SmartDelta

Automated Quality Assurance and Optimization in Incremental Industrial Software Systems Development

D3.5 Delta-oriented Quality Assurance Toolset

Submission date of deliverable: January 31, 2025
Edited by: Eduard Enoiu (MDU) and SmartDelta WP3 partners

Project start date	Dec 1, 2021
Project duration	36 months
Project coordinator	Dr. Mehrdad Saadatmand, RISE Research Institutes of Sweden
Project number & call	20023 - ITEA 3 Call 7
Project website	https://itea4.org/project/smartdelta.html & https://smartdelta.org/

Contributing partners	WP3 partners
Version number	V1.0
Work package	WP3
Work package leader	Eduard Paul Enoiu, MDU
Dissemination level	Public

Description	The D3.5 Software Delta-oriented Quality Assurance Toolset presents the final version of the SmartDelta software quality assurance solutions. The toolset adopts a delta-oriented approach, enabling developers and quality assurance professionals to use the SmartDelta toolset

Executive Summary

The D3.5 Delta-oriented Quality Assurance Toolset final deliverable includes the tools made and improved within the SmartDelta project, focusing on improving quality assurance for incremental industrial software systems. This document presents a suite of tools tailored to address challenges in incremental software development, including requirements analysis, regression testing, technical debt management, and anomaly detection.

The deliverable includes brief descriptions of the quality assurance tools and readiness levels. Access information is provided, distinguishing between open-source tools, available through public repositories, and closed-source tools requiring permission. Contact details and support channels for each tool are outlined to facilitate user engagement and support. Additionally, the document shows some training and feedback mechanisms, offering user guides and tutorials to aid in tool adoption. This deliverable serves as a resource for software developers and quality assurance professionals interested in the delta-oriented quality assurance toolset.

Table of Contents

1. Access Information

The SmartDelta quality assurance tools are hosted across a combination of open-access and restricted repositories, reflecting their respective classifications into open-source and closed-source categories. Open-source tools are available through individual repositories, offering access to source code, documentation, and usage guidelines. In comparison, closed-source tools are proprietary and require access permissions. These tools are typically managed through secure, restricted repositories and are shared only with authorized partners and collaborators upon request. Access is granted through formal channels, ensuring that proper agreements and usage guidelines are adhered to. Open-source tools, such as NALABS, MUT4SLX, SoHist, and VARA+, are available through public repositories like GitHub, providing unrestricted access to source code, documentation, and tutorials. Closed-source tools are kept in secure repositories requiring authorization. Access to these tools is managed through requests directed to specific contacts associated with each tool or the SmartDelta coordination team.

2. SmartDelta Quality Assurance Tools Overview

The SmartDelta WP3 toolset features a wide array of quality assurance tools addressing specific challenges in incremental software development, from requirement validation to regression testing. Table 1 provides a detailed overview of these tools.

Table 1. An Overview of the tools included in the final WP3 toolset.

Tool Name	Type	TRL	Input Format	Output Format	SmartDelta Contribution	Key Partners
Requirement and Test Specification (NALABS)	Open-Source	5	Excel, CSV	Metrics Reports	Requirement ambiguity detection	MDU, RISE
Discovery of Microservice Anomalies (DIA4M)	Open-Source	4	Logs, ElasticSearch Data	Anomaly Reports	Anomaly detection in microservices	NetRD
Model-Based Test Case Generation (IFAK-TCG)	Proprietary	5	UML, Petri Nets	Test Cases	Model-based test generation	IFAK
Test Amplification (FOKUS-CBTS)	Proprietary	4	Code Changes	Amplified Test Suites	Regression test amplification	FOKUS
Variability Analysis (VARA+)	Open-Source	5	Text, CSV	Recommendations	Requirements reuse management	RISE, MDU, Alstom
Historical Quality Analysis (SoHist)	Open-Source	3	Git Repositories	Historical Metrics	Retrospective code quality assessment	UIBK
Software Structure Analysis (SONATA)	Proprietary	4	Code Bases	Knowledge Graphs	Test recommendations via knowledge graphs	Izertis

