ITEA Office
High Tech Campus 69 - 3
5656 AG Eindhoven
The Netherlands

T + 31 88 003 6136
E info@itea3.org
W www.itea3.org

*ITEA 3 is a EUREKA strategic ICT cluster programme*

# Exploitable Results by Third Parties

## 16032 TESTOMAT Project

### Project details

| | |
|---|---|
| Project leader: | Sigrid Eldh |
| Email: | Sigrid.eldh@ericsson.com |
| Website: | testomat.eu |

## Model and requirements-based test prioritization

| Input(s): | Main feature(s) | Output(s): |
|---|---|---|
| ▪ Requirements<br>▪ Specification model<br>▪ Test Cases (Generated)<br>▪ Prioritization attributes (metrics) | ▪ Prioritization of test cases using a hierarchical clustering and requirements-based metrics<br>▪ Classification and requirements-based analysis of generated test cases | ▪ Test Case Priorities<br>▪ Optimized test run order |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ Automated test case prioritization based on an initial requirement assessment<br>▪ Allows for a deeper insight into generated test cases and more careful selection of required test cases to satisfy a specified test goal |
| Integration constraint(s): | ▪ Currently only applicable for generated test cases |
| Intended user(s): | ▪ Test engineers |
| Provider: | ▪ Ifak |
| Contact point: | ▪ karsten.meinecke@ifak.eu |
| Condition(s) for reuse: | ▪ Prototype available per request |

*Latest update: 24.08.2020*

| Modular test adapter | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| ▪ Generic test commands/steps (MQTT Protocol)<br>▪ Adapter Configuration | ▪ Translation of generic test steps into protocol-specific commands/messages<br>▪ Designed for easy extensibility<br>▪ Current supported protocols: OPC UA, Modbus, UDP | ▪ Target protocol specific test commands/steps |
| Unique Selling Proposition(s): | ▪ Allows a protocol-independent test case design<br>▪ Efficient testing of SUT with multiple interfaces using the same test suite | |
| Integration constraint(s): | ▪ MQTT-support of test tool (to send/receive generic test commands) | |
| Intended user(s): | ▪ Test engineers, test tool/service provider, | |
| Provider: | ▪ ifak | |
| Contact point: | ▪ karsten.meinecke@ifak.eu | |
| Condition(s) for reuse: | ▪ Prototype available per request | |
| | *Latest update: 24.08.2020* | |

| SZZ Unleashed | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| ▪ Issue tracker<br>▪ Git repository | ▪ Finds bug-introducing commits from a set of bug-fixing commits | ▪ commits.json: all commits that have been blamed to be bug introducing<br>▪ annotations.json: a representation of the graph that is generated by the algorithm in the blaming phase<br>▪ fix_and_bug_introducing_pairs.json: all possible pairs which could lead to a bug introduction and fix |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ SZZ Unleashed is an implementation of the SZZ algorithm, i.e. an approach to identify bug-introducing commits, introduced by Śliwerski et al's in "When Do Changes Induce Fixes?", in *Proc. of the International Workshop on Mining Software Repositories*, May 17, 2005.<br>▪ This repository responds to the call for public SZZ implementations by Rodríguez-Pérez, Robles, and González-Barahona. "Reproducibility and Credibility in Empirical Software Engineering: A Case Study Based on a Systematic Literature Review of the use of the SZZ Algorithm", *Information and Software Technology*, Volume 99, 2018. |
| Integration constraint(s): | ▪ The only supported issue tracker is Jira (should be easy to extend).<br>▪ Only git repositories are supported. |
| Intended user(s): | ▪ Software engineering researchers<br>▪ Tool developers<br>▪ Anyone interested in bug prediction |
| Provider: | ▪ Oscar Svensson and Kristian Berg (formed MSc students supervised by RISE Research Institutes of Sweden) (contact: Markus Borg, RISE) |
| Contact point: | ▪ https://github.com/wogscpar/SZZUnleashed , markus.borg@ri.se |
| Condition(s) for reuse: | ▪ Open source software under an MIT license |

