Global code completion and architecture of clangd

Ilya Biryukov
Google

EuroLLVM
April 16, 2018

Overview

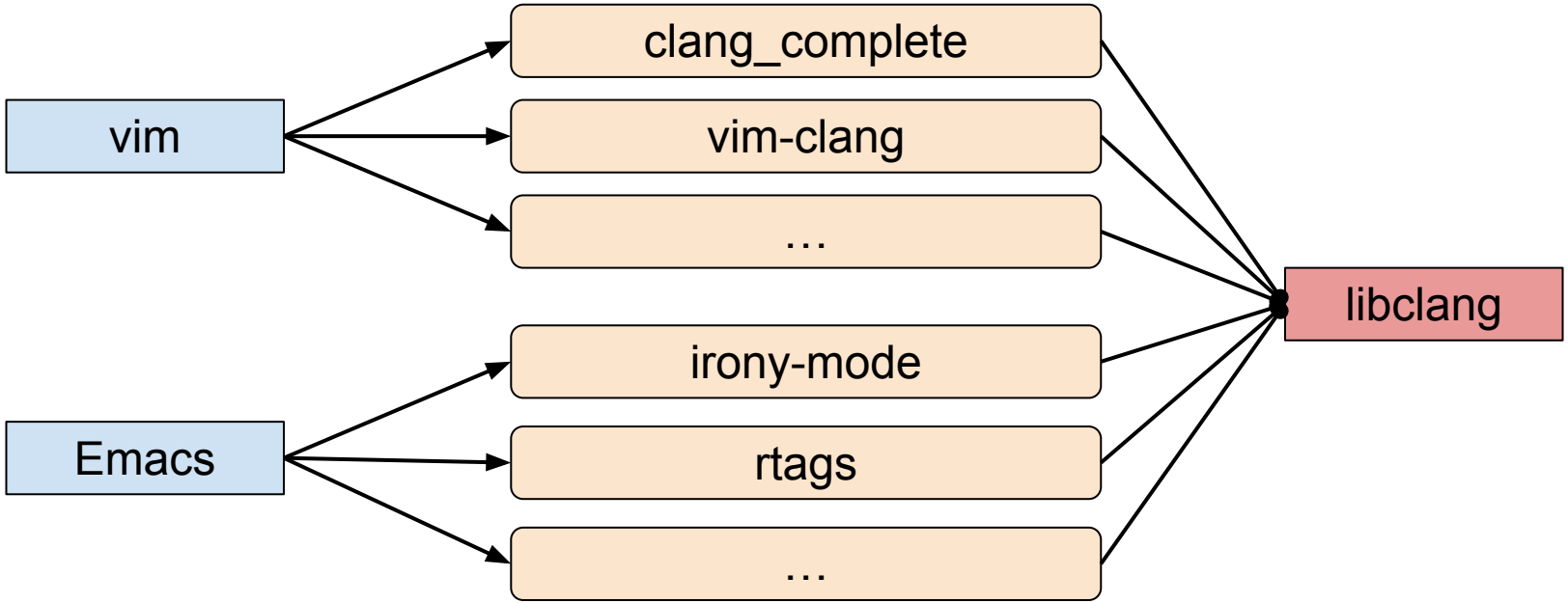
1. Language Server Protocol
2. Architecture of clangd
3. Code completion
4. Current state and future work