UI Test Generator	Proprietary	5	Java or React code of the UI	Java or TypeScript Automated Tests	UI-focused test generation	Vaadin
Mutation Testing (MUT4SLX)	Open-Source	3	Simulink Models	Coverage Reports	Simulink model fault evaluation	Antwerp
Static Analysis of Block-based programs (DRACONIS)	Open-Source	4	PLC Simulink	Validation Reports	Delta-aware regression testing	MDU, Alstom
Test Effort Estimation	Proprietary	3	Historical Metrics	Effort Predictions	Effort estimation for testing activities	MDU, Alstom
Python Static Analysis (PyLC)	Open-Source	4	Python Code	Analysis Reports	Python industrial control systems testing	MDU, Alstom
Formal Verification (GW2UPPAAL)	Open-Source	3	Graph Models	Test Cases	Formal verification using UPPAAL	MDU
PLC-EARS	Proprietary	4	PLC Requirements	Validation Reports	PLC requirement validation	MDU, Alstom
Cloud Test Monitoring (Jazure)	Proprietary	9	Java Applications	Cloud Test Logs	Cloud-based test monitoring	Arcelik, Vaadin
Code Smell Detection (Smellyzer)	Closed-Source	9	Code Bases	Refactoring Suggestions	Code maintainability improvement	Bilkent University
Traceability Mapping (Relink)	Proprietary	9	System Changes	Traceability Reports	Mapping requirements and tests	Bilkent University
Telemetry Analysis (TAA)	Proprietary	4	Telemetry Data	Anomaly Reports	Early detection of system issues	Hoxhunt

Each tool in the WP3 toolset is designed to address a specific challenge within the complex domain of incremental software development. These tools focus on a range of areas, such as requirement validation, ambiguity detection, regression testing, and anomaly analysis.

3. SmartDelta Quality Assurance Tools

By concentrating on tasks like model-based test generation, variability analysis, historical quality assessment, and formal verification, the WP3 tools provide targeted solutions that improve efficiency and effectiveness in the software development process. Each tool plays a role in supporting various stages of the software lifecycle.

3.1. Requirement and Test Specification Static Checker (NALABS)

NALABS identifies flaws in natural language requirements using automated metrics like readability and ambiguity detection.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 5 – Prototype
- CRL: Early adoption phase.

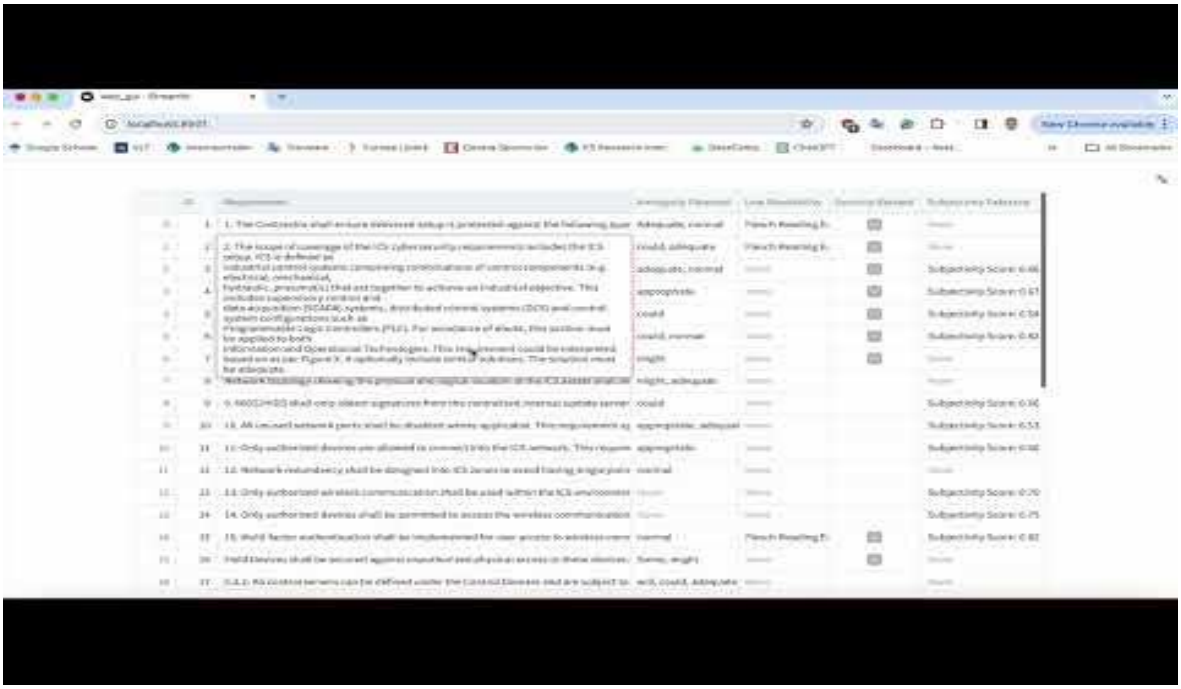
Main Partner and Owner
Mälardalen University (MDU). eduard.paul.enoiu@mdu.se and jean.malm@mdu.se. Issues and features requests to be reported via https://github.com/eduardenoiu/NALABS/issues

