Putting Software Testing Terminology to the Test

M.A.Sc. Seminar

Samuel Crawford, B.Eng.

McMaster University
Department of Computing and Software

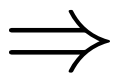
Fall 2024

Table of Contents

Table of Contents

The Need for Standardized Terminology

- Engineering is applied science
- Scientific fields use precise terminology

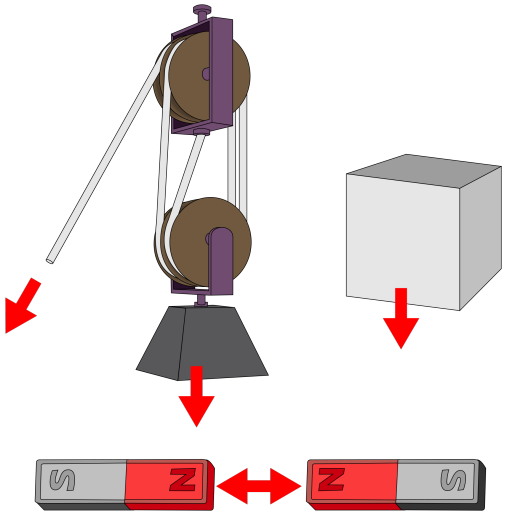


SOFTWARE ENGINEERING

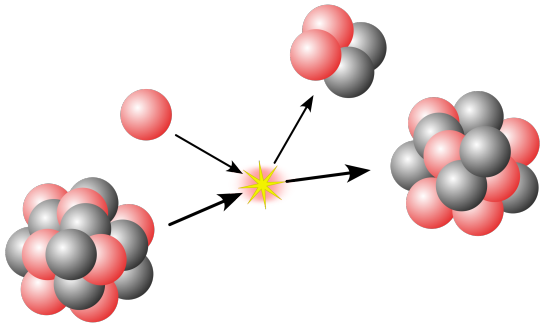
The Need for Standardized Terminology

- Engineering is applied science
- Scientific fields use precise terminology

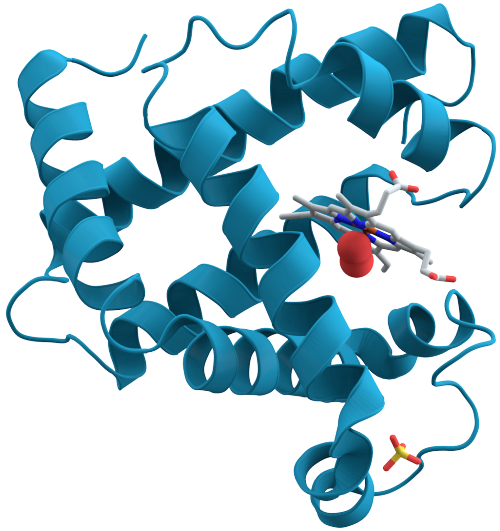
$\Rightarrow$ SOFTWARE ENGINEERING



Penubag and Ramey (2010)



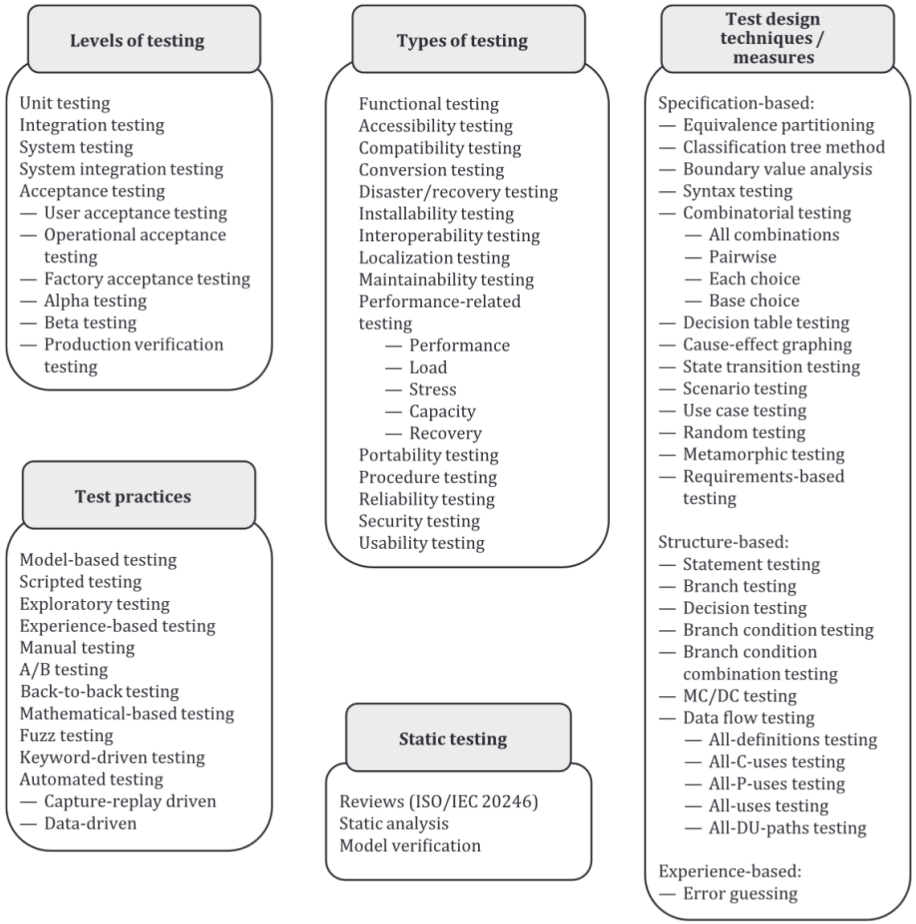
Kjerish (2016)



AzaToth (2008)

The Lack of Standardized Terminology

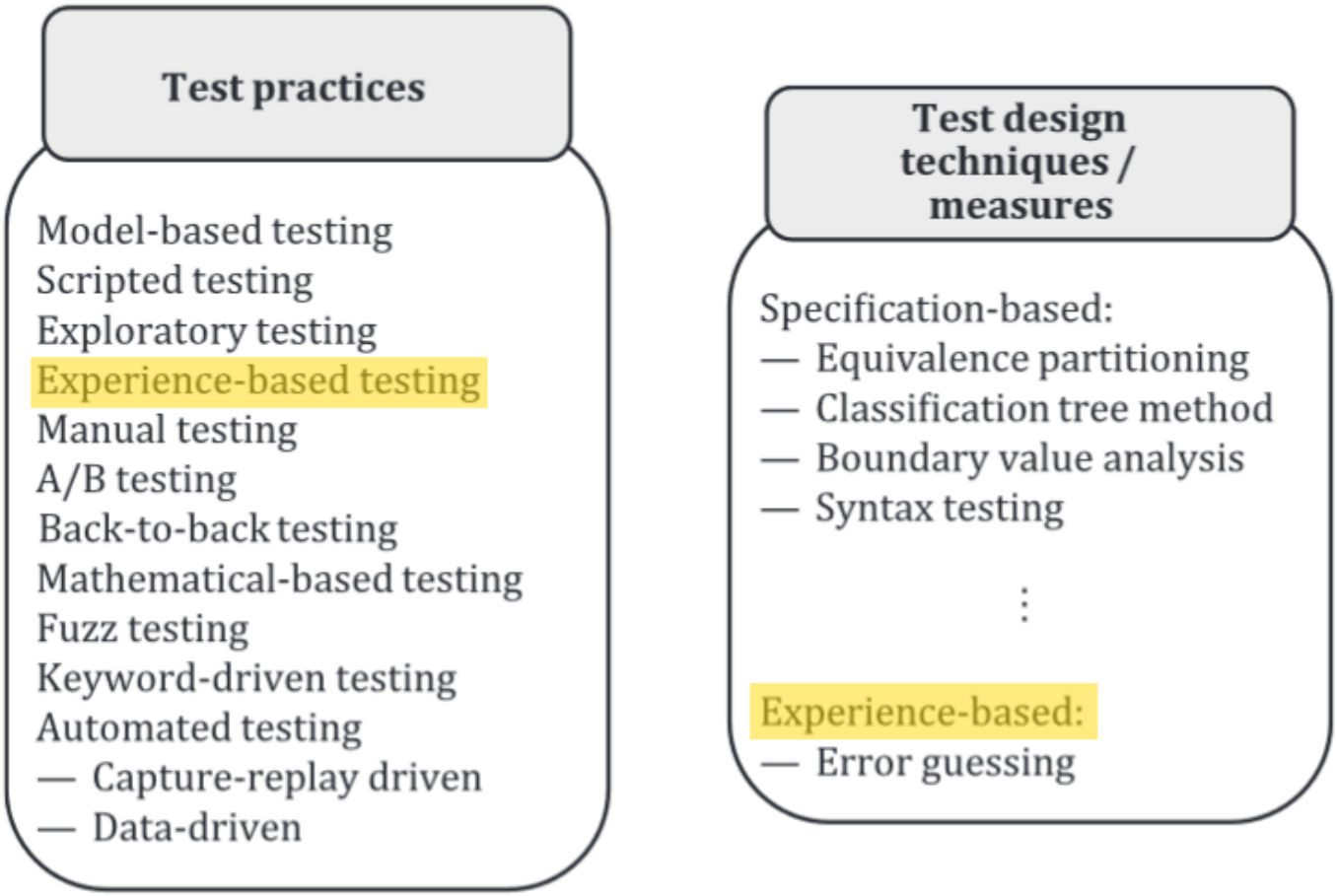
"The Problem"



(ISO/IEC and IEEE, 2022, Fig. 2)

The Lack of Standardized Terminology

"The Problem"



Adapted from (ISO/IEC and IEEE, 2022, Fig. 2)

The Lack of Standardized Terminology

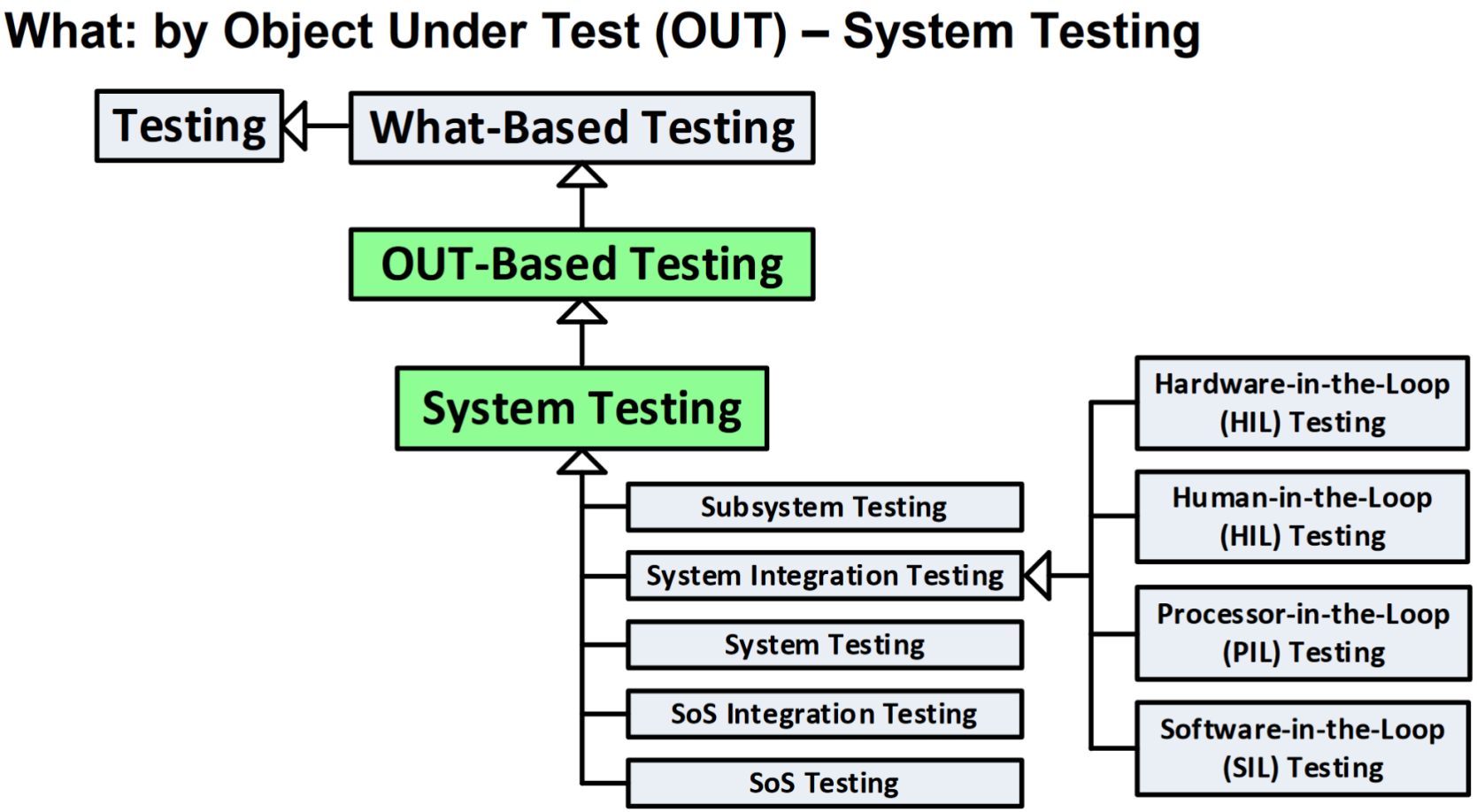
"The Problem"

ISO/IEC/IEEE 29119-4 describes the experience-based test design technique of error guessing. Other experience-based test practices include (but are not limited to) exploratory testing (see 4.4.3.3), tours, attacks, and checklist-based testing.

