EcoStruxure Machine Advisor

Code Analysis

User Guide

Original instructions

EIO0000003915.04
12/2022

Legal Information

The Schneider Electric brand and any trademarks of Schneider Electric SE and its subsidiaries referred to in this guide are the property of Schneider Electric SE or its subsidiaries. All other brands may be trademarks of their respective owners.

This guide and its content are protected under applicable copyright laws and furnished for informational use only. No part of this guide may be reproduced or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), for any purpose, without the prior written permission of Schneider Electric.

Schneider Electric does not grant any right or license for commercial use of the guide or its content, except for a non-exclusive and personal license to consult it on an "as is" basis. Schneider Electric products and equipment should be installed, operated, serviced, and maintained only by qualified personnel.

As standards, specifications, and designs change from time to time, information contained in this guide may be subject to change without notice.

To the extent permitted by applicable law, no responsibility or liability is assumed by Schneider Electric and its subsidiaries for any errors or omissions in the informational content of this material or consequences arising out of or resulting from the use of the information contained herein.

Table of Contents

Safety Information 7
About the Book....8
Introduction....11
    General Information on the EcoStruxure Machine Advisor Code Analysis Portal....11
    Components of the EcoStruxure Machine Advisor Code Analysis Portal....12
HOME....13
    HOME....13
PROJECT SNAPSHOTS....15
    OVERVIEW....15
    METRICS DASHBOARD....17
    CONVENTIONS DASHBOARD....18
    DETAILS VIEW....19
    METRICS ARCHIVE & EXPORT....22
    CONVENTIONS ARCHIVE & EXPORT....24
    CODE QUALITY REPORT....25
    ALL MY PROJECTS....26
PROJECT TRENDS....28
    DASHBOARD....28
    METRICS DETAILS VIEW....29
    CONVENTIONS DETAILS VIEW....30
    METRICS ARCHIVE & EXPORT....32
    CONVENTIONS ARCHIVE & EXPORT....34
    ALL MY PROJECTS....36
COMPARE PROJECTS....37
    DASHBOARD....37
    DETAILS VIEW....38
    METRICS ARCHIVE & EXPORT....40
ADMINISTRATION....42
    SNAPSHOTS....42
    PROCESSING JOBS....44
    RULE SET....45
    DEVELOPMENT PHASES....47
    ACCESS TOKENS....48
    ALL MY PROJECTS....50
    ALL QUERY CHAINS....50
    ALL COMPANY PROJECTS....51
    USER AND USER ROLES....53
    PURCHASES....55
    User and Company Settings....55
How to Use the EcoStruxure Machine Advisor Code Analysis....57
    Create a Company Account and Invite Users....57
    Create a New Analysis Project and Assign Users....58
    Create a Snapshot and Upload (with Personal User Account)....59
    Create an Access Token and Upload a Snapshot with Python Script....60
    Connect an Analysis Project to an SVN Repository....61

Appendices 63
Metrics 64
Metric: Application Size (Code) 64
Metric: Application Size (Code+Data) 65
Metric: Application Size (Data) 65
Metric: Call In 66
Metric: Call Out 66
Metric: Commented Variables (All) Ratio 66
Metric: Commented Variables (In+Out+Global) Ratio 67
Metric: Cyclomatic Complexity 67
Metric: Extended By 69
Metric: Extends 69
Metric: Fan In 70
Metric: Fan Out 70
Metric: Halstead Complexity 71
Metric: Implemented By 74
Metric: Implements 75
Metric: Lines Of Code (LOC) 76
Metric: Memory Size (Data) 76
Metric: Number Of Actions 77
Metric: Number Of GVL Usages 78
Metric: Number Of Header Comment Lines 78
Metric: Number Of Instances 79
Metric: Number Of Library References 80
Metric: Number Of Messages 80
Metric: Number Of Methods 80
Metric: Number Of Multiline Comments 81
Metric: Number Of Properties 81
Metric: Number Of Reads 82
Metric: Number Of Tasks 82
Metric: Number Of Transitions 83
Metric: Number Of Variables 83
Metric: Number Of Writes 84
Metric: Number Of FBD Networks 84
Metric: Source Code Comment Ratio 85
Metric: Stack Size 86
Conventions 87
Convention: Access to Global Variable in FB_Init + FB_Exit 87
Convention: Compile Messages 88
Convention: Complex POU With Low Comment Ratio 88
Convention: Complex Type Name Checks 88
Convention: Empty Implementation 89
Convention: Global Variable Accessed Only in One POU 89
Convention: Inheritance Depth Limit 89
Convention: Input Variable Read Check 90
Convention: Input Variable Type Check 91
Convention: Input Variable Write Check 91
Convention: Multiline Comment Usage 92
Convention: No Header Comment 93
Convention: Number of Methods Limit 93
Convention: Number Of Pins Limit (Input/Output) 93

Convention: Number Of Pins Limit (Input) 94
Convention: Number Of Pins Limit (Output) 94
Convention: Number of Properties Limit 94
Convention: Output Variable Read Check 95
Convention: Output Variable Type Check 95
Convention: Persistent Usage Check 96
Convention: Retain Usage Check 96
Convention: Uncommented Variable (All) 97
Convention: Uncommented Variable (In+Out+Global) 97
Convention: Unused Enum Constants Check 97
Convention: Unused Variables Check 98
Convention: Useless DUT 98
Convention: Variable Name Checks 99
Convention: Variable Name Length Check 100
Index 101

Safety Information

Important Information

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

The addition of this symbol to a “Danger” or “Warning” safety label indicates that an electrical hazard exists which will result in personal injury if the instructions are not followed.

This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

⚠ DANGER

DANGER indicates a hazardous situation which, if not avoided, **will result in** death or serious injury.

⚠ WARNING

WARNING indicates a hazardous situation which, if not avoided, **could result in** death or serious injury.

⚠ CAUTION

CAUTION indicates a hazardous situation which, if not avoided, **could result** in minor or moderate injury.

NOTICE

NOTICE is used to address practices not related to physical injury.

Please Note

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

About the Book

Document Scope

This document describes the graphical user interface of the EcoStruxure Machine Advisor Code Analysis and the functions it provides.

Validity Note

This document has been updated for the release of EcoStruxure Machine Advisor Code Analysis V1.4.

Related Documents

Title of documenation	Reference number
EcoStruxure Machine Expert Code Analysis User Guide	EIO0000002710 (ENG); EIO0000004079 (FRE) EIO0000002711 (GER); EIO0000004080 (SPA) EIO0000004081 (ITA)

EcoStruxure Machine Expert Programming Guide	EIO0000002854 (ENG);
	EIO0000002855 (FRE);
	EIO0000002856 (GER);
	EIO0000002858 (SPA);
	EIO0000002857 (ITA);
	EIO0000002859 (CHS)

Product Related Information

⚠ **WARNING**

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.[1]
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

[1] For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction

and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

⚠ **WARNING**

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in this manual, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
IEC 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2015	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2015	Safety of machinery - Emergency stop - Principles for design
IEC 62061:2015	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2016	Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the present document may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a *hazard zone* or *danger zone* in the *Machinery Directive* (*2006/42/EC*) and *ISO 12100:2010*.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

Introduction

General Information on the EcoStruxure Machine Advisor Code Analysis Portal

Overview

The EcoStruxure Machine Advisor Code Analysis is based on the Code Analysis which is integrated into EcoStruxure Machine Expert Logic Builder. This plugin is used to analyze projects focusing on the source code quality. Refer to Code Analysis (see EcoStruxure Machine Expert, Code Analysis, User Guide).

The EcoStruxure Machine Advisor Code Analysis is an internet-based portal that allows:

- To upload snapshots of a project (only the analysis data created with the Code Analysis) to the portal.

  The snapshots that are uploaded to EcoStruxure Machine Advisor Code Analysis do not contain source code. They only contain names of objects, methods, variables, and so on, and information on the relationship between them (call, read, define).
- To get an overview over the source code development trend of your projects by comparing several snapshots.
- To limit user-access to specific functionalities by different user rights, page 54.

EcoStruxure Machine Advisor Code Analysis Portal

The portal is used by project managers, quality officers, and developers:

- To have a closed quality loop available (continuous measurement, visualization, and improvement of source code quality).
- To measure and visualize quality relevant project indicators over time.
- To monitor the history of quality indicator measures in projects.
- To get fast high-level information via dashboards for:
  - Single snapshots, page 15
  - A whole project, page 28
  - A comparison between projects, page 37
- To explore the code down to POU level to help find the root cause of negative development trends.

Supported Browsers

You can use browsers that support HTML5/CSS3.

For example:

- Mozilla Firefox
- Google Chrome

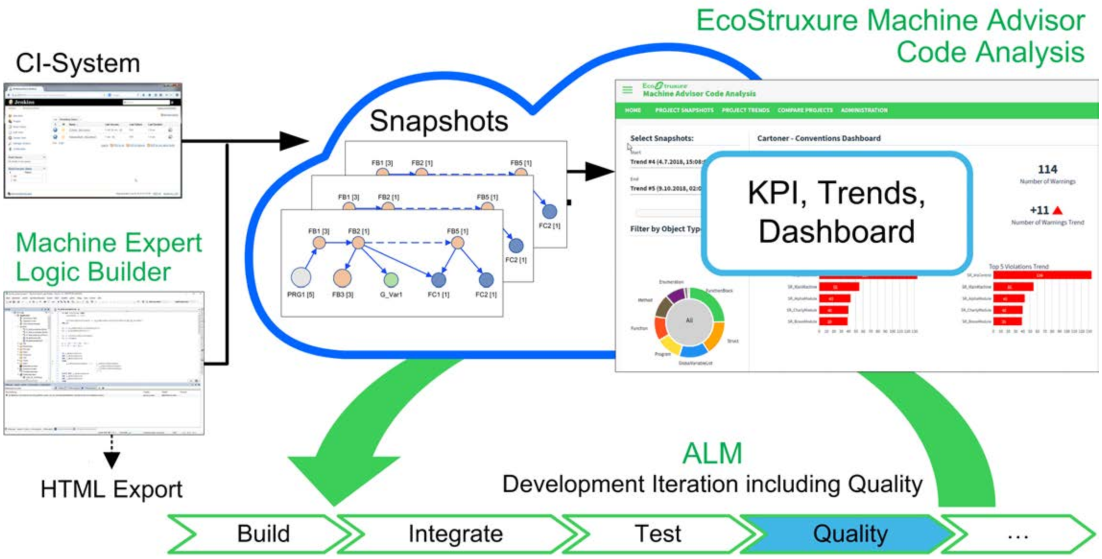
  NOTE: To help ensure that you have up-to-date Cybersecurity protection, use the latest version available for your browser.

Components of the EcoStruxure Machine Advisor Code Analysis Portal

Overview

You can use EcoStruxure Machine Advisor Code Analysis in your ALM (Application Lifecycle Management) to help verify and improve code quality during your development process.

The diagram gives an overview of the components used by the EcoStruxure Machine Advisor Code Analysis portal:



- EcoStruxure Machine Expert Logic Builder

  To analyze your single projects and to create snapshots from these analysis data.

- EcoStruxure Machine Advisor Code Analysis

  To create dashboards and trends to visualize source code development trends represented by KPIs (Key Performance Indicators).

- CI-System (Continuous Integration Systems)

  To use a CI-System as an additional way to upload snapshots, for example via Python scripts to the portal.

HOME

HOME

Overview

Click **HOME** to open this window. This window is also displayed after login to the EcoStruxure Machine Advisor Code Analysis portal.

The **HOME** window provides the following parts:

- **8-Week Trend: Cyclomatic Complexity (Average)**

  Line chart providing the **Cyclomatic Complexity (Average)** metric values for the selected projects over the period of the last eight weeks.

- **Cross Project Overview**

  Based on the projects to which you are assigned, the most relevant figures are displayed.

8-Week Trend: Cyclomatic Complexity (Average)

Element	Description
Selected Analysis Projects	Select the projects you want to display in the **Cross Project Overview** view. By default, up to three projects are selected. You can only select projects you are assigned to.
Line Chart	This line chart provides the **Cyclomatic Complexity (Average)** metric values for the selected projects over the period of the last eight weeks. A legend provides the names of the selected projects and their associated colors in the line chart. The lines displayed in the line chart provide additional information via tooltip.

Cross Project Overview

Based on the projects to which you are assigned, the most relevant figures are displayed.

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

A table provides the following information:

Column	Description
Project Name	Names of the projects you are assigned to. Click an entry in this column to open the corresponding **<Project> - Snapshot Overview** window, page 15.
Activity Status	Activity status of the project: - **High Activity (Steady Uploads)** - **Low Activity (Odd Uploads)** - **No Activity** A **Refresh** icon in this column indicates that snapshots are currently analyzed. Click an entry in this column to open the corresponding **<Project> - Snapshot Overview** window, page 15.
LOCs (Sum)	Value of **Lines Of Code** with aggregation type **Sum** for the latest snapshot. Click an entry in this column to open the corresponding **<Project> - Metrics Dashboard** window, page 17.
Cyclomatic Complexity (Average)	Value of **Cyclomatic Complexity** with aggregation type **Average** for the latest snapshot. The orientation and the color of an arrow beside the value indicate the development trend of the calculated metric: - Down arrow, green: Decreasing number of code violations. - Up arrow, red: Increasing number of code violations. Click an entry in this column to open the corresponding **<Project> - Metrics Dashboard** window, page 17.
Violations (Sum)	Value of the **Violations** with aggregation type **Sum**. The orientation and the color of an arrow beside the value indicate the development trend of the calculated metric: - Down arrow, green: Decreasing number of code violations. - Up arrow, red: Increasing number of code violations. Click an entry in this column to open the corresponding **<Project> - Conventions Dashboard** window, page 18.
Last Snapshot Upload	Date of the latest uploaded snapshot. Click an entry in this column to open the corresponding **<Project> - Snapshot Overview** window, page 15.
Click a column header to sort the table according to the respective column.	

PROJECT SNAPSHOTS

OVERVIEW

Overview

Click **PROJECT SNAPSHOTS > <PROJECT> > OVERVIEW** to open this window.

The **OVERVIEW** window provides the following parts:

- **Select Snapshot**

  Selected Snapshot which data is displayed in the **<Project> - Snapshot Overview** view.

  By default, the latest snapshot is preselected.

- **<Project> - Snapshot Overview**

  Based on the selected snapshot, a snapshot overview is displayed.

Select Snapshot

Element	Description
Selected Snapshot	Select a snapshot from the list. The list provides the name, the version (if available), and the analysis date for the snapshot.

<Project> - Snapshot Overview

Based on the selected snapshot, a snapshot overview is displayed.

Element	Description
Lines of Code	Total number of code lines for all objects in the snapshot.
Maximum Halstead Difficulty	Object with the greatest **Halstead Difficulty** value in the snapshot.
Maximum Cyclomatic Complexity	Object with the greatest **Cyclomatic Complexity** value in the snapshot.
Number of Objects	Number of objects in the snapshot.
Average Halstead Difficulty	Average value of **Halstead Difficulty** of all objects in the snapshot.
Average Cyclomatic Complexity	Average value of **Cyclomatic Complexity** of all objects in the snapshot.
Changelog	Comment which source code modifications were done. This comment is entered at the **ADMINISTRATION > <PROJECT> > SNAPSHOTS** window, page 42 or has been part of the uploaded snapshot.

Element	Description
Notes	Notes are entered at the **ADMINISTRATION > <PROJECT> > SNAPSHOTS** window, page 42.
SVN-CHANGELOG button	Launches the **Tortoise SVN** tool (if installed). This tool provides a list of the modifications made between the revision that is assigned to the selected snapshot and the last snapshot analyzed (**ADMINISTRATION > <PROJECT> > <SNAPSHOT> > Edit Snapshot > Snapshot Details** window, page 42). To see the modifications in detail, install the Logic Builder Diff Viewer (see EcoStruxure Machine Expert Logic Builder Diff Viewer, User Guide). Prerequisites to use the **SVN-CHANGELOG** button: • A **Version Control Base URI** is defined for the project (**ADMINISTRATION > ALL COMPANY PROJECTS > <Analysis Project > Details** window, page 51). • A **Revision in Source Control** is defined for the snapshot (**ADMINISTRATION > <PROJECT> > <SNAPSHOT> > Edit Snapshot > Snapshot Details** window, page 42).

The data for the **Modified Objects**, **New Objects**, and **Removed Objects** result from a comparison with the last snapshot analyzed.

Each section (**Modified Objects**, **New Objects**, **Removed Objects**) provides a filter and navigation toolbar.

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.

A table provides the following information for modified, new, and removed objects.

Column	Description
Type	Object type (such as variable).
Name	Object name.
Path	Folder path that contains the affected object.
Click a column header to sort the table according to the respective column.	

Click a table entry to open the related detail view as displayed in the **PROJECT SNAPSHOTS > <PROJECT> > DETAILS VIEW** window, page 19.

Click the magnifier button, right-hand side of a table entry, to open the Logic Builder Diff Viewer (see EcoStruxure Machine Expert Logic Builder Diff Viewer, User Guide) for the selected object.

METRICS DASHBOARD

Overview

Click **PROJECT SNAPSHOTS > <PROJECT> > METRICS DASHBOARD** to open this window.

The **METRICS DASHBOARD** window provides the following parts:

- **Select Snapshots**

  Period for which data is displayed in the **<Project> - Metrics Dashboard** view.

  By default, the last two snapshots analyzed are selected.

- **Filter by Object Type**

  Allows you to filter the metrics data for specific object types.

- **<Project> - Metrics Dashboard**

  Based on the period (selected snapshots), the applied filter, and the selected metrics the most relevant figures and nodes are displayed.

Select Snapshots

Element	Description
Start	Select a start snapshot from the list. The list provides the name, the version (if available), and the analysis date of the snapshots. The differences between the start snapshot and the end snapshot are displayed on the dashboard.
End	Select an end snapshot from the list.
Slider	Alternatively, you can use a slider to define the start snapshot and end snapshot.

