Project[1] Number: [731529]

Project Acronym: [STAMP]

Project title: [Software Testing AMPlification]

Periodic Technical Report

Part B

Period covered by the report: from [01/06/2018] to [30/11/2019]

Periodic report: 2nd

[1] The term ‘project’ used in this template equates to an ‘action’ in certain other Horizon 2020 documentation

1 Explanation of the work carried out by the beneficiaries and Overview of the progress

1.1 Objectives

Objective 1. Provide an approach to automatically amplify unit test cases when a change is introduced in a program The automatic generation of variant test cases aims at triggering a larger variety of behaviors and at observing a larger variety of program states in order to increase the number of regression bugs detected before going in production.

This objective has been addressed by the research, development and experimental work of WP1. The **main achievements of WP1**, which address this objective are as follows.

DSpot, a tool for the automatic amplification of unit Java test cases. In the second period of STAMP we have focused the research about amplification on the following question: can DSpot amplify test cases that focus on the code changed in one commit and can these test cases detect the behavior change introduced by the commit? A second major activity around DSpot focused on performance improvement through extensive profiling and code optimization activities.

We investigate extreme mutation in order to improve the assessment of test cases written by developers as well as the test cases amplified by DSpot. For this we have developed **Descartes, a tool that automatically reports on pseudo-tested methods and weak test cases**. The key research question for second period was: can we generate hints that indicate what test improvement actions developers can take to improve their test suite?

These research and development activities performed to address Objective 1 have been consolidated into industry-ready packages. We have published a survey about test amplification in the Journal of Systems and Software, a top software engineering journal. The works on DSpot for commits and automatic suggestion improvements are under review in the journals of Empirical Software Engineering and Transactions on Software Engineering.

Objective 2. Provide an approach to automatically generate, deploy and test large numbers of system configurations. The specification and deployment of system configurations (assembly of unit components to form a complete configuration of the system under test) is currently a task that relies on huge manual effort.

In this period, we addressed this objective in WP2 mostly, through those main achievements:

- Develop new features in the CAMP tool, including
  - new strategies to reduce the number of generated configurations to be tested and to select the most relevant configurations
  - integration with JUnit to execute unit tests on the different configurations and aggregate reports
  - integration with JMeter to execute performance tests on the different configurations and aggregate reports
  - weaving of instrumentations in CAMP configurations to monitor the execution of configurations with monitoring tools (perf, strace) and analyze the traces produced by those tools to quantify how different configurations differ from each other.
- Consolidate CAMP to prepare for exploitation, including
  - extensive refactoring of the code base, including migrating to Python 3.x,
  - improving the documentation,
  - fixing 45 issues (over 62 opened) and releasing 28 versions
- Provide active support to use case providers and extensive validation in WP5

- Conduct experiments outside STAMP use cases on two large open-source projects, respectively with 13k and 36k commits, and 2.5k and 48k stars on GitHub, where we detected important issues.
- In WP6, confirm the scientific relevance of CAMP through a publication at IEEE ISSRE, the top conference for software reliability engineering, and its practical applicability through a number of demos, tutorials and technical talks e.g. at the A-TEST workshop at FSE.

Objective 3. Provide an approach to automatically amplify, optimize and analyze production logs in order to retrieve test cases that verify code changes against real world conditions. While the collection of information from production is an intrinsic and key feature of DevOps, the connection between this log data (collected in Ops) and the testing activity (on the Dev side) remains weak.

In this period, we addressed this objective in WP3 mostly, through those main achievements:

- Development of the first crash reproduction benchmark for Java (JCrashPack).
- Reimplementation of the EvoCrash approach as Botsing, an extensible and license friendly framework reproducing crashes from stack traces collected from log data.
- Extension of EvoSuite for Runtime AMPlification (RAMP) to take the behavior of the software under test into account to generate unit tests (model seeding).
- Coordination of development activities around Botsing between the different partners
    - Support of the maven plugin assured during the various phases of developments, including the re-engineering to Botsing
    - Development of a pre-processor module for the input stack trace and a parallelized version of Botsing for crash reproduction.
- Dissemination of the results in the scientific community through high-quality publications.

Objective 4. Develop three test amplification microservices that can be integrated in different toolchains. This objective aims at developing a flexible architecture for test amplification services, leveraging a microservice approach.

The objective has been addressed by completing the development of WP4. Key final results towards Objective 4 are:

- components (a microservice and a Jira plugin) to integrate STAMP tools with Jira Software and Github Issues issue trackers;
- a new Jenkins plugin, a pipeline library and pipelines to enhance CI/CD processes with STAMP features;
- courseware based on documentation, sample pipelines and a Docker image containing a full stack CI/CD environment enhanced with STAMP features, available in DockerHub;
- enhancing the collaborative platform with a dedicated server configured to have a complete STAMP CI/CD environment;
- Eclipse plugin to run STAMP tools: DSpot, Descartes, Botsing and RAMP within the IDE;
- new releases of Maven plugins developed during the first period, keeping them aligned with the evolution of STAMP tools.

Objective 5. Validate the relevance and effectiveness of amplification on 5 use cases. This objective has a dual dimension: validate the effectiveness of test amplification techniques; and validate the relevance of amplification beyond software production within specific application domain. The first aspect aims at demonstrating that test amplification can be applied on the use cases, which are industry-strength code bases, representative of the type of application targeted by STAMP: applications that run continuously in the cloud. The second aspect aims at demonstrating that test amplification addresses a significant problem that occurs in multiple domains.

This objective has been addressed by the industrial evaluation activities conducted by the STAMP industrial partners in WP5:

- KPIs have been reviewed and tuned to better match the reality and to provide more interesting results. New metrics have been added to also provide better visualization of results.
- Each Industrial UCs have experimented all the STAMP tools by applying them into their different case projects they selected for evaluation.
- Industrial UCs have incorporated some of the STAMP tools (and technologies) into their CI/CD procedures (e.g. pipelines) and/or their software development processes (either by through project management tools or an IDE).
- Industrial partners have computed KPI metrics, based on conducted experiments, and reported them. From these KPIs measurements, they have drawn conclusions about the effectiveness of the STAMP test amplification techniques on their industrial software development processes.
- Moreover, industrial partners have answered industrial validation questions, based on KPIs measurements, that aim at evaluating the relevance of amplification techniques beyond software engineering in the diversity of their industrial relevant context.

Objective 6. Disseminate and exploit the open source STAMP test amplification services. STAMP's test amplification techniques are developed as open source software.

This objective has been addressed through several activities conducted in WP6:

- **Website**. Developed and curated a website with content such as product overviews, videos, interviews, list of scientific publications. The website is the main dissemination entry point to the project while the GitHub repository is the technical entry point.
- **Collateral**. Developed a full range of communication material including an efficient visual identity, 14 videos [18 over 3 years], a tri-fold brochure, an A5 flyer for the beta-testing campaign, a roll-up poster for exhibition, t-shirts for beta-testers.
- **Beta-testing campaign**. Developed and supported a beta-testing campaign with dedicated web pages, communication material and incentives. It resulted in the collection of 12 questionnaires with relevant feedback.
- **Events**. Showcased the STAMP projects in 15 industry events worldwide [27 over 3 years] including Devoxx, EclipseCon Europe, FOSDEM, OpenStack Summit, OSCON, Paris Open Source Summit, etc.
- **Workshops**. Three webinars and 16 workshops [20 over 3 years] were organized for in-depth presentations of the tools to academic and industry potential users. They include large organisations such as Orange, DGIT (European Commission), Capgemini, GFI, Sopra-Steria, Inria, Spotify, Ericsson, Station F (start-up incubator), Dagstuhl, etc.
- **Cooperation:** Three workshops organised and colocated with the ElasTest team where both projects explored alignment, synergies, common demos, software advances, etc.
- **Communication**. The consortium had an on-going communication activity directed at the press and social media with three press releases resulting in two full featured articles in major industry publications during the second period, a total of 600 tweets resulting in 360 followers, and 22 discussion topics on LinkedIn.
- **Scientific dissemination**. The consortium published 12 papers in scientific publications [31 over three years], enabled the completion of three PhD's papers and participated in dedicated scientific and academic conferences such as ISSRE (Symposium on Software Reliability Engineering), ISSTA (International Symposium on Software Testing and Analysis), ICSE (International Conference on Software Engineering), ICST (International Conf. on Software Testing, Verification & Validation).

1.2 1.2 Explanation of the work carried per WP

1.2.1 Work package 1

Task 1.1 - State of practice for unit testing and test assessment (m1-m6)

This task was completed in the first period of STAMP

Task 1.2 - Measurement tool to analyze the interplay between test suites and program (m1-m20)

This task is a scientific and technical task that focuses on the development of the Descartes tool to measure the interplay between a test suite and the methods they cover. Descartes is a tool for software developers who wish to strengthen their test suite Descartes is a service that spots weak test cases and suggests improvements. Unlike code coverage tools, Descartes provides actionable feedback to improve test suites.

In the second period of STAMP, the development and research about Descartes have focused on novel program analyses to generate concrete suggestions for developers to improve their test cases (through the augmentation of existing test cases with test inputs or assertions or by adding a new test case)

Contribution	**Partner(s)**
Metrics discussions and updates. New metrics added to KPI definitions. Developed several scripts and tools for flaky test handling/monitoring (and its integration in Jenkins) and Global Coverage measurements and comparisons between executions, using Clover and Jenkins.	XWiki/XWiki Romania
Develop the Descartes tool to automatically spot pseudo-tested methods. In the second period, we focused on new dynamic analyses to augment the reports of Decartes with suggestions to improve test cases. These new analyses have been experimented with STAMP partners and have been submitted for publication.	INRIA
Advise on state of the art in the area of test analysis and investigate the emergence of new bugs in software applications.	TUDelft

Task 1.3 - Automatic generation of test cases variants (m1-m20)

This task is articulated around research, development and consolidation activities to build the DSpot tool for the automatic amplification of unit test cases. The research activities of the second period have focused on the specialization and the evaluation of DSpot to amplify test cases that are specifically targeting changes introduced by a commit.

Contribution	**Partner(s)**
Lead the task of development and research about unit test amplification. Maintain DSpot, address issues reported by use case partners as well as by external users. Experiment with DSpot in the continuous integration process. Submit research results to the journal of Empirical Software Engineering.	INRIA
Collaborate with INRIA and KTH to design experiments that assess the validity of test amplification.	SINTEF
Advise on the design of search-based algorithms for the amplification of test cases, provide feedback about the experiments and about the scientific findings in the area of unit test amplification.	TUDelft
Various meetings and discussions, including issues reported in the GitHub repository of the various tool (Descartes, DSpot).	XWiki

Task 1.4 - Test execution (m12-m36)

Contribution	**Partner(s)**

Optimization of theDSpot performance and resources (e.g. memory) consumption: ● Profiling of the DSpot execution for performance, memory consumption, threads, obtaining a detailed telemetry. ● Analysis of the profiling report. ● Identification of the performance bottlenecks, memory leaks, etc ● Proposition and implementation of DSpot code patches for performance and memory consumption improvement. ● Parallelization of the DSpot amplified test execution for JUnit4/5 ● Experimentation: DSpot execution on the Dhell target project with different input flags. Analysis of performance and resources consumption improvements for each applied patch. Evidence: ● Pull requests proposed by Jesús Gorroñogoitia in DSpot GitHub repository: https://github.com/STAMP-project/dspot/pulls ● Experimental DSpot performance results reported in D1.4	Atos
Collaborate with ATOS to analyze, evolve and refactor the code of DSpot to improve performance. Address the issues related to the performance of DSpot. ● Listing of leads to improve DSpot performance and memory consumption, ● Feedback on new ways to improve DSpot performance and memory consumption according to the analysis done by Jesús Gorroñogoitia, Evidence: ● see the discussions on required and proposed changes done by Jesús Gorroñogoitia in DSpot GitHub repository: https://github.com/STAMP-project/dspot/	INRIA
Contribute expertise in the area of profiling, performance evaluation, existing tools and performance metrics collection	SINTEF

Task 1.5 - Industrialization (m18-m36)

This task aims at strengthening the software prototypes of WP1 to deliver clearly packaged services, well documented and tested.

The activities of this reporting period were focused on two aspects:

- include in DSpot non-core functionalities useful for production environment (e.g. support for multi-modules projects)
- support regular releases, documentation and packaging.

Contribution	**Partner(s)**
DSpot ● Analysis on the possibility to expose Dspot “as a Service” ● Support for Gradle Java projects included ● Dspot diff and multi-modules support ○ evaluation of possible solutions ○ Jenkins based solution selected	ENG

○ selected solution implemented and tested, in particular Dspot was extended to provide some functionalities offered by Jenkins PitMP ● Support for threshold parameters related to number of pseudo-tested methods and partially tested methods ● Final version of PitMP tested and released	
Collaborated with ENG for the packaging of Descartes and DSpot. Provide regular releases of the tools on Maven Central, with associated release notes. Coordinate these releases with features and improvements reported in the issue trackers of the tools.	INRIA

Development Status

Tools are available in Github:

- DSpot: https://github.com/STAMP-project/dspot
- Descartes: https://github.com/STAMP-project/pitest-descartes
- PitMP: https://github.com/STAMP-project/pitmp-maven-plugin

The following table summarizes the development for the second period.

	DSpot & Maven Plugin & Gradle AutomaticBuilder		**Descartes**	
	31-May-18	**30-Nov-19**	**31-May-18**	**30-Nov-19**
# commits since the previous version	332	357	116	113
# changes (git diff)	408 files changed, 31170 insertions(+), 12680 deletions(-)	510 files changed, 12285 insertions(+), 33588 deletions(-)	54 files changed, 5682 insertions(+), 36 deletions(-)	82 files changed, 221,267 insertions(+), 907 deletions(-)
# files	242	252	51	145
# LOC	1532	15116	5310	227672

Conceptual increment (innovation integrated in each tool during the project)

DSpot

- Compute diff target for test amplification
  - changes in test code
  - changes in app code

- Amplify tests on a diff in the CI
- Extreme mutation to steer amplification
- Code coverage to steer amplification
- Reuse objects that exist in the test code
- Integration in a Maven automatic build
- Qualitative assessment with open source software developers
- Qualitative assessment with STAMP use case partners

Descartes

- Efficient implementation of extreme transformations
- Filter irrelevant pseudo-tested methods
- Compute the list of tests that cover the pseudo-tested methods
- Dynamic analysis to diagnose undetected extreme transformations
    - missing input
    - weak oracle
    - hidden infected state
- Generation of textual hints to support developers improve the test suite
- Integration in automatic build pipeline
- Qualitative assessment with open source software developers
- Qualitative assessment with STAMP use case partners
- Efficient implementation of extreme transformations

Target users

DSpot

- Software developers
- Integration team to suggest test improvements

Descartes

- Software developers who wish to consolidate their test suite
- Project managers responsible for risk management
- Integration team to check the test suites quality on commit
- Quality team to find code review guidance

1.2.2 Work package 2

Task 2.1 - Survey state of practice for configuration specification and automatic configuration planning (m1-m6)

Completed during 1st period. D2.1 accepted.

Task 2.2 - Abstract configuration model (m6-m12)

Completed during 1st period. D2.2 accepted

Task 2.3 - Automatic configuration generation (m9-m30)

Contribution	**Partner(s)**
New features: Rewrite of documentation, simplification of CAMP input models, input models validation, coverage over numerical variables, port to Python 3, execution of	SINTEF

tests and aggregation of results, various bug fixes and three installation options: via docker, a script or manual installation.	
Contribution to D2.3 and D2.4. Work on Kubernetes and OpenShift as configuration targets. Input to development of CAMP to generate configurations for Kubernetes.	TellU
Created and improved the CAMP/TestContainers framework (integrated with JUnit tests) to be able to execute the configuration tests on developer's machines and thus be able to debug them.	XWiki/XWiki Romania

Task 2.4 - Configuration assessment and selection. (m12-m30)

Contribution	**Partner(s)**
Input to CAMP development. Work on different types of configuration and how to assess them. Criteria for CAMP amplification. Amplification for performance optimization. Creating docker containers for running TelluCloud components with CAMP.	TellU
Develop new strategies to generate smaller sets of configurations, still covering all single variations (option --coverage). Development of a dashboard to visualize and process data collected from different configurations generated by CAMP	SINTEF
Implemented support of CAMP/TestContainers in CI (Jenkins) with pipelines development. Implemented 3 jobs to support the various configurations and spread the load on various agents. Recreated and reprovisioned all XWiki agents to use Docker and modified CAMP/TestContainers and tests to support Docker in Docker.	XWiki/XWiki Romania

Task 2.5 - Configuration execution and instrumentation (m18-m36)

Contribution	**Partner(s)**
Instrumentation of CAMP configurations with a profiler (perf/flamegraph) to monitor the execution of those configurations, collect and analyze traces.	Activeeon
Collaborate with Activeeon and SINTEF on the instrumentation of CAMP configurations with perf/flamegraph.	INRIA
Rewrite of CAMP execute to iterate over all generated configurations, run all JUnit tests (or other test frameworks through lightweight extensions) and aggregate test reports. Collaborate with Activeeon and INRIA on perf/flamegraph traces, develop analyzers and visualisations for strace traces.	SINTEF
Support on the finalization of CAMP, in particular concerning testing ● Installation, configuration and preliminary tests on the first releases of CAMP JMeter support for CAMP ● Design, implementation and documentation ● testing, in cooperation with Atos Documentation on CAMP included in STAMP Documentation	ENG
Assessment of the usage of CAMP instrumentation to detect common and uncommon behavior.	TUDelft

Study and reporting on the current state-of-the-art of random configuration selection (https://doi.org/10.1109/ICST.2019.00032).	

