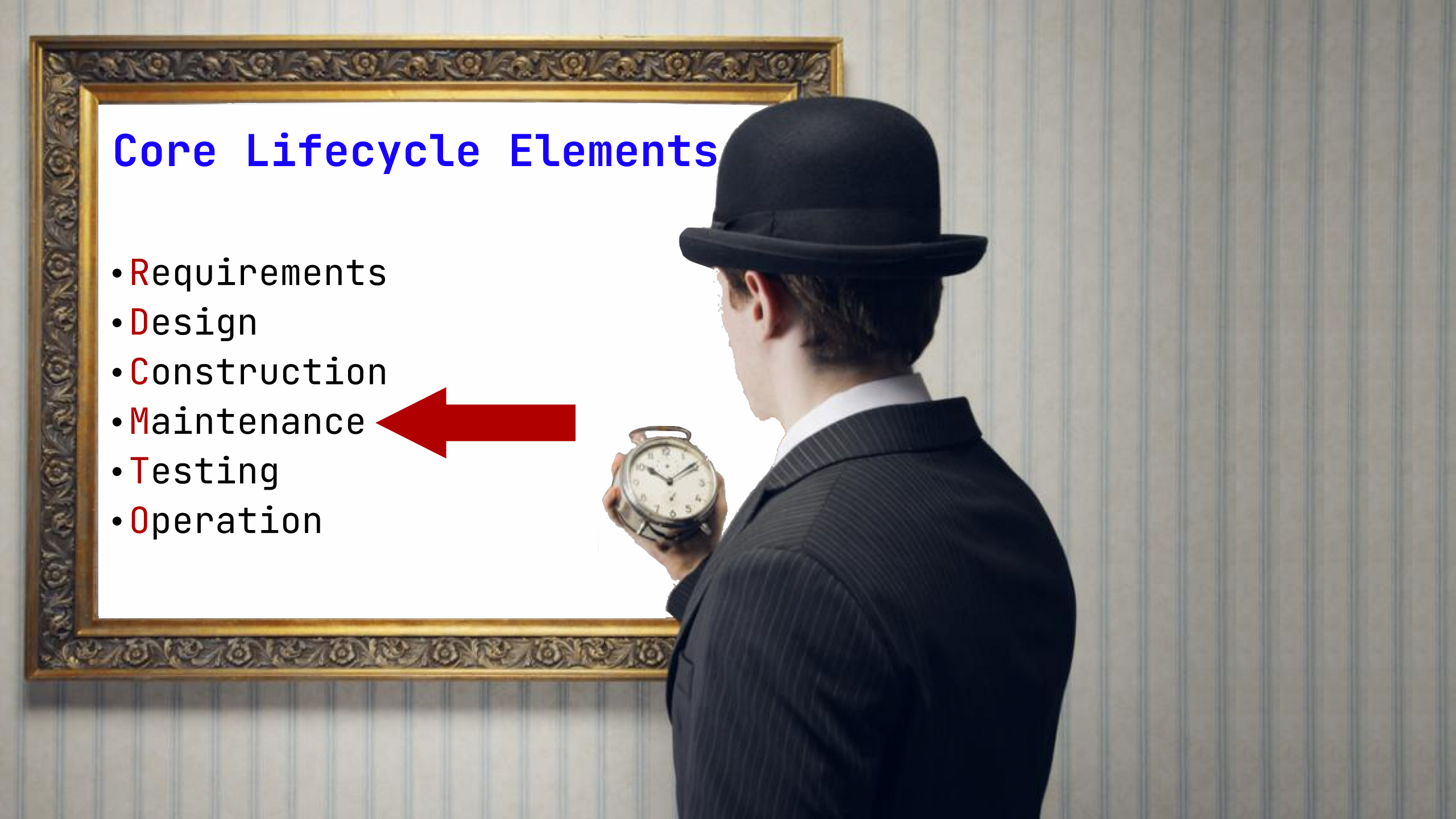
Modelling in Testing

Software Systems — Design — L2T5

Dr. Vadim Zaytsev aka @grammarware, November 2020

UNIVERSITY OF TWENTE.
Formal Methods & Tools


Core Lifecycle Elements
•Requirements
•Design
•Construction
•Maintenance
•Testing
•Operation

Test Policy

- Written on the level of an organisation
- What does testing mean for the company?
  - reviews?
  - user acceptance?
  - best effort?
  - coverage?
- Rarely changes
- Reusable across projects

Test Strategy

- Written on the level of a system
- Defines
  - objective/scope of testing
  - risks to be addressed
  - test levels
  - test types
  - testing methodology
  - defect classification

Test Plan

- Written on the level of a project
- Concrete instantiations for test strategy issues
- Declares
  - testing schedule
  - test roles & responsibilities
  - test deliverables
  - test reporting

to be continued
in P-L6T1, P-L6T4

ISO/IEC/IEEE 29119-3:2013: *Software Testing – Test Documentation*

Test Story

- Written on the level of a user
- One example scenario
- Often templated
  - A user can …
  - As a … I can …, so that …
  - In order to … as a …, I can …
  - As … … …, I … because …
- 2D visualisation of a product backlog

Test Case

- Written on the level of code
- Can specify:
  - inputs
  - expected outputs
  - execution conditions
  - procedure to be automated
  - documentation & category
- Realistic projects have thousands!

to be continued
in P-L7T2, P-L7T3

Test Techniques [1/2]

- Experience-based
  - error guessing
- Structure-based
  - branch testing / decision testing
  - branch condition testing
  - branch condition combination testing
  - data flow testing
  - statement testing

Test Techniques [2/2]

- Specification-based
  - equivalence partitioning
  - boundary value analysis
  - cause-effect graphing
  - classification tree method
  - combinatorial testing
  - decision tree testing
- random testing
- scenario testing
- state transition testing
- syntax testing

Test-Driven Development

- Slogans
  - test first
  - keep it simple
  - you ain’t gonna need it
  - fake it till you make it
  - code without tests is bad code
- TDD is a discipline
  - it does not come naturally

a.o. M.Feathers, *Working Effectively with Legacy Code*, 2004.

Test-Driven Development Laws

- First Law
  - no code before a failing test case
- Second Law
  - no more test code than needed to fail
- Third Law
  - no more code than needed to pass

R.C.Martin, *Clean Code*, Chapter 9

Conclusion

- Test X
  - policy ⇒ strategy ⇒ plan ⇒ story ⇒ case
- Test suite
  - collection of all test cases
- Test harness
  - the framework used to run tests
- Use TDD!
- Automate!

Topics/Slides Disclaimer

- Good ✅
  - watch before Q&A
  - embrace reality
  - try out at labs
  - ask for feedback
  - apply to project
  - dig deeper
  - recall from slides
- Bad ❌
  - slides over videos
  - assumptions
  - blanks
  - timing