Tech Mahindra

LCaaS

(Legacy Code analysis as Service)

User Guide

Contents

1. LCaaS Overview

LCaaS is a Legacy Code Analysis Tool developed by TechM's Enterprise Architecture team which incorporates the industry best practices needed for reverse engineering and aims at reducing the manual efforts & costs drastically. It helps automate end-to-end legacy code analysis on Mainframe and AS400 through various introspective/intuitive reports.

LCaaS will be extremely valuable for your customers who intend to modernize their legacy applications in AS400 to Java or .Net or SAP or any technology of choice. Even for those customers who do not intend to modernize, LCaaS will be valuable in providing in-depth documentation of the legacy applications.

2. LCaaS Reports

LCaaS generates lot of intuitive reports and charts, flow diagrams that help us identifying Size, Complexity, I/O intensity, Dependency and Stability of every components as part of legacy modernization and optimization.

These reports will be helpful in doing technical and functional documentation, technical debt analysis and rule extraction that is carried out as part of reverse engineering projects.

Also the data that are collected from these reports would be helpful to come up with modernization estimation and planning and to devise optimal sequence of modernization as part of transformation roadmap creation.

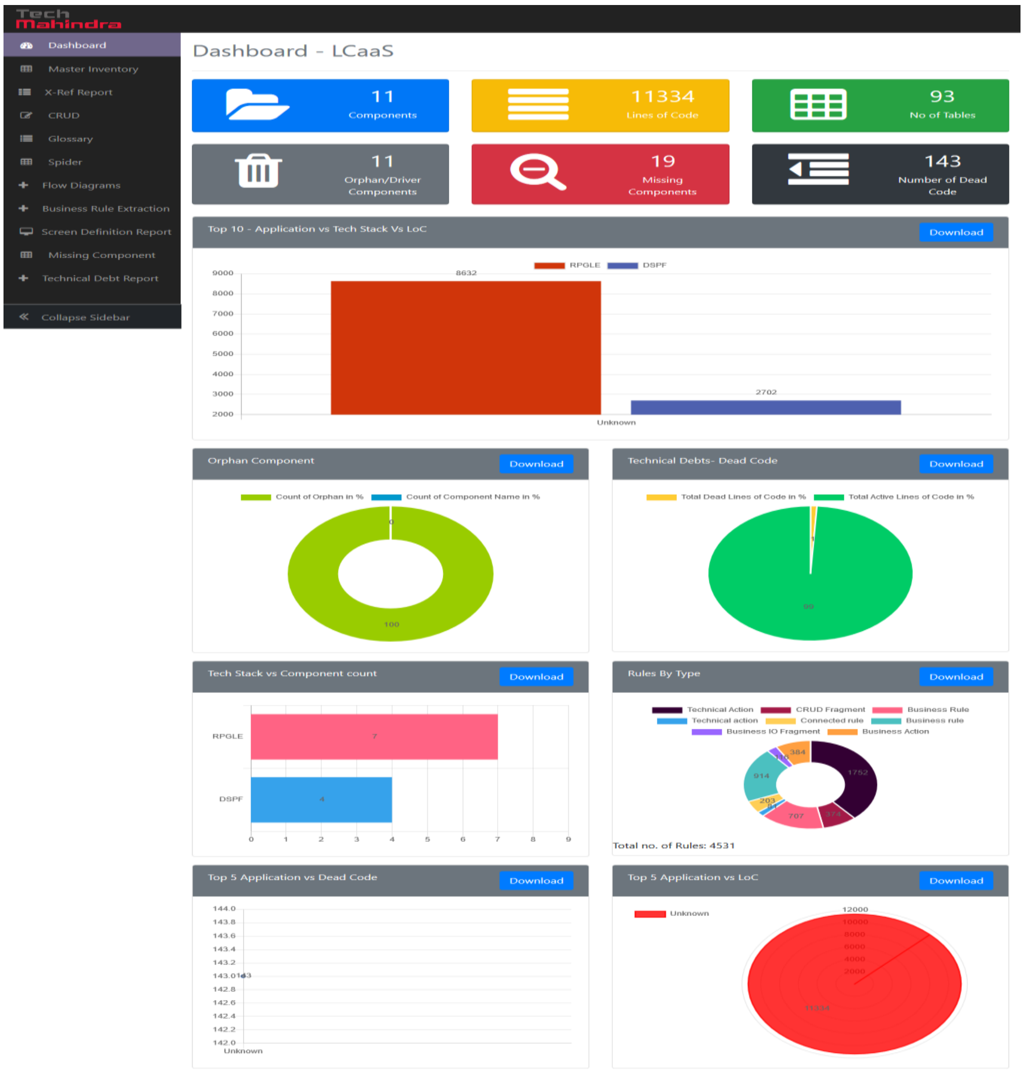
The various reports/Charts that can be generated by LCaaS are as follows:

1. Dash Board
2. Master Inventory Report
3. Cross reference Report (XREF)
4. CRUD Report
5. Missing Report
6. Orphan/Dead Report
7. Dead para Report
8. Spider Diagram
9. Process Flow Diagram
10. Detailed Flow chart
11. Glossary Report
12. BRE (Business Rule Extraction) Report
    a. Detailed Report
    b. Rules Report
13. Screen Definition Report

2.1 LCaaS Dashboard

LCaaS dashboard will facilitate all the statistical data of the application code, that is in scope for your analysis. It shows the total number of components, total size of the application, in terms of Lines of code, number of DB2 tables, number of orphan components, number of missing components, and number of dead lines that resides in the application code base.