Development Status

Tool is available in Github:

- CAMP: https://github.com/STAMP-project/camp
- CAMP Documentation: https://stamp-project.github.io/camp
- CAMP samples, tutorials, etc: https://github.com/STAMP-project/camp-samples

The following table summarizes the development for the second period.

	CAMP	
	31-May-18	**30-Nov-19**
# commits since the previous version	187	734
# changes (git diff)	233 files changed, 165'841 insertions(+), 26 deletions(-)	465 files changed, 257 insertions(+), 11'602'014 deletions(-)
# files	232	1695
# LOC	2724	6308

Conceptual increment (innovation integrated in each tool during the project)

CAMP

- Modeling configuration variability inspired by component-based approaches
- Generation of configurations blueprints using SMT constraint (i.e., Microsoft Z3 solver)
- Generation of all configurations blueprints including user-made domain-specific constraints
- Realization of configurations to convert blueprints into deployable artefacts
- Variability model validation to improve CAMP's usability
- Generation of covering arrays to reduce configuration testing workload
- Execution of configuration, including the collection and aggregation of test reports
  - Support for functional tests using the de facto JUnit XML standard
  - Support for performance tests using JMeter
- Support execution of selected configurations
- Support fetching Docker log files in addition of test-reports, to ease debugging
- Experimental support for Flamegraph instrumentation
- Experimentation with 3rd party open source projects (i.e., Sphinx and Atom)
- Qualitative assessment with STAMP use case partners

Target users

CAMP

- Software developers/managers who need to assess portability or interoperability
- Software developers/managers who need optimize performance
- Project managers responsible for risk management
- Integration team to check for portability and interoperability defects before releasing

1.2.3 Work package 3

Task 3.1 Modern logging practices (m1-m6)

Contribution	**Partner(s)**
Implementation of corrective actions after feedback from the intermediate review: interview of the STAMP industrial partners to identify current logging practices (See revised D31); revision of D31 after feedback from the intermediate review. Submission of a systematic literature review on software monitoring to TSE.	TUDelft

Task 3.2 - Log Data optimization for debugging (m7-m36)

The evaluation of EvoCrash (a crash reproduction approach, implemented in Botsing) and the assessment of the current crash replication capabilities and limitations has been revised and extended. During this evaluation, we defined several actions that can be performed to preprocess a stack trace and enhance the crash reproduction capabilities of Botsing. Those preprocessing steps have been implemented and integrated with the Botsing framework.

We also devised and implemented a model generation tool able to generate state machines representing the common behavior of the classes of the software under test. Those models are used in the subsequents T3.3 and T3.4 to enhance crash reproduction and test case generation.

Finally, we extended the code instrumentation mechanism of Botsing (i.e., the injection of probes into the bytecode of a Java application to monitor and log the runtime behavior of specific classes) to enhance crash reproduction and generation of class integration tests.

Contribution	**Partner(s)**
Reporting of the results of the evaluation using JCrashPack in a journal article accepted for the journal of Empirical Software Engineering (https://doi.org/10.1007/s10664-019-09762-1). Implementation of corrective actions after feedback from the intermediate review: revision of task 3.2 description; revision of D32 to add content related to stack trace preprocessing. Coordination of the development of the stack trace preprocessor to filter and clean stack trace log messages (see https://github.com/STAMP-project/botsing/issues/9 and https://github.com/STAMP-project/botsing/pull/60). Development of botsing-model-generation. First released as part of botsing-1.0.4 (https://github.com/STAMP-project/botsing/releases/tag/botsing-1.0.4). Extension of the code instrumentation capabilities (https://github.com/STAMP-project/botsing/pull/101 and https://github.com/STAMP-project/botsing/pull/86).	TUDelft
Botsing log preprocessor:	ENG

● design, implementation, integration ● maintenance and bug fixing ● support, documentation and tutorials.	
Discussion and feedback on the protocol to build JCrashPack and evaluation of EvoCrash with ExRunner. And review of the paper describing the work before its submission to the Empirical Software Engineering journal.	INRIA

Task 3.3 - Test amplification for anomalies replication (m14-m23)

We reimplemented the EvoCrash approach in Botsing, an extendable and license friendly framework for crash reproduction and test case generation. We extended Botsing to use test and behavioral models (see T3.2 for the generation of such models) as *seeds*. Botsing uses these seeds for crash reproduction to enhance its reproduction capabilities. Finally, we redefined the fitness function to take advantage of the enhanced code instrumentation defined in T3.2.

Contribution	**Partner(s)**
Implementation of corrective actions after feedback from the intermediate review: revision of D33. Redevelopment from scratch of the crash replication tool as a framework including the different tools developed within WP3 (https://github.com/STAMP-project/botsing). Implementation and evaluation of model seeding for crash reproduction (first release in https://github.com/STAMP-project/botsing/releases/tag/botsing-1.0.4), and reporting in a journal article submitted to ICSE'18 and rejected. Extension of the article under submission at the Software Testing, Verification & Reliability (STVR) journal. Coordination of the external contributions to the Botsing implementation (see pull requests #42, 43, 49, 50, 51 in https://github.com/STAMP-project/botsing/pulls). Redefinition of the fitness function to use the new code instrumentation (https://github.com/STAMP-project/botsing/pull/86). Definition of exception specific fitness functions (extension of the instrumentation is available in https://github.com/STAMP-project/evosuite-ramp/pull/1, redefinition of the fitness function is currently under development). This work is performed by Shang Xiang, a master student at TU Delft, planned to graduate in February 2020.	TUDelft
Update and refinement of demo https://github.com/STAMP-project/EvoCrash-demo/commits/master	Activeeon
In the various development phases ENG assured the compatibility of Botsing with the maven plugin, especially during the refactoring from Evocrash.	ENG
Discussion and participation in the definition of seeding strategies. Discussion and participation in the design of Botsing and support for its continuous integration in Travis.	INRIA/USTL
Evaluation of Botsing on Joram project: see explanations at https://gitlab.ow2.org/stamp/joram/wikis/Botsing-experiments-(flaky-tests) Integration of Botsing with STAMP ci-tools, as a Gitlab web hook: see https://github.com/STAMP-project/botsing-git-webapp	OW2
Discussion and participation in the definition of seeding strategies. Discussion and participation in the design of Botsing and support for its documentation on a public website.	SINTEF

Collecting and analyzing stack traces from TelluCloud logs. Interacting with developers on tools. Working with developers to find strategies to replicate crashes.	TellU

Task 3.4 Test amplification for common behaviors (m24-m32)

Behavioral model seeding (see T3.2 for the generation of such models) has been applied to test case generation using RAMP (available in https://github.com/STAMP-project/evosuite-ramp). These test cases focus on common, non-exceptional behavior.

We evaluated the effect of model seeding on unit test generation through an empirical evaluation of RAMP with behavioral model seeding, compared to EvoSuite without behavioral model seeding.

Contribution	**Partner(s)**
Implementation of behavioral model seeding in RAMP for unit test generation using common behavioral usage of objects and first evaluation of the approach (https://github.com/STAMP-project/evosuite-ramp). Coordination of the full evaluation of behavioral model seeding in RAMP, performed by Activeeon (https://github.com/STAMP-project/evosuite-model-seeding-empirical-evaluation). Definition and development of unit test generation approach common/uncommon behavior, based on execution frequencies of the different blocks of code observed at runtime (https://github.com/STAMP-project/evosuite/tree/cub-test-gen). This work is performed by Björn Evers, a master student at TU Delft, planned to graduate in March 2020.	TUDelft
Evaluation of behavioral model seeding in RAMP.	Activeeon
Discussion and participation in the definition of seeding strategies. Help in the design of the evaluation protocol for unit test generation using model seeding.	INRIA/USTL
Evaluation of RAMP on Joram project. 2 JUnit test cases added with 69 tests: see explanations at https://gitlab.ow2.org/stamp/joram/wikis/Botsing-experiments-(flaky-tests), and commit at https://gitlab.ow2.org/stamp/joram/commit/64effbe7adc58b9617704c3cd64447f91cff4ce1 . Overall statistics reveal an increase of 5,7% line coverage, and more than 10% of mutation coverage, making model-seeding more efficient than Botsing alone (see https://gitlab.ow2.org/stamp/joram/blob/master/descartes-reports/JORAM_KPI.png , comparing lines "Amplified without model seeding" and "All tests").	OW2
Discussion and participation in the definition of seeding strategies. Help in the identification of sources available to collect observations about common behavior.	SINTEF
Comparative test generation experiments with RAMP and without model seeding, to compare the approaches and analyse differences.	TellU

Task 3.5 Strategies for speeding-up runtime amplification (m12-m36)

This task is focused on the optimization of Botsing and the analysis of potential strategies to speed-up the amplification. The activities started from an evaluation performed by several partners to define a list of requirements to ease the usage and speed up the overall execution of Botsing for crash reproduction.

According to the results of this evaluation, a parallel version of Botsing that tried to reproduce a stack trace a given amount of time was developed and released. Furthermore further evaluations on the application of genetic algorithms were produced and documented.

Contribution	**Partner(s)**
Analysis on the optimization strategies to be applied on Botsing Design, Implementation, testing and documentation of Botsing Parallel Module Analysis on the possibility to use distributed genetic algorithms to increase the performance of Botsing (results included in D3.5)	ENG
On site (visit at Atos on 7/10/19 and during STAMP executive committee meetings) + remote support for the deployment and usage of Botsing and RAMP. Reporting of the results from the partners is available at https://github.com/STAMP-project/botsing-usecases-output. Definition of requirements (https://github.com/STAMP-project/botsing-parallel/issues/1) + coordination of the development of a parallel version of Botsing (https://github.com/STAMP-project/botsing-parallel). Development of ExRunner-bash (https://github.com/STAMP-project/ExRunner-bash), a command line version of ExRunner developed in T3.2, allowing high parallelization of Botsing for crash reproduction.	TUDelft

Development Status

Tool is available in Github:

- Botsing: https://github.com/STAMP-project/botsing
- RAMP: https://github.com/STAMP-project/evosuite-ramp
- Botsing-parallel: https://github.com/STAMP-project/botsing-parallel

The following table summarizes the development for the second period.

	Botsing		**RAMP**		**Botsing-parallel**	
	31-May-18	**30-Nov-19**	**31-May-18**	**30-Nov-19**	**31-May-18**	**30-Nov-19**
# commits since the previous version	0	733	0	42	0	26
# changes	0	24,503 insertions, 5 deletions	0	4,180 insertions, 1,731 deletions	0	6,316 insertions, 2 deletions
# files	0	253 files changed	0	104 files changed	0	79 files changed
# LOC	0	20,232	0	N.A.[2]	0	438

[2] Since we extended EvoSuite, it is not possible to accurately report the number of lines of codes. The difference between the last commit after the fork and the HEAD is negative (863,320 LOC - 862,862 LOC).

Conceptual increment (innovation integrated in each tool during the project)

Botsing

- Search-based crash reproduction using
  - weighted sum and a guided genetic algorithm
  - multi-objectivization
  - integration testing and a many objectives algorithm
- Extension of the code instrumentation for
  - integration testing
  - the definition of exception specific fitness functions
- Preprocessing of stack traces to ease crash reproduction
- Adaptation of test seeding to crash reproduction
- Learning of objects usages from static and dynamic analysis of the source code
- Usage of model seeding to improve crash reproduction
- Empirical (quantitative and qualitative) evaluations of search-based crash reproduction using JCrashPack + ExRunner
- Qualitative assessment with STAMP use case partners

RAMP

- Usage of model seeding for unit test generation
- Empirical evaluation of model seeding for unit test generation on SF110
- Generation of unit tests, based on the common and uncommon execution paths
- Qualitative assessment with STAMP use case partners

Target users

Botsing

- Software developers to reproduce a crash for debugging purposes
- Researchers to experiment with search-based crash reproduction

RAMP

- Software developers to automatically generate test cases using common patterns of object usages

1.2.4 Work package 4

Work package 4 is focused on the integration of the three amplification services in different tool chains and on the documentation. During this reporting period the work was performed on the complete or almost-complete versions of the tools in order to obtain a complete and consistent CI/CD scenario.

Main achievements:

- collaborative platform and tools up, running and maintained
- Maven plugins for each tool released
- plugin for Botsing (Jira, GitHub) released
- CI/CD processes defined and Jenkins pipelines released
- documentation, courseware and Docker image for testing purposes
- integrated demo.

Task 4.1 - Collaborative Software Engineering Platform setup and management (m1-m36)

This task is focused on the collaborative platform to support all the project activities, including build and test of STAMP assets. This task originally ended at M12 but was extended to embrace the entire project.

Contribution	**Partner(s)**

Maintained and hosted platform (as achieved in first period), along the whole STAMP project.	OW2
Test collaborative tools and provide feedback.	Activeeon
DSpot and Descartes Jenkins plugins adopted in the Atos Supersede IF Jenkins CI. Testing of some aspects of the collaborative platform, focusing on CI/CD pipeline and JIRA tracker.	Atos
Components installation ● installation of all STAMP tools on the Demo Server ● integrated environment available Day by day support on the infrastructure ● support on the operational platform ● maintenance of STAMP Platform (STAMP tools updates, OS Updates) ● preparation of the environment to support the new components (pre requirements, including Docker, Jenkins, SSL and Python 3) Installation of Jira Software with perpetual license for 10 users Installation and updates of Botsing Server and Apache MQ instance	ENG
Maintained the public and private document references on Github and Gitlab. Maintained the STAMP private wiki (meetings, members, events, etc) and mailing lists.	INRIA
Discussions about requirements from XWiki's point of view about the platform and the tools needed and how they're to be used, in a full development cycle context.	XWiki

Task 4.2 Stamp product architecture definition and implementation (m1-m30)

This task aims at defining the micro services and the API to provide a basis for the project integration. During this reporting period the activities were focused on the definition of CI/CD scenarios on which the integration will be based, and on the installation of the first prototypes.

Contribution	**Partner(s)**
STAMP CI/CD requirement elicitation starting from CI/CD scenario described in 4.3 Definition of CI/CD processes to exploit STAMP features Development of Jenkins pipelines to support CI/CD processes Prototypes of Dspot and Botsing GitHub Server Github integration ● hub client configuration ● Pipeline library snippet based on Github API enabling Github integration without using hub client (according to agreed requirements) Validation tests of CI/CD processes STAMP-enhanced through real projects selected by use case providers New version of pipeline library released, offering the following functionalities: ● search for successful builds among Jenkins build history ● local storage of retrieved builds ● opening pull requests by using GitHub API or Hub client	ENG

Deploy and provide support for ProActive Workflow and Scheduling to be used by other partners Develop and maintain Botsing-gradle-plugin Test and provide feedback about integration tools	ActiveEon
Proof of concept on the microservice architecture on Descartes micro-service (Github application): https://github.com/STAMP-project/descartes-github-app Technical support and advice for ENG to implement a github app for Descartes and DSpot	INRIA/UR1

Task 4.3 STAMP assets integration in various software factory (m1-m36)

This task aims at the production of the plugins supporting the activities of the tools in an integrated environment and on the integration of STAMP assets. During this reporting period the integration was completed and lugins were released to support the tools on Jenkins and GitHub.

Contribution	**Partner(s)**
STAMP integration plugins: ● DSpot Jenkins plugin completed and released ● Botsing Maven plugin completed and released: ○ integration with all the provided Botsing Modules (pre-processing, server and common behavior) ● Botsing Jira plugin ● Finalization of Descartes Jenkins Plugin Botsing Server ● integration of Botsing Github App with Botsing Jira App, released during this reporting period ● Botsing Microservice, including ○ queue management system (based on ActiveMQ) ○ support for Botsing Maven Plugin at GitHub App side Integration, use cases and demo ● collaboration with OW2 to analyse Lutece as potential Use Case for Camp ● Dspot Diff integrated in Jenkins Pipeline ● CAMP in STAMP Demo Server ● Finalization of the demo	ENG
Development updates in the DSpot, Descartes and Botsing Eclipse IDE Jenkins plugins: support for new input parameters, behavioral model seeding in Botsing, updates on tool version, bug fixing Development of the RAMP Eclipse IDE, with support for behavioral model seeding. Testing STAMP plugins for Eclipse IDE. Integration of Descartes plugin with JIRA issue tracker Evidence:	Atos

● see commits in STAMP IDE repository: https://github.com/STAMP-project/stamp-ide ● see reported issues in STAMP IDE Github tracker: https://github.com/STAMP-project/stamp-ide	
Provided expertise in Gradle, especially test that clover is not usable in a multi-module project with Gradle	Activeeon
Provided expertise in Maven by reviewing and commenting on the chosen implementation and especially integrations in Maven builds and CI.	XWiki Romania

Task 4.4 Stamp assets documentation (m12-m36)

This task is focused on the documentation of the produced assets. During this reporting period the last produced tools were documented and all the project documentation was organized in a common, consistent and easy-to-access location.

