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

Downloaded from: https://research.chalmers.se, 2019-01-10 10:14 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.

research.chalmers.se offers the possibility of retrieving research publications produced at Chalmers University of Technology. It covers all kind of research output: articles, dissertations, conference papers, reports etc. since 2004. research.chalmers.se is administrated and maintained by Chalmers Library
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 ICSSP 2018. In this paper, we investigate the applicability of existing architectural frameworks to facilitate the design and description of continuous integration and delivery systems.

Historically, the problem of transforming lines of source code into functioning, verified products running in their target environment could be regarded as a question of enterprise architecture: of organizational responsibilities and manual processes. With the advent and growth of continuous integration [2, 3] and delivery [4, 5], however, and the automation this brings, this is increasingly becoming a domain of software engineering: we see ever more sophisticated software systems being constructed, with the purpose of compiling, integrating, testing, delivering and deploying other software.

While such systems are generally perceived as adding value and increasing the efficiency of the development project, we have found in previous work that the exact nature of these benefits is highly uncertain and varies from case to case [8]. Furthermore, even though there are numerous popular tools that do much of the heavy lifting in these integration systems, they only address isolated parts of a very large problem domain. In all our industry case studies [8, 9, 11, 12] we have never found a complete off-the-shelf solution for continuous integration. Rather, the integration systems we find often use similar tools, but configured differently, put to different purposes and integrated with one another in varying constellations. Not surprisingly, a review of literature reveals that reported continuous integration systems display a high degree of variance [10]. In other words, as a rule, continuous integration and delivery systems are highly customized and purpose-built software products in their own right.

This recognition leads up to the research question that drives the study presented in this article: In what way can the paradigm of architecture frameworks favorably be applied to facilitate the design and description of continuous integration and delivery systems?

Similarly to the variance in system design, there is little consensus on the exact definition of continuous integration and delivery, particularly as opposed to related terms such as continuous testing, continuous release or continuous deployment. For the purposes of this paper, we use the term continuous integration and delivery system to mean any system of automated activities performed in order to transform source code into working and potentially shippable and deployable products with known quality, content and functionality, i.e. including compilation, linking, packaging, testing, profiling, documentation generation and much more, serving to ensure that “the software can be released to production at any time” [4].

In this paper we investigate the applicability of existing architectural frameworks and two built-for-purpose modeling techniques...
— ASIF [10] and CIViT [7] — to the problem of designing and docu-
menting continuous integration and delivery systems. Based on this
investigation a new architectural framework, Cinders, is proposed
and subsequently evaluated in an industry context.

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 de-
dinitions 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 work-
ing 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 con-
cepts 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 genera-
tion and would fit well into the Eiffel community’s implementation
architecture.

3 CONCLUSION
In this article we establish that the construction of continuous inte-
gration and delivery systems is an area of considerable investment
in the industry, yet lacking in supporting tools and methods, co-
inciding 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 frame-
work can be designed to encompass the previous continuous in-
tegration and delivery modeling techniques ASIF and CIViT, rep-
resenting 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 ac-
tivities 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 iden-
tified. 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
accessed 27-March-2018].
Wesley Professional.
articles/continuousIntegration.html. (2006). [Online; accessed 12-February-
2016].
releases through build, test, and deployment automation. Pearson Education.
ties to support continuous integration: A multiple case study. In Agile Processes
in industry software product development: A case study. In The 12th IASTED
International Conference on Software Engineering. 736–743.
[9] Daniel Ståhl and Jan Bosch. 2014. Automated software integration flows in
industry: a multiple-case study. In Companion Proceedings of the 36th Interna-
differences in industry software development. Journal of Systems and Software
modeling: a multiple-case study. In Proceedings of the 38th International Conference
scale continuous integration and delivery: Deployment, usage and validation of
DevOps: Beyond the Buzz, What Does It All Mean?. In 43rd Euroisico Conference