*Latest update:* **2020-08-12**

| SWEET | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| ▪ C Code | ▪ Program Flow Analysis<br>▪ Approximate BCET/WCET calculation<br>▪ Other static analyses (slicing, value analysis, …) | ▪ Flow facts (program flow constraints)<br>▪ BCET/WCET estimates |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ Advanced, precise program flow analysis<br>▪ Expressive flow fact language for program flow constraints |
| Integration constraint(s): | ▪ GMP<br>▪ CMake 2.8 or later<br>▪ gcc 4.2 or later |
| Intended user(s): | ▪ End users working with validation of (hard) real-time systems, for instance to find loop constraints or perform program slicing to guide testing efforts on real-time tasks.<br>▪ Researchers focusing on program flow or timing analysis. |
| Provider: | ▪ Mälardalen University, Programming Languages group<br>▪ Open Source – code available at: https://www.mrtc.mdh.se/projects/wcet/sweet.html |
| Contact point: | ▪ Björn Lisper, bjorn.lisper@mdh.se |
| Condition(s) for reuse: | ▪ BSD style license |

*Latest update: 2020-08-13*

## Web application for model-based testing with CI/CD integration

| Input(s): | Main feature(s) | Output(s): |
|---|---|---|
| ▪ Requirements | ▪ Requirements-based model and test generation<br>▪ Remote test execution<br>▪ CI/CD Integration for model-based testing<br>▪ Traceability from test cases/test results to requirements | ▪ Requirements models<br>▪ Generated test cases<br>▪ Test results |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ Newly designed web application for model-based testing with high usability in mind<br>▪ Highly automated tool for generation of test cases from requirement models<br>▪ Fully automated intermediate steps like synthesis of a specification model from requirements and subsequent test generation<br>▪ Textual and graphical editors for requirement models |
| Integration constraint(s): | ▪ web-based solution requires docker environment |
| Intended user(s): | ▪ Test engineers, test service provider |
| Provider: | ▪ Ifak |
| Contact point: | ▪ karsten.meinecke@ifak.eu |
| Condition(s) for reuse: | ▪ demo available, commercial license planned |
| | *Latest update: 24.08.2020* |

| RF::CAT (Cover Automatic Tester) | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| ▪ XML file exported from TiA V15.1 (Siemens) | ▪ Parsing of XML format of TiA<br>▪ Analyze relations between signals and step chains<br>▪ Check naming of signals regarding standards<br>▪ Checking unambiguousness of the transition conditions of a step into an step chain | ▪ Test result as Text file or PDF |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ Complete automated testing of the unambiguousness of the transition conditions of a step into an step chain to 100% |
| Integration constraint(s): | ▪ Is created only for Windows operation systems via .NET C#<br>▪ Can only import step change from TiA via Siemens XML format<br>▪ Additional test routines cannot be added by an user without programming skills into C# and the source code of RF::CAT (Cover Automatic Tester) |
| Intended user(s): | ▪ PLC programmer<br>▪ Virtual Commissioning Engineer<br>▪ Project coordinator |
| Provider: | ▪ EKS InTec GmbH (License Required) |
| Contact point: | ▪ Thorsten Schmitz, thorsten.schmitz@eks-intec.de |
| Condition(s) for reuse: | ▪ License Required |

*Latest update: 14.07.2020*

| ABBRobotTester | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| ▪ Native ABB robot programs | ▪ Completely automated testing of ABB robot programs regarding to the movements | ▪ Test result as Text file or PDF |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ Completely automated testing of ABB robot programs regarding to the movements |
| Integration constraint(s): | ▪ RobotStudio 6.08 (or higher)<br>▪ .Net / C#<br>▪ GUI – Windows Forms |
| Intended user(s): | ▪ PLC programmer<br>▪ Virtual Commissioning Engineer<br>▪ Project coordinator |
| Provider: | ▪ EKS InTec GmbH (License Required) |
| Contact point: | ▪ Thorsten Schmitz, thorsten.schmitz@eks-intec.de |
| Condition(s) for reuse: | ▪ License Required |