Key Features
- Detects ambiguous and vague requirements.
- Provides quality metrics like ARI and word count.
- Works with Excel and CSV formats.

Collaboration Partners
- Alstom
- RISE

Webpage and Documentation
- NALABS Repository

Training Material
Guides and usage tutorials are available in the repository.
https://youtu.be/bvS1BWIpn04?si=ZnueiECUI-M34_Qd



Licensing and Intellectual Property
Open Source.

3.2. Discovery of Interactions and Anomalies for Microservices (DIA4M)

DIA4M is designed to monitor microservice-based platforms, map interactions, and detect anomalies.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 4 – Proof of Concept
- CRL: Tested in microservice environments.

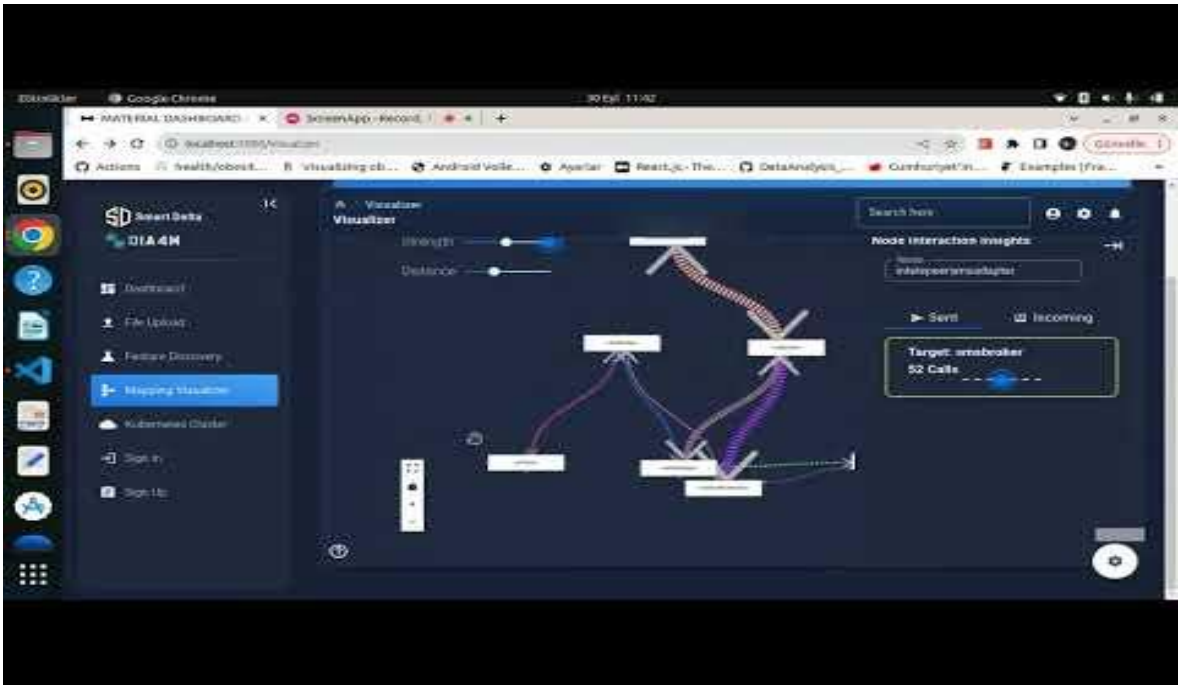
Main Partner and Owner
NetRD (Orion Innovation). hakan.kilinc@orioninc.com

