CROSSMINER
Developer-Centric Knowledge Mining from Large Open-Source Software Repositories

Project Number 732223

D4.3 DevOps Dashboard

Version 1.0
28 June 2019
Final

Public Distribution

Athens University of Economics and Business (AUEB)

Project Partners: **Athens University of Economics & Business**, **Bitergia**, **Castalia Solutions**, **Centrum Wiskunde & Informatica**, **Eclipse Foundation Europe**, **Edge Hill University**, **FrontEndART**, **OW2**, **SOFTEAM**, **The Open Group**, **University of L'Aquila**, **University of York**, **Unparallel Innovation**

Project Partner Contact Information

Athens University of Economics & Business Diomidis Spinellis Patision 76 104-34 Athens Greece Tel: +30 210 820 3621 E-mail: dds@aueb.gr	**Bitergia** José Manrique Lopez de la Fuente Calle Navarra 5, 4D 28921 Alcorcón Madrid Spain Tel: +34 6 999 279 58 E-mail: jsmanrique@bitergia.com
Castalia Solutions Boris Baldassari 10 Rue de Penthièvre 75008 Paris France Tel: +33 6 48 03 82 89 E-mail: boris.baldassari@castalia.solutions	**Centrum Wiskunde & Informatica** Jurgen J. Vinju Science Park 123 1098 XG Amsterdam Netherlands Tel: +31 20 592 4102 E-mail: jurgen.vinju@cwi.nl
Eclipse Foundation Europe Philippe Krief Annastrasse 46 64673 Zwingenberg Germany Tel: +33 62 101 0681 E-mail: philippe.krief@eclipse.org	**Edge Hill University** Yannis Korkontzelos St Helens Road Ormskirk L39 4QP United Kingdom Tel: +44 1695 654393 E-mail: yannis.korkontzelos@edgehill.ac.uk
FrontEndART Rudolf Ferenc Zászló u. 3 I./5 H-6722 Szeged Hungary Tel: +36 62 319 372 E-mail: ferenc@frontendart.com	**OW2 Consortium** Cedric Thomas 114 Boulevard Haussmann 75008 Paris France Tel: +33 6 45 81 62 02 E-mail: cedric.thomas@ow2.org
SOFTEAM Alessandra Bagnato 21 Avenue Victor Hugo 75016 Paris France Tel: +33 1 30 12 16 60 E-mail: alessandra.bagnato@softeam.fr	**The Open Group** Scott Hansen Rond Point Schuman 6, 5th Floor 1040 Brussels Belgium Tel: +32 2 675 1136 E-mail: s.hansen@opengroup.org
University of L′Aquila Davide Di Ruscio Piazza Vincenzo Rivera 1 67100 L′Aquila Italy Tel: +39 0862 433735 E-mail: davide.diruscio@univaq.it	**University of York** Dimitris Kolovos Deramore Lane York YO10 5GH United Kingdom Tel: +44 1904 325167 E-mail: dimitris.kolovos@york.ac.uk
Unparallel Innovation Bruno Almeida Rua das Lendas Algarvias, Lote 123 8500-794 Portimão Portugal Tel: +351 282 485052 E-mail: bruno.almeida@unparallel.pt	

Document Control

Version	Status	Date
0.1	Initial outline	5 June 2019
0.5	Internal release	18 June 2019
0.8	Internal release to partners for review	24 June 2019
1.0	Final version incorporating internal review comments	28 June 2019

Table of Contents

Executive Summary

This document presents the final version of the DevOps Dashboard developed in the context of **Task 4.3** for the Crossminer platform: **Task 4.3**: Design and Development of DevOps Dashboard

In this deliverable, we report the development of DevOps Dashboard. We present the data sources that are used to create the Dashboard, the metrics that are included in the Dashboard and the information that can be extracted from the visualisations of the dashboard.

1 Introduction

Infrastructure as Code (IaC) [1] is the practice of specifying computing system configurations through code, automating system deployment, and managing the system configurations through traditional software engineering methods. Infrastructure as Code has been gaining more and more popularity as it facilitates the work of DevOps engineers. The increasing size and complexity of configuration files give rise to the need for assessing, maintaining and improving the configuration code quality. In this context traditional software engineering knowledge and best practices associated with code quality management can be leveraged to assess and manage configuration code quality. Such practices are utilised by the System Configuration Code Analyser of the Crossminer Platform, which has been implemented in the context of **Task 4.2**. The outputs of the System Configuration Code Analyser are incorporated in the DevOps Dashboard in order to be more easily and intuitively assessed and examined.

