Second Committee Meeting

Updated Progress Report

Samuel Crawford

McMaster University

Fall 2025

Table of Contents

Table of Contents

Where Were We?

Introduction

- We wanted to generate test cases in **Drasil**, our software artifact generation framework
  - Started writing test cases manually

Where Were We?

Introduction

- We wanted to generate test cases in **Drasil**, our software artifact generation framework
  - Started writing test cases manually
  - We stopped to understand the domain of software testing to follow its standards

Where Were We?

Introduction

- We wanted to generate test cases in **Drasil**, our software artifact generation framework
  - Started writing test cases manually
  - We stopped to understand the domain of software testing to follow its standards
- What happened?
  - The domain of software testing is *much* larger than we expected
  - Software testing terminology and standards are *not* standardized

Existing Taxonomies?

Introduction

- Existing software testing taxonomies:
  - Tebes et al. (2020)
  - Souza et al. (2017)
  - Firesmith (2015)
  - Unterkalmsteiner et al. (2014)

Existing Taxonomies?

Introduction

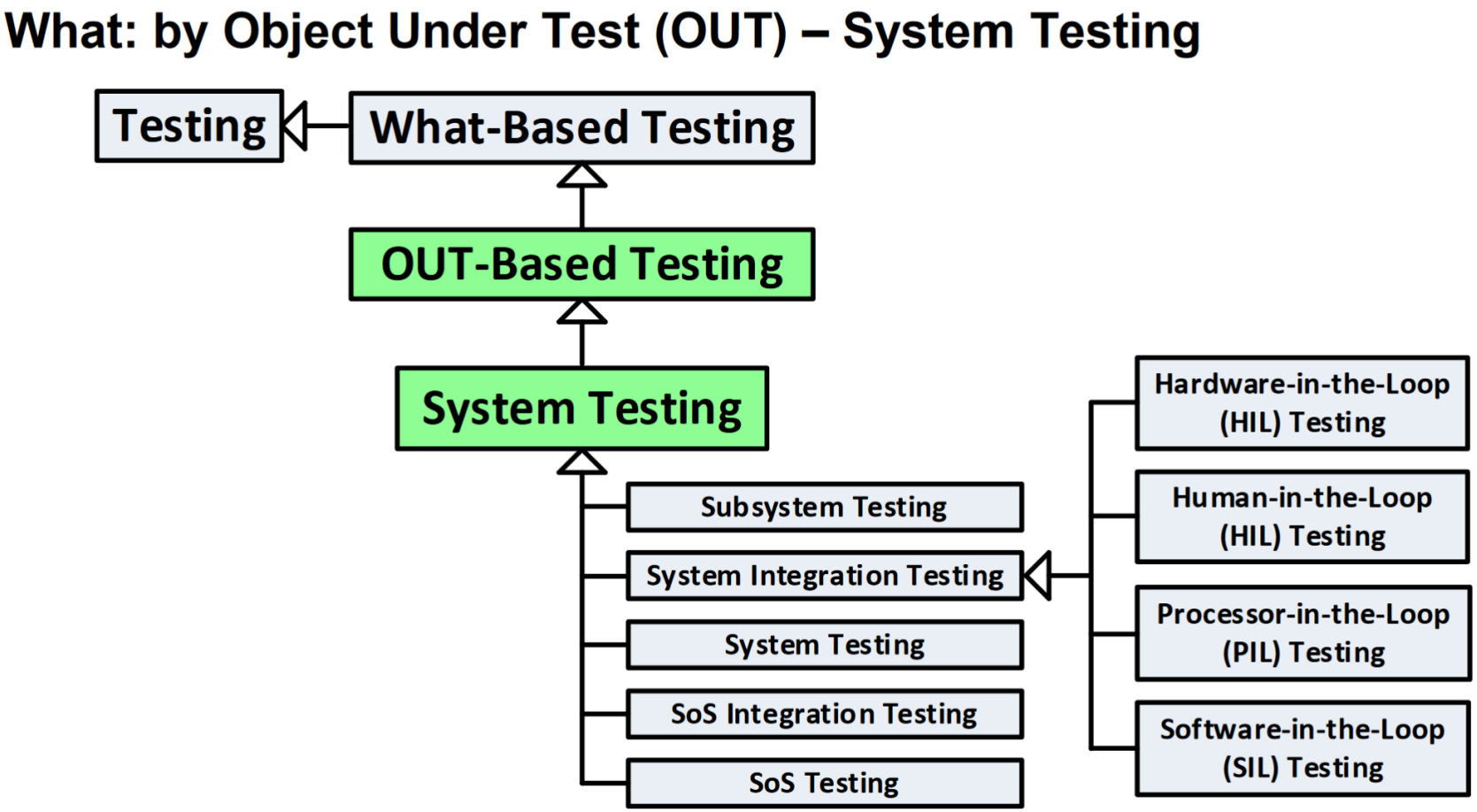
- Existing software testing taxonomies:
  - Tebes et al. (2020)
  - Souza et al. (2017)
  - Firesmith (2015)
  - Unterkalmsteiner et al. (2014)

Focus on:

The Testing Process
Organizing Terminology
Relations between Approaches
Traceability between Stages

Existing Taxonomies?

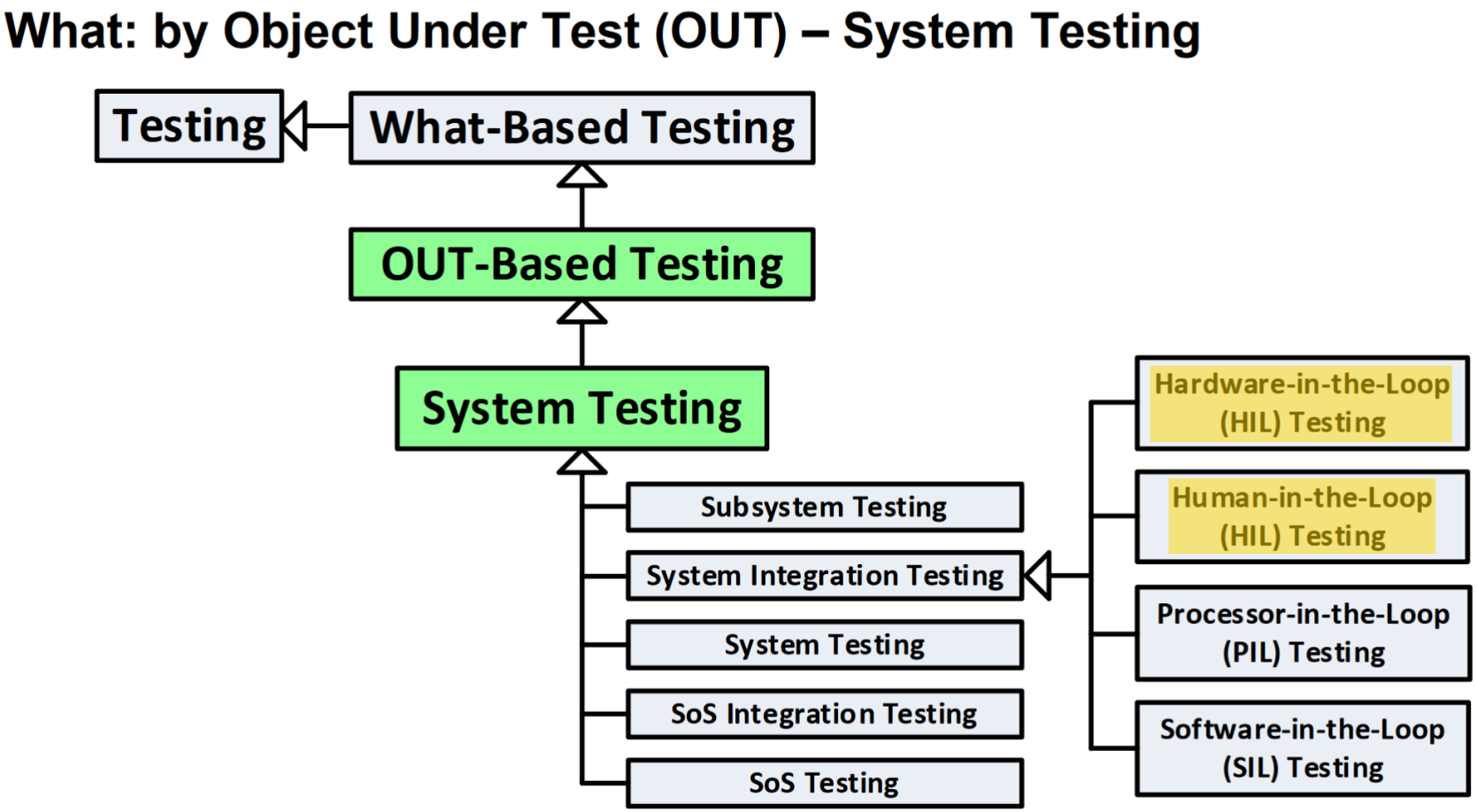
Introduction



(Firesmith, 2015, p. 23)

Existing Taxonomies?

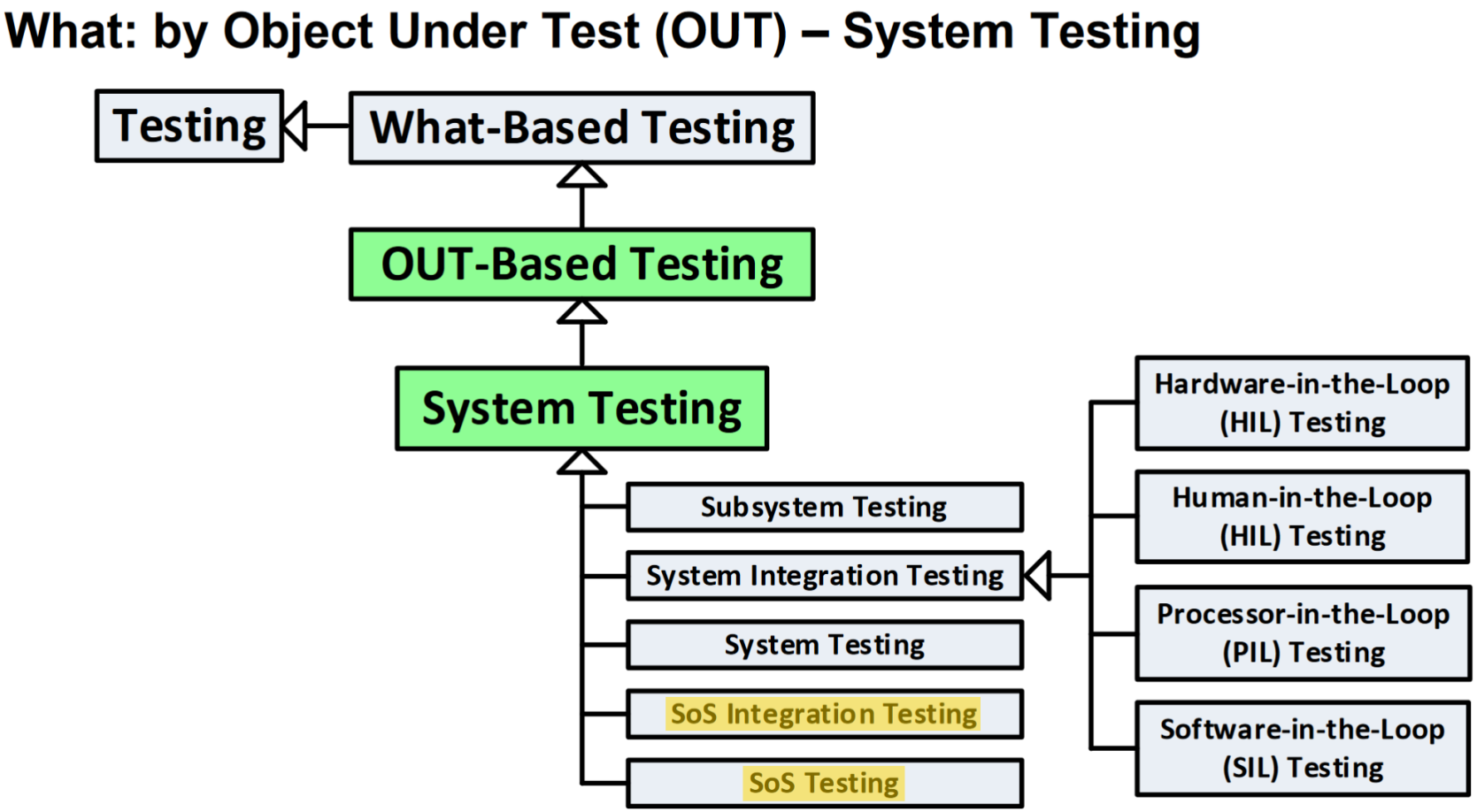
Introduction



Adapted from (Firesmith, 2015, p. 23)