Key Features
- Visual interaction mapping of microservices.
- Anomaly detection using machine learning.
- Integrated with ElasticSearch.

Collaboration Partners
- CPaaS environments and cloud platforms.

Webpage and Documentation
Toolset: https://github.com/smartdeltanetrd

Training Material
The training material shows a guide with detailed steps, an interactive concept and an explanation or demonstration:
- https://github.com/smartdeltanetrd/smartdelta-setup
- https://astro-eren.vercel.app/work/nested/duvet-genius
- https://www.youtube.com/watch?v=Bz_X0XcyY18



Detailed documentation is provided upon request.

Licensing and Intellectual Property
Open Source.

3.3. Model-Based Test Case Generation (IFAK-TCG)

Generates tests systematically from specification models like UML state machines and Petri nets.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 5 – Prototype
- CRL: Not yet commercially deployed.

Main Partner and Owner

IFAK. robin.groepler@ifak.eu

Key Features

- Supports Petri net unfoldings for test case generation.
- Integrates with model-based development environments.

Webpage and Documentation

Detailed documentation for IFAK-TCG is available upon request, providing guidance on features, integration, and usage within model-based development environments.

Training Material

Training materials to support the use of IFAK-TCG are available upon request, providing resources to enhance understanding and application.

Licensing and Intellectual Property

Proprietary with academic use rights.

3.4. Test Amplification (FOKUS-CBTS)

Amplifies test suites by automatically generating additional test cases based on code changes.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 4 – Prototype
- CRL: Tested in Vaadin CI pipelines.

Main Partner and Owner

Fraunhofer FOKUS. marc-florian.wendland@fokus.fraunhofer.de

Key Features

- AST-based change impact analysis.
- Supports regression test selection.

Collaboration Partners

- Vaadin

Webpage and Documentation

Documentation for FOKUS-CBTS is available internally and can be requested as needed for project-related purposes.

Training Material
Training materials tailored to project-specific needs are available upon request as well as guidance on using the tool for improving test suites based on code changes.

Licensing and Intellectual Property
Proprietary.

3.5. VARA+: Variability-Aware Requirements Management and Analysis

VARA+ recommends reuse of software based on requirements. It also identifies, analyse and allocate requirements using natural language models.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 5 – Prototype
- CRL: Pilot testing in the railway industry.

Main Partner and Owner
RISE. muhammad.abbas@ri.se, issues and features requests to be reported via email.

Key Features

- Requirement reuse recommendations.
- Smart requirements allocation to teams.
- Integration with NALABS for quality checks.

Collaboration Partners

- Alstom, MDU

Webpage and Documentation

- Requirements Identification: Available at https://github.com/a66as/REFSQ2023-ReqORNot
- Requirements Quality: Part of the toolchain.
- Smart Requirements Allocation: Part of the toolchain
- Reuse: Part of the toolchain
- Toolchain: Not publicly available and partially implemented: https://github.com/a66as/Vara-Tool

Training Material
Installation guides and tutorials are available internally and can be provided upon request.

Licensing and Intellectual Property
Proprietary, partially open-source.

3.6. SoHist

SoHist is a historical quality evolution tool that extends the capabilities of SonarQube to analyze and visualize the quality evolution of software projects over time. SoHist enables retrospective code analysis by providing historical insights into code quality metrics such as technical debt, bugs, and code smells. Simplified, it works by connecting to Git repositories and running SonarQube analysis several times.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)
TRL: 3 - Prototype
CRL: Not yet commercially deployed.

