# A New Taxonomy of Software Testing Approaches

## Seeking More Standardized Standards

Samuel Joseph Crawford
crawfs1@mcmaster.ca
Department of Computing and Software, McMaster University

April 22, 2023

## Background

The first step to any formal process is understanding the underlying domain well. Therefore, a systematic and rigorous understanding of software testing approaches is needed to develop formal tools to test software. In my specific case, my motivation was seeing which kinds of testing could be generated automatically by Drasil, "a framework for generating all of the software artifacts for (well understood) research software" (Carette et al., 2021).

Most software testing ontologies seem to focus on the high-level testing process rather than the testing techniques themselves. For example, the terms and definitions (Tebes et al., 2020b) from TestTDO (2020a) mainly focus on parts of the testing process (e.g., test goal, testable entity) and how they relate to one another. Unterkalmsteiner et al. (2014) provide a foundation for classification but do not "do[] not aim at providing a systematic and exhaustive state-of-the-art survey of [either domain]" (p. A:2).

## Methodology

- If a taxonomy doesn't already exist, I should create one!
  - I started with an ad hoc approach, focusing on textbooks trusted at McMaster (Patton, 2006; Peters and Pedrycz, 2000; van Vliet, 2000)
  - We then realized that this was too arbitrary, so I then started from more established sources (ISO/IEC and IEEE, 2022; Washizaki, 2024; Bourque and Fairley, 2014; ISO/IEC and IEEE, 2017; ISO/IEC, 2023; Hamburg and Mogyorodi, 2024)
  - The goal of this approach is to iterate, eventually revisiting the original textbooks, until I build up enough knowledge to encounter diminishing returns (ideally no returns!)
- Since there are many standardized documents about software testing (or software in general), this should be trivial, no?

## In My Experience

NO.

## Examples

(ISO/IEC and IEEE, 2022) is a standard for general concepts related to software testing. However, it is not comprehensive. For example, as shown in Figure 1, most (55 out of 99) testing approaches mentioned in this standard do not have an accompanying definition! Eight of these were present in the previous version of this standard (2013), and nine were present in another IEEE standard (2017) that would have been available upon publication of this standard. However, the presence of a definition does not guarantee that it is useful! See Figure 1 for some examples.

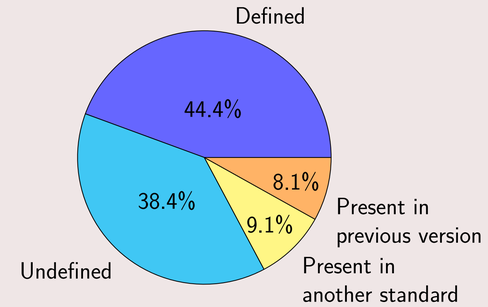


Figure 1: Breakdown of testing approach definitions from ISO/IEC and IEEE (2022).

**3.3809**
**software element**
***1.*** system element that is software
*cf.* system element, software/system element

**3.1486**
**event sequence analysis**
***1.*** per

**3.2697**
**operable**
***1.*** state of

Figure 2: Some less-than-helpful definitions from ISO/IEC and IEEE (2017).

## Conclusions & Future Work

Leveraging the strictly-typed nature of Elm and its model-view-update architecture, we were able to create a simplified multi-user framework, requiring the programmer to write no server-side code. In addition to the upcoming pedagogical study, future work includes a data modelling extension allowing persistent, structured data, an authentication/authorization scheme, a binary data format to reduce network communication, and curriculum development for a TEASync-based summer camp.

## References

Pierre Bourque and Richard E. Fairley, editors. *Guide to the Software Engineering Body of Knowledge, Version 3.0*. IEEE Computer Society Press, Washington, DC, USA, 2014. ISBN 0-7695-5166-1. URL www.swebok.org.

Jacques Carette, Spencer Smith, Jason Balaci, Ting-Yu Wu, Samuel Crawford, Dong Chen, Dan Szymczak, Brooks MacLachlan, Dan Scime, and Maryyam Niazi. Drasil, February 2021. URL https://github.com/JacquesCarette/Drasil/tree/v0.1-alpha.

Matthias Hamburg and Gary Mogyorodi, editors. ISTQB Glossary, v4.3, 2024. URL https://glossary.istqb.org/en_US/search.

ISO/IEC. ISO/IEC 25010:2023 - Systems and software engineering –Systems and software Quality Requirements and Evaluation (SQuaRE) –Product quality model. *ISO/IEC 25010:2023*, November 2023. URL https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-2:v1:en.

ISO/IEC and IEEE. ISO/IEC/IEEE International Standard - Systems and software engineering –Software testing –Part 1: General concepts. *ISO/IEC/IEEE 29119-1:2013*, September 2013. doi: 10.1109/IEEESTD.2013.6588537.

ISO/IEC and IEEE. ISO/IEC/IEEE International Standard - Systems and software engineering–Vocabulary. *ISO/IEC/IEEE 24765:2017(E)*, September 2017. doi: 10.1109/IEEESTD.2017.8016712.

ISO/IEC and IEEE. ISO/IEC/IEEE International Standard - Systems and software engineering –Software testing –Part 1: General concepts. *ISO/IEC/IEEE 29119-1:2022(E)*, January 2022. doi: 10.1109/IEEESTD.2022.9698145.

Ron Patton. *Software Testing*. Sams Publishing, Indianapolis, IN, USA, 2nd edition, 2006. ISBN 0-672-32798-8.

J.F. Peters and W. Pedrycz. *Software Engineering: An Engineering Approach*. Worldwide series in computer science. John Wiley & Sons, Ltd., 2000. ISBN 978-0-471-18964-0.

Guido Tebes, Luis Olsina, Denis Peppino, and Pablo Becker. TestTDO: A Top-Domain Software Testing Ontology. pages 364–377, Curitiba, Brazil, May 2020a. ISBN 978-1-71381-853-3.

Guido Tebes, Luis Olsina, Denis Peppino, and Pablo Becker. TestTDO_terms_definitions_vfinal.pdf, February 2020b. URL https://drive.google.com/file/d/19TWHd50HF04K6PPvVixQzR6c7HiW2kED/