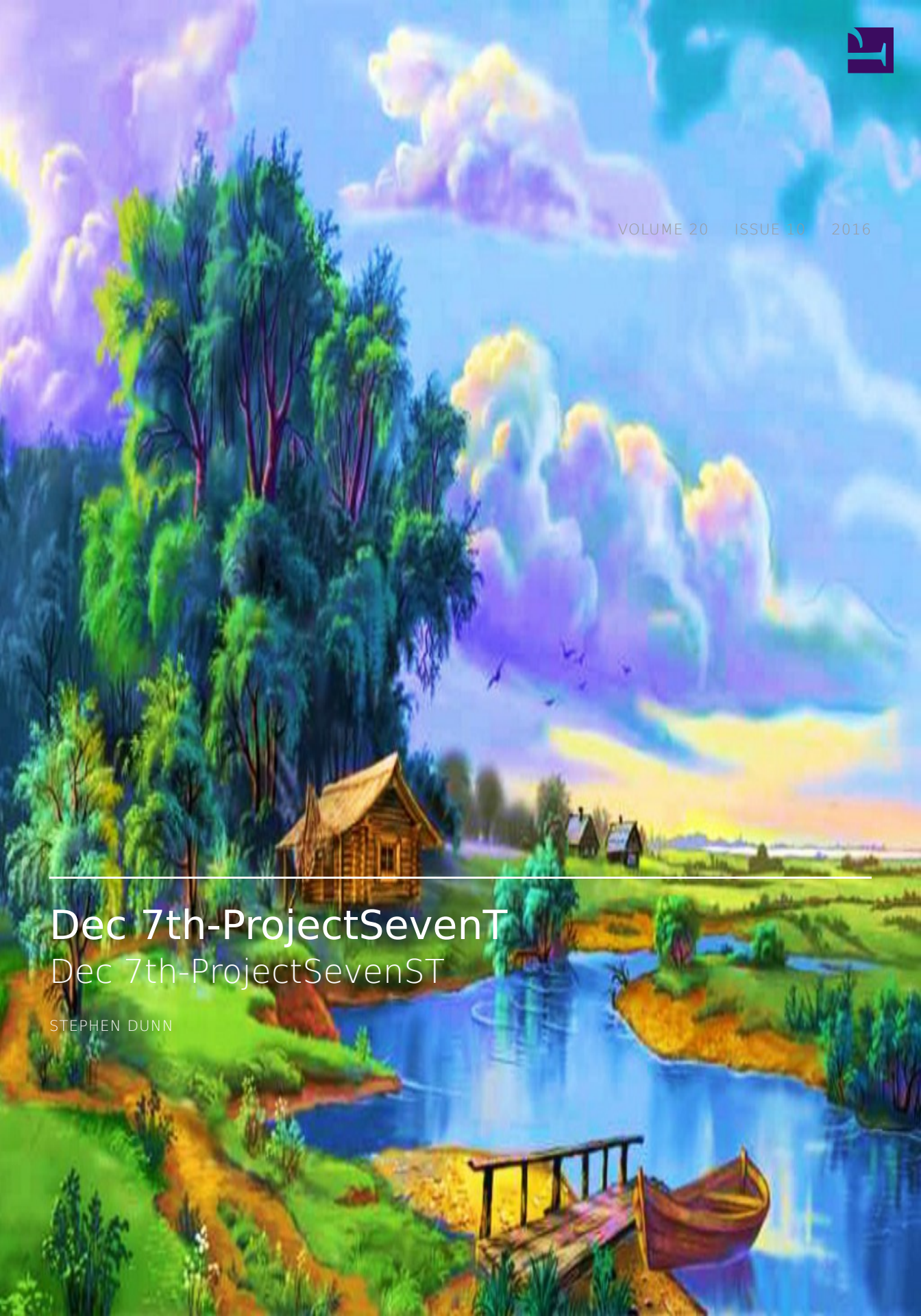

VOLUME 20 ISSUE 10 2016

Dec 7th-ProjectSevenT

Dec 7th-ProjectSevenST

STEPHEN DUNN

DEC 7TH-PROJECTSEVENT :DEC 7TH-PROJECTSEVENST

Stephen Dunn

JAPP

A common practice of software testing is that testing is performed by an independent group of testers after the functionality is developed, before it is shipped to the customer.[47] This practice often results in the testing phase being used as a project buffer to compensate for project delays, thereby compromising the time devoted to testing.[48]

Another practice is to start software testing at the same moment the project starts and it is a continuous process until the project finishes.[49]

Further information: Capability Maturity Model Integration and Waterfall model

A common practice of software testing is that testing is performed by an independent group of testers after the functionality is developed, before it is shipped to the customer.[47] This practice often results in the testing phase being used as a project buffer to compensate for project delays, thereby compromising the time devoted to testing.[48]

Another practice is to start software testing at the same moment the project starts and it is a continuous process until the project finishes.[49]

Further information: Capability Maturity Model Integration and Waterfall model

JSectionn

It has been proved that each class is strictly included into the next. For instance, testing when we assume that the behavior of the implementation under test can be denoted by a deterministic finite-state machine for some known finite sets of inputs and outputs and with some known number of states belongs to Class I (and all subsequent classes). However, if the number of states is not known, then it only belongs to all classes from Class II on. If the implementation under test must be a deterministic finite-state machine failing the specification for a single trace (and its continuations), and its number of states is unknown, then it only belongs to classes from Class III on. Testing temporal machines where transitions are triggered if inputs are produced within some real-bounded interval only belongs to classes from Class IV on, whereas testing many non-deterministic systems only belongs to Class V (but not all, and some even belong to Class I). The inclusion into Class I does not require the simplicity of the assumed computation model, as some testing cases involving implementations written in any programming language, and testing implementations defined as machines depending on continuous magnitudes, have been proved to be in Class I. Other elaborated cases, such as the testing framework by Matthew Hennessy under must semantics, and temporal machines with rational timeouts, belong to Class II.

JREFERENCES

In contrast, some emerging software disciplines such as extreme programming and the agile software development movement, adhere to a "test-driven software development" model. In this process, unit tests are written first, by the software engineers (often with pair programming in the extreme programming methodology). Of course these tests fail initially; as they are expected to. Then as code is written it passes incrementally larger portions of the test suites. The test suites are continuously updated as new failure conditions and corner cases are discovered, and they are integrated with any regression tests that are developed. Unit tests are maintained along with the rest of the software source code and generally integrated into the build process (with inherently interactive tests being relegated to a partially manual build acceptance process). The ultimate goal of this test process is to achieve continuous integration where software updates can be published to the public frequently. [50] [51]

This methodology increases the testing effort done by development, before reaching any formal testing team. In some other development models, most of the test execution occurs after the requirements have been defined and the coding process has been completed.

Volume20, 2016 test url,

DOI: 10.18848/7647-4738/CGP/V20I10/1-25 VOLUME: 20 ISSUE: 10
Permissions: cg-support@commongroundpublishing.com