Adapted from (ISO/IEC and IEEE, 2022, p. 34)

The Lack of Standardized Terminology

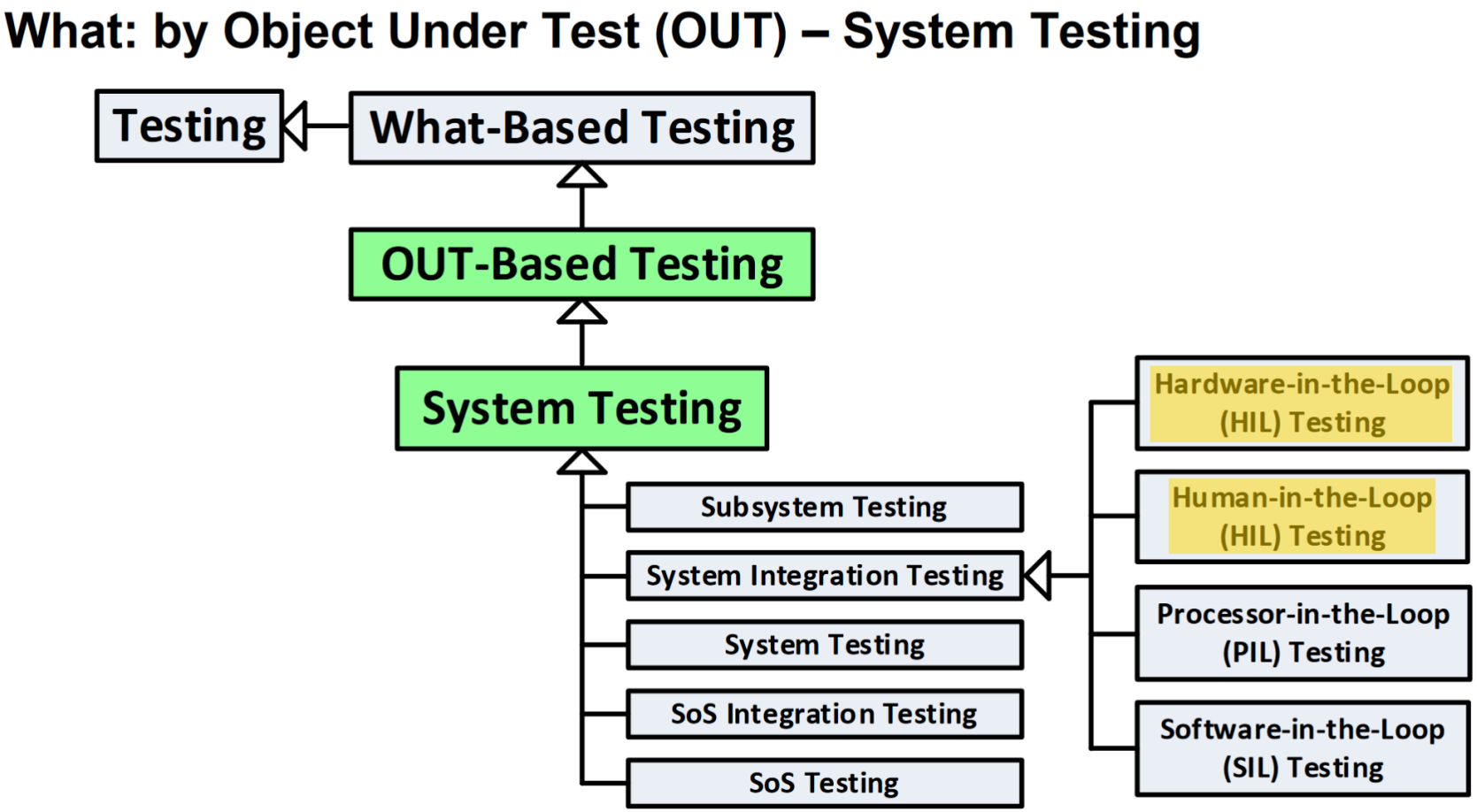
"The Problem" (cont.)



(Firesmith, 2015, p. 23)

The Lack of Standardized Terminology

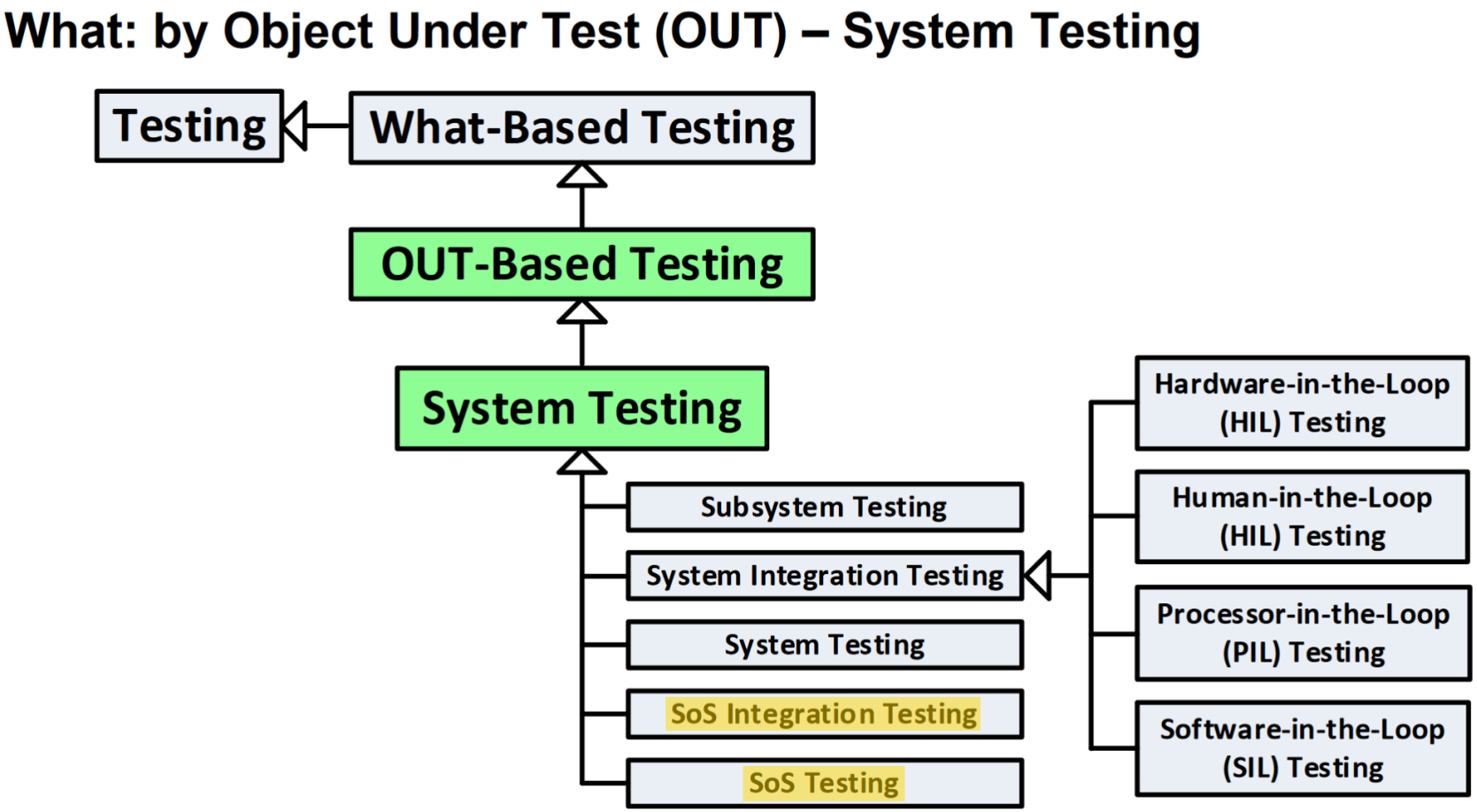
"The Problem" (cont.)



Adapted from (Firesmith, 2015, p. 23)

The Lack of Standardized Terminology

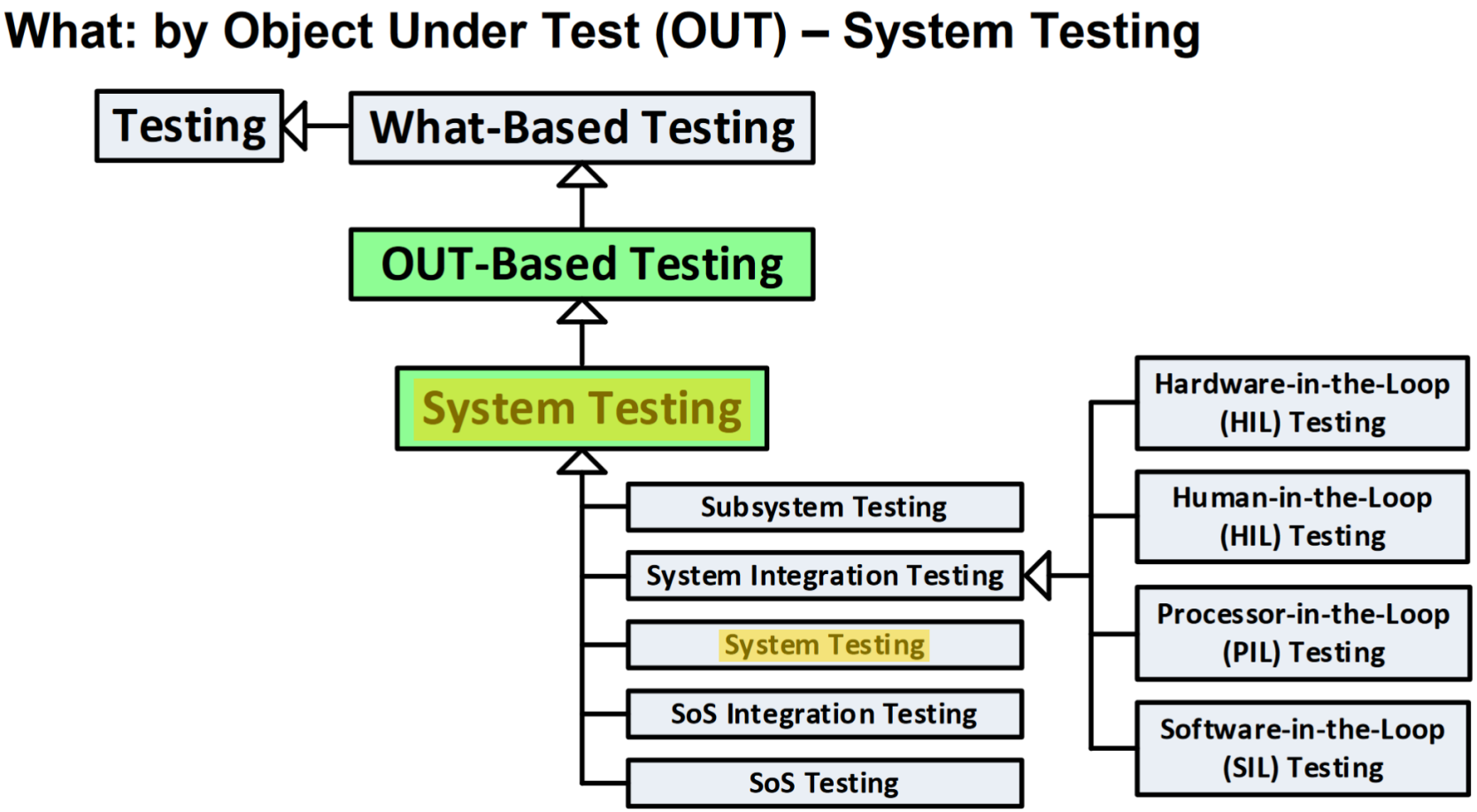
"The Problem" (cont.)



Adapted from (Firesmith, 2015, p. 23)

The Lack of Standardized Terminology

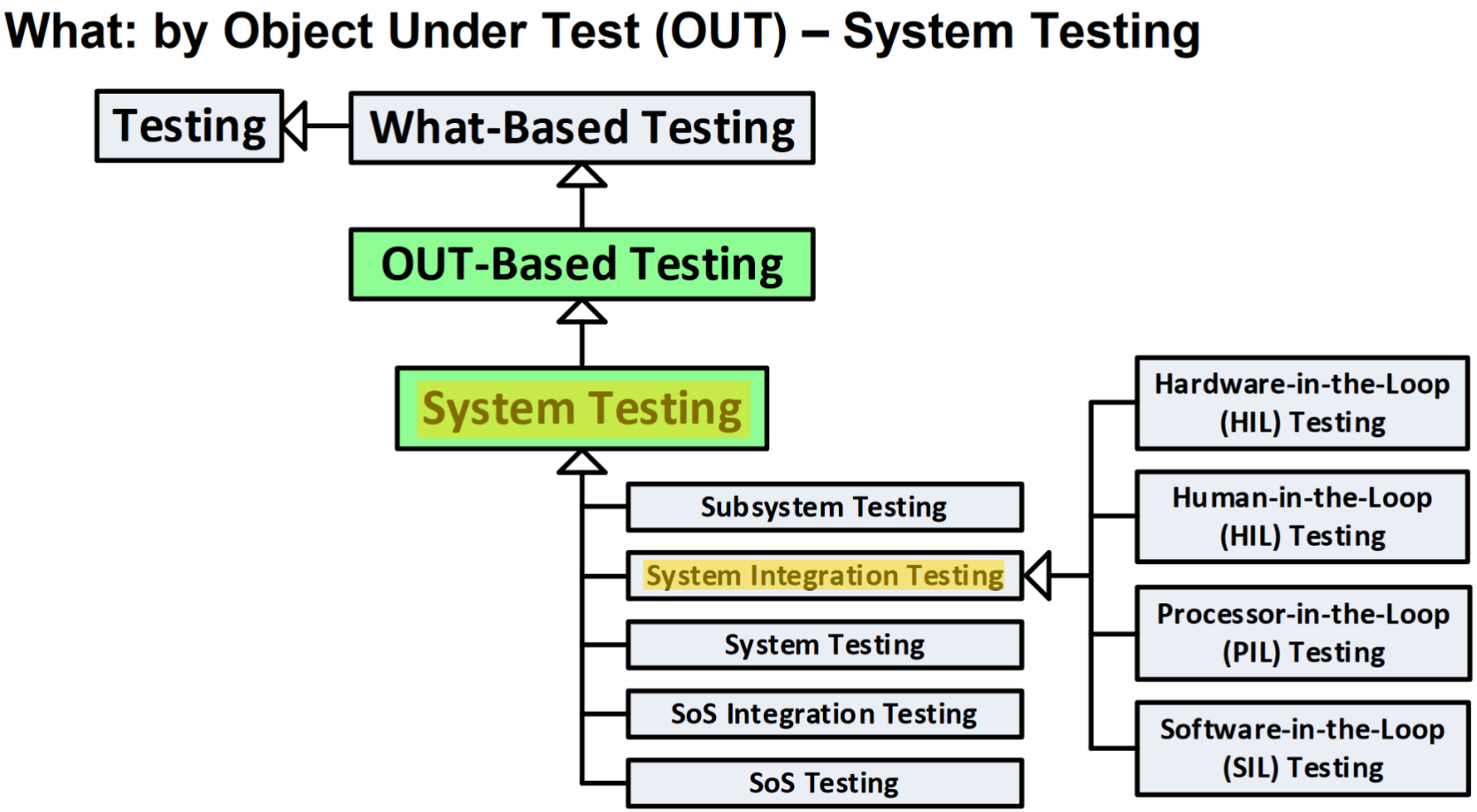
"The Problem" (cont.)