Contribution	**Partner(s)**
new STAMP-CI repo (obtained starting from the old STAMP-Jenkins repo) to host contributions useful for CI/CD purposes (code snippets, docs, libraries, pipelines, etc) from all partners STAMP DevOps Docker image ● full-stack Docker image containing Jenkins CI Server, CAMP, Python3, Maven - used to execute DSpot/Descartes/PitMP/RAMP ● released in DockerHub ● documentation and support for STAMP adopters to use the Docker image to test STAMP Documentation and courseware on STAMP components ● the documentation has been produced by ENG or collected from the other partners ● all the documentation has been organized in a dedicated repository (https://github.com/STAMP-project/stamp-courseware)	ENG
Descartes Tutorial at ASE'18 (https://www.slideshare.net/OscarLuisVeraPrez/let-the-ci-spot-the-holes-in-tested-code-with-the-descartes-tool-122306608) Mutation Testing Workshop at Ericsson Stockholm (https://www.slideshare.net/stamp-project/mutation-testing-workshop-at-ericsson-kista-sweden) Development of teaching material on STAMP research and technologies (https://github.com/KTH/devops-course/ ; https://oscarlvp.github.io/#teaching ; https://github.com/Software-Testing) Support for the development of the various plugins related to WP1 tools developed within WP4.	INRIA / KTH
Tutorial on Botsing (https://github.com/STAMP-project/botsing-demo) and RAMP (https://github.com/STAMP-project/evosuite-ramp-tutorial). Development of teaching material on STAMP research and technologies (https://github.com/STAMP-project/stamp-courseware).	TUDelft

Support for the development of the various plugins related to WP3 tools developed within WP4.	

Development Status

Tools, docs and courseware are available in GitHub:

- DSpot Maven Plugin: https://github.com/STAMP-project/dspot/tree/master/dspot-maven
- PitMP: https://github.com/STAMP-project/pitmp-maven-plugin
- Botsing Maven Plugin: https://github.com/STAMP-project/botsing/tree/master/botsing-maven
- Botsing Gradle Plugin: https://github.com/STAMP-project/botsing-gradle-plugin
- Stamp CI (Jenkins plugin, pipelines, CI/CD libraries, etc): https://github.com/STAMP-project/stamp-ci
- STAMP Pipeline library: https://github.com/STAMP-project/pipeline-library
- Botsing Server: https://github.com/STAMP-project/botsing-server
- Botsing Jira plugin: https://github.com/STAMP-project/botsing-jira-plugin
- STAMP IDE: https://github.com/STAMP-project/stamp-ide
- Documentation & Courseware: https://github.com/STAMP-project/stamp-courseware/

Following tables summarize the development for the second period:

	DSpot Maven Plugin		**PitMP**		**Botsing Maven Plugin**	
	31-May-18	**30-Nov-19**	**31-May-18**	**30-Nov-19**	**31-May-18**	**30-Nov-19**
# commits since the previous version	135	308	196	90	0	586
# changes (git diff)	2 files changed, 287 insertions(+)	18 files changed, 641 insertions(+), 263 deletions(-)	90 files changed, 5885 insertions(+), 2 deletions(-)	117 files changed, 7797 insertions(+), 2556 deletions(-)	0	14 files changed, 2074 insertions(+)
# files	25	26	69	340	0	104
# LOC	2742	2730	3558	24231	0	12491

	Botsing Gradle Plugin		**STAMP CI**		**STAMP Pipeline Library**		**Botsing Server**	
	31-May-18	**30-Nov-19**	**31-May-18**	**30-Nov-19**	**31-May-18**	**30-Nov-19**	**31-May-18**	**30-Nov-19**

# commits since the previous version	0	283	0	85	0		0	63
# changes (git diff)	0	26 files changed, 1507 insertions (+), 106 deletions(-)	0	180 files changed, 16699 insertions (+), 161 deletions(-)	0	5 files changed, 130 insertions (+), 18 deletions(-)	0	88 files changed, 4304 insertions (+)
# files	0	13	0	674	0	3	0	85
# LOC	0	865	0	97071	0	97	0	2957

	Botsing Jira		**STAMP IDE**		**Docs & Courseware**	
	31-May-18	**30-Nov-19**	**31-May-18**	**30-Nov-19**	**31-May-18**	**30-Nov-19**
# commits since the previous version	0	16	294	326	0	
# changes (git diff)	0	34 files changed, 2522 insertions(+), 2 deletions(-)	102 files changed, 3965 insertions(+), 746 deletions(-)	274 files changed, 13802 insertions(+), 749 deletions(-)	0	17 files changed, 233 insertions(+), 2 deletions(-)
# files	0	31	47	153	0	11
# LOC	0	2221	2530	11881	0	172

1.2.5 Work package 5

During this reporting period, WP5 has conducted the third empirical industrial evaluation of the STAMP tools.

Main achievements:

- Empirical evaluation of all STAMP technologies and tools (e.g. DSpot, Descartes, Botsing, RAMP), CI/CD services, pipelines, STAMP IDE and other interfaces for each of the five industrial cases
- Computation of KPI metrics, based on conducted experiments.

- Quantitative and qualitative evaluation reporting in D5.7

Task 5.1 Industrial requirements and metrics for validation (m1-m20)

Contribution	Partner(s)
Full review of KPIs and metrics after Brussell's EU review. Added new metrics for existing KPIs, added explanations for removed KPI, added new KPIs. All this to improve adequacy of KPIs with scope of project and to get more interesting results. Organized and led various meetings to discuss and agree about KPIs and measurements.	XWiki/XWiki Romania
Contributions to D5.3: description of methods for computing some KPIs Evidence: see D5.3	ATOS
Provide a description of the KPI computation for the STAMP tools in Activeeon process	Activeeon
Provided support on the usage of tools for unit test amplification and collected feedback on unit testing.	INRIA
Provided feedback and helped refine KPI definitions toward more genericity: OW2 applies STAMP tools to projects from 3rd-party developers, which requires more automation than applying them to projects of your own.	OW2
Provide guidance and assistance to case study providers around CAMP	SINTEF
Contributed to deliverable D5.3, describing metrics.	TellU
Contributions to the definition of KPIs, associated metrics and measurement methods. Development of a web crawler to collect stack traces in JIRA (https://github.com/STAMP-project/jira-stacktrace-crawler/). Participation in discussions related to KPI definition in periodic meetings and general assemblies.	TUDelft

Task 5.2 Validation Roadmap and Framework (m3-m20)

Contribution	Partner(s)
WP leadership activities: ● preparation of regular meetings ● preparation of GA WP5 slots and presentations ● preparation of reviews, WP5 slot and presentations ● defense of WP5 at reviews Editor and main contributor to D5.4 Evidence: see D5.4	ATOS
Evaluate KPI with several tools (sonar, jacoco, clover) in order to find the most relevant to gather the KPI	ActiveEon
Focus use-cases in cooperation with OW2 community: most work done on Joram and Lutece projects, as we obtained pro-bono help by development teams. Development of integration tools (Gitlab / Botsing / Descartes / DSpot integration gateways and APIs, in stamp-ci repository).	OW2

Organization of workshops and dissemination events.	
Planning validation phases for TelluCloud use case, participation in meetings and discussions.	TellU
Provided explicit strategies, tools and usage of tools to validate the various KPIs. Participated in all discussions.	XWiki/XWiki Romania

Task 5.3 ProActive Scheduling and Workflows (ActiveEon) case validation (m6-m36)

Contribution	**Partner(s)**
Integrate Dspot in the pipeline, analysed results and provide feedback about the generated tests. Use CAMP in order to run test suites and find configuration bugs. Report the found bugs, fix them and validate CAMP business value for Activeeon. Evaluate and report the last Botsing version on the library programming and report the results Generation of tests thanks to behavioral model seeding in Evosuite. Analysed those tests and report their impact on Activeeon software test suite.	ActiveEon
Participated in online working sessions to help partner to interpret DSpot and Descartes results and collected feedback on unit testing.	INRIA
Provide guidance and assistance to case study providers around CAMP using ad-hoc discussion between Franck Chauvel, Brice Morin (SINTEF) and Mohammed Boussaa (Activeon). Fix issues related to the project (Issue 46).	SINTEF
Live coding sessions during the STAMP meetings. Provided support trough issue trackers of the STAMP GitHub organisation and private STAMP Gitlab. Remote assistance and support through video conference calls. Collection of results of EvoCrash/Botsing in https://github.com/STAMP-project/evocrash-usecases-output.	TUDelft

Task 5.4 FIWARE Ecosystem (ATOS) case validation (m6-m36)

Contribution	**Partner(s)**
Development of Atos UC: ● Development of required artefacts for STAMP tools adoption ● Development of CI/CD: Jenkins jobs and pipelines ● Development of Docker descriptors (including composition) for: ○ CityGo deployment ○ Botsing sandbox experimentation for IF ○ Jenkins CI/CD ● Development of stress (JMeter/BlazeMeter) and functional tests (Selenium) ● Evaluation of STAMP tools: ○ Experiments, data gathering, analysis and reporting ○ Contributions to D5.6. Leading D5.7	Atos/Atos IT

Evidence: ● See STAMP Gitlab repository for Atos CityGo UC developments (private repository): https://gitlab.atosresearch.eu/ari/stamp_docker_citygoApp ● See Supersede GitHub integration repository for Atos Supersede IF development (stamp-baseline and stamp-treatment branches): https://github.com/supersede-project/integration ● See Atos Jenkins CI for both Supersede IF and CityGo (private CI): https://62.14.219.13:7777/ ● See D5.6 and D5.7 ● See STAMP GitHub trackers for DSpot, Descartes, CAMP and Botsing (and other tools). Look for issues opened by Atos team ● See STAMP GitHub repositories for UC reporting: <tool>-usecase-output (tool = dspot, descartes, camp, botsing)	
Technical support to Atos for using DSpot and Descartes tools.	INRIA
Provide guidance and assistance to case study providers around CAMP using impromptu discussion and vision conference between Franck Chauvel (SINTEF and Fernando Mendez Requena (AtoS) and Jesús Gorroñogoitia Cruz (AtoS) Especially, add new features (Releases v0.4.0, v0.5.0, v0.6.0, v0.7.0) and fix bugs related to the AtoS case-study (Issues 74, 75, 78, 79).	SINTEF

Task 5.5 TellU case validation (m6-m36)

Contribution	**Partner(s)**
The period includes the two main validation phases, reported in D5.6 and D5.7 respectively. ● Testing of Descartes on three use case projects, with fixing raised issues and reporting, collecting KPIs 1 and 3. ● Testing of DSpot, reporting issues, collecting KPIs 1 and 3. ● Working on DSpot Windows compatibility ● Looking for and fixing flaky tests, collecting KPI 2. ● TelluCloud deployment configuration and testing. ● System tests, test automation. ● Testing CAMP on TelluCloud Docker configuration, collecting KPIs 4, 5, 6. ● Testing CAMP on TelluCloud Kubernetes configuration, collecting KPIs 4, 5, 6. ● Collecting production stack traces, validation of Botsing, collecting KPI 8. ● Validation of Botsing with introduced errors. ● Testing of RAMP on use case projects, generating tests, collecting KPIs 1, 3 and 9. ● Test reporting, calculating global metrics and changes, D5.6 and D5.7.	TellU
Support Tellu to evaluate CAMP on Kubernetes deployments	SINTEF
Live coding sessions during the STAMP meetings. Provided support trough issue trackers of the STAMP GitHub organisation and private STAMP Gitlab.	TUDelft

Remote assistance and support through video conference calls. Collection of results of EvoCrash/Botsing in https://github.com/STAMP-project/evocrash-usecases-output.	

Task 5.6 XWiki case validation (m6-m36)

Contribution	**Partner(s)**
Continued testing the various tools (Descartes, DSpot, CAMP/TestContainers, EvoCrash/Botsing, RAMP) on the XWiki use case. Computed KPIs and metrics. Provided feedback to tool developers to improve the tools. Migrated tests to JUnit5 and tested the tools on JUnit5. Migrated functional tests to Docker to be able to test them on CAMP/TestContainers. Created configurations for CAMP/TestContainers.	XWiki/XWiki Romania
Support for DSpot on commit and Descartes in the CI	INRIA
Live coding sessions during the STAMP meetings. Provided support trough issue trackers of the STAMP GitHub organisation and private STAMP Gitlab. Remote assistance and support through video conference calls. Collection of results of EvoCrash/Botsing in https://github.com/STAMP-project/evocrash-usecases-output.	TUDelft

Task 5.7 OW2 case validation

Contribution	**Partner(s)**
Sat4j use case: we conducted deeper experimentations on Dspot and Descartes to investigate the maximum number of configurations available. We put the focus on the three first KPIs on this use case. The main challenge is to improve code coverage (K01) because of the modular implementation of the core engine. We worked closely with DSpot team in order to obtain relevant amplified tests. Provided feedback from Descartes experimentations which lead to new commits in the software development repository. Sat4j is the use case covered in the D5.6 deliverable. Lutece use case: we conducted CAMP experimentation, working in close cooperation with both Lutece and CAMP teams. We identified two strategies to validate the tool: unit test strategy on amplified configurations on one hand and performance test strategy on the other. We also conducted with Lutece some experimentations with the three other tools: DSpot, Descartes and Botsing. Integrated the work on the CI pipeline with the cooperation of ENG, three netmeetings between OW2 and ENG in order to define the strategy. Authzforce use case: Descartes, DSpot and Botsing experimentations. Joram use-case: Descartes experimentations lead to bug fixes (issue filed then solved by Joram team). DSpot partially fixes the bugs, some being due to insufficient tests (but human-developed tests appear better, because of more legible code). Sample of how this was conducted published as a tutorial on STAMP web site. Development of new tests (68 classes with about 250 test added), and experiments with Descartes and	OW2

Botsing/RAMP to increase KPIs. Work along with Joram team (pro-bono help, and bug fixing upon issues).	
Support OW2 with applying CAMP to the Lutece application, using point to point interaction between Franck Chauvel from SINTEF and Assad Montasser from (OW2). Add features (Releases 0.7.0, 0.8.0) and fix bugs blocking the case-study (Issues 82 and 98).	SINTEF

1.2.6 Work package 6

Main achievements:

- Communication strongly reinforced: multiple conferences and industrial hands-on workshops, online resources including interviews and videos, social network presence.
- STAMP disseminated in many industrial events, including major ones (EclipseCon, OSCON…), and scientific workshops (with papers accepted).
- Beta-testing campaign conducted, with 12 industrial participations validated.

Task 6.1 Communication (m1-m36)

Contribution	**Partner(s)**
Kept developing and updating the project web site and maintaining flow on social networks. Conducted new interviews bringing their own angles on the project, including external software testing experts. Supported technical postings on the website. Finalized the production of a 3' video primer posted to the web site and used on industry event booths. Increased face-to-face communication during the second half of the project though 15 conferences and 16 workshops focused on testing automation. Communication toward the press and social media generated four press releases resulting in two full featured articles in major industry publications and ten articles posted on online publications, 600 tweets resulting in 360 followers, and 22 discussion topics on LinkedIn. Industry and public IT interactions such as Sopra Steria and EC DGIT are direct results of these communication efforts.	OW2
Maintaining the internal wiki, managing the private and public document references.	INRIA

Task 6.2 Dissemination (m1-m36)

Contribution	**Partner(s)**
Arranged public showcases (presentations, papers, booths) at industry events such as Devoxx, EclipseCon, FOSDEM, OW2con, OSCON, Paris Open Source Summit, etc. During the second half of the project, 13 public talks were offered during 14 industry and open source conferences. In three years, the STAMP concepts and tools were showcased at 27 industry events worldwide. Launched the STAMP beta testing campaign with dedicated dissemination material including web pages, questionnaire, flyer, mailings, webinars with STAMP demos and workshops with early users (EC-DGIT, Ericsson, Orange, Sopra-Steria, etc.). A dozen of feedback were received from developers in the industry, e-commerce, system integrators and academia.	OW2
Presented how Activeeon use STAMP tool at a Telecom Valley talk.	ActiveEon

Presented how CAMP helped Activeeon to do configuration testing at Station F.	
Talks and workshops in industry conferences and scientific event. Organized presentations and hands-on session with industry companies. Contributions to presentation slideshows, interviews and videos. Papers for scientific journals and articles for consumer magazines. Master courses on test automation and amplification. Contribute to social media news (Twitter, LinkedIn, Medium, etc).	INRIA
CAMP paper accepted at ISSRE'19 and presented by B. Morin CAMP tutorial at STAMP Workshop in Nice/Sophia Antipolis (Jan. 2019) by F. Chauvel CAMP tutorial at OW2 con in Paris (Jun. 2019) by F. Chauvel CAMP tutorial in the A-Test workshop at EFSE 2019 (Aug. 2019) by F. Chauvel CAMP tutorial as part of the Inria Tech Talk (Feb. 2019) by E. Garcia.	SINTEF
Teaching the rest of Tellu about Descartes and other STAMP tools, informing partners.	TellU
Planning and organization of scientific workshops (A-MOST '18 - https://amost2018.wordpress.com; MASES '18 - https://mases18.github.io; EASEAI '19 - https://easeai.github.io; MaLTeSQuE '19 - https://maltesque2019.github.io). Participated in a scientific workshop in The Netherlands (Lorentz Workshop: In-Vivo Analytics for Big Software Quality - http://www.lorentzcenter.nl/lc/web/2018/1008/info.php3?wsid=1008&venue=Oort). Contributed to regular scientific conferences and scientific papers (https://www.stamp-project.eu/view/main/publications). Shared social media news. Planning and organization of the Dutch Test Day 2018 (https://testdag2018.github.io), an industrial conference around the software testing industry in the Netherlands (with ~130 participants).	TUDelft
Conferences talks about STAMP projects for the period: ● Paris JUG: "New types of tests for Java projects" (http://massol.myxwiki.org/xwiki/bin/view/Activities/Data/ParisJUG2018Mutation/) ● Lecture at KTH university about Building XWiki and how the STAMP tools are used (http://massol.myxwiki.org/xwiki/bin/view/Activities/Data/KTHLectureCI2019/) ● Devoxx France 2019: "Configuration Testing with Docker & TestContainers" (http://massol.myxwiki.org/xwiki/bin/view/Activities/Data/TestContainersDevoxx2019/) ● Planned and upcoming: "New types of tests for Java projects" at the Montreal JUG at end of year. Blog posts about topics developed during STAMP for the period: ● "Automatic Test Generation with DSpot" (http://massol.myxwiki.org/xwiki/bin/view/Blog/TestGenerationDspot**)** ● "Resolving Maven Artifacts with ShrinkWrap... or not" (http://massol.myxwiki.org/xwiki/bin/view/Blog/ResolveMavenArtifactsShrinkWrap) ● "Global vs Local Coverage" (http://massol.myxwiki.org/xwiki/bin/view/Blog/GlobalLocalCoverage)	XWiki