Filter by Object Type

You can filter the **<Project> - Metrics Dashboard** content based on object types. By default no object type check box is activated and data for all object types is displayed. Activate the check box of an object type to limit the displayed data to one object type. You can also activate several check boxes.

<Project> - Metrics Dashboard

Based on the period (selected snapshots), the applied filter, and the selected metrics the most relevant figures and nodes are displayed.

For **Metric 1**, **Lines of Code** is selected by default.

For **Metric 2**, **Halstead Difficulty** is selected by default.

The following table is valid for **Metric 1** and **Metric 2**.

Element	Description
Metric 1	Select a metric from the list. Click the **Show Metric Description** icon beside the **Metric 1** list to display a description of the selected metric.
Sum <Metric 1> / **Max <Metric 1>**	Sum of **Metric 1** for the latest snapshot in the selected period. If **Sum** is not supported as aggregation type for **Metric 1**, the maximum value is displayed.

Element	Description
Average <Metric 1>	Average of **Metric 1** over the objects in the snapshots of the selected period.
Sum <Metric 1> Trend / **Max <Metric 1> Trend**	Difference of **Metric 1** between the earliest snapshot and the latest snapshot in the selected period.
Average <Metric 1> Trend	Average of **Metric 1** over the objects in the earliest snapshot and the latest snapshot
Arrow beside a value	The orientation and the color of an arrow indicate the development trend of the calculated metric: • Down arrow, green: Decreasing number of code violations. • Up arrow, red: Increasing number of code violations. There is no trend assigned to the **Lines of Code** metric.
Top 5 - Present Values bar chart	Displays the 5 objects with the highest **Metric 1** value. Click a bar in the bar chart to open the corresponding **<Project> - Snapshot Details** window, page 19.
Top 5 Trend - Relative Changes bar chart	Compares the earliest snapshot and the latest snapshot and displays the five objects which differ most between the earliest snapshot and the latest snapshot. Click a bar in the bar chart to open the corresponding **<Project> - Snapshot Details** window, page 19.
The objects displayed in a bar chart provide additional information via tooltip.	

CONVENTIONS DASHBOARD

Overview

Click **PROJECT SNAPSHOTS > <PROJECT> > CONVENTIONS DASHBOARD** to open this window.

The **CONVENTIONS DASHBOARD** window provides the following parts:

- **Select Snapshots**

  Period for which data is displayed in the **<Project> - Conventions Dashboard** view.

  By default, the last two snapshots analyzed are selected.

- **Filter by Object Type**

  Allows you to filter the conventions data for specific object types.

- **<Project> - Conventions Dashboard**

  Based on the period (selected snapshots), the applied filter, and the selected conventions, the most relevant figures and nodes are displayed.

Select Snapshots

Element	Description
Start	Select a start snapshot from the list. The list provides the name, the version (if available), and the analysis date of the snapshots. The differences between the start snapshot and the end snapshot are displayed on the dashboard.
End	Select an end snapshot from the list.
Slider	Alternatively, you can use a slider to define the start snapshot and end snapshot.

Filter by Object Type

You can filter the **<Project> - Conventions Dashboard** content based on object types. By default no object type check box is activated and data for all object types is displayed. Activate the check box of an object type to limit the displayed data to one object type. You can also activate several check boxes.

<Project> - Conventions Dashboard

Based on the period (selected snapshots), the applied filter, and the selected conventions, the most relevant figures and nodes are displayed.

Element	Description
Violations Total	Total number of violations.
Number of Errors	Number of violations with severity **Error**.
Number of Warnings	Number of violations with severity **Warning**.
Number of Information	Number of violations with severity **Information**.
Violations Total Trend	Difference of total number of violations between the earliest snapshot and the latest snapshot in the selected period.
Number of Errors Trend	Average of violations with severity **Error** over the objects in the earliest snapshot and the latest snapshot.
Number of Warnings Trend	Average of violations with severity **Warning** over the objects in the earliest snapshot and the latest snapshot
Number of Information Trend	Average of violations with severity **Information** over the objects in the earliest snapshot and the latest snapshot
Arrow beside a value	The orientation and the color of an arrow indicate the development trend of the convention violations: • Down arrow, green: Decreasing number of violations. • Up arrow, red: Increasing number of violations.
Top 5 Violations bar chart	Displays the five objects with the highest number of violations. Click a bar in the bar chart to open the corresponding **<Project> - Snapshot Details** window, page 19.
Top 5 Violations Trend bar chart	Compares the earliest snapshot and the latest snapshot and displays the five objects which differ the most between the earliest snapshot and the latest snapshot. **NOTE:** Although, for example, a method is displayed in the bar chart, the detected violation can refer to a violation in a variable of this method. Click a bar in the bar chart to open the corresponding **<Project> - Snapshot Details** window, page 19.
The objects displayed in a bar chart provide additional information via tooltip.	

DETAILS VIEW

Overview

Click **PROJECT SNAPSHOTS > <PROJECT> > DETAILS VIEW** to open this window.

The **DETAILS VIEW** window provides the following parts:

- **Select Snapshot**

  Selected Snapshot which data is displayed in the **<Project> - Snapshot Details** view.

  By default, the latest snapshot is preselected.

- **Folders/Groups**

  Allows you to filter the metrics data according to folders/groups provided by your analysis project. The **Groups** tab is only displayed if groups are defined in your analysis project.

  Also refer to chapter *Dependency View (Groups)* in the *Code Analysis User Guide* (see EcoStruxure Machine Expert, Code Analysis, User Guide).
- **Filter by Object Type**

  Allows you to filter the metrics data for specific object types.
- **Show Trend For**

  Period for which the trends are displayed in the **<Project> - Snapshot Details** view (only displayed if you select a single object in the heatmap that is not a folder or a group in your analysis project).
- **<Project> - Snapshot Details**

  Based on the selected snapshot, the selected folders/groups, and the applied filter metrics data is displayed. Additionally you can apply further filter criteria in this view and explore your metric data down to the point of a single object, for example, a function block.

Select Snapshot

Element	Description
Selected Snapshot	Select a snapshot from the list. The list provides the name, the version (if available), and the analysis date for the snapshot.

Folders/Groups

Allows you to filter the metrics data according to folders/groups provided by your analysis project. The **Groups** tab is only displayed if groups are defined in your analysis project.

This filter is associated with the **Subfolders of Selected Folders/Subfolders of Selected Groups** check box in the **<Project> - Snapshot Details** view under **Show Metrics Filtered By**. By default **FB/PRG/FC/ITF/GVL/DUT** is selected.

Filter by Object Type

You can filter the **<Project> - Snapshot Details** content based on object types. By default no object type check box is activated and data for all object types of the list is displayed. Activate the check box of an object type to limit the displayed data to the desired object type. You can also activate several check boxes.

<Project> - Snapshot Details

Based on the selected snapshot, the selected folders/groups, and the applied filter metrics data is displayed. Additionally you can apply further filter criteria in this view and explore your metric data down to the point of a single object, for example, a function block.

- The upper section of this view (from **Impact Metric (Size)** down to the heatmap) is permanently displayed.
- The lower section of this view (from **Impact Metric Trend** down to a list of detailed values) is only displayed if you select a single object in the heatmap that is not a folder or a group in your analysis project, for example, a function block.

Upper section of **<Project> - Snapshot Details**:

Element	Description
Impact Metric (Size)	By default, **Lines Of Code** with aggregation type **Sum** is selected. This value is associated with the size of the rectangle in the heatmap. (The greater the metric value of an object, the greater the size of the corresponding rectangle is displayed in the heatmap).
Quality Metric (Color)	By default, **Halstead Difficulty** with aggregation type **Max** is selected. This value is associated with the color of the rectangle in the heatmap. The color range extends from green (good quality) to red (poor quality).
Impact Metric Aggregation Type / Quality Metric Aggregation Type	Select an aggregation type (**Max**, **Min**, **Sum**, **Average**). Which aggregation types can be selected depends on the metric. Only the available combinations are displayed. For example, for **Halstead Difficulty**, only **Max** and **Average** are displayed as **Sum** is not available for **Halstead Difficulty**.
Group Metrics by	• **Folders / Groups** Activate this check box to organize the rectangles in the heatmap hierarchically according to the folders/groups in your analysis project. • **FB/PRG/FC/ITF/GVL/DUT** By default this check box is activated. Activate this check box to ignore the folders/groups structure of the application. The rectangles in the heatmap display directly the **FB/PRG/FC/ITF/GVL/DUT** of the analysis project.
Heatmap	At the top of the heatmap, a breadcrumb navigation is provided. Click a folder/group to see its subnodes (subfolders/subgroups/objects). Click an object (FB/PRG/FC/ITF/GVL/DUT) to see its actions, methods, and so on. The objects displayed in a heatmap provide additional information via tooltip.

Lower section of **<Project> - Snapshot Details**:

Element	Description
Show Trend For	Select a period for which the **Impact Metric Trend** and the **Impact Quality Trend** are displayed.
Impact Metric Trend	Line chart for the development trend of the selected **Impact Metric (Size)**.
Impact Quality Trend	Line chart for the development trend of the selected **Quality Metric (Color)**.

Element	Description
Detailed information	Detailed information for the selected object. For example, for a function block the following is provided: • **<Name>** Name of the selected object. • Number of **Call In** • Number of **Call Outs** • **Lines Of Code** • **Number of Variables**. • Number of **Reads**. • Number of **Writes**.
Table of convention violations	The table provides a filter and navigation toolbar. The table provides: • **Convention Type** Name of the query chain group. • **Convention** Convention violation. • **Object Name** Name of the affected object, for example, variable name. Click a column header to sort the table according to the respective column.

METRICS ARCHIVE & EXPORT

Overview

Click **PROJECT SNAPSHOTS > <PROJECT> > METRICS ARCHIVE & EXPORT** to open this window.

The **METRICS ARCHIVE & EXPORT** window provides the following parts:

- **Select Snapshot**

  Selected Snapshot which data is displayed in the **<Project> - Metrics Archive & Export** view.

  By default, the latest snapshot is preselected.

- **Metrics**

  Metrics which data is displayed in the **<Project> - Metrics Archive & Export** view.

- **<Project> - Metrics Archive & Export**

  Based on the selected snapshot and the selected metrics, metrics data is displayed and can be exported.

Select Snapshot

Element	Description
Selected Snapshot	Select a snapshot from the list. The list provides the name, the version (if available), and the analysis date for the snapshot.

Metrics

Select the metrics you want to display in the **<Project> - Metrics Archive & Export** view. By default, all metrics are selected.

Activate/deactivate the check box at the top of the list to select/deselect the metrics all at once.

If you activate a metric, a new column is added on the right-hand side of the **<Project> - Metrics Archive & Export** table.

<Project> - Metrics Archive & Export

Based on the selected snapshot and the selected metrics, metrics data is displayed and can be exported.

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries. The filter is only applied to the table but not to the export itself.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

Metrics table

Column	Description
Type	Object type (such as action, method, function).
Name	Object name.
<Metric>	Metrics according to selection on the left-hand side of the window. A number in this column indicates how many hits this metric created. If no number is displayed, this query has not been executed.
Path	Folder path that contains the affected object.
Click a column header to sort the table according to the respective column.	

Click the **EXPORT** button to export the metrics data. A dialog box is displayed to open or save the metrics data as an Excel file.

The exported Excel file provides two worksheets:

- The **Overview** worksheet provides the following information:
  - Analysis project
  - Export time
  - Snapshot name
  - Snapshot upload time
  - Snapshot creation date
- The **Data** worksheet provides the data as displayed in the table.

CONVENTIONS ARCHIVE & EXPORT

Overview

Click **PROJECT SNAPSHOTS > <PROJECT> > CONVENTIONS ARCHIVE & EXPORT** to open this window.

The **CONVENTIONS ARCHIVE & EXPORT** window provides the following parts:

- **Select Snapshot**

  Selected Snapshot which data is displayed in the **<Project> - Conventions Archive & Export** view.

  By default, the latest snapshot is preselected.
- **Conventions**

  Convention groups which data is displayed in the **<Project> - Conventions Archive & Export** view.
- **<Project> - Conventions Archive & Export**

  Based on the selected snapshot, convention violations are displayed and can be exported.

Select Snapshot

Element	Description
Selected Snapshot	Select a snapshot from the list. The list provides the name, the version (if available), and the analysis date for the snapshot.

Conventions

Select the conventions you want to display in the **<Project> - Conventions Archive & Export** view. By default, all conventions are selected.

Activate/deactivate the check box at the top of the list to select/deselect the conventions all at once.

If you activate a convention, a new column is added on the right-hand side of the **<Project> - Conventions Archive & Export** table.

<Project> - Conventions Archive & Export

Based on the selected snapshot, convention violations are displayed and can be exported.

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries. The filter is only applied to the table but not to the export itself.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.

Element	Description
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

Conventions table

Column	Description
Type	Object type (such as variable).
Name	Object name.
Convention	Conventions used in the snapshot. Conventions according to selection on the left-hand side of the window.
Severity	Severity of code violation (**Information** or **Warning**).
Message	Detailed information on the code violation.
Path	Folder path that contains the affected object.
Click a column header to sort the table according to the respective column.	

Click the **EXPORT** button to export the convention violations. A dialog box is displayed to open or save the conventions data as an Excel file.

The exported Excel file provides two worksheets:

- The **Overview** worksheet provides the following information:
  - Analysis project
  - Export time
  - Snapshot name
  - Snapshot upload time
  - Snapshot creation date
- The **Data** worksheet provides the data as displayed in the table.

CODE QUALITY REPORT

Overview

Click **PROJECT SNAPSHOTS > <PROJECT> > CODE QUALITY REPORT** to open this window.

The **CODE QUALITY REPORT** window provides the following parts:

- **Select Snapshots**

  Period for which the quality report is defined in the **<Project> - Code Quality Report** view.
- **<Project> - Code Quality Report**

  Based on the period (selected snapshots), the scope of the quality report can be defined.

Select Snapshots

Element	Description
Snapshot	Select a snapshot from the list. The list provides the name, the version (if available), and the analysis date of the snapshots.
Comparison Snapshot	Select a snapshot from the list to use as a basis for comparison. By default, the last snapshot that was released is selected

<Project> - Code Quality Report

Based on the selected snapshots, the scope of the quality report can be defined.

Element	Description
Scope definition	Select one or several of the following options to define the scope of the quality report: • Create chapter with only the new convention violations. • Create chapter with all convention violations. • Create chapter with description of metric and convention checks used.
Download PDF	Click this button to save the quality report as a *<Name>.pdf* file on your local PC.

ALL MY PROJECTS

Overview

Open this window by clicking:

- **PROJECT SNAPSHOTS > ALL MY PROJECTS** or
- **PROJECT TRENDS > ALL MY PROJECTS** or
- **ADMINISTRATION > ALL MY PROJECTS**

The **ALL MY PROJECTS** window lists your projects and provides links to the different windows.

Project Snapshots

- **> Overview** window, page 15
- **> Metrics Dashboard** window, page 17
- **> Conventions Dashboard** window, page 18
- **> Details View** window, page 19
- **> Metrics Archive & Export** window, page 22
- **> Conventions Archive & Export** window, page 24

Project Trends

- **> Dashboard** window, page 28
- **> Details View** window, page 29
- **> Metrics Archive & Export** window, page 32

Administration

- **> Snapshots** window, page 42
- **> Processing Jobs** window, page 44
- **> Rule Set** window, page 45
- **> Development Phases** window, page 47
- **> Access Tokens** window, page 48

PROJECT TRENDS

DASHBOARD

Overview

Click **PROJECT TRENDS > <PROJECT> > DASHBOARD** to open this window.

The **DASHBOARD** window provides the following parts:

- **Select Snapshots**

  Period for which data is displayed in the **<Project> - Trend Dashboard** view.

  By default, the last two snapshots analyzed are selected.
- **<Project> - Trend Dashboard**

  Based on the period (selected snapshots), the metrics are displayed with their development trend.

Select Snapshots

Element	Description
Start	Select a start snapshot from the list. The list provides the name, the version (if available), and the analysis date of the snapshots. The differences between the start snapshot and the end snapshot are displayed on the dashboard.
End	Select an end snapshot from the list.
Slider	Alternatively, you can use a slider to define the start snapshot and end snapshot.

<Project> - Trend Dashboard

Based on the period (selected snapshots), the metrics are displayed with their development trend.

The bar chart displays the development trends of the metrics as percentage values.

The bar chart provides one bar per combination of metric and aggregation type (**Max**, **Sum**, **Average**).

Only the available combinations are displayed. For example, for **Halstead Difficulty**, only **Max** and **Average** are displayed as **Sum** is not available for **Halstead Difficulty**.

Depending on the trend, the bars are colored:

- Green: Positive development trend.
- Red: Negative development trend.
- Blue: Neutral development trend. (Cannot be rated as positive or negative, for example, the **Lines of Code** value).

Whether a development trend is considered as positive, negative, or neutral depends on the metric.

Click a bar in the bar chart to open the corresponding **<Project> - Trend Details Dashboard** window, page 29.

The objects displayed in a bar chart provide additional information via tooltip.

METRICS DETAILS VIEW

Overview

Click **PROJECT TRENDS > <PROJECT> > METRICS DETAILS VIEW** to open this window.

The **METRICS DETAILS VIEW** window provides the following parts:

- **Select Snapshots**

  Period for which data is displayed in the **<Project> - Metric Trend Details** view.

  By default, the most recent entry is compared to the 30th entry in the history.

- **Metrics**

  Metrics which data is displayed in the **<Project> - Metric Trend Details** view.

- **<Project> - Metric Trend Details**

  Based on the period (selected snapshots) and the selected metrics the metrics are displayed with their detailed development trend.

Select Snapshots

Element	Description
Start	Select a start snapshot from the list. The list provides the name, the version (if available), and the analysis date of the snapshots. The differences between the start snapshot and the end snapshot are displayed on the dashboard.
End	Select an end snapshot from the list.
Slider	Alternatively, you can use a slider to define the start snapshot and end snapshot.