Adapted from (Firesmith, 2015, p. 23)

The Lack of Standardized Terminology

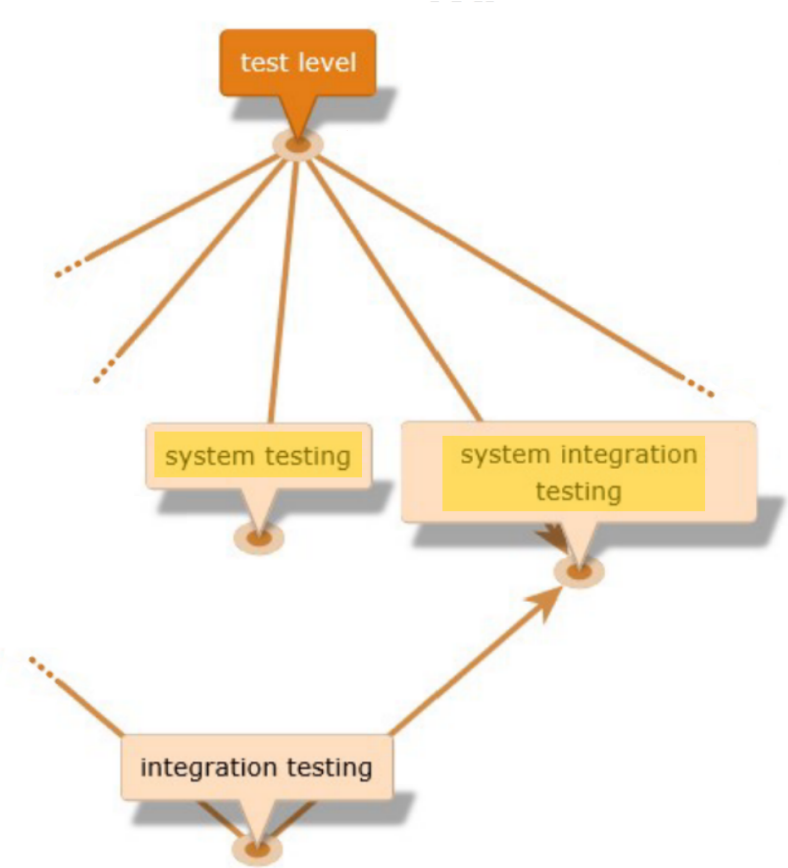
"The Problem" (cont.)



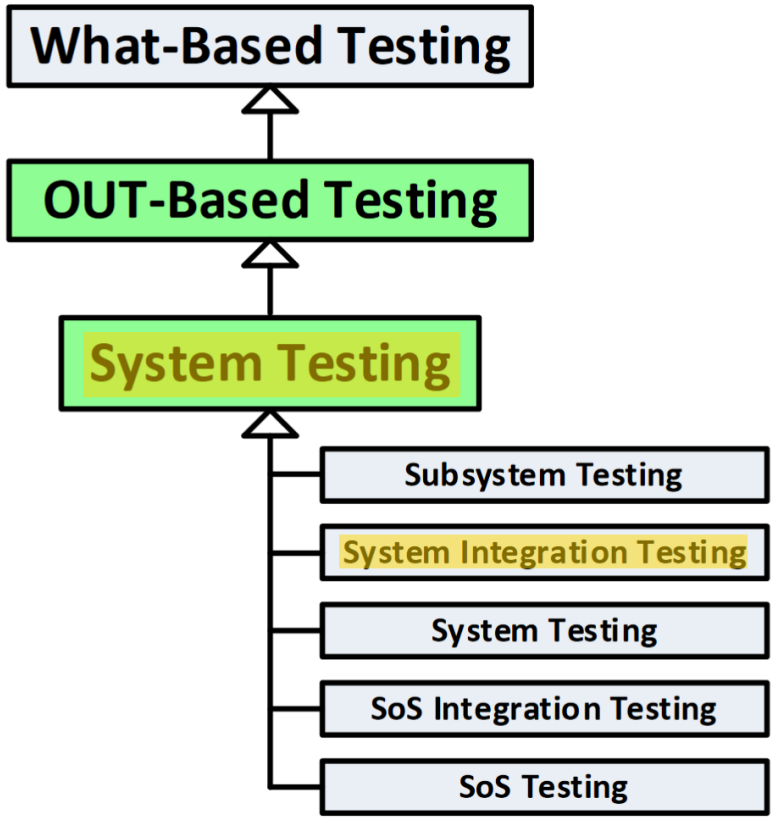
Adapted from (Firesmith, 2015, p. 23)

The Lack of Standardized Terminology

"The Problem" (cont.)



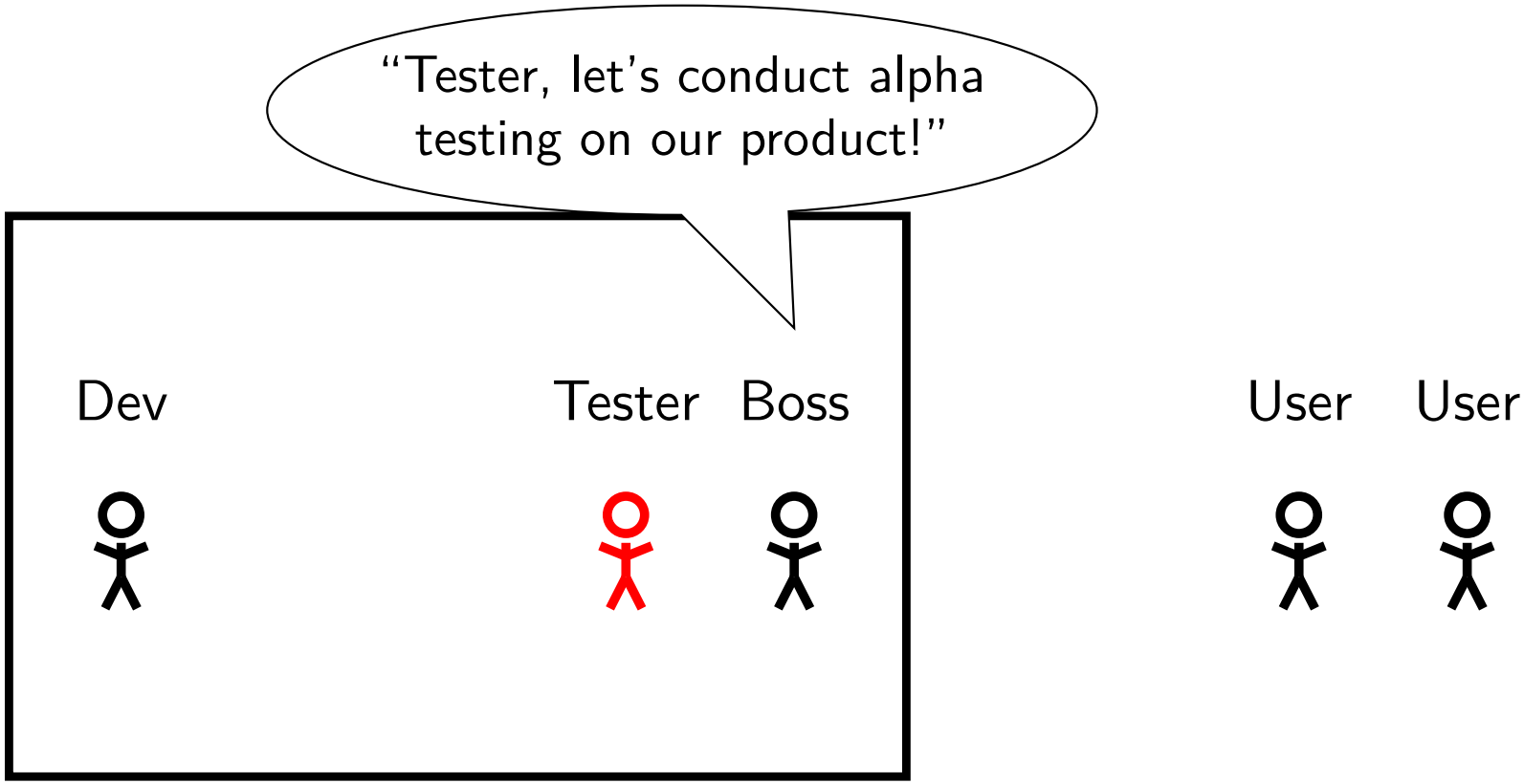
Adapted from (Hamburg and Mogyorodi, 2024)



Adapted from (Firesmith, 2015, p. 23)

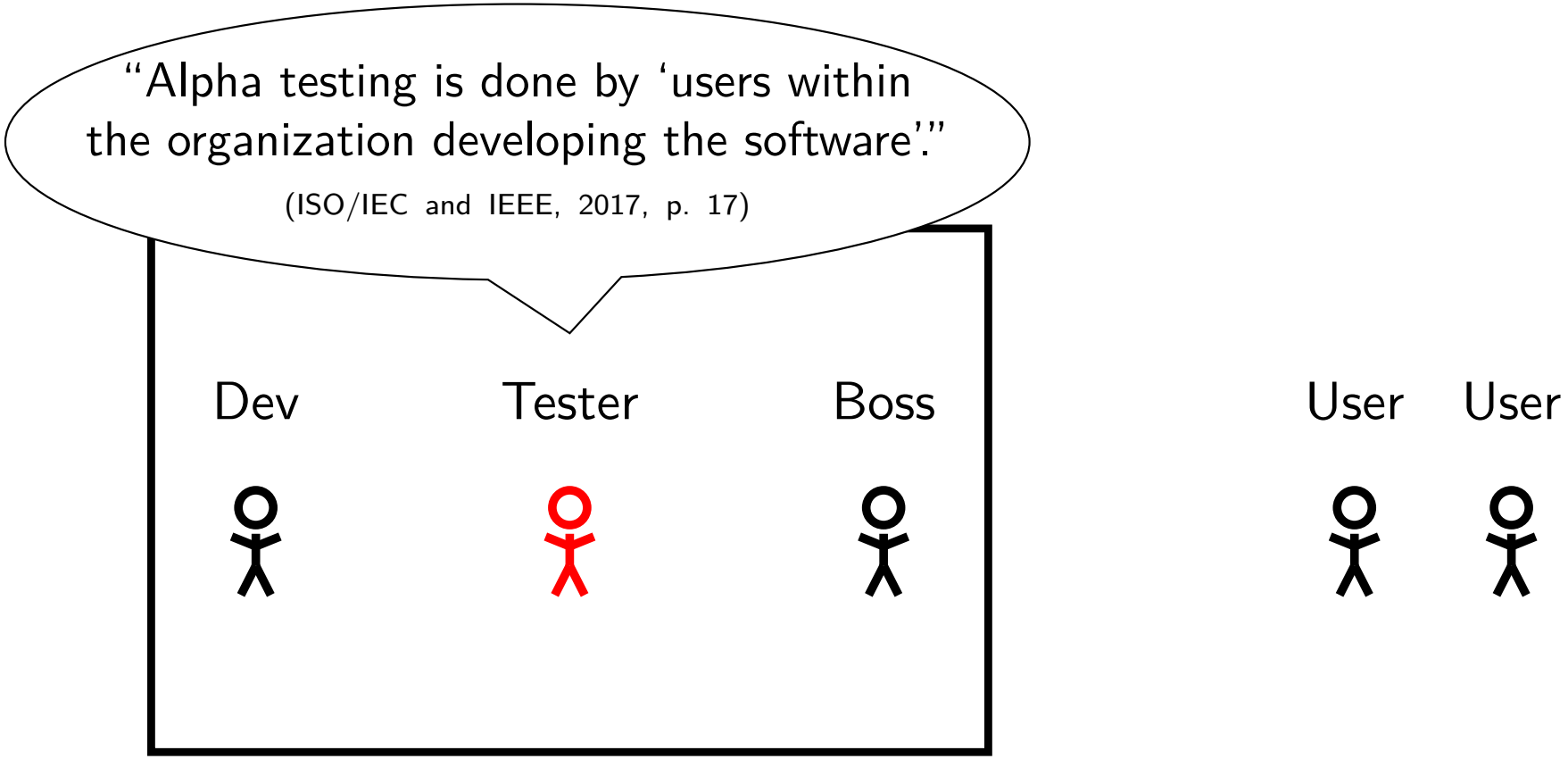
The Lack of Standardized Terminology

"The Problem" (cont.)