Existing Taxonomies?

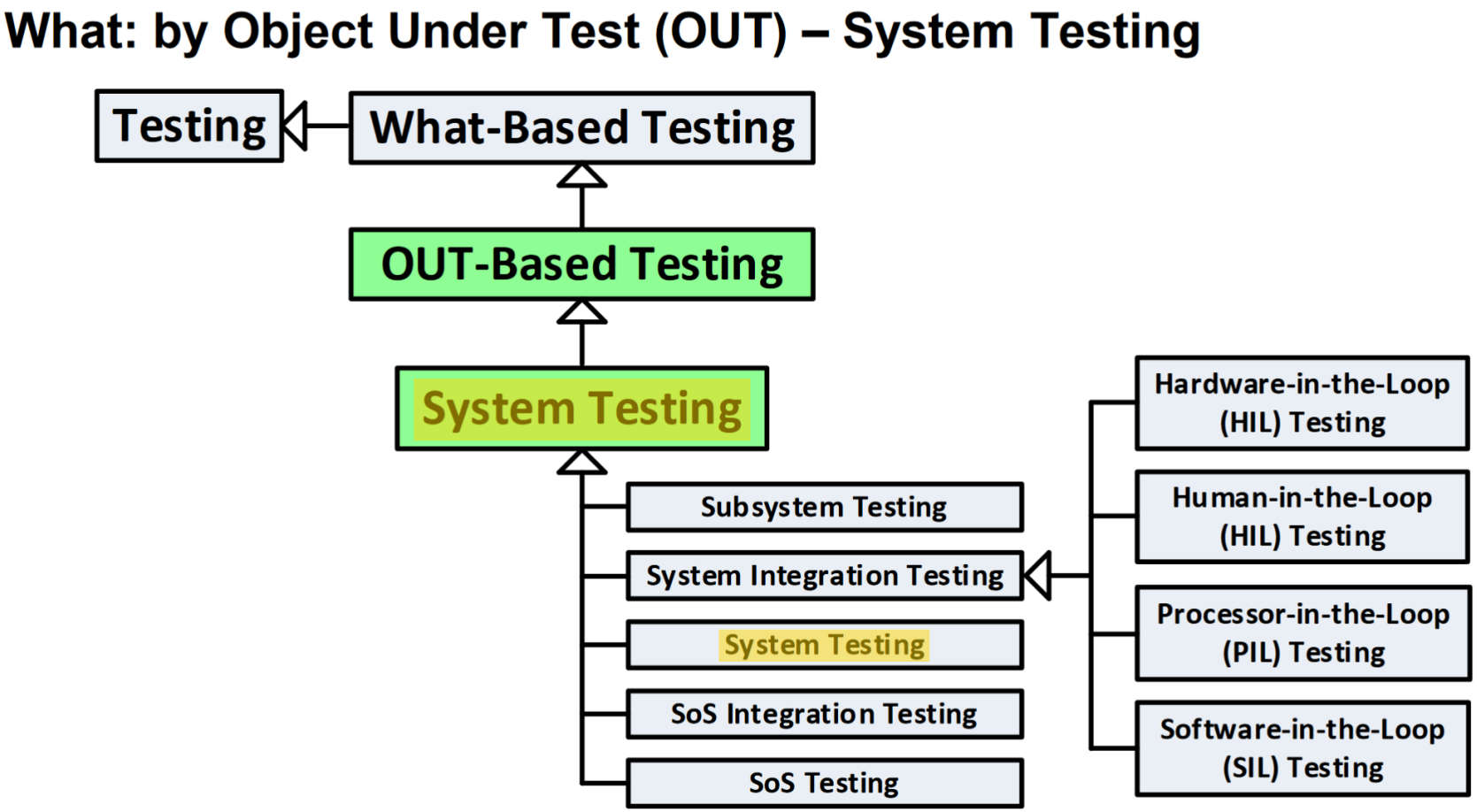
Introduction



Adapted from (Firesmith, 2015, p. 23)

Existing Taxonomies?

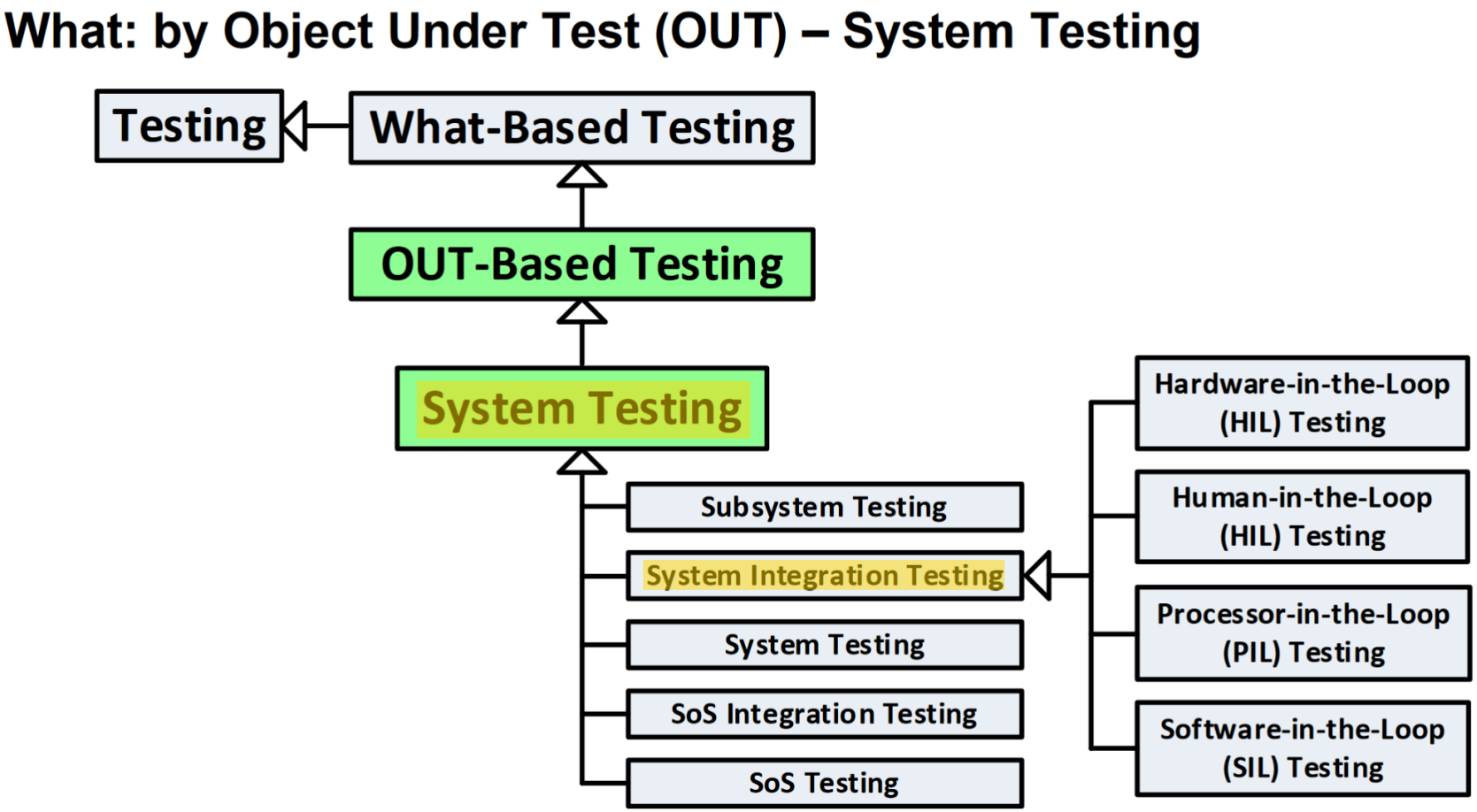
Introduction



Adapted from (Firesmith, 2015, p. 23)

Existing Taxonomies?