Metrics

Select up to three metrics you want to display in the **<Project> - Metric Trend Details** view. By default, **Lines Of Code** with aggregation type **Sum** is selected.

Element	Description
Metric Name	Select a metric from the list.
Aggregation Type	Select an aggregation type (**Max**, **Min**, **Sum**, **Average**). Which aggregation types can be selected depends on the metric.
ADD METRIC button	Click this button to add a metric. Up to three metrics can be displayed.
DELETE button (minus sign)	Click this button to remove the selected metric. At least one metric must be present.

<Project> - Metric Trend Details

Based on the period (selected snapshots) and the selected metrics the metrics are displayed with their detailed development trend.

Element	Description
Metric Trend	This line chart provides the total values for the selected metrics over time. The X-axis displays the selected snapshots. One Y-axis (with different scaling and color) is displayed for each selected metric. The objects displayed in a line chart provide additional information via tooltip.
Show Growth Rate by	These line charts provide the growth rate for the selected metrics over time. One line chart is displayed for each selected metric. The growth rate can be displayed by: • **Snapshot** • **Day** Average of the snapshots per day. • **Week** Average of the snapshots per week. • **Month** Average of the snapshots per month. This setting applies to all growth rate line charts. The objects displayed in a line chart provide additional information via tooltip.
Top Changes by <Metric Name> bar chart	These bar charts provide the five objects with the highest value. For this, the earliest snapshot and the latest snapshot are compared. One bar chart is displayed for each selected metric. Depending on the trend, the bars are colored: • Green: Positive development trend. • Red: Negative development trend. • Blue: Neutral development trend. (Cannot be rated as positive or negative, for example, the **Lines of Code** value). Whether a development trend is considered as positive, negative, or neutral depends on the metric. Click a bar in the bar chart to open the corresponding **<Project> - Snapshot Details** window, page 19. The objects displayed in a bar chart provide additional information via tooltip.

CONVENTIONS DETAILS VIEW

Overview

Click **PROJECT TRENDS > <PROJECT> > CONVENTIONS DETAILS VIEW** to open this window.

The **CONVENTIONS DETAILS VIEW** window provides the following parts:

- **Select Snapshots**

  Period for which data is displayed in the **<Project> - Convention Trend Details** view.

  By default, the most recent entry is compared to the 30th entry in the history.

- **Conventions**

  Convention groups which data is displayed in the **<Project> - Convention Trend Details** view.

- **<Project> - Convention Trend Details**

  Based on the period (selected snapshots) and the selected convention groups the conventions are displayed with their detailed development trend.

Select Snapshots

Element	Description
Start	Select a start snapshot from the list. The list provides the name, the version (if available), and the analysis date of the snapshots. The differences between the start snapshot and the end snapshot are displayed on the dashboard.
End	Select an end snapshot from the list.
Slider	Alternatively, you can use a slider to define the start snapshot and end snapshot.

Conventions

Select the convention groups you want to display in the **<Project> - Conventions Trend Details** view. By default, all convention groups are selected.

<Project> - Conventions Trend Details

Based on the period (selected snapshots) and the selected convention groups the convention groups are displayed with their detailed development trend.

Element	Description
Convention Group Trend	This line chart provides the total values for the selected convention group over time. The X-axis displays the selected snapshots. One Y-axis (with different scaling and color) is displayed for each selected convention group. The objects displayed in a line chart provide additional information via tooltip. (For example, the name of the snapshot, the analysis date and so on).
Show Growth Rate by	These line charts provide the growth rate for the selected convention group over time. One line chart is displayed for each selected convention group. The growth rate can be displayed by: • **Snapshot** • **Day** Average of the snapshots per day. • **Week** Average of the snapshots per week. • **Month** Average of the snapshots per month. This setting applies to all growth rate line charts. The objects displayed in a line chart provide additional information via tooltip.
Top Changes bar chart	These bar charts provide the 20 objects with the highest trend development. For this, the earliest snapshot and the latest snapshot are compared. Depending on the trend, the bars are colored: • Green: Positive development trend. • Red: Negative development trend. • Blue: Neutral development trend. (Cannot be rated as positive or negative. Whether a development trend is considered as positive, negative, or neutral depends on the convention group. Click a bar in the bar chart to open the corresponding **<Project> - Snapshot Details** window, page 19. The objects displayed in a bar chart provide additional information via tooltip.

METRICS ARCHIVE & EXPORT

Overview

Click **PROJECT TRENDS > <PROJECT> > METRICS ARCHIVE & EXPORT** to open this window.

The **METRICS ARCHIVE & EXPORT** window provides the following parts:

- **Select Snapshots**

  Period for which data is displayed in the **<Project> - Trends Archive & Export** view.

  By default, the last two snapshots analyzed are selected.

- **Metrics**

  Metrics which data is displayed in the **<Project> - Trends Archive & Export** view.

- **<Project> - Trends Archive & Export**

  Based on the period (selected snapshots) and the selected metrics, metrics data is displayed and can be exported.

Select Snapshots

Element	Description
Start	Select a start snapshot from the list. The list provides the name, the version (if available), and the analysis date of the snapshots. The differences between the start snapshot and the end snapshot are displayed on the dashboard.
End	Select an end snapshot from the list.
Slider	Alternatively, you can use a slider to define the start snapshot and end snapshot.

Metrics

Select the metrics you want to display in the **<Project> - Trends Archive & Export** view. By default, all metrics are selected.

Activate/deactivate the check box at the top of the list to select/deselect the metrics all at once.

If you activate a metric, a new column is added on the right-hand side of the **<Project> - Trends Archive & Export** table.

For each metric the **old** (end snapshot), the **new** (start snapshot), and the **delta** value is displayed in the table.

<Project> - Trends Archive & Export

Based on the period (selected snapshots) and the selected metrics, metrics data is displayed and can be exported.

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries. The filter is only applied to the table but not to the export itself.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

Metrics table

Column	Description
Type	Object type (such as action, method, function).
Name	Object name.
• **<Metric> old** • **<Metric> new** • **<Metric> delta**	• Value of the **End** snapshot A number in this column indicates how many hits this metric created. If no number is displayed, this query has not been executed.

Column	Description
	• Value of the **Start** snapshot A number in this column indicates how many hits this metric created. If no number is displayed, this query has not been executed. • Difference between the **Start** and **End** value. The metrics are displayed according to the selection on the left-hand side of the window.
Path	Folder path that contains the affected object.
Click a column header to sort the table according to the respective column.	

Click the **EXPORT** button to export the metrics data. A dialog box is displayed to open or save the metrics data as an Excel file.

The exported Excel file provides two worksheets:

- The **Overview** worksheet provides the following information:
  - Analysis project
  - Export time
  - Snapshot start name
  - Snapshot start upload time
  - Snapshot start creation date
  - Snapshot end name
  - Snapshot end upload time
  - Snapshot end creation date
- The **Data** worksheet provides the data as displayed in the table.

CONVENTIONS ARCHIVE & EXPORT

Overview

Click **PROJECT TRENDS > <PROJECT> > CONVENTIONS ARCHIVE & EXPORT** to open this window.

The **CONVENTIONS ARCHIVE & EXPORT** window provides the following parts:

- **Select Snapshots**

  Period for which data is displayed in the **<Project> - New Convention Violations Archive & Export** view.

  By default, the last 30 snapshots analyzed are selected.
- **Conventions**

  Convention groups which data is displayed in the **<Project> - New Convention Violations Archive & Export** view.
- **<Project> - New Convention Violations Archive & Export**

  Based on the period (selected snapshots) and the selected conventions, new convention violations are displayed and can be exported.

Select Snapshots

Element	Description
Start	Select a start snapshot from the list. The list provides the name, the version (if available), and the analysis date of the snapshots. The differences between the start snapshot and the end snapshot are displayed on the dashboard.
End	Select an end snapshot from the list.
Slider	Alternatively, you can use a slider to define the start snapshot and end snapshot.

Conventions

Select the conventions you want to display in the **<Project> - New Convention Violations Archive & Export** view. By default, all conventions are selected.

Activate/deactivate the check box at the top of the list to select/deselect the conventions all at once.

If you activate a convention, a new row is added to the **<Project> - New Convention Violations Archive & Export** table.

For each convention, the **Severity** and the respective **Message** is displayed in the table.

<Project> - New Convention Violations Archive & Export

Based on the period (selected snapshots) and the selected conventions, new convention violations are displayed and can be exported.

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries. The filter is only applied to the table but not to the export itself.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

Conventions table

Column	Description
Type	Object type (such as programs, global variable lists).
Name	Object name.
Convention	Conventions used in the snapshot.
Severity	Severity of code violation (**Information** or **Warning**).
Message	Detailed information on the code violation.

Column	Description
Path	Folder path that contains the affected object.
Click a column header to sort the table according to the respective column.	

Click the **EXPORT** button to export the new convention violations. A dialog box is displayed to open or save the conventions data as an Excel file.

The exported Excel file provides two worksheets:

- The **Overview** worksheet provides the following information:
  - Analysis project
  - Export time
  - Snapshot name
  - Snapshot upload time
  - Snapshot start creation date
  - Snapshot base name
  - Snapshot base upload time
  - Snapshot base creation date
- The **New Convention Violations** worksheet provides the data as displayed in the table.

ALL MY PROJECTS

Overview

Refer to ALL MY PROJECTS, page 26.

COMPARE PROJECTS

DASHBOARD

Overview

Click **COMPARE PROJECTS > DASHBOARD** to open this window.

The **DASHBOARD** window provides the following parts:

- **Reference Project**

  Project the **Comparison Projects** are compared with.
- **Comparison Projects**

  Projects to be compared with the **Reference Project**.
- **Compare Filter**

  Allows you to define whether all objects or only the modified objects are to be compared.
- **Compare Projects** view

  Based on the selected reference project snapshot and the selected comparison project snapshots, the metrics are displayed with their development trend.

Reference Project

Element	Description
Select Project	Select a project from the list the **Comparison Projects** are compared with.
Select Snapshot	Select a snapshot from the list. The list provides the name, the version (if available), and the analysis date for the snapshot.

Comparison Projects

Element	Description
Select Project	Select a project from the list to be compared with the **Reference Project**.
Select Snapshot	Select a snapshot from the list. The list provides the name, the version (if available), and the analysis date for the snapshot.
ADD PROJECT button	Click this button to add a project. A new line is added to the **Comparison Projects**. Select project and snapshot for the added project.
DELETE button (minus sign)	Click this button to remove the selected project. At least one project must be present.

Compare Filter

Activate one of the check boxes to compare all objects or only the modified objects of the projects.

Compare Projects

Based on the selected reference project snapshot and the selected comparison project snapshots, the metrics are displayed with their development trend.

The bar chart displays the development trends of the metrics as percentage values.

The bar chart provides one bar per combination of metric and aggregation type (**Max**, **Sum**, **Average**).

The values of the different project snapshots are differentiated by colors.

The objects displayed in a bar chart provide additional information via tooltip.

(Percentage value, project, snapshot).

DETAILS VIEW

Overview

Click **COMPARE PROJECTS > DETAILS VIEW** to open this window.

The **DETAILS VIEW** window provides the following parts:

- **Reference Project**

  Project the **Compare Project** is compared with.

- **Compare Project**

  Project to be compared with the **Reference Project**.

- **Compare Projects Details** view

  Based on the selected reference project snapshot and the selected comparison project snapshot, the top five metrics with their positive (**Higher Metrics Top 5**) and negative (**Lower Metrics Top 5**) development trends, and the top five objects (for a selected metric) with their positive and negative development trends are displayed.

Reference Project

Element	Description
Select Project	Select a project from the list the **Comparison Project** is compared with.
Select Snapshot	Select a snapshot from the list. The list provides the name, the version (if available), and the analysis date for the snapshot.

Compare Project

Element	Description
Select Project	Select a project from the list to be compared with the **Reference Project**.
Select Snapshot	Select a snapshot from the list. The list provides the name, the version (if available), and the analysis date for the snapshot.

Compare Projects Details

Based on the selected reference project snapshot and the selected comparison project snapshot, the top five metrics with their positive (**Higher Metrics Top 5**) and negative (**Lower Metrics Top 5**) development trends, and the top five objects (for a selected metric) with their positive and negative development trends are displayed.

Element	Description
Compare All Objects/ Compare Changed Objects Only	Activate one of the check boxes to compare all objects or only the modified objects of the projects.
Lower Metrics Top 5 bar chart	Displays the top five metrics with their negative development trends based on the selected reference project snapshot and the selected comparison project snapshot. The bar chart displays the development trends of the metrics as percentage values. Depending on the trend, the bars are colored: • Green: Positive development trend. • Red: Negative development trend. • Blue: Neutral development trend. (Cannot be rated as positive or negative, for example, the **Lines of Code** value). Whether a development trend is considered as positive, negative, or neutral depends on the metric.
Higher Metrics Top 5 bar chart	Displays the top five metrics with their positive development trends based on the selected reference project snapshot and the selected comparison project snapshot. The bar chart displays the development trends of the metrics as percentage values. Depending on the trend, the bars are colored: • Green: Positive development trend. • Red: Negative development trend. • Blue: Neutral development trend. (Cannot be rated as positive or negative, for example, the **Lines of Code** value). Whether a development trend is considered as positive, negative, or neutral depends on the metric.
Metric Name	Select a metric from the list.
Aggregation Type	Select an aggregation type (**Max**, **Min**, **Sum**, **Average**). Which aggregation types can be selected depends on the metric.
Top 5 Object Difference (Negative) of <Metric Name> - <Aggregation Type> bar chart	Displays the top five objects with their negative development trends. The bar chart displays the development trends of the metrics as percentage values. Depending on the trend, the bars are colored: • Green: Positive development trend. • Red: Negative development trend. • Blue: Neutral development trend. (Cannot be rated as positive or negative, for example, the **Lines of Code** value). Whether a development trend is considered as positive, negative, or neutral depends on the metric.

Element	Description
Top 5 Object Difference (Positive) of <Metric Name> - <Aggregation Type> bar chart	Displays the top five objects with their positive development trends. The bar chart displays the development trends of the metrics as percentage values. Depending on the trend, the bars are colored: • Green: Positive development trend. • Red: Negative development trend. • Blue: Neutral development trend. (Cannot be rated as positive or negative, for example, the **Lines of Code** value). Whether a development trend is considered as positive, negative, or neutral depends on the metric.
The objects displayed in a bar chart provide additional information via tooltip.	

METRICS ARCHIVE & EXPORT

Overview

Click **COMPARE PROJECTS > METRICS ARCHIVE & EXPORT** to open this window.

The **METRICS ARCHIVE & EXPORT** window provides the following parts:

- **Reference Project**

  Project the **Comparison Projects** are compared with.

- **Comparison Projects**

  Projects to be compared with the **Reference Project**.

- **Compare Filter**

  Allows you to define whether all objects or only the modified objects are to be compared.

- **Compare Projects Archive & Export** view

  Based on the selected reference project snapshot and the selected comparison project snapshots, the metrics data is displayed and can be exported.

Reference Project

Element	Description
Select Project	Select a project from the list the **Comparison Projects** are compared with.
Select Snapshot	Select a snapshot from the list. The list provides the name, the version (if available), and the analysis date for the snapshot.

Comparison Projects

The values for up to five **Comparison Projects** can be displayed in the **Compare Projects Archive & Export** view.

Element	Description
Select Project	Select a project from the list to be compared with the **Reference Project**.
Select Snapshot	Select a snapshot from the list. The list provides the name, the version (if available), and the analysis date for the snapshot.
ADD PROJECT button	Click this button to add a project. A new line is added to the **Comparison Projects**. Select project and snapshot for the added project.
DELETE button (minus sign)	Click this button to remove the selected project. At least one project must be present.

Compare Filter

Activate one of the check boxes to compare all objects or only the modified objects of the projects.

Compare Projects Archive & Export

Based on the selected reference project snapshot and the selected comparison project snapshots, the metrics data is displayed and can be exported.

Metrics table

Column	Description
Metric Type	The column provides one row per combination of metric and aggregation type (**Max**, **Sum**, **Average**). Only the available combinations are displayed.
Reference Project	Metrics value of the reference project.
Delta Project 1	Difference of the metrics value of project 1 compared to the reference project.
...	...
Delta Project 5	Difference of the metrics value of project 5 compared to the reference project.
Click a column header to sort the table according to the respective column.	

Click the **EXPORT** button to export the metrics data. A dialog box is displayed to open or save the metrics data as an Excel file.

The exported Excel file provides two worksheets:

- The **Overview** worksheet provides the following information:
  - Export time
  - Reference analysis project
  - Reference snapshot
  - Reference snapshot creation time
  - Compare analysis project <1...n>
  - Compare reference snapshot <1...n>
  - Compare reference snapshot creation time <1...n>
- The **Data** worksheet provides the data as displayed in the table.

ADMINISTRATION

SNAPSHOTS

Overview

Click **ADMINISTRATION > <PROJECT> > SNAPSHOTS** to open this window.

The **SNAPSHOTS** window provides the following parts:

- **Upload**

  Section to upload a snapshot.
- **<Project> - Snapshots**

  Based on the **<PROJECT>** an overview of the already uploaded project snapshots with detailed information is displayed.

Upload

This section provides two options to upload a snapshot:

- Drag a snapshot file to this section (dashed rectangle).
- Click **Browse**. A dialog box is displayed to select a snapshot file for upload.

<Project> - Snapshots

Based on the **<PROJECT>** an overview of the already uploaded project snapshots with detailed information is displayed.

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

A table provides the following information:

Column	Description
Name	Name of the snapshot created in EcoStruxure Machine Expert.
Version	Version of the snapshot.
Analysis Date	Analysis date of the snapshot.
Processing Date	Processing date of the snapshot.
Upload Date	Upload date of the snapshot.

Column	Description
Size	Size of the snapshot in Megabytes and number of entries.
Click a column header to sort the table according to the respective column.	

