct client in the RFC destination.

⚠️ **Remark:** Upgrading to SP8, it is recommended to redo the Solman Setup and restart the EFWK customizing for the UPL extractors. This will introduce a new postdata loader feature.

### 5.4.3.3 Directly in the managed system without SAP Solution Manager 7.1 (not preferred!!!, only for test purpose)

This is only needed if there is no Solution Manager 7.1 system available.
In this case implement note **1683134** and execute the program `/SDF/UPL_CONTROL`.

With this program you can start and stop the recording of UPL data. The default setting of days to be collected is 14 days, but can be increased or decreased.

### 5.5 How to read the UPL data in the managed system?

#### RFC function modules

Use the RFC function module to read UPL data for defined days: `/SDF/UPL_GET_USAGE`

UPL is available in managed system you can use: `/SDF/UPL_SYSTEM_STATUS`

RFC for quick check of UPL availability: `/SDF/UPL_CHECK`

#### Direct online view

Use the report `/SDF/SHOW_UPL`

This report is a quick ALV viewer to decrypt the UPL data. This includes viewing of existing time slices and also the current UPL collection in progress. In worst case the executed elements can be accessed after 45 min. In most cases the usage information is instantly available. The UPL output format is very easy to read.

#### Output format

*Date*: Date of the corresponding UPL slice. All list entries with the same UPL date were executed at this date.

*Time*: Unused at the moment. Will give you information at which time the UPL data was written.

*Program ID*: Internal technical identifier for ABAP programming units

*Object Type*: Describes the type of objects. PROG for programs frames, FUGR for Function groups,...

*Object Name in Object Directory*: Name of the surrounding ABAP repository object

*Original System of Object*: Not filled. Available in SAP Solution Manager BW cube

*Person Responsible for a Repository Object*: Not filled. Available in SAP Solution Manager BW cube

*Package*: Not filled. Available in SAP Solution Manager BW cube

*Tcode/Program*: Name of the ABAP Include containing the ABAP procedure

*Type*: Type of ABAP processing block. This is the most important information. You are able to distinct between executions of function modules, class methods, selection screens, program events, PAI, PBO, subroutines, user exits,...

*Name of Processing Block*: The real name of the ABAP procedure. This contains also the names of any subroutine of a called ABAP program.

*Class of Processing Block*: Name of the class, if a method execution is involved

*Accumulated Executions (Absolute)*: Number of real executions in the corresponding UPL time slices

*Natural number*: Help counter to support sum creation in the ALV output. Can be ignored in API access case.

#### Example evaluation of UPL data

Filtering the "Objectname" with Z* we will find several entries for the same program.
In this example the customer program with name ZUNLOCK_USER contains a main program entry and 3 procedures or events within the same program. You can determine how often a specific event within this program was executed. In this case the program itself has 2 events like START-OF-SELECTION and END-OF-SELECTION executed 10 times. And the subroutine FORM GET_LOCATION was executed 10 times. So it is also possible to detect if a program was started and left before the real execution with key F8 or if there was executed. Furthermore the given subroutine was not part of a Boolean condition and was executed also 10 times.

Remark: With Netweaver release 7.31 the entry with Type PROG will disappear. The only valid indicator will be type SSEL with START-OF-SELECTION:00 or :01.

Another example analyzed the usage of a BADI execution. In this example the constructor of the BADI class was called 6 times, we can assume from 6 different tasks, but the request for the BADI instance was executed more than 11,000 times. This is an example how this BADI was used in the productive environment.

5.6 How to access data in the Solution Manager System?

5.6.1 BW Access

Here we have defined a cube called 0SM_CCL_UPL with the info provider 0SM_UPL. You will be able to define any BW query on top of this cube data.

5.6.2 Access preconfigured UPL BW queries

1. Start your SAP Solution Manager Easy Access Menu (TCode S000)
2. Right mouse click on 'Favorites' and select 'Add additional objects'
3. Select 'SAP BW Query URL'
4. Enter any text description
5. As object description use
   1. `<prt_protcl>://<prt_server>/<bi_launcher>?QUERY=
   2. Followed by one of the following query names

0SM_CCL_UPL_CUSTOBJ_AGGR  CCL Customer Objects UPL aggregated
0SM_CCL_UPL_MONTH         UPL monthly Report
0SM_CCL_UPL_CUSTOBJ       CCL Custom Objects UPL
0SM_CCL_UPL_OBJM          UPL Data for Object
### 5.6.3 Define BW Queries on UPL data

Precondition: Install the BEx Explorer of your SAP GUI.