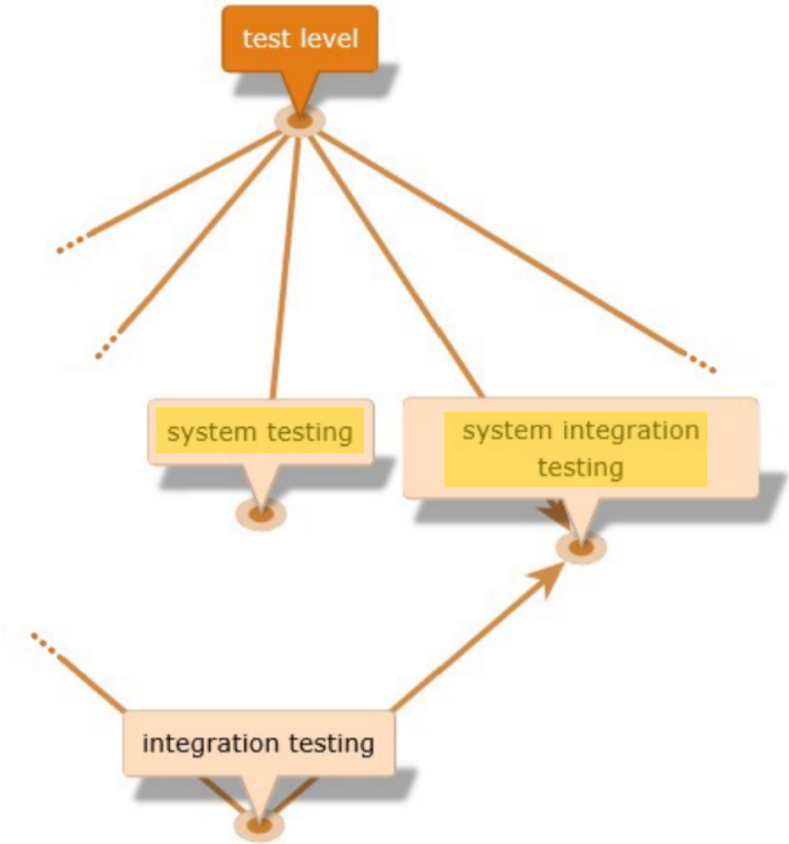
Introduction



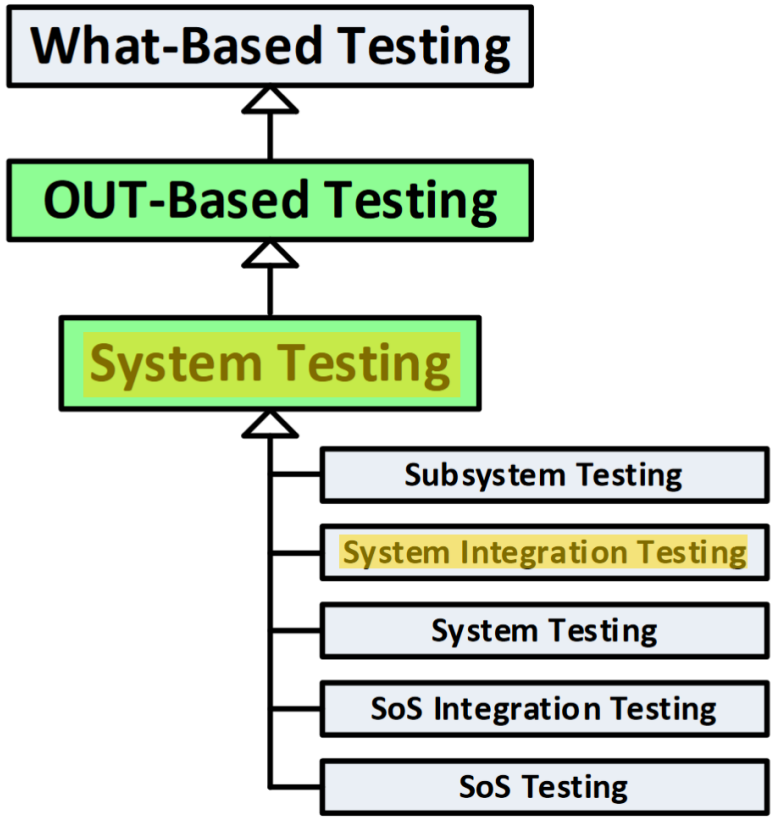
Adapted from (Firesmith, 2015, p. 23)

Existing Taxonomies?

Introduction



Adapted from (Hamburg and Mogyorodi, 2024)



Adapted from (Firesmith, 2015, p. 23)

Table of Contents

Research Questions

Research Question 1

What test approaches do the literature describe?

Research Question 2

Are these descriptions consistent?

Research Question 3

Can we systematically resolve any of these inconsistencies?

Research Question 1

What test approaches do the literature describe?

1. Identify authoritative sources on software testing and "snowball" from them
2. Identify all test approaches and testing-related terms described in these authoritative sources
3. Record all relevant data, including implicit data, for each term identified in step 2; test approach data are comprised of:
   1. Names
   2. Categories
   3. Definitions
   4. Synonyms
   5. Parents
   6. Flaws
4. Repeat steps 1 to 3 for any missing or unclear terms until the stopping criteria is reached

Research Question 2

Are these descriptions consistent?

5. Analyze recorded test approach data for additional flaws
    1. Generate relation graphs
    2. Automatically detect certain classes of flaws
    3. Automatically analyze manually recorded flaws from step 3.6
6. Report results of flaw analysis

Research Question 3

Can we systematically resolve any of these inconsistencies?

7. Provide examples of how to resolve these flaws

- We build a glossary with a row for each test approach

Name	Category	Definition	Parent(s)	Synonym(s)
A/B Testing	Practice (Fig. 2)	Testing "that allows testers to determine which of two systems or components performs better" (pp. 1, 36)	Statistical Testing (pp. 1, 36), ...	Split-Run Testing (pp. 1, 36)

Information from (ISO/IEC and IEEE, 2022)

- We build a glossary with a row for each test approach

Name	Category	Definition	Parent(s)	Synonym(s)
A/B Testing	Practice (Fig. 2)	Testing "that allows testers to determine which of two systems or components performs better" (pp. 1, 36)	Statistical Testing (pp. 1, 36), ...	Split-Run Testing (pp. 1, 36)

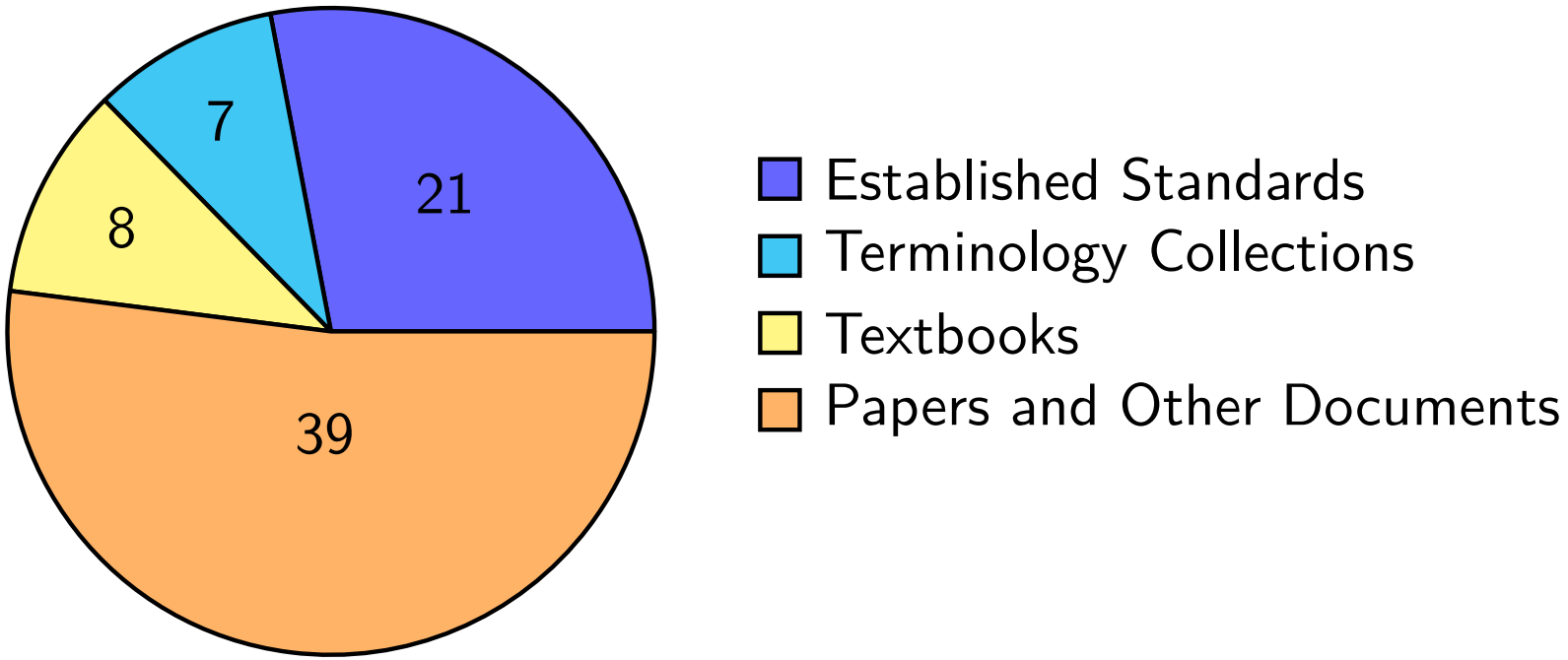
Information from (ISO/IEC and IEEE, 2022)

- We gather this information from sources by looking for:
  - Glossaries, taxonomies, hierarchies, etc.
  - Testing-related terms
  - Terms described *by* other approaches
  - Terms that *imply* other approaches

Methodology

Sources

In total, we investigate 75 sources

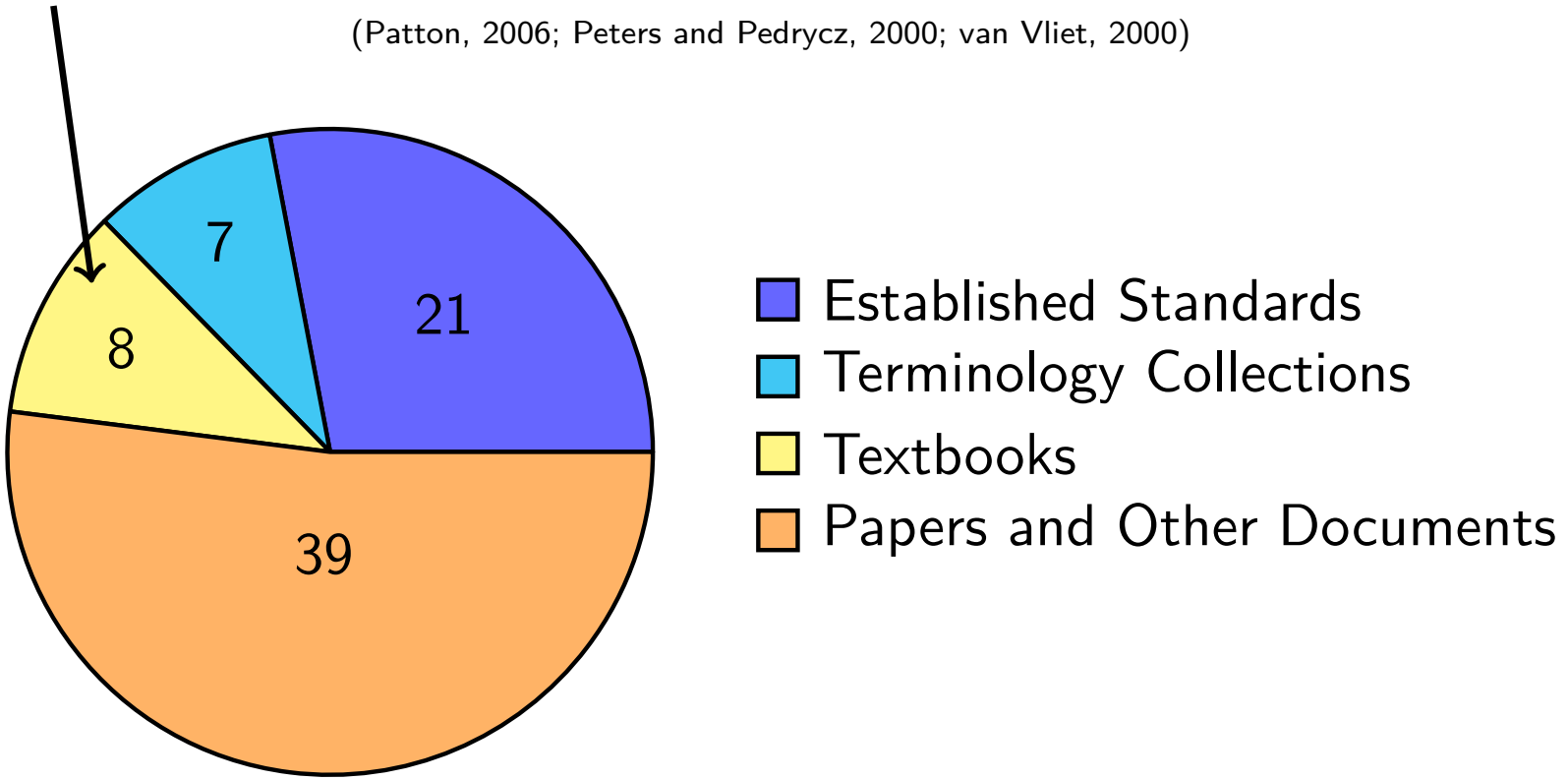


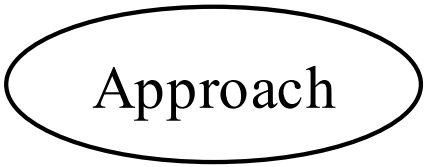
Methodology

Sources

Textbooks used at McMaster were our ad hoc starting points

(Patton, 2006; Peters and Pedrycz, 2000; van Vliet, 2000)