The following icons are provided for each snapshot:

Icon	Description
Recycle bin (delete)	Click this icon to delete a snapshot. Confirm the prompt with **Yes** or cancel with **No**. **NOTE:** All assigned analysis data is removed. The deletion cannot be undone.
Pencil (edit)	Click this icon to open the corresponding **<Project> - Snapshot Details** window, page 19. You can edit the following entries: • **Name** • **Version** • **Released** (**Yes**/**No**) • **Branch in Source Control** Subpath from the base URI (defined in the **ADMINISTRATION > ALL COMPANY PROJECTS > <Analysis Project> Details** window, page 51) to the memory location where the source code of the snapshot is provided. • **Changelog** Comment which source code modifications were done. • **Notes** Confirm your edits with the **UPDATE** button. The following information is displayed, but cannot be edited: • **Processing Date** • **Upload Date** • **Uploaded By** • **Revision in Source Control** Revision ID of the project that the uploaded snapshot represents. • **Development Phase** If a development phase was defined, the snapshot is automatically assigned to it and the development phase name is displayed here.
Block List	Click this icon to open the corresponding **<Project> - Blocklist Entries** window which provides the following information for the blocked objects: • **Name** • **Category** • **Comment** This block list is part of the uploaded snapshot. Entries cannot be modified or deleted. Also refer to chapter *Block List* in the *Code Analysis User Guide* (see EcoStruxure Machine Expert, Code Analysis, User Guide).
Reprocess Snapshot	If you reprocess, the snapshot of the existing analysis data is removed. Confirm the prompt with **Yes** or cancel with **No**.
Download Snapshot	Click this icon to save the snapshot as a *<Name>.secas* file on your local PC.

PROCESSING JOBS

Overview

Click **ADMINISTRATION > <PROJECT> > PROCESSING JOBS** to open this window.

The **PROCESSING JOBS** window provides the following:

- **<Project> - Processing Jobs**

  Based on the **<PROJECT>** an overview of the processing jobs with detailed information is displayed.

<Project> - Processing Jobs

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

Activate the **Show Done Jobs** check box to display also the jobs with status **Done**.

A table provides the following information:

Column	Description
Snapshot	Name of the snapshot.
Version	Version of the snapshot.
Released	Release status of the snapshot.
Creation Date	Creation date of the snapshot.
Progress	Analyze progress of the processing job: • **Ready for execution** • **Processing** • **Processing done** • **Processing failed** • **Integration** • **Integration done** • **Integration failed** • **Done** • **Canceled**
Click a column header to sort the table according to the respective column.	

The following icons are provided for each snapshot:

Column	Description
Status	Different icons visualize the status of processing: • **Standing** Processing not started. • **Running** Processing is running. • **Done** Processing is finished. • **Alert** An error is detected during processing. • **No License** The license to launch the processing job is not available or expired.
Delete	Click this icon to delete the processing job.
Show Log	Click this button to open the process log file.

RULE SET

Overview

Click **ADMINISTRATION > <PROJECT> > RULE SET** to open this window.

The **RULE SET** window provides the following parts:

- **Available Rule Sets**

  This section lists the rule sets available for your company and provides information on the rule sets.You can apply a rule set to your project.
- **Available Queries/Active Queries**

  Based on the assigned rule set an overview of the queries is displayed.
- **Reprocessing**

  Section to reprocess the selected queries.

If you open this window and a rule set that is assigned to an analysis project was modified, a dialog box is displayed that informs you that the active rule set has changed. You can confirm with the **Update now** button or postpone with the **Later** button.

Available Rule Sets

This section lists the rule sets available for your company and provides information on the rule sets. The legacy projects are assigned to the rule set **Custom**. If you created a new project, the **Schneider Electric Standard Rules** are assigned. The pre-defined rule sets cannot be modified but you can create a customized rule set for your project in EcoStruxure Machine Expert (version ≥ V2.0).

Element	Description
Available Rules Sets	Lists the rule sets available for your company.
Description	Short description of the selected rule set.

Element	Description
DETAILS button	Click this button to display a list of the queries of the selected rule set.
APPLY button	Click this button to apply the selected rule set to your project. A dialog box is displayed that informs you that applying a new rule set creates processing jobs because a set of queries might be activated or deactivated. This is done with processing jobs. You can confirm with **Yes** or cancel with **No**. Refer to section **Reprocessing** below.

Available Queries (<Rule set>)/Active Queries (<Rule set>)

Depending on the selected rule set this title changes:

- For the pre-defined rule sets (for example, **Schneider Electric Standard Rules**), the **Active Queries** are displayed.
- For the **Custom** rule set, the **Available Queries** are displayed.

Based on the assigned rule set an overview of the queries is displayed.

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

If the **Custom** rule set is activated, two additional buttons are available beside the filter and navigation toolbar:

- **Assign All Queries**: To assign all queries (latest version) to the project. A processing job (if not already available) is prepared to perform this modification.
- **Remove All Queries**: To remove all queries from the project. A processing job (if not already available) is prepared to perform this modification.

A table provides the following information:

Column	Description
Category	Type of the query (convention or metric).
Group	Group name of the query as defined in EcoStruxure Machine Expert Logic Builder.
Name	Name of the query.
Revision	Revision number and date of the query. Click the table entry for the revision number and select a revision you want to assign to the project. A processing job (if not already available) is prepared to perform this modification.
Click a column header to sort the table according to the respective column.	

The following button is provided for each table row:

Button	Description
Magnifier	Click the **Magnifier** button to open the corresponding **<Query Chain> - Details View** which provides the following information: • **General Information** ◦ **Name**, **Group**, **Category** ◦ **Updated by**, **Last Update** ◦ **Description** • **Query Chain Parameters** ◦ **Parameter Name**, **Parameter Value**, **Parameter Type** • **Query Detail Information** ◦ **<Query Name>** ◦ **Source Code** (SPARQL)

If the **Custom** rule set is activated, two additional buttons are provided for each table row:

Button	Description
Add to active query set	Click this button to assign the query (latest version) to the active query set. A processing job (if not already available) is prepared to perform this modification.
Remove from active query set	Click this button to remove the query from the active query set. A processing job (if not already available) is prepared to perform this modification.

Reprocessing

This section provides the number of prepared processing jobs (one for each affected project), and a list of queries that are reprocessed within these jobs.

Click **START REPROCESSING** to reprocess the listed queries.

If you reprocess the queries, the existing analysis data is replaced.

DEVELOPMENT PHASES

Overview

Click **ADMINISTRATION > <PROJECT> > DEVELOPMENT PHASES** to open this window.

The **DEVELOPMENT PHASES** window provides the following:

- **<Project> - Development Phases**

  Based on the **<PROJECT>** an overview of the development phases is displayed. You can add or remove development phases.

<Project> - Development Phases

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items

Element	Description
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

A table provides the following information:

Column	Description
Name	Name of the development phase (for example, alpha, beta, field test, etc.) Click an entry in this column to edit this text. Confirm with **OK** or cancel with **X**.
Description	Description of the development phase. Click an entry in this column to edit this text. Confirm with **OK** or cancel with **X**.
Start Date	Start date of the development phase. Click an entry in this column to edit the date. Confirm with **OK** or cancel with **X**.
End Date	End date of the development phase. Click an entry in this column to edit the date. Confirm with **OK** or cancel with **X**.
Click a column header to sort the table according to the respective column.	

The following icon is provided for each development phase row:

Icon	Description
Recycle bin (delete)	Click this icon to delete a development phase. Confirm the prompt with **Yes** or cancel with **No**.

Click the **ADD DEVELOPMENT PHASE** button to add a development phase. A new line is added to the table. Edit **Name** and **Description**, and select **Start Date** and **End Date** for the added development phase.

ACCESS TOKENS

Overview

You can create a technical user (access token) that is allowed to upload a snapshot for one specific analysis project. For example, for CI-Systems (Continuous Integration Systems), it is possible to upload snapshots via Python scripts.

Click **ADMINISTRATION > <PROJECT> > ACCESS TOKENS** to open this window.

The **ACCESS TOKENS** window provides the following parts:

- **New Access Token**

  Section to create a new access token.

- **<Project> - Access Tokens**

  Based on the **<PROJECT>** an overview of access tokens is displayed. You can remove access tokens from this table.

New Access Token

Step	Action
1	Enter a **Name**.
2	Select an **Expiration Date**. After this date, the access token cannot be used to upload a snapshot.
3	Click the **CREATE** button. **Result**: A window is displayed which provides the access token as a character string.
4	Retain this string in a secure manner because the string is provided only once.
5	**Close** the window. **Result**: A new row is displayed in the **<Project> - Access Tokens** table.

<Project> - Access Tokens

Based on the **<PROJECT>** an overview of access tokens is displayed. You can remove access tokens from this table.

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.

Activate the **Show Expired Access Tokens** check box to also display access tokens which are expired.

A table provides the following information:

Column	Description
Name	Name of the access token.
Last Login	Date of last login with this access token.
Expiration Date	Date when the access token expires.
Expired	Status of the access token (**No**/**Yes**).
Creation Date	Date when the access token was created.
Click a column header to sort the table according to the respective column.	

The following icon is provided for each access token:

Column	Description
Recycle bin (delete)	Click this icon to delete an access token. Confirm the prompt with **Yes** or cancel with **No**.

ALL MY PROJECTS

Overview

Refer to ALL MY PROJECTS, page 26.

ALL QUERY CHAINS

Overview

Click **ADMINISTRATION > ALL QUERY CHAINS** to open this window.

The **ALL QUERY CHAINS** window provides the following:

- **All Query Chains**

  An overview of all query chains with detailed information is displayed.

All Query Chains

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

A table provides the following information:

Column	Description
Category	Type of the query chain (convention or metric).
Group	Group name of the query chain as defined in EcoStruxure Machine Expert Logic Builder.
Name	Name of the query chain.
Upload Date	Upload date of the query chain.
Revision Number	Revision number of the query chain.
Previous Revision	Information on the previous revision: • **Revision Number** • **Upload Date**
Click a column header to sort the table according to the respective column.	

The following icon is provided for each table row:

Icon	Description
Magnifier	Click a table row or the **Magnifier** icon to open the corresponding **<Query Chain> - Details View** which provides the following information: • **General Information** ◦ **Name**, **Group**, **Category** ◦ **Updated By**, **Last Update** ◦ **Description** • **Query Detail Information** ◦ **<Query Name>** ◦ **Source Code** (SPARQL)

ALL COMPANY PROJECTS

Overview

Click **ADMINISTRATION > ALL COMPANY PROJECTS** to open this window.

The **ALL COMPANY PROJECTS** window provides the following:

- **Analysis Projects**

  An overview of all analysis projects with detailed information is displayed.

Analysis Projects

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

A table provides the following information:

Column	Description
Name	Name of the analysis project.
Remaining Uploads	Remaining snapshot that can be uploaded to the EcoStruxure Machine Advisor Code Analysis portal.
License Information	• No license • License valid until <expiring date>
Description	Description of the analysis project.
Click a column header to sort the table according to the respective column.	

The following icons are provided for each analysis project:

Icon	Description
Pencil (edit)	Click this icon to open the corresponding **<Analysis Project > Details** window. You can edit the following entries: • **Name** • **Version Control Base URI** You need this URI to launch the **Tortoise SVN** tool. Refer to snapshot overview, page 15. • **Description** Confirm your edits with the **UPDATE** button. Additionally the **<Analysis Project > Details** window provides the following sections: • **Assigned Users** This section lists the users assigned to the analysis project. You can remove an assigned user from the project. A removed user is added to **Available Users** list. • **Available Users** This section lists the users available for your company. You can add a user to the project. The user is added to **Assigned Users** list.
Shopping Cart	Click this icon to open the **Present License / Available Packages and Licenses** window, which provides: • **Present License** Information on the present license of the project: ◦ Validity period (for example, license for one year) ◦ Valid until <expiring date> ◦ Remaining uploads • **Analysis Project License** Provides information on the license you can buy: ◦ Validity period. ◦ Number of snapshots that can be uploaded. ◦ Number of seats the license can be used on. ◦ Number of Machine Advisor Points (Schneider Electric currency) ◦ **BUY** button: Click this button to buy the license. After completing your registration on the EcoStruxure Machine Advisor portal, you are able to buy Machine Advisor Points with different payment options. **NOTE:** To buy licenses, you need to complete your registration on the EcoStruxure Machine Advisor portal, and fill your wallet with Machine Advisor Points. • **Snapshot Upload / Analysis Package** Provides information on the license upgrade you can purchase: ◦ Number of snapshots that can be uploaded. ◦ Number of Machine Advisor points (Schneider Electric currency) ◦ **BUY** button: Click this button to buy the license upgrade.
Recycle bin (delete)	Click this icon to delete an analysis project. Confirm the prompt with **Yes** or cancel with **No**. **NOTE:** All snapshots with their analysis data are removed. Deletion is only possible if no license has been activated for that snapshot. The deletion cannot be undone.
->	Click this icon to move an analysis project to another company. If you move an analysis project to another company, analysis data is removed from your project and you have to reprocess your analysis. **NOTE:** Moving is only possible if no license has been activated for the snapshot.

Click the **ADD ANALYSIS PROJECT** button to open the **<Analysis Project > Details** window. Edit the details and assign users to the new project.

USER AND USER ROLES

Overview

Click **ADMINISTRATION > USER AND USER ROLES** to open this window.

The **USER AND USER ROLES** window provides the following parts:

- **Invite User**

  Section to invite a user to your company.

- **Users**

  An overview of all users assigned to your company with detailed information is displayed.

Invite User

Step	Action
1	Enter the email address of the user you want to invite to your company.
2	Click the **INVITE** button. **Result**: An email is sent to the invited user. This email provides a link to the EcoStruxure Machine Advisor Code Analysis portal.
3	The invited user must sign up for the EcoStruxure Machine Advisor Code Analysis portal.
4	After signing up, the invited user can log into the EcoStruxure Machine Advisor Code Analysis and the invited user is directly assigned to your company. The projects the user should have access to can be selected in the **<User > Details** window of the invited user. See below.

Users

An overview of all users assigned to your company with detailed information is displayed.

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.

A table provides the following information:

Column	Description
Name	Name of the user.
Email	Email address of the user.
Click a column header to sort the table according to the respective column.	

The following icons are provided for each analysis project:

Icon	Description
Pencil (edit)	Click this icon to open the corresponding **<User > Details** window. The **User** section provides the following information: • **Name**: User Name • **Email**: Email address of the user • **Last login**: Date and time of last login • **Role**: User role (refer to User Roles/User Rights, page 54) ◦ **Administrator** ◦ **Quality Officer** ◦ **User** You can edit the user role. Confirm your edits with the **UPDATE** button. Additionally, the **<User > Details** window provides the following sections: • **Assigned Projects** This section lists the projects assigned to the user. You can remove an assigned project from the user. A removed project is added to **Available Projects** list. • **Available Projects** This section lists the projects available for your company. You can add a project to a user. The project is added to **Assigned Projects** list.
X (delete)	Click this icon to delete a user. Confirm the prompt with **Yes** or cancel with **No**.

User Roles/User Rights

The following table provides an overview of user roles/user rights:

	User	**Quality Officer**	**Administrator**
HOME	Yes	Yes	Yes
PROJECT SNAPSHOTS	Yes	Yes	Yes
PROJECT TRENDS	Yes	Yes	Yes
COMPARE PROJECTS	Yes	Yes	Yes
ADMINISTRATION >			
<PROJECT> SNAPSHOTS	No	Yes	Yes
<PROJECT> SNAPSHOTS > Block List	No	Yes	Yes
<PROJECT> PROCESSING JOBS	No	Yes	Yes
<PROJECT> RULE SET	No	Yes	Yes
<PROJECT> DEVELOPMENT PHASES	No	Yes	Yes
<PROJECT> ACCESS TOKENS	No	No	Yes
ALL MY PROJECTS	No	Yes	Yes
ALL QUERY CHAINS	No	Yes	Yes
ALL COMPANY PROJECTS	No	No	Yes
USER AND USER ROLES	No	No	Yes
COMPANY SETTINGS	No	No	Yes
LICENSE	Yes	Yes	Yes
Yes = User rights are available. No = User rights are not available.			

PURCHASES

Overview

Click **ADMINISTRATION > PURCHASES** to open this window.

The window provides information on the already acquired licenses.

Filter and navigation toolbar

Element	Description
Filter	Enter text to filter the table entries.
Items Per Page	Select a number from the list to define how many items are displayed per page.
1 - xx/zzz	**xx** = Number of items per page **zzz** = Total number of items
|<	Go to the first page.
<	Go back one page.
>	Go forward one page.
>|	Go to the last page.
Refresh	Refresh view.

A table provides the following information:

Column	Description
Project Name	Name of your analysis project.
Type	Full license or license upgrade.
Number of SE Points	Number of SE points (Schneider Electric currency).
License Key	Key used for licensing on the EcoStruxure Machine Advisor portal.
Included Analysis of Snapshots	Number of snapshots that could be uploaded/analyzed.
Bought by	Name of the acquirer.
Bought at	Date of acquisition.
Click a column header to sort the table according to the respective column.	

User and Company Settings

Overview

Click the **<User Name>** icon in the upper right corner of the Machine Advisor Code Analysis GUI to open the **User and Company Settings** window.

The window provides the following parts:

- **User**
  - **Update user settings**
  - **Forget me**
- **Company**
  - **Update company settings**
  - **Forget me and my company**

User

Click **Update user settings** to open the **Profile** window.

- In the **My Profile** tab, you can enter/modify the **Name**, **Language Preference**, **Email**, **Mobile Number**, and **Password**.
- In the **My Address** tab, you can enter/modify your postal address.
- In the **My Company** tab, you can enter/modify the **Company Name**, **Employee Size**, **Location**, **State/Province**, **Function**, **Business Type**, **My Industry Segment**, **Sub Segment**, and **Industries I Serve**.

  Click the **UPDATE** button to confirm your edits or quit with **Cancel**.

Click **Forget me** to delete the user account.

- A dialog box prompts you to confirm the deletion.

  NOTE: If you confirm the deletion, the user account is removed from the company and you are logged out.

Company

This window is only available if you are logged in as **Administrator**.