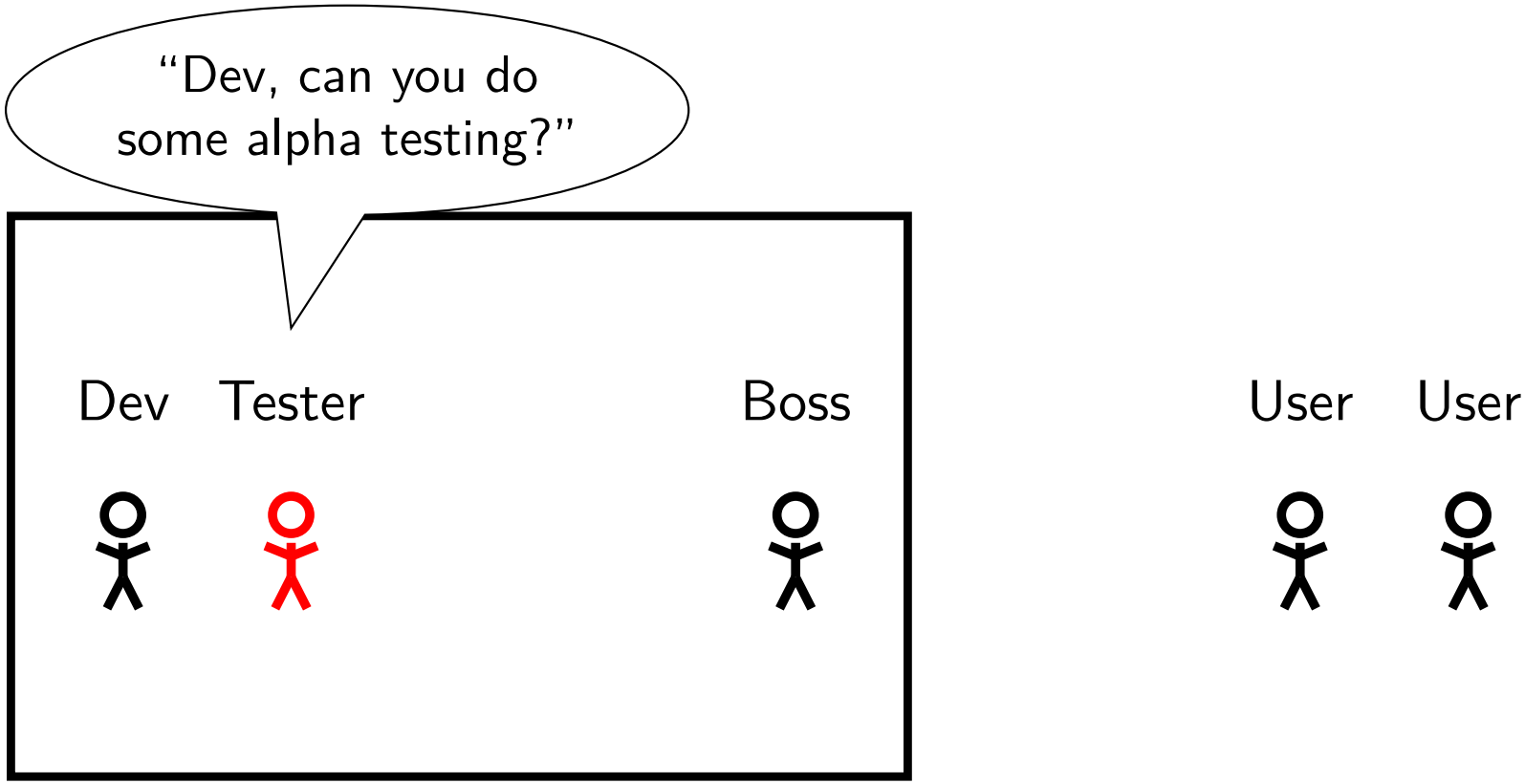
The Lack of Standardized Terminology

"The Problem" (cont.)



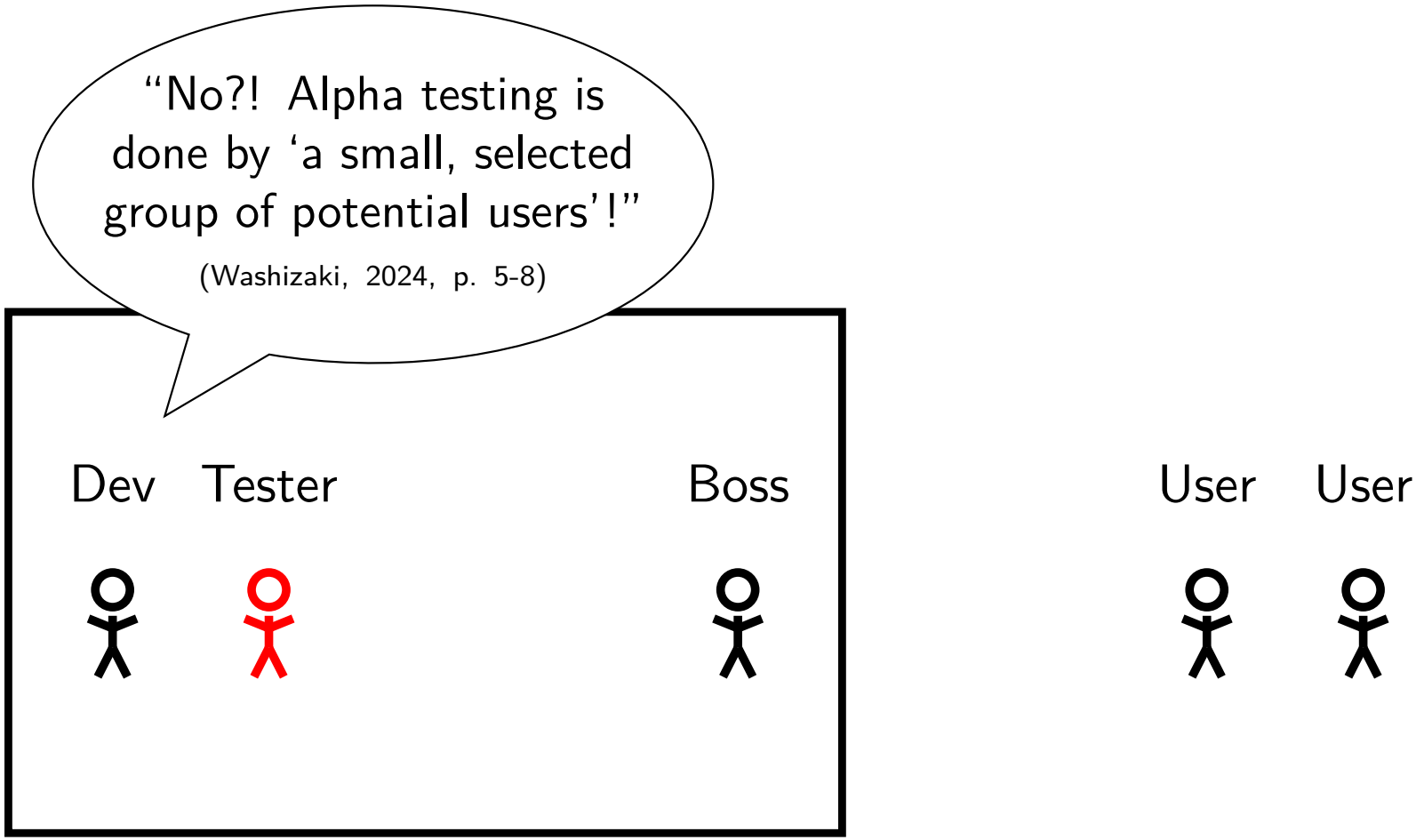
The Lack of Standardized Terminology

"The Problem" (cont.)



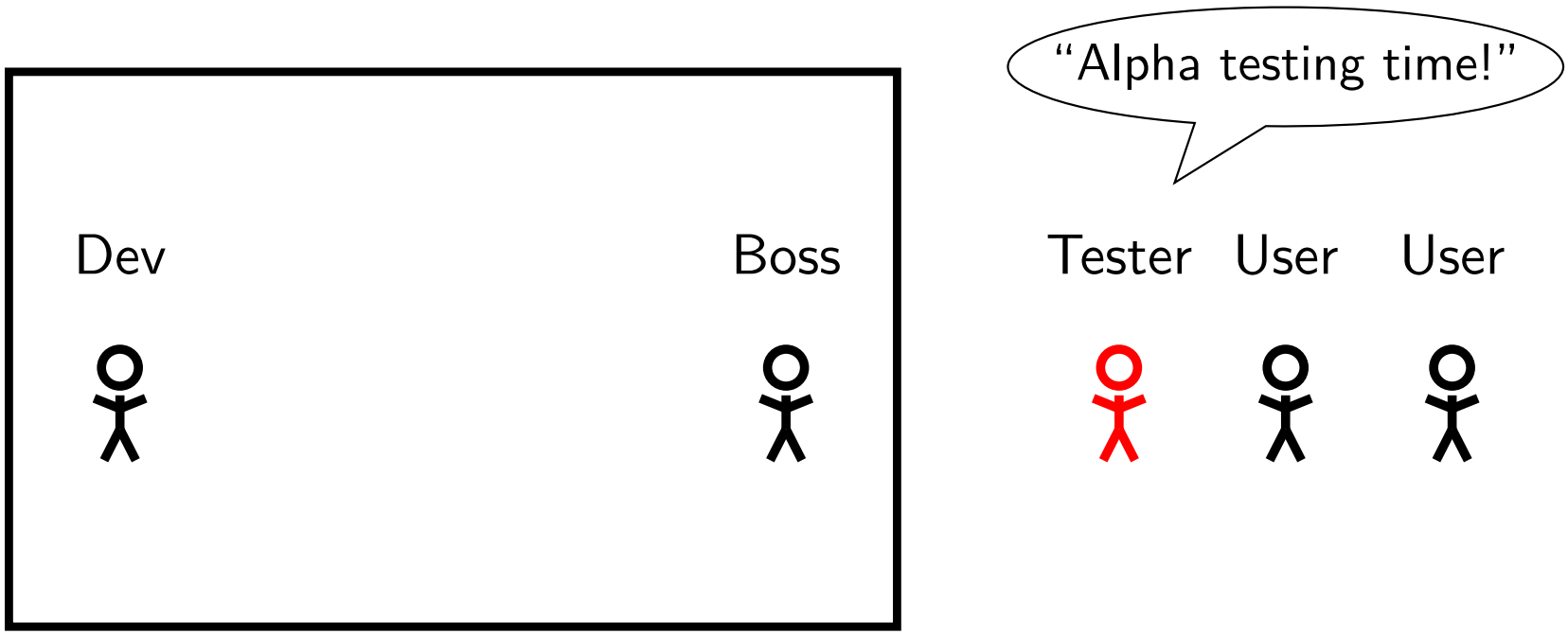
The Lack of Standardized Terminology

"The Problem" (cont.)



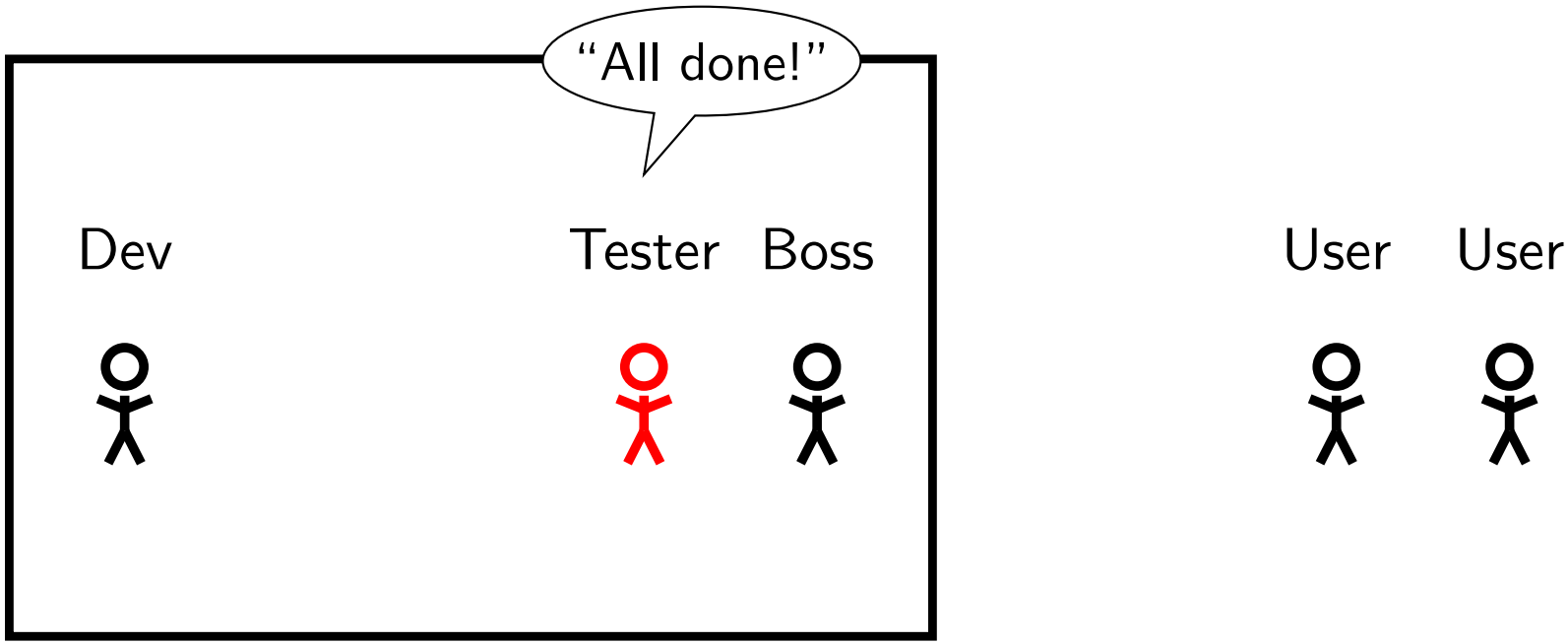
The Lack of Standardized Terminology

"The Problem" (cont.)