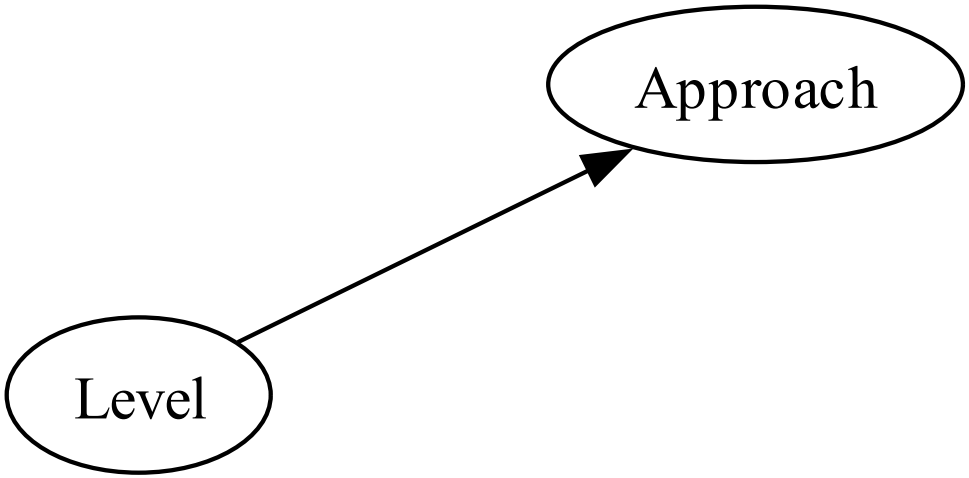




Approach: a "high-level test implementation choice" (ISO/IEC and IEEE, 2022, p. 10) used to "pick the particular test case values" (2017, p. 465)

Methodology

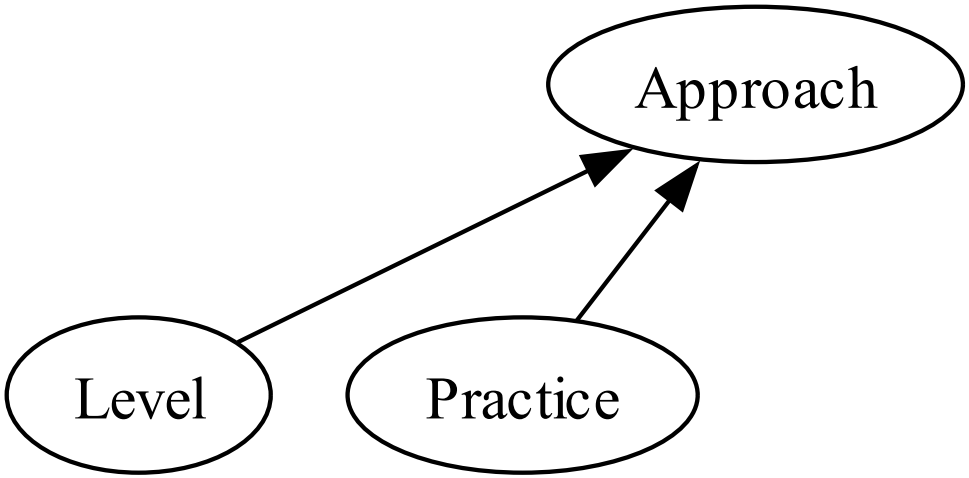
Categories



Level: a stage of testing with "particular objectives and . . . risks", each performed in sequence (ISO/IEC and IEEE, 2022, p. 12; 2021a, p. 6; 2021c, p. 6)

Methodology

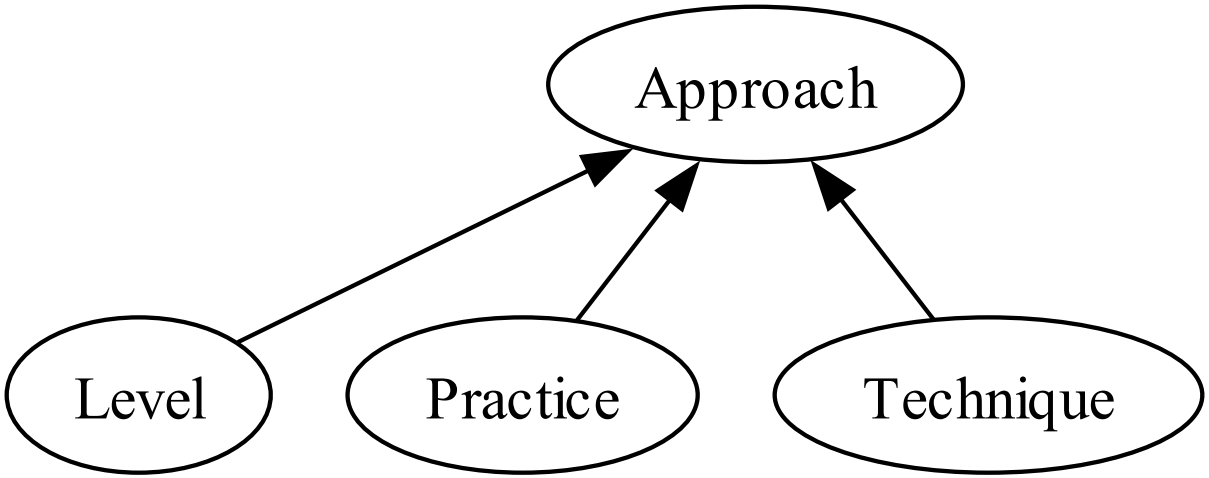
Categories



Practice: a "conceptual framework that can be applied to . . . [a] test process to facilitate testing" (ISO/IEC and IEEE, 2022, p. 14; 2017, p. 471)

Methodology

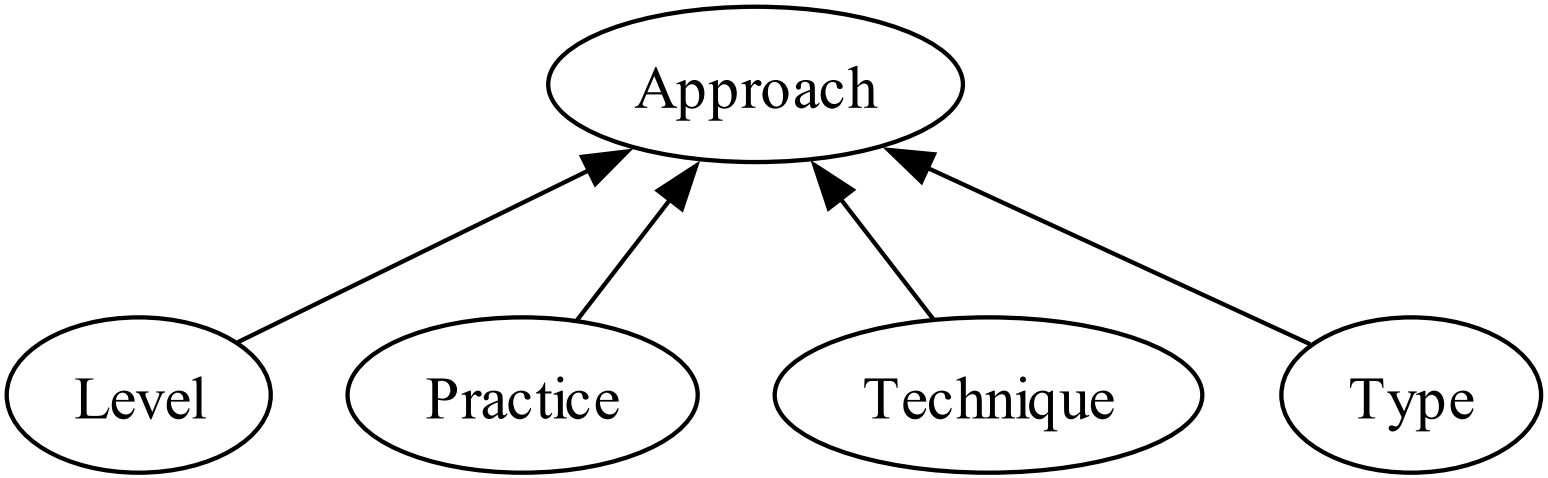
Categories



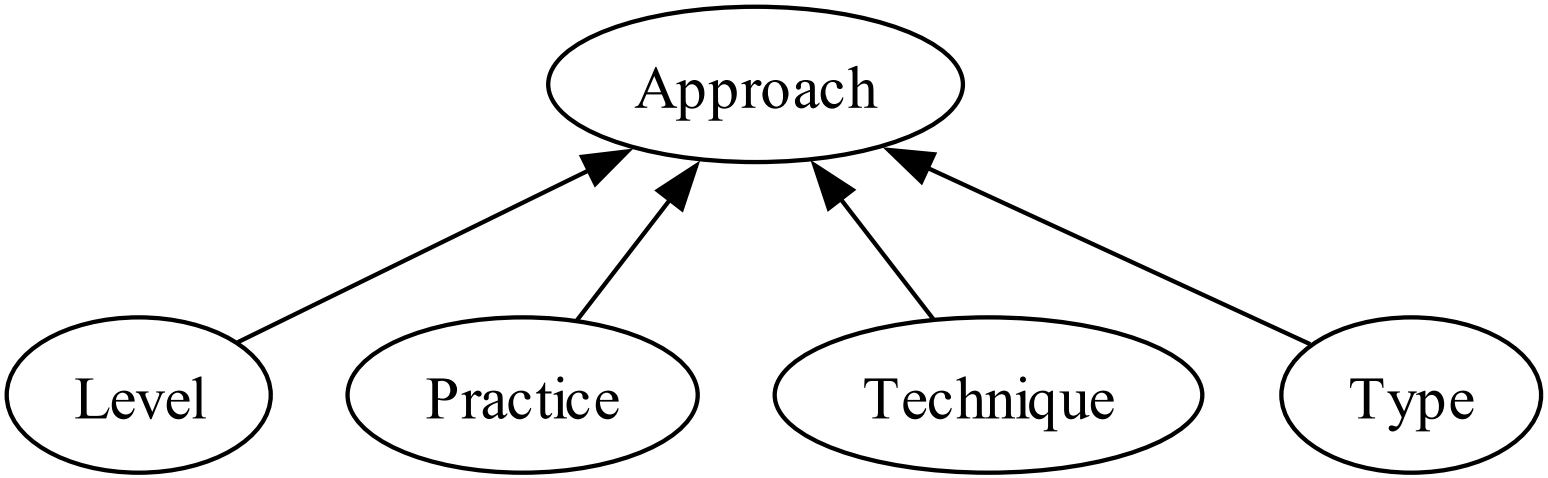
Technique: a "procedure used to create or select a test model, identify test coverage items, and derive corresponding test cases" (2022, p. 11; 2021a, p. 5; similar in 2017, p. 467)