Click **Update company** to open the **Company settings** window.

- You can enter/edit the **Name**, **Street**, and **City** of the postal address of your company.

  Click the **UPDATE** button to confirm your edits or quit with **Cancel**.

Click **Forget me and my company** to delete the company account.

- A dialog box prompts you to confirm the deletion. Confirm with **Yes** or cancel with **No**.

  NOTE: If you confirm the deletion:

  - The company is removed.
  - All projects and all snapshots are removed.
  - All assigned users are removed from the company. (But can still be assigned to other companies).
  - You are logged out.

How to Use the EcoStruxure Machine Advisor Code Analysis

Create a Company Account and Invite Users

Overview

If you want to assign more than one user to your company, invite the users to your company before they sign up.

This is because for each user that is not invited to an existing company, a new company is created during sign-up. Therefore, a separate license is required and its own analysis projects have to be created. Licenses and analysis projects cannot be shared by two companies.

Sign Up

Step	Action
1	Paste *ecostruxure-machine-advisor-codeanalysis.se.app* into your internet browser, page 11. **Result**: You are directed to the EcoStruxure Machine Advisor Code Analysis platform.
2	Click the **Sign Up** button. **Result**: The **Sign Up** window is displayed.
3	Enter your email address and confirm with **Next**.
4	Enter your personal and the company information.
5	Read the **SE Terms & Conditions of Use** and if you so choose, select the option **I have read, understood, and agree to the SE Terms & Conditions of Use including those relating to the associated communications**, and click **Sign Up**. **Result**: You receive an email to complete your registration.

Login

Step	Action
1	Click the **Confirm your email and complete your registration** link in the email you received. **Result**: You are directed to the EcoStruxure Machine Advisor Code Analysis platform.
2	Enter your email address and the password you defined in the **Sign Up** window.
3	Click **Login**. **Result**: The **HOME** windows, page 13 opens.

Invite Users

Step	Action
1	Click **ADMINISTRATION > USER AND USER ROLES** in the menu bar.
2	Enter the **Email** address of the user you want to invite.
3	Click the **INVITE** button.

Step	Action
	Result: An email is sent to the invited user. This email provides a link to the EcoStruxure Machine Advisor Code Analysis platform.
4	The invited user must sign up for the EcoStruxure Machine Advisor Code Analysis platform.
5	After signing up, the invited user can log into the EcoStruxure Machine Advisor Code Analysis and the invited user is directly assigned to your company.

Assign User Role and Projects

Assign user role:

Step	Action
1	Click **ADMINISTRATION > USER AND USER ROLES** in the menu bar.
2	Click the **Pencil** icon right-hand side of a listed user to open the corresponding **<User > Details** window.
3	Select a **Role** (**Administrator**, **Quality Officer**, **User**) for the user and confirm with the **UPDATE** button. Refer to User Roles/User Rights, page 54.

Assign Projects:

Step	Action
1	Click **ADMINISTRATION > USER AND USER ROLES** in the menu bar.
2	Click the **Pencil** icon right-hand side of a listed user to open the corresponding **<User > Details** window.
3	Click the **+** icon right-hand side of a listed project in the **Available Projects**. **Result**: The project is added to the **Assigned Projects** list.
4	Click the **X** icon right-hand side of a listed project in the **Assigned Projects**. **Result**: The project is removed from the **Assigned Projects** list and added to the **Available Projects**.

Create a New Analysis Project and Assign Users

Configure a New Analysis Project

Step	Action
1	Click **ADMINISTRATION > ALL COMPANY PROJECTS** in the menu bar (only available for administrators).
2	Click the **ADD ANALYSIS PROJECT** button to open the **<Analysis Project > Details** window.
3	Enter: • A **Name** for the project. • A **Version Control Base URI**. This entry can contain a URI that provides the location of the SVN source code of the project. This URI is used to open, for example, **SVN-CHANGELOG** (supported by the **Tortoise SVN** tool). Refer to <Project> - Snapshot Overview, page 15. If no value is configured for **Version Control Base URI**, the value is updated while uploading a snapshot of an EcoStruxure Machine Expert Logic Builder project which is SVN bounded. • A **Description** for the project.

Assign Users to a Project/Remove Users from a Project

Step	Action
1	Click the **+** icon right-hand side of a user in the **Available Users** list. **Result**: The user is added to the **Assigned Users** list of the project.
2	Click the **X** icon right-hand side of a user in the **Assigned Users** list. **Result**: The user is removed from the **Assigned Users** list and added to the **Available Users** list.

Create a Snapshot and Upload (with Personal User Account)

Overview

You can upload a snapshot to the EcoStruxure Machine Advisor Code Analysis portal:

- With your personal user account.
- With a technical user account (access token).

  Refer to Create an Access Token and Upload a Snapshot with Python Script, page 60.

Upload Automatically

Step	Action
1	Launch the EcoStruxure Machine Expert Logic Builder and open your project.
2	Right-click the **Application** node in the **Application tree**.
3	Select **Add Object > Code Analysis Manager** from the contextual menu.
4	Select the **Cloud Connection** tab and click the **Login…** button. **Result**: The **Code Analysis Cloud Login** window is displayed. **NOTE:** To use the **Cloud Connection**, a license is required.
5	Log in with your personal account (email address and password) to the EcoStruxure Machine Advisor Code Analysis portal. **Result**: After a successful login, the **Code Analysis Cloud Login** window disappears.
6	In the **Cloud Connection** tab, select the **Company** and the **Analysis Project** the snapshot should be uploaded to.
7	Click the **Analyze Project** button. **Result**: The project is analyzed and the snapshot is created.
8	Click the **Upload Snapshot** button. **Result**: The snapshot is automatically uploaded to the EcoStruxure Machine Advisor Code Analysis portal and the uploaded snapshot is displayed in the **<Project> - Snapshots** overview, page 42.

Upload Manually

Step	Action
1	Launch the EcoStruxure Machine Expert Logic Builder and open your project.
2	Right-click the **Application** node in the **Application tree**.
3	Select **Add Object > Code Analysis Manager** from the contextual menu.
4	Select the **Cloud Connection** tab.
5	Click the **Analyze Project** button. **Result**: The project is analyzed and the snapshot is created.
6	Click the **Store Snapshot** button to save the snapshot as a file on your local PC.
7	Log in with your personal account (email address and password) to the EcoStruxure Machine Advisor Code Analysis portal.
8	Click **ADMINISTRATION > <PROJECT> > SNAPSHOTS** to open the **<Project> - Snapshots** overview, page 42.
9	Drag the stored snapshot file to the **Upload** section (dashed rectangle). **Result**: The uploaded snapshot is displayed in the **<Project> - Snapshots** overview, page 42.
10	Alternatively click **Browse** in the **Upload** section (dashed rectangle). **Result**: A dialog box is displayed to select a snapshot file for upload.
11	Confirm upload with **OK**. **Result**: The uploaded snapshot is displayed in the **<Project> - Snapshots** overview, page 42.

Create an Access Token and Upload a Snapshot with Python Script

Overview

You can upload a snapshot to the EcoStruxure Machine Advisor Code Analysis portal:

- With your personal user account.

  Refer to Create a Snapshot and Upload (with Personal User Account), page 59.
- With a technical user account (access token).

You can create a technical user (access token) that is allowed to upload a snapshot for one specific analysis project. For example, for CI-Systems (Continuous Integration Systems), it is possible to upload snapshots via Python scripts.

Click **ADMINISTRATION > <PROJECT> > ACCESS TOKENS** to open this window.

Create an Access Token

Step	Action
1	Enter a **Name**.
2	Select an **Expiration Date**. After this date, the access token cannot be used to upload a snapshot.
3	Click the **CREATE** button.

Step	Action
	Result: A window is displayed which provides the access token as a character string.
4	Retain this string in a secure manner because the string is provided only once.
5	**Close** the window. **Result**: A new row is displayed in the **<Project> - Access Tokens** table.

Python Script Example

You can use the token (saved character string) to log into the EcoStruxure Machine Advisor Code Analysis portal via a Python script.

> **NOTE:** The **Application** node of the EcoStruxure Machine Expert Logic Builder project which snapshot should be uploaded to the EcoStruxure Machine Advisor Code Analysis portal must provide a **Code Analysis Manager** subnode.

Example of a Python script:

```
# application object searched

app = projects.primary.find("Application", True)[0]

# full analysis will be executed

app.code_analysis.perform_full_analysis()

#login

projects.primary.code_analysis.portal.login
('eyJVcmwiOiJodHRwczovL2Vjb3N0cnV4dXJlLW1hY2hpbmUtYWR2aXNv-
ci1jb2RlYW5h'
+'bHlzaXMuc2UuYXBwIiwiQ2xpZW50SWQiOiJlNWM5aDI4ODdiZDV3YW-
ZiajlpMWw5N2Zl'
+'NDMwZGNqOSIsIkNsaWVudFNlY3JldCI6Ijc2U1VBWkNVU0VEUDExMTExM-
TExIn0='
)
app.code_analysis.portal.snapshots.upload("mySnapshot")

# logout

app.code_analysis.portal.logout()
```

Result: After executing the Python script, the uploaded snapshot is displayed in the **<Project> - Snapshots** overview, page 42.

Connect an Analysis Project to an SVN Repository

Overview

The snapshots that are uploaded to EcoStruxure Machine Advisor Code Analysis do not contain source code. They only contain names of objects, methods, variables, and so on, and information on the relationship between them (call, read, define).

To find the source code modifications between different snapshots, you can link the Machine Advisor Code Analysis to an SVN repository which provides the source code with its history.

Prerequisites

To use the Machine Advisor Code Analysis with an SVN repository, the following software must be installed on the system Machine Advisor Code Analysis is running on:

- Tortoise SVN
- Logic Builder Diff Viewer (see EcoStruxure Machine Expert Logic Builder Diff Viewer, User Guide) (part of the Schneider Electric Software Installer).

  The Logic Builder Diff Viewer must be started at least once so that it can be executed by the EcoStruxure Machine Advisor Code Analysis portal.

Requirements for the Project/Snapshot

To link an analysis project in the Machine Advisor Code Analysis to an SVN repository, the snapshots that should be uploaded to Machine Advisor Code Analysis must provide additional data.

This data is automatically added to the snapshots, if the Machine Advisor Code Analysis the snapshots are created with, has the SVN plugin activated and the analyzed project is checked out from the SVN repository.

Data Automatically Added to the Snapshot

If the requirements are met, the Machine Advisor Code Analysis automatically adds data to the created snapshot:

- The base URI (**Version Control Base URI**) where the project source code can be found in general (refer to **ADMINISTRATION > ALL COMPANY PROJECTS > <Analysis Project > Details** window, page 51).
- A subpath (**Branch in Source Control**), not mandatory, that describes the path, for example to the trunk or the feature branch (refer to **ADMINISTRATION > <PROJECT> > <SNAPSHOT> > Edit Snapshot > Snapshot Details** window, page 42).
- The revision id (**Revision in Source Control**) of the project that the uploaded snapshot represents (refer to **ADMINISTRATION > <PROJECT> > <SNAPSHOT> > Edit Snapshot > Snapshot Details** window, page 42).

Additional Buttons

Based on the automatically added data, the Machine Advisor Code Analysis displays additional buttons (**SVN-CHANGELOG** button, magnifier button) on the **Snapshot Overview** window (refer to **PROJECT SNAPSHOTS > <PROJECT> > OVERVIEW** window, page 15).

With help of these buttons, the Tortoise SVN or the Logic Builder Diff Viewer are started with the appropriate information to find the differences between the revisions of an analyzed project.

Appendices

What’s in This Part

Metrics 64
Conventions 87

Metrics

What's in This Chapter

Metric: Application Size (Code)..64
Metric: Application Size (Code+Data)...65
Metric: Application Size (Data) 65
Metric: Call In...66
Metric: Call Out 66
Metric: Commented Variables (All) Ratio 66
Metric: Commented Variables (In+Out+Global) Ratio 67
Metric: Cyclomatic Complexity...67
Metric: Extended By 69
Metric: Extends 69
Metric: Fan In...70
Metric: Fan Out 70
Metric: Halstead Complexity..71
Metric: Implemented By 74
Metric: Implements...75
Metric: Lines Of Code (LOC) 76
Metric: Memory Size (Data)..76
Metric: Number Of Actions 77
Metric: Number Of GVL Usages 78
Metric: Number Of Header Comment Lines 78
Metric: Number Of Instances..79
Metric: Number Of Library References 80
Metric: Number Of Messages 80
Metric: Number Of Methods 80
Metric: Number Of Multiline Comments 81
Metric: Number Of Properties 81
Metric: Number Of Reads...82
Metric: Number Of Tasks..82
Metric: Number Of Transitions 83
Metric: Number Of Variables 83
Metric: Number Of Writes...84
Metric: Number Of FBD Networks...84
Metric: Source Code Comment Ratio 85
Metric: Stack Size...86

Metric: Application Size (Code)

User Description

When logged into a controller, the source code is compiled and an executable is sent to the controller.

The executable consists of code and data sections when loaded into controller memory.

The application code size is the amount of memory needed on the controller to manage the application code.

> **NOTE:** This metric can only be calculated when you activate the **Consider Code and Data Size** option in the code analyzer **Configuration** tab (see EcoStruxure Machine Expert, Code Analysis, User Guide).

Metric Calculation

The application code size is calculated based on the size of generated machine code for each POU (program, function block, function).

Metric: Application Size (Code+Data)

User Description

When logged into a controller, the source code is compiled (incl. Generate Code) and an executable is sent to the controller.

The executable consists of code and data sections when loaded into controller memory.

The application code and data size is the minimum amount of memory needed on the controller to run the application.

> **NOTE:** This metric can only be calculated when you activate the **Consider Code and Data Size** option in the code analyzer **Configuration** tab (see EcoStruxure Machine Expert, Code Analysis, User Guide).

Metric Calculation

The application code size is calculated based on the size of generated machine code for each POU (program, function block, function).

The application data size is calculated based on the size of the variables in the application.

> **NOTE:** The sum of application code size and application data size (see dedicated metric) is not exactly the application code size + data size (this metric) due to alignment or code page size of the underlaying controller hardware.

Metric: Application Size (Data)

User Description

When logged into a controller, the source code is compiled (incl. Generate Code) and an executable is sent to the controller.

The executable consists of code and data sections when loaded into controller memory.

The application data size is the amount of memory needed on the controller to manage the data needed to execute the application code.

> **NOTE:** This metric can only be calculated when you activate the **Consider Code and Data Size** option in the code analyzer **Configuration** tab (see EcoStruxure Machine Expert, Code Analysis, User Guide).

Metric Calculation

The application data size is calculated based on the size of the variables in the application.

Metric: Call In

User Description

The Call In metric is used to get information about who is calling a method, function, function block, and so on.

Metric Calculation

Each call to an implementation is considered, but if the same object (method, function, etc.) is called twice in the same implementation, it is only counted once.

Example

Call In calculation example:

```
METH();

// Some other implementation

METH();
```

Call In Result (for method *METH*)

```
Call In = 1
```

Metric: Call Out

User Description

The Call Out metric is used to get information about which other objects (method, function, function block, etc.) are called by the implementation.

Metric Calculation

Each call to an implementation is considered, but if the same object (method, function, etc.) is called twice in the same implementation, it is only counted once.

Example

Call Out calculation example:

```
METH();

// Some other implementation

METH();
```

Call Out Result (for method *METH*)

```
Call Out = 1
```

Metric: Commented Variables (All) Ratio

User Description

This metric calculates the ratio (Unit: %) between commented and not commented variables in an object.

Metric Calculation

Each variable declaration in objects (function (FUN), function block (FB), Data Unit Type (DUT), Global Variable List (GVL), and so on) is verified whether it is commented or not.

The ratio between these two values is provided with this metric.

Example

Commented variables ratio calculation example:

```
Declaration:
PROGRAM SR_Main
VAR
    xCheck1: BOOL;//flag to identify
    uiMyVariable2: UINT;
    xFlag: BOOL;
END_VAR
```

Commented variables ratio result: 33.33 %

Metric: Commented Variables (In+Out+Global) Ratio

User Description

This metric calculates the ratio (Unit: %) between commented and not commented variables that are defined in *VAR_GLOBAL*, *VAR_INPUT*, *VAR_OUTPUT*, or *VAR_IN_OUT*.

Metric Calculation

Each variable declaration in objects (function (FUN), function block (FB), Data Unit Type (DUT), Global Variable List (GVL), and so on) are verified whether they are commented or not.

The ratio between these two values is provided with this metric.

Example

Commented variables (in+out+global) ratio calculation example:

```
Declaration:
PROGRAM SR_Main
VAR_IN
    i_xCheck1: BOOL;//flag to identify
    i_uiMyVariable2: UINT;
END_VAR
VAR
    xFlag: BOOL;
END_VAR
```

Commented variables (in+out+global) ratio result: 50 %

Metric: Cyclomatic Complexity

User Description

The Cyclomatic Complexity metric is used to measure the complexity of a program by counting the number of linearly independent paths in the source code.

Metric Calculation

Cyclomatic Complexity is computed using the control flow graph of the program. The complexity depends on the condition and decision points of the control flow graph.

For example:

- No condition or decision point: Complexity = 1 (one path through the code).
- One IF statement: Complexity = 2 (two paths through the code).
- One IF statement with two conditions: Complexity = 3 (three paths through the code).

There are different interpretations/implementation of Cyclomatic Complexity, depending on the analysis tool. Some tools do not consider expressions with AND/OR/etc. in IF, REPEAT, WHILE, etc. statements. The McCabe Cyclomatic Complexity is always increased by +1. Other tools also consider the expressions in the code flow (outside an IF, REPEAT, etc. statement) but later used in an IF or REPEAT statement which results in a higher Cyclomatic Complexity result. The Cyclomatic Complexity implementation in EcoStruxure Machine Expert considers expressions with AND/OR/etc. but does not consider pre-calculated expressions in the code flow or specified in a method call.