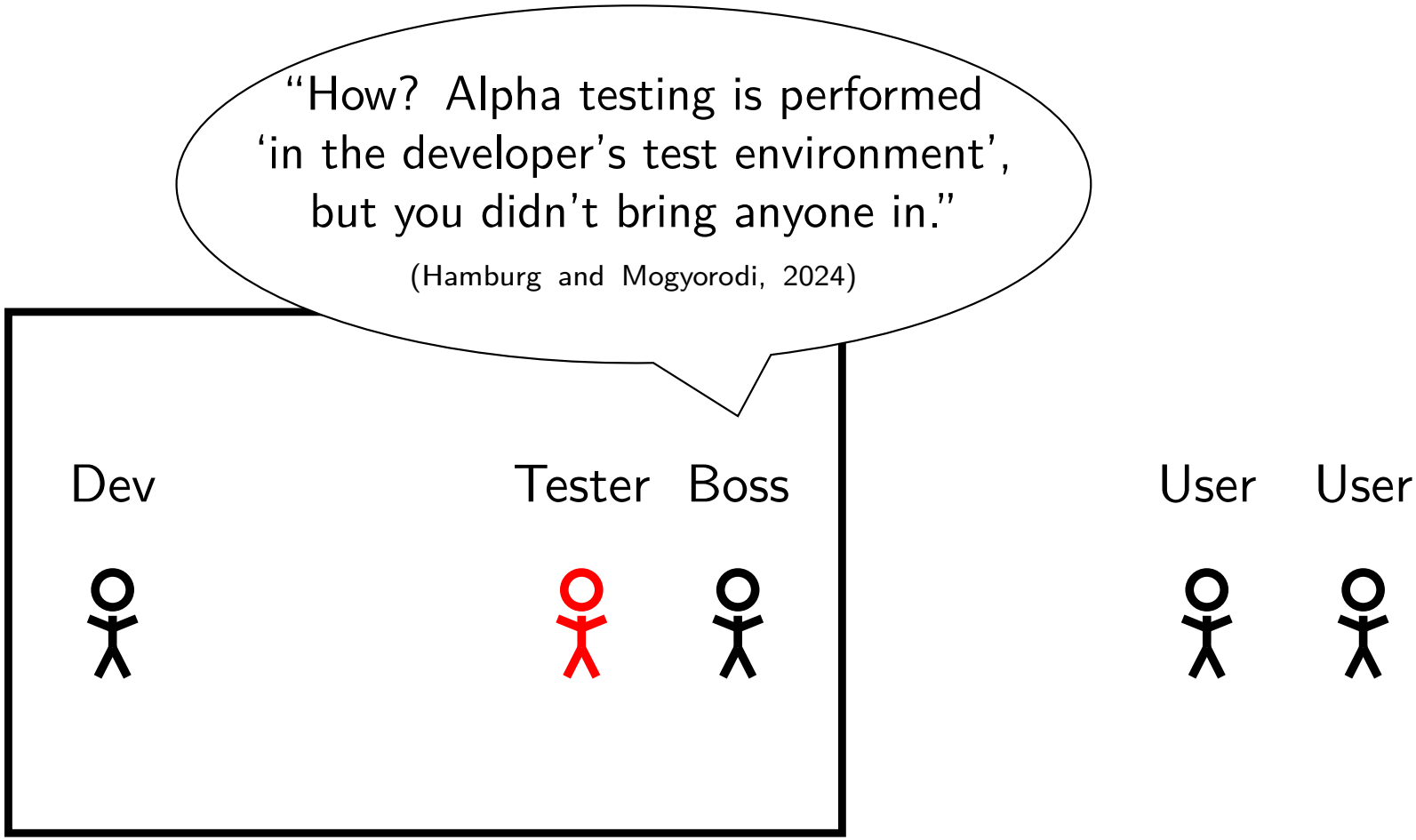
The Lack of Standardized Terminology

"The Problem" (cont.)



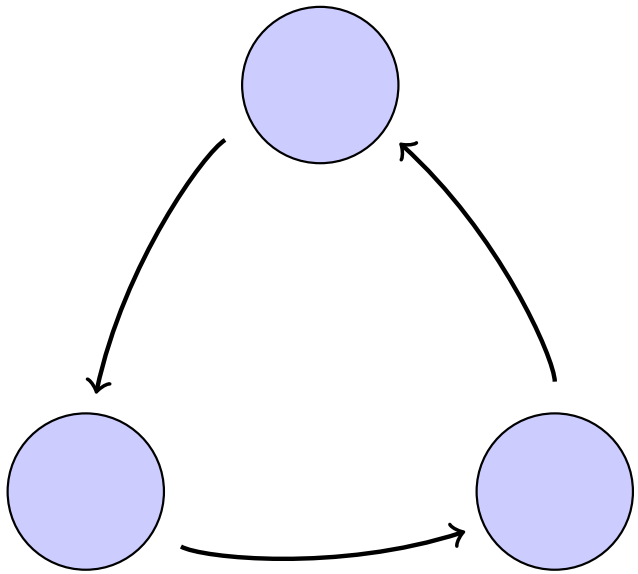
The Lack of Standardized Terminology

"The Problem" (cont.)



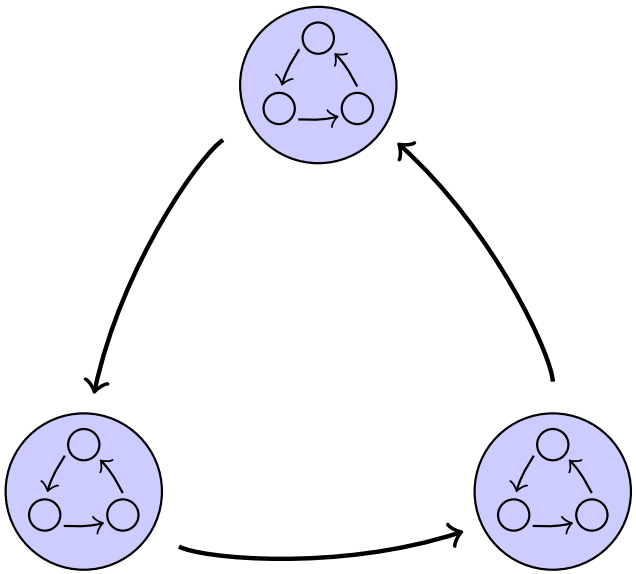
Interorganizational

Schools, companies, etc.



Interorganizational

Schools, companies, etc.



Intraorganizational

"Complete testing" could require the tester to:

- discover every bug,
- exhaust the time allocated,
- implement every planned test,
- . . . (Kaner et al., 2011, p. 7)

Taxonomies to the Rescue?

"The Problem" (cont.)

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

Taxonomies to the Rescue?

"The Problem" (cont.)

- 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

Table of Contents

Research Questions

Research Question 1

What testing approaches do the literature describe?

Research Question 2

Are these descriptions consistent?

Research Question 3

Can we systematically resolve any of these inconsistencies?

Methodology

Overview

Research Question 1

What testing approaches do the literature describe?

1. Identify authoritative sources
2. Identify software testing terminology from each source, including test approaches and terms that imply them
3. For each test approach, record its:
    1. Name
    2. Category
    3. Definition
    4. Synonyms
    5. Parents
    6. Flaws
    7. Other relevant notes
4. Repeat steps 1 to 3 for any missing or unclear terminology

Methodology

Overview

Research Question 2

Are these descriptions consistent?

5. Analyze these data for 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

- A row is created for each test approach

Name	Category	Definition	Parent(s)	Synonym(s)
A/B Testing	Practice (p. 22)	Testing "that allows testers to determine which of two systems or components performs better" (p. 1)	Statistical Testing (pp. 1, 35), ...	Split-Run Testing (pp. 1, 35)

Information from (ISO/IEC and IEEE, 2022)

- A row is created for each test approach

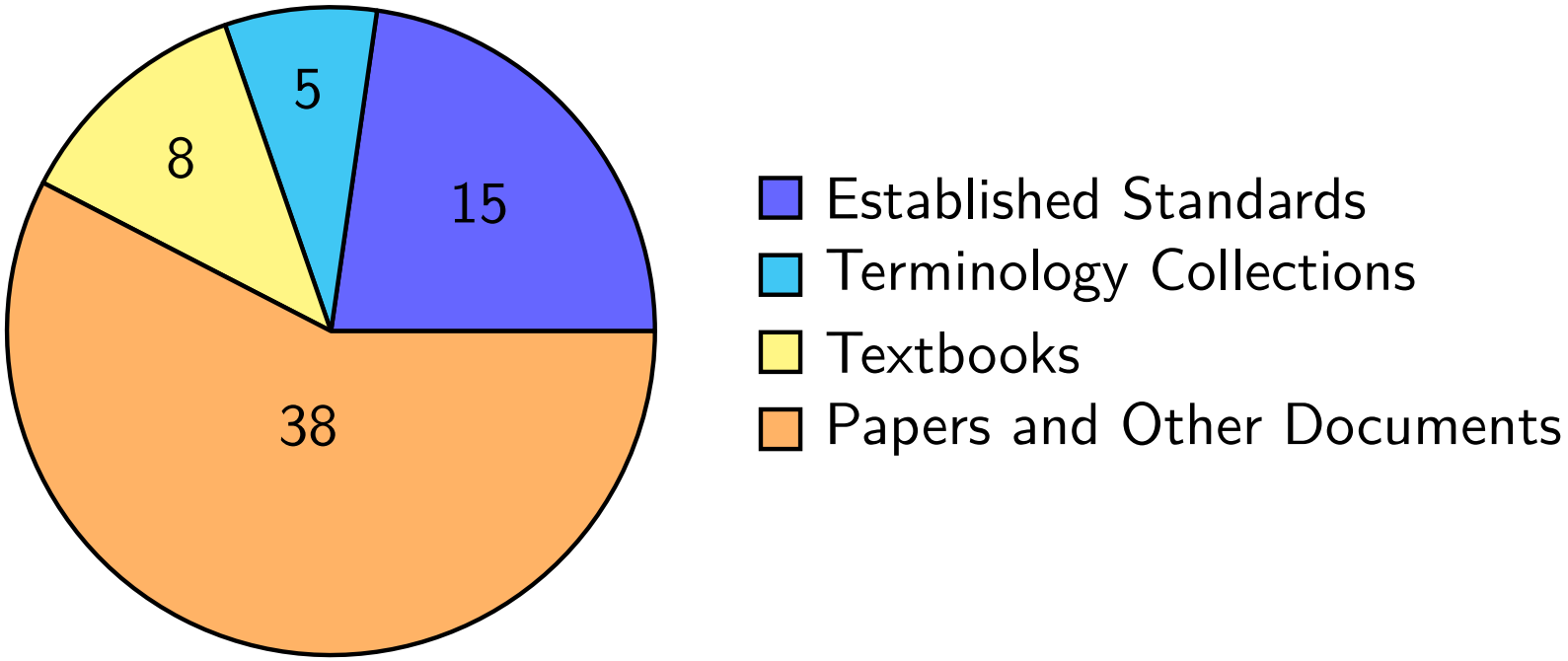
Name	Category	Definition	Parent(s)	Synonym(s)
A/B Testing	Practice (p. 22)	Testing "that allows testers to determine which of two systems or components performs better" (p. 1)	Statistical Testing (pp. 1, 35), ...	Split-Run Testing (pp. 1, 35)

Information from (ISO/IEC and IEEE, 2022)

- This information is gathered from sources by looking for
  - Glossaries
  - Testing-related terms
  - Terms described *by* other approaches
  - Terms that *imply* other approaches