Methodology

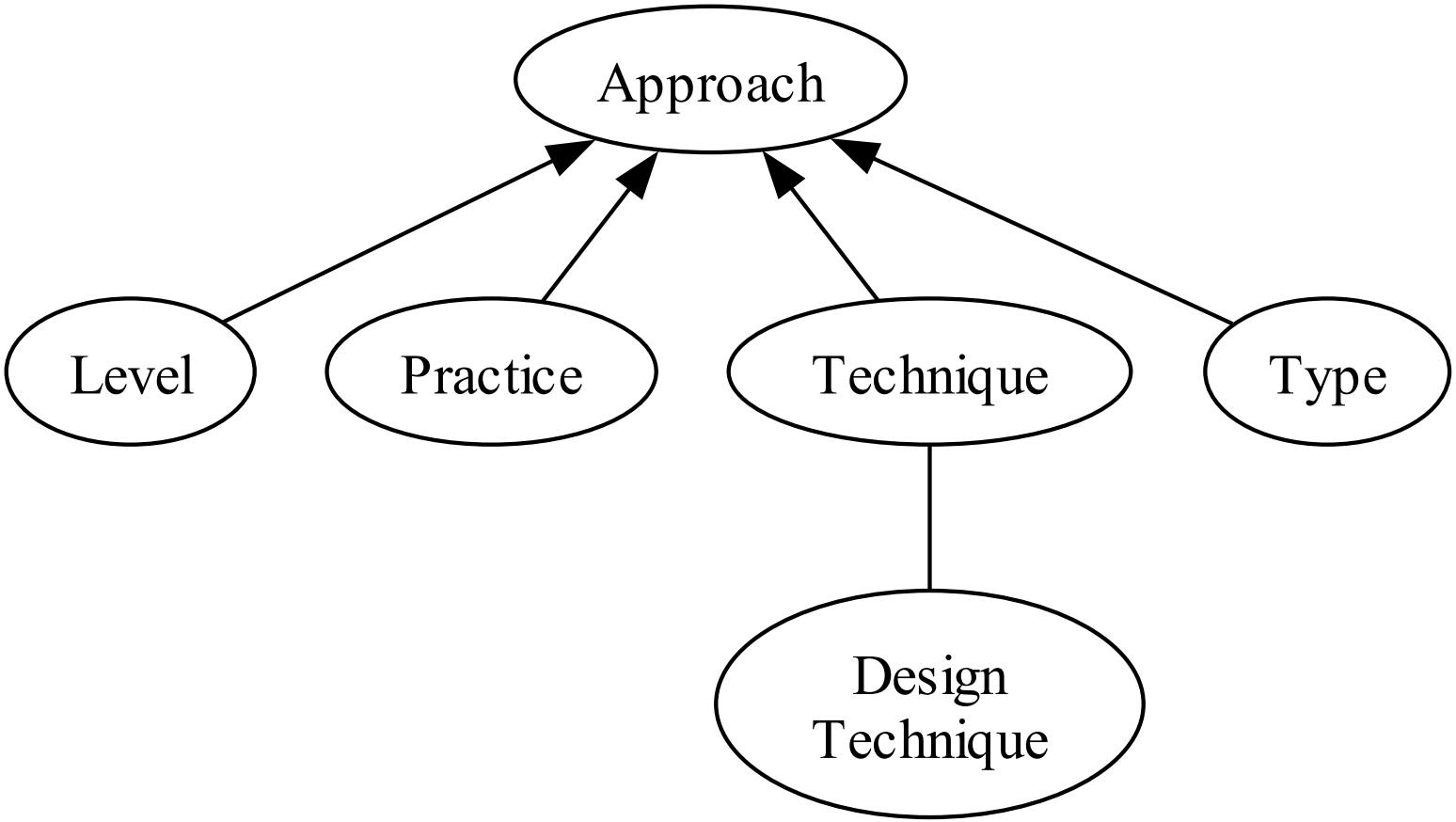
Categories



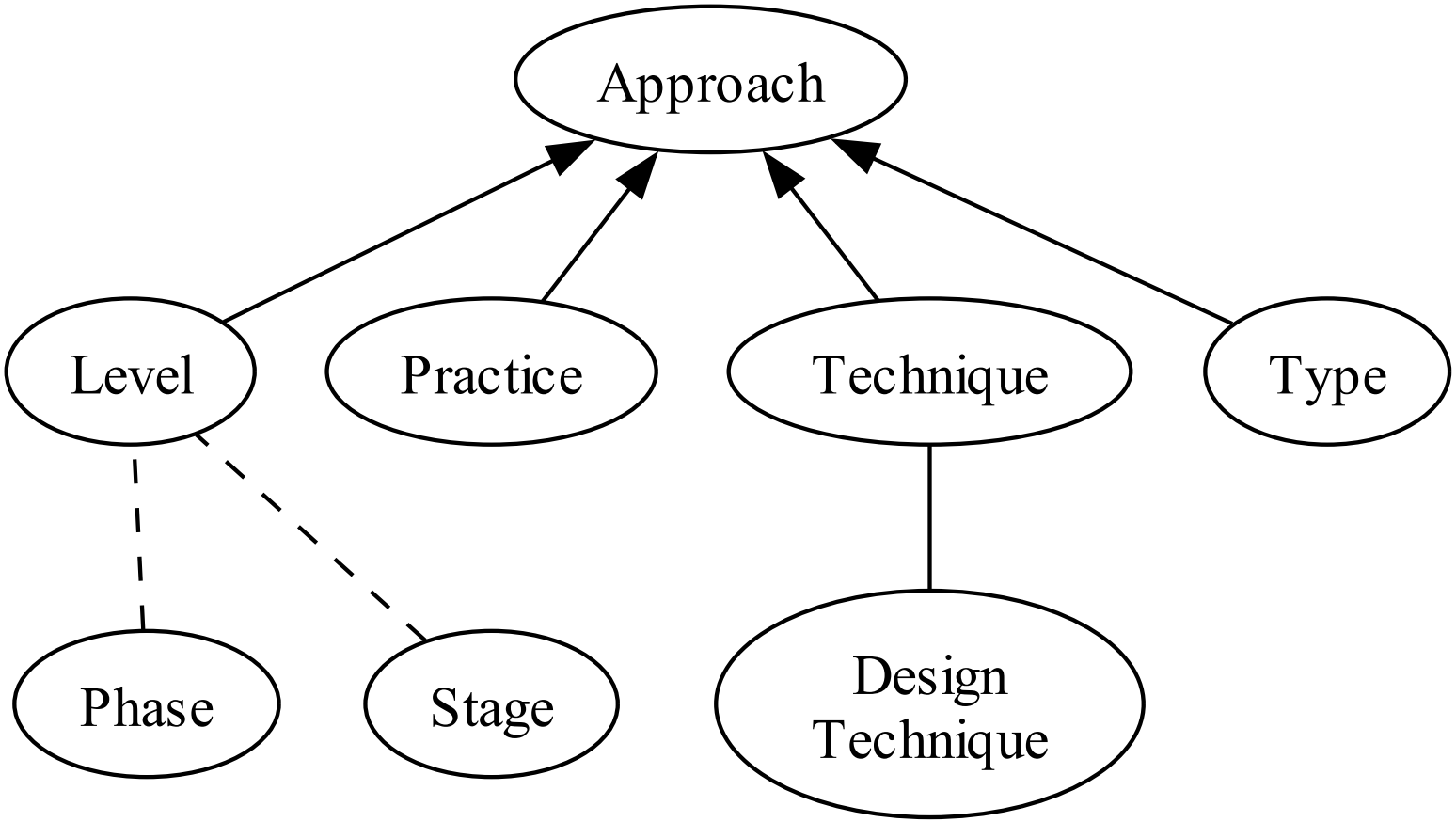
Type: "Testing that is focused on specific quality characteristics" (ISO/IEC and IEEE, 2022, p. 15; 2021c, p. 7; 2017, p. 473)



Arrows point from a *child* node to a *parent* node.



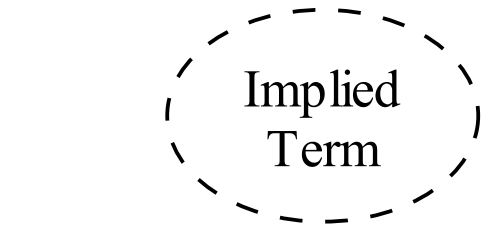
Lines without arrowheads connect *synonyms*.



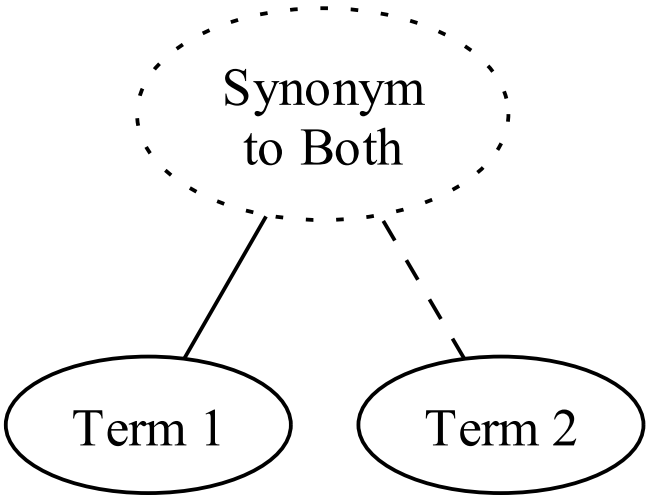
Dashed lines indicate a relationship is *implicit*.

Methodology

Visualization Notation



Dashed outlines indicate a term is *implicit*.



