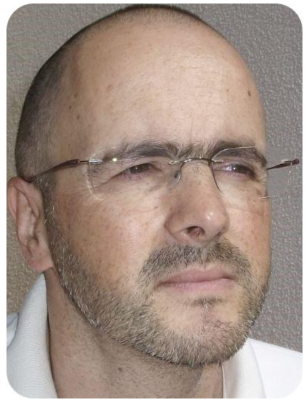
Temas de Tese 2020-2021



- Os temas de teses aqui apresentado serão orientados por mim, Paulo Ferreira, e co-orientados pelo Luis Veiga e pelo Rodrigo Bruno

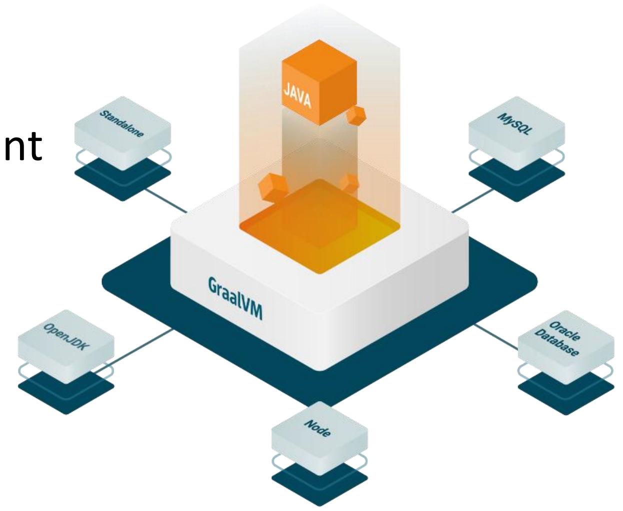
- Neste momento, as pessoas em causa encontram-se em:
  - Paulo Ferreira: na UiO (University of Oslo)
  - Luis Veiga: no INESC ID (Alameda ou Tagus)
  - Rodrigo Bruno: no ETH (Zurique)

- QQ contacto com Paulo Ferreira ou Rodrigo Bruno terá de ser via email ou Skype ou Zoom (ou qq outro método análogo); qq contacto com Luis Veiga, poderá ser feita presencialmente

- Mais informações:
  - Email: paulo.ferreira@inesc-id.pt / paulofe@ifi.uio.no
  - Skype: pjpf-inesc-id

GCgraalMS– Garbage Collection for Micro Services

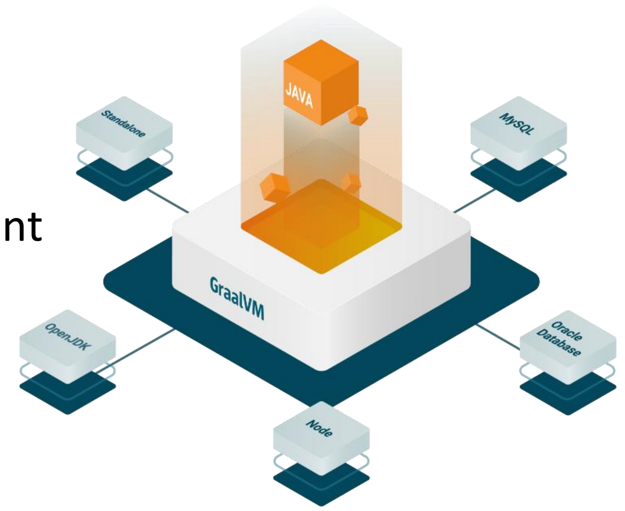
- **Background:**
  - Currently, the NG2C Garbage Collector uses the ROLP system (running in the OpenJDK JVM) to detect the age of created objects so that the NG2C allocates such objects close to each other
  - Such allocation contributes to improve GC latency while keeping a good throughput
- **Goal:**
  - Develop ROLP for the GraalVM (it is a innovative Virtual Machine that supports several languages - Java, JS, Ruby, etc.)
  - The GC must aim at micro services running on top of GraalVM
  - The development should be language agnostic
- **Requirements:**
  - The candidate must enjoy and have adequate skills to deal with systems' implementation
  - The candidate should be able to program in C++ and Java
  - Also relevant is a good tracking record (grades, classes done), enthusiasm, and commitment



