Putting Software Testing Terminology to the Test

M.A.Sc. Seminar

Samuel Crawford, B.Eng.

McMaster University
Department of Computing and Software

Fall 2024

Table of Contents

Table of Contents

The Need for Standardized Terminology

- Engineering is applied science
- Scientific fields use precise terminology
- Therefore, the same should be true of software engineering!

The Need for Standardized Terminology

- Engineering is applied science
- Scientific fields use precise terminology
- Therefore, the same should be true of software engineering!
- Imagine if other fields used unclear, inconsistent, and incorrect terminology:
  - Force
  - Isotope
  - Phalange

The Need for Standardized Terminology

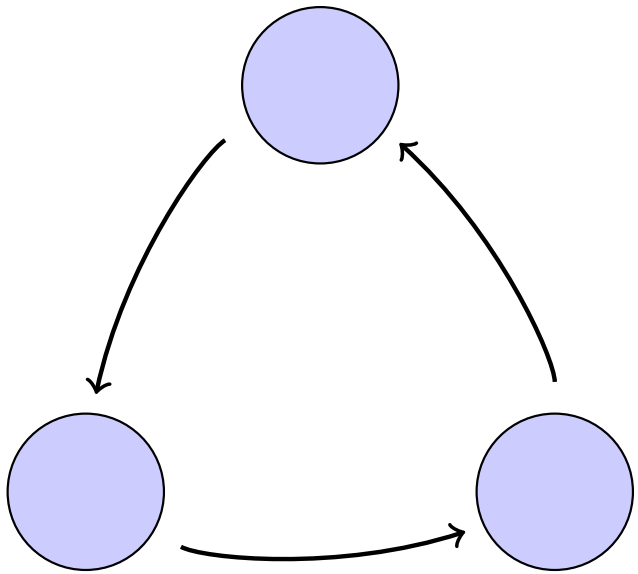
- Engineering is applied science
- Scientific fields use precise terminology
- Therefore, the same should be true of software engineering!
- Imagine if other fields used unclear, inconsistent, and incorrect terminology:
  - Force
  - Isotope
  - Phalange

If software engineering holds code to high standards of clarity, consistency, and robustness, the same should apply to its supporting literature!

Improved Communication

Interorganizational

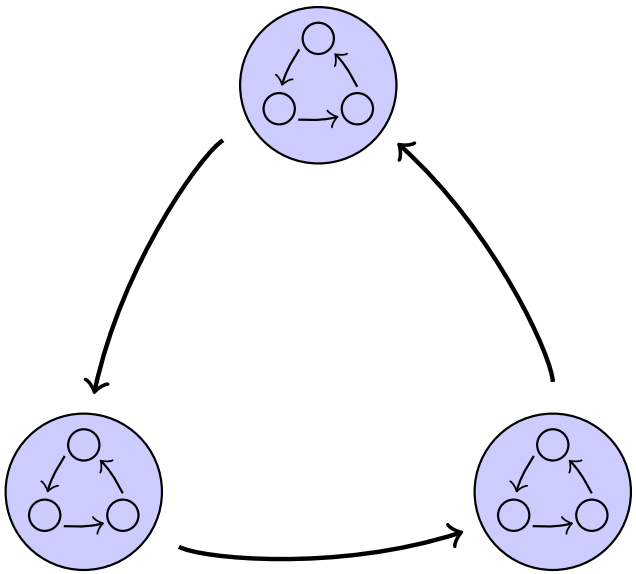
Schools, companies, etc.



Improved Communication

Interorganizational

Schools, companies, etc.



Intraorganizational

Kaner et al. (2011, p. 7) say "complete testing" could require the tester to:

- discover "every bug",
- exhaust the time allocated,
- implement every planned test,
- ...

The Lack of Standardized Terminology

"The Problem"

- Unfortunately, a search for a systematic, rigorous, and complete taxonomy for software testing revealed that the existing ones are inadequate:
  - Tebes et al. (2020) focus on *parts* of the testing process (e.g., test goal, testable entity),
  - Souza et al. (2017) prioritize organizing testing approaches over defining them, and
  - Unterkalmsteiner et al. (2014) focus on the "information linkage or transfer" (p. A:6) between requirements engineering and software testing.