Main Partner and Owner
University of Innsbruck (UIBK). benedikt.dornauer@uibk.ac.at

Key Features
- Integrates with Dockerized environments for ease of deployment.
- Filters results by time range, contributors, or branches.
- Provides different types of visualization of retrospective analysis of code quality metrics.
- Utilizes 2007 analyzed past projects to simplify the comparison of the currently analyzed metrics with the distribution of results from previous projects.

Collaboration Partners
- c.c.com Moser GmbH
- Software AG

Webpage and Documentation
- GitHub Repository: SoHist
- EASE Paper
- Documentation: Available in the repository.

Training Material: Availability of User Guides, Tutorials, and Training Content
User guides and setup instructions are included in the GitHub repository. Technical details are provided in the EASE paper documentation.

Licensing and Intellectual Property
Open Source. Licensing details are provided in the repository.

3.7. SONATA

SONATA extracts software structure and generates test recommendations through knowledge graphs.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)
- TRL: 4 – Proof of Concept
- CRL: Under testing.

Main Partner and Owner
Izertis. pharb@izertis.com

Key Features
- Recommends test cases based on code analysis.
- Visual representation of class diagrams.
- Knowledge graph-based comparisons.

Collaboration Partners
- Izertis

Webpage and Documentation
Internal documentation for SONATA is available and can be provided upon request to support users in understanding and utilizing the tool.

Training Material
Training materials for SONATA are designed for internal teams and can be made available upon request to improve their application of the tool.

Licensing and Intellectual Property
Proprietary.

3.8. UI Test Generator - User Interface Test Design

Automates the creation of UI unit and end-to-end tests for Vaadin applications.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)
- TRL: 5 – Prototype
- CRL: Integrated within Vaadin Copilot as an experimental plugin.

Main Partner and Owner
Vaadin. yuriy@vaadin.com

Key Features
- Supports Playwright-based UI testing.
- Visual regression testing capabilities.
- Maven plugin support for integration into non-Vaadin environments.

Collaboration Partners
- Vaadin

Webpage and Documentation
- Documentation is accessible internally. The private GitHub repository is available at: https://github.com/vaadin/ui-test-generator. Access can be provided on request after agreeing to terms.
- Issues can be reported as GitHub tickets in Vaadin Copilot's repository at: https://github.com/vaadin/copilot. For other environments, issues can be reported via email to yuriy@vaadin.com.

Training Material
Technical documentation, including setup instructions, configuration details, and usage examples, is available upon request. The topic of Generating UI Tests with AI has been highlighted in several conference presentations, showcasing the capabilities of the Vaadin UI Test Generator. For those seeking practical insights and further information, presentation slides and recordings are available upon request.

Licensing and Intellectual Property
- Open source: Includes the API without actual implementation.
- Commercial: Covers proprietary elements related to LLM prompt handling and the logic implementation.

3.9. MUT4SLX – Mutation Testing for Simulink

MUT4SLX performs mutation testing for Simulink models by injecting faults to evaluate the fault-detection capability of test suites.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 3 – Proof of Concept
- CRL: Tested in automotive and avionics domains.

Main Partner and Owner

University of Antwerp. serge.demeyer@uantwerpen.be

Key Features

- Applies fault injection on Simulink models.
- Supports single and multi-mode fault generation.
- Evaluates test coverage and robustness.