1. Start the tool “BEx Query Designer”
2. Connect to your Solution Manager system with valid logon credentials
3. Use the Icon "New Query" to create a new query
4. Insert in the file NAME the technical infoprovider 0SM_UPL and press the button OK
5. On the next screen you can design your query with Drag & Drop
6. It is possible to define filters and the row/columns for the output (take care of the selected tab)
7. “Save As” your query under your favorites. After leaving the BEx Query designer the query will appear in your menu under favorites in the normal GUI.

### 5.6.4 Preconfigured function modules:

- AGS_CC_EXTRACT_UPL_COMMON_MDX Extract data from query 0SM_CCL_UPL
- AGS_CC_EXTRACT_UPL_DAY_MDX Extract daily customer objects by MDX
- AGS_CC_EXTRACT_UPL_MONTH_MDX Extract monthly aggregated execution by MDX
- AGS_CC_EXTRACT_UPL_MONTH_SDA Extract monthly aggregated execution by MDX for Solution Documentation Assistant
- AGS_CC_UPL_EXTRACT_DATES (you get an overview of the available days of data for UPL) UPL data of a specific system is available in SOLMAN
- AGS_CC_UPL_TIMERANGE_BPCA UPL data of a specific system from SOLMAN you use

### 5.7 Integrations and tools using UPL data

#### 5.7.1 Solution Documentation Assistant

Within the Solution Documentation Assistant in Solution Manager 7.1 (SP3) a new check step type was introduced to access UPL usage data.

⚠️ Remark: The tool Solution Documentation Assistant (SDA) is only able to evaluate UPL data in check item, if there are at least data for one completed month available. The BW will aggregate the daily data slices to a monthly aggregate.

#### 5.7.2 Custom Code Lifecycle Management

The CCLM tool is supporting in addition to statistical workload data on monthly and weekly aggregation the daily data collection on UPL. This can be activated in CCLM by configuring and activation of the UPL collector job RAGS_CC_GET_UPL_DAY
5.7.3 Custom Development Management Cockpit (CDMC)

With Solution Manager 7.1 SP9, the tool CDMC supports the usage of UPL data by accessing the Solman BW cube infosource. Similar to the jobs collecting workload data, table statistics etc., and a new program was developed to support the data extraction. Supported types and detail in note 1316694

5.7.4 Business Process Change Analyzer

Business Process Change Analyzer requires the user to create used technical object lists for the business processes called TBOMs. There are broadly two types of TBOMs - Static and Dynamic. Dynamic TBOMs are created when the user runs through the business process on an ERP system and a trace is enabled to collect the used SAP objects. Creating dynamic TBOMs can be time consuming as the user has to go through the business processes. Usage of automated scripts will increase the speed of creation of the TBOMs. The BPCA tool in future can make use of the UPL data sources to improve the automation of the technical bill of material (TBOM) creation process. Using the UPL BPCA will be able to create a new “semi-dynamic” TBOM automatically for all business transactions of the customer. Here BPCA will first look at the source code of a business transaction and start collecting all the technical objects, BPCA will then filter the object list for only the used call stack elements as collected in the UPL data.

Watch the demo https://service.sap.com/~sapidb/011000358700000043912014E.EXE

• Overview of Business Process Change Analyzer
• How to guide of Business Process Change Analyzer
5.7.5 System Recommendation

Usage information about the objects which are part of the correction note, security note or hotnews to validate the necessity of implementation. The main idea is to find out if some notes are really relevant for implementation. It also provides an impact calculation how a Support package will influence a running system including the daily impact. Access on UPL data via BW query. Detailed information in SAP Service Marketplace.

5.7.6 Custom code Apps - Clonefinder

Clonefinder is able to access the UPL data in the managed system for the last 7 days. This will give you information how often found clones are used in productive environments. UPL data will be displayed within the clonefinder result list by adding additional columns in the output. These layout fields are only available if UPL is activated in managed system.

5.7.7 SAP Worklist Tool (SWLT)

Tool to analyze the HANA potential including the HANA relevant code fixes. UPL data will be used to determine the worklist priority. The tool will be released with NW 7.02 SP14.

5.7.8 Custom Code HANA effort estimator (precessor of SWLT)

Tool to combine HANA potential calculation in combination of workload statistics, code inspector checks, UPL usage to calculate an adaption effort before migration to a HANA scenario.