Unstandardized Standards

- The structure of tours is:
  - quite general (ISO/IEC and IEEE, 2022, p. 34)

Unstandardized Standards

- The structure of tours is:
  - quite general (ISO/IEC and IEEE, 2022, p. 34)
  - “organized around a special focus” (Hamburg and Mogyorodi, 2024)

Unstandardized Standards

- Load testing is performed with:
  - loads "between anticipated conditions of low, typical, and peak usage" (ISO/IEC and IEEE, 2022, p. 5)

Unstandardized Standards

- Load testing is performed with:
  - loads "between anticipated conditions of low, typical, and peak usage" (ISO/IEC and IEEE, 2022, p. 5)
  - loads that are as large as possible (Patton, 2006, p. 86)

Unstandardized Standards

- Alpha testing is performed by:
  - "users within the organization developing the software" (ISO/IEC and IEEE, 2017, p. 17)

Unstandardized Standards

- Alpha testing is performed by:
  - “users within the organization developing the software” (ISO/IEC and IEEE, 2017, p. 17)
  - “a small, selected group of potential users” (Washizaki, 2024, p. 5-8)

Unstandardized Standards

- Alpha testing is performed by:
  - "users within the organization developing the software" (ISO/IEC and IEEE, 2017, p. 17)
  - "a small, selected group of potential users" (Washizaki, 2024, p. 5-8)
  - "roles outside the development organization" conducted "in the developer's test environment" (Hamburg and Mogyorodi, 2024)

Unstandardized Standards

- Alpha testing is performed by:
  - "users within the organization developing the software" (ISO/IEC and IEEE, 2017, p. 17)
  - "a small, selected group of potential users" (Washizaki, 2024, p. 5-8)
  - "roles outside the development organization" conducted "in the developer's test environment" (Hamburg and Mogyorodi, 2024)

"Okay testing team, we want to conduct alpha testing on our product. What's our timeline? Budget? Sample size?"

Table of Contents

Acknowledgment

- Dr. Smith and Dr. Carette have been great supervisors in the past and have, both then and now, provided me with valuable guidance and feedback
  - They have helped me refine the scope of this project
  - The project itself was originally posed by Dr. Smith back in 2020!

Acknowledgment

- Dr. Smith and Dr. Carette have been great supervisors in the past and have, both then and now, provided me with valuable guidance and feedback
  - They have helped me refine the scope of this project
  - The project itself was originally posed by Dr. Smith back in 2020!
- The format of this presentation was *heavily* based on a previous presentation by Jason Balaci, who also provided a great thesis template

Acknowledgment

- Dr. Smith and Dr. Carette have been great supervisors in the past and have, both then and now, provided me with valuable guidance and feedback
  - They have helped me refine the scope of this project
  - The project itself was originally posed by Dr. Smith back in 2020!
- The format of this presentation was *heavily* based on a previous presentation by Jason Balaci, who also provided a great thesis template
- The past and current Drasil team have created a truly amazing framework!

Thank you!

Questions?

References I

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

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.

Cem Kaner, James Bach, and Bret Pettichord. *Lessons Learned in Software Testing: A Context-Driven Approach*. John Wiley & Sons, December 2011. ISBN 978-0-471-08112-8. URL `https://www.wiley.com/en-ca/Lessons+Learned+in+Software+Testing%3A+A+Context-Driven+Approach-p-9780471081128`.

References II

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

Erica Souza, Ricardo Falbo, and Nandamudi Vijaykumar. ROoST: Reference Ontology on Software Testing. *Applied Ontology*, 12:1–32, March 2017. doi: 10.3233/AO-170177.

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

Michael Unterkalmsteiner, Robert Feldt, and Tony Gorschek. A Taxonomy for Requirements Engineering and Software Test Alignment. *ACM Transactions on Software Engineering and Methodology*, 23(2):1–38, March 2014. ISSN 1049-331X, 1557-7392. doi: 10.1145/2523088. URL http://arxiv.org/abs/2307.12477. arXiv:2307.12477 [cs].

References III

Hironori Washizaki, editor. *Guide to the Software Engineering Body of Knowledge, Version 4.0*. January 2024. URL `https://waseda.app.box.com/v/SWEBOK4-book`.