LCaaS also generates intuitive graphs that can help us understand, Top 10 Application vs Tech Stack Vs Lines of code, the percentage of Orphan components in the overall inventory, the percentage of dead code, in the overall lines of code. Also it generates graphical views on Top 5 Application vs Dead Code, Top 5 Application vs Lines of code, the spread of Business Rules on each application. All these data, and the stats in the dashboard, would really help us understand, the size, complexity, Technical debts in the application, and a snap shot of rules spread, at a high level.



2.2 Master Inventory

Master Listing of all components identified within the Legacy system and loaded in code repository and also gives LoC, Commented lines, SLoC, Cyclomatic Complexity, dead lines and dead paras

S.No	Column Name	Description
1	Component_name	Source component name
2	Component_type	Source component type e.g. RPGLE,SQLRPGLE,DSPF ,etc.
3	Application_Name	Application name of the component
4	Loc	Lines of code
5	Commented_lines	Number of a commented lines.
6	Blank_lines	Number of blank/empty lines.
7	Sloc	Number of executable source lines of code. (excluding commented lines and blank lines)
8	Cyclomatic_complexity	Cyclomatic complexity is a software metric used to measure the complexity of a program. These metric, measures independent paths through program source code. This metric was developed by Thomas J. McCabe and it is based on a control flow representation of the program
9	No_of_dead_lines	Number of lines of code, which are not reachable/not executed in the control of a program. The source lines of code that are contributing to the dead paragraphs, are accumulated to come up with no of dead lines.
10	Dead_para_count	Number of para's which are not referenced/used in the program in the control flow path
11	Total_para_count	Total number of paragraphs for each component.
12	Path	Source file location.

2.3 Cross reference Report

One to one relationships between "Calling" and "Called" components that are loaded in repository. Helps in identifying the program level integration touch points.

S.No	Column Name	Description
1.	Component_name	Calling component Name
2.	Component_type	Calling component Type e.g. RPGLE, SQLRPGLE, DSPF, etc...
3.	Calling_app_name	Calling component's application name
4.	Called_name	Called component Name
5.	Called_type	Called component Type e.g. RPGLE, SQLRPGLE, DSPF, etc....
6.	Called_app_name	Called component's application name
7.	Access_mode	File access mode (Read/Write) in Jobs (Applicable for Shells scripts)
8.	Comments	Additional information For Cobol Xref, comments will have the directory details from where the Cobol is executed For Sort step, comments will have the actual sort card used in job script For FTP/Email steps, comments will have details on ftp

S.No	Column Name	Description
		directory /email commands/email attachment details coded in job script

2.4 CRUD Report

List of one to one relationships between Program and DB2 tables and its usage (Create, Report, Update, Delete). Helps in identifying the Database level integration touch points

S.No	Column Name	Description
1	Component_name	Source component name
2	Component_type	Calling component Type e.g. RPGLE, SQLRPGLE, DSPF, etc…
3	Application_Name	Source file application name.
4	Table	Table name that is accessed by program
5	CRUD	Table operation .e.g. Create, Read, Update, Delete.
6	SQL	Embedded SQL Query used in the program to retrieve data from DB, which is used to understand the purpose of the DB operation, columns impacted and the criteria involved

2.5 Missing component report

List of components that are referred by the components loaded in the LCaaS repository, but have no source code in the LCaaS repository.

S.No	Column Name	Description
1	Component_name	Source component name
2	Component_type	Calling component Type e.g. RPGLE, SQLRPGLE, DSPF, etc…

2.6 Orphan Report

List of components that are not called/referenced by any other components loaded in LCaaS repository

S.No	Column Name	Description
1	Component_name	Source component name
2	Component_type	Calling component Type e.g. RPGLE, SQLRPGLE, DSPF, etc…

