

## **Compile-Time Function Call Interception for Testing in C/C++**

**Gábor Márton, Zoltán Porkoláb**

Department of Programming Languages and Compilers, Eötvös Loránd University

`martongabesz@gmail.com`, `gsd@elte.hu`

In C/C++, during the test development process we often have to modify the public interface of a class to replace existing dependencies; e.g. supplementary setter or constructor functions or extra template parameters are added for dependency injection. These solutions may have serious detrimental effects on the code structure and sometimes on the run-time performance as well. We introduce a new technique that makes dependency replacement possible without the modification of the production code, thus it provides an alternative way to add unit tests. Our new compile-time instrumentation technique enables us to intercept function calls and replace them in runtime. Contrary to existing function call interception (FCI) methods, we instrument the call expression instead of the callee, thus we can avoid the modification and recompilation of the function in order to intercept the call. This has a clear advantage in case of system libraries and third party shared libraries, thus it provides an alternative way to automatize tests for legacy software. We created a prototype implementation based on the LLVM compiler infrastructure which is publicly available for testing.