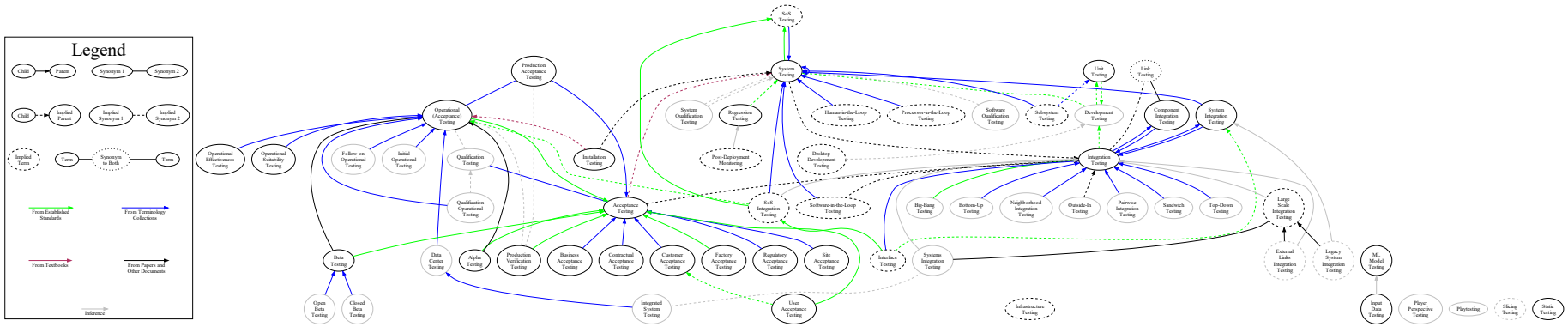
Dotted outlines indicate a term is a *synonym* to more than one term.

Graph of Test Approaches

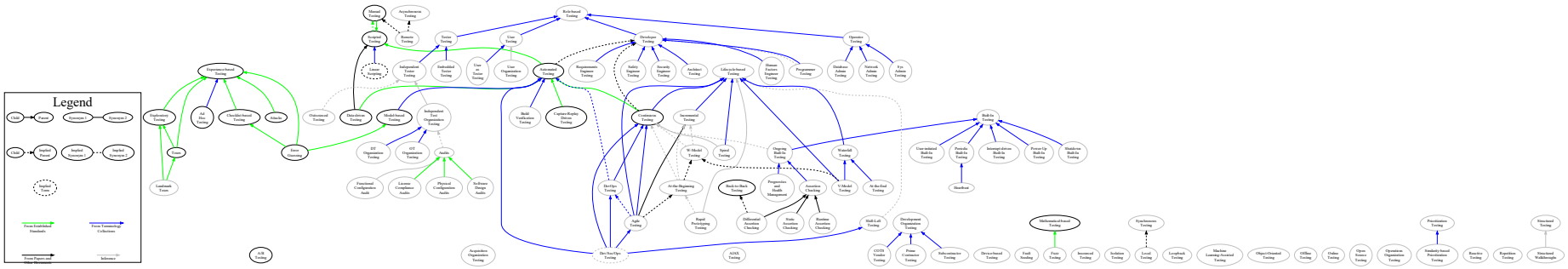
Graph of Test Approaches

```
! Dimension too large.
```

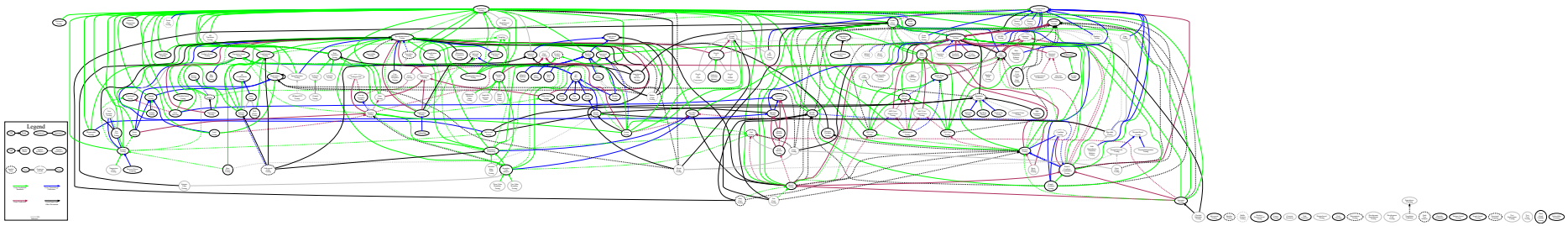
Graph of Test Levels



Graph of Test Practices



Graph of Test Techniques



Graph of Test Types

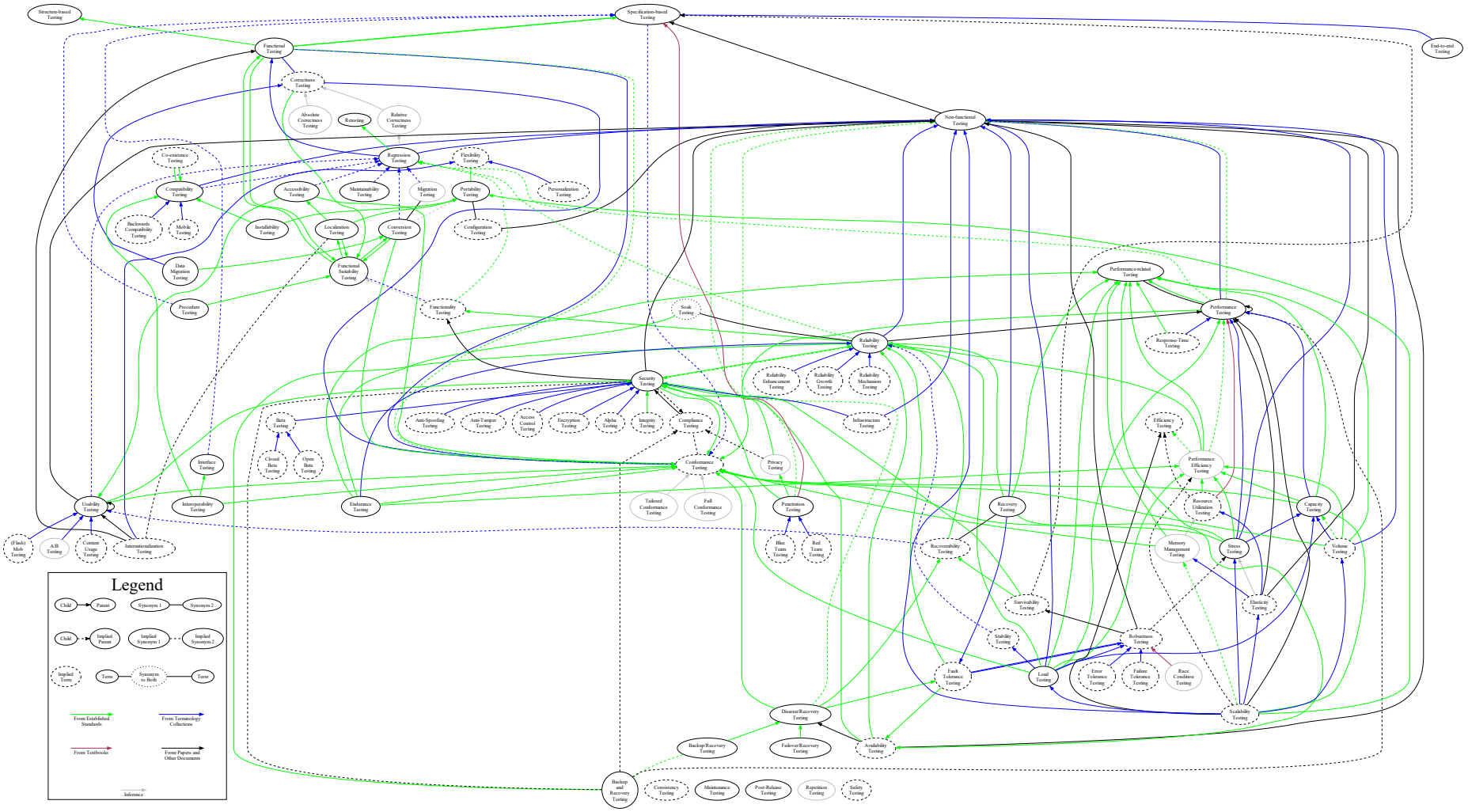
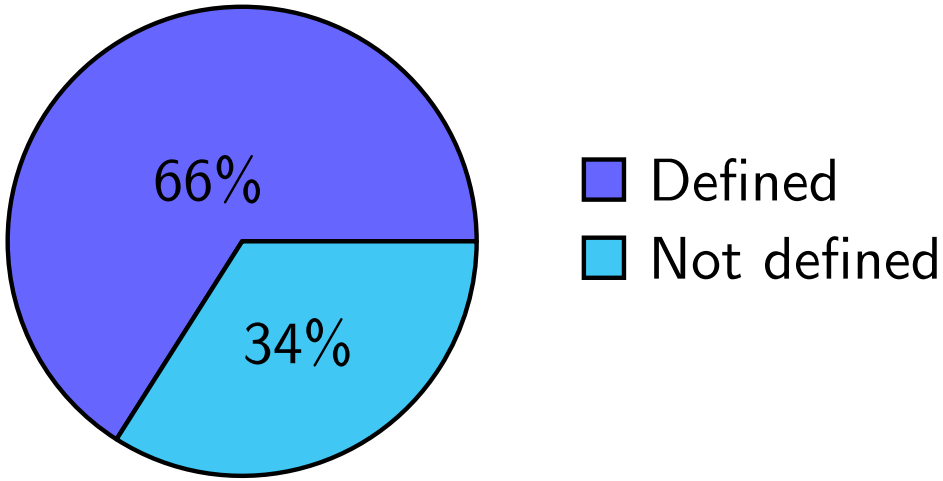


Table of Contents

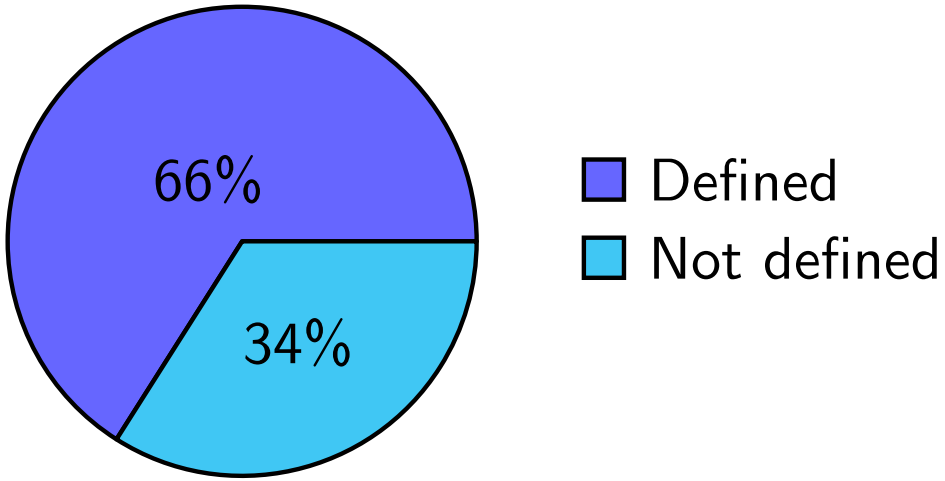
Overview

- 561 test approaches $\rightarrow$



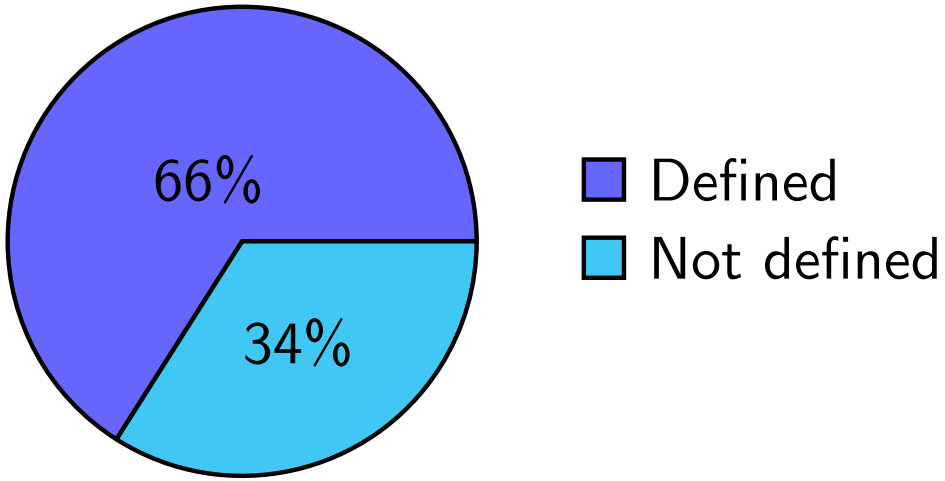
Overview

- 561 test approaches $\rightarrow$
- 77 software qualities
  (may imply test approaches)