Collaboration Partners

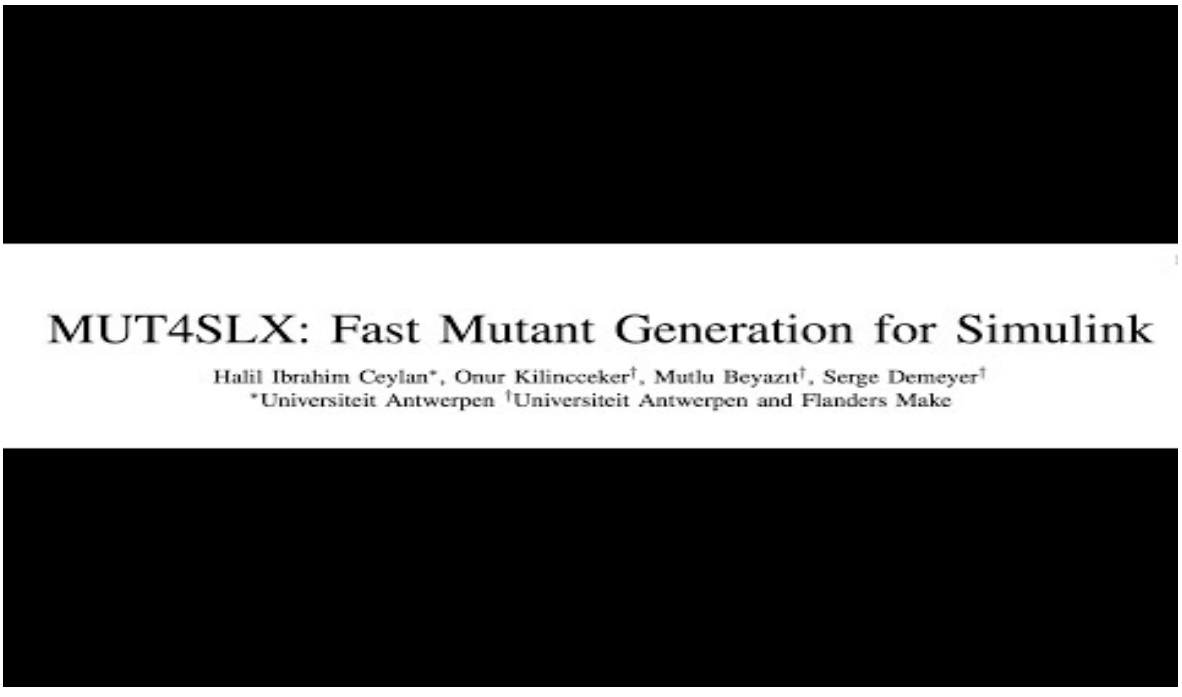
- Alstom

Webpage and Documentation

The repository provides documentation for MUT4SLX, detailing its features and offering guidance for installation and use: https://github.com/haliliceylan/MUT4SLX

Training Material

A video tutorial is available, offering a step-by-step walkthrough for understanding and using MUT4SLX: https://youtu.be/inud_NRGutc?si=JM7JU2eDV5uUIUMK



Licensing and Intellectual Property

Proprietary.

3.10. DRACONIS

DRACONIS is an analysis tool designed for automating analysis-based validation tasks, with a focus on deltas.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 4 – Prototype
- CRL: Pilot-tested in CI/CD pipelines.

Main Partner and Owner

MDU. jean.malm@mdu.se

Key Features

- Extendable program analyser for block-based programming languages.
- Automated Delta analysis of program units with suggested follow-up actions.
- Supports both web-based and command-line interface.

Collaboration Partners

- Alstom
- RISE

Webpage and Documentation

The repository provides documentation for DRACONIS: https://github.com/jean-malm-mdh/draconis

Training Material

Initial training material for DRACONIS is available in the repository in GitHub: https://github.com/jean-malm-mdh/draconis

Licensing and Intellectual Property

Open Source. Licensing details are provided in the repository.

3.11. Test Effort Estimator

This tool estimates the effort required for testing based on code complexity, changes, and historical metrics.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 3 – Proof of Concept
- CRL: Prototype evaluation phase.

Main Partner and Owner

MDU. eduard.paul.enoiu@mdu.se

Key Features

- Estimates test effort based on historical data.
- Provides cost and time predictions for testing activities.

Collaboration Partners

- Alstom

Webpage and Documentation
The specific documentation is restricted to internal access within the project consortium and authorized personnel. For access the main owner should be contacted. A webpage is available: https://github.com/eduardenoiu/Test-Effort-Estimator/

Training Material
Training material related to test effort estimation methods developed is available in the GitHub repository: https://github.com/eduardenoiu/Test-Effort-Estimator/

Licensing and Intellectual Property
Proprietary.

3.12. PyLC

PyLC supports test case generation and model-to-code transformation for Python-based industrial control systems.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)
- TRL: 4 – Prototype
- CRL: Under testing in industrial pipelines.