5.7.9 Scope & Effort Analyzer (SEA)

Use UPL usage information to determine the test scope and efforts for code adaptions in upgrades or enhancement package implementations.

Further information at Overview of Scope and Effort Analyzer.

5.7.10 SAP Code Inspector (SCI)

Usage date collected on managed system can be extracted as file using the report /SDF/DAT,SHOW_UPL and can be uploaded via Object Collector of the Codeinspector.

5.8 How to evaluate UPL results

The following evaluations are based on the results in the ALV of the report /SDF/DAT,SHOW_UPL.

Please remark there exists a new version of the report shipped via note 1875325.
It offers you new features like

- Display the lowest possible date where UPL data are available in managed system
- Including fast filters for types, names, procedure types, procedure names, author and packages
- Aggregation capability of result on man object level
- Aggregation capability of result for different days
- Conversion of Runtime to Design Time for evaluation of usage of WebDynpro, Webservices, Business Server Pages BSP, SE16, SE37, Enqueues, Dequeues, BW elements decryption supported with an API plugin.
- Possibility to save result as txt or xls file
- Possibility for additional file compression with zip
- Download feature to export usage data in ABAP Code Inspector format. Can be imported as object set in SCI.
- Forward navigation to ABAP workbench by hotspot activities in result ALV list.

Display Usage & Procedure Logging Data

<table>
<thead>
<tr>
<th>Selection of UPL data time frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start date</td>
</tr>
<tr>
<td>End date</td>
</tr>
</tbody>
</table>

Filter options

- Object Type
- Object Name
- Processing Block Type
- Processing Block
- Package
- Author

Parameters

- Aggregation on time level
- Aggregation on main object level
- Runtime -> Design Time conversion
- Reset ALV counter in output

Display & download options

- ALV display
- Download to file
- Download for Code Inspector

Save as ZIP

© 2015 SAP SE or an SAP affiliate company. All rights reserved.
5.8.1 How to find out which database tables where displayed with TCODE SE16

Analyze the column ‘Object Name’ and search for the pattern /1BCDWB/DB.*. Remove the prefix /1BCDWB/DB and the remaining string is the name of the access database table. The value behind the subtype INIT and ESEL will give you the information if the SE16 was just started (INIT) or really executed (ESEL).

5.8.2 How to find executed function modules and some security risks

An interesting use case is the detection of used function modules in a productive environment. For this analysis filter the column TYPE by value FUNC and you will get all executed functionn modules. This includes also the Database enqueue and dequeue function, the customer exits and BTE enhancement executions.

If you look in the workload statistics, you can also see how often the transaction SE37 was executed. But what kind of function modules where executed in SE37? In this transaction you are also able to make a test run of the function module. This is a critical situation because you are able to change business data by execution of specific function modules. To get a detailed list of all tested function modules please a filter ‘*=FT’ on column Object name so see how often a test environment was started and finally executed.
The entries with the subroutines FORM GET_VARIABLES will give you an information how often the function was executed. The number of FORM PARAMETERS_GET dived by the overall number of executions will give you the information how many parameters the interface of the function module are defined and finally the number of executions behind FORM FUNCTION_CALL will tell you if the function module was really executed or not.

5.8.3 Find all of used programs

The benefit of UPL is possibility to detect also dynamically called programs, programs called with a SUBMIT ABAP statement or any other techniques not supported by ST03 workload statistic. Often we receive the question if it is possible to find out if a report was started only or really executed. Or if specific report events like INITIALIZATION, AT-SELECTION_SCREEN, etc were started and how often. UPL will give you the answer.

Please follow this procedure:

• Start the /SDF/SHOW_UPL for the wished date range
• Filter column Objecttype with the value Prog
• Filter the column Objectname with the program to be evaluated
• Evaluate the column Typ and detect which events were executed how often

In the given example we can see an starting and also an execution of the report was done 9 times.

To evaluate the execution of a report please focus on one of the following events depending on the code implementation of the report. We recommend focusing on any SSEL event type because it indicates the expected usage of the report.