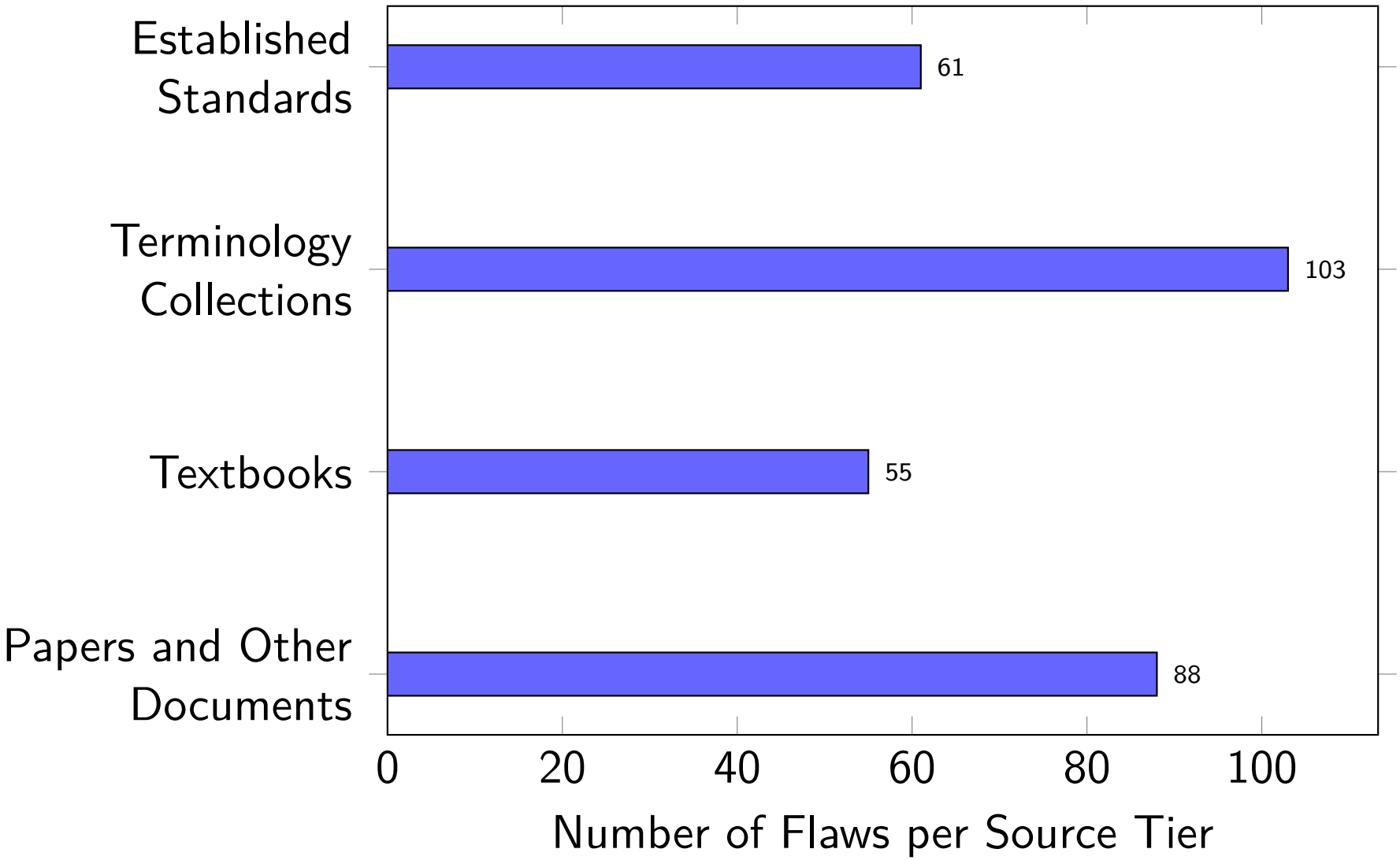


Overview

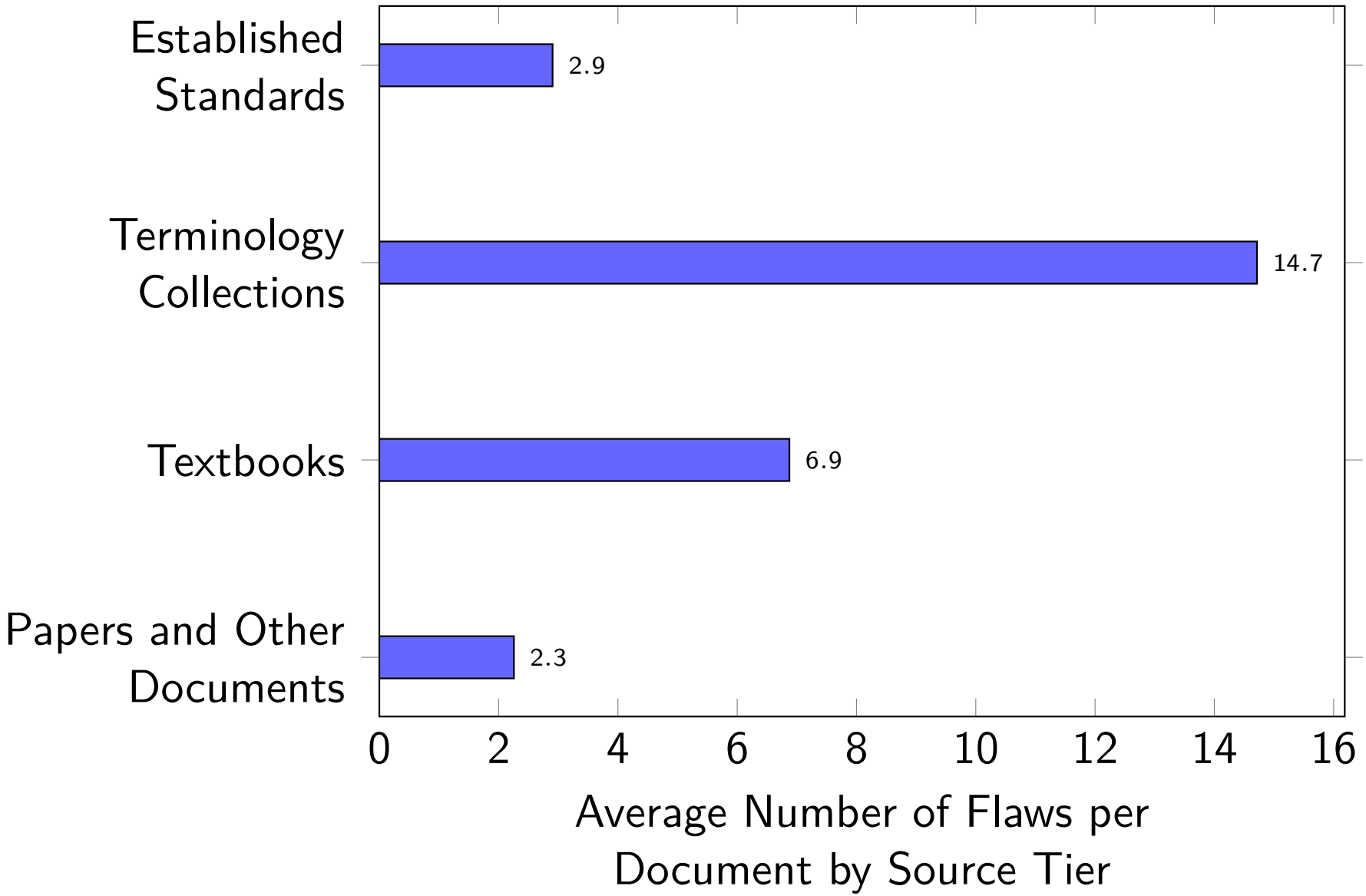
- 561 test approaches $\rightarrow$
- 77 software qualities (may imply test approaches)
- 307 flaws in the software testing literature