Example

Cyclomatic Complexity calculation example:

```
// MCC +1 (Initial Value)

// MCC +0 (Pre calculation of condition not considered)
a := b OR c;

// MCC +0 (Method call with condition not considered in
calling implementation)
METH4(a);

IF a AND b OR c XOR d AND NOT e THEN

   // MCC +5 (IF with 5 conditions)

    CASE i OF
       1..4:
           // MCC +1 (CASE range is considered as one
condition)

           FOR i := 1 TO 10 DO               // MCC +1

               METH1();
           END_FOR

       10, 11, 12, 13:
           // MCC +1 (multiple CASE labels considered as one
condition)

           REPEAT
               // MCC +1 (one condition in REPEAT)

               WHILE (a = TRUE AND b = FALSE) DO
                   // MCC +2 (two conditions in WHILE)

                   METH2();

               END_WHILE

           UNTIL (TRUE) END_REPEAT

        ELSE
            // MCC +0 (Default path through CASE statement)
            METH3();
    END_CASE
END_IF
```

Cyclomatic Complexity Result

```
Cyclomatic Complexity (MCC) = 12
```

Metric: Extended By

User Description

The Extended By metric is used to get information about how often a function block or an interface is extended by another function block or interface.

Metric Calculation

A function block can extend exactly one function block and implement multiple interfaces. An interface can extend multiple interfaces, but cannot implement other interfaces.

A function block or an interface can be extended by none or several interfaces. The number of direct extended interfaces is counted.

Example

Extended By calculation example:

```
FB_Test extends FB_Base implements IF_Test1, IF_Test2

FB_Base implements IF_Test4

IF_Test2 extends IF_Test3, IF_Test5
```

Implemented By Results

```
Extended By (FB_Base) = 1
Extended By (IF_Test3) = 1
Extended By (IF_Test5) = 1
```

Metric: Extends

User Description

The Extends metric is used to get information about how many interfaces are extended by a function block or an interface.

Metric Calculation

A function block can extend exactly one function block and implement multiple interfaces. An interface can extend multiple interfaces, but cannot implement other interfaces.

A function block or an interface can extend none or several interfaces. The number of direct extended interfaces is counted.

Example

Extends calculation example:

```
FB_Test extends FB_Base implements IF_Test1, IF_Test2
```

```
FB_Base implements IF_Test4

IF_Test2 extends IF_Test3, IF_Test5
```

Implemented By Results

```
Extends (FB_Test) = 1
Extends (IF_Test2) = 2
```

Metric: Fan In

User Description

The Fan In metric is used to get information about how many incoming dependencies (reads, writes, calls, and so on) to a node in the analysis data model (Dependency Model) are available. Incoming dependency means, that for example, a node is called and another node depends on this node.

Metric Calculation

Each incoming dependency is considered for a node. A node can be a function block, program, function, variable, library, property, method, task, and so on.

Example

Dependency example (list of dependencies):

```
FunctionBlockA defines MethodA
FunctionBlockA defines MethodB
FunctionBlockA defines VariableC
FunctionBlockA calls MethodA
MethodA calls MethodB
MethodB reads VariableC
```

Fan In Results

```
Fan In (FunctionBlockA) = 0
Fan In (MethodA) = 2
Fan In (MethodB) = 2
Fan In (VariableC) = 2
```

Metric: Fan Out

User Description

The Fan Out metric is used to get information about how many outgoing dependencies (reads, writes, calls, and so on) a node in the analysis data model (Dependency Model) has. Outgoing dependency means, that for example, a node is called and another node depends on this node.

Metric Calculation

Each outgoing dependency is considered for a node. A node can be a function block, program, function, variable, library, property, method, task, and so on.

Example

Dependency example (list of dependencies):

```
FunctionBlockA defines MethodA
FunctionBlockA defines MethodB
FunctionBlockA defines VariableC
FunctionBlockA calls MethodA
MethodA calls MethodB
MethodB reads VariableC
```

Fan Out Results

```
Fan Out (FunctionBlockA) = 4
Fan Out (MethodA) = 1
Fan Out (MethodB) = 1
Fan Out (VariableC) = 0
```

Metric: Halstead Complexity

User Description

The Halstead complexity metric is used to measure the complexity of a software program without running the program itself.

This metric is a static testing method where measurable software properties are identified and evaluated. The source code is analyzed and broken down to a sequence of tokens. The tokens are then classified and counted as operators or operands.

The operators and operands are classified and counted as follows:

Parameter	Description
n1	Number of distinct operators
n2	Number of distinct operands
N1	Total number of operators
N2	Total number of operands

There are a number of metric values that can be calculated to represent different aspects of complexity:

- *Halstead Difficulty (D)*
- *Halstead Length (N)*
- *Halstead CalculatedLength (Nx)*
- *Halstead Volume (V)*
- *Halstead Effort (E)*
- *Halstead Vocabulary (n)*

Halstead Complexity for POUs Implemented in Structured Text (ST)

The Halstead complexity was originally developed for textual languages (like C, C ++, Pascal, etc.) and is applicable to POUs implemented in structured text (ST).

NOTE: By default, the *Halstead Difficulty* is displayed.

Halstead Complexity for POUs Implemented in Function Block Diagram (FBD)

The function block diagram (FBD) belongs to the group of graphical implementation languages and is not text-based. A POU consists of multiple FBD networks. Then the Halstead complexity metric must be adapted to be applicable to graphical languages. Operands and operators and their frequency (per FBD network) are considered as presented to the user (see Example for function block diagram (FBD)).

The Halstead complexity results calculated per FBD network are aggregated across the FBD networks and attached on POU (program, function block, function, method, or property) level.

> **NOTE:** The calculated Halstead values (per FBD network) are *FBD Network Halstead Difficulty* and *FBD Network Halstead Length*.

The following aggregation types are applied per FBD network Halstead metric values (*Halstead Difficulty* and *Halstead Length*):

- Average
- Minimum
- Maximum
- Sum
- Consistency

> **NOTE:** The most relevant aggregated values are *FBD Halstead Difficulty Network Max*, *FBD Halstead Difficulty Network Consistency*, *FBD Halstead Length Network Max*, and *FBD Halstead Length Network Consistency*. All other combinations (**Min**, **Sum**, and **Average**) are calculated and attached to the model but not displayed by default.

Metric Calculation

Value	**Formula**
Halstead Difficulty (D)	D = (n1 / 2) * (N2 / n2)
Halstead Length (N)	N = N1 + N2
Halstead CalculatedLength (Nx)	Nx = n1 * log2(n1) + n2 * log2(n2)
Halstead Volume (V)	V = N * log2(n)
Halstead Effort (E)	E = V * D
Halstead Vocabulary (n)	n = n1 + n2

> **NOTE:** An expression in an *IF <expression> THEN* statement must not have parenthesis. They are considered as always available.

Metric Aggregation

Metric results like *FBD Network Halstead Difficulty* and *FBD Network Halstead Length* are aggregated across the FBD networks of a POU.

The *values* are the list of values of the same metric (for example, *FBD Network Halstead Length)* of the FBD networks of a POU.

The consistency value is a result of the Gini coefficient. The Gini coefficient is a measure of statistical dispersion. It measures the inequality among values of a frequent distribution. A Gini coefficient of 0 expresses equality, where all values are the same. A Gini coefficient of 1 expresses maximum inequality among values.

Example for structured text (ST)

Halstead calculation example (only the implementation part is considered for calculation):

```
IF (xInit = FALSE) THEN

    PerformInitialization();
    xInit := TRUE;
```

```
ELSE

    FOR i := 1 TO 5 DO

        iAxisId := i + 7;
        sAxisName := Standard.CONCAT('MyAxis ', INT_TO_
STRING(iAxisId));

        // Do some math calculations for each axis here
        udiResult := CalculateStuff(sName := sAxisName, iID
:= iAxisId);
    END_FOR

END_IF
```

List of Operators and its Frequencies:

```
Operator                            Frequency
========                            =========
(operators)
If                                          1
Then                                        1
LeftParenthesis                             6
RightParenthesis                            6
Equal                                       1
Semicolon                                   5
Assign                                      7
Else                                        1
For                                         1
EndFor                                      1
Do                                          1
Plus                                        1
Period                                      1
INT_TO_STRING                               1
Colon                                       2
EndIf                                       1
(n1)  16                             (N1)  37
```

List of Operands and its Frequencies:

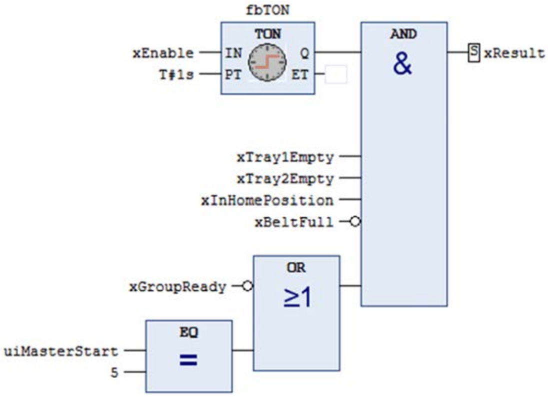
```
Operand                             Frequency
=======                             =========
(variables/methods/functions)
xInit                                       2
PerformInitialization                       1
i                                           2
iAxisId                                     3
sAxisName                                   2
Standard                                    1
CONCAT                                      1
udiResult                                   1
CalculateStuff                              1
sName                                       1
iID                                         1

(constants)
FALSE                                       1
TRUE                                        1
INT#1                                       1
INT#5                                       1
INT#7                                       1
'MyAxis '                                   1
(n2)  17                             (N2)  22
```

Halstead Difficulty Result

```
Halstead Difficulty (D = (16/2) * (22/17) = 10.3529411764706
```

Example for function block diagram (FBD)

Halstead calculation example implemented in FBD (only the implementation part is considered for calculation):



List of Operators and its Frequencies:

```
Operator                            Frequency
========                            =========
(operators)
Assign                                      4
Set2                                        1
And                                         1
Negation2                                   2
Or                                          1
Eq                                          1
(n1)   6                             (N1)  10
```

List of Operands and its Frequencies:

```
Operand                             Frequency
=======                             =========
(variables/methods/functions/constants)
xResult                                     1
TON                                         1
fbTON                                       1
xEnable                                     1
T#1s                                        1
IN                                          1
PT                                          1
Q                                           1
ET                                          1
xTray1Empty                                 1
xTray2Empty                                 1
xInHomePosition                             1
xBeltFull                                   1
xGroupReady                                 1
uiMasterStart                               1
5                                           1
(n2)  16                             (N2)  16
```

FBD Network Halstead Difficulty Result

```
FBD Network Halstead Difficulty (D) = (6/2) * (16/16) = 3
```

FBD Network Halstead Length

```
FBD Network Halstead Length (D) = 10 + 16 = 26
```

Metric: Implemented By

User Description

The Implemented By metric is used to get information about how often an interface is implemented by a function block.

Metric Calculation

A function block can extend exactly one function block and implement multiple interfaces. An interface can extend multiple interfaces, but cannot implement other interfaces.

An interface can be implemented by several function blocks. The number of direct implemented interfaces is counted.

> **NOTE:** If the function block extends another function block or an interface extends other interface, the derived implemented interfaces are not considered. If an interface is implemented in base function block and derived function block, it is counted twice.

Example

Implemented By calculation example:

```
FB_Test extends FB_Base implements IF_Test1, IF_Test2

FB_Base implements IF_Test4, IF_Test1

IF_Test2 extends IF_Test3, IF_Test5
```

Implemented By Results

```
Implemented By (IF_Test1) = 2
Implemented By (IF_Test2) = 1
Implemented By (IF_Test3) = 1
Implemented By (IF_Test4) = 1
Implemented By (IF_Test5) = 1
```

Metric: Implements

User Description

The Implements metric is used to get information about how many interfaces are implemented by a function block.

Metric Calculation

A function block can extend exactly one function block and implement multiple interfaces. An interface can extend multiple interfaces, but cannot implement other interfaces.

A function block can implement none or several interfaces. The number of direct implemented interfaces is counted.

> **NOTE:** If the function block extends another function block or an interface extends other interface, the derived implemented interfaces are not considered.

Example

Implements calculation example:

```
FB_Test extends FB_Base implements IF_Test1, IF_Test2

FB_Base implements IF_Test4

IF_Test2 extends IF_Test3, IF_Test5
```

Implements Results

```
Implements (FB_Test) = 2
Implements (FB_Base) = 1
```

Metric: Lines Of Code (LOC)

User Description

The software metric Lines Of Code (LOC) counts the number of source code lines of a program. This metric can be used to estimate the workload for program development, the programming productivity, and the maintainability of the application.

Metric Calculation

Each line in a textual implemented object (Function (FUN), function block (FB), data unit type (DUT), global variable list (GVL), and so on) is considered in the Lines Of Code metric.

Example

Lines Of Code calculation example:

```
Declaration:
1: PROGRAM SR_Main
2: VAR
3:     x: BOOL;
4: END_VAR

Implementation:
1:
2: IF (x = TRUE) THEN
3:     DoSomething();
4: END_IF
5:
6: // A nice comment
7: SpecialMethod();
```

Lines Of Code Result

```
Lines Of Code (LOC) = 11
```

Metric: Memory Size (Data)

User Description

An application or library is organized by complex types such as programs, function blocks, global variable lists, methods, actions, functions, structures, and so on. Inside each of these types, variables can be defined.

The complex types function blocks and structures can be instantiated multiple times and placed as a block inside the memory.

Each complex type definition (type information and variables), when instantiated, needs a specific amount of memory. Information about how much memory must be allocated and processed, for example in online modification situations or when used as input argument to methods. That information can then be used to identify large complex types which can cause issues when repeatedly instantiated.

Metric Calculation

For a function block or structure, the sizes of the variables are summed up. In addition, the function block type needs memory (list of methods, implemented interfaces, and so on). Based on the underlying controller architecture, memory alignment must be considered too. It is based on the variable type and order.

Example

Memory Size (Data) calculation example:

```
FUNCTION_BLOCK FB_XXX
VAR
    xVar1: BOOL; // 1 bit
    xVar2: BOOL; // 1 bit
    xVar3: BOOL; // 1 bit
    // 5 additional bits for alignment

    iVar4: INT; // 4 byte (on 32-bit architectures)
    xVar5: BOOL; // 1 bit
    // 7 additional bits for alignment

    fbComplex: FB_Test; // 20 byte
END_VAR


FUNCTION_BLOCK FB_YYY
VAREND_VAR

FUNCTION_BLOCK FB_ZZZ
VAR
    iVar1: INT;
END_VAR
```

Memory Size (Data) Results

```
Memory Size (Data) (FB_XXX) = 32
Memory Size (Data) (FB_YYY) = 4
Memory Size (Data) (FB_ZZZ) = 8
```

Metric: Number Of Actions

User Description

The Number of Actions metric is used to get information about how many actions are attached to a program or a function block.

Metric Calculation

Each Action attached to a program or function block is considered. Unused Actions are considered too.

Example

Number Of Action calculation example:

```
FB_MyAlphaModule
  - ACT_InitAxis1 (Action)
  - ACT_InitAxis2 (Action)
  - UpdateStatus (Method)
  - Calculate (Method)
  - Enabled (Property)
     - Get (Property Get)
     - Set (Property Set)
  - Status (Property)
     - Get (Property Get)
     - Set (Property Set)
  - SwitchNextState (Transition)
```

Number Of Actions Result (for variable *FB_MyAlphaModule*)

```
Number Of Actions = 2
```

Metric: Number Of GVL Usages

User Description

The Number Of GVL Usages metric is used to get information about how many global variables a programming object (programs, function blocks, functions, methods, and so on) uses (reading or writing).

Metric Calculation

Each programming object (program, function block, function, method, property get, property set, action,and so on) is handled individually.

Example

Number Of GVL Usages calculation example:

```
FB_MyAlphaModule
VAR
   iMyState : INT;
END_VAR

iMyState := GVL_IOs.G_iState;
if (iMyState = 10) THEN
   ;
END_IF
```

Number Of GVL Usages Result (for variable *FB_MyAlphaModule*)

```
Number Of GVL Usages = 1
```

Metric: Number Of Header Comment Lines

User Description

This metric counts the number of comments in the header of the declaration part.

These comments can help developers to understand what this object is doing, for what it is and how it is working.

Metric Calculation

Each comment line in the header of a declaration part in IEC objects (function (FUN), function block (FB), Data Unit Type (DUT), Global Variable List (GVL), and so on) is counted.

Example

Header comments calculation example:

```
Declaration:
//This PRG is the start point
//Methodes:
// - ....
// - ....
PROGRAM SR_Main
VAR
    x: BOOL;
END_VAR
```

Header comments metric result: 4

Metric: Number Of Instances

User Description

The Number Of Instances metric is used to get information about how often a complex type (function block, enumeration, structure, and so on) is used as variable type on programming objects (programs, function blocks, and so on).

Metric Calculation

Inside the declaration part, you can define variables. Each variable has an associated data type (complex type or elementary type). When used, the instance count of this data type is increased by +1.

> **NOTE:** If the variable data type is an array data type, the underlaying base data type is used and the instance count is handled as +1. The array length is not considered.

> **NOTE:** Instantiation paths through different complex types are not considered. For example, if a function block is instantiated multiple times, the complex types inside are only counted once.

Example

Number Of Instances calculation example:

```
SR_Main
VAR
    fbMyAlphaModule: FB_MyAlphaModule;
END_VAR

FB_MyAlphaModule
VAR
   astAxisStructures: ARRAY [1..10] OF ST_MyAxisStructure;
   fbSubModule: FB_MySubModule;
END_VAR

FB_MySubModule
VAR
    fbAxis: FB_MyAxis;
END_VAR

ST_MyAxisStructure
VAR
   iID: INT;
   fbAxis: FB_MyAxis;
END_VAR

FB_MyAxis
VAR
END_VAR
```

Number Of Instances Results

```
Number Of Instances (FB_MyAlphaModule) = 1
Number Of Instances (FB_MySubModule) = 1
Number Of Instances (ST_MyAxisStructure) = 1
Number Of Instances (FB_MyAxis) = 2
```

Metric: Number Of Library References

User Description

The Number Of Library References metric is used to get information about how many libraries are directly referenced by an application or POU space.

