

Rapid Software Composition by Assessing Untrusted Components

Today no organizations build software-intensive systems from the ground up; everyone builds applications on top of existing platforms, frameworks, components, and tools. Hence today's software development paradigm challenges developers to build trusted systems that include increasing numbers of untrusted components.

The software industry as a whole has increasingly adopted open source and commercial components as fundamental building-blocks of their systems. The U.S. Army has recently created an initiative to deliver capability more quickly—the Rapid Capability Office. While third-party components, including open source components, have long been one of the foundations for DoD software, there is a recognition that we may need to adopt greater numbers of such components, and in a more agile fashion. There is likewise a recognition that, to deliver capabilities more rapidly, we may need to take on more risk.

Our research challenge is: how to speed up the component qualification, analysis, and evaluation process while choosing appropriate levels of risk? Component scorecards, automatically constructed, can provide rapid insight into many important quality attributes and community attributes. These indicators can then be used to determine risk and to plan additional (human-intensive) analyses [1].

[1] H. Cervantes, J. Ryoo, R. Kazman, "Data-driven selection of application frameworks during architectural design," Proceedings of HICSS 52, January 2019



Example Component Scorecard

QUALITY ATTRIBUTE	TOOL	INDICATOR	COMPONENT	
			Dlib 19.10	OpenCV 3.3.1
Performance	Instrumentation	Time (ms)	44,172	55,978
	gperf	Time (ms)	47,480	58,400
	valgrind callgrind	Instructions (billions)	491	272
Memory	Memcheck	Bytes lost	288	17,127
	Memcheck	Heap usage (Mbytes)	4,591	1,093
Modifiability	DV8	Decoupling level	0.51	0.79
	DV8	Propagation cost	0.31	0.14
	Understand	SLOC	276,825	783,344
Security	FlawfinderRaw	Hits 3+	162	676
Community	CodeMaat	Authors >5 commits	13	234
		Total commits	7,191	18,272

In this research we have shown how to increase both the speed and confidence of the component selection process. We have provided component scorecards based on project health measures and quality attribute indicators that enable the automated early assessment of external components with greater developer confidence, supporting rapid software delivery. Our approach is to apply existing automated analysis techniques and tools (e.g., code and software project repository analyses), following the current industry trend towards DevOps, mapping the extracted information to common quality indicators from DoD projects.

Such scorecards are not the end of analysis, but rather the beginning. They can give rapid insight that allows architects to do triage, quickly and with confidence eliminating some components and providing a context for additional deeper analysis on the remaining components. Raw scores can be aggregated using weighting functions that reflect the importance of each measure to the project, for example

$$\begin{aligned} \text{Score} = & (wM_1 * wM_2) \\ & + 2(wP_1 * \log wP_2) \\ & + 3(wS_1) \end{aligned}$$

Furthermore, by automating the analyses, components can be re-qualified every time they change for relatively low incremental costs. If an indicator changes in a non-trivial way, a deeper analysis can then be performed.

In this way we can balance the needs of agility with the needs of proper component qualification.

Copyright 2018 Carnegie Mellon University. All Rights Reserved.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT. [DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

Internal use:* Permission to reproduce this material and to prepare derivative works from this material for internal use is granted, provided the copyright and "No Warranty" statements are included with all reproductions and derivative works.

External use:* This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other external and/or commercial use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

* These restrictions do not apply to U.S. government entities.

Carnegie Mellon® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

DM18-1141