Main Partner and Owner
MDU. mikael.salari@mdu.se

Key Features
- Python-based transformation and test generation for PLC systems.
- Integrates with PLC tools for automated workflows.

Collaboration Partners
- Alstom

Webpage and Documentation
The documentation[1] [2] includes insights into the PyLC methodologies and applications. Additionally, the repository for PyLC provides the source code, examples, and setup instructions for hands-on exploration and implementation: https://github.com/MikaelSalari/PyLC

Training Material
The usage process is available in the repository: https://github.com/MikaelSalari/PyLC

Licensing and Intellectual Property
Open Source.

[1] https://dl.acm.org/doi/abs/10.1145/3555776.3577698
[2] https://ieeexplore.ieee.org/abstract/document/10479451

3.13. GW2UPPAAL

GW2UPPAAL integrates graph-based models with UPPAAL for formal verification and test case generation.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 3 – Proof of Concept
- CRL: Experimental stage.

Main Partner and Owner

MDU. eduard.paul.enoiu@mdu.se Issues and features requests to be reported via email.

Key Features

- Formal verification for model-based systems.
- Test generation using UPPAAL models.

Collaboration Partners

- Alstom

Webpage and Documentation

The repository for GW2UPPAAL provides documentation on the tool, its setup, and usage instructions, along with the source code and examples:
https://github.com/eduardenoiu/GW2UPPAAL

Training Material

Training resources for GW2UPPAAL are available, including a repository section with examples and explanations and a video tutorial offering step-by-step guidance:
https://github.com/eduardenoiu/GW2UPPAAL
https://youtu.be/EtsJ-GNSjJM?si=PjEvQ_fJmh_OgEEK

Licensing and Intellectual Property

Open Source

3.14. PLC-EARS: Testing Utilizing EARS Notation in PLC Programs

This tool tests PLC programs using the Easy Approach to Requirements Syntax (EARS) for structured requirements validation.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 4 – Prototype
- CRL: Pilot evaluation stage.

Main Partner and Owner

MDU. mikael.salari@mdu.se and eduard.paul.enoiu@mdu.se

Key Features

- EARS-based requirements validation.
- Test generation and execution for PLC programs.

Collaboration Partners

- Alstom

Webpage and Documentation

Relevant resources include a report [3] and a repository containing the presentation and supplementary materials [4].

Training Material

Training material in the repository includes an introduction to the EARS-based testing approach for transforming natural language requirements into testable specifications, followed by detailed steps for test generation, execution, and analysis in PLC environments such as CODESYS.

Licensing and Intellectual Property

Proprietary.

3.15. Jazure

A background job thar syncs Jira workitems with Azure DevOps to link PRs with related work items.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 9 – System,Test & Ops
- CRL: Deployed To One Client

Main Partner and Owner

Arcelik. omercan.devran@beko.com

Key Features

- Integration with Azure services.

[3] https://ieeexplore.ieee.org/abstract/document/10132248/
[4] https://github.com/MikaelSalari/EARS-for-PLCS/tree/main

Collaboration Partners

- Bilkent University

Webpage and Documentation

Internal documentation for Jazure is available upon request, providing a detailed guide on its functionality and integration.

Training Material

Training material is internal and can be provided upon request to support users in utilizing Jazure.

Licensing and Intellectual Property

Proprietary.

3.16. Smellyzer

Smellyzer is a standalone tool that detects process smells in code review (CR) and bug tracking (BT) processes by analyzing data from platforms like GitHub, Jira, and GitLab, offering statistical insights and visualizations to improve software development practices.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 9 – System,Test & Ops
- CRL: Deployed To One Client

Main Partner and Owner

Bilkent University / Arcelik. eraytuzun@cs.bilkent.edu.tr

Key Features

- Process Smell Detection: Identifies CR and BT smells based on established taxonomies, helping pinpoint suboptimal practices in software development.
- Platform Support: Analyzes data from multiple platforms including Git, GitHub, Azure DevOps, GitLab, Gerrit, Jira, and BitBucket.
- Visualization Capabilities: Provides visual insights into smell occurrences by year, developer, and type for easier interpretation.

