University of Zurich

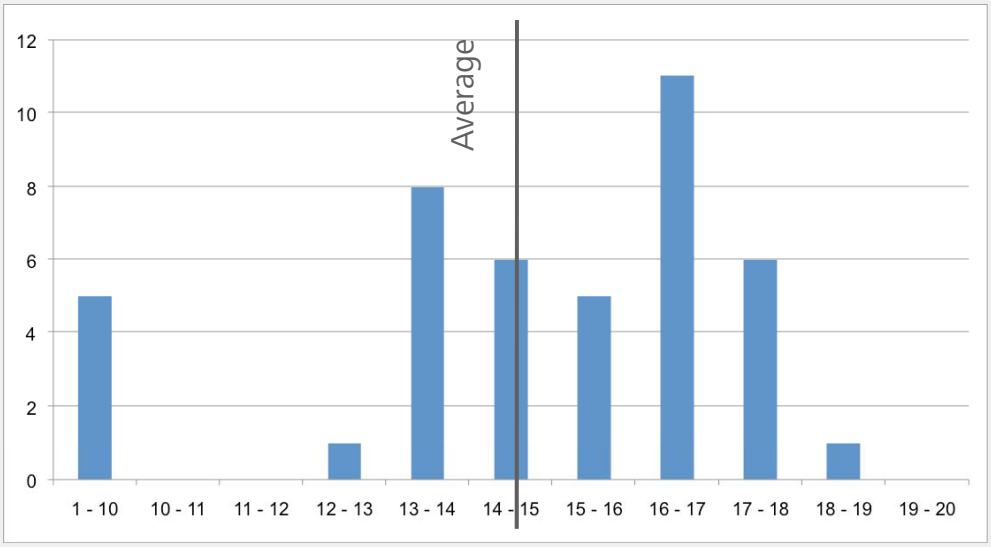
Discussion SE Exercise 6

Dustin Wüest and Cédric Jeanneret

Requirements Engineering Research Group
Department of Informatics
University of Zurich

University of Zurich

SE Exercise 6 Results



Ex 2.1.a: Black Box and White Box

System Testing:

Black Box
(state inspection for vehicles)

White Box
(inspection of a car by a mechanic)

Unit Testing (like the *getShortName()* method):

Black Box
(*User* experience)

White Box
(*Developer* experience)

Ex 2.1.a: Black Box Testing

Agent Names for JClusim…

Equivalence Classes

- Names without slashes: *SimpleAgent*
- Names with slashes: *IFI/RERG/SimpleAgent*
- Names with slashes at the end: *IFI/RERG/*

Boundary Values

- Empty names
- Names only made of slashes: *////*

Ex 2.1.b: Black Box Testing

University of Zurich

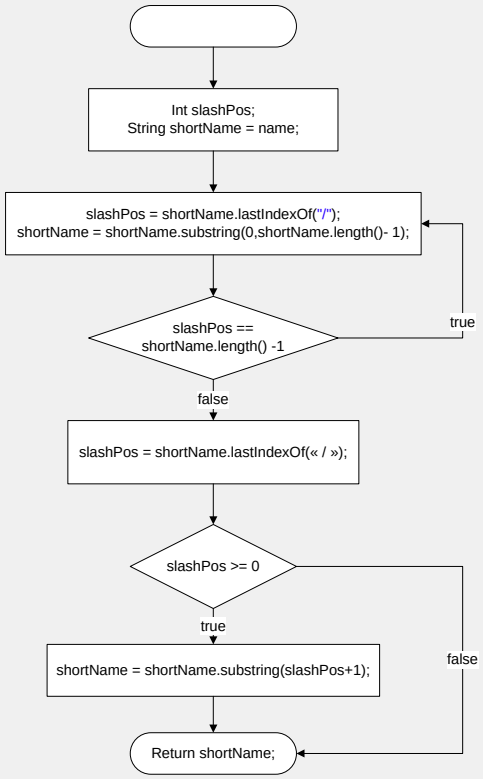
#	Input	Expected Output	Actual Output	Result
1	SimpleAgent	SimpleAgent	SimpleAgen	Failure
2	IFI/RERG/SimpleAgent	SimpleAgent	SimpleAgen	Failure
3	IFI/RERG/	RERG	RERG	Success
4	(Empty String)	(Empty String)	(Exception)	Failure
5	/////	(Empty String)	(Empty String)	Success

Ex 2.1.c: Branch Coverage

University of Zurich

4 branches:
•while (true/false)
•If (true/false)

3 branches covered
→ 75%



Ex 2.1.d: Branch Coverage

It is impossible to achieve 100% branch coverage with a single test case: the IF-branch cannot be evaluated to true and false within a single execution!

With two test cases:

#	Input	Expected Output	Actual Output	Result
1	/A/A	A	A	Success
2	A/	A	A	Success

Ex 2.1.e: White Box Testing

Easy to detect with white box testing:

- Wrongly implemented functionality or programming errors (localisation of the defect)
- Dead code (by trying to achieve full instruction coverage)

Hard to detect with white box testing:

- Errors in the specifications
- Errors at the interfaces
- Problems with interactions with other components

Ex. 2.2.a: GQM
Examples

University of Zurich

Easy and rapid registration for students

Factors:

- Clarity of the user interface
  - Does the user know which task he is currently performing?
- Intuitivity of the user interface
  - Is the UI designed in a way that the users feel comfortable?
- Simplicity of registration process
  - How many steps are required for the registration?
- Response time of the server
  - Has the system acceptable response time?
- Number of problems
  - How many students required an intervention from the secretary?

Ex. 2.2.b: GQM
Examples

University of Zurich

Measures (type of scale):

- Does the user know which task he is currently performing?
  - Support provided by the system (nominal: yes/no)
- Is the UI designed in a way that the users feel comfortable?
  - Users satisfaction (ordinal: --, -, ~, +, ++)
- How many steps are required for the registration?
  - Number of steps for a normal registration (absolute)
- Response time of the server
  - Latency (ratio scale)
- How many students required an intervention from the secretary?
  - Number of problematic registration (absolute)