Because of the abundance of data that we designed to include in the DevOps Dashboard, it is divided in two parts. The DevOps Smells Dashboard, that contains metrics and information about the smells that are detected by the System Configuration Code Analyser and the DevOps Dependencies Dashboard, that showcases information about the detected third-party dependencies of the analysed projects.

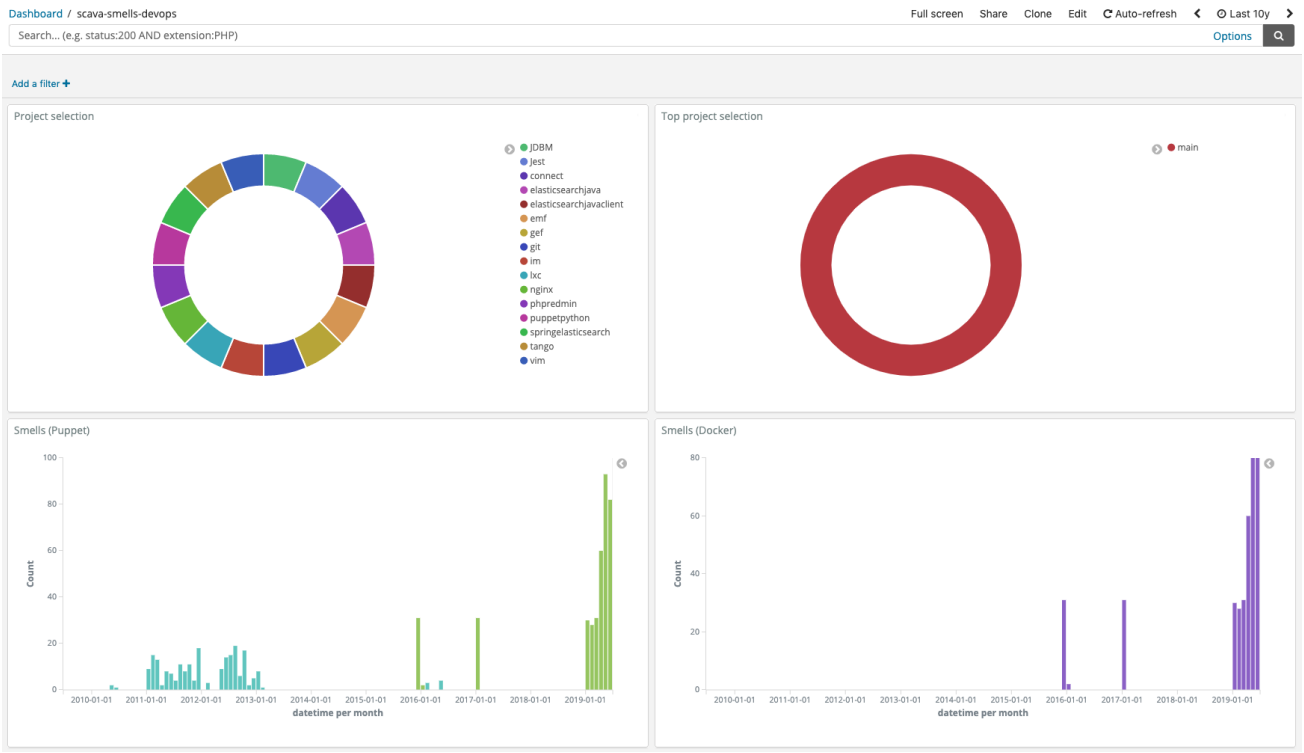

Figure 1: The upper part of DevOps Smells DashBoard

2 DevOps Smells DashBoard

The DevOps Smells Dashboard focuses and showcases information about smells detected in Docker and Puppet based projects. The insights that can be extracted from this dashboard can help a DevOps engineer to make a quality assessment about the status of the configuration code of the analysed project.

As shown in Figure 1, in the upper part of the dashboard we can see the analysed projects and select the project that we want to focus on. Moreover, there are two barchart visualisations. In these visualisations, we can observe the evolution of the number of Puppet and Docker related smells through time. From these barcharts, we can determine if the quality of the configuration code of the projects improves or deteriorates as time passes.