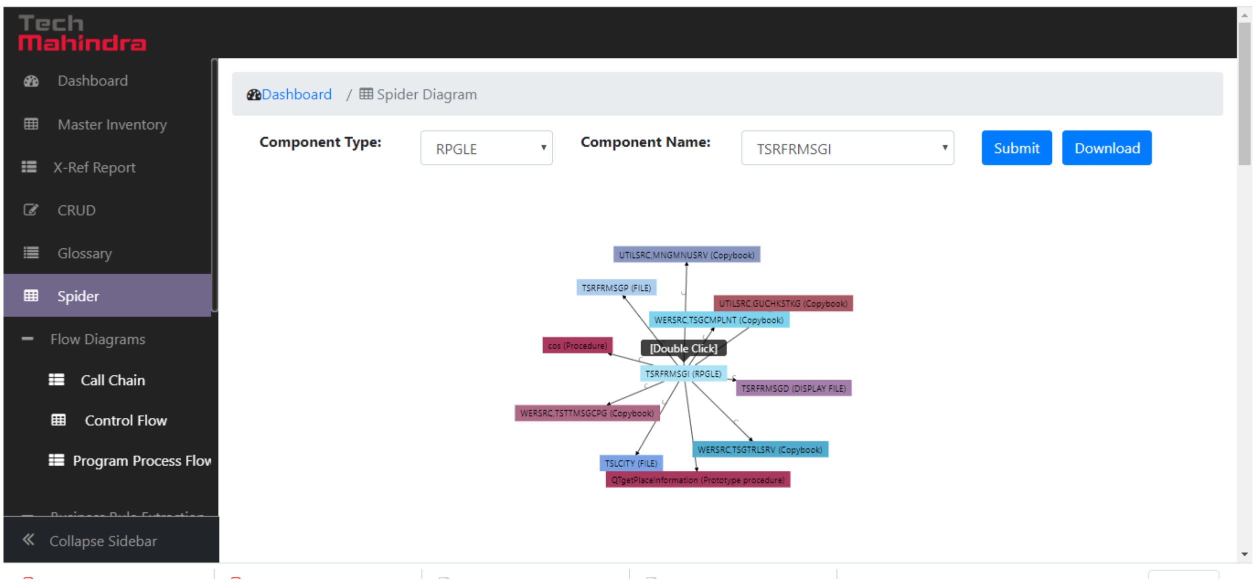
2.7 Dead para report:

List of paragraphs that could be potentially dead, unreachable or unused in nature

S.No	Column Name	Description
1	component_name	Source component name
2	component_type	Calling component Type e.g. RPGLE, SQLRPGLE, DSPF, etc…
3	no_of_dead_lines	Number of lines of code which are not reachable/not executed in the control of a program
4	dead_para_count	Number of para's which are not referenced/used in the program in the control flow path
5	dead_para_list	List of paragraphs that are identified as dead by LCaaS under each Cobol program.

2.8 Spider Flow Chart

Spider chart gives the dependency diagram for a selected component, which gives the Calling and called components relationship in one click and also helps to drill down to any further level from the chart by a double click on any specific component of any type in the chart. Thus, this chart helps in carrying out impact analysis of any component and helps to trace both backward (Calling) and forward (Called) impacts.



Component Type: RPGLE

Component Name: TSRFRMSGI

2.9 Program Process Flow Diagram

Program process flow diagram gives the complete control flow at functions level within a RPG program.

Note: Function refers to Sub-routine in case of AS400 RPG program.

2.10 Program Flow Chart

Program Flow Chart gives a detailed flow diagram of a selected function (sub-routine) within Process flow diagram

Note: Function refers to Sub-routine in case of AS400 RPG program.

2.11 Glossary Report

Glossary report, captures all the variables, defined in the AS400 application programs. This feature in LCaaS, enables the user to map the business meaning of the variables, which will be propagated and annotated in other LCaaS reports, such as the flow charts and BRE- Rule report. This way, LCaaS makes the reports and flow diagrams more readable, annotated with business meanings.

S.No	Column Name	Description
1	component name	Calling component Name
2	Variable	Variable name
3	Business Meaning	User can add Business Meaning's to the variables

2.12 BRE Report

BRE (Business Rules Extraction) report is very crucial report and helps extracting the potential business rules/validations/ technical logics that are coded with in application programs.

Detailed View:

Detailed view of rules will give the line by line rule category tagging and statement grouping and fragement id . The Key features of LCaaS BRE report are as follows:

- Separates the business logic from technical code
- Refactors the code to bring all the logic for a business rule together
- Organizes the business rules into a flow that represents the required sequence of rule execution in the new system.

S.No	Column Name	Description
1	fragment_Id	It organizes the rule into flow that represents the rule sequence to be executed in the target platform.
2	para_name	Name of the para where each fragment id belongs.
3	source_statements	Actual source statement/lines from respective program
4	statement_group	Grouping of each source statements based on the type of operation it performs
5	rule_category	Rule classification of statement grouping.
6	parent_rule_id	Fragment id of the rules that the respective source statement/logic belongs to. Parent rule id the key parameter which helps to bring all the logic for a business rule together

Rules Report:

Rules report gives a view focusing the business rules and the logics that are being actioned under the respective rules

S.No	Column Name	Description
1	pgm_name	Source component name
2	para_name	Name of the sub-routine from which the rule resides
3	source_statements	Actual source statement/lines from respective program
4	rule_description	Rule description with Annotated version on Source Statements. Users can use this place holder to update the business meaning of the rules in LCaaS
5	Rule	Rule sequence number
6	rule_relation	Rule sequence number that helps to connect the parent rule number from where it get branched

2.13 Screen Definition Report

The screen definition report will have the fields defined in the DSPF file of screens the positions and lengths of fields and Literals.

S.No	Column Name	Description
1	COMP_NAME	Source component name
2	MAPSET_NAME	Mapset Name
3	MAP_NAME	Map Name
4	LENGTH	Length of the field
5	TYPE	Type of the field
6	Access_Mode	Field Access Mode(input/output)
7	POS	Position of the field on screen