Webpage and Documentation

Documentation for Smellyzer is internal and not publicly available. It can be provided upon request to authorized individuals or teams.

Training Material

Training material for Smellyzer is available on demand.

Licensing and Intellectual Property

Closed Source.

3.17. Relink

ReLink is a semi-automated predictive tool designed to recover and establish traceability links between pull requests (PRs) and issues in software development, leveraging machine learning and heuristic methods for improved project efficiency and transparency.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 9 – System,Test & Ops
- CRL: Deployed To One Client

Main Partner and Owner

Bilkent University / Arcelik. eraytuzun@cs.bilkent.edu.tr

Key Features

- Semi-Automated PR-Issue Linking: Combines text similarity and heuristic rules to suggest potential links with confidence scores normalized between 0 and 100.
- Visualization Capabilities: Offers visual insights into linkage trends over time, historical graphs, and analysis dashboards to monitor performance.
- Multi-Platform Support: Integrates with GitHub, Jira, and Azure DevOps via REST APIs for seamless data retrieval.

Webpage and Documentation

Internal documentation for ReLink is available upon request with specific guides on its features and integration for use. Some documentation details are included in a published report [5].

Training Material

Training material for ReLink is available upon demand to support users in utilizing its capabilities.

Licensing and Intellectual Property

Open Source.

3.18. Telemetry Anomaly Analyzer (TAA)

TAA detects anomalies in OpenTelemetry compatible telemetry data to identify system issues early.

Technical Readiness Level (TRL) and Commercial Readiness Level (CRL)

- TRL: 4 – Prototype
- CRL: Pilot testing in SaaS environments.

Main Partner and Owner

Hoxhunt. janne.piironen@hoxhunt.com

Key Features

- Machine learning-based anomaly detection.
- Visual analysis of telemetry data trends.

Webpage and Documentation

Internal documentation for TAA is available upon request, providing insights into its capabilities.

Training Material

---

5 https://www.researchgate.net/publication/387983699_Evaluating_ReLink_for_Traceability_Link_Recovery_in_Practice

Training material for TAA is available upon demand to help users understand and apply its features.

Licensing and Intellectual Property
Proprietary.

4. Training, Support, and Feedback

We provide a set of resources designed to meet the needs of users and collaborators, enabling the adoption and integration of the WP3 tools into specific software development workflows.

Training materials provided by SmartDelta are tool-specific and include guides and tutorials. For example, NALABS, an open-source tool for detecting ambiguity in requirements, includes a repository with user guides and setup instructions, complemented by a video walkthrough available on YouTube. Similarly, MUT4SLX, designed for mutation testing of Simulink models, provides its training materials directly through its GitHub repository and instructional videos, allowing users to understand its fault injection and evaluation capabilities.

Workshops and seminars organized by SmartDelta partners during the project emphasize the hands-on learning and practical applications of the WP3 tools. To facilitate user feedback and ensure improvement, SmartDelta uses specific channels based on the WP3 tool's classification. Open-source tools, such as PyLC for Python industrial control systems testing and SoHist for code quality assessment, utilize repositories for issue tracking and feature requests. Closed-source tools rely on direct channel support or structured feedback forms provided by the tool owners.

Documentation is part of SmartDelta's resources for the WP3 toolset, with tools supported by manuals, references, and case studies. For instance, VARA+, provides recommendations for requirements reuse, integrates its documentation within its GitHub repository, supplemented by internal guides for pilot industry implementations. Tools such as UI Test Generator and Relink use API references to simplify integration into development pipelines, while SoHist includes case studies illustrating its application in code quality analysis.

5. Summary

The SmartDelta project deliverable, D3.5 Delta-oriented Quality Assurance Toolset final version, is a guide focused on introducing a suite of tools designed to optimize software quality assurance, technical debt management and requirements analysis. Each tool addresses specific challenges in software development, ranging from test generation to managing technical debt and improving user interface testing. This final version of the toolset details the functionalities, input and output formats, access information, and contact details for each tool, providing a resource for developers and quality assurance professionals in the field of software engineering.