Cinders: The continuous integration and delivery architecture framework: Journal-first selected article - Extended abstract

Downloaded from: https://research.chalmers.se, 2022-02-20 00:09 UTC

Citation for the original published paper (version of record):
Stahl, D., Bosch, J. (2018)
Cinders: The continuous integration and delivery architecture framework: Journal-first selected article - Extended abstract
ACM International Conference Proceeding Series: 128-129
http://dx.doi.org/10.1145/3202710.3203165

N.B. When citing this work, cite the original published paper.
Cinders: The Continuous Integration and Delivery Architecture Framework

Journal-First Selected Article – Extended Abstract

Daniel Ståhl
Ericsson AB
Linköping, Sweden
daniel.stahl@ericsson.com

Jan Bosch
Chalmers University of Technology
Gothenburg, Sweden
jan@janbosch.com

ABSTRACT

This extended abstract summarizes an article, which has been published in Information and Software Technology and was selected for the Journal-First presentations at the International Conference on Software and System Process (ICSSP 2018).


CCS CONCEPTS

• Software and its engineering → Architecture description languages; Software development methods; Software configuration management and version control systems; Agile software development;

KEYWORDS

cinders; software integration; software testing; continuous integration; continuous delivery; architecture framework

ACM Reference Format:

1 SUMMARY

The popular agile practices of continuous integration and delivery have become an essential part of the software development process in many companies, yet effective methods and tools to support design, description and communication of continuous integration and delivery systems are lacking. This paper addresses the problem of constructing systems for rapidly and frequently transforming source code changes into verified and deliverable software product revisions with known content, known functionality and known quality — through software integration and test — in a controlled and methodical way.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

ICSSP ’18, May 26–27, 2018, Gothenburg, Sweden
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-6459-1/18/05.
https://doi.org/10.1145/3202710.3203165
2 NEW INSIGHTS

There have been two noteworthy developments since the original journal article was published. First, as noted in the summary above, there is a lack of consensus in industry as well as in research as to the exact meaning of terms such as continuous integration and continuous delivery. In subsequent work we have investigated this further and analyzed usage of the terms in published literature. Based on this investigation we then propose less ambiguous definitions of a set of related terms [13]. Relevant in this context, we argue that continuous integration is a “developer practice where developers integrate their work frequently, usually each person integrates at least daily, leading to multiple integrations per day” and that continuous delivery is a “development practice where every change is treated as a potential release candidate to be frequently and rapidly evaluated through one’s continuous delivery pipeline, and that one is always able to deploy and/or release the latest working version, but may decide not to, e.g. for business reasons”. These definitions are in line with, but more elaborate than, the definition of a continuous integration and delivery system provided in this article.

The second development is that the Eiffel protocol for real time automated documentation of continuous integration and delivery activities, which is mentioned in passing in the original article, has since been released as open source along with multiple service implementations [1]. The Eiffel protocol is based on similar concepts as Cinders, and even though there is currently no open source implementation of Cinders descriptions generated from Eiffel data, the data model of Cinders is well suited for such automated generation and would fit well into the Eiffel community’s implementation architecture.

3 CONCLUSION

In this article we establish that the construction of continuous integration and delivery systems is an area of considerable investment in the industry, yet lacking in supporting tools and methods, coinciding with a tendency by studied industry cases to not address its challenges using as rigorous an approach as in regular product development. Based on thematic analysis of statements in literature, twelve requirements for an architectural framework for continuous integration and delivery are phrased. Using these requirements, existing architecture frameworks as listed by ISO/IEC/IEEE [6] are evaluated, finding that none satisfactorily addresses the identified requirements.

Consequently Cinders, a new architecture framework designed to address the identified requirements, is presented. Influenced by prominent enterprise and software architecture frameworks, Cinders offers four separate viewpoints of the same underlying data model, with six optional layers of additional information which can be used to adjust the focus and level of detail within each of those viewpoints, as fits the particular use case and circumstances. This framework is then applied in a workshop format in two separate industry cases, and interviews are conducted with a total of twelve practitioners.

Based on this work it is shown that a single architectural framework can be designed to encompass the previous continuous integration and delivery modeling techniques ASIF and CIViT, representing their specific concerns as viewpoints rendered from the same underlying data model. It is also shown that this architecture framework represents an improvement over these techniques, in that it separates different types of entity relationships into separate viewpoints, allows the level of abstraction of each viewpoint to be modified, shows confidence afforded by conducted activities, represents both manual and automated test activities, can map activities onto physical environments and can visualize overlapping test activities. In workshops and interviews with practitioners of continuous integration and delivery in two separate companies it is confirmed that Cinders is viewed as relevant and useful in an industry setting, even while areas of possible improvement are identified. Therefore we find that Cinders represents a significant step forward in continuous integration and delivery architecture design and description, constituting a relevant and helpful tool for industry professionals to better document, analyze and communicate their systems.

REFERENCES