Extending and Enhancing SAS (Static Analysis Suite)

5th September 2016

David Ho
University of Cambridge
Supervisors: Joschka Lingemann,
Benedikt Hegner

Contents

What is SAS?

- **Static analysis tool for C & C++ code**
- **Wraps around Clang**

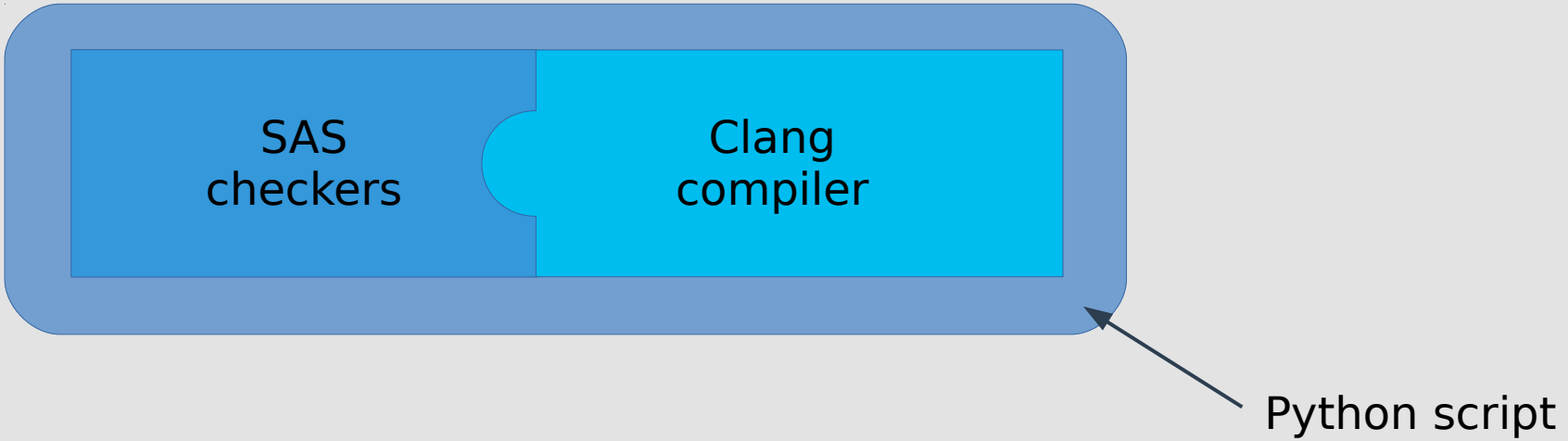
The Clang Ecosystem

- **Compiles using LLVM**
- **Produces Abstract Syntax Tree**
- **Can run plugins at compile time:**
  - clang-format
  - clang-modernize
  - Analyzers (checkers):
    - Performance, thread-safety, coding conventions

My Project

- **Enabled integration of SAS into projects via CMake**
- **Added mechanism to generate HTML report index**
- **Streamlined checker addition; implemented code generation tool**
- **Added FCCSW-specific checkers**
- **Created documentation**

Using SAS – Before

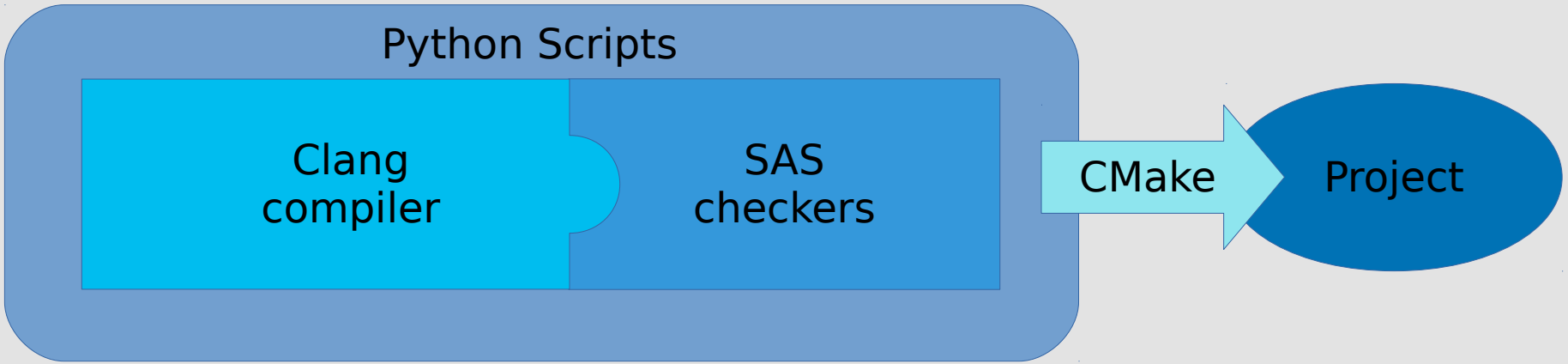


- **Custom Clang analyzers**
- **Python script steered by environment variables**
- **Scan-build reporting**