• “Scheduled Jenkinsfile” (http://massol.myxwiki.org/xwiki/bin/view/Blog/ScheduledJenkinsfile)	

Task 6.3 Exploitation (m1-m36)

Contribution	**Partner(s)**
Identification, and validation of the value proposition of the different exploitable results obtained in the project. Identification of the intellectual property rights and assessment of the potential of issues. Construction of the software license policy in coordination with the tools and plugin developers. Assessment of this license policy in terms of tools and plugin compatibility as well as commercial exploitability. Identification the individual exploitation plan in coordination with all partners Clarification and validation of the exploitation scheme of the industrial partners, to ensure effective and valuable exploitation of the project results within the time of the project and after the end of the project Assessment of the sustainability of the results after the end of the project https://drive.google.com/open?id=0ByCGGnbDGB7uMW04RVA3OUxyTWc	Activeeon
Business models analysis and definition of best approaches for the project outcomes Support the WP leader in the identification, and validation of the value proposition of STAMP tools IPR management and approaches, licensing analysis and assessment of the potential of issues/compatibilities Atos exploitation plans Presentation of STAMP tools and advantages to other units in Atos such as the QA testing team in Seville and Consulting company for early adoption of tools STAMP outcomes in the roadmap for enriching the Software Development Methodology for Atos Research Unit in the testing chapter. Business plan main editor.	Atos
Included a section on STAMP and test amplification techniques in “Test Process” course at Engineering IT and Management Academy • several sessions performed or planned for the current year Analysis of the improvements introduced by the adoption of the implemented components in ENG production pipelines	ENG
Kept communicating on STAMP within the OW2 community. Further to discussions with members to test the value proposition initially drafted finally developed the internal "black box" business model for long-term exploitation. This approach is being fed back into the use case.	OW2
Working on value proposition for Tellu, contributing to exploitation deliverable, planning exploitation.	TellU
Participated to several meetings and provided exploitation commitments from XWiki point of view. Started implementing those commitments in the XWiki open source project.	XWiki/XWiki Romania

Task 6.4 Market Analysis and business modeling (m18-m36)

Contribution	**Partner(s)**
Identification of the market stakeholders and segments Identification and review of adequate business reports concerning the methodology and market of software testing. Designing and dissemination of internal and external surveys to obtain focused insight on the particular aspects of software testing methodologies addressed by the STAMP project Compilation and consistency checking of the results obtained with both report review and STAMP surveys Competition analysis Assessment of the project directions in regards to this market analysis	Activeeon
Proposal of initial business plan and approaches Industrial adoption options and identification of the market options, segmentation, approaches for using STAMP tools in business Coordinating all actions to obtain focused insights from other H2020 projects such as Elastest Business planning and assessment of the project options in regards to the market analysis and Github marketplace Main contributor in all WP6 deliverables: D6.3, D6.4 and final/chief editor of D6.5	Atos
Presentation of STAMP at OW2 Conference 2018 Preliminary analysis on the business advantages of the introduction of STAMP in ENG processes Development and deployment of Descartes Jenkins mutation report plugin in order to increase the capability of ENG to evaluate the quality of the produced software and of the unit tests (objective of this activity is to evaluate the business impact of this innovation) Definition of the business advantages of the introduction of STAMP in ENG processes (contribution to D6.4)	ENG
	INRIA
Organised face-to-face STAMP workshops at Station-F, OW2con’19, Orange Grenoble, EC-DGIT to meet Java developers. Recruited developers to try the STAMP tools and managed feedback from this beta-testing. Organised three webinars targeted at large technology companies: CGI, Orange, Sopra-Steria. Participated in internal calls to work on business plan metrics.	OW2
Participated to workshop/interviews about business models and community. Market analysis of the GitHub Marketplace ecosystem for code quality. Evaluation of the size of the GitHub Java repositories and interest for a Descartes GitHub app. Poll about interest of such a service on Twitter.	XWiki/XWiki Romania

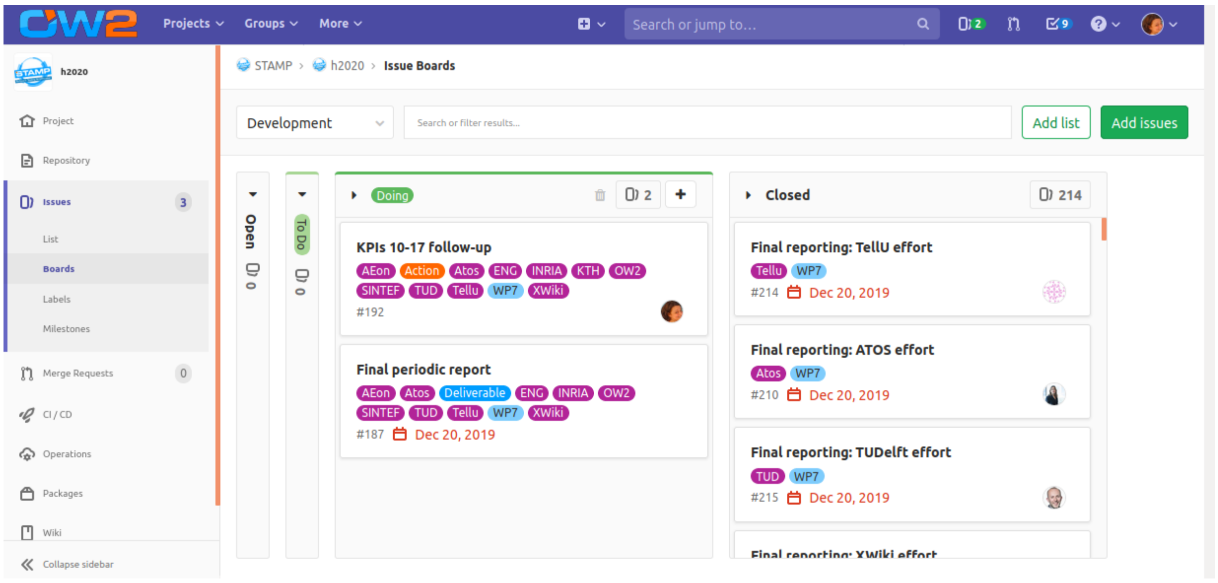
1.2.7 Work package 7

Project tracking

We had:

- weekly calls between KTH and INRIA for the project management and coordination
- monthly plenary calls
- regular calls for WP5 and WP4 coordination
- WP1, WP2 and WP3 and dedicated calls and working sessions by video conferences
- 4 in-person meetings

We have continued to use the Gitlab issue tracker to track deliverables, milestones and actions, and to have a global view of the project:



KPIs 10-17 follow-up
AEon Action Atos ENG INRIA KTH OW2 SINTEF TUD Tellu WP7 XWiki
#192

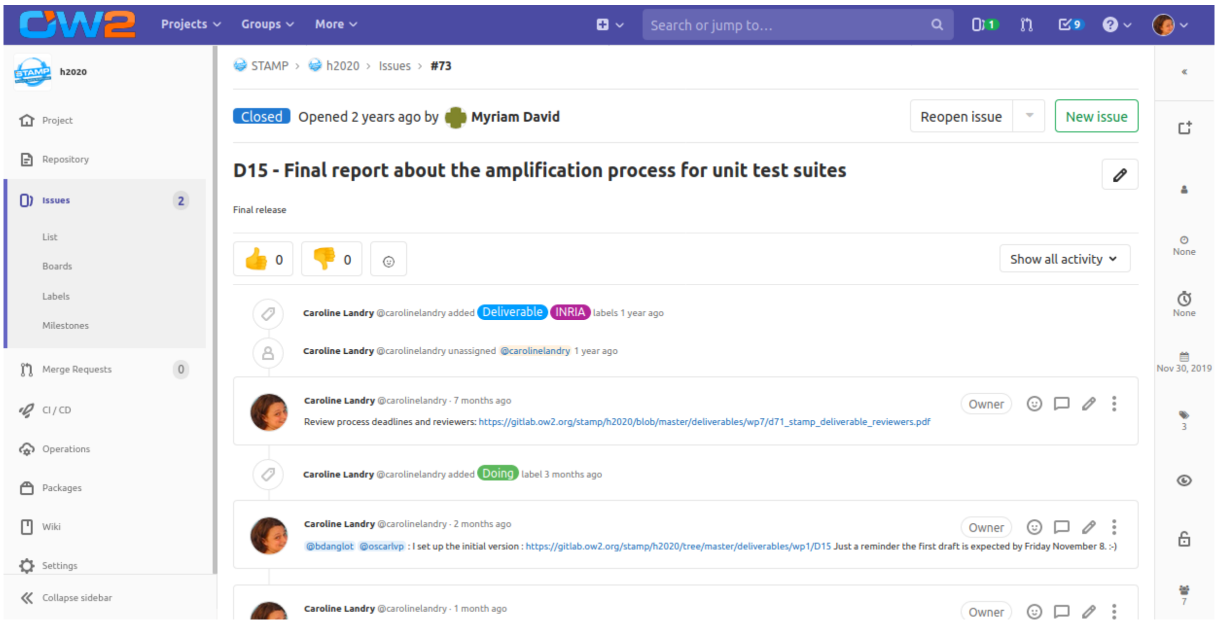
Final periodic report
AEon Atos Deliverable ENG INRIA OW2 SINTEF TUD Tellu WP7 XWiki
#187 Dec 20, 2019

Final reporting: TellU effort
Tellu WP7
#214 Dec 20, 2019

Final reporting: ATOS effort
Atos WP7
#210 Dec 20, 2019

Final reporting: TUDelft effort
TUD WP7
#215 Dec 20, 2019

Final reporting: XWiki effort



Closed Opened 2 years ago by Myriam David

D15 - Final report about the amplification process for unit test suites

Final release

Caroline Landry @carolinelandry added Deliverable INRIA labels 1 year ago

Caroline Landry @carolinelandry unassigned @carolinelandry 1 year ago

Caroline Landry @carolinelandry · 7 months ago Owner
Review process deadlines and reviewers: https://gitlab.ow2.org/stamp/h2020/blob/master/deliverables/wp7/d71_stamp_deliverable_reviewers.pdf

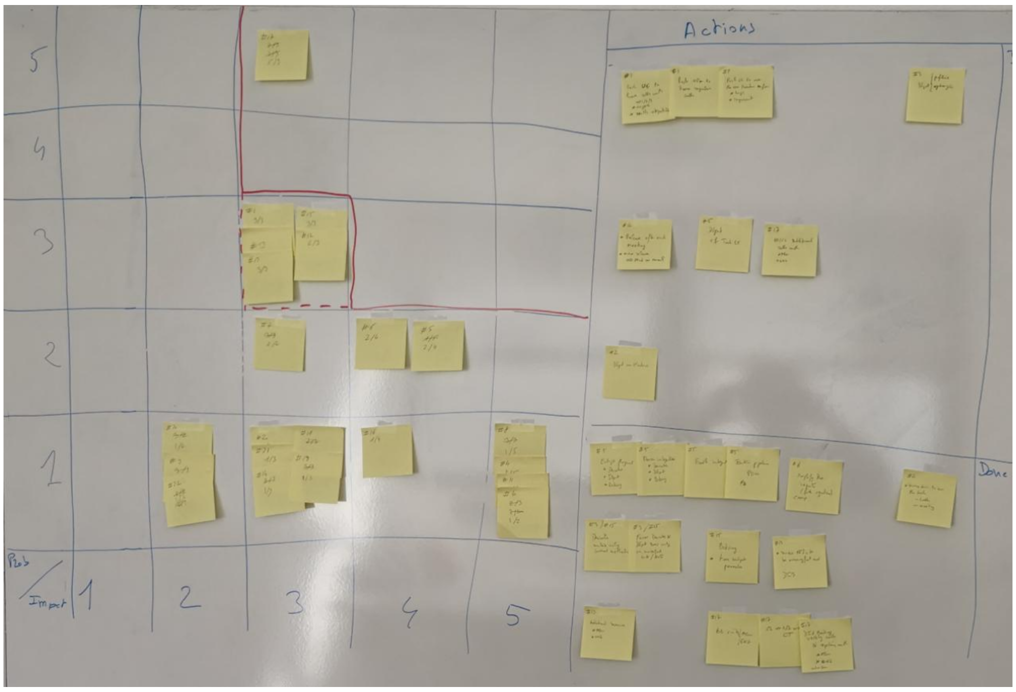
Caroline Landry @carolinelandry added Doing label 3 months ago

Caroline Landry @carolinelandry · 2 months ago Owner
@bdanglot @oscarlvp : I set up the initial version : https://gitlab.ow2.org/stamp/h2020/tree/master/deliverables/wp1/D15 Just a reminder the first draft is expected by Friday November 8. :-)

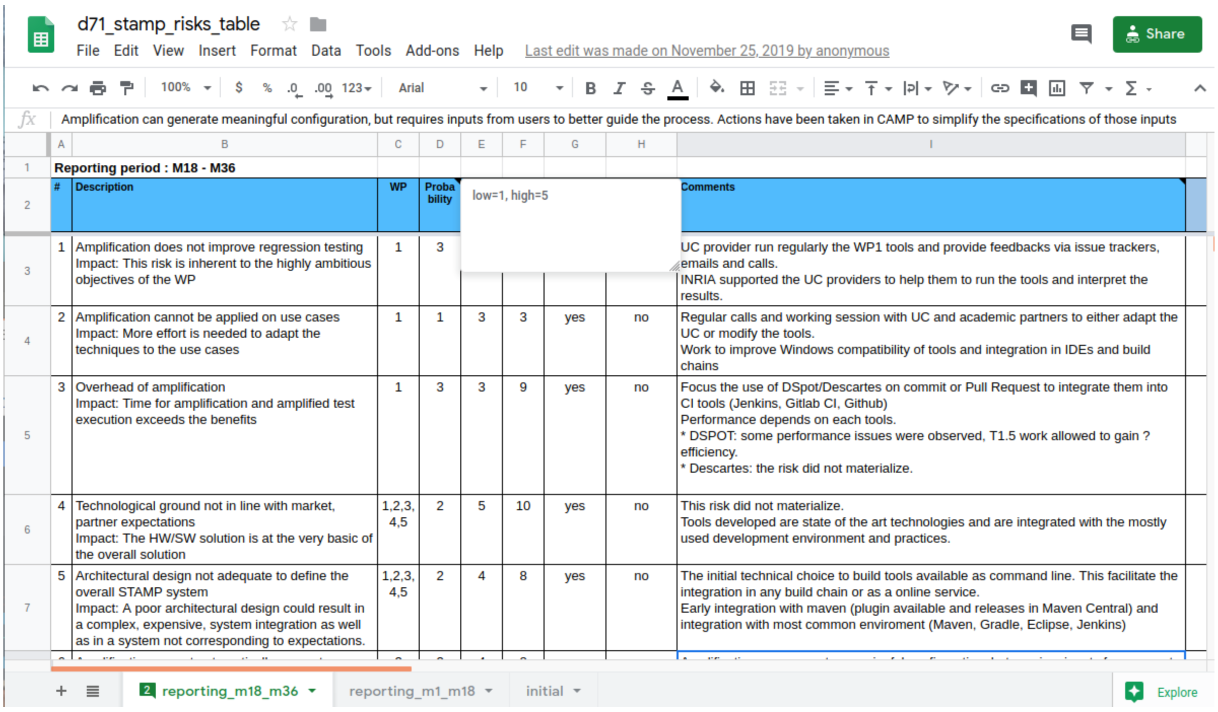
Caroline Landry @carolinelandry · 1 month ago Owner

Risk Management

To follow-up the risks, we jointly used a risk board:



And an online shared document:


d71_stamp_risks_table

File Edit View Insert Format Data Tools Add-ons Help Last edit was made on November 25, 2019 by anonymous

Amplification can generate meaningful configuration, but requires inputs from users to better guide the process. Actions have been taken in CAMP to simplify the specifications of those inputs

Reporting period : M18 - M36

#	Description	WP	Probability					Comments
1	Amplification does not improve regression testing Impact: This risk is inherent to the highly ambitious objectives of the WP	1	3					UC provider run regularly the WP1 tools and provide feedbacks via issue trackers, emails and calls. INRIA supported the UC providers to help them to run the tools and interpret the results.
2	Amplification cannot be applied on use cases Impact: More effort is needed to adapt the techniques to the use cases	1	1	3	3	yes	no	Regular calls and working session with UC and academic partners to either adapt the UC or modify the tools. Work to improve Windows compatibility of tools and integration in IDEs and build chains
3	Overhead of amplification Impact: Time for amplification and amplified test execution exceeds the benefits	1	3	3	9	yes	no	Focus the use of DSpot/Descartes on commit or Pull Request to integrate them into CI tools (Jenkins, Gitlab CI, Github) Performance depends on each tools. * DSPOT: some performance issues were observed, T1.5 work allowed to gain ? efficiency. * Descartes: the risk did not materialize.
4	Technological ground not in line with market, partner expectations Impact: The HW/SW solution is at the very basic of the overall solution	1,2,3,4,5	2	5	10	yes	no	This risk did not materialize. Tools developed are state of the art technologies and are integrated with the mostly used development environment and practices.
5	Architectural design not adequate to define the overall STAMP system Impact: A poor architectural design could result in a complex, expensive, system integration as well as in a system not corresponding to expectations.	1,2,3,4,5	2	4	8	yes	no	The initial technical choice to build tools available as command line. This facilitate the integration in any build chain or as a online service. Early integration with maven (plugin available and releases in Maven Central) and integration with most common enviroment (Maven, Gradle, Eclipse, Jenkins)

low=1, high=5

reporting_m18_m36 reporting_m1_m18 initial Explore

The overall development strategy and mitigation actions have avoided the materialization of most of the risks. The main perturbations we had during the project were the KPIs revision (see WP5) and the resource management for one partner, due to employee resignations.