<table>
<thead>
<tr>
<th>TYPE</th>
<th>Processing Block</th>
<th>Description</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSEL</td>
<td>START-OF-SELECTION:00</td>
<td>Generated event START-OF-SELECTION</td>
<td>should be used for the executable report execution if F8 was executed</td>
</tr>
<tr>
<td>SSEL</td>
<td>START-OF-SELECTION:01</td>
<td>Explicit START-OF-SELECTION event in report</td>
<td></td>
</tr>
<tr>
<td>INIT</td>
<td>INITIALIZATION:00</td>
<td>Explicit INITIALIZATION event in report</td>
<td></td>
</tr>
<tr>
<td>ESEL</td>
<td>END-OF-SELECTION:00</td>
<td>Generated event END-OF-SELECTION</td>
<td></td>
</tr>
<tr>
<td>ESEL</td>
<td>END-OF-SELECTION:01</td>
<td>Explicit END-OF-SELECTION event in report</td>
<td></td>
</tr>
</tbody>
</table>
Please remark that the field content in column TYPE with value PROG will be available in SAP_BASIS 7.01 only. With release 7.02 and higher you have to consider the processing blocks.

This will give you also a hint if end users found this report and had fear to really execute it. They just looked at the starting screen and closed it again. Maybe the number of given UI parameters was too high and too complex. This could be an indicator for additional "training demand".

### 5.8.4 Evaluation of procedure usage - enhanced evaluation

Executable programs and their related subroutines can be evaluated very easy with the guide in the previous step. A more complicated situation is the use of multi includes or subroutine pools. Both methods are not recommended to use but ABAP offers this capability and developers use these possibilities. In a first example we focus on multiple includes.

First we develop a classic include with a simple subroutine.

**Include Z_MULTI_INCL**

```abap
REPORT Z_MULTI_INCL.
  INCLUDE Z_MULTI_INCL.
  START-OF-SELECTION.
  perform call_me.
  ENDFORM.
```

Now let’s create 2 different programs using the same include.

```abap
REPORT Z_MULTI_1.
  INCLUDE Z_MULTI_INCL.
  START-OF-SELECTION.
  perform call_me.
```

Copy the report Z_MULTI_1 to Z_MULTI_2 and execute both several times and watch the result in /SDF/SHOW_UPL.
As we can see in the UPL result, there are no entries for the INCLUDE Z_MULTI_INCL. This is clear because UPL is a runtime logging and an include will be merged into the corresponding main program as soon as you activate the code.

The subroutine CALL_ME will have 2 results line. One per main program if INCLUDE. This will give you the possibility to see how often a subroutine was called in the relation to the main program. But we strictly recommend to follow the ABAP Development Guidelines not to use includes in multiple main programs.

The second example will take care about dynamic calls of external subroutines of a subroutine pool.

Start wil developing a main program.

REPORT  Z_CARLOS_2.

data: LV_SUB type char10 value 'CALL_ME'.
PERFORM (LV_SUB) in PROGRAM Z_CARLOS_1.

This represents a dynamic call of the subroutine 'CALL_ME' in the subroutine pool Z_CARLOS_1. ABAP also offers you the possibility to call any subroutine in any available program or report.

From a code perspective it looks similar as the multiple includes, but there is a huge difference in the processing logic. As soon as the external perform statement will be executed, you will leave the program and load the other program into the ABAP kernel. Evaluation of the UPL data is displaying the following.

The Subroutine CALL_ME is still linked to its own frame program. The Report Z_CARLOS_2 doesn't track any call of the subroutine.

⚠️ Remember: UPL is not representing the call stack. It tracks the executions of subroutines in its code environment at the runtime. Includes will be unpacked as soon as you activate the code.

### 5.8.5 Find used cloned programs

SAP Clonefinder 3.0 has now full UPL data support. Displays the usage of found clone over the last 7 workdays and derive a strategy to remove the clone or optimize it. Find more details in your managed system and the documentation of [Clonefinder 3.0](https://www.sap.com)
5.8.6 Find all used function modules

Function modules will not be logged in ST03N. But it is also important to know which function modules and in best case which custom developed function modules are executed and how often it happens. If a function module was RFC enabled and started then it is also logged in the ST03N statistic. But we need more details.

Please follow the procedure:

- Start the /SDF/SHOW_UPL for the wished date range
- Filter column Type with the value FUNC

5.8.7 Find all used customer function exits

Customer enhancements are very important. SAP provides a whole bunch of capabilities to enhance SAP applications with a customer specific flavor. Detection if an implementation of the enhancement concept Customer user Exit is activated is simple to find out by calling the transaction CMOD. But this is just an information to see the customer exit is activated and ready for use. But we are not able to find out how often this specific customer enhancement was executed. Wouldn't it make sense to evaluate the business value of such an implementation by analyzing the usage.