In Figure 2 the lower part of the dashboard is depicted. There, we can see the list of the smells detected in a project. These smells are related to the last commit of the project. The available information consists of the name of the project, the type of configuration management technology (e.g. Docker or Puppet), the type and the name of the smell, the file and the line number, where the smell is detected and the reason of the smell.

Furthermore, we can see the list of antipatterns that are detected on the project. Essentially, this is the history of the smells of a project. There is a drop-down list, where we can select the date about which we want to see the list of the smells. For each date, the smells refer to the last commit of this specific date.

The metrics providers that produce the metrics that are incorporated in this dashboard are the following (the details about the metrics are available in deliverable D4.4):

- org.eclipse.scava.metricprovider.historic.configuration.puppet.designsmells
- org.eclipse.scava.metricprovider.historic.configuration.puppet.implementationsmells

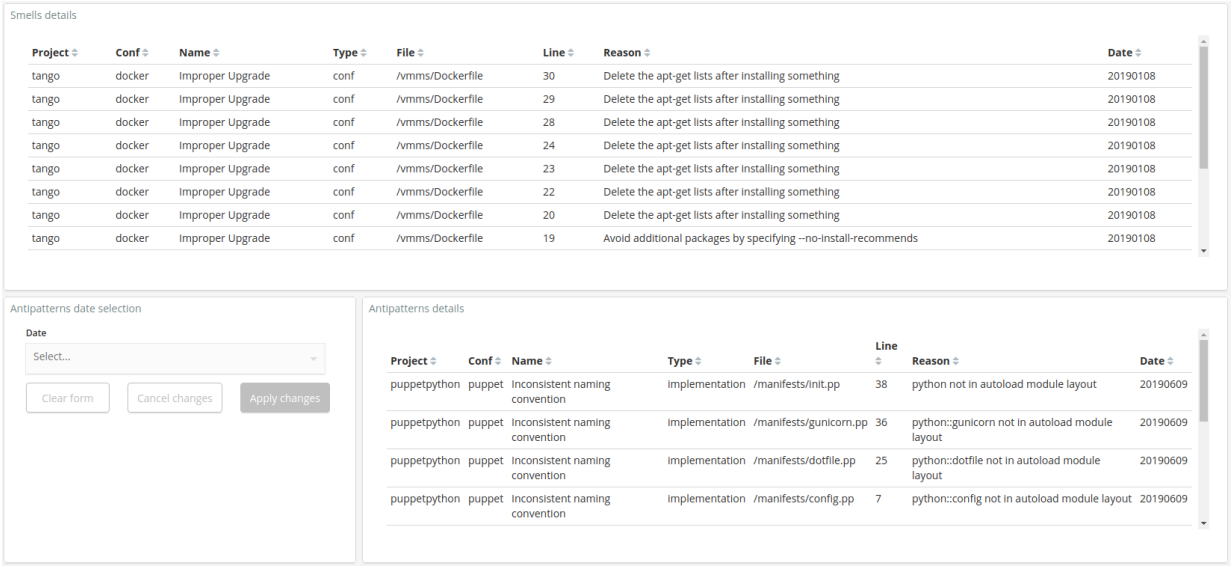

Figure 2: The lower part of DevOps Smells DashBoard

- org.eclipse.scava.metricprovider.historic.configuration.docker.smells
- org.eclipse.scava.metricprovider.trans.configuration.puppet.designsmells
- org.eclipse.scava.metricprovider.trans.configuration.puppet.implementationsmells
- org.eclipse.scava.metricprovider.trans.configuration.docker.smells
- org.eclipse.scava.metricprovider.trans.configuration.puppet.designantipatterns
- org.eclipse.scava.metricprovider.trans.configuration.puppet.implementationantipatterns
- org.eclipse.scava.metricprovider.trans.configuration.docker.antipatterns