Activities and tasks not planned and carried out during the 2nd period:

- Amendment
  - Change tasks assignments of T1.4 and 5.4 from ATOS Turkey to ATOS IT
  - Update WP3 objectives, task 3.2, deliverables D3.2 and D3.3 deadline
  - Update WP4 task 4.1
  - Update WP5 project KPIs
  - Updated version of the DoA part A
  - Updated version of the DoA part B

        - Request letter
- Document reviews
    - Revised version of D1.2
    - Revised version of D3.1
    - Revised version of D3.2
- Additional project interim review

Advisory board

We had an advisory board composed of:

- Henry Coles (PIT),
- Francisco Gortazar (University Rey Juan Carlos)
- Manuel Martinez (Nokia)

We have organized meetings with members in conjunction with the plenary meetings:

- December 2017, Madrid
- October 2018, Paris
- January 2019, Sophia

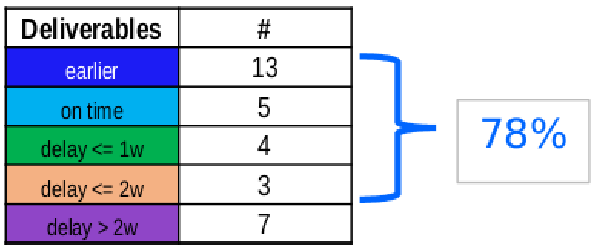
Deliverables and milestones status

The following table summarizes the planned and actual dates for the deliverables:

Deliverable	**Deadline**	**Actual delivery date**	**# days delivery**
D71	31-Jan-17	31-Jan-17	**0**
D61	28-Feb-17	7-Mar-17	**7**
D62	31-Mar-17	27-Mar-17	**-4**
D11	31-May-17	31-May-17	**0**
D21	31-May-17	31-May-17	**0**
D31	31-May-17	31-May-17	**0**
D41	31-May-17	29-May-17	**-2**
D51	31-May-17	31-May-17	**0**
D52	31-Aug-17	11-Sep-17	**11**
D12	30-Nov-17	29-Nov-17	**-1**
D22	30-Nov-17	29-Nov-17	**-1**

D32	30-Nov-17	13-Dec-17	13
D42	31-Jan-18	8-Feb-18	8
D55	31-May-18	22-Jun-18	22
D63	31-May-18	26-Jun-18	26
D13	31-Jul-18	27-Aug-18	27
D23	31-Jul-18	7-Sep-18	38
D53	31-Jul-18	11-Sep-18	42
D54	31-Jul-18	10-Sep-18	41
D33	30-Nov-18	22-Nov-18	**-8**
D43	30-Nov-18	5-Dec-18	**5**
D56	31-May-19	4-Dec-18	**-178**
D64	31-May-19	24-Jun-19	**24**
D14	30-Sep-19	24-Sep-19	**-6**
D24	30-Sep-19	1-Oct-19	**1**
D34	30-Sep-19	1-Oct-19	**1**
D15	30-Nov-19	29-Nov-19	**-1**
D25	30-Nov-19	29-Nov-19	**-1**
D35	30-Nov-19	29-Nov-19	**-1**
D44	30-Nov-19	29-Nov-19	**-1**
D57	30-Nov-19	29-Nov-19	**-1**
D65	30-Nov-19	29-Nov-19	**-1**
		average delay (#days)	**2**

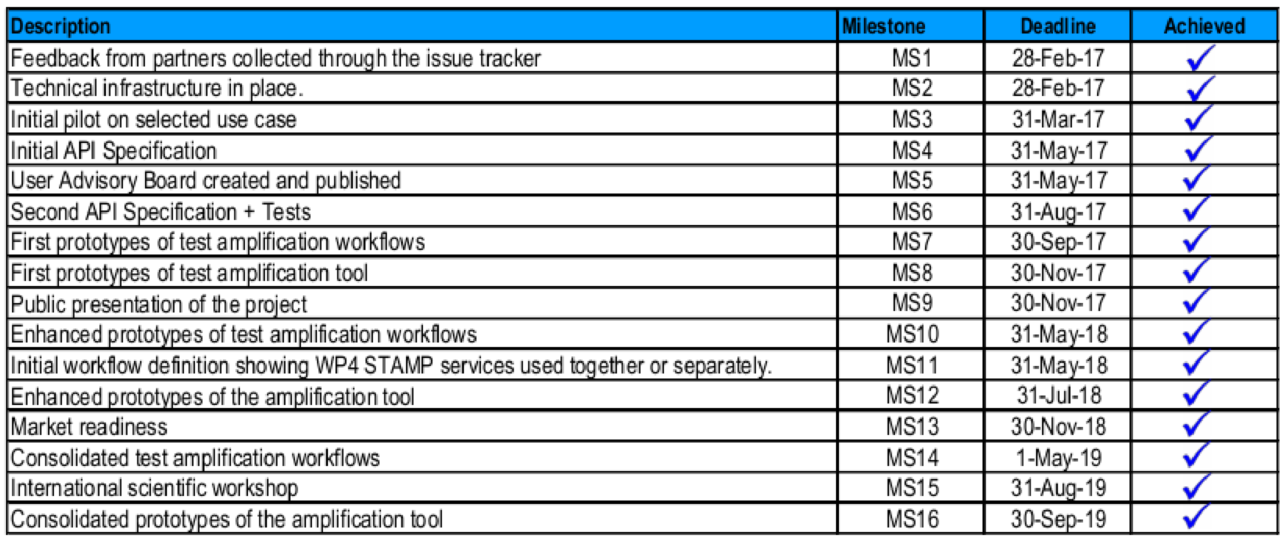
The following graphs summarize the comparison between planned and actual delivery dates:



Deliverables	#
earlier	13
on time	5
delay <= 1w	4
delay <= 2w	3
delay > 2w	7

78%

The following table summarizes the status of the milestones:



Description	Milestone	Deadline	Achieved
Feedback from partners collected through the issue tracker	MS1	28-Feb-17	✓
Technical infrastructure in place.	MS2	28-Feb-17	✓
Initial pilot on selected use case	MS3	31-Mar-17	✓
Initial API Specification	MS4	31-May-17	✓
User Advisory Board created and published	MS5	31-May-17	✓
Second API Specification + Tests	MS6	31-Aug-17	✓
First prototypes of test amplification workflows	MS7	30-Sep-17	✓
First prototypes of test amplification tool	MS8	30-Nov-17	✓
Public presentation of the project	MS9	30-Nov-17	✓
Enhanced prototypes of test amplification workflows	MS10	31-May-18	✓
Initial workflow definition showing WP4 STAMP services used together or separately.	MS11	31-May-18	✓
Enhanced prototypes of the amplification tool	MS12	31-Jul-18	✓
Market readiness	MS13	30-Nov-18	✓
Consolidated test amplification workflows	MS14	1-May-19	✓
International scientific workshop	MS15	31-Aug-19	✓
Consolidated prototypes of the amplification tool	MS16	30-Sep-19	✓

Project KPIs

KPI01 to KPI09, KPI12: related to WP5, see deliverable D57.

KPI13 to KPI17: related to WP6, see section “Update of the plan for exploitation and dissemination of result (if applicable)” of this document.

KPI10 and KP11

Objective(s)	**Description**	**ID**	**target**	**achieved**	**%**
			# services		
4	3 test amplification services integrated in 2 different toolchains	KPI10	3	4	**133 %**
			# toolchains		
		Descartes	2	4	**200 %**
		DSpot	2	4	**200 %**
		CAMP	2	4	**200 %**
		Botsing	2	4	**200 %**
		RAMP	*2*	*1*	50 %
5	Validation of each test amplification service by at least 3 use cases.	**KPI11**			
		Descartes	3	8	**267 %**

		DSpot	3	7	**233 %**
		CAMP	3	5	**167 %**
		Botsing	3	8	**267 %**
		RAMP	3	4	**133 %**

1.3 Impact

	How STAMP addressed it in the second period
Reduction of the time to market of the new generations of software enabled products and services;	Activeeon: STAMP tools increased the quality of our code and our tests during the development phase. They enabled to drastically speed up the quality phase before each release. Indeed highlighting the bugs and fixing them sooner speeds up the whole release process. Moreover, increasing the test quality also reduces the number of bugs introduced by developers, be doing so we also speed up the whole process. Furthermore, it reduces the chance of doing a maintenance release for clients that have blocking non-discovered bugs. Atos: STAMP tools have contributed significantly to reduce the time and effort required to deliver the CityGo system into our customer's infrastructure. In particular, thanks to STAMP the CityGo CI/CD process has been automated in pipelines, for different delivery environments. Moreover, the process to identify optimal delivery configurations has been simplified in a great manner. CityGo can be verified for functional compatibility in a large number of environments. ENG: STAMP tools have been introduced in some pilot projects (using Jenkins-based pipelines) in order to increase the quality of performed tests. The results of the first pilot activities confirmed this expectation: since bugs are caught earlier, production times decreases. In this sense, the advantage will be the possibility to envisage faster delivery times in the next technical proposals. OW2: Increases in code coverage, like those provided by the Descartes + DSpot mix and by Botsing/RAMP (reproducing errors in production, including random ones), significantly improve regression testing. New versions of software get bulletproof at little effort, which we consider a time-to market enhancement: moreover, our tests revealed that these tools are applicable to most projects in our code base (at least, the ones with modern and clean testing). Tellu: DSpot and CAMP have shown clear speed benefits in the TelluCloud use case, producing additional test coverage and increased mutation score with minimal effort. CAMP greatly speeds up the process of deploying and testing a Docker image, so it speeds up component and system tests which can run on Docker deployments. CAMP has also helped explore the configuration space for performance optimization of Kubernetes deployments. STAMP tools have helped the testing of TelluCloud in a period where Tellu has moved towards devops and more frequent releases. XWiki: Thanks to configuration testing, XWiki invested in the creation of Dockerized versions of XWiki. This makes the distribution of XWiki simpler and more powerful, also allowing the XWiki project to go towards Continuous Deployment, and thus reducing the time to market. More generally, the usage of STAMP tools allowed the XWiki project to automate testing more (for example the XWiki project used to test various configurations with manual tests before STAMP) thus reducing the time to market, leading to faster development cycles and the ability to release the software faster.
significant and substantiated productivity increase in all aspects of	Activeeon: Our microservices architecture is hard to test on several configurations and before using STAMP tools, we had to configure a specific virtual machine for doing configuration testing. By using CAMP

software life-cycle especially for distributed systems;	we were able to easily run our test suite against several configurations. It highlighted bugs that could only have been discovered at the end of the release process or on the client environment. Discovering them at the early development stage speeds up our release process and increases our client confidence by providing them software with better quality. Atos: STAMP tools have increased the productivity of the test engineering for both the Supersede and CityGo systems. The development of test cases that cover non-executed lines of code has been largely accelerated thanks to DSpot and RAMP. Additionally, runtime issues not covered by test cases are now covered thanks to Botsing. CAMP has simplified the development of delivery configurations that are functionally compatible with CityGo and it has also accelerated the search for optimal configuration for performance. ENG: This is the main result for ENG. In particular, the introduction of STAMP tools in the pilot projects based on a Jenkins pipeline aimed at increasing the automation and, as a consequence, the productivity. This is confirmed by the results of the pilot activities. OW2: On that point, CAMP and Botsing are key: the former on multi-platform and multi-environment software (complex server projects like Lutece, for example), because multiple environments and configurations can be tested automatically. And the latter because it can cover production issues (crashes) in complex environments, help understand and fix them, and provide tests to avoid regressions (so a fixed issue never appears again). TellU: Following the refactoring into a micro-service architecture early in the STAMP project, and deployment in various cloud infrastructures, testing and test automation has increased in importance to ensure quality and avoid regressions as complexity is increasing. Through STAMP, Tellu has developed a test framework and learned about tools and methodologies which has helped keep productivity up. Test generation by STAMP tools DSpot and Botsing has shown promising results, providing significant test coverage increase with minimal effort. XWiki: Thanks to more test automation and especially on configuration testing, developers are now able to find production-related bugs during the development phase and fix them before the bugs make their way to production. In addition, Botsing supports developers when analysing production stack traces, making it easier to pinpoint the issues.
Ability to meet software quality levels required by a fast growing number of software-enabled products and services;	Activeeon: Having a huge codebase with multiple modules has a direct impact on the code quality. Maintaining and improving projects will be done by several developers. STAMP tools will help future developers to fix bugs and implementing new features without regressions. Indeed, the unit tests generated by Dspot prevent regressions when they are added to the test suite. It ensures that future developers will not break the existing functionalities when adding new features. Moreover Descartes enables to increase the test suite quality, which will also prevent future developers from altering the existing functionalities. Atos: STAMP tools have largely contributed to an increase in the quality confidence of Supersede and CityGo systems: thanks to CAMP's ability to detect configuration related issues, optimal deliveries are now easier; thanks to Descartes it is now possible to detect test suite

	deficiencies and we have improved our test cases; thanks to DSpot, we have incorporated new test cases that exercise non-favorable testing, and thanks to Botsing, we are now able to easily detect anomalous behavior in our test cases. All these actions applied to both the code base and test suites have increased qualitatively the quality levels of our software systems. ENG: even in projects with a high level of automation, testing activities were mainly based on human decisions. The first results of the introduction of STAMP tools in the pilot projects encouraged the automation also on this aspect. It is expected that, as for the pilot projects, in general the products will come out from the pipelines with less bugs, so, better quality. The use of a common production process based on CI/CD concepts for most of the business units will provide further advantages. OW2: Code quality is a big challenge for OW2, as it is very diverse among projects (we host more than 60). We try to harmonize and certify code quality, to gain more users by providing trust: OW2 initiated a process called MRL (market readiness level), that includes technical code quality assessment. STAMP tools are among the tools we use to assess MRL, by recommending them to project leaders, and taking into account STAMP usage to increase technical MRL. Tellu: STAMP validation has demonstrated significant increases in test coverage and quality (mutation score) in the TelluCloud use case. Through use of CAMP, the confidence in running TelluCloud under various environments has also increased. The increase in test scope and quality is reflected in software quality, and has helped Tellu bring out new versions of TelluCloud in the market. XWiki: STAMP allowed the XWiki project to have a lot more automated tests (be them unit tests or functional tests executing on various configurations). This resulted in a large increase in both overall test coverage and test quality (increased mutation score). We already noticed a decrease of issues in XWiki's issue tracker, especially related to various configurations in production.
Increased reuse of code, design or functional requirements in the development of new software.	Activeeon: When starting the project our logging practices were really basic. Indeed all projects shared the same configuration and all logs were stored in one file. By working with Botsing, we improved and increased the way we do logging. We improve our design to have a configuration file for each project and dedicated logging files. This improvement was shared among all the projects and will be reused for new projects. Having better logging practices speed up debugging client issues in production. Atos: The adoption of the STAMP technologies has pushed us to automate the building and the delivery of the CityGo system to different customers, thanks to containerization technology (e.g. Docker). CAMP enables to reuse pre-existing delivery configurations from where we can derive variants that fit to our customer infrastructures. Other STAMP tools (e.g. DSpot, RAMP) have reused our

	existing test suites to amplify them, largely simplifying the maintenance of our QA system. ENG: the perceived advantages produced by the introduction of STAMP will encourage the adoption of CI/CD concepts in software production. Currently such an approach is only partially adopted in some production units: a massive introduction will increase reuse of code, not only for testing. OW2: At OW2, reuse is considered from an outside point of view (external developers reuse code from our projects), and between projects (code reuse inside the community). Introducing code quality assessment in CI/CD processes, as well as in code evaluation metrics, is key for code reuse: one can monitor the quality of what she reuses, and adapt it to her requirements. STAMP tools, and notably Descartes, are of great help to improve such everyday monitoring. Tellu: Test assets such as test code and configurations are reused through amplification. The increased focus on testing and code quality also encourage reuse of tested code over developing new code. XWiki: Most STAMP tools are amplification tools, building on existing assets (existing tests) and generating new ones from them. Thus, existing code (test assets - test cases and test configurations in form of Docker and Docker Compose files - in the STAMP case) is reused and reused more than before STAMP, leading to an increased reuse of code in general.

Impact on the software industry

AEon

STAMP had a huge impact on Activeeon's testing practices. Our engineers learnt new solutions to improve our test suites' quality. We learnt about the concept of mutation testing and the tools that can be used on our projects. By participating in conferences with other test specialists and exchanging with them, we learnt about their solutions and found out how they would help us. We now have a set of solutions that we can use to improve our quality thanks to the dissemination that was provided through the STAMP project.

