# What Quality Aspects Influence the Adoption of Docker Images?

GIOVANNI ROSA, STAKE Lab, University of Molise, Italy
SIMONE SCALABRINO, STAKE Lab, University of Molise, Italy
GABRIELE BAVOTA, Software Institute, USI Università della Svizzera Italiana, Switzerland
ROCCO OLIVETO, STAKE Lab, University of Molise, Italy

Docker is a containerization technology that allows developers to ship software applications along with their dependencies in Docker images. Developers can extend existing images using them as base images when writing Dockerfiles. However, a lot of alternative functionally-equivalent base images are available. While many studies define and evaluate quality features that can be extracted from Docker artifacts, it is still unclear what are the criteria on which developers choose a base image over another.

In this paper, we aim to fill this gap. First, we conduct a literature review through which we define a taxonomy of quality features, identifying two main groups: *Configuration-related features* (*i.e.,* mainly related to the Dockerfile and image build process), and *externally observable features* (*i.e.,* what the Docker image users can observe). Second, we ran an empirical study considering the developers' preference for 2,441 Docker images in 1,911 open-source software projects. We want to understand (i) how the *externally observable features* influence the developers' preferences, and (ii) how they are related to the *configuration-related features*. Our results pave the way to the definition of a reliable quality measure for Docker artifacts, along with tools that support developers for a quality-aware development of them.

CCS Concepts: • **Software and its engineering → Software notations and tools**.

Additional Key Words and Phrases: Empirical software engineering, Software maintenance, Container virtualization, Docker

**ACM Reference Format:**
Giovanni Rosa, Simone Scalabrino, Gabriele Bavota, and Rocco Oliveto. 2018. What Quality Aspects Influence the Adoption of Docker Images?. In . ACM, New York, NY, USA, 29 pages. https://doi.org/XXXXXXX.XXXXXXX

## 1 INTRODUCTION

Deploying software and keeping it in operation is technically challenging. Moreover, the production environment is rarely identical to the development environment, which increases the risk of failures, *e.g.,* due to missing runtime dependencies, or even security vulnerabilities.

Containerization allows developers to ship software applications along with dependencies and the execution environment. Thanks to containerization, it is possible to run the application on any system [5] and test it in the same environment used in production. Docker[1] is one of the most popular containerization platforms used in the DevOps workflow. Docker allows releasing applications with their dependencies through containers (*i.e.,* virtual environments) sharing the kernel of the host operating system. The specification file of a Docker image is called Dockerfile. DockerHub[2]

[1] https://www.docker.com/
[2] https://hub.docker.com/


Manuscript submitted to ACM