*Latest update: 14.07.2020*

| Algorithms for Rare Event Simulation | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| Criticality Monitor<br>Stochastic Model<br>Description of the search domain | Uses different algorithms to guide the system towards rare (critical) events<br>Provides a quantitative estimate for the risk that a specific event occurs | Report of the critical regions in the search domain |
| Unique Selling Proposition(s): | The included algorithms are more effective at finding rare (critical) events as classic approaches like Monte-Carlo Simulations.<br>This leads to a reduced time effort and thus less costs in comparison to classical methods. | |
| Integration constraint(s): | Implementations for the interfaces of the model and of the monitor have to be created. | |
| Intended user(s): | Technology Provider | |
| Provider: | OFFIS e.V. | |
| Contact point: | s2x-license@v.offis.de | |
| Condition(s) for reuse: | Commissioning in forms of a research / transfer project. | |
| | *Latest update: 21 August 2020* | |

## Dextool Mutate: A mutation testing tool for C/C++

| Input(s): | Main feature(s) | Output(s): |
|---|---|---|
| ▪ Safety-critical software<br>▪ Mission-critical software<br>▪ Business-critical software | ▪ Assess C/C++ test code quality (measure the tests ability to detect artificial faults)<br>▪ Provides human readable reports<br>▪ Help improve test quality | ▪ Test quality reports<br>▪ Suggestions for tests that can be improved<br>▪ HTML GUI |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ Tracks how individual tests perform<br>▪ Reports if tests are useless (does not find faults)<br>▪ Reports tests that are redundant (detects the exact same faults) |
| Integration constraint(s): | ▪ Linux Standard Based OS<br>▪ The project build system must be able to generate a JSON compilation database. |
| Intended user(s): | ▪ Software Developers<br>▪ Software Engineers<br>▪ Project Managers<br>▪ Team Leaders<br>▪ Test Leaders |
| Provider: | ▪ Saab Aeronautics |
| Contact point: | ▪ Joakim Brännström – joakim.k.brannstrom@saabgroup.com |
| Condition(s) for reuse: | ▪ Mozilla Public License<br>▪ Available on github: https://github.com/joakim-brannstrom/dextool |

*Latest update: 13 August 2020*

## API testing solution for IIoT protocols

| Input(s): | Main feature(s) | Output(s): |
|---|---|---|
| ▪ Requirements<br>▪ Specification model<br>▪ Test Cases | ▪ Automated execution of test-cases using a variety of industrial protocols (such as OPC UA, ModBus and UDP). | ▪ Test results |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ Very flexible platform for API testing for IIoT (Industrial Internet of Things) and Industrial Automation systems. |
| Integration constraint(s): | ▪ Works currently with “Test Adaptor” from ifak (www.ifak.eu). Integration via MQTT protocol. |
| Intended user(s): | ▪ Test engineers |
| Provider: | ▪ Parasoft |
| Contact point: | ▪ rix.groenboom@parasoft.com |
| Condition(s) for reuse: | ▪ Prototype available per request |

*Latest update: 31.08.2020*

## AI Driven classification of Static Analysis Findings

| Input(s): | Main feature(s) | Output(s): |
|---|---|---|
| ▪ Static analysis results for C/C++, JAVA and C# | ▪ Artificial Intelligence based clustering of the most relevant findings | ▪ Prioritization of the coding errors |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ Automated suggestion of most relevant static analysis findings based on historic (previous) remediation. |
| Integration constraint(s): | ▪ Algortihm is available with the DTP platform, and works with Parasoft static analysis tools. |
| Intended user(s): | ▪ Software engineers |
| Provider: | ▪ Parasoft |
| Contact point: | ▪ Rix.Groenboom@parasoft.com |
| Condition(s) for reuse: | ▪ Product is available under commercial license. |
| | *Latest update: 31.08.2020* |