Using SAS – Before

```
clang++ <all the options to compile
the unit>  -Xclang -analyze -Xclang
-analyzer-output=text -Xclang -load
-Xclang $SASBUILDDIR/lib/libSas.so
-Xclang -analyzer-
checker=MyAnalyzer1 -Xclang
-analyzer-checker=MyAnalyzer2
-Xclang -analyzer-
checker=MyAnalyzer3...
```

Using SAS – Now



Using SAS – Now

In CMakeLists.txt:

```
find_package(sas)
enable_sas()
```

Using SAS – Now

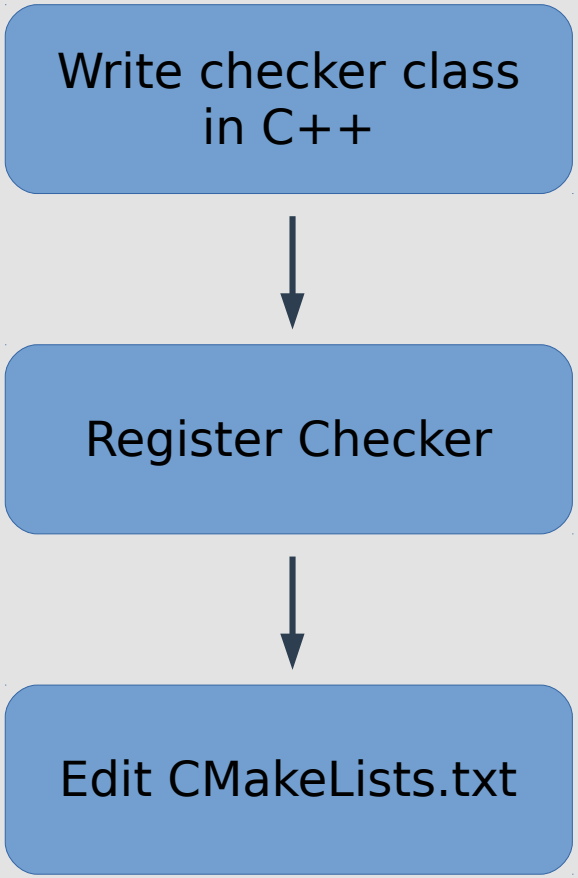
In CMakeLists.txt:

```
find_package(sas)
enable_sas([options])
```

Options include:

- Enabling/disabling checkers
- Enabling formatting/modernization checks
- Directories to ignore
- Output verbosity

Adding SAS Checkers – Before



- **Requires knowledge of Clang Abstract Syntax Tree**
- **Lots of boilerplate code**