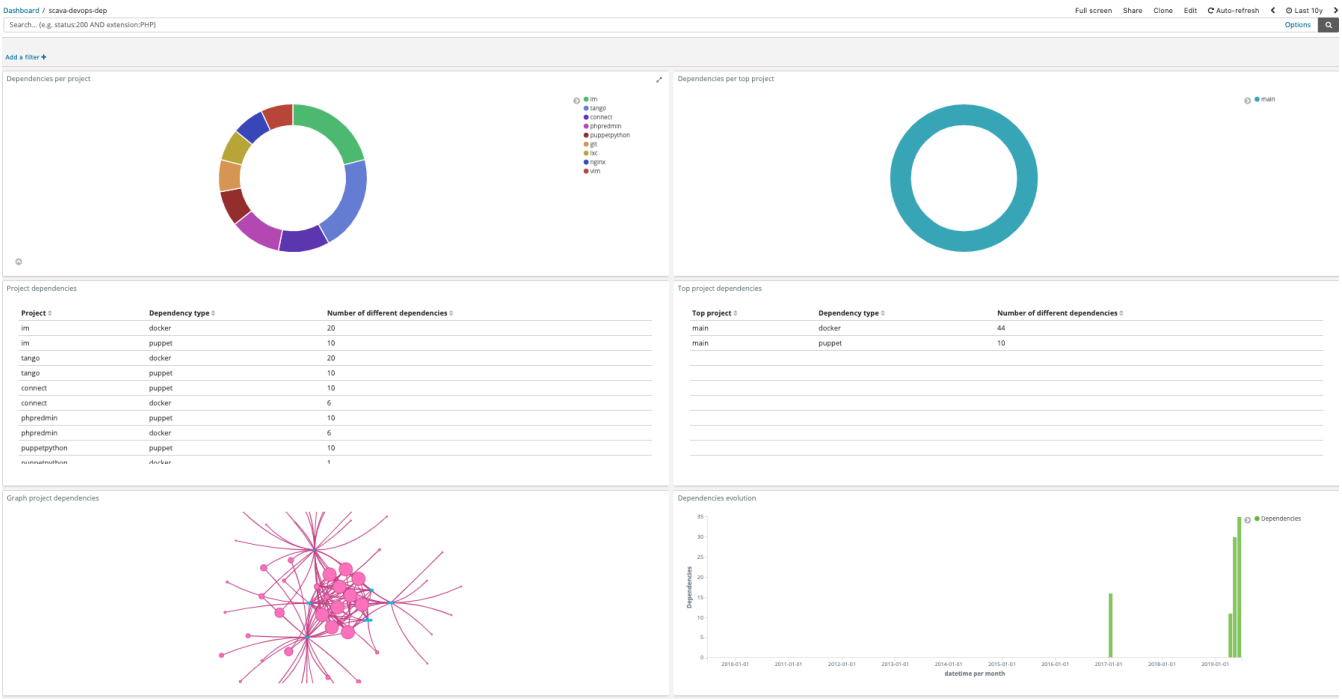

Figure 3: The upper part of DevOps Dependencies DashBoard

3 DevOps Dependencies DashBoard

The DevOps Dependencies DashBoard provides information about the dependencies that are detected in each project. In Figure 3 we can see the upper part of the dashboard, where the analysed projects are available and we can select the project, for which we want to show further details. The available information of this part of the dashboard contains the number of dependencies that are detected in each project, the evolution of the number of dependencies through time about each project and the representation of dependencies of a project as a dependency graph. This graph can help us analyze which other components a project is relying on.

If we want to list the projects that use a specific project as a third-party component, we can use the search box of the dashboard, providing as search term the name of the one that we are interested in. We can find the list of the projects that are returned from this query in the table "Project dependencies".

As shown in Figure 4, the lower part of the dashboard, we can see the actual list of dependencies of each project. The available information is the name of the dependency, the version of the dependency (if it is available) and the project that each dependency refers to. Also, we can find the dependencies for which a new version is available. For them, the name of the dependency along with the current and the new version numbers are shown.

The last information that is provided from the dashboard is the relations between the projects that are analysed by the platform. If a project is declared as a third-party dependency in another analysed project, we can find an entry about this relation in the table "Configuration project relations (details)". The names of the related projects and type of their relation is shown. A graph representation of these relations is also available.

