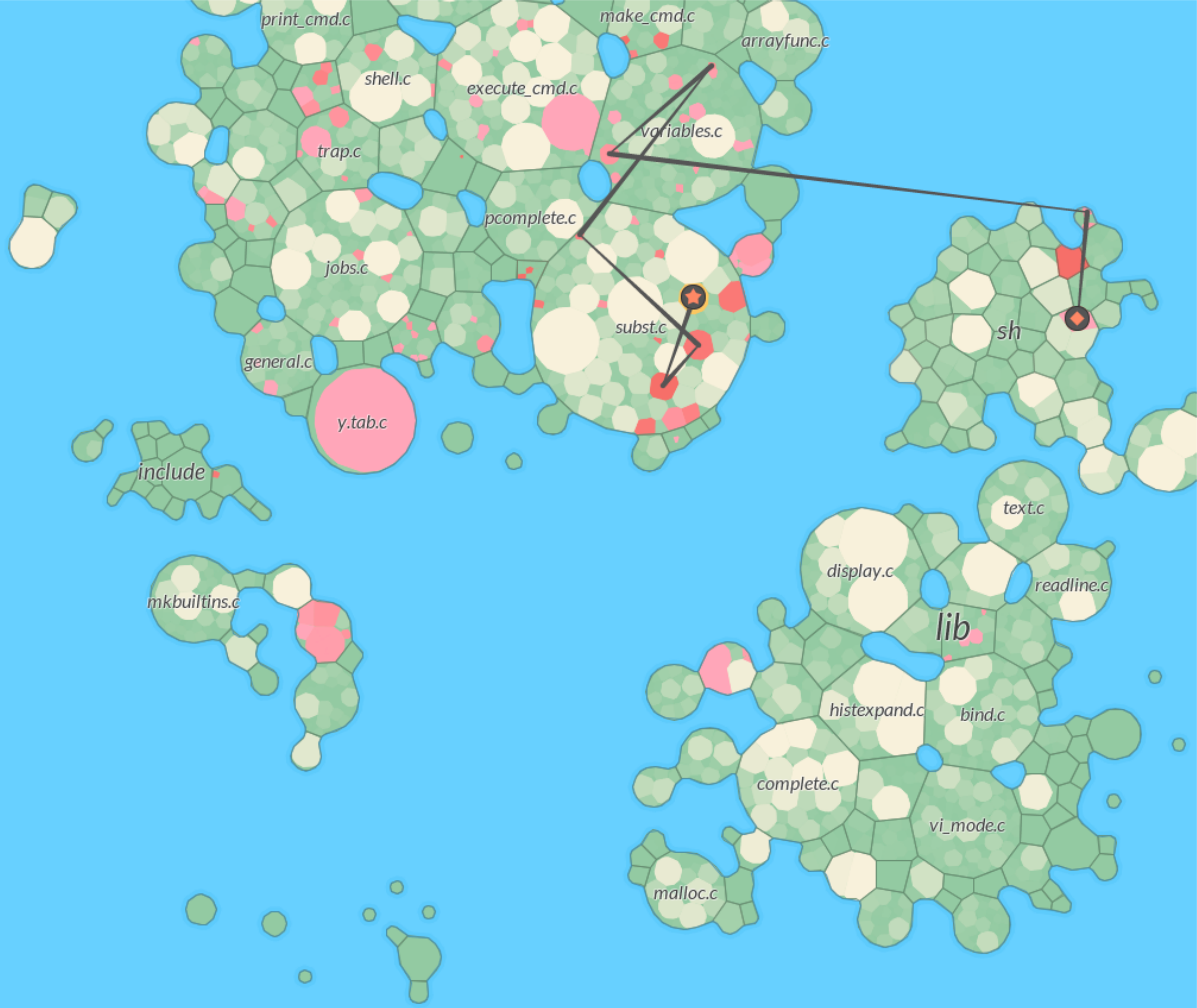
Frappé

Using Clang to Visualize Large Codebases



Nathan Hawes and Ben Barham
Oracle Labs Australia
October 2014

ORACLE

| Frappé: a code comprehension tool for large codebases

Safe Harbour

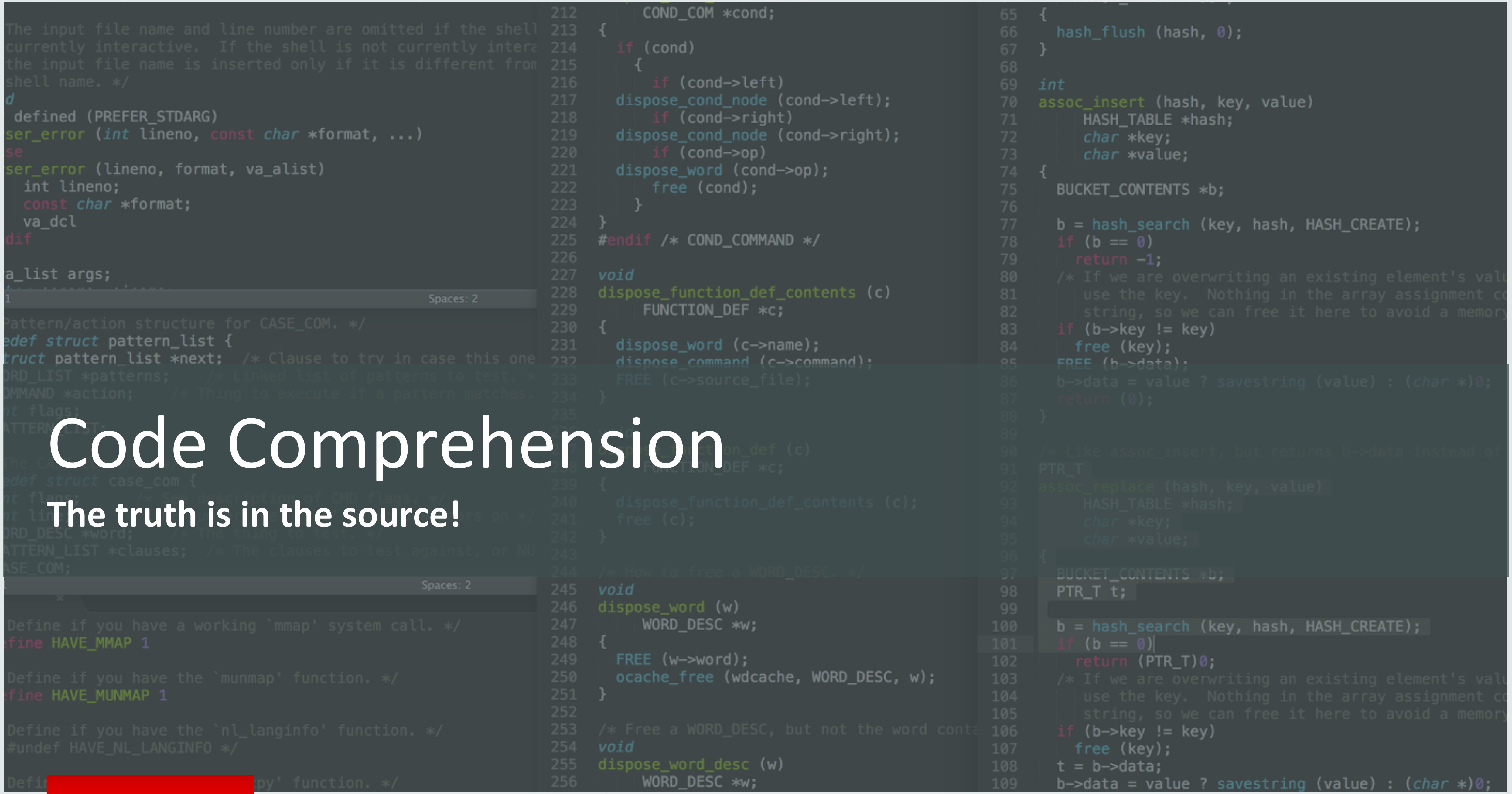
The following is intended to provide some insight into a line of research in Oracle Labs. It is intended for information purposes only, and may not be incorporated into any contract.

It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. Oracle reserves the right to alter its development plans and practices at any time, and the development, release, and timing of any features or functionality described in connection with any Oracle product or service remains at the sole discretion of Oracle.

Any views expressed in this presentation are my own and do not necessarily reflect the views of Oracle.

Code Comprehension

The truth is in the source!



```
The input file name and line number are omitted if the shell
currently interactive.  If the shell is not currently intera
the input file name is inserted only if it is different from
shell name. */
d
 defined (PREFER_STDARG)
ser_error (int lineno, const char *format, ...)
se
ser_error (lineno, format, va_alist)
  int lineno;
  const char *format;