Metric Calculation

Each reference from an application to a library or from a library to another library is considered.

Example

Number Of Library References calculation example:

```
Application
   --> Library A
      --> Library B
   --> Library B
   --> Library C
```

Number Of Library References Result (for the application)

```
Number Of Library References = 3
```

Metric: Number Of Messages

User Description

The Number Of Messages (information, advisory, detected error, unrecoverable error) metric is used to get information about how many messages are emitted to the **Message View** during compilation.

Metric Calculation

The majority of messages are associated to a programming object such as function blocks, functions, and so on. Each message that is not associated to a programming object is attached to the application (or to the POU space for analysis of POU space only).

> **NOTE:** The code analysis is based on a compilable application without compile errors. Messages of category error and unrecoverable error emitted during compilation cannot occur in analysis data.

Metric: Number Of Methods

User Description

The Number of Methods metric is used to get information about how many methods are attached to a program or a function block.

Metric Calculation

Each Method attached to a program or function block is considered. Unused Methods are considered too.

Example

Number Of Methods calculation example:

```
FB_MyAlphaModule
  - ACT_InitAxis1 (Action)
  - ACT_InitAxis2 (Action)
  - UpdateStatus (Method)
  - Calculate (Method)
  - Enabled (Property)
     - Get (Property Get)
     - Set (Property Set)
  - Status (Property)
     - Get (Property Get)
     - Set (Property Set)
  - SwitchNextState (Transition)
```

Number Of Methods Result (for variable *FB_MyAlphaModule*)

```
Number Of Methods = 2
```

Metric: Number Of Multiline Comments

User Description

This metric counts the multiline comments in an object.

Do not use multiline comments because the start and end of such a comment could get lost while merging.

For example, a commented out source code may unintentionally become part of the program again.

Metric Calculation

Example

Multiline comments calculation example:

```
Declaration:
(*This is a multiline
 comment in header*)
PROGRAM SR_Main
VAR
    xCheck1: BOOL;(*not needed
    uiMyVariable2: UINT;*)
    xFlag: BOOL;
END_VAR
```

Multiline comments result: 2

Metric: Number Of Properties

User Description

The Number Of Properties metric is used to get information about how many properties are attached to a program or a function block.

Metric Calculation

Each Property attached to a program or function block is considered. Unused Properties are considered too.

Example

Number Of Properties calculation example:

```
FB_MyAlphaModule
  - ACT_InitAxis1 (Action)
  - ACT_InitAxis2 (Action)
  - UpdateStatus (Method)
  - Calculate (Method)
  - Enabled (Property)
     - Get (Property Get)
     - Set (Property Set)
  - Status (Property)
     - Get (Property Get)
     - Set (Property Set)
  - SwitchNextState (Transition)
```

Number Of Properties Result (for variable *FB_MyAlphaModule*)

```
Number Of Properties = 2
```

Metric: Number Of Reads

User Description

The Number of Reads metric is used to get information about which variables are read.

Metric Calculation

Each read of a variable in an implementation is considered, but if the same variable is read twice in an implementation, it is only counted once.

Example

Number Of Read calculation example:

```
METH(iMyVariable);

// Some other implementation

METH(iMyVariable);
```

Number Of Reads Result (for variable *iMyVariable*)

```
Number Of Reads = 1
```

Metric: Number Of Tasks

User Description

The Number Of Tasks metric is used to get information about how many tasks an application has.

Metric Calculation

Tasks can only be defined in applications. The number of defined tasks of an application is returned.

Example

Number Of Tasks calculation example:

```
Application
  - SR_Main (Program)
  - FB_Test1 (FunctionBlock)
  - FB_Test2 (FunctionBlock)
  - TaskConfiguration
    - TASK_SR_Main (Task)
    - TASK_Visu (Task
```

Number Of Tasks Result (for the application)

```
Number Of Tasks = 2
```

Metric: Number Of Transitions

User Description

The Number Of Transitions metric is used to get information about how many transitions are attached to a program or a function block.

Metric Calculation

Each transition attached to a program or function block is considered. Unused transitions are considered too.

Example

Number Of Transitions calculation example:

```
FB_MyAlphaModule
  - ACT_InitAxis1 (Action)
  - ACT_InitAxis2 (Action)
  - UpdateStatus (Method)
  - Calculate (Method)
  - Enabled (Property)
     - Get (Property Get)
     - Set (Property Set)
  - Status (Property)
     - Get (Property Get)
     - Set (Property Set)
  - SwitchNextState (Transition)
```

Number Of Transitions Result (for variable *FB_MyAlphaModule*)

```
Number Of Properties = 2
```

Metric: Number Of Variables

User Description

The Number Of Variables metric is used to get information about how many variables are defined in the declaration part of programs, function blocks, functions, methods, property *Get* or *Set*, transitions, global variable lists, and so on.

Metric Calculation

Each Variable defined in a declaration part is considered. Unused Variables are considered too.

Example

Number Of Variables calculation example:

```
FB_MyAlphaModule
VAR
   i: INT;
END_VAR
VAR_INPUT
   i_iCommand: INT;
   i_lrPosition: LREAL;
END_VAR
VAR_OUTPUT
   q_iStatus: INT;
END_VAR
```

Number Of Variables Result (for variable *FB_MyAlphaModule*)

```
Number Of Variables = 4
```

Metric: Number Of Writes

User Description

The Number of Writes metric is used to get information about which variables are written.

Metric Calculation

Each writing of a variable in an implementation is considered, but if the same variable is written twice in an implementation, it is only counted once.

Example

Number Of Write calculation example:

```
iMyVariable := 1;

// Some other implementation

iMyVariable := 2;
```

Number Of Writes Result (for variable *iMyVariable*)

```
Number Of Writes = 1
```

Metric: Number Of FBD Networks

User Description

The Number of Function Block Diagram (FBD) Networks metric is used to get information about how many networks are available in an FBD implemented program, function block, function, method, or property.

Metric Calculation

Each FBD network available in a program, function block, function, method, or property is considered.

Example

Number Of FBD Networks calculation example:

```
FB_MyAlphaModule (FB) - implemented in FBD
 - FBD Network 1
 - FBD Network 2
 - FBD Network 3
```

Number Of FBD Network Result (for variable *FB_MyAlphaModule)*

```
Number Of FBD Networks = 3
```

Metric: Source Code Comment Ratio

User Description

Comments can help developers to understand what the code is doing, for what it is and how it is working.

This metric calculates the ratio (Unit: %) between CLOC (Comment Lines Of Code) and SLOC (Source Lines Of Code) of the implementation part of an object.

CLOC: Number of comment lines, including lines that have source code and comment.

SLOC: Number of lines without comments and blank lines, including lines that have source code and comment.

Metric Calculation

Each line in a textual implemented object (function (FUN), function block (FB), Data Unit Type (DUT), program (PRG), and so on) is verified whether it contains a comment or source code.

The ratio between these two values is provided with this metric.

Example

Source code comment ratio calculation example:

```

IF (x = TRUE) THEN
    DoSomething();//This is very important and hard to understand
END_IF

// A nice comment
SpecialMethod();
```

Source code comment ratio result: 50 %

Metric: Stack Size

User Description

An application or library is organized by complex types such as programs, function blocks, global variable lists, methods, actions, functions, structures, and so on. Inside each of these types, variables can be defined.

When a complex type like a function, method, action, property *Get*, property *Set*, or a transition is called, memory on the stack is needed to execute the method. The stack size information can be used to identify the complex type which is using too much memory of the stack.

NOTE: Stack memory that is available per task is limited and defined by the controller used. Large consumption of stack size can result in exceptions.

NOTE: If a function block type is used as method input variable type (call by value), the memory size of the complex type is needed (refer to Memory Size Data, page 76). Do not use call by value for complex types like method or function inputs.

Metric Calculation

For a complex type like function or method, the sizes of the variables are summed up. When the complex type is called, the size is allocated on top of the stack and the input values are copied the allocated memory. During the code execution of the function or method, these memory values are used.

NOTE: Each method or function call has its own memory and does not conflict if a method is called in parallel by another task.

Example

Stack Size calculation example:

```
FUNCTION_BLOCK FB_XXX
VAR
    fbComplex: FB_Test; // 20 byte
END_VAR

// method call of FB_XXX
Meth1(TRUE);

METHOD Meth1
VAR_INPUT
   xTest1: BOOL;
END_VARVAR
   iTest2: INT;
END_VAR


METHOD METH2
VAR_INPUT
    fbComp: FB_XXX;
END_VAR
```

Stack Size Results

```
Stack Size (METH1) = 8
Stack Size (METH2) = 32
```

Conventions

What’s in This Chapter

Convention: Access to Global Variable in FB_Init + FB_Exit.............................87
Convention: Compile Messages 88
Convention: Complex POU With Low Comment Ratio.....................................88
Convention: Complex Type Name Checks..88
Convention: Empty Implementation 89
Convention: Global Variable Accessed Only in One POU 89
Convention: Inheritance Depth Limit 89
Convention: Input Variable Read Check 90
Convention: Input Variable Type Check..91
Convention: Input Variable Write Check 91
Convention: Multiline Comment Usage 92
Convention: No Header Comment 93
Convention: Number of Methods Limit 93
Convention: Number Of Pins Limit (Input/Output)..93
Convention: Number Of Pins Limit (Input)...94
Convention: Number Of Pins Limit (Output) 94
Convention: Number of Properties Limit 94
Convention: Output Variable Read Check 95
Convention: Output Variable Type Check 95
Convention: Persistent Usage Check..96
Convention: Retain Usage Check 96
Convention: Uncommented Variable (All) 97
Convention: Uncommented Variable (In+Out+Global) 97
Convention: Unused Enum Constants Check 97
Convention: Unused Variables Check 98
Convention: Useless DUT 98
Convention: Variable Name Checks..99
Convention: Variable Name Length Check 100

Convention: Access to Global Variable in FB_Init + FB_Exit

User Description

This convention detects a read or write access to a global variable from the *FB_INIT*, *FB_EXIT* or *FB_REINIT* method of a function block.

NOTE: Do not access global variables while executing these methods.

NOTE: Activate the option **Consider Implicit Methods** in the **Configuration** tab of the **Code Analysis Manager** to consider implicit methods during analysis. Refer to Configuration\Consider Implicit Methods.

Example

```
FUNCTION_BLOCK FB_TEST
VAR
END_VAR

METHOD FB_INIT: BOOL
VAR_INPUT
    bInitRetains: BOOL;
    bInCopyCode: BOOL;
END_VAR

GVL.g_iInitState := 3; // Global varible is accessed from
FB_INIT
```

Convention: Compile Messages

User Description

When compiling an application, the compiler reports detected unrecoverable errors, errors, advisories (often referred to as warnings), and information to the developer.

Detected unrecoverable errors and errors must be handled by the developer to get the application to run on a controller.

In addition, advisories are reported to the developer. Advisories should be handled by the developer. The number of advisories should be as close to zero as possible when an application is planned to be released.

Information messages are reported, for example, to inform the developer about the progress or needed memory sizes of the compiled application.

> **NOTE:** A code analysis run can only be made on an application for which no irreconcilable errors are detected. Only compile messages are considered during code analysis. Therefore, error messages are not supported.

Convention Verification Rule

For convention verification, the available compile messages (of the complete analysis data) are considered and reported. The compile message severity is used as severity of the convention violation.

Convention: Complex POU With Low Comment Ratio

User Description

This convention lists all complex objects with low comment ratio.

In contrast to other metrics where only a focus is set to the ratio between CLOC (Comment Lines Of Code) and SLOC (Source Lines Of Code), this convention allows you to filter to objects with high complexity and low comment ratio.

This convention has a focus on the implementation part of an object.

Convention: Complex Type Name Checks

User Description

Coding style is a set of rules or guidelines applied when writing source code. Following a specified coding style helps:

- To read and understand the source code
- To avoid and find programming issues
- To maintain the source code

Based on the Programming Guidelines (Naming Conventions (see EcoStruxure Machine Expert, Programming Guide), Prefixes (see EcoStruxure Machine Expert, Programming Guide)) for source code, complex type name convention queries are available to verify the suggested complex type names per type.

Convention Verification Rule

For convention verification, the complex type is combined with the object name.

Complex type name prefixes based on complex types:

- Function block: *FB_* as prefix
- Program: *SR_* as prefix
- Enumeration: *ET_* as prefix
- Structure: *ST_* as prefix
- etc.

Example

```
FB_MyAxisWrapper
ST_MyDataStruct
ET_MyEnumeration
SR_Main
etc.
```

Convention: Empty Implementation

User Description

This convention detects POUs that are implemented in structured text language and have no source code in the implementation part.

Convention: Global Variable Accessed Only in One POU

User Description

This convention detects the access of a global variable only in one POU.

Locally used variables should only be declared locally.

Convention: Inheritance Depth Limit

User Description

IEC-61131-3 provides language features to extend function blocks or implement interfaces. This is called inheritance and can result in a chain of inheritances. In theory, there is no limit for the inheritance depth, but nesting can become too complex to understand the inheritance tree of interfaces and function blocks.

For the application maintainability reasons, the inheritance depth limit can be verified and reported via convention violation rules.

Convention Verification Rule

The keyword *extends* between function blocks and function blocks, or interfaces and other interfaces is used to verify the chain length.

Example

The inheritance depth of the example is 6.

```
FB_Test1 extends FB_Test2
FB_Test2 extends FB_Test3
FB_Test3 extends FB_Test4
FB_Test4 extends FB_Test5
FB_Test5 extends FB_Test6
FB_Test6 extends FB_Test7
```

Convention: Input Variable Read Check

User Description

In the declaration part of a program, function block, method, or function, input variables can be defined. When objects of this type are called, input values must be specified. When a method or a function is called, the input values are copied to the stack. These values (memory area) are only used by the method or function call.

Compared to programs where exactly one instance exists in memory (or function blocks which are instantiated multiple times in memory), the programs or function blocks can be called multiple times, by multiple tasks, and the same memory location for the input variable is used (in parallel).

For application execution stability, the input variable should not be read from outside the program or function block.

NOTE: Keep in mind that access to an input variable from a method defined below a function block is also considered as access from outside the function block. The convention violation of the input *Variable Read Check* could be a false positive, if the developer verifies, for example, by code review, that the input variable is initialized before using it in a method.

Convention Verification Rule

Each read access to an input variable from outside the function block (body) or program (body) implementation itself is reported as convention violation.

Example

```
SR_Main
VAR
   fbTest: FB_Test;
END_VAR

// call of FB method without calling FB (Body) before
fbTest.Meth();


FB_Test
VAR_INPUT
   i_lrCurrentAxisPosition: LREAL;
END_VAR


METHOD Meth()
VAR
   i_lrVar1: LREAL;
END_VAR
```

```
// potential access to not properly initialized variable
i_lrVar1 := i_lrCurrentAxisPosition;
```

Convention: Input Variable Type Check

User Description

In the declaration part of a POU, input variables can be defined. When this POU is called, input values must be specified.

These input values are copied by value (memory copy).

For application execution stability, the input variable should not be of type function block.

Convention Verification Rule

Each input variable of type function block is reported as convention violation.

Example

```
SR_Main
VAR
   fbTest: FB_Test;
   fbArg: FB_MyArg;
END_VAR

// call of FB method without calling FB (Body) before
fbTest(i_fbMyArg := fbArg);

FB_MyArg
VAR_INPUT
END_VAR

FB_Test
VAR_INPUT
   i_fbMyArg: FB_MyArg;
END_VAR
```

Convention: Input Variable Write Check

User Description

In the declaration part of a program, function block, method, or function, input variables can be defined. When objects of this type are called, input values must be specified. When a method or a function is called, the input values are copied to the stack. These values (memory area) are only used by the method or function call.

Compared to programs where exactly one instance exists in memory (or function blocks which are instantiated multiple times in memory), the programs or function blocks can be called multiple times, by multiple tasks, and the same memory location for the input variable is used (in parallel).

For application execution stability, the input variable should only be written by the caller (from outside) the program or function block.

NOTE: Variables of category input means that data is transferred to that construct, be it a program, function block, method, or function. Results of the call are of category output. Input variables should be restricted to those passed to the construct, and input variables of the construct should not be read from outside the construct nor written to from inside the construct.

Convention Verification Rule

Each write access to an input variable from inside the function block is reported as convention violation.

Example

```
SR_Main
VAR
   fbTest: FB_Test;
   xTest: BOOL;
END_VAR

// call of FB method without calling FB (Body) before
fbTest(TRUE);

//potential violation reading outside the construct
xTest := fbTest.i_xEnable ;

FB_Test
VAR_INPUT
   i_xEnable: BOOL;
END_VAR

// potential violation. Now the input value has changed its
value.
i_xEnable := FALSE;
```

Convention: Multiline Comment Usage

User Description

This convention verifies if multiline comments are used in objects.

Do not use multiline comments because the start and end of such a comment could get lost while merging.

For example, a commented out code may unintentionally become part of the program again.

Example

Multiline comments calculation example.

Declaration:

```
(*This is a multiline
comment in header*)
PROGRAM SR_Main
VAR
    xCheck1: BOOL;(*not needed
    uiMyVariable2: UINT;*)
    xFlag: BOOL;
```

```
8: END_VAR
```

Convention: No Header Comment

User Description

Many coding style guides suggest that a general description about what a POU is doing and how it is working be present in the header of the declaration part.

Example

All variants of comments are counted for this verification:

```
- //my comment
- ///my doc comment
- (*my multi line comment*)
```

Convention: Number of Methods Limit

User Description

For maintainability reasons of applications, there are design principles available how to organize your code. For example, only one job per function block or per method.

Applying code design rules can help to detect that, for example, too many methods are attached to one function block and is an indicator to split the function block itself into code pieces.

Convention Verification Rule

The number of methods attached to a function block or program is used to verify whether the limit is exceeded.