Please follow the procedure:

- Start the /SDF/SHOW_UPL for the wished date range
- Filter column Type with the value FUNC
- Filter the _name of processing block_ with wildcard EXIT_*
- Sort descending the column accumulated executions

Use this information as an indicator to deactivate unused customer function exit implementations and ensure the quality of such really used customer enhancements is very well. Define your test scoping on basis of UPL data.

5.8.7.1 Find unwanted Field Exits.

Coming soon

5.8.7.2 Find all used classes and methods

Coming soon
5.8.7.3 Find modified objects

Coming soon

5.8.7.4 Determine usage of most important Smartforms

Smartforms are generated objects to print any information on printers with help of dynamically determined data from applications.

Over time the number of smartforms is increasing based on business requirements (new fields, more fields) to be printed or by organization changes e.g. changes of postal code, telephone number or logo. There is always the need to find out which smartforms are really in use. Any changes on smartform will increase maintenance efforts and costs. Having a comprehensive overview about the real usage will support lowering the TCO.

Usage cases

- Detect really used smartforms
- Detect important smartforms and generate a kind of ABC analysis and priority list for change projects
- Drive clearing activities and remove unused smartforms "housekeeping"
- Track usage and focus on missusage of outdated smartforms

5.8.7.5 How to detect the usage of Smartforms

- Generate a list of all available smartforms in the system reading table STXFADMI

<table>
<thead>
<tr>
<th>FURNAME</th>
<th>FRNUMB</th>
<th>FRNUMED</th>
<th>INTYERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>BC470_FLOMD_LATE</td>
<td>00000005</td>
<td>00000000</td>
<td>00007</td>
</tr>
<tr>
<td>BC470_FLOMD_UNFINISHED</td>
<td>00000006</td>
<td>00000000</td>
<td>00007</td>
</tr>
<tr>
<td>BC470_FLOMD</td>
<td>00000007</td>
<td>00000000</td>
<td>00007</td>
</tr>
<tr>
<td>MEFMI_MSTUE_001</td>
<td>00000008</td>
<td>00000000</td>
<td>00007</td>
</tr>
<tr>
<td>HR_CMP_TCS</td>
<td>00000001</td>
<td>00000000</td>
<td>00007</td>
</tr>
<tr>
<td>J_7L8B6EBLEG_CEN4</td>
<td>00000009</td>
<td>00000000</td>
<td>00047</td>
</tr>
<tr>
<td>LE_BIL_INVOICE</td>
<td>00000013</td>
<td>00000000</td>
<td>00047</td>
</tr>
<tr>
<td>LE_SHIP_DELNOTE</td>
<td>00000012</td>
<td>00000000</td>
<td>00047</td>
</tr>
<tr>
<td>FRP_MAILTO_BODY</td>
<td>00000011</td>
<td>00000000</td>
<td>00047</td>
</tr>
<tr>
<td>FRP_MAILTO_SUBJECT</td>
<td>00000010</td>
<td>00000000</td>
<td>00047</td>
</tr>
<tr>
<td>TRYXR0_SAP_PAYSLIP</td>
<td>00000003</td>
<td>00000000</td>
<td>00007</td>
</tr>
<tr>
<td>TRYXR0_SAP_PAYSLIP_DE</td>
<td>00000004</td>
<td>00000000</td>
<td>00047</td>
</tr>
<tr>
<td>TCS_ABE</td>
<td>00000002</td>
<td>00000000</td>
<td>00005</td>
</tr>
</tbody>
</table>

- Check the result tables of UPL with report /SDF/SHOW_UPL in the managed system
- Filter the column "Objectname" with the prefix "*/1BCDWB/SF*"
• Split the results into 2 parts e.g. /1BCDWB/SF and 00000012

• Look in your table STXFADM for the column FMNUMB for the number e.g 00000012
As a result you will see which smartform with the name "LE_SHP_DELNOTE" was executed and how often

With note 1875325, this conversion is part of the standard display logic.

5.8.7.6 Other print forms

In newer releases of report /SDF/SHOW_UPL you will be able to analyze the usage of SAPscripts, Adobe Printforms and Smartforms. If you choose the option “Runtime-Designtime” conversion the result list of UPL will be analyzed for potential conversions. A new column “Designtime” will be introduced and will translate back the generated print classes/reports to the origin entry. Please implement note 1875325

5.8.7.7 How to detect used BRF+ Rules?

Check the result tables of UPL with report /SDF/SHOW_UPL in the managed system

  • Set a filter in column Objectname with "/FDT/*"

Example:

In this example take the object name value from the row accessing the processing block "IF_FDT_GEN~PROCESS". This is the central execution routine of BRFplus.
Take the string of the classname "/FDT/DDVO8PZ7VKMTUQLSXYOQX41M0" and look into table FDT_CC_0000 with object name = classname.