- **Local:** The work will be developed at INESC ID.
- **Observations**: Possible integration into a research project with scholarship.
- **Degrees:** MEIC-A, MEIC-T, METI

LocalGraal– Locality-aware Data Structures in GraalVM

- **Background:**
  - Managed languages provide no control over data strucutre locality; thus, objects are allocated anywhere in memory
  - The NG2C (N-Generational Garbage Collector) was proposed for OpenJDK and proposes multiple allocation spaces
  - Multiple allocation spaces maximize locality (and therefore also performance) and minimize GC overhead
- **Goal:**
  - Implement multiple allocation spaces in GraalVM leveraging the tools available in the VM and compiler
    - GraalVM is a recent Virtual Machine that supports several languages - Java, JS, Ruby, etc.
  - Measure throughput and latency improvements in realistic applications
  - The development should be as language agnostic as possible
- **Requirements:**
  - The candidate must enjoy and have adequate skills to deal with systems' implementation
  - The candidate should be able to program in C++ and Java
  - Also relevant is a good tracking record (grades, classes done), enthusiasm, and commitment



- **Local:** The work will be developed at INESC ID.
- **Observations**: Possible integration into a research project with scholarship.
- **Degrees:** MEIC-A, MEIC-T, METI

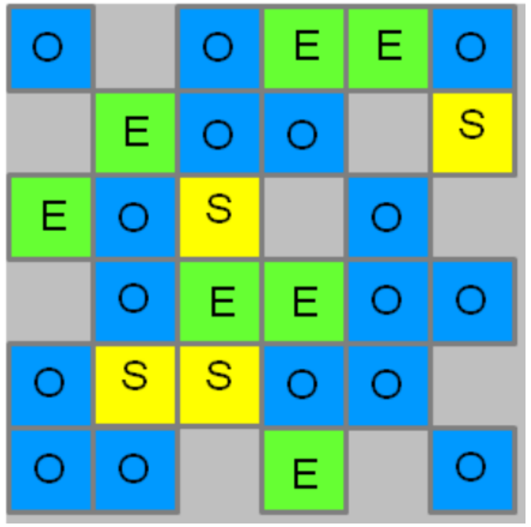
FullJavaG1 - Full Java implementation of Garbage First for GraalVM

- **Background:**
  - Garbage collectors have been historically a component implemented in low-level languages such as C/C++
  - GraalVM is making an effort to re-write components of the JVM in managed languages, which are easier to develop and debug
  - The GraalVM GC implementation is, however, very primitive
  - GraalVM supports using the C++ implementation of G1 from OpenJDK HotSpot but does not have one full-Java implementation;

- **Goals:**
  - Implement a full-Java Garbage First GC
  - The collector should have similar behavior to the one available in HotSpot
  - Implement the fundamental G1 features: heap traversal, heap regions, minor GCs, mixed GCs, full GC
  - Evaluate the performance and compare it to the current GC and to G1 implementation in HotSpot using well-known GC benchmarks.

- **Requirements:**
  - The candidate must enjoy and have adequate skills to deal with systems' implementation
  - The candidate should be able to program in C, C++, and Java
  - Also relevant is a good tracking record (grades, classes done), enthusiasm, and commitment

- **Local:** The work will be developed at INESC ID.
- **Observations**: Possible integration into a research project with scholarship.
- **Degrees:** MEIC-A, MEIC-T, METI



MagTrans – Android Magnetic Signatures of Transport Modes

- **Background:**
  - Woorti (http://www.woorti.com/) is a smartphone app (Android and iOS) that detects the transport mode that is used (https://www.youtube.com/watch?time_continue=3&v=o2E1md1t69U)
  - It transparently monitors your transport modality using sensors
  - However, sometimes Woorti is not capable of making a correct detection of the transport mode
  - It is based on a ML classifier running in the smarphone

- **Goal:**
  - Develop a module for Android based on the magnetic signatures of different transport modes
  - Improve the detection accuracy of Woorti
  - Design, implement, and evaluate the solution developed

- **Requirements:**
  - Enjoy and have adequate skills to deal with Java, Android, and mobile system issues.

- **Local:** The work will be developed at INESC ID.

- **Observations**: Possible integration into a research project with scholarship.

- **Degrees:** MEIC-A, MEIC-T, METI