Working with the STAMP tools provided us with solutions for problems that we had at the beginning of the project. Before working on the STAMP project, we did not have a good solution to test automatically our solution against several configuration. We usually discovered issues when using the solution in the client environment. CAMP provided us a solution to test our configuration before installing it in our clients environment.

Using the tools developed during the STAMP project enables us to **upgrade our test suites and improve our code quality**. Indeed, during the tools testing phase we discovered several bugs thanks to the tools results. Our code coverage increased when using Dspot and Descartes and highlighted bugs that were not covered.

We improve our solution's architecture to match the STAMP tools requirements. Botsing required a minimum logging capacity that we didn't have at the beginning of the project. Improving our logging architecture in order to fit Botsing requirements is a benefit that we use everyday when reading the logs.

ATOS

STAMP outcomes are innovative tools in the area of test amplification that support the software-driven industry. The knowledge acquired and the available tools has and will allow Atos to provide better quality services. The key benefit is that software testing saves money (and of course time). Software testing presents many benefits for Atos and one of the most important ones is that it is cost-effective. STAMP advances and outcomes in areas such as test automation, DevOps and agile testing are crucial in this respect. Having **testing teams involved in delivering better software products provides benefits** because it saves time and ensures better quality of products/software in the long run. Software development consists of many stages, and if bugs are found at earlier stages it costs much less to fix them. That is why amplification tools for testing is a must. Getting testers and QA team technically capable and with the right tools for delivering software projects is fundamental for the industry. Our knowledge transfer from R&D to the Testing Factory team in Seville has been accompanied with the use cases that we have tested in the project.

As an example, STAMP's results helped us to **increase the quality of products** delivered by the use case providers of the project, in fact for CityGO, STAMP tools helped to automate its delivery on optimal and functionally compatible environments. We have seen an increase (11% for CityGO, 66% for SUPERSEDE) in the diversity of behaviors covered by amplified test suites and a significant increase in the number of valid configuration bugs detected during testing, which reduced the risks of letting regression bugs leak in production code (e.g. mutation score was increased to 69%) and hence increases our trust in the delivered service. Our ambition is to enhance it through DevOps tools and amplification testing methodologies developed in STAMP.

At the same time it is important for the cloud-offering in Atos, for example the CAMP tool could assist our cloud-offering in Atos to adapt the delivery of software applications to different environments. So, we would like to test if CAMP is suitable for configuration management and to be used in the optimization of the delivery. In the process of validation of an application in the moment where the deployment is made, the tool would allow to configure according to certain tests and check the functional/sanity checks and performance tests.

Moreover, Atos has a well-known and appreciated offering in testing service provided by a dedicated ATOS Software Testing Factory located in the south of Spain, in Seville. The factory offerts testing services to many customers in public administrations and in the private sector. **STAMP outcomes have already been shared with them and provide great value for this factory** by allowing them to facilitate and amplify as well as to automate some of the testing tasks the Devops team performs, and for the automation of the deployment CI-CD. The best practices for testing are at the same time used in the ARI common software development methodology developed in ARI. The testing is of paramount importance and will serve for adopting the testing best practices into it.

ENG

ENG's business model is mainly based on contract activities along with a limited set of products. These activities include the production, deployment and operation of software to support the services included in the contract. For such a business model, the possibility to reduce the production time and increase the quality is a big added value. Efficient testing mechanisms which embrace the whole lifecycle of produced software are fundamental to go in this direction. This is one of the most important motivations by which ENG joined STAMP: automatic test amplification is a good possibility to innovate production processes.

The introduction of STAMP in ENG's processes consists of three phases:

- in parallel with the development of the tools, **STAMP has been introduced in the educational program of ENG's IT Academy**. This increased the skills of testing teams in both test amplification and STAMP tools

- as soon as the tools reached an adequate maturity level, they have been **introduced in the production pipelines of some pilot projects**. This experimentation is still in progress, but the first results are good
- as a consequence of the results of the previous phase, STAMP has already been included as a testing tool in the technical proposal for some potential customers. This means that, if these proposals will be accepted, STAMP, or some of the STAMP tools, will be used in the production of the proposed solutions.

At time of writing this document the proposals are being evaluated by the customers. In any case the experimentation and the training sessions will continue in order to better evaluate the advantages with respect to legacy testing procedures.

OW2

OW2 is on the path to provide trust to OSS users: our 60+ projects have diverse maturity levels, and this has to be made clear for users, most of them being professionals from the industrial and academic world.

We have initiated a process called MRL, for "Market Readiness Level" (https://oscar.ow2.org/view/MRL/). As mentioned in the page, outcomes from our research projects, particularly STAMP tools, are involved to gather and process projects data, and improve code quality.

CI/CD code quality assessment, as well as test improvements and coverage, are key factors of code trust: STAMP tools have proven their efficiency on many of our projects (particularly those with modern-style testing and CI/CD processes), as STAMP allowed to validate them on almost 10% of the code base (4 projects in our own use-case + deep usage on XWiki and ActiveEon OW2 projects), which is a significant and representative sample.

At OW2, we consider tool sets like STAMP a key driver for OSS acceptance: and growing OSS acceptance has a deep impact on the structure of the software industry.

Tellu

During the STAMP project, TelluCloud was refactored into micro-services and deployed commercially in various cloud infrastructures, including AWS and Microsoft Azure. The complexity of the system as a whole has increased, especially with respect to configuration and deployment, as there are many more parts to deploy, including queues and other infrastructure. Tellu started the project with a desire to move towards DevOps, with the objective of quicker and more automated cycles of development and release, but without much knowledge of how to get there. The test suite was weak: test coverage was low, and there was no concept of test quality. STAMP has been a learning experience for Tellu. **The project participation has brought us much insight into testing and test quality**. We have learned how important testing automation is, especially in the DevOps context. It has given us a chance to experiment and learn.

The KPIs from the validation show that the STAMP tools have produced good results in the Tellu use case. We have a total **test coverage increase of 44,5%** and a **mutation score increase of 45,5%** thanks to STAMP. Descartes has proved a good way to measure mutation score, and finding pseudo- and partially-tested methods has provided insight and helped us improve the test quality, raising trust in the tests. Tellu continues to use Descartes, and has included it in our DevOps toolset. The DSpot and RAMP experiments provided a good deal of new test coverage. The new tests have been committed and helps test TelluCloud. Working with CAMP, new micro-service and system tests have been put in place. CAMP amplifies Kubernetes configuration files, allowing us to explore the configuration space for performance in cloud deployments. With CAMP we also learned how to deploy the whole system of micro-services in a Docker container with Docker compose, a good approach to running integration tests which does not need cloud infrastructure.

STAMP tools and project participation has helped ensure the quality of the TelluCloud code as the system got more complex, making sure TelluCloud meets high demands for availability and quality. TelluCloud will continue to be deployed in new types of cloud infrastructure, and automated testing is more important than ever. STAMP insights and tools can help Tellu ensure correctness and robustness in a cost-effective manner, thereby staying competitive.

XWiki

The STAMP research project has been a boon for the XWiki open source project. It has allowed the project to gain a lot in terms of quality. The key domains that have substantially benefited the quality are:

- **Increase of the test coverage**. At the start of STAMP the XWiki project already had a substantial automated suite of tests covering 65.29% of the whole case base. Not only the project was able to increase it to over 71% (a major achievement for a large code base of a million lines of code such as XWiki) but also improve the quality of these tests themselves thanks to increasing the mutation score for them by using PIT/Descartes.
- **Addition of configuration testing**. At the start of STAMP the XWiki project was only testing automatically a single configuration (latest HSQLDB + latest Jetty + latest Firefox). From time to time some testers were doing manual tests on different configurations but this was a very intensive process and very random and ad hoc. We were having a substantial number of issues raised in the XWiki issue tracker about configuration-related bugs. Thanks to STAMP the XWiki project has been able to cover all the configurations it supports (about 31 at the moment of writing) and to execute its functional UI tests on all of them (every day, every week and every month based on different criteria). This is leading to a huge improvement in quality and in developer productivity since developers don't need to manually setup multiple environments on their machine to test their new code (that was very time-consuming and difficult when onboarding new developers).

XWiki has spread its own engineering practices to the other STAMP project members and has benefitted from the interactions with the students, researchers and project members to accrue and firm up its own practices.

Last but not least, the STAMP research project has allowed the XWiki project to get time to work on testing in general, on its **CI/CD pipeline**, on adding **new distribution packagings** (such as the new Docker-based distribution now) which were prerequisites for STAMP-developed tools but which have tremendous benefits by themselves even outside of pure testing. Globally this has raised the bar for the project, placing it even higher in the category of projects with a strong engineering practice with controlled quality. The net result is a better and more stable product with an increased development productivity. Globally STAMP has allowed a European software editor (XWiki SAS) and open source software (XWiki) to match and possibly even surpass non-European (American, etc) software editors in terms of engineering practices.

Impact on science and education

INRIA, KTH

PhD Theses

- Benjamin Danglot: DSpot is an essential contribution of the PhD thesis (defended November 2019)
- Oscar Luis Vera Pérez: Descartes is an essential contribution of the PhD thesis of (defended December 2019)

Master thesis: Simon Bihel (2018), Andrew Bowgi (2018 - 2019), Quentin Le Dilavrec (2019), Henry Luong Phu (2019), Zeming Dong (2019), Quentin Angeli (2019), Jérémy Esnault (2019), Pierre Miola (2019), Kévin Paratre-Badois (2019), Laurent Pierre (2019)

Courses

- Descartes and DSpot are used in our Master 2 course on "Validation & Verification" since 2018: https://oscarlvp.github.io/#teaching ; https://github.com/Software-Testing)
- DevOps and automatic testing course at KTH: https://github.com/KTH/devops-course/
- DevOps course at INSA: https://github.com/arnobl/RennesGo

Scientific talks and papers

- 11 publications, inc. 7 papers in peer-reviewed international journals (Journal of Systems and Software and Journal of Empirical Software Engineering) and 4 papers in the proceedings of international conferences.

New projects

- Industry PhD with a software testing company in France; Industry PhD with Ericsson.

TUD

PhD degrees: Botsing is an essential contribution of the PhD thesis of Pouria Derakhshanfar

Master degrees: 4 master thesis related to STAMP research and technologies: Bjorn Evers (TU Delft, 2020), Shang Xiang (TU Delft, 2020), Jeroen Castelein (TU Delft, 2017), Sven Popping (TU Delft, 2020), Boris Cherry (University of Namur, Belgium, 2020).

Courses:

- edX Courses (9,546 students in year 1, and 4,018 students in year 2)
  - Automated Software Testing: Unit Testing, Coverage Criteria and Design for Testability - https://www.edx.org/course/automated-software-testing-unit-testing-coverage-criteria-and-design-for-testability
  - Automated Software Testing: Model and State-based Testing - https://www.edx.org/course/automated-software-testing-model-and-state-based-testing
- Software Quality and Testing - https://se.ewi.tudelft.nl/cse1110-2019/ (TU Delft course)
- Software Testing and Reverse Engineering - https://studiegids.tudelft.nl/a101_displayCourse.do?course_id=51125 (TU Delft course)
- Introduction to mutation testing (Youtube videos, online on January 2019):
  - Mutation testing - https://youtu.be/QYbqz-gFWAk (790 views, 07/11/2019)
  - Mutation operators - https://youtu.be/KXQTWLyR5CA (773 views, 07/11/2019)
  - Mutation score - https://youtu.be/BEBhTtSZAlw (731 views, 07/11/2019)

Scientific talks and papers:

- 10 publications, inc. 5 papers in peer-reviewed international journals (Journal of Systems and Software, Journal of Empirical Software Engineering) and 4 papers in the proceedings of international conferences, and 1 paper in the proceedings of an international workshop.
- A tutorial: "Automated Crash Replication with Search-based approaches," presented at the 11th International Symposium on Search-Based Software Engineering (SSBSE '19), Tallinn, Estonia.

New projects:

- 5 year NWO Vici grant for Andy Zaidman on software testing (project “TestShift”)

SINTEF

Publications in excellent conference and journals assess this impact: https://www.stamp-project.eu/view/main/publications

Scientific talks and papers

4 live tutorials, 1 video tutorial and 2 publications, inc. 1 paper in a peer-reviewed international journal (Software Impacts Journal) and 1 paper in the proceedings of the 30th International Symposium of Software Reliability Engineering.

- Franck Chauvel, Brice Morin, Enrique Garcia-Ceja. Amplifying Integration Tests with CAMP in the Proceedings of the 30th International Symposium of Software Reliability Engineering. IEEE Computer Society. 2019. To be published.
- Franck Chauvel, Brice Morin, Enrique Garcia-Ceja. CAMP: A Tool to Amplify Configuration Tests. In Software Impacts Journal. Volume 3 2020. Accepted for publication on Dec. 16, 2019.
- CAMP tutorial presented at STAMP Workshop in Nice/Sophia Antipolis (Jan. 2019) by F. Chauvel
- CAMP tutorial presented at OW2’con 2019 in Paris (Jun. 2019) by F. Chauvel
- CAMP tutorial presented at the A-Test workshop at EFSE 2019 (Aug. 2019) by F. Chauvel
- CAMP tutorial presented at the Inria Tech Talk (Feb. 2019) by E. Garcia-Ceja.
- CAMP video tutorial (Nov. 2019) by F. Chauvel. Youtube: https://youtu.be/81_2H7GOQwg

New Project

- Internal SINTEF project on Software Quality, where CAMP is used as an example for advanced testing techniques.

Impact on society

Publications in the IT press during the second period of the project include:

- A 12 pages article, Test amplification for DevOps, by Caroline Laundry (Inria) in Linux Magazine, June 2019
- A 3 pages article, Automated tests to support DevOps teams, by the STAMP project team, in Programmez, March 2019

More than 400 developers learned about STAMP tools at Devoxx 2019 in Paris thanks to Caroline Landry (Inria) and Vincent Massol (XWiki) talks.

The 20 STAMP face-to-face workshops and 3 webinars have reached several hundreds of industry developers and testers, including 200+ Orange engineers in Paris and Grenoble, 70+ EC DGIT team members in Brussels, 25 CGI employees, 25 startup employees at Station-F in Paris, 25 DevOps in Sophia-Antipolis Telecom Valley.

STAMP meetings with industrial companies such as Ericsson, Harmonic, Kerevol, Orange Labs, Solocal, Spotify, and Veonum have created constructive interactions with software developers and testers.

STAMP Website 36 technical articles provide useful news and links about software testing tools and their integration in CI/CD environments

2 Update of the plan for exploitation and dissemination of results (if applicable)

Dissemination objectives

During the second half of the STAMP project, the main dissemination activities are focused on reaching out to several hundreds of international professional developers and testers. This was possible while participating in 15 industry events, 15 workshops, 3 scheduled webinars, and through 13 STAMP talks provided at professional events (M18-M36). We've also launched a beta-testing campaign which collected 12 relevant items of feedback from potential early users of the STAMP tools that were shared with the project's partners. The project concepts and outcomes were spread in scientific research and applied research groups, where exchanges and collaboration have started with related EU-funded projects such as Elastest. The main dissemination and communication activities have been summarized in the table below.

Table: **STAMP Dissemination and communication activities**

	First half of the project	**Second half of the project**	**Total**
STAMP Workshops	4	16	20
STAMP Webinars	0	3	3
Beta-testing feedback	0	12	12
Industry events	12	15	27
Industry talks	2	13	15
Press releases	1	3	4
Articles incl. podcasts	4	12	16
Technical articles	9	27	36
Scientific publications	18	13	31
STAMP videos	4	14	18

Collaboration Platform

The collaboration platform facilitated smooth interactions between partners during the project, through remote meetings, mailing list, group discussions, and the private wiki. Also, the online software repositories have been extensively used for collaborative developments, showing a high level of engagement in the STAMP tools not only by the project partners but also by third party stakeholders including members from java DevOps teams and Eclipse committers.

Scientific publications

With 31 papers in scientific publications, the STAMP partners co-organized and participated in multiple scientific and academic conferences including, for example, ISSRE (Symposium on Software Reliability Engineering), ISSTA (International Symposium on Software Testing and Analysis), ICSE (International Conference on Software Engineering), ICST (International Conf. on Software Testing, Verification & Validation). STAMP R&D activities also supported three PhD's papers.

Project website and social networks

The STAMP website has been continuously updated. It offers content such as product overviews, 18 videos, 14 interviews, 35 technical articles, 31 scientific publications. The 1420 unique visitors (recorded as of Nov 12) read an average of 6 pages in about 7:05 minutes. Next to the Homepage, the Discover and Software sections are the most popular. The top-10 visitor's countries include 8 European countries, the USA (5th) and India (10th).
The consortium had an on-going communication activity directed at the press and social media with four press releases resulting in two full featured articles in major industry publications, 13 web articles and one podcast. On the social media, we sent 600+ tweets, attracted 360 twitter followers, and brought 22 discussions topics on LinkedIn.

External collaborations

More than 70 DGIT developers attended the STAMP workshop for EC-DGIT, showing great interest in mutation testing and testing automation to improve the Java test environment in use at the European Commission online services. STAMP was also introduced at companies like CGI, Ericsson, Orange, Sopra-Steria and Spotify. Interactions between developers and the STAMP partners about the integration of the STAMP tools in their CI/CD pipeline, provided relevant feedback to improve the tools. A dozen of developers from a diverse set of organisations also provided relevant feedback thanks to the beta-testing campaign.
A brilliant external contribution to DSpot open source test amplification tool came from the University of Antwerp. Thanks to Mehrdad Abdi, Henrique Rocha and Serge Demeyer, DSpot is no longer limited to Java applications. The researcher team has successfully applied the core of DSpot to Smalltalk, developing test amplification in the Pharo environment. The three researchers improved a test suite's mutation score, applying DSpot on a simple Bank application with a few methods and test cases.