Methodology

Sources

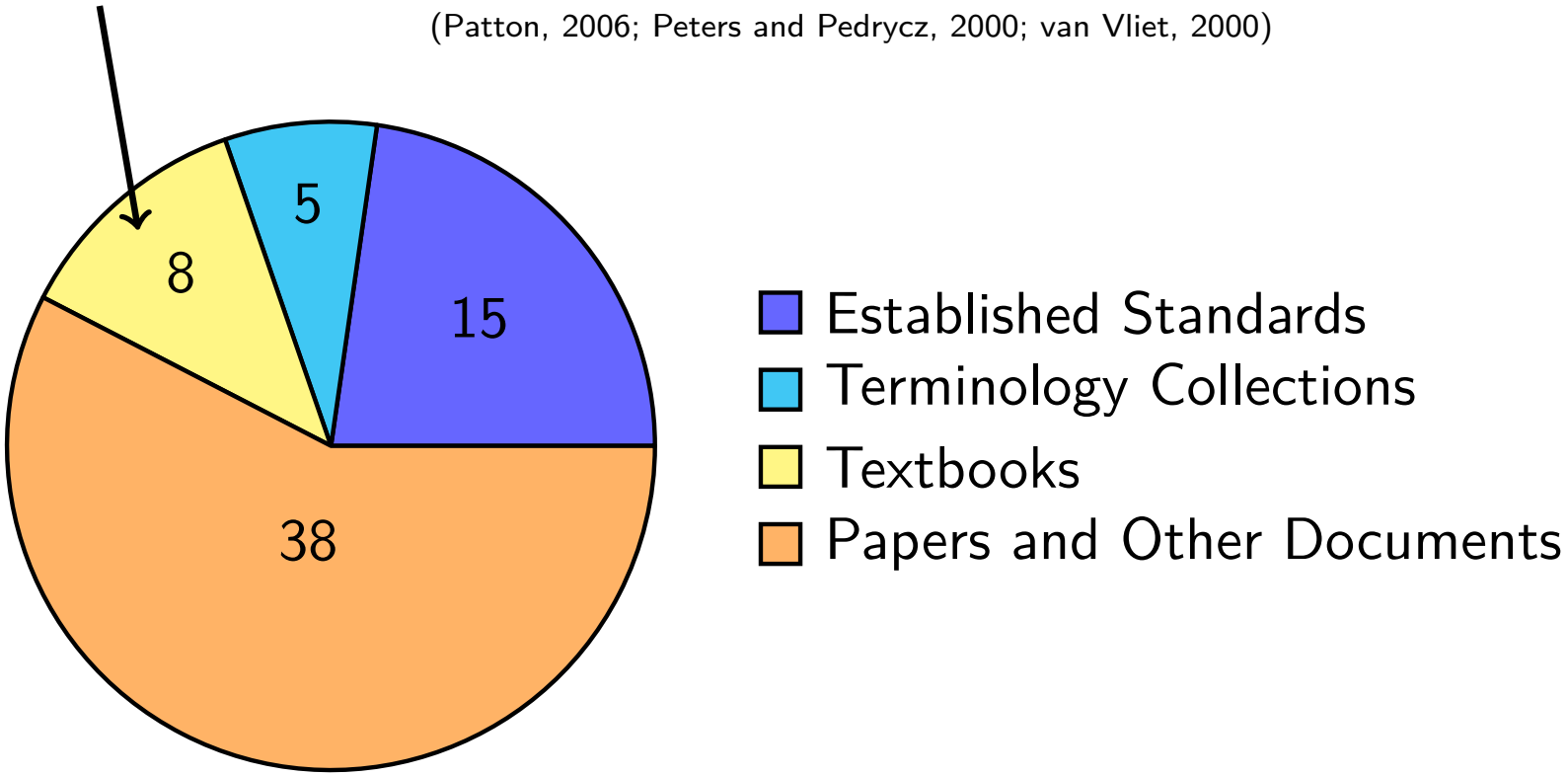


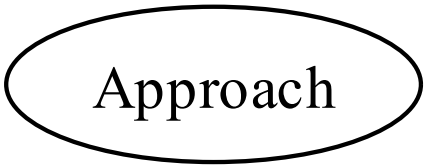
Methodology

Sources

Textbooks trusted at McMaster were our ad hoc starting points

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

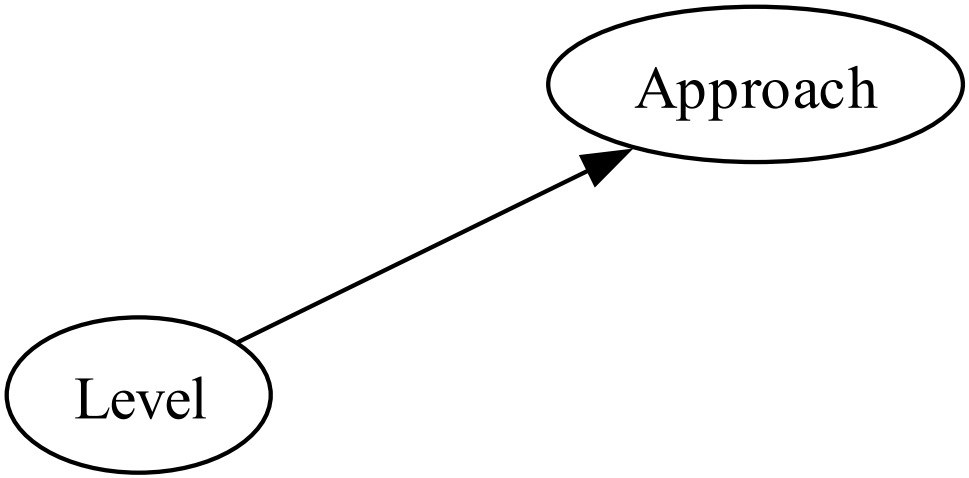




Approach: a "high-level test implementation choice" (ISO/IEC and IEEE, 2022, p. 10) used to "pick the particular test case values" (2017, p. 465)

Methodology

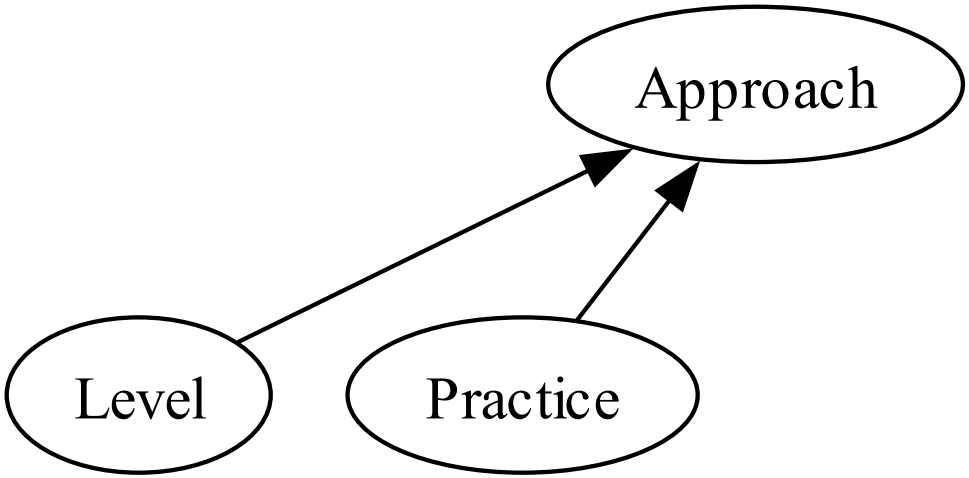
Categories



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

Methodology

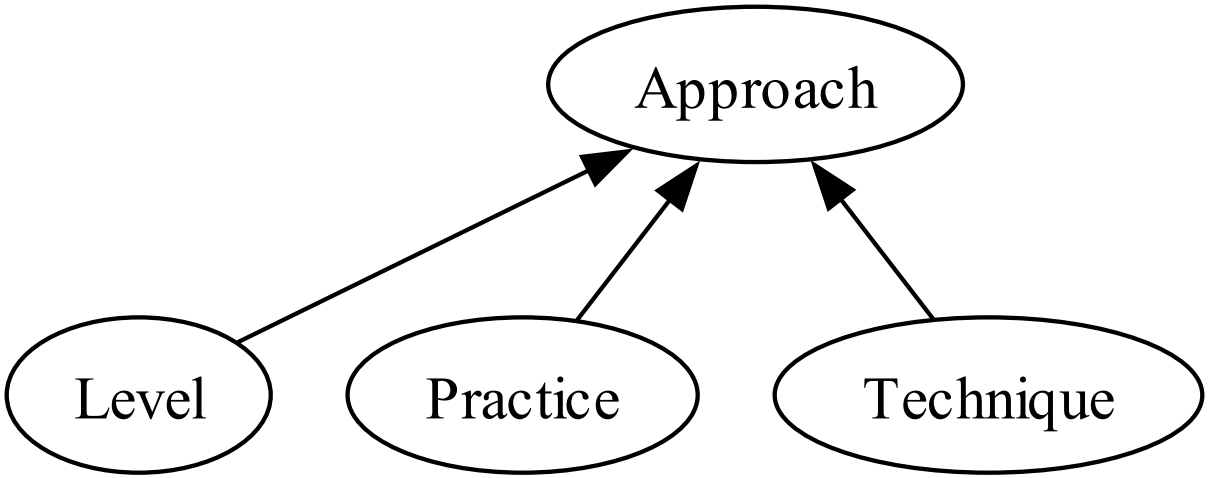
Categories



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

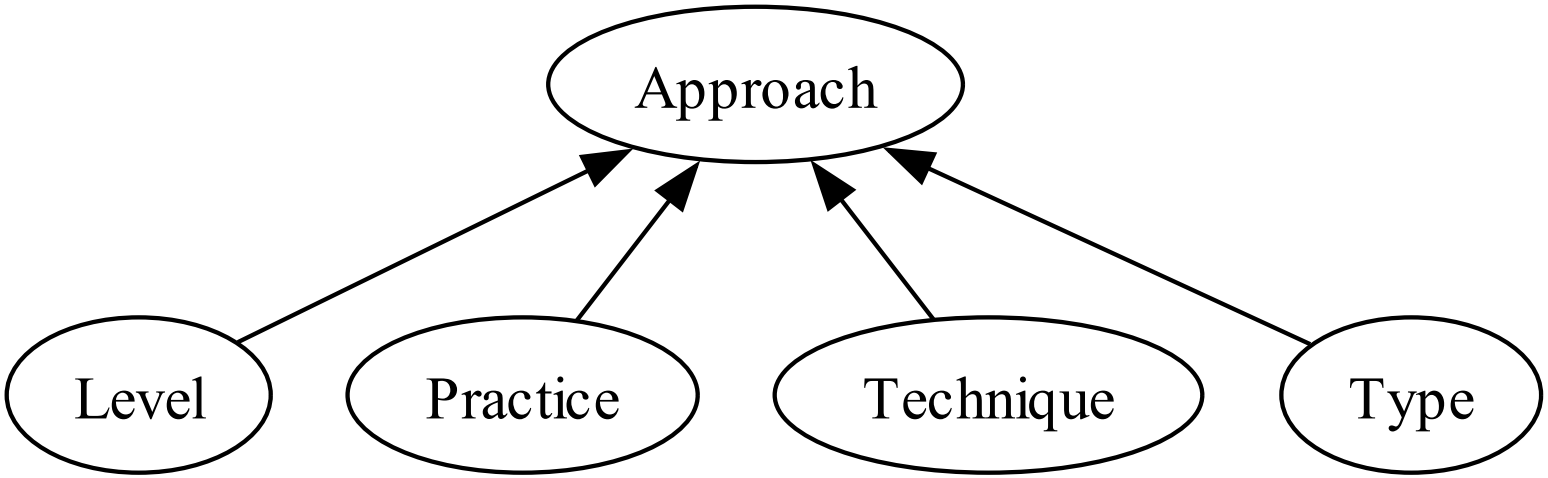
Categories



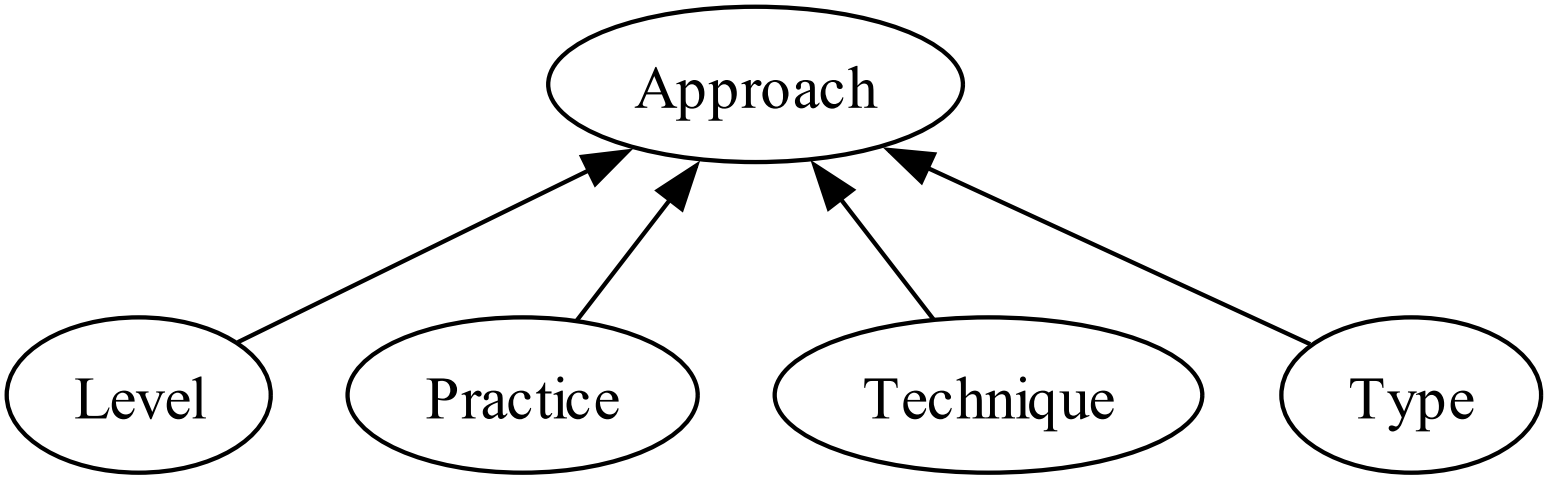
Technique: a "defined" and "systematic" (ISO/IEC and IEEE, 2017, p. 464) "procedure used to create or select a test model, identify test coverage items, and derive corresponding test cases" (2022, p. 11)

