Bachelor Thesis Review

Faculty of Mathematics and Physics, Charles University

Thesis author	Stanislav Lukeš
Thesis title	Rozhraní pro generování C# kódu
Year submitted	2020
Study program	Computer Science
Study branch	Programming and Software Development
Review author	Filip Kliber Reviewer
Department	Department of Distributed and Dependable Systems

Overall

	good	OK	poor	insufficient
Assignment difficulty	X			
Assignment fulfilled	X			
Total size *... text and code, overall workload*	X			

The goal of this thesis is to design and implement a C# code generator library, with an emphasis on generating readable and (syntax) error-free code. This library is then used in writing a code generator that translates GraphQL schemas into immutable C# classes. The goals of this thesis have been fulfilled and the solution is of very high quality. Author came up with a tool titled *Coberec*.

Thesis Text

	good	OK	poor	insufficient
Form *... language, typography, references*	X			
Structure *... context, goals, analysis, design, evaluation, level of detail*		X		
Problem analysis	X			
Developer documentation	X			
User documentation		X		

Textual part of the thesis is well-written and well-structured. In the first two chapters, author gives us an insight into various techniques and frameworks used in code generation, not only for C# language, such as Scala macros or D mixins. I have to admit I was surprised not to see an appearance of C macros there (as most likely one of oldest code generating machinery still in use today), but overall, the analysis is well made. The following chapter is dedicated to the design of the *Coberec* tool. It explains important decisions made during the development of the project and also explores different approaches. Next chapter contains an example of usage of the library, as well as lists available API. The *Hello World* example is well explained and gives the user a nice starting point on how to start using the library. The thesis also contains extensive lists of examples of how to generate important C# code fragments (such as calling a method, or declaring a type). The rest of the thesis is dedicated to technical and implementation details.

In the thesis author talks about *Coberec* without explaining it beforehand (that it is the tool this thesis is about), which might a bit confusing as it is explained in the final part of the thesis. Some parts of the thesis are more difficult to understand, which is probably because they expect a deeper knowledge of internals of the *ILSpy* tool.

(pokračování na další straně)

In Section 3.8, author states "Since local function is almost equivalent to a lambda assigned to a local variable, we have decided to simplify the concept and only support lambda function. When the lambda function is immediately assigned to a variable, Coberec will translate it into a C# local function". I am confused about whether only lambda functions are supported since Coberec will translate them into local functions.

In some parts of the thesis, author was trying to value the ability of being able to debug generated code, but it felt like the thesis lack the outcome of this discussion. But on the other hand, I feel that it is usually better to debug the code generator, instead of the generated code.

Thesis Code

	good	OK	poor	insufficient
Design *... architecture, algorithms, data structures, used technologies*	X			
Implementation *... naming conventions, formatting, comments, testing*	X			
Stability		X		

The project contains an implementation of the *Coberec* tool. It is based on ILSpy decompiler for .NET. The code is modern and mostly readable. There were some problems with compiling all parts of the project (`SampleProject` had errors), but I was able to run both the CLI version as well as use the library. For trying out the library, I had to use it from it's test project (as recommended by the author). Creating new project and referencing `ExprCS` didn't work for me, as I had to add required dependencies manually, as they were not documented anywhere.

Most code generation frameworks usually have easy to use API to allow for fast generation of boilerplate code. The API of the *Coberec* tool on the other hand is quite complex and requires a prior study before one can use it properly. This is because author wanted the resulting code to be readable and thus requires additional input from the user. The proposed and implemented API is very robust and allows generation of most of available C# constructs.

The project also contains a high amount of unit and integration tests, which is definitely something worth appreciating, because making sure that generated code is correct is very important in this case.

Overall grade Výborně

Award level thesis Ano

Date

Signature