Language Server Protocol

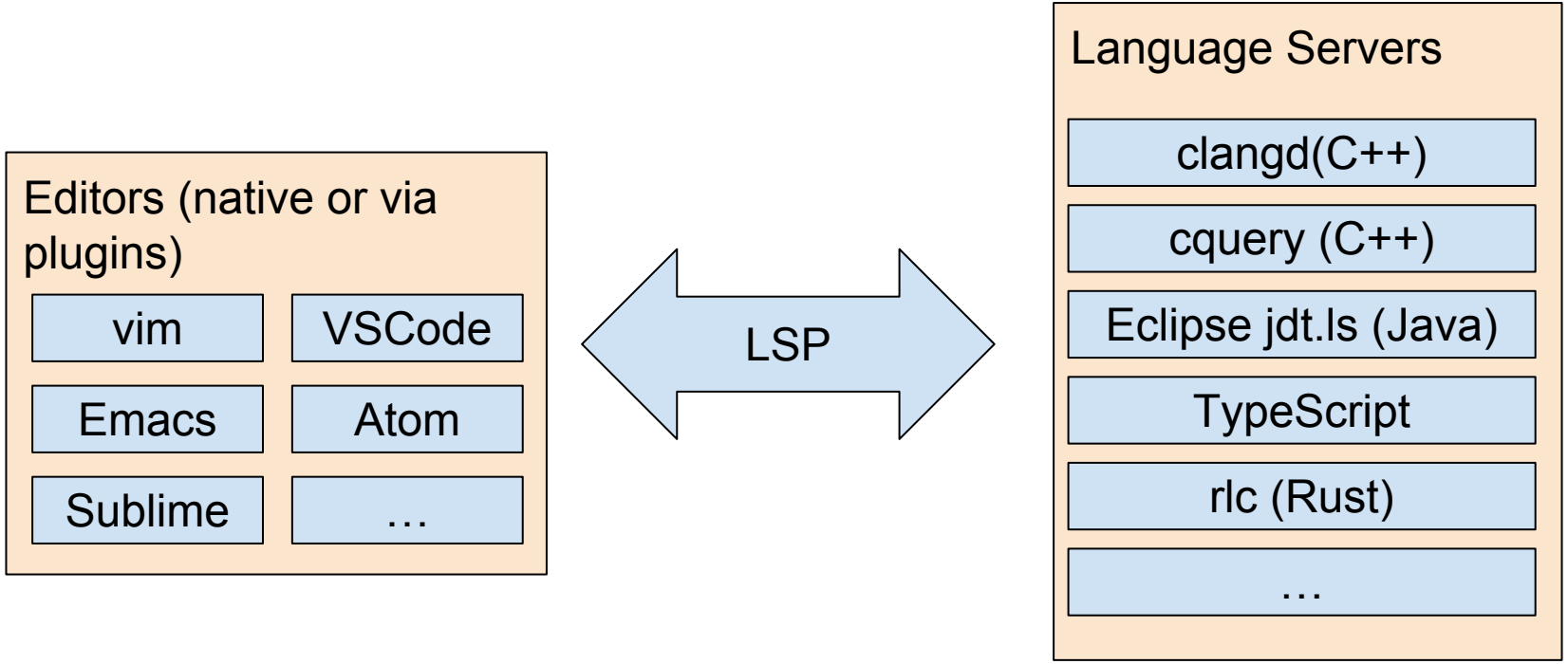
Editor integration problem

	C++	Java	C#
Vim	C++ Vim plugin	Java Vim plugin	C# Vim plugin
Emacs	C++ Emacs plugin	Java Emacs plugin	C# Emacs plugin
Atom	C++ Atom plugin	Java Atom plugin	C# Atom plugin
Sublime Text	C++ Sublime plugin	Java Sublime plugin	C# Sublime plugin