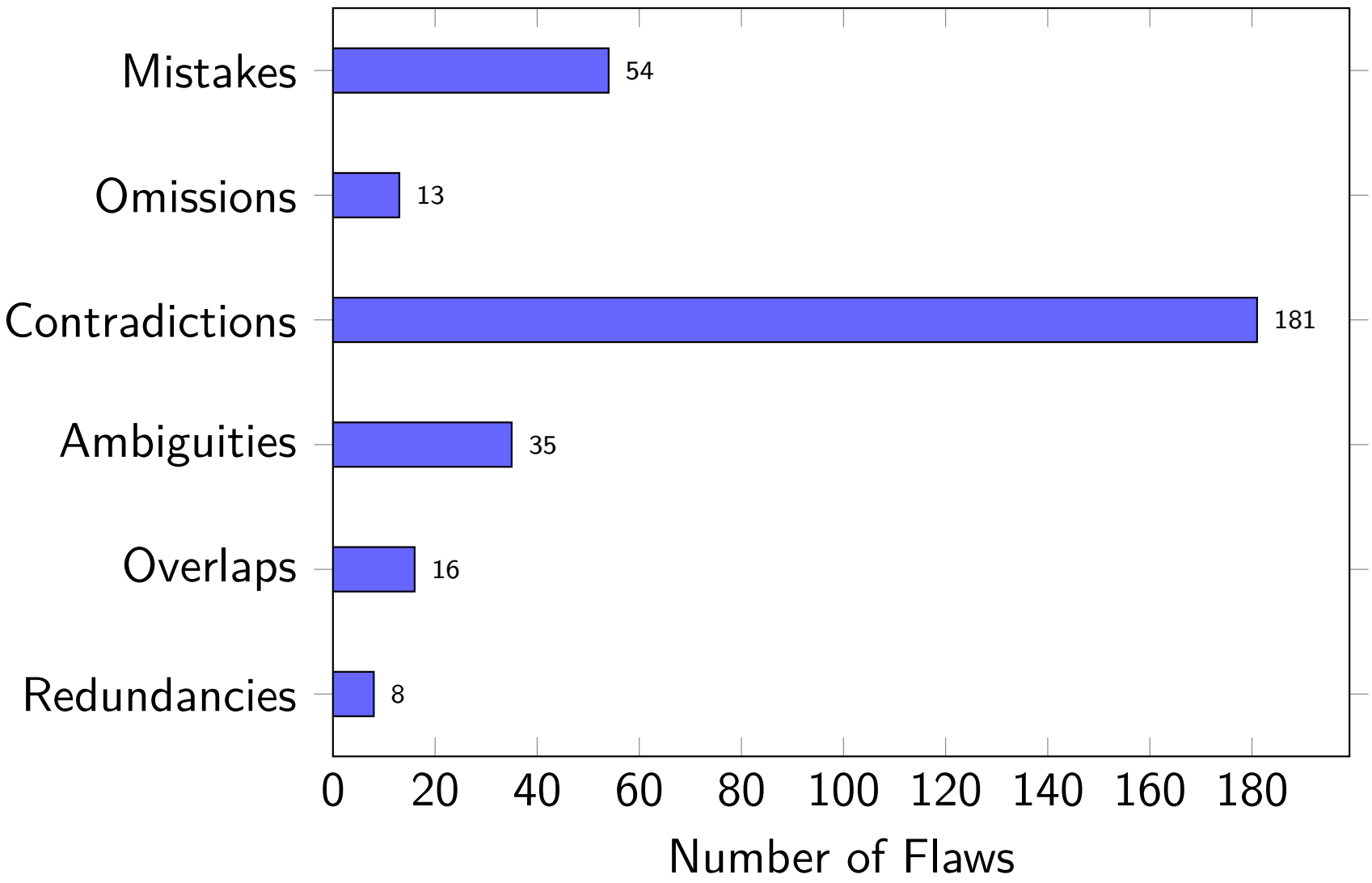
Flaw Summary by Source Tier



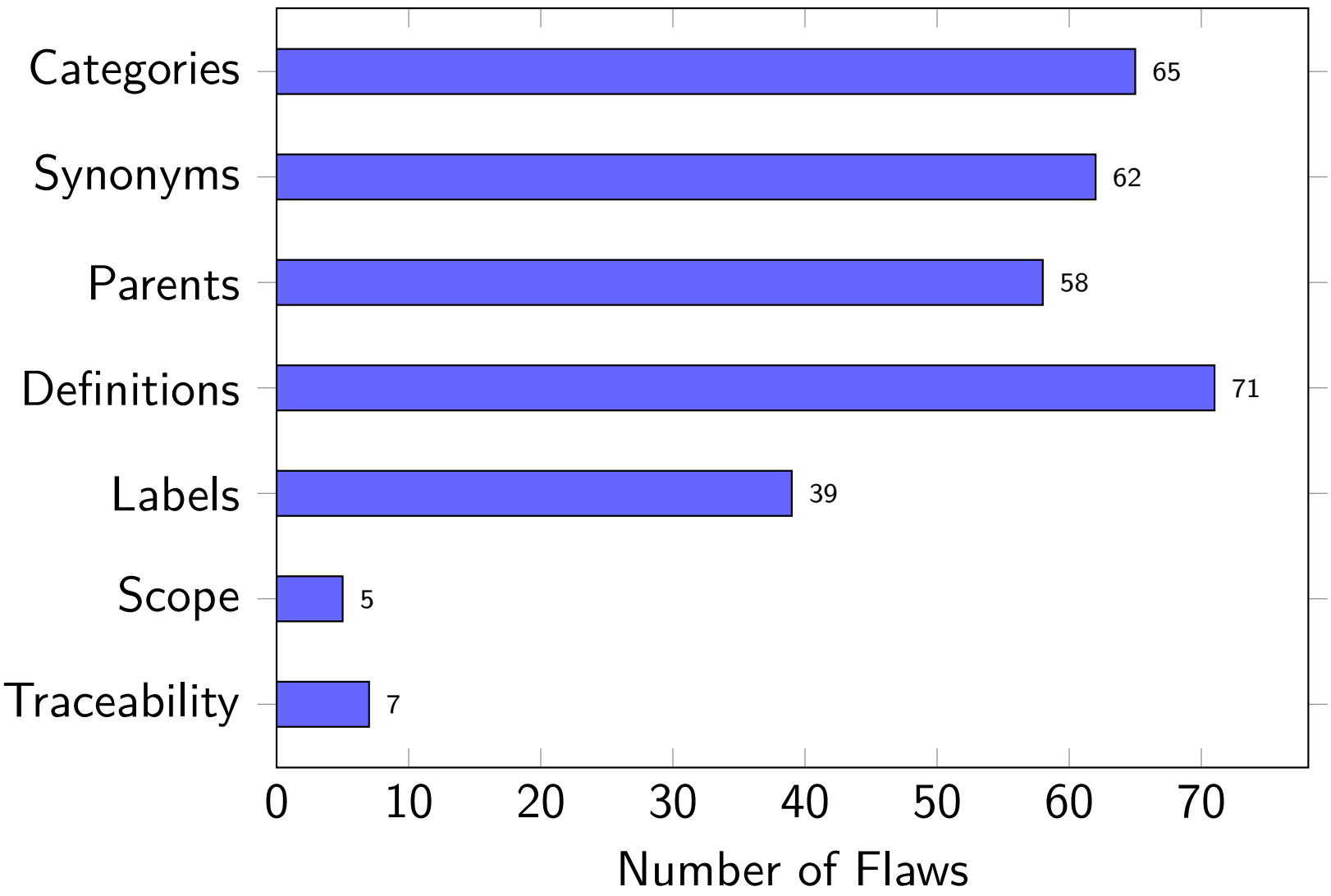
Normalized Flaw Summary



Flaw Summary by Manifestation



Flaw Summary by Domain



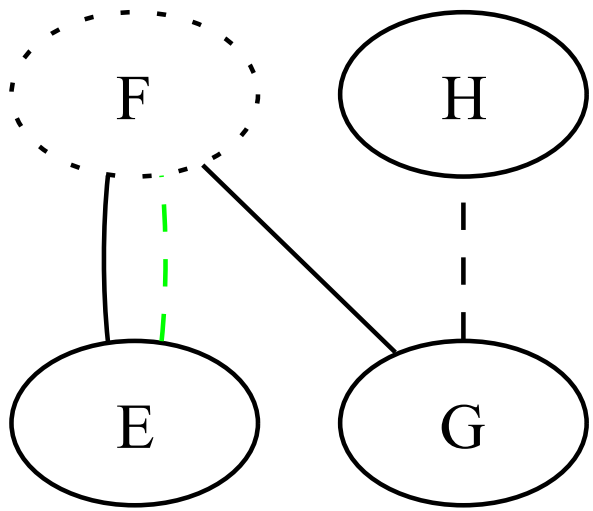
Automated Flaws

- Some terms are given as a synonym to two (or more) disjoint, unrelated terms, making the relation between the given synonyms ambiguous

Automated Flaws

- Some terms are given as a synonym to two (or more) disjoint, unrelated terms, making the relation between the given synonyms ambiguous
- These are included in generated visualizations automatically

Name	Synonym(s)
E	F (Author, 2022; implied by StdAuthor, 2021)
G	F (Author, 2017), H (implied by 2022)
H	X (StdAuthor, 2021)



Automated Flaws

Prominent examples of these “multi-synonyms”:

1. **Soak Testing:**
   - Endurance Testing
   - Reliability Testing

Source(s)

(ISO/IEC and IEEE, 2021c, p. 39)

(Gerrard, 2000a, Tab. 2; 2000b, Tab. 1, p. 26)

Automated Flaws

Prominent examples of these "multi-synonyms":

	Source(s)
1. Soak Testing:	
• Endurance Testing	(ISO/IEC and IEEE, 2021c, p. 39)
• Reliability Testing	(Gerrard, 2000a, Tab. 2; 2000b, Tab. 1, p. 26)
2. Functional Testing:	
• Behavioural Testing	(Kam, 2008, p. 45)
• Correctness Testing	(Washizaki, 2024, p. 5-7)
• Specification-based Testing	(ISO/IEC and IEEE, 2017, p. 196; ...)

Automated Flaws

Prominent examples of these "multi-synonyms":

	Source(s)
1 Soak Testing:	
• Endurance Testing	(ISO/IEC and IEEE, 2021c, p. 39)
• Reliability Testing	(Gerrard, 2000a, Tab. 2; 2000b, Tab. 1, p. 26)
2 Functional Testing:	
• Behavioural Testing	(Kam, 2008, p. 45)
• Correctness Testing	(Washizaki, 2024, p. 5-7)
• Specification-based Testing	(ISO/IEC and IEEE, 2017, p. 196; ...)
3 Link Testing:	
• Branch Testing	(implied by ISO/IEC and IEEE, 2021c, p. 24)
• Component Integration Testing	(Kam, 2008, p. 45)
• Integration Testing	(implied by Gerrard, 2000a, p. 13)

Acknowledgment

- Dr. Spencer Smith and Dr. Jacques Carette have been great supervisors and valuable sources of guidance and feedback
  - They have helped me refine the scope of this project
  - Dr. Smith first suggested generating test cases back in 2020!

Acknowledgment

- Dr. Spencer Smith and Dr. Jacques Carette have been great supervisors and valuable sources of guidance and feedback
  - They have helped me refine the scope of this project
  - Dr. Smith first suggested generating test cases 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. Spencer Smith and Dr. Jacques Carette have been great supervisors and valuable sources of guidance and feedback
  - They have helped me refine the scope of this project
  - Dr. Smith first suggested generating test cases 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

Donald G. Firesmith. A Taxonomy of Testing Types, 2015. URL https://apps.dtic.mil/sti/pdfs/AD1147163.pdf.

Paul Gerrard. Risk-based E-business Testing - Part 1: Risks and Test Strategy. Technical report, Systeme Evolutif, London, UK, 2000a. URL https://www.agileconnection.com/sites/default/files/article/file/2013/XUS129342file1_0.pdf.

Paul Gerrard. Risk-based E-business Testing - Part 2: Test Techniques and Tools. Technical report, Systeme Evolutif, London, UK, 2000b. URL wenku.uml.com.cn/document/test/EBTestingPart2.pdf.

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.

References II

ISO/IEC and IEEE. ISO/IEC/IEEE International Standard - Software and systems engineering –Software testing –Part 2: Test processes. *ISO/IEC/IEEE 29119-2:2021(E)*, October 2021a. doi: 10.1109/IEEESTD.2021.9591508.

ISO/IEC and IEEE. ISO/IEC/IEEE International Standard - Software and systems engineering –Software testing –Part 4: Test techniques. *ISO/IEC/IEEE 29119-4:2021(E)*, October 2021c. doi: 10.1109/IEEESTD.2021.9591574.

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.

Ben Kam. Web Applications Testing. Technical Report 2008-550, Queen's University, Kingston, ON, Canada, October 2008. URL `https://research.cs.queensu.ca/TechReports/Reports/2008-550.pdf`.

References III

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.

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.

References IV

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].

Hans van Vliet. *Software Engineering: Principles and Practice*. John Wiley & Sons, Ltd., Chichester, England, 2nd edition, 2000. ISBN 0-471-97508-7.

Hironori Washizaki, editor. *Guide to the Software Engineering Body of Knowledge, Version 4.0*. January 2024.