The metric providers that produce the metrics that are incorporated in this dashboard are the following (the details about the metrics are available in deliverable D4.4):

- org.eclipse.scava.metricprovider.historic.configuration.puppet.dependencies

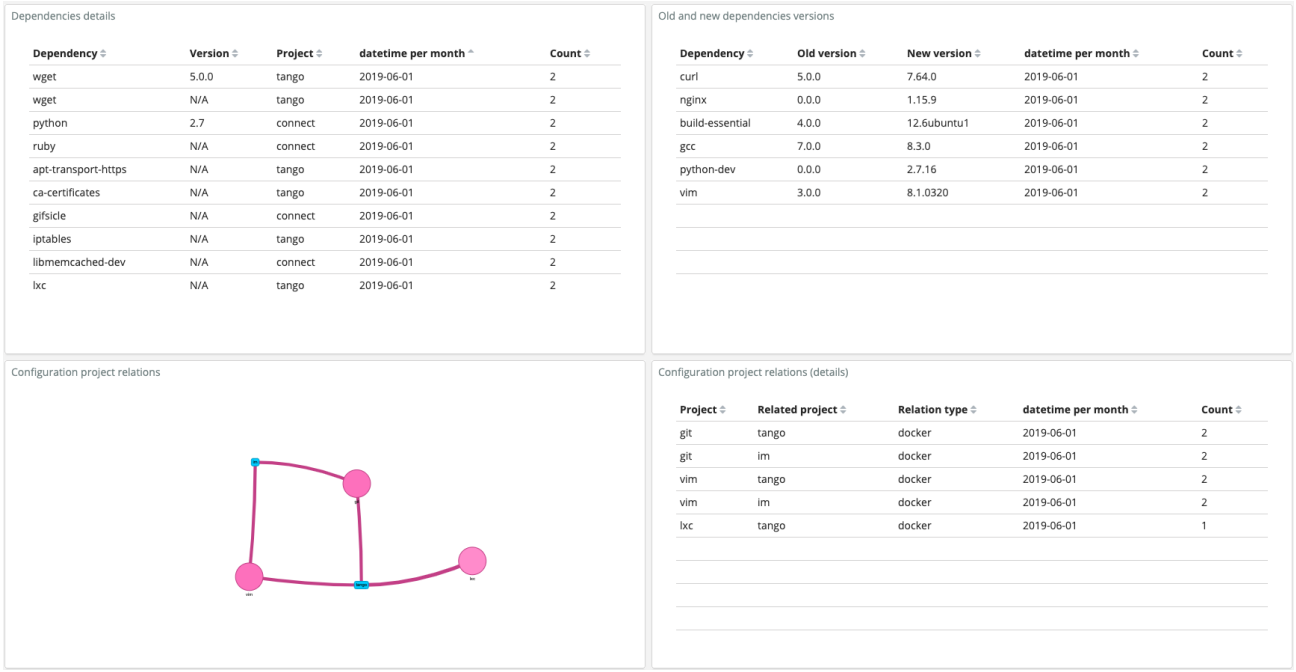

Figure 4: The lower part of DevOps Dependencies DashBoard

- org.eclipse.scava.metricprovider.historic.configuration.docker.dependencies
- org.eclipse.scava.metricprovider.trans.configuration.puppet.dependencies
- org.eclipse.scava.metricprovider.trans.configuration.docker.dependencies
- org.eclipse.scava.metricprovider.trans.newversion.docker
- org.eclipse.scava.metricprovider.trans.newversion.puppet
- org.eclipse.scava.metricprovider.trans.configuration.projects.relations

4 Satisfaction of CROSSMINER Requirements

Req. No.	Requirement	Priority	Status
D40	The configuration code analysis tool shall create a directed acyclic graph of all the third party libraries on which a project depends	SHALL	Full: The DevOps Dependencies Dashboard showcases a graph representation based on the outputs of the System Configuration Code Analyser
D41	The configuration code analysis tool shall provide an interface to extract data from the dependency graph	SHALL	Full: The DevOps Dependencies Dashboard provides an interactive graph, which the user can search either with the dedicated search box of the dashboard or by selecting specific nodes of the graph
D43	Able to identify and list projects that use a “provider project” as a third-party component through its API	SHOULD	Full: The DevOps Dependencies Dashboard provides this functionality by the search box of the dashboard, which results to the filtering of the projects that use the project that was used as the search query as a third-party component
U81	Able to identify and list the third-party components used in a project	SHALL	Full: he DevOps Dependencies Dashboard lists all the detected dependencies
U82	Able to identify and list projects that use a "provider project" as a third-party component through its API	SHOULD	cf. D43
U84	Able to search the result of dependency analysis of projects	SHALL	Full: The DevOps Dependencies Dashboard provides this functionality by the search box of the dashboard

Table 1: Satisfaction of CROSSMINER requirements extracted from **D1.1 – Project Requirements**.

References

[1] Jez Humble and David Farley. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (Adobe Reader)*. Pearson Education, 2010.