The result GUID in column ID will be e.g. E219011881301BF1A00C005056944A28 and this is the one key for the corresponding BRF Application.

With this ID look in table FDT_ADMN_0000 and find the corresponding BRFplus application in column NAME.

Use the transaction codes BRFPLUS or FDT_WORKBENCH to start the administration UI of the BRFplus application.
5.8.7.8 How to determine usage of BW queries, transformation rules etc.? 

All relevant BW elements are generated programs with the prefix ‘GP’ followed by a GUID number. To identify these elements in column 'Object name' these elements are visible. With forward navigation double click on name you will be able to jump directly in the program editor. In the header of the program you will find the relevant information. By implementing note 1847431

You will get a BW specific analysis report with the purpose to convert these generated programs into the real BW elements. In a later release it is planned to combine UPL with this BW analysis.

6 Troubleshooting

Issue: I try to start the program /SDF/UPL_CONTROL, but I get a dump, what is to do?
Solution: Verify that SCOV is deactivated. It is not possible to use SCOV and UPL together at the same time. We recommend using UPL on productive clients only. SCOV itself was designed to run only in a development or quality assurance environment also based on fact the classic SCOV will have a performance impact. To avoid the dump while starting the UPL_CONTROL, please implement note 1822227.

Issue: As soon as I restart the UPL engine, it looks like I loose data?
Solution: This is the standard behavior. To avoid incorrect usage tracking information, as soon as the UPL engine will be stopped with the report /SDF/UPL_CONTROL, the collected data of the current day will be reset to zero. This standard behavior cannot be influenced. Data already collected remain untouched.

Issue: SCOV has still defined some active Testkeys
Solution: Stop the UPL collector by starting report /SDF/UPL_CONTROL. Start the transactioncode SCOV. Switch to EDIT mode. Select the entry "Testgroups". Select all defined "Testgroups" and delete them. Then save changes.

Issue: In extractor framework the "Custom Code UPL Extractor" entry for my system is missing
Solution: Please ensure your "Managed system configuration" was successfully configured with SolMan_Setup for your wished system.
In the guided procedure you have to follow and complete all steps. In step 8 (Configure automatically) you have be sure that the Extractor setup was completed with a green light.
Issue: Tracked accumulated execution figures seem to be too high.
Solution: Please implement note 1785251 valid for Kernel 720 and Downward-compatible kernel 7.0 and 7.01

Issue: There is an error message "Data collection was not performed" in the status monitor of SCOV
Solution: Ensure the settings and servers are correct. See section "How to find data collection is running". If not please use UPL_CONTROL to stop UPL mode. Start SCOV and correct the server settings. Then reactivate the UPL again.

Issue: Periodic extraction of UPL results in sporadic shortdump with error CX_SCV_LITE_TRUNCATE_FAILED
Solution: Please implement note 1753038
Cause of the problem could be that the data collector of the Coverage Analyzer is running periodically and writes to the table COVRES. However there is no chance to confirm this as the 'TRUNCATE' function that is called by ABAP language indicates that the call has failed but does not reveal the reasons. Hence it results in a deadlock situation and subsequent dump. As a workaround, please plan the job not at 00:00:01, but slightly later(say 45 minutes)

Issue: In the Extractor Framework of the Solution Manager EFWK, it is not possible to activate the UPL extractors based on missing authorizations
Solution: Please follow in this case the Solution Manager Security Guide and ask your central Solution Manager administrator (default user name SOLMAN_ADMIN) of assign the powerful authorization role SAP_SOLMAN_ADMIN_COMP.

