Master M2 internship: *"Investigating co-evolution in modern polyglot software applications"* (**possible follow-up with a PHD**)

Location: Rennes, IRISA Lab, DiverSE team : http://www.diverse-team.fr

It can be done remotely as well due to Covid-19.

Duration: 6 months.

Supervisor 1:

Djamel E. Khelladi, CNRS researcher

Email : djamel-eddine.khelladi@irisa.fr

Supervisor 2:

Benoit Combemale, Prof., Univ. Rennes 1

Email : benoit.combemale@irisa.fr

Supervisor 3:

Arnaud Blouin, Assoc. Prof. INSA Rennes

Email : arnaud.blouin@irisa.fr

Context and Problem.

The evolution of software engineering discipline has seen the emergence of a multitude of programming languages (PL), each dedicated to a particular application concern.

Nowadays software development is complex to the point where not only a single language is sufficient for the implementation of a complex application. Indeed, we observe more and more software projects that are implemented with various PLs, in particular, to combine their strengths and counterbalance their weaknesses. We refer to such software development as ***polyglot***.

Just on GitHub, one can see thousands of projects using more than one PL, such as Apache Arrow (https://github.com/apache/arrow [C++, Java, Rust, Python, TypeScript, Ruby, …]) or Kafka (https://github.com/apache/kafka [Java, Scala, Python, Shell, …]). Other emerging examples can be found on the GrallVM (https://www.graalvm.org/why-graalvm) virtual machine that allows the combination of several PLs, and also on the Note Book side, e.g., with PolyNote (https://polynote.org).

Polyglot programming of software do offer many advantages, however, they come with cost. One of the issues that arise is w.r.t. evolution over time. Indeed, since programs written in different languages co-exists, when one evolves, others may be impacted, and then must be co-evolved accordingly.

A simple example would be a variable/method declaration in Python, and its usage in Java or Scala. If the variable/method type changes or it is renamed, then its references in the other programs are impacted and must be co-evolved.

This scenario is amplified in collaborative development and in modern architecture like microservices, where different developers work on different parts of the code. In the case of Polyglot software, developers may work separately on a single PL without being aware of how the other programs in other PLs evolve. This may cause build failures in the CI pipeline, introduce silent bugs, and slow down the development. Therefore, it is essential to support co-evolution to maintain a global consistency with delivering feedback to developers

Objectives.

This internship proposes to first investigate the current state of polyglot software development and the existing challenges w.r.t. evolution and co-evolution. To fit the internship timeframe, will first classify existing scenarios of polyglot programming and the current state of practices and techniques involved to address them. In particular, techniques that focus on handling the co-construction and their evolution of the software.

The second goal is to lay the foundations of a co-evolution framework dedicated for polyglot software. In particular, the conceptual architecture and design of a polyglot-based co-evolution framework that can cope with different PLs, and its prototype implementation.

This internship can follow up with a PhD thesis on the continuation of this work, *depending on the obtained results, demonstrated dedication and hard work of the candidate.* This internship as well as the PhD will be in the context of an ANR project.

We will provide the necessary supervision and technical infrastructure to carry out this ambitious project.

Required skills.

- Being (highly) autonomous, a problem solver, and desire to work in a team environment.

- Appetence for programming languages, and programming styles

- Strong technical skills in object-oriented programming.

- Very good level of written and spoken English.

- Experience in the development of IDE components will be a plus

To apply, please, send me the following documents:

- CV.
- Grades for your bachelor degree and Master degree (in computer science and software engineering).

- Motivation letter explaining why you want to work on this topic.

Below we provide some references (not exhaustive!). [1, 2] are preliminary studies around polyglot software. [3] is an approach to mine polyglot software in GitHub. [4, 5, 6] are additional references for works around polyglot development. Finally, [7, 8, 9] are for understanding the challenge of co-evolutuion in general.

References:

[1] Tomassetti, Federico, and Marco Torchiano. "An empirical assessment of polyglot-ism in github." In Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, pp. 1-4. 2014.

[2] Tomassetti, Federico, Giuseppe Rizzo, and Raphaël Troncy. "Crosslanguagespotter: a library for detecting relations in polyglot frameworks." In Proceedings of the 23rd International Conference on World Wide Web, pp. 583-586. 2014.

[3] Barmpis, Konstantinos, Patrick Neubauer, Jonathan Co, Dimitris Kolovos, Nicholas Matragkas, and Richard F. Paige. "Polyglot and Distributed Software Repository Mining with Crossflow." In Proceedings of the 17th International Conference on Mining Software Repositories, pp. 374-384. 2020.

[4] Niephaus, Fabio, Tim Felgentreff, and Robert Hirschfeld. "GraalSqueak: toward a smalltalk-based tooling platform for polyglot programming." In *Proceedings of the 16th ACM SIGPLAN International Conference on Managed Programming Languages and Runtimes*, pp. 14-26. 2019.

[5] Wampler, Dean, and Tony Clark. "Guest editors' introduction: multiparadigm programming." *IEEE Software* 27, no. 5 (2010): 20-24.

[6] Chen, Nicholas, and Ralph Johnson. "Toward refactoring in a polyglot world: extending automated refactoring support across java and xml." In *Proceedings of the 2nd Workshop on Refactoring Tools*, pp. 1-4. 2008.

[7] Djamel E. Khelladi, Roland Kretschmer, Alexander Egyed: Change Propagation-based and Composition-based Co-evolution of Transformations with Evolving Metamodels. *MODELS 2018.*

[8] Djamel E. Khelladi, Reda Bendraou, Regina Hebig, Marie-Pierre Gervais: A semi-automatic maintenance and co-evolution of OCL constraints with (meta)model evolution. *JSS 2017*.

[9] Khelladi, D. E., Combemale B., Mathieu A., Barais O., Jézéquel J-M. Co-Evolving Code with Evolving Metamodels. IEEE/ACM 42th International Conference on Software Engineering, ICSE 2020, http://people.irisa.fr/Djamel-Eddine.Khelladi/papers/ICSE-2020.pdf.