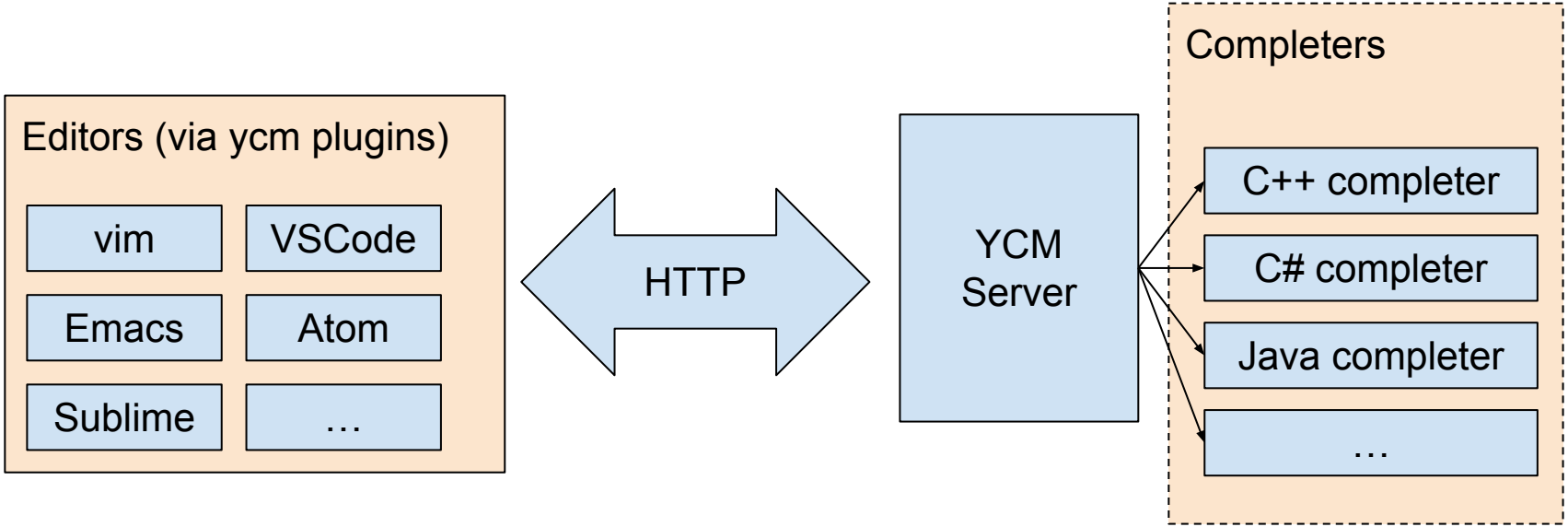
Editor integration problem



Language Server Protocol (LSP)