Issues: While trying to activate the EFWK extractors for UPL for a specific system you will get an error "PPMS Modelling is not valid. Check PPMS keys"
Solution: Please rerun the Managed System Setup activity for the corresponding system to regenerate the correct settings. If this will not help, please open a customer message ticket on component SV-SMG-DIA-SRV-EFW

Issue: Error exception CX_SCV_LITE_NO_TRIGGER_COLLECT while activating UPL data collection
Solution: This error might occur in rare situations if the UPL try to schedule the periodic job RSCVR_TRIGGER_COLLECT on the batch server. If the server was defined as "no batch support" the UPL will not be able to schedule a periodic job. As result the UPL will collect data, but after a while a data overflow will happen. Execute the following actions:

1. Stop the UPL collector in report /sdf/upl_control
2. Start SM37 and delete all scheduled jobs RSCVR_TRIGGER_COLLECT including those to be run in the future.
3. Start transaction SCOV and look under "Settings". Take note of the name of the batch server for triggering data collection.
4. Start RZ03 to get an overview of all servers. For the used collection server there must be a 'B' in the column services.
5. Take note of any server with a 'B' in the column services
6. Start SCOV, go to "Settings", change to edit mode and enter the noted server of step 5; save
7. Start the UPL again.
8. Under "Consistency checks" you can try to check and repair the status and jobs. Follow the guided messages

Issue: In SCOV under monitoring you can see an error message: Failed background job: RSCVR_TRIGGER_COLLECT
Solution: This happens if the maintained batch server under Settings is not existing. In case you make a system copy the issue will happen in some cases. To fix it, please maintain a valid, existing server with free batch processes under "Settings" and save.
Issue: On my file system of server I found a huge file called COVDATA or COVDATA1
Solution: this might happen if the job RSCVR_TRIGGER_COLLECT is not started and shared memory will not be able to flush the data to database. In this case an emergency concept will write the usage data on file system. Please ensure the job is started. Either by deactivating and reactivating UPL or fix settings issue in transaction SCOV. As soon as the rigger collect job is running again, the file content will be processed. If not we recommend to delete the COVDATA1 file and copy the 16 byte file COVDATA to COVDATA1.
Please ask you administrator or use the report RSBDCOS0 to execute an operating system command.

Issue: in SCOV monitoring I receive an error "errno = 28 in error message „Can’t write to data file „"
Solution: This error indicated the files system on harddisc is full and further data can’t be written. In this case please delete the file COVDATA1 and copy the file COVDATA to COVDATA1.

Issue: SDF/UPL_CONTROL Dumps with exception CX_SCV_LITE_NO_TRIGGER_COLLECT in managed system or raises the error text "Could not schedule data collector"
Solution: In transaction SCOV under monitor you will find an error message "Job: RSCVR_TRIGGER_COLLECT # : Cannot Define Submit Step (3)".
This indicates the needed background job to transfer the UPL data from shared memory to database layer cannot be initiated. The reason is a missing basis authorization of Job scheduling AUTHORITY-CHECK OBJECT 'S_PROGRAM' ID 'P_GROUP' FIELD <usergroup> ID 'P_ACTION' FIELD 'BTCSUBMIT'. This happens if the user starting the UPL has no authorization to submit a program in a background batch job.

7 Frequently Asked Questions (FAQs)

Q: How to find out if UPL collection is collecting data?

A: Start transaction code SCOV. If the UPL is activated, you will see a status information "SCOV lite is activated!" The report /SDF/UPL-CONTROL will check this automatically and will give you the needed information.

Furthermore the traffic light under "Data collection" should be green. In this case everything is fine. If the traffic light is red or yellow, something went wrong in the initial UPL activation. Select in the navigation tree on the left side the entry "On/off Status" and find out which server will have an impact.
Selection the line entry you can STOP and RESTART the collection. If it is not possible to switch the status to green there is an internal server error. In most cases a server restart might help. Or contact your SAP support team for help.

Another possibility is the server in a row does not exist anymore. You can validate the name of servers by executing transaction SE51. If the server is not listed here, you can select the server in SCOV and remove it from the collection list by pressing the trashcan icon.

Further ensure under "Settings" the server under "Background job for triggering collection" has a valid and existing server of your landscape.

Q: Will UPL have any impact on the system performance?
A: No, there is no measurable impact, because we count the usage as soon as the ABAP compiler is loading the code. This is confirmed by the SAP benchmark team.

Q: Are there any risks to activate UPL?
A: No, there is no known risk to activate UPL.

Q: Are there any possibilities to see user related information in UPL date?
A: No this is not possible. There are 2 reasons. First one is the dramatic increase of data volume generated. Every user will multiply the entries of passed ABAP units. At the end there will be a huge datavolume difficult to analyze. Second the local work councils will have a legal issue with such data. SAP is not allowed to offer such a build in feature with user related information.

Q: How much data will be consumed in the managed system?
A: We collect usage data on a daily basis. As soon as one ABAP program was executed, we increase only the execution counter. From our experience the needed DB space is between 2-10 MB for 14 days of data. But this really depends on the real usage of different programs.