| OpenUiData dataset | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| Community contribution | An open source project collecting annotated screenshot data from multiple different software UI's to help democratize machine vision development (https://github.com/openuidata/openuidataset ) | Machine vision training corpus for computer systems |

| | |
|---|---|
| Unique Selling Proposition(s): | First open source dataset of annotated ui screenshots, with UI elements as the annotated data |
| Integration constraint(s): | . |
| Intended user(s): | Machine learning and machine vision developers |
| Provider: | Qentinel Quality Intelligence Oy |
| Contact point: | Henri Terho (henri.terho@gmail.com) |
| Condition(s) for reuse: | License terms in repo |
| | *Latest update: 2020-08-20* |

| TESTONA | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| ▪ Subject of test modelled as classification tree<br>▪ Import from different file formats | ▪ Automatic test case generation<br>▪ Test case prioritization<br>▪ Test suite creation | ▪ Testcases as Generated code<br>▪ Export to different tools<br>▪ Testcase Specification |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ Traceability of requirements<br>▪ Automatic test case prioritization<br>▪ Adjustable test coverage |
| Integration constraint(s): | Stand alone or integration into Eclipse |
| Intended user(s): | System and Software Developers |
| Provider: | Expleo Germany GmbH |
| Contact point: | support_testona@expleogroup.com<br>testona.net |
| Condition(s) for reuse: | Licensing conditions available at https://www.expleo-germany.com/en/products/testona/download-price/ |

*Latest update: 28.08.2020*

| MODICA | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| Subject of test modelled as MODICA model | ▪ Variant management<br>▪ Validation<br>▪ Test case generation | testcases |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ Requirements management<br>▪ Variant management<br>▪ Parallel state charts |
| Integration constraint(s): | Stand alone |
| Intended user(s): | System and Software Developers |
| Provider: | Expleo Germany GmbH |
| Contact point: | support_testona@expleogroup.com<br>testona.net |
| Condition(s) for reuse: | Licensing conditions available at https://www.expleo-germany.com/en/products/testona/download-price/ |

*Latest update:28.08.2020*

| UML Testing Profile 2 | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| Test Basis | Graphical test modelling language<br>Concepts for test automation architectures<br>Language support for test analysis, test design, test execution, test logging, verdict calculation | Test specifications<br>Test architectures |

| | |
|---|---|
| Unique Selling Proposition(s): | The UML Testing Profile 2 |
| Integration constraint(s): | None |
| Intended user(s): | Test Engineers, Test Architects |
| Provider: | OMG |
| Contact point: | marc-florian.wendland@fokus.fraunhofer.de |
| Condition(s) for reuse: | Reuse in general allowed - look into the specification for further information about the Copyright |

*Latest update: June 2019*

| Cascading Arbiter | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| • UTP2-based arbitration specification;<br>• Result of comparison of expected vs. actual result;<br>• Specification of user-defined verdicts | • Verdict calculation on three levels (test action, test case and test set);<br>• Cascading reuse of arbitration specifications for different arbitration targets<br>• User-defined verdicts and verdict calculation rules | (User-defined) verdicts for test actions, test cases and test sets |

| | |
|---|---|
| Unique Selling Proposition(s): | It is difficult to predict whether a not specified and newly explored behavior of the system under test is erroneous or not. Therefore, it is required to go beyond the pure functional specification of pass/fail-criteria. The cascading arbitration specification facility of the UTP 2 standard and its implementation within Fokus!MBT enables testers to model sophisticated pass/fail-criteria in particular for non-functional test evaluation such as fault tolerance testing, security or performance testing. |
| Integration constraint(s): | Based on the UML Testing Profile 2 |
| Intended user(s): | Test engineers |
| Provider: | Fraunhofer FOKUS |
| Contact point: | Marc-florian.wendland@fokus.fraunhofer.de |
| Condition(s) for reuse: | Per request |