Convention: Number Of Pins Limit (Input/Output)

User Description

The number of input/output variables (*VAR_IN_OUT*) should be within a limited range set by your organization. Refer to your coding style guides.

Using a function block with too many pins affects readability (a limit could be around 10 for graphical programming languages).

If more input, output, or input/output variables are required, consider reducing the number of pins (input, output, input/output variables) of POUs by introducing a structure to group some input, output, or input/output variables.

Convention: Number Of Pins Limit (Input)

User Description

The number of input variables (*VAR_INPUT*) should be within a limited range set by your organization. Refer to your coding style guides.

Using a function block with too many pins affects readability (a limit could be around 10 for graphical programming languages).

If more input, output, or input/output variables are required, consider reducing the number of pins (input, output, input/output variables) of POUs by introducing a structure to group some input, output, or input/output variables.

Convention: Number Of Pins Limit (Output)

User Description

The number of output variables (*VAR_OUTPUT*) should be within a limited range set by your organization. Refer to your coding style guides.

Using a function block with too many pins affects readability (a limit could be around 10 for graphical programming languages).

If more input, output, or input/output variables are required, consider reducing the number of pins (input, output, input/output variables) of POUs by introducing a structure to group some input, output, or input/output variables.

Convention: Number of Properties Limit

User Description

IEC-61131-3 provides language features to organize an application in programs, function blocks, and Global Variable Lists (GVL). To reduce complexity and support object orientation, properties can be attached. Each property provides functional access to the information behind.

Too many attached properties:

- Are not easy to handle or not easy to understand by the developer.
- Can result in naming conflicts.
- Can be an indicator that a program or function block realizes multiple jobs.

The convention result can be used as an indicator to split a program/function block into several programs/function blocks, each with one job only.

Thus the maintainability of applications can be improved.

Convention Verification Rule

The number of properties attached to a function block, program, or Global Variable List (GVL) is used to verify whether the limit is exceeded.

Convention: Output Variable Read Check

User Description

In the declaration part of a program, function block, method, or function, output variables can be defined. When objects of this type are called, output value targets can be specified. When a method or a function is called, the output values are copied to the stack. These values (memory area) are only assigned by the method or function call and reused by its caller.

Compared to programs where exactly one instance exists in memory (or function blocks which are instantiated multiple times in memory), the programs or function blocks can be called multiple times, by multiple tasks, and the same memory location for the input variable is used (in parallel).

For application execution stability, the output variable should only be read by the caller (from outside) the program or function block.

> **NOTE:** Reading an output variable from within an implementation could be false positive, if the developer verifies, for example, by code review, that the output variable is written before using it later in the code.

Convention Verification Rule

Each read access of an output variable from inside the function block is reported as convention violation because it cannot be verified that a proper value was assigned before.

Example

```
SR_Main
VAR
   xResult: BOOL;
   fbTest: FB_Test;
END_VAR


// call of FB method without calling FB (Body) before
fbTest(q_xEnable => xResult);


FB_Test
VAR_OUTPUT
   q_xEnable: BOOL;
END_VAR


// potential violation. Now the input value has changed its
value.
IF (q_xEnable) THEN
     ; // Violation. Unclear value of output variable.
END_IF
```

Convention: Output Variable Type Check

User Description

In the declaration part of a POU, output variables can be defined. When this POU is called, output values can be assigned.

These output values are copied by value (memory copy).

For application execution stability, the output variable should not be of type function block.

Convention Verification Rule

Each output variable of type function block is reported as convention violation.

Example

```
SR_Main
VAR
    fbTest: FB_Test;
    fbArg: FB_MyArg;
END_VAR

// call of FB method without calling FB (Body) before
fbTest(q_fbMyArg => fbArg);

FB_MyArg
VAR_INPUT
END_VAR

FB_Test
VAR_OUTPUT
    q_fbMyArg: FB_MyArg;
END_VAR
```

Convention: Persistent Usage Check

User Description

If you declare a variable as PERSISTENT in a function block, then the entire instance of this function block is saved in the persistent range (all data of the block), but only the declared PERSISTENT variable is restored.

This increased memory consumption and additional handling of persistent variables can cause performance issues.

> **NOTE:** The compiler treats a VAR PERSISTENT declaration just like a VAR PERSISTENT RETAIN or VAR RETAIN PERSISTENT declaration.

Convention: Retain Usage Check

User Description

If you declare a variable as RETAIN in a function block, then the entire instance of this function block is saved in the retain range (all data of the block), but only the declared RETAIN variable is restored.

This increased memory consumption and additional handling of retain variables can cause performance issues.

Convention: Uncommented Variable (All)

User Description

This convention verifies whether uncommented variables exist in an object.

Example

Declaration:

```
PROGRAM SR_Main
VAR
    xCheck1: BOOL;//flag to identify
    uiMyVariable2: UINT;
    xFlag: BOOL;
END_VAR
```

uiMyVariable2 and *xFlag* are not commented. Therefore a convention violation is created.

Convention: Uncommented Variable (In+Out+Global)

User Description

This convention verifies whether uncommented variables defined in *VAR_GLOBAL*, *VAR_INPUT*, *VAR_OUTPUT*, or *VAR_IN_OUT* exist in source code.

Example

Declaration:

```
PROGRAM SR_Main
VAR_IN
    i_xCheck1: BOOL;//flag to identify
    i_uiMyVariable2: UINT;
END_VAR
VAR
    xFlag: BOOL;
END_VAR
```

i_uiMyVariable2 is not commented. Therefore a convention violation is created.

Convention: Unused Enum Constants Check

User Description

This convention detects enumeration constants that are not used in your application.

Not used enumeration constants, for example, in state machines, may indicate that they are not used properly or are incomplete, or that outdated source code is present.

Convention: Unused Variables Check

User Description

In the declaration of a program, function block, method, or function variables can be defined. Normally, these variables are used inside the code. If a variable is defined but not used (read or written) in the code, it consumes memory.

Convention Verification Rule

Each variable without a read or write access is reported as convention violation.

Example

```
SR_Main
VAR
   xMyUnusedVariableResult: BOOL;
   fbTest: FB_Test;
END_VAR

// call of FB method but output is not assigned to intended
result variable.
fbTest(q_xEnable => );
```

Convention: Useless DUT

User Description

This convention detects DUTs (Data Unit Types) that consist only of one element.

This convention violation can indicate an incomplete refactoring activity or a feature that has not been completed.

Example

```
TYPE UT_MyUnion :
UNION
    xInit : BOOL; //only one element in union
END_UNION
END_TYPE

TYPE ST_MyStruct :
STRUCT
    xInit : BOOL; //only one element in struct
END_STRUCT
END_TYPE

TYPE ET_MyEnum :
(
    State := 1 //only one element in enum
);
END_TYPE
```

Convention: Variable Name Checks

User Description

Coding style is a set of rules or guidelines applied when writing source code. Following a specified coding style helps:

- To read and understand the source code
- To avoid and find programming issues
- To maintain the source code

Based on the Programming Guidelines (Naming Conventions (see EcoStruxure Machine Expert, Programming Guide), Prefixes (see EcoStruxure Machine Expert, Programming Guide)) for source code, variable name convention queries are available to verify the suggested variable name per data type and variable scope.

Convention Verification Rule

For convention verification, the variable name is combined with its linked data type and the scope where the variable is defined in.

Scopes:

- Local variable scope: No special scope prefix (*VAR* ... *END_VAR*)
- Input variable scope: *i_* as prefix (*VAR_INPUT* ... *END_VAR*)
- Output variable scope: *q_* as prefix (*VAR_OUTPUT* ... *END_VAR*)
- In-/Output variable scope: *iq_* as prefix (*VAR_IN_OUT_* ... *END_VAR*)
- Global variable scope: *G_* as prefix
- Global constants scope: *Gc_* as prefix
- etc.

Variable name prefixes based on data type:

- INT: *i* as prefix
- DINT: *di* as prefix
- UDINT: *udi* as prefix
- REAL: *r* as prefix
- LREAL: *lr* as prefix
- Function block: *fb* as prefix
- POINTER TO: *p* as prefix
- etc.

Example

```
VAR
    iMyVariable1: INT;
    uiMyVariable1: UINT;
    rMyVariable1: REAL;
    piMyVariable7: POINTER TO INT;
END_VAR
VAR_INPUT
    i_iMyVariable2: INT;
    i_uiMyVariable2: UINT;
    i_rMyVariable2: REAL;
END_VAR
VAR_IN_OUT
    iq_iMyVariable3: INT;
    iq_uiMyVariable3: UINT;
    iq_rMyVariable3: REAL;
END_VAR
VAR_OUTPUT
    iq_iMyVariable2: INT;
    iq_uiMyVariable2: UINT;
    iq_rMyVariable2: REAL;
END_VAR
```

Convention: Variable Name Length Check

User Description

Coding style is a set of rules or guidelines applied when writing source code. Following a specified coding style helps:

- To read and understand the source code
- To avoid and find programming issues
- To maintain the source code

For readability reasons, there are suggestions for variable names and length. The length of variables can be verified against a user-defined limit.

Convention Verification Rule

The length of a variable name is compared with a length threshold.

Example

```
iMyVariable                                 // length 11 -->
OK
iSpecialAndVeryLongAndHardToReadVariable    // length 20 -->
Not OK
```

Index

A

access rights
  Machine Advisor Code Analysis 53
access to global variable in FB_Init and FB_Exit
  conventions 87
ACCESS TOKENS (administration)
  Machine Advisor Code Analysis 48
ADMINISTRATION
  Machine Advisor Code Analysis 42
ALL COMPANY PROJECTS (administration)
  Machine Advisor Code Analysis 51
ALL MY PROJECTS
  Machine Advisor Code Analysis 26
ALL MY PROJECTS (administration)
  Machine Advisor Code Analysis 50
ALL MY PROJECTS (trends)
  Machine Advisor Code Analysis 36
ALL QUERY CHAINS (administration)
  Machine Advisor Code Analysis 50
application size (code)
  metrics 64
application size (code+data)
  metrics 65
application size (data)
  metrics 65
assign users
  Machine Advisor Code Analysis 58

C

call in
  metrics 66
call out
  metrics 66
CODE QUALITY REPORT (project snapshots)
  Machine Advisor Code Analysis 25
commented variables (all) ratio
  metrics 66
commented variables (in+out+global) ratio
  metrics 67
company settings
  Machine Advisor Code Analysis 55
COMPARE PROJECTS
  Machine Advisor Code Analysis 37
compile messages
  conventions 88
complex POU with low comment ratio
  conventions 88
complex type name checks
  conventions 88
components
  Machine Advisor Code Analysis 12
connect analysis project to SVN repository
  Machine Advisor Code Analysis 61
conventions
  access to global variable in FB_Init and FB_Exit 87
  compile messages 88
  complex POU with low comment ratio 88
  complex type name checks 88
  empty implementation 89
  global variable accessed only in one POU 89
  inheritance depth limit 89
  input variable read check 90
  input variable type check 91
  input variable write check 91
  Machine Advisor Code Analysis 87
  multiline comment usage 92
  no header comment 93
  number of methods limit 93
  number of pins limit (input) 94
  number of pins limit (input/output) 93
  number of pins limit (output) 94
  number of properties limit 94
  output variable read check 95
  output variable type check 95
  persistent usage check 96
  retain usage check 96
  uncommented variable (all) 97
  uncommented variable (in+out+global) 97
  unused enum constants check 97
  unused variables check 98
  useless DUT 98
  variable name checks 99
  variable name length check 100
CONVENTIONS ARCHIVE & EXPORT
  Machine Advisor Code Analysis 24
CONVENTIONS ARCHIVE & EXPORT (project trends)
  Machine Advisor Code Analysis 34
CONVENTIONS DASHBOARD
  Machine Advisor Code Analysis 18
CONVENTIONS DETAILS VIEW (project trends)
  Machine Advisor Code Analysis 30
create a snapshot
  Machine Advisor Code Analysis 59
create an access token
  Machine Advisor Code Analysis 60
create analysis project
  Machine Advisor Code Analysis 58
create company account
  Machine Advisor Code Analysis 57
cyclomatic complexity
  metrics 67

D

DASHBOARD (compare projects)
  Machine Advisor Code Analysis 37
DASHBOARD (project trends)
  Machine Advisor Code Analysis 28
DETAILS VIEW (compare projects)
  Machine Advisor Code Analysis 38
DETAILS VIEW (snapshots)
  Machine Advisor Code Analysis 19
DEVELOPMENT PHASES (administration)
  Machine Advisor Code Analysis 47

E

empty implementation
  conventions 89
extended by
  metrics 69
extends
  metrics 69

F

fan in
  metrics 70
fan out
  metrics 70

G

general information
Machine Advisor Code Analysis 11
global variable accessed only in one POU
conventions 89

H

Halstead complexity
metrics 71
HOME
Machine Advisor Code Analysis 13
how to use
Machine Advisor Code Analysis 57

I

implemented by
metrics 74
implements
metrics 75
inheritance depth limit
conventions 89
input variable read check
conventions 90
input variable type check
conventions 91
input variable write check
conventions 91
invite users
Machine Advisor Code Analysis 57

L

lines of code
metrics 76

M

Machine Advisor Code Analysis
access rights 53
ACCESS TOKENS (administration) 48
ADMINISTRATION 42
ALL COMPANY PROJECTS (administration) 51
ALL MY PROJECTS 26
ALL MY PROJECTS (administration) 50
ALL MY PROJECTS (trends) 36
ALL QUERY CHAINS (administration) 50
assign users 58
CODE QUALITY REPORT (project snapshots) 25
COMPARE PROJECTS 37
components 12
connect analysis project to SVN repository 61
conventions 87
CONVENTIONS ARCHIVE & EXPORT 24
CONVENTIONS ARCHIVE & EXPORT (project trends) 34
CONVENTIONS DASHBOARD 18
CONVENTIONS DETAILS VIEW (project trends) 30
create a snapshot 59
create an access token 60
create analysis project 58
create company account 57
DASHBOARD (compare projects) 37
DASHBOARD (project trends) 28
DETAILS VIEW (compare projects) 38
DETAILS VIEW (snapshots) 19
DEVELOPMENT PHASES (administration) 47
general information 11
HOME 13
how to use 57
invite users 57
metrics 64
METRICS ARCHIVE & EXPORT 22
METRICS ARCHIVE & EXPORT (compare projects) 40
METRICS ARCHIVE & EXPORT (project trends) 32
METRICS DASHBOARD 17
METRICS DETAILS VIEW (project trends) 29
OVERVIEW (snapshots) 15
PROCESSING JOBS (administration) 44
PROJECT SNAPSHOTS 15
PROJECT TRENDS 28
PURCHASES (administration) 55
RULE SET (administration) 45
SNAPSHOTS (administration) 42
upload snapshot 59
upload with python script 60
user and company settings 55
USER AND USER ROLES (administration) 53
memory size
metrics 76
metrics
application size (code) 64
application size (code+data) 65
application size (data) 65
call in 66
call out 66
commented variables (all) ratio 66
commented variables (in+out+global) ratio 67
cyclomatic complexity 67
extended by 69
extends 69
fan in 70
fan out 70
Halstead complexity 71
implemented by 74
implements 75
lines of code 76
Machine Advisor Code Analysis 64
memory size 76
number of actions 77
number of FBD networks 84
number of GVL usages 78
number of header comment lines 78
number of instances 79
number of library references 80
number of messages 80
number of methods 80
number of multiline comments 81
number of properties 81
number of reads 82
number of tasks 82
number of transitions 83
number of variables 83
number of writes 84
source code comment ratio 85
stack size 86
METRICS ARCHIVE & EXPORT
Machine Advisor Code Analysis 22
METRICS ARCHIVE & EXPORT (compare projects)
Machine Advisor Code Analysis 40
METRICS ARCHIVE & EXPORT (project trends)

Machine Advisor Code Analysis 32
METRICS DASHBOARD
Machine Advisor Code Analysis 17
METRICS DETAILS VIEW (project trends)
Machine Advisor Code Analysis 29
multiline comment usage
conventions 92

N

no header comment
conventions 93
number of actions
metrics 77
number of FBD networks
metrics 84
number of GVL usages
metrics 78
number of header comment lines
metrics 78
number of instances
metrics 79
number of library references
metrics 80
number of messages
metrics 80
number of methods
metrics 80
number of methods limit
conventions 93
number of multiline comments
metrics 81
number of pins limit (input)
conventions 94
number of pins limit (input/output)
conventions 93
number of pins limit (output)
conventions 94
number of properties
metrics 81
number of properties limit
conventions 94
number of reads
metrics 82
number of tasks
metrics 82
number of transitions
metrics 83
number of variables
metrics 83
number of writes
metrics 84

O

output variable read check
conventions 95
output variable type check
conventions 95
OVERVIEW (snapshots)
Machine Advisor Code Analysis 15

P

persistent usage check
conventions 96
PROCESSING JOBS (administration)
Machine Advisor Code Analysis 44
PROJECT SNAPSHOTS
Machine Advisor Code Analysis 15
PROJECT TRENDS
Machine Advisor Code Analysis 28
PURCHASES (administration)
Machine Advisor Code Analysis 55

R

retain usage check
conventions 96
RULE SET (administration)
Machine Advisor Code Analysis 45

S

SNAPSHOTS (administration)
Machine Advisor Code Analysis 42
source code comment ratio
metrics 85
stack size
metrics 86

U

uncommented variable (all)
conventions 97
uncommented variable (in+out+global)
conventions 97
unused enum constants check
conventions 97
unused variables check
conventions 98
upload snapshot
Machine Advisor Code Analysis 59
upload with python script
Machine Advisor Code Analysis 60
useless DUT
conventions 98
USER AND USER ROLES (administration)
Machine Advisor Code Analysis 53
user settings
Machine Advisor Code Analysis 55

V

variable name checks
conventions 99
variable name length check
conventions 100

Schneider Electric
35 rue Joseph Monier
92500 Rueil Malmaison
France

+ 33 (0) 1 41 29 70 00

www.se.com

As standards, specifications, and design change from time to time, please ask for confirmation of the information given in this publication.