Q: What is the trace level of UPL?
A: UPL is logging the execution of ABAP units and procedures. There is no additional detail level planned to be added. Every additional element like transaction, call stack or user information will boost the size of persistency. We recommend to focus in a first step on the current capabilities and enjoy the available use cases. UPL is one level deeper than the common ST03N workload statistics.

Q: Is UPL able to log also access to database tables?
A: This is not planned at the moment. This would mix different approaches into UPL. UPL is a high level trace but more accurate than ST03N with focus on non-performance impact. Changing the UPL logic to log more information like access on database tables via SQL statements, call stacks and more will have a real performance impact. You can expect a performance loss of more than 35% (if branch coverage is activated) and a dramatic data volume increase.

Q: How much data will be collected in the managed system?
A: This depends on the number of days in the retention time. Example 552260 Entries in UPL Log for 19 days needs around ~28 MB of database allocation. Per day one code objects will create only one entry in this logging table. So it really depends on the how your system was used. As a thumb rule per day you need for ~30000 different code logging ~1,5 MB of disk space. The default setting is limited to 14 days. The database space per entries is around 55 bytes per UPL entry.

Q: How much data will saved in the BW of the Solution Manager?
A: Depends when you start archiving the corresponding infocube behind. You can find details on the sizing calculation in SAP note 2159129.

We recommend using the Solution Manager Data Volume Management tool to observe the UPL amounts.
Q: Is there an archiving strategy for UPL cube data?
A: We recommend archiving unneeded UPL data after a retention period of 730 days on a daily basis. UPL data for more than 2 years will not improve any analysis on usage. We recommend designing your BW on UPL cube to create monthly or year aggregates. Monthly aggregates can be archived after 14 months. Yearly aggregates created after 5 or 10 years.

Q: How to make an emergency off of UPL data collection if I feel unwell?
A: Start report /SDF/UPL_CONTROL and click on the STOP button. Please remark if you stop and restart the collected data will be reset to zero.

Q: I execute a program, start the report /SDF/SHOW_UPL, but I can’t see the program?
A: The logging will not be persisted in real-time. It can take up to 45 min until a demo executed program will be visible in the output.

Q: How can I change the UPL retention time of the housekeeping of UPL data?
A: With the report /SDF/UPL_CONTROL you can set a new value up to 99 days. Please be careful about data volume. The right place for historical data is the Solution Manager 7.1 BW cube.

Q: What is the difference between UPL and SCOV?
A: UPL is an optimized and light version of the SCOV reducing the logging capabilities to an absolute minimum to ensure full performance.

Q: How to check the setting of the EFWK in database?
A: Select table E2E_ACTIVE_WLI with Extractor = *UPL* and Active = 'X'. Get a list of all activated extractor. Changing the ACTIVE flag will deactivate extraction.

Q: How can I evaluate the UPL status of any managed system via RFC function call?
A: Please implement note
and implement a status evaluation report. This API will give you all needed information about UPL, the retention periods, current available data and status of jobs.

Q: It is possible to track procedures of INCLUDES?
A: UPL is logging the usage of any procedures of INCLUDES, but will assign it to the runtime executable ABAP object. This will give you the benefit to see which caller executed the procedure of the INCLUDE and how often. If 2 executable reports uses the same INCLUDE and call the same subroutine, the usage of the subroutine will be assigned and split between the 2 executable reports.

INCLUDES like TOP Includes containing in most cases only data declarations will not be logged by UPL. In the activation process of an ABAP program all INCLUDES will be unpacked and are not visible anymore in the runtime environment of ABAP.

Q: Analyzing the UPL data from my regular executed report doesn't look to be accurate?
A: UPL is a continuous tracking of the executions for an unchanged ABAP procedure. If you schedule a job with a program on a regular basis, the potential differences might be possible if the execution was done around midnight. But the average execution over several days should fit.

If the execution counter is really different, please check the kernel release and implement the needed notes. Doing UPL tests in a development system there might be the issue of "wrong" execution counters. The reason is rather simple. UPL is tracking the accumulated executions of an unchanged source code. As soon as the ABAP workbench is changing the source code, the execution counters of the corresponding procedure block will be reset to zero, but only for this procedure with a change. All other counters of remain unchanged. There is also a difference between kernel 7.01 and kernel 7.10. In older releases the counter will not be reset also in case of a change. In 7.10 and higher the counter will be reset, if there is a load-relevant change in the source code. Netweaver is evaluation this situation and will provide a correction if necessary.