*Latest update: March 2020*

| Fuzzino | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| ▪ Interface specification of the system under test (SUT)<br>▪ Grammar of formal parameters<br>▪ Existing functional test cases | ▪ Automated generation of test data based on the type and/or grammar of SUT interface parameters<br>▪ Automated prioritization of test data generation and test execution based on fuzzing heuristics<br>▪ Automated generation of fuzz test cases | ▪ Test reports for fuzz tests |

| | |
|---|---|
| Unique Selling Proposition(s): | ▪ TBD |
| Integration constraint(s): | ▪ Java; for further information refer to https://github.com/fraunhofer-fokus/Fuzzino<br>▪ |
| Intended user(s): | ▪ Test engineers, Security testers |
| Provider: | ▪ Fraunhofer FOKUS |
| Contact point: | ▪ martin.schneider@fokus.fraunhofer.de |
| Condition(s) for reuse: | ▪ Fuzzino is licensed under the Apache License 2.0. |

*Latest update: June 2020*

| TESTAR | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| AN EXECUTABLE GUI | PROVIDES SCRIPTLESS GUI TESTS | A TESTED GUI AND A SET OF BUGS (IF THEY EXISTS) |

| | |
|---|---|
| UNIQUE SELLING PROPOSITION(S): | Open source scriptless GUI testing tool |
| INTEGRATION CONSTRAINT(S): | Currently supports automated testing of Windows desktop and web applications |
| INTENDED USER(S): | Anyone developing software with a GUI |
| PROVIDER: | The main developers are Open Universiteit (NL) and Universitat Politècnica de València (Spain) |
| CONTACT POINT: | pekka.aho@ou.nl |
| CONDITION(S) FOR REUSE: | Open source BSD3 license |

*Latest update: 27 Aug 2020*

| **Testura - A mutation test tool for C#** | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| ▪ Any C# code or software<br>▪ Automated test cases | ▪ Assess C# test code quality<br>▪ Guides test code improvement<br>▪ Fast and flexible<br>▪ Possible to select mutations<br>▪ Results in realtime | ▪ Test quality reports<br>▪ Number of survived mutantsl |
| Unique Selling Proposition(s): | ▪ Tracks how individual tests perform<br>▪ Reports if tests does not find faults<br>▪ Works on code deltas (change based) to reduce execution time | |
| Integration constraint(s): | ▪ Can be executed stand-alone or integrated as an extension with Visual Studio and in CI/CD build chain. | |
| Intended user(s): | ▪ Software Developers<br>▪ Software Engineers<br>▪ Test Leaders<br>▪ Quality Assurance | |
| Provider: | System Verification Sweden AB | |
| Contact point: | ● Mille Boström – mille.bostrom@systemverification.com<br>Magnus C Ohlsson – magnus.c.ohlsson@systemverification.com | |
| Condition(s) for reuse: | ▪ Mozilla Public License<br>▪ Available on github: https://github.com/Testura | |
| *Latest update: 2020-08-27>* | | |

| Test automation maturity assessment instrument (questionnaire) | | |
|---|---|---|
| Input(s): | Main feature(s) | Output(s): |
| • Understanding of adopted test automation practices<br>• Efforts to answer questionnaire | • Assess test automation maturity based on adopted practices<br>• Benchmark the current state of test automation<br>• Compare with the rest of the industry | • Feedback to test automation maturity<br>• Test automation maturity Improvement steps |

| | |
|---|---|
| Unique Selling Proposition(s): | A self-assessment instrument practitioners can use to assess and improve test automation maturity |
| Integration constraint(s): | Practitioners should consider individual differences of their test automation when using this instrument to assess test automation maturity |
| Intended user(s): | Test automation practitioners |
| Provider: | University of Oulu |
| Contact point: | **yuqing.wang@oulu.fi**, **mika.mantyla@oulu.fi** |
| Condition(s) for reuse: | Utilization means you comply to share your data (which is anonymized) with the researchers. |

*Latest update: <24.08.2020>*