Elastest and STAMP project teams have detected possible synergies between their respective tools. They identified use cases where STAMP services can be deployed in ElasTest. The final benefit would be for developers to understand how the different software pieces are working together and where to drill down into the code to improve their application, while chasing hidden bugs.

Elastest is focused on providing tools to understand what is going on in the code when something fails in the application, while STAMP offers tools to improve the quality of test cases. It would be possible to use STAMP tools within Elastest to provide more information about how a DevOps team is performing on the test technologies, including mutation testing and software testing automation.

Table: **Industry events and open source events, including STAMP workshops**

Events	**2017 STAMP Presence**	**2018 STAMP Presence**	**2019 STAMP Presence**	**STAMP Participation**
Fosdem, Brussels	1	1	1	3 incl. booths & talks
Cloud World Expo, Paris	1	1	0	2 incl. booth & talks
Devoxx, Paris	0	1	1	2 with talks
OW2con, Paris (2016-2019)	2	1	1	4 inc. talks & workshop
OSCON, Austin	1	0	0	1 booth
EclipseCon France	0	1	0	1 talk
SophiaConf, Nice	0	1	0	1 talk
Cloudwatch Hub, Amsterdam	1	0	0	1 meeting

Java User Group: Rennes, Paris, Madrid	0	2	2	4 talks
OpenStack Summit	1	0	0	1 booth
EclipseCon Europe, Germany	1	0	1	2 incl. booth & talks
Cloud Europe Expo, Paris	0	1	0	1 booth
NetFutures, Brussels	1	0	0	1 meeting
Open Source Summit, Paris	1	1	1	3 incl. booth & talks
STAMP Workshops	0	4	14	18
Total	10	14	21	45

For more details about STAMP industry/Open Source events, and workshops, please visit: https://www.stamp-project.eu/view/main/events/list

Table: **STAMP WP6 KPIs at M36**

KPIs as described in D6.1	**M36 achievements**	**Project Target**	**% achieved**
OSS events & scientific publications			
KPI16 - Presentations of STAMP technologies in most important international OSS forums	27	5	540%
KPI17 - Papers accepted about software engineering research (conf & journals)	31	10	310%
Website and Social Media			
KPI15 - Website visitors	1420	500, 700, 1000 (Y1, Y2, Y3)	65%
KPI14 - Twitter followers	322 outside the consortium	200 outside the consortium	161%
Adoption of STAMP Technologies			
KPI13 - External contributions	59 contributions from 23 contributors	>15 at M36 from 3+ third party org.	393% 767%

KPI15 is not achieved, mainly because as we essentially promoted the tools to a technical audience, visitors went directly to the STAMP Github repository, and this traffic is not counted in this KPI.

3 Update of the data management plan (if applicable)

N/A

4 Follow-up of recommendations and comments from previous review(s) (if applicable)

Reviewers recommendations after the first review for m1-m18

Deliverable D1.2 must be revised to include a clear specification of the baseline of DSpot and the functionalities that will be implemented in STAMP.	Revised D1.2, including a curated list of DSpot commits for DSpot since the beginning of STAMP

	Delivered on November 30, 2018
Deliverable D3.1 must be radically revised to match the DoA, namely to document the state of the practice by conducting interviews with developers from the industrial partners involved in this project, as well as with developers from other institutions. The revised version of D3.1 should also provide a clear specification of the baseline for EvoCrash, and a clear outline of the roadmap of innovation that will be created and implemented into EvoCrash in the context of STAMP.	D3.1 has been radically revised Delivered on November 30, 2018
Deliverable D3.2 must be radically revised to match the work described in the DoA, namely the implementation of the log optimization tool is needed in Task 3.4.	D3.2 has been radically revised Delivered on November 30, 2018
The Periodic Project Report shall include the actual and planned effort, both at WP and partner level, as well as an update of the exploitation and dissemination plan (per the template) as the project has not delivered a document about that in M18. In particular, we request a detailed justification for the 19 Person-Months that were spent on Tasks 4.1 ("Collaborative Software Engineering Platform setup and management"). The contribution per partner and WP (per the template) is also to be included in the upcoming M36 version.	Periodic Report updated: ● Contribution table per task (section 1.2) ● Tables and charts of effort per WP and per partner in part B (section 5.2) ● Dissemination activities The initial DoA plans M1-12 for Task 4.1, but due to the nature of the task (platform setup and management), the duration should last up to M36 to assure the following actions: ● Continuous maintenance of the infrastructure, support to end-users, changes to adapt it to changing needs or conditions of the project ● Managing user permissions, setting-up new services (Github application, Jenkins, ProActive Workflow, etc.), solving unexpected issues (for example, it can happen that VMs over consume disk space and ENG and OW2 must work together to solve the problem) ENG proposed an adjustment in the duration, included in the amendment submitted in October 2018.
All the revised versions of the aforementioned deliverables should be submitted no later than 30 November 2018 (Month 24)	Revised version of D1.2, D3.1 and D3.2 has been delivered on November 30, 2018

Reviewers recommendations for future work after the first review

Deliverable D5.3 should include a rigorous argumentation for each of the KPIs that have been changed, as well as a demonstration that the change in the KPIs does not affect the actual assessment of the project's objectives. Moreover, the deliverable should provide an explicit mapping between KPIs and use-cases (i.e. which KPIs will be validated in each use case), as well as between KPIs and tools (i.e. with KPIs are relevant for each of the tools). This mapping should be ideally provided in form of two matrices.	KPIs have been refined and clarified ● KPIs vs Tools matrice ● KPIs vs Use Cases matrice ● Updated KPIs Objective table Part of the amendment submitted in October 2018
Deliverable D5.6 shall be delivered in M24 instead of in M30, so there will be a control point to check the validation strategy with the use case partners. This should contain the real updated status of WP5 and better descriptions of the case studies (with RQs, variables/metrics/KPI, threats to validity, etc.) for evaluation of WP1, WP2 and WP3 results with the real systems defined by the case study partners. In this context, we also recommend that the industrial partners will put some additional effort into ensuring that each of the use cases are exercising more of the techniques and tools created in the context of STAMP.	The whole consortium dedicated extra resources to deliver a complete and thorough report 6 months early Part of the amendment submitted in October 2018
Deliverables describing tools should provide a clear overview of which specific functionalities have been implemented in the iteration that they are documenting and specify if these have been partially or fully implemented. The aim with this recommendation is to make is easy for the reader to understand the advances achieved in any given iteration. These deliverables should also include a description of the internal architecture as well as clear installation instructions and how to use the tools.	All deliverables about tools include a section about updates Regular release notes to keep track of the progress beyond the deliverables
Since all the proposed STAMP tools are aimed for DevOps teams, include an explanation / manual of how these can be integrated with common CI/CD tools such as Maven, Jenkins, and so on, to support the application lifecycle. To this end, for the next period it will be expected to see, as part of WP4, all STAMP tools integrated as much as possible. In case a tool or a plugin has not been integrated in the STAMP suite, a rigorous explanation shall be provided.	D4.3 describes how STAMP tools are integrated in CI/CD scenario. CI/CD integration is meant to demonstrate how to use STAMP amplification features within a CI/CD platform made of widely used tools (Jenkins server, amplified tests pushed automatically in the git repository within a dedicated branch, and a pull request is automatically opened in the master branch). DSpot and Descartes CI/CD integration is already available in the STAMP collaborative platform. A POC GitHub App exposing Descartes features is runs in the STAMP collaborative platform. CAMP and Botsing integration is planned for Spring 2019. Real-world projects were selected as target for STAMP CI/CD integration.
Explicitly define which partner will exploit the results beyond the use in the use cases and how; for instance, OW2 explained that they aim to launch an additional service to	STAMP partners are engaged in exploitation : ● ENG is offers consultancy services to its clients around STAMP results.

what OSCAR offers in their repository, but the concrete strategy / business plan for achieving needs to be clearly specified.	● ATOS pushes the adoption in its testing factory in Seville ● OW2 integrates STAMP tools in its IT infrastructure to offer technical services to all its hosted projects, ENG is planning to do the same ● SINTEF, TUDelft, Inria, KTH and XWiki have consolidated plans to elaborate on research and development around STAMP results ● Academic partners, as well as ENG, are running face to face courses around STAMP concepts and tools
Evaluate the chosen licensing schema (LGPL) and the business services that can revolve around that 5licensing. We recommend to either replace the license with a more business friendly one, or justify the selection of the LGPL license.	The different STAMP results have different licenses (LGPL, MIT, Apache 2, EPL) according to their technological environment (e.g. Eclipse plugins have to be released with EPL license). LGPL is compatible with the exploitation of STAMP technology as a service: ● This is the most widely adopted model for software tooling ● This is the model envisioned as early as the STAMP proposal
Define the business scenarios for the different delivery models of the tools, considering their pro's and con's as well as business models and if possible, potential pricing schemas; for instance, OW2 claims that they aim to use the STAMP techniques and tools as quality indicators by having them contribute to the "Market Readiness Level". However, it remains unclear how this can be done considering the fact that the goal of STAMP is to reduce the testing effort, and not to provide quality indicators.	Preliminary business scenarios and routes for exploitation have been analysed for each tool: ● DSpot, Descartes, CAMP, Botsing (SaaS, OSS, TAaaS – Tools for Amplification Testing as a Service) ● Tools with different interfaces: CLI, Maven, Gradle, Jenkins, Eclipse IDE, Docker (SaaS, OS, Support consultancy and training) ● Services: REST APIs (SaaS, OS) We have already analysed and agreed on the potential business models that will be reflected in the business model deliverable (2019/11/30). We are currently evaluating pricing schema based on a study of similar offers in the domain of automated testing. All STAMP results will be presented into WP6 presentation and exploitation deliverable. OW2 plans to integrate STAMP in the Market Readiness Level (MRL) methodology as an indicator of market readiness. It is not the results that count but the usage of STAMP. The methodology does not consider that STAMP provides quality indicators but considers that using STAMP reflects greater attention to quality and thus adds credits to project MRL scoring.
Strengthen the activities related to project and risk	Project and risk management are increased.

management. Improve Part B (i.e. the technical part) of the Periodical Project Report with better and more detailed information. For each activity (and deliverable) the role and contributions of each partner in each WP should be clearly specified. Also, the use of resources should be accompanied by a more detailed explanation, including the planned and actual spent budget and effort for the period, as well as why deviations occurred.	Systematic tracking and increased monitoring of activities. Periodic report has been improved: ● More details for each WP ● Development status for each tool (section 1.2), including status on December 1, 2016 ● Contribution table for each task summarizing contribution of each partner ● Use of resources is explained in detail, including effort tables and charts to compare actual vs planned effort, and explanation of deviations

Reviewers recommendations after the extraordinary interim review for m20-m27

Task 3.2 should be extended to Month 36. The title should be slightly changed to better reflect its content. A suggestion for the title is “Log Optimization and Benchmarking”. The content of the task should also be updated to explain that the focus on stack trace analysis has been increased compared to what was initially proposed in the DoA. Also, indicate in Task 3.2 that log optimization is about massaging the stack traces logs (pre-processing) for better crash replication results.	Annex1 part A has been modified accordingly. Part of the amendment submitted in May 2019.
In order to achieve a better alignment of Deliverable 3.2 with the DoA, we recommend the following: (i) slightly rename the deliverable (e.g. “og Optimization and Benchmarking”) to reflect better its current content; (ii) move from Deliverable D3.3 the content related to the pre-processing stack traces for better crash replication (i.e. Chapter 4); (iii) add the content related to the work to be done on “adding new probes and new log messages”.	D3.2 has been revised accordingly. D3.2 v1.2 delivered on 21-nov-2018.
Deliverable D3.1 should include a paragraph that explains why the study on logs in Task 3.1 led to the conclusion that the needs of the industrial partners require to work on WP3 to reduce the emphasis on log analysis while focussing more on the analysis of stack traces (in Task 3.2 and Task 3.3). In other words, the paragraph should highlight how the conclusions of the survey are reflected in the actions to be undertaken in future WP3 deliverables. Deliverable 3.1 should add a statement that explicitly indicates that the first chapters come from a master thesis that was finalised before the start of the project and provide a reference to it.	D3.1 has been revised accordingly (see “Introduction” in D3.1). D3.1 v2.20 delivered on 03-dec-2018.
Deliverable 3.4 should include the parts related to the behavioural patterns.	Done, see chapter 1 ”Test amplification for common behaviors” in D3.4. D3.4 delivered on 01-dec-2019.

Make sure that the final Deliverable (D3.5) also serves as an integrator of the entire work package, presenting a cohesive story line of what each of the previous deliverables has been covering.	The Introduction of D3.5 provides an integrated view of the work done in WP3. D3.5 delivered on 29-nov-2019.

Reviewers recommendations for future work after the extraordinary interim review

Implement the corrective actions for WP3 that results from the recommendations contained in this report (see above).	Done, see above paragraph "Reviewers recommendations after the extraordinary interim review for m20-m27".
Present a brief roadmap for each of the tools developed during the project. The roadmap should outline (i) the innovation created integrated in each tool during the project, namely the conceptual increment compared to the state of the tool at the start of the project; and (ii) clearly identify in each case the potential target users (e.g. individual developers or system integrators) and what each tool offers for each of them. Figures regarding the effort related to the development of the new features are welcome, but they should not substitute the outline of the two aforementioned points.	Done, see paragraphs "Conceptual Increment" and "Target Users" in sections WP1, WP2 and WP3 of this document.
Analyse the different pricing strategies that STAMP as a service could have in order to become the "Travis of testing", taking into consideration the costs of running STAMP as a service in an infrastructure.	Done, see deliverable D6.5.
In the final periodic report, provide an overview of the actual vs. planned effort for the involvement of each industrial partner in the scientific work-packages (WP1-WP3). For the industrial partners that have an actual effort of 3PM or more on a scientific work-package, provide a brief explanation on what was the actual major work that the industrial partner has done on that work-package.	Done, see section "Use of Resources" of this document.

5 Deviations from Annex 1 and Annex 2 (if applicable)

5.1 Tasks

WP3 amendment. The amendment consists in:

- the inclusion in T3.2 of crash reproduction benchmarking and the extension of T3.2 to month 36;
- the reallocation of ENG's effort from T3.3 and T3.4 to T3.2 to develop a preprocessor for stack traces;
- the update of deliverables D3.2 and D3.3 deadlines.

TUD and other partners involved have performed according to the new version of WP3 with respect to objectives, KPIs and efforts.

WP4 amendment. The amendment consists in an adjustment in the duration of T4.1 ("Collaborative Software Engineering Platform setup and management") from the original 12 months (M1-12) to 36 months (the whole duration of the project) at no additional costs. This modification has been requested to better reflect the actual activities of the project: specifically, T4.1 concerns the Platform supporting STAMP and the Platform must be up, running and maintained for the whole duration of the project. ENG and other involved partners have performed the activities of WP4 and documented them in D4.4 and in this document, according to this new version of the Work Package.

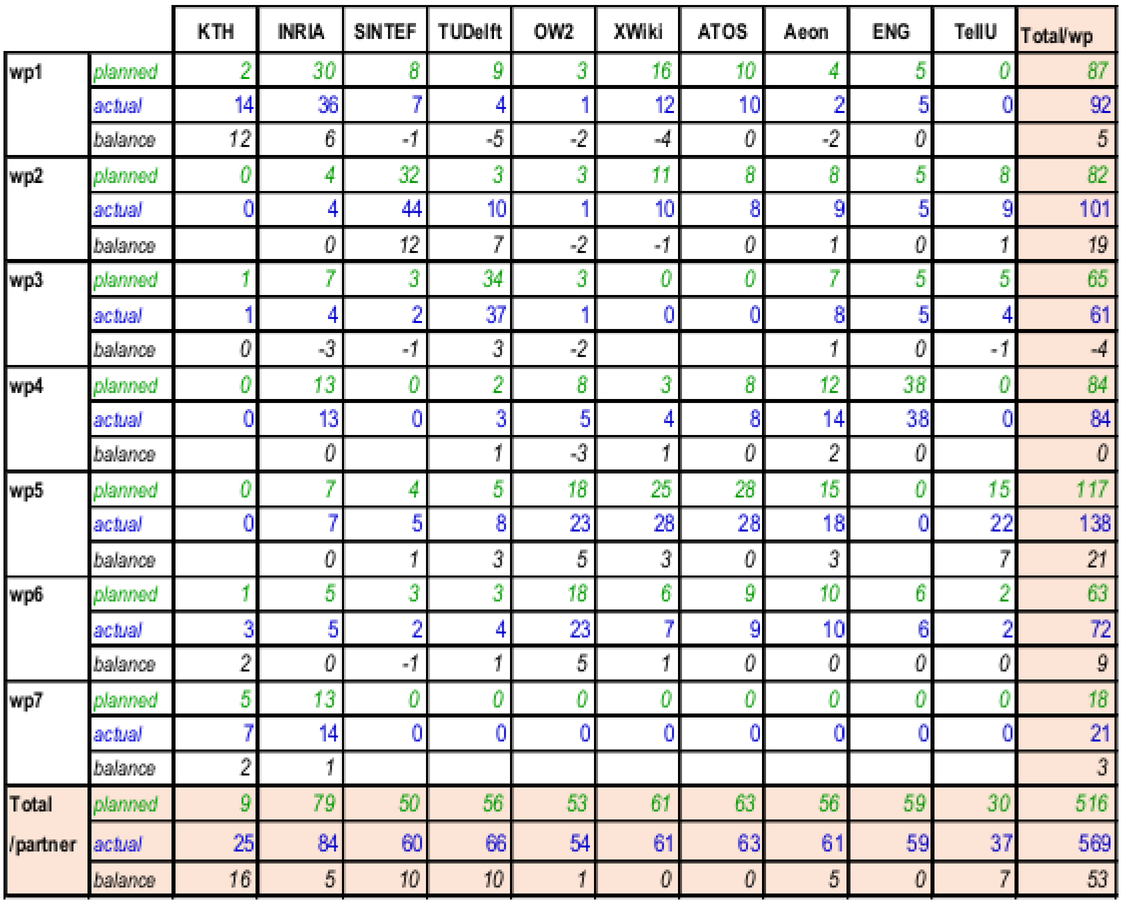
WP5 amendment. Update the WP5 project KPIs
The definition and relevancy of the KPIs have been adjusted to match better the tools developed and so that they can be measured, while retaining to the maximum the original intents and targets/objectives. In addition we've added new metrics and new KPIs where we thought that they were missing (e.g., measuring test quality). Here is an updated version of the WP5 KPIs vs Objectives table you can find in the DoA, the changes are in blue.