YouCompleteMe



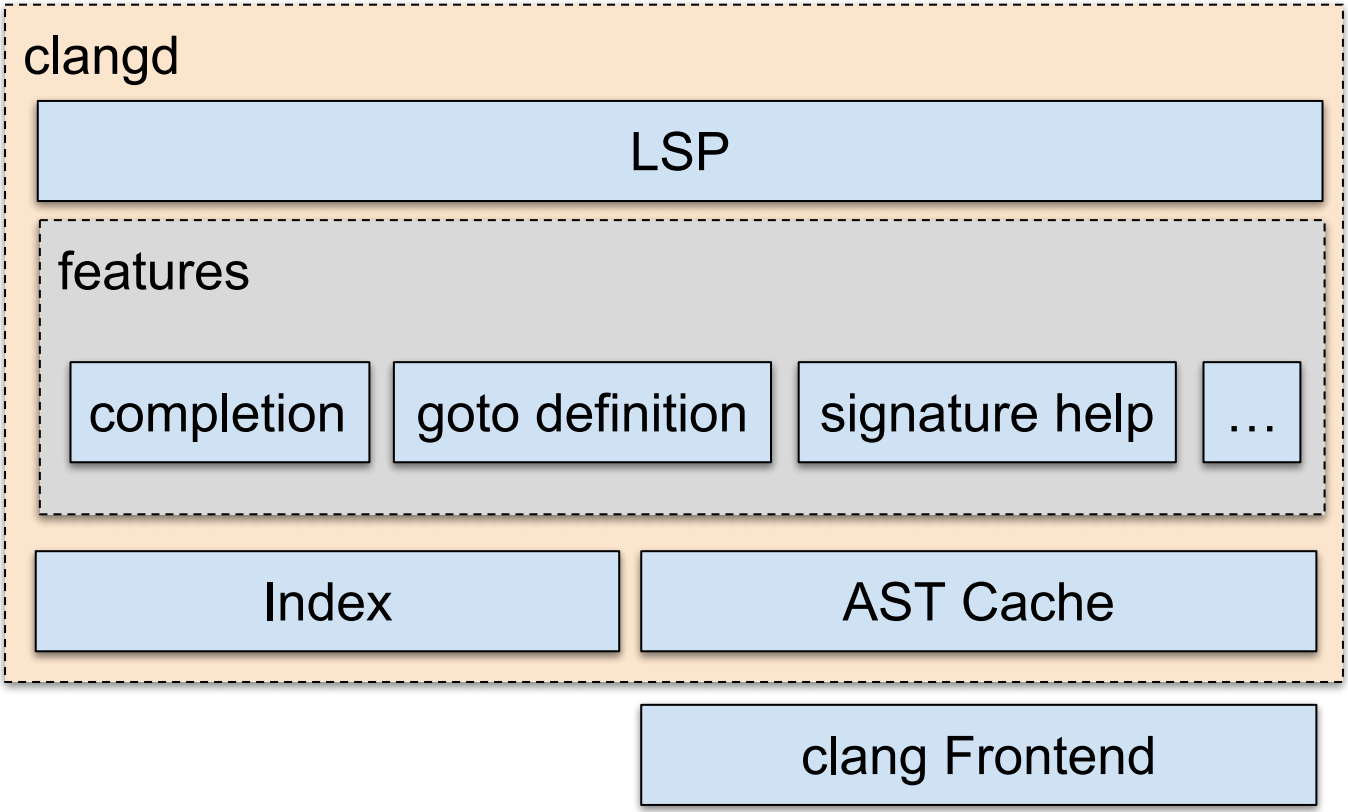
clangd

clangd

Language Server based on clang.

- Developed as part of LLVM in clang-tools-extra
- Supported features
  - Diagnostics
  - Code completion
  - Go to definition
  - Documentation
  - Local rename
  - Formatting
  - …

Architecture of clangd



Index

- Provides project-wide information
- AST **Decl** → Index **Symbol**
- Example:

```
namespace llvm {
  class StringRef { /* ... */ };
}
int dropFirst(llvm::StringRef S, size_t N);
```

Name: StringRef
Scope: llvm
Kind: class
ID: llvm::StringRef
…

Name: dropFirst
Scope: <global-namespace>
Kind: function
ID: dropFirst(llvm::StringRef, size_t)
…

Models of your program

AST

- Single file
- Up-to-date

Index

Dynamic

- Active files
- ~1 minute old

Static

- All files
- ~1 day old
- external?

Code completion

Clang completion of class members.

GlobalSymbolBuilderMain.cpp

```
1  #include "llvm/ADT/StringRef.h"
2
3  llvm::StringRef dropFirst(llvm::StringRef S, int N) {
4    S.
5  }
```

```
consume_back(StringRef Suffix)                    bool
consume_front(StringRef Prefix)
back() const
begin() const
bytes() const
bytes_begin() const
bytes_end() const
compare(StringRef RHS) const
compare_lower(StringRef RHS) const
compare_numeric(StringRef RHS) const
consumeInteger(unsigned int Radix, T &Result)
consumeInteger(unsigned int Radix, T &Result)
```

Clang completion inside namespaces.

GlobalSymbolBuilderMain.cpp

```
1  #include "llvm/ADT/StringRef.h"
2
3  llvm::StringRef dropFirst(llvm::StringRef S, int N) {
4    std::vecto
5  }
```

vector<class _Tp>
has_virtual_destructor<class _Tp>

Clang completion inside namespaces.

GlobalSymbolBuilderMain.cpp

```
1  #include "llvm/ADT/StringRef.h"
2
3  llvm::StringRef dropFirst(llvm::StringRef S, int N) {
4    std::map
5  }
```

```
make_pair(_T1 &&__t1, _T2 &&__t2)  pair<typename __m…
make_tuple(_Tp &&__t...)
make_heap(_RandomAccessIterator __first, _RandomAccessI
make_heap(_RandomAccessIterator __first, _RandomAccessI
make_exception_ptr(_Ep __e)
```

Idea of global completion

ClangdServer.cpp

```
1  int main() {
2    std::
3  }
```

Idea of global completion

ClangdServer.cpp

```
1  int main() {
2    std::vec
3  }
```

- vector
- __vector_base
- __vector_base_common
- __map_value_compare
- has_virtual_destructor

Idea of global completion

ClangdServer.cpp

```
1  #include <vector>
2
3  int main() {
4    std::vector
5  }
```

Global completion

- Completion returns results from **all** headers
  - adds #include if needed
- General case is hard

```
std::vector<bool>
vec_bool.flip(); // ok

std::vector<int> vec_int;
vec_int.flip(); // error
```

Completion items: AST or Index?

```
namespace llvm {
int getAsUnsignedInteger(const char *);

class StringRef {
  const char *Ptr;

public:
  StringRef substr(size_t N) {
    const char *NewPtr = Ptr + N;
    |
  }
```

