2.1 Implementation

PAPI is implemented o9 top of the Java Native Interface (JNI) and MPI functions.

JNI [Lia99] is used to dynamically load and run the MPI application.
It also provides a way to allow communications between the Java side and the MPI side. Actually, the root node of the MPI application is loaded into the memory space of the JVM through the use of JNI. In this way, the Java application and the root node code of the MPI application are able to exchange data information.

MPI [For94] [GLS94] functions are used to openly pass on data inside the MPI world.

The MPI application is dynamically loaded by the Java side by the means of the `System.loadLibrary()` function as a standard shared library[3]:

The user calls the `PAPI.job()` method in order to run his/her application on the MPI side. In this way, the `PAPI_Job()` C function must embodying all user calls and data settings to run correctly the user MPI application. In this function, the call to `MPI_Init()`. Is not allowed as it