Objective							KPIs
ID	1	2	3	4	5	6	
KPI1	X						Increase the diversity of execution paths covered by 40% Secondary metrics to compute the coverage changes brought by each STAMP tool.
KPI2	X						Decrease by 20% the number of flaky tests Metrics to count the number of flaky tests identified and handled, and to count the number of flaky tests fixed
KPI3	X						~~Increase by 20% the number of lines of product code, which are executed for each second of time spent running tests.~~ Improve mutation score of at least 20% over the course of the project.
KPI4		X					Increase by 40% the number of unique invocation traces between services in a global perspective Measure this KPI also for monolithic projects
KPI5		X					Increase by 30% the number of valid bugs detected during testing which are specific to the generated configurations Metric to count the number of new configuration-related bugs discovered

KPI6		X					Increase the number of automatically tested configurations by 50%. Reduce by 30% the time on configuring and deploying products for testing purpose: compare manual testing with automated testing
~~KPI7~~			~~X~~				~~Reduce the size of log files by an order of magnitude, keeping all essential information~~ Dropped
KPI8							Increase by 70% the number of crash replicating test cases
KPI9			X				Enhance existing test suites with 10% of production-level test cases

5.2 Use of resources (not applicable for MCSA)

The following table summarizes the effort planned vs actual per work package and per partner for the whole project.



		KTH	INRIA	SINTEF	TUDelft	OW2	XWiki	ATOS	Aeon	ENG	TellU	Total/wp
wp1	planned	2	30	8	9	3	16	10	4	5	0	87
	actual	14	36	7	4	1	12	10	2	5	0	92
	balance	12	6	-1	-5	-2	-4	0	-2	0		5
wp2	planned	0	4	32	3	3	11	8	8	5	8	82
	actual	0	4	44	10	1	10	8	9	5	9	101
	balance		0	12	7	-2	-1	0	1	0	1	19
wp3	planned	1	7	3	34	3	0	0	7	5	5	65
	actual	1	4	2	37	1	0	0	8	5	4	61
	balance	0	-3	-1	3	-2			1	0	-1	-4
wp4	planned	0	13	0	2	8	3	8	12	38	0	84
	actual	0	13	0	3	5	4	8	14	38	0	84
	balance		0		1	-3	1	0	2	0		0
wp5	planned	0	7	4	5	18	25	28	15	0	15	117
	actual	0	7	5	8	23	28	28	18	0	22	138
	balance		0	1	3	5	3	0	3		7	21
wp6	planned	1	5	3	3	18	6	9	10	6	2	63
	actual	3	5	2	4	23	7	9	10	6	2	72
	balance	2	0	-1	1	5	1	0	0	0	0	9
wp7	planned	5	13	0	0	0	0	0	0	0	0	18
	actual	7	14	0	0	0	0	0	0	0	0	21
	balance	2	1									3
Total /partner	planned	9	79	50	56	53	61	63	56	59	30	516
	actual	25	84	60	66	54	61	63	61	59	37	569
	balance	16	5	10	10	1	0	0	5	0	7	53

Use of resources comments

KTH

The planned efforts were underestimated, when moving the coordination from INRIA to KTH because of the move of M. Monperrus and B. Baudry from INRIA to KTH. INRIA was coordinator for STAMP from m1 to M9. Then the coordination moved to KTH. Yet, costs are significantly higher at KTH, hence we had to limit efforts to stay in budget. Meanwhile, Baudry and Monperrus investigated significant efforts in the coordination of the whole project and in the research and supervision activities in technical work packages. These additional efforts have been funded by other grants acquired by Monperrus and Baudry.

INRIA

Some PMs have been moved from WP3 o WP1 and some additional PMs have been allocated to WP1, to achieve the maturity level of tools expected by industrial partners, especially on Descartes which allows to evaluate the maturity of the tests, which is a growing need in the industry. The small overspending of WP7 is mainly due to unplanned tasks (additional review, amendments and detailed management of some partners).

SINTEF

The budget/effort of SINTEF is partly dependent on the EUR/NOK exchange rate. In STAMP, this meant that more effort (PM) could be used for the same EUR budget. The additional effort (+10PM compared to the plan) has been mostly spent on WP2, lead by SINTEF. As the key personnel from SINTEF initially involved in STAMP went through a complete turnover during those 3 years, with people initially involved moving out of SINTEF of internally to other responsibilities, this additional effort has been mostly used to ensure a proper transfer of knowledge and know-how to the personnel taking over. Regarding the slight deviations in other WPs, it corresponds to tasks partly overlapping with WP2. In that sense, part of the WP2 overspending could/should have been spent to compensate for the slight underspending in other WPs.

TUD

For TU Delft, quite a bit of the effort started later than originally planned due to the difficulty in finding suitable personnel. Also, with the reorientation of WP3 after the mid-term evaluation, effort was divided differently over the work packages than originally planned: this includes extra effort for WP3 and WP2, with less effort spent on WP1.

OW2

As reported in the M18 report, lack of personnel resources impacted the PM count in WP1 and WP2, but we still delivered the expected set of quality metrics (WP1), and processed answers from six OW2 use-case projects (WP2) as expected.

For the same reasons as for WP1/WP2, work began later than scheduled on WP3. Then, most of that work was in fact use-case development, transferred to WP5.

The platform (WP4) was successfully delivered and managed in 5 PMs : it was used during the whole project life, and is still online.

The remaining PMs were re-assigned to ensure the completion of the use-case (WP5), as this is where the effort was needed: to provide some CI tools development that was not scheduled at the time of the proposal (WP5), and to ensure success of WP6 given OW2's leadership in delivering the added value of the WP, resulting in even more dissemination and communication than initially expected.

XWiki

The work done for the technical work packages (WP1/WP2) was faster than planned and the extra time has been put on the industrial exploitation of the tools, hence the increased PMs in industrial work packages (WP4/5/6). The moved effort has been put mainly on WP5 to achieve the metrics and the KPIs.

Activeeon

Due to a departure at the end of the first period, Activeeon was not able to finish the remaining 1.9 PM of planned work in WP1. Thanks to an arrangement with INRIA, this 1.9 PM work was conducted by INRIA engineers to end the planned task.

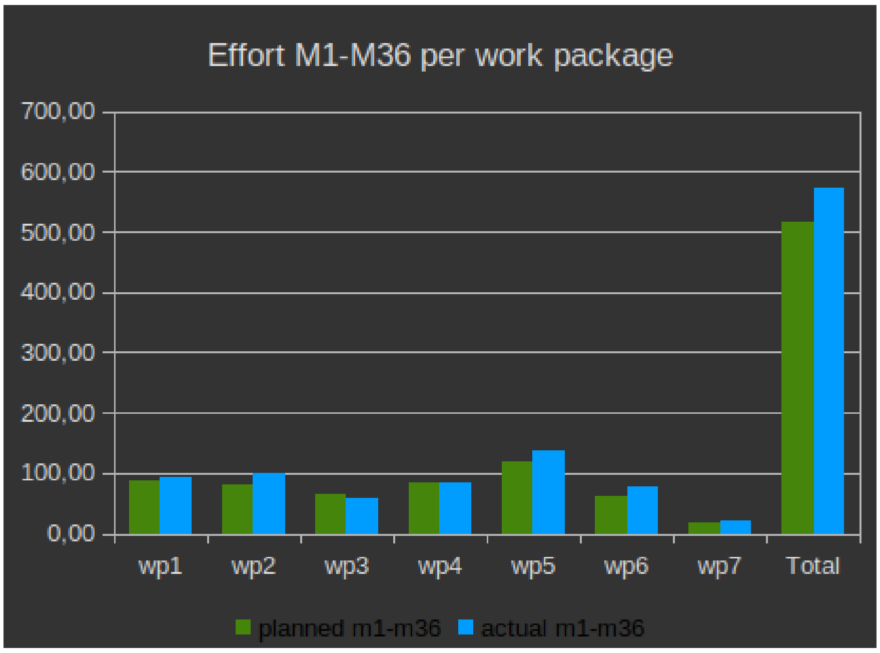
During the second period, Activeeon involved two juniors to replace the first period departure. By involving two juniors instead of one senior, it increased by 7PM the total time spent working on the technical work packages (WP2, WP3, WP4, WP5).

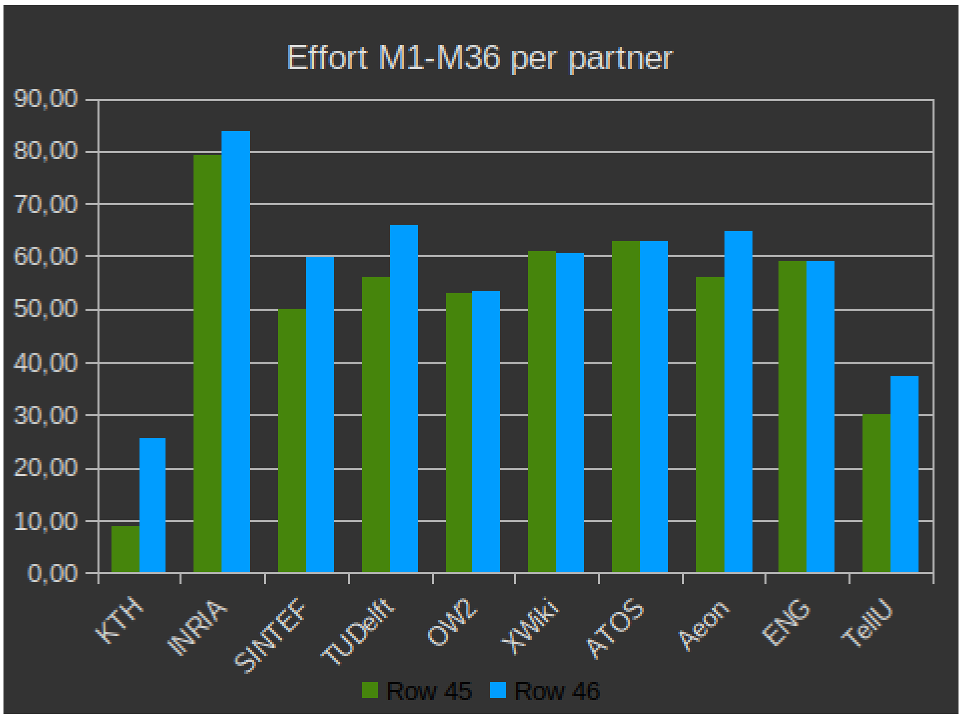
TellU

Some scientific effort was shifted from WP3 to WP2, as this was where the effort was needed. Tellu's small role in WP3 was served with less effort than planned, while the more extensive involvement in WP2 meant a bit extra effort was needed here.

Extra effort was spent in the final year, working on the TelluCloud use case and evaluating the STAMP tools (WP5).

And the following charts compare the planned effort to the actual effort, separately per work package and per partner.





Industry contributions in scientific work-packages

Following the recommendation of the reviewers, from the review report for the extraordinary review meeting, we provide in the following section detailed explanations for industry contributions in scientific work-packages (WP1-WP3) for the industry partners that have an effort greater than 3PM.

ActiveEon

Main **ActiveEon** contributions to WP2:

- Investigate and search for the adequate software technology to profile CAMP generated software.
- Collaborate with SINTEF to understand how to adapt the selected technology (i.e., Linux perf tool and Flamegraphs) to CAMP.
- Develop a tool that automatically profiles CAMP generated software and produce Flamegraphs related to Java and system stack traces. The tool is available here: https://github.com/STAMP-project/camp/tree/docker-flame-graphs/samples/docker-flame-graphs
- Use and evaluate the Flamegraph-based tool (provided by SINTEF) to extract the number of unique execution traces per configuration https://github.com/STAMP-project/docker-traces-xp

Main **ActiveEon** contributions to WP3:

- Set up the empirical evaluation of RAMP https://github.com/STAMP-project/evosuite-model-seeding-empirical-evaluation
- Work at TUDelft during two days to select the right benchmark project and set up the experiment. Since then, the discussion has continued remotely.
- Provide scripts to automatically build jars from SF110 benchmark project (http://www.evosuite.org/files/SF110-20130704-src.zip), generate models using Botsing, and finally run RAMP and evosuite without model seeding over all projects.
- Collect result data from the 110 benchmark projects and analyze/compare them.

ATOS

Main **ATOS** contributions to WP1:

Atos has worked on the Optimization of the DSpot performance and resources (e.g. memory) consumption. In particular, Atos has worked on:

- The profiling of the DSpot execution for performance, memory consumption, threads, obtaining a detailed telemetry.
- The analysis of the profiling report.
- The identification of the performance bottlenecks, memory leaks, etc.
- The proposition and implementation of DSpot code patches for performance and memory consumption improvement.
- The parallelization of the DSpot amplified test execution for JUnit4/5
- Experimentation: DSpot execution on Dhell target project with different input flags. Analysis of the performance and the resources consumption improvements for each applied patch.

ENG

Main **ENG** contributions to WP1:

- Support for Gradle Java projects
- Dspot support for changes (DSpot diff) and multi-modules
- Final version of PitMP, including the support for threshold parameters related to the number of pseudo-tested methods and partially tested methods

Main **ENG** contributions to WP2:

- Installation, configuration and preliminary tests on the first releases of CAMP
- Design, implementation and documentation of a module providing JMeter support for CAMP
- CAMP documentation

Main **ENG** contributions to WP3:

- Support to the continuous compatibility between the Maven plugin and Botsing during the various development phases (including the evolution from Evocrash)
- Design, implementation and documentation of the Botsing log preprocessor
- Design, Implementation, testing and documentation of the Botsing Parallel Module
- Analysis on the applicability of distributed genetic algorithms to increase the performance of Botsing

Tellu

Main Tellu contributions to WP2 (total effort 8,8 PM):

- Developed test framework and utilities for configuring and running system and micro-service tests.
- Documented the various forms of configuration options for a micro-service cloud system.
- Experimented with different tools and frameworks for cloud orchestration, such as Kubernetes and OpenShift, looking at how these can be used to configure cloud deployments and how to make system test automation and amplification possible.
- Input to the development of CAMP to generate configurations for Kubernetes.
- Worked on configuration amplification for performance optimization.
- Contributed to deliverables.

Main Tellu contributions to WP3 (total effort 3,7 PM):

- Participation in the identification of the current state-of-the-practice in the industry.
- Collecting and analyzing stack traces from production systems, evaluate Botsing.
- Interacting with developers on tools. Working with developers to find strategies to replicate crashes.
- Comparative test generation experiments with RAMP and without model seeding, to compare the approaches and analyse differences. Reported in https://github.com/STAMP-project/evosuite-model-seeding-usecases-output/tree/master/tellu

XWiki

Main XWiki contributions to WP1:

- Provided experience of XWiki on testing, to decide which tools to use, how to measure quality, how to collect metrics.
- Provided analysis and feedback to tool developers (Descartes, DSpot) to guide development of the tool features.
- Provided experiments and strategies to use all the tools in a CI context.
- Contributed to eliminate false positives, to identify and classify the result patterns of Descartes results, and to improve the findings. Defined an approach to manage flaky tests in CI environment
- Definition of metrics / KPIs.
- Developed several tools and scripts, among which:
    - Developed a tool to implement the strategy to handle flaky tests, including an integration with JIRA, and a helper to suggest identification of flaky tests. Created generic Groovy scripts to manage flaky tests in a CI environment.
    - Developed a CI integration to integrate the Descartes mutation score in Java Maven projects.
    - Developed Groovy scripts to measure global test coverage and compare reports.

Main XWiki contributions to WP2:

- Participated to discussions and provided architecture ideas for CAMP
- Provided feedback to tool developer (CAMP) to guide development of the tool features + use case needs. Provided assistance to use XWiki as the software on which to perform internal tool development testing for CAMP.
- Conducted several experiments and POCs to decide the best approach to integrate CAMP into a development process. See https://massol.myxwiki.org/xwiki/bin/view/Blog/EnvironmentTestingExperimentations
- Created and improved the CAMP/TestContainers framework (integrated with JUnit tests) to be able to execute the configuration tests on developer's machines and thus be able to debug them.
- Provided Jenkins pipelines scripts to integrate CAMP/TestContainers in the CI

5.2.1 Unforeseen subcontracting (if applicable) (not applicable for MSCA)

N/A

5.2.2 Unforeseen use of in kind contribution from third party against payment or free of charges (if applicable) (not applicable for MSCA)

N/A