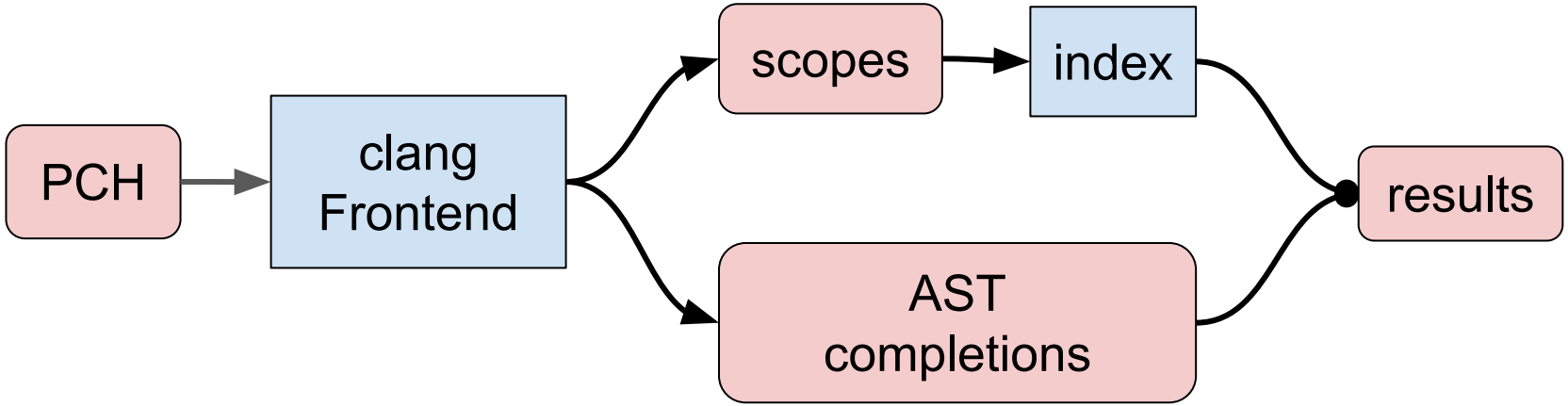
llvm::Optional

llvm::DenseSet

llvm::ArrayRef

…

Code completion flow



Scopes

```
#include "my_class.h"

using namespace std;

void my_class::do_work() {
  vec|
}
```

```
Index.fuzzyFind(/*Scopes*/ {"::", "std::"}, /*Query*/ "vec");
```

Ranking

- Query match score
- Symbol signals
  - Uses
  - Type information
  - Proximity

Ranking example

Query: “uniq”

Symbol	Match		Uses		Total
unique	0.9	×	0.3	=	0.27
unique_ptr	0.9		1.0		0.9
make_**uniq**ue	0.5		0.6		0.30
Make**Uniq**ue	0.5		0.7		0.35

Ranking example

Query: “uniq”

Symbol	Total
unique_ptr	0.9
Make**Uniq**ue	0.35
make_**uniq**ue	0.30
unique	0.27

Current state and future work

Current state

- Basic LSP support
  - completion, diagnostics, fix-its, signature help, local rename, go to definition, formatting
- Global completion (experimental)
- Project-wide index (experimental)

Embedding in a Web IDE

- Used by internal Web IDE at Google
- Runs in Cloud
- Implementation challenges:
  - Virtual File System
  - Buildsystem integration
  - Tracing requests

cquery

- Language Server based on libclang
- More features than clangd
- Highly optimized in-memory index
- Designed to run locally

Future plans

- More LSP features
- clang-tidy
- index-while-build
- Better build system integration

Thanks to all contributors!

- Ben Jackson
- Benjamin Kramer
- Eric Liu
- Haojian Wu
- Krasimir Georgiev
- Marc-André Laperle
- Raoul Wols
- Sam McCall
- Simon Marchi
- Stanislav Ionascu
- William Enright
- …

- Feedback welcome!
  - https://clang.llvm.org/extra/clangd.html
- Contributions welcome!
  - https://reviews.llvm.org/source/CTE/browse/clang-tools-extra/trunk/clangd/
  - https://clang.llvm.org/extra/clangd.html#getting-involved
- Reach us at “cfe-dev”