Methodology

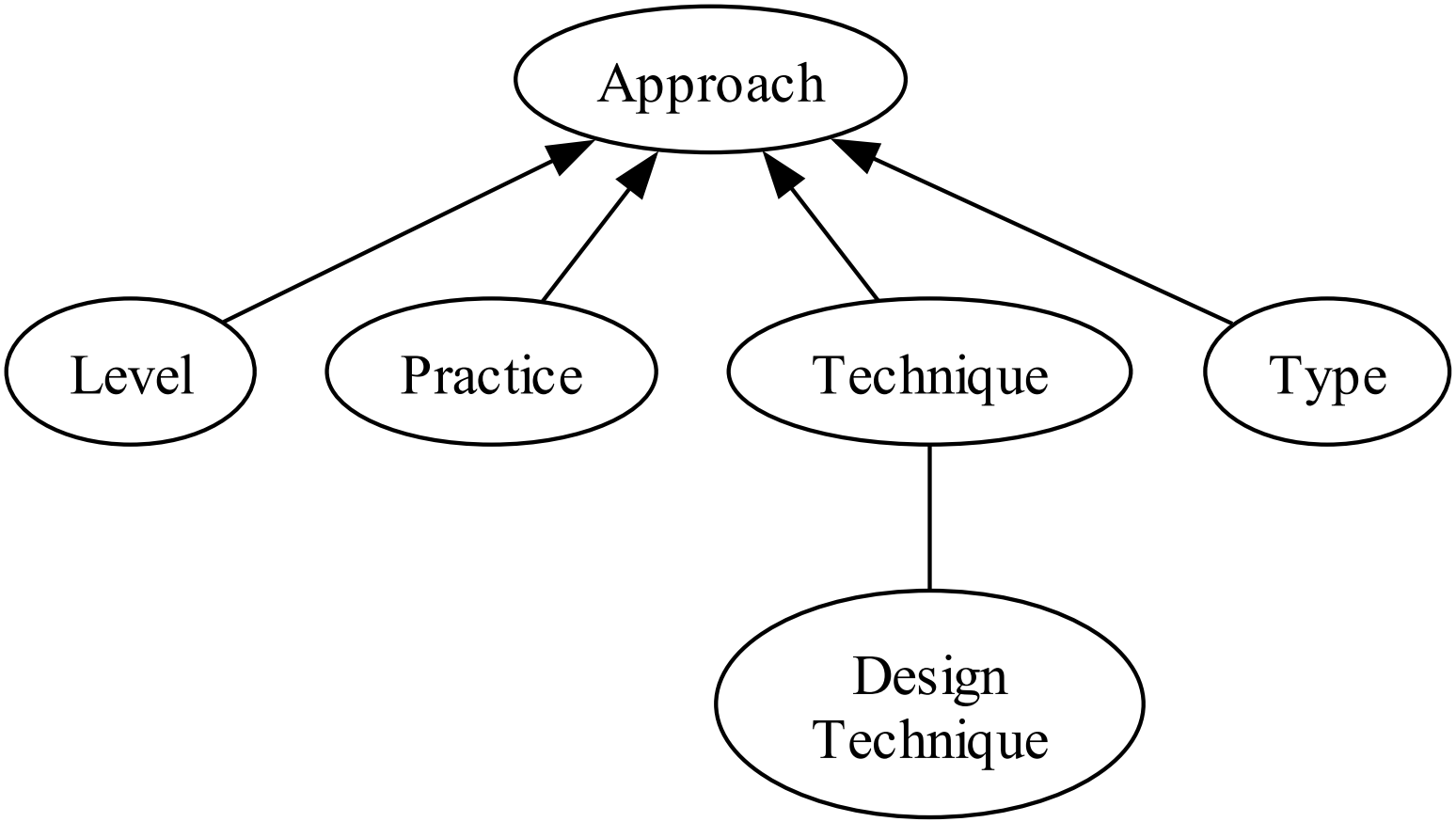
Categories



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



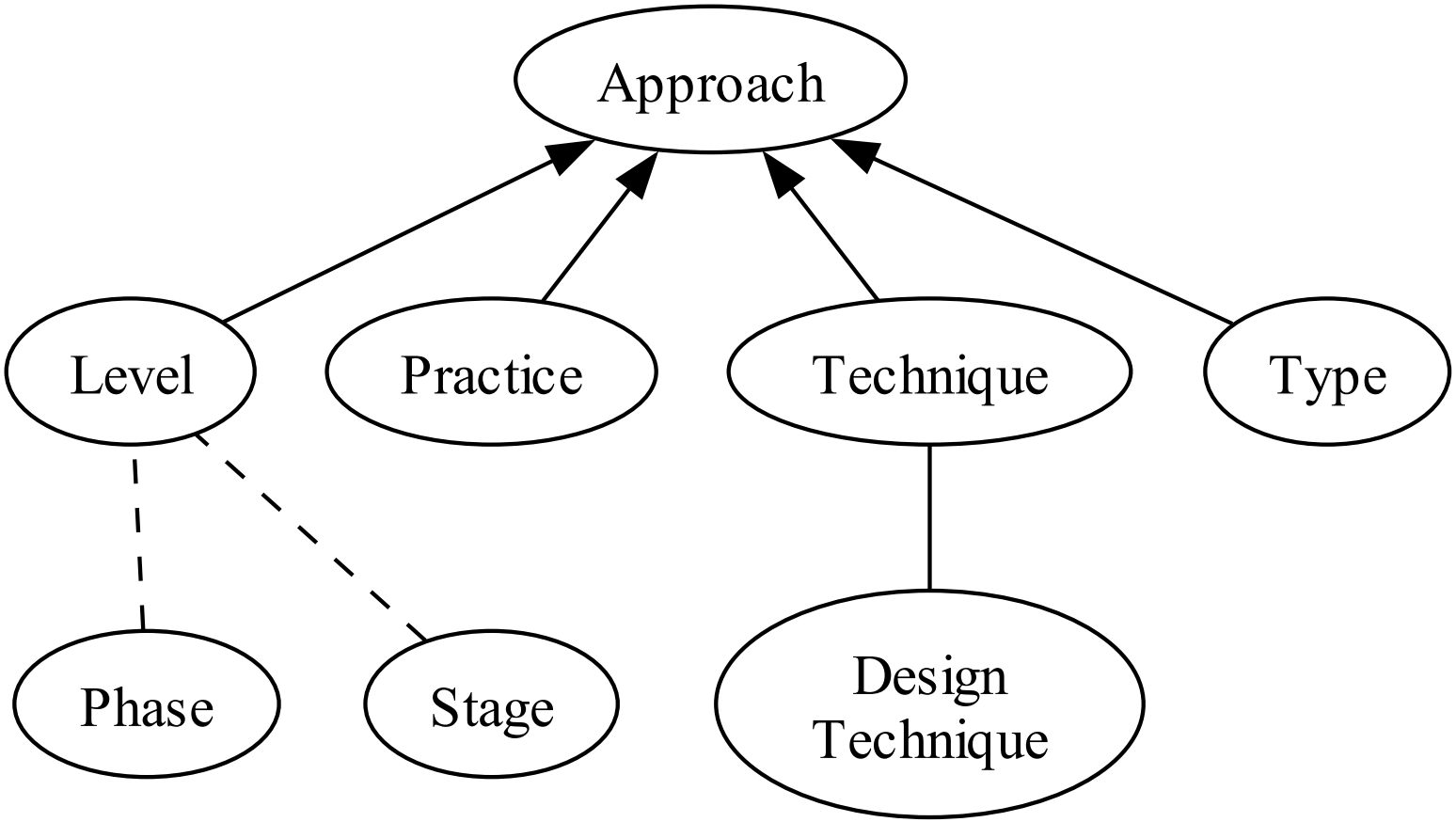
Arrows point from a *child* node to a *parent* node.



Lines without arrowheads connect *synonyms*.

Methodology

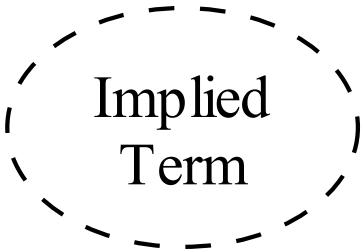
Graph Notation



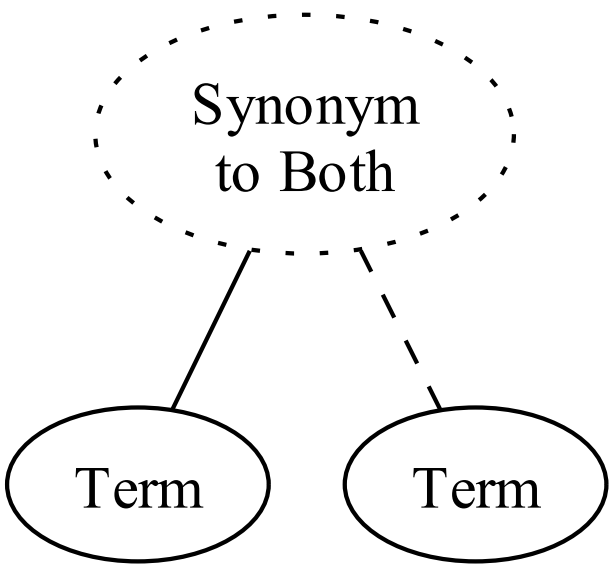
Dashed lines indicate a relationship is *implicit*.

Methodology

Graph Notation



Dashed outlines indicate a term is *implied*.



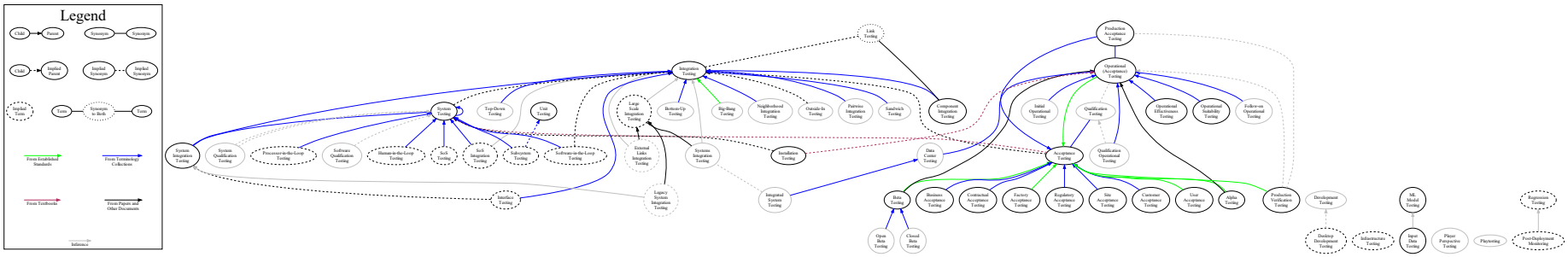
Dotted outlines indicate a term is a *synonym* to more than one term.

Graph of Test Approaches

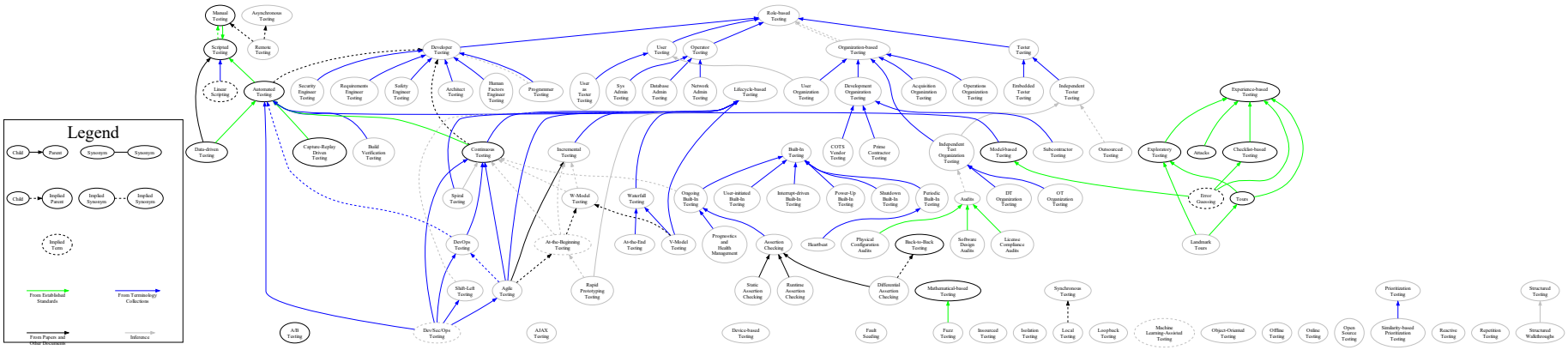
Graph of Test Approaches

```
! Dimension too large.
```

