



# Review report of a final thesis

**Reviewer:** doc. Ing. Filip Křikava, Ph.D.  
**Student:** Bc. Filip Dolník  
**Thesis title:** Exception flow analysis for Kotlin  
**Branch / specialization:** System Programming  
**Created on:** 31 May 2022

## Evaluation criteria

### 1. Fulfillment of the assignment

- ▶ [1] assignment fulfilled
- [2] assignment fulfilled with minor objections
- [3] assignment fulfilled with major objections
- [4] assignment not fulfilled

The assignment is complex one as the student decided to work with a real, fairly new and fairly feature-full programming language. The work done definitely goes beyond the standards of master thesis at FIT.

### 2. Main written part 100/100 (A)

I'm extremely pleased with the document. Honestly, I do not remember the last time I enjoyed so much reviewing a thesis. The first chapter should be shown in the course where students are learning how to do a thesis. The document clearly describes not only what has been done, but why it has been done and what were the choices assessing their trades-offs. There is a clear and well presented story. While at some parts the writing style falls into a blog-like form, overall it is clear and engaging. One thing I would try is to shorten the second chapter by making it a bit more formal. The language description is rather verbose. The evaluation is rather qualitative and it would be quite nice to do a quantitative one - take existing projects, identify domain exceptions (however synthetically) and perform the analysis. It would be interesting to see what all could be analyzed by the prototype.

### 3. Non-written part, attachments 100/100 (A)

The work implied extensive implementation (tokei reports over 30K SLOC) which also goes beyond the scope what is required. Big portion of this are tests. I'm not a Kotlin developer, but the code seems to be fine, easy to follow. I managed to run the tests and I played with the provided demo.

#### **4. Evaluation of results, publication outputs and awards** 100 /100 (A)

Next to the tool which could eventually reach production, it would be great to split the thesis into a series of blog posts about building static analysis for Kotlin.

#### **The overall evaluation** 100 /100 (A)

Overall, I'm very happy with the thesis. It goes beyond what is expected, it is well written and comes with extensive implementation.

#### **Questions for the defense**

1. How difficult would be to support analysis of closed-source libraries (say on the JVM)? This problem is mentioned on page 101, but the first part is not completely clear and I would be wondering how would you approach it without the need for manual specification.
2. The same goes for the reflection, could you explain the "elegant solution if the loaded class source code is available."?
3. One thing that bothered me while reading the thesis is the problem that it is trying to solve. It feels that the essence of the problem is somewhere else. Isn't the extensive use of exceptions an anti-pattern? What would be the alternatives?
4. Right now there are no checked exceptions in Kotlin. In principle, they are good as they enhance the information about the method contract. Yet, having them, the Java-style is cumbersome and generally acknowledged as a bad design. Is there any middle ground?

## **Instructions**

### **Fulfillment of the assignment**

Assess whether the submitted FT defines the objectives sufficiently and in line with the assignment; whether the objectives are formulated correctly and fulfilled sufficiently. In the comment, specify the points of the assignment that have not been met, assess the severity, impact, and, if appropriate, also the cause of the deficiencies. If the assignment differs substantially from the standards for the FT or if the student has developed the FT beyond the assignment, describe the way it got reflected on the quality of the assignment's fulfilment and the way it affected your final evaluation.

### **Main written part**

Evaluate whether the extent of the FT is adequate to its content and scope: are all the parts of the FT contentful and necessary? Next, consider whether the submitted FT is actually correct – are there factual errors or inaccuracies?

Evaluate the logical structure of the FT, the thematic flow between chapters and whether the text is comprehensible to the reader. Assess whether the formal notations in the FT are used correctly. Assess the typographic and language aspects of the FT, follow the Dean's Directive No. 52/2021, Art. 3.

Evaluate whether the relevant sources are properly used, quoted and cited. Verify that all quotes are properly distinguished from the results achieved in the FT, thus, that the citation ethics has not been violated and that the citations are complete and in accordance with citation practices and standards. Finally, evaluate whether the software and other copyrighted works have been used in accordance with their license terms.

### **Non-written part, attachments**

Depending on the nature of the FT, comment on the non-written part of the thesis. For example: SW work – the overall quality of the program. Is the technology used (from the development to deployment) suitable and adequate? HW – functional sample. Evaluate the technology and tools used. Research and experimental work – repeatability of the experiment.

### **Evaluation of results, publication outputs and awards**

Depending on the nature of the thesis, estimate whether the thesis results could be deployed in practice; alternatively, evaluate whether the results of the FT extend the already published/known results or whether they bring in completely new findings.

### **The overall evaluation**

Summarize which of the aspects of the FT affected your grading process the most. The overall grade does not need to be an arithmetic mean (or other value) calculated from the evaluation in the previous criteria. Generally, a well-fulfilled assignment is assessed by grade A.