- **More boilerplate code**

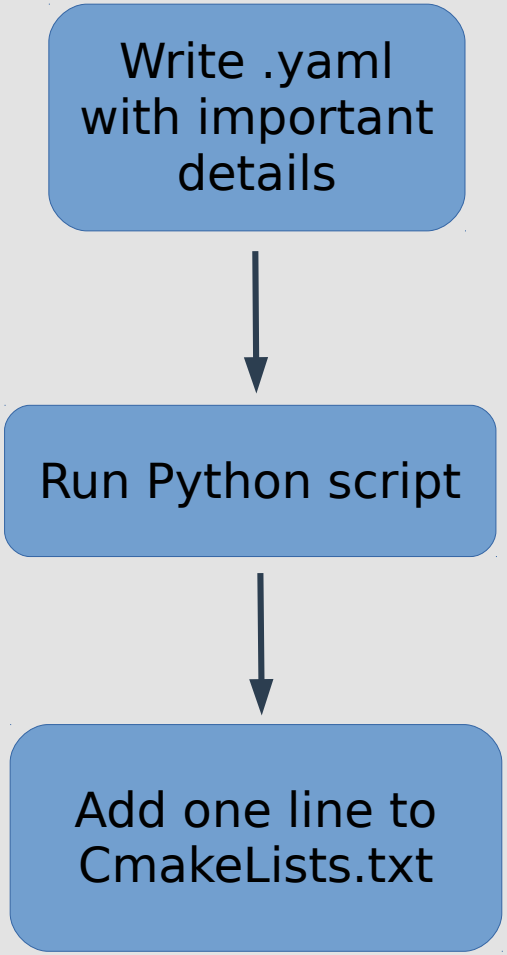
- **One big CMakeLists.txt**

Adding SAS Checkers – Before

```cpp
#include "NamespaceChecker.h"

#include <clang/StaticAnalyzer/Core/BugReporter/BugReporter.h>
#include <clang/StaticAnalyzer/Core/BugReporter/PathDiagnostic.h>
#include <clang/StaticAnalyzer/Core/Checker.h>
#include <clang/StaticAnalyzer/Core/PathSensitive/AnalysisManager.h>
#include <regex>

namespace sas {
namespace CodingConventions {
namespace FCCSW {
void NamespaceChecker::checkASTDecl(const clang::NamespaceDecl* D,
                                    clang::ento::AnalysisManager& Mgr,
                                    clang::ento::BugReporter& BR) const {
  const char *reportDescription = "[sas.CodingConventions.FCCSW.Namespace] Namespace"
                                  " names may only contain lowercase letters.";
  std::regex correctRegex("[a-z]+");
  auto nameString = D->getNameAsString();

if (!std::regex_match(nameString, correctRegex)) {
    Report(D, reportDescription, BR);
  }

}
}
}
} // end namespace sas
```

```
add_clang_plugin(libSas
   src/SasException.cpp
#    src/ClassDumper.cpp
   src/CatchAll.cpp
   src/CommentCheckerDisabler.cpp
   src/jsonxx.cc
   src/BlackWhiteListCheckerDisabler.cpp
   # New Structure
   src/CodingConventions/General/UsingNamespace.cpp
   src/CodingConventions/General/StdPrintoutsChecker.cpp
   src/CodingConventions/ROOT/RN3Checker.cpp
   src/CodingConventions/ROOT/RN4Checker.cpp
   src/CodingConventions/ROOT/RN6Checker.cpp
   src/CodingConventions/ROOT/RN9Checker.cpp
   src/CodingConventions/ROOT/RN10Checker.cpp
   src/CodingConventions/ROOT/RN11Checker.cpp
   src/CodingConventions/ROOT/RN12Checker.cpp
   src/CodingConventions/ROOT/RN13Checker.cpp
   src/CodingConventions/ROOT/PtrCastWinChecker.cpp
   src/CodingConventions/FCCSW/LocalVariableLowercaseChecker.cpp
   src/CodingConventions/FCCSW/ClassNameUppercaseChecker.cpp
   src/CodingConventions/FCCSW/DataMemberChecker.cpp
   src/ThreadSafety/ConstCastChecker.cpp
   src/ThreadSafety/ConstCastAwayChecker.cpp
   src/ThreadSafety/GlobalStaticChecker.cpp
   src/ThreadSafety/StaticLocalChecker.cpp
   src/ThreadSafety/MutableMemberChecker.cpp
   src/Performance/ArgSizeChecker.cpp
   src/Performance/FiniteMathChecker.cpp
   src/Example/VarnameChecker.cpp
   src/ClangSasCheckerPluginRegister.cpp
#   src/GlobalAccInCtorChecker.cpp
      # Add your checker here!
)
```

Adding SAS Checkers – Now




```
Namespace:
    name: "Namespace"
    regex: "[a-z]+"
    description: "Namespace names may
                  only contain lowercase letters."
```

```
add_checker(FunctionChecker.cpp)
add_checker(DataMemberChecker.cpp)
add_checker(NamespaceChecker.cpp)
add_checker(ConstChecker.cpp)
add_checker(EnumeratorChecker.cpp)
add_checker(TypeChecker.cpp)
```

Current State of SAS

- **Developed with testing on project with relative small C++ code base (podio)**
- **Run on FCCSW using Jenkins**
  - *FCCSW Report Index*
- **Run on ROOT for scalability testing**

Summary

- **SAS is:**
  - More usable
  - Easy to integrate
  - Scalable
- **In the future:**
  - Filtering of external sources for continuous integration
  - Improve report index
  - Usability testing

Give it a try!

- **Clone from github:**

https://github.com/dpiparo/SAS

- **Documentation in markdown in repository**

Thank you for listening

Any Questions?