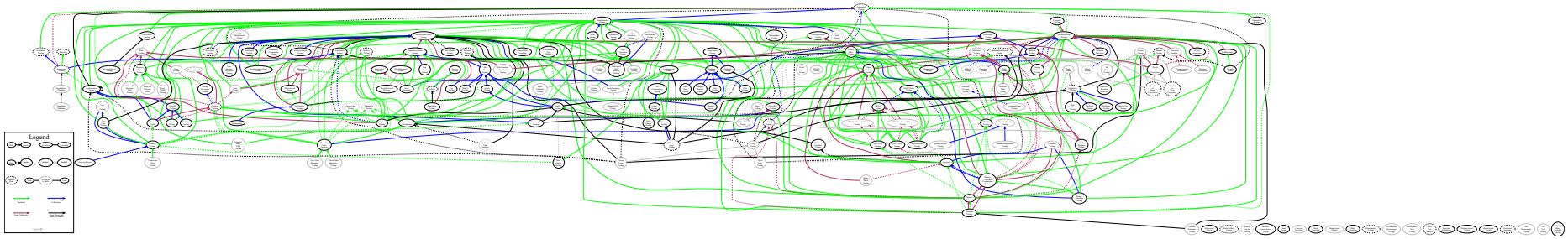
Graph of Test Levels



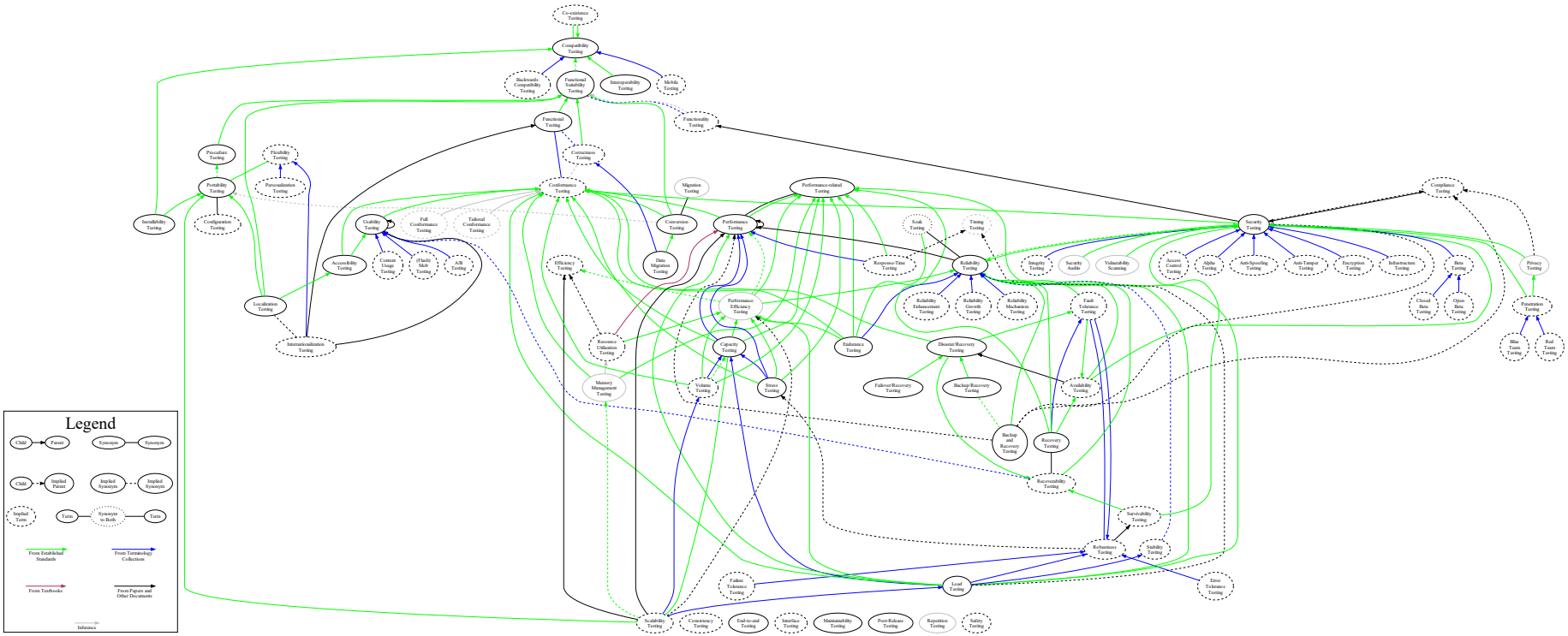
Graph of Test Practices



Graph of Test Techniques

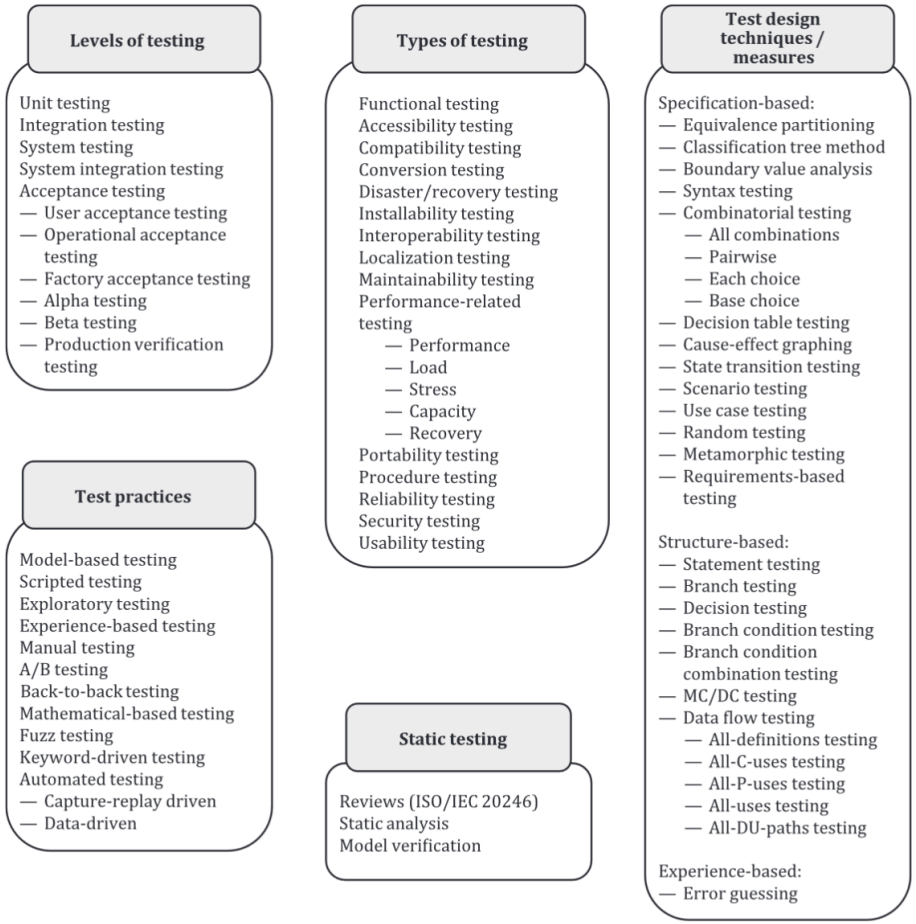


Graph of Test Types



Methodology

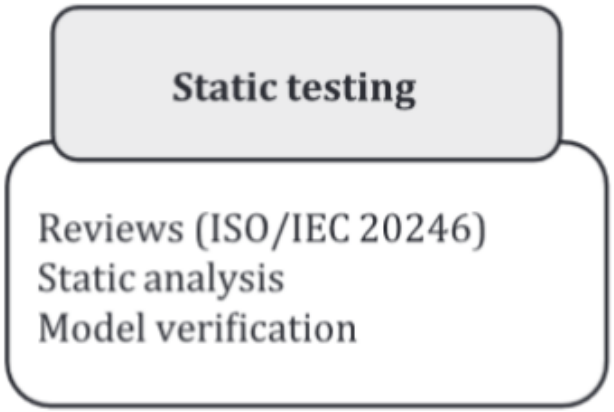
Graph Notation



(ISO/IEC and IEEE, 2022, Fig. 2)

Methodology

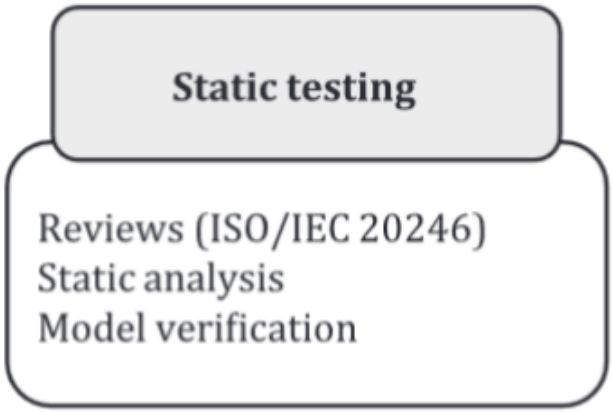
Graph Notation



Adapted from (ISO/IEC and IEEE, 2022, Fig. 2)

Methodology

Graph Notation

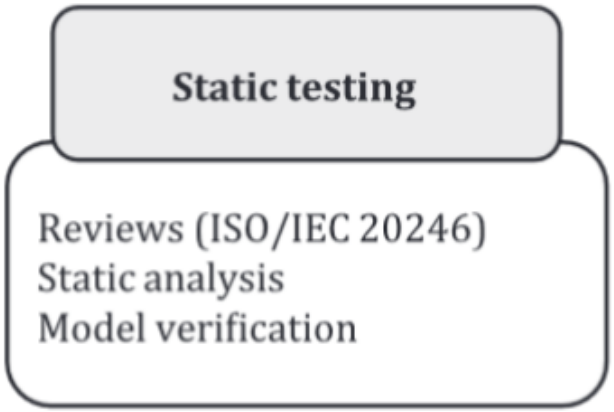


- Quite distinct but not necessarily orthogonal

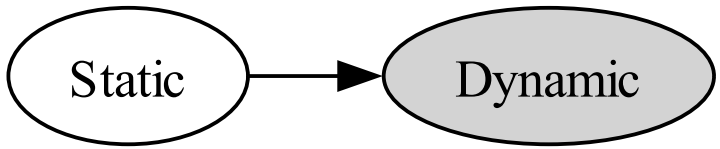
Adapted from (ISO/IEC and IEEE, 2022, Fig. 2)

Methodology

Graph Notation



- Quite distinct but not necessarily orthogonal
- When considering static testing in isolation, related *dynamic approaches* have grey backgrounds



Adapted from (ISO/IEC and IEEE, 2022, Fig. 2)

Graph of *Static* Test Approaches

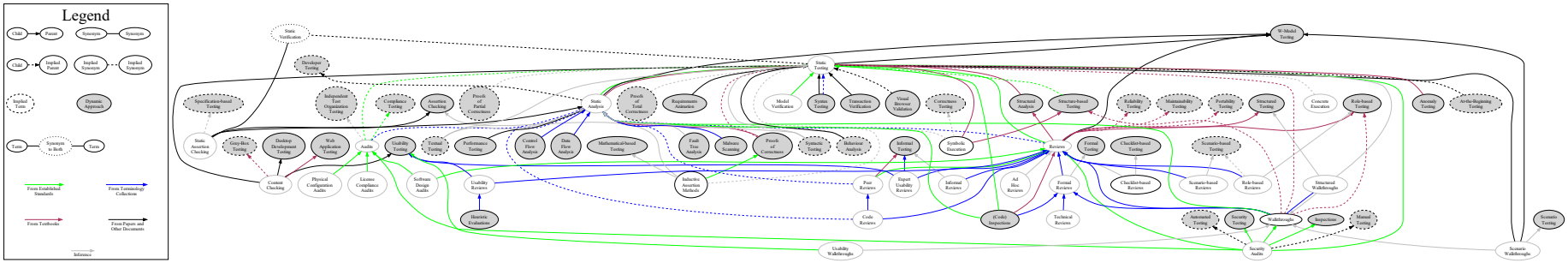


Table of Contents

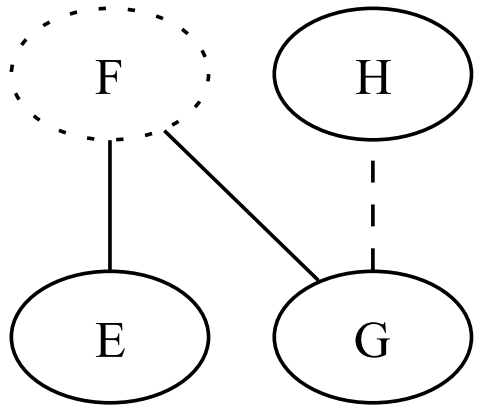
Automated Flaws

- 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 graphs automatically

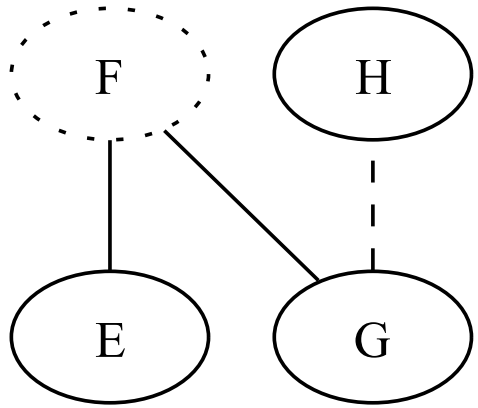
Name	Synonym(s)
E	F (Author, 0000; implied by 0001)
G	F (Author, 0002), H (implied by 0000)
H	X



Automated Flaws

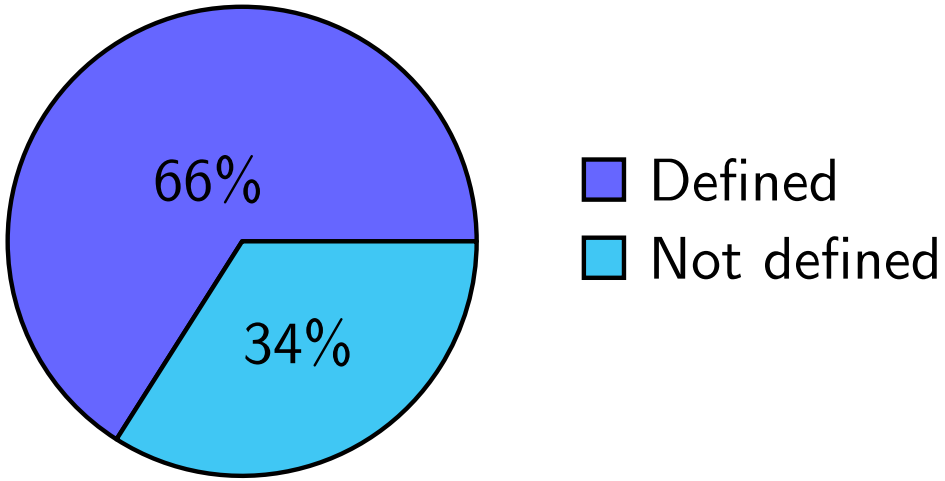
- Some terms are given as a synonym to two (or more) disjoint, unrelated terms, making the relation between the given synonyms ambiguous
- These are included in generated graphs automatically

Name	Synonym(s)
E	F (Author, 0000; implied by 0001)
G	F (Author, 0002), H (implied by 0000)
H	X



Results

- 560 test approaches $\rightarrow$
- 75 software qualities
  (may imply test approaches)



Automated Flaws

Prominent examples of these “multi-synonyms”:

1. **Invalid Testing:**
   - Error Tolerance Testing
   - Negative Testing

Source(s)

(Kam, 2008, p. 45)

(Hamburg and Mogyorodi, 2024)

Automated Flaws

Prominent examples of these “multi-synonyms”:

	Source(s)
1. **Invalid Testing:**	
• Error Tolerance Testing	(Kam, 2008, p. 45)
• Negative Testing	(Hamburg and Mogyorodi, 2024)
2. **Soak Testing:**	
• Endurance Testing	(ISO/IEC and IEEE, 2021, p. 39)
• Reliability Testing	(Gerrard, 2000a, Tab. 2; 2000b, Tab. 1, p. 26)

Automated Flaws

Prominent examples of these "multi-synonyms":

	Source(s)
1 Invalid Testing:	
• Error Tolerance Testing	(Kam, 2008, p. 45)
• Negative Testing	(Hamburg and Mogyorodi, 2024)
2 Soak Testing:	
• Endurance Testing	(ISO/IEC and IEEE, 2021, p. 39)
• Reliability Testing	(Gerrard, 2000a, Tab. 2; 2000b, Tab. 1, p. 26)
3 Link Testing:	
• Branch Testing	(implied by ISO/IEC and IEEE, 2021, 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

AzaToth. Myoglobin 3D structure, February 2008. URL https://commons.wikimedia.org/wiki/File:Myoglobin.png.

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.

References II

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 - Software and systems engineering –Software testing –Part 4: Test techniques. *ISO/IEC/IEEE 29119-4:2021(E)*, October 2021. 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

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.

Kjerish. Part of CNO cycle diagram, made just to be illustrative for nuclear reactions in general, December 2016. URL https://commons.wikimedia.org/wiki/File:NuclearReaction.svg.

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

Penubag and Arnaud Ramey. A few images illustrating forces, August 2010. URL https://commons.wikimedia.org/wiki/File:Force_examples.svg.

References IV

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.

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 V

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. URL `https://waseda.app.box.com/v/SWEBOK4-book`.