Task Description – Bachelor Thesis
Visual Studio Code Integration for the Dafny Language and Program Verifier
FS 2017

1. Client & Supervisor

- *Dr. K. Rustan M. Leino, Microsoft Research, Redmond, WA, USA*
- *Client Contact:* leino@microsoft.com
- *Supervisor:* Prof. Dr. Farhad Mehta, HSR Rapperswil

2. Students

- Mr. Markus Schaden
- Mr. Rafael Krucker

3. Setting

Dafny is a language developed by Microsoft which offers built-in specification constructs. These include pre- and postconditions, frame specifications as well as termination metrics. Further support such as ghost variables and recursive functions are also implemented. Through such specification primitives, the Danfy verifier, invoked during compilation, can be used to verify the specified aspects of the functional correctness of a program.

Dafny is typically used via its Visual Studio [1] IDE integration under the Windows operating system. This integration allows for an efficient workflow of editing a program while constantly being given feedback about its the functional correctness. The Dafny compiler and verifier can additionally be invoked from the command line.

Microsoft would like to integrate of Dafny into the cross-platform Visual Studio Code [2] IDE. Work on this has already been started through a plugin by Jonathan Rionatan [3]. It currently works within the mono-environment [4] and provides feedback from the verifier.

4. Goals

The main goal of this thesis project is to improve the existing integration of Dafny within Visual Studio Code and thereby allow Dafny to be effectively used in a cross-platform setting. In particular, the following improvements and additions to the existing Visual Studio Code plugin are proposed:

1. Stable Working Release of the Plugin on the following Platforms
    - Windows 10 (.net-environment)
    - Linux (mono-environment)

    - MacOS (mono-environment)
2. Easy installation of the plugin, with an automated download of Dafny and the automatic setting of all system variables
3. Syntax-Highlighting
4. Compilation of Dafny Best Practices and reporting of their violations within the plugin
5. Automatic generation of contract/specification/manual proof suggestions for common and simple cases
6. Autocompletion for identifiers

Goals 1, 2, 3 and 4 have the highest priority since they provide the beginner with the greatest help.

Goal 5 is probably the most interesting feature, because it could bring much of the power of Dafny to the programmer with relatively little effort on his side. Since this feature does not have strong parallels to standard IDEs for programming, it will require thought and research to execute. Due to this, the focus of the project is currently planned here, after having learnt enough of the setting from the preceding goals.

Goal 6 currently has the lowest priority. It is unclear if autocompletion in the setting of Dafny is conceptually significantly different to IDEs for programming. The execution of this task is heavily dependent on the existing support from Visual Studio Code and the Dafny compiler, whereas its novelty and effectivity for the user is debatable to be currently placed higher in the list of priorities.

In addition to the goals stated above, the following points will be considered during the course of the project:

1. The use of Dafny in order to implement the features discussed.
2. Other currently unknown improvements to the workflow and IDE tooling.

5. Guidelines

The students and the supervisor will plan weekly meetings to check and discuss progress. The student will schedule meetings with the client as and when required (recommendation: 1 meeting per week of 1 hour duration).

All meetings are to be prepared by the students with an agenda. The agenda will be sent at least 24h prior to the meeting. The results will be documented in meeting minutes that will be sent to the supervisor.

A project plan must be developed at the beginning of the thesis to promote continuous and visible work progress. For every milestone defined in the project plan, the temporary versions of all artefacts need to be submitted. The students will receive a provisional feedback for the submitted milestone results. The definitive grading is however only based on the final results of the formally submitted report.

6. Documentation

The project must be documented according to the regulations of the Computer Science Department at HSR (see https://www.hsr.ch/Allgemeine-Infos-Bachelor-und.4418.0.html). All required documents are to be listed in the project plan. All documents must be continuously updated, and should document the project results in a consistent form upon final submission. All documentation and work artefacts have to be completely submitted in three copies on CD/DVD (one copy each for the client, university, and supervisor). Three printed copies of the report need to be submitted (one copy each for the client, external examiner, and supervisor).

7. Important Dates

Please refer to https://www.hsr.ch/Semesterdaten-2016-2017.13924.0.html.

8. Workload

A successful bachelor thesis project results in 12 ECTS credit points per student. One ECTS points corresponds to a work effort of 30 hours.

All time spent on the project must be recorded and documented.

9. Grading

The HSR supervisor is responsible for grading the bachelor thesis. The following table gives an overview of the weights used for grading.

Facet	Weight
1. Organisation, Execution	1/6
2. Report	1/6
3. Content	3/6
4. Final Presentation & Examination	1/6

The effective regulations of the HSR and Department of Computer Science apply (see https://www.hsr.ch/Ablaeufe-und-Regelungen-Studie.7479.0.html).

Rapperswil, 25.11.2016
Prof. Dr. Farhad Mehta

References:

[1] https://github.com/Microsoft/dafny/wiki/INSTALL

[2] https://code.visualstudio.com

[3] https://github.com/ferry-/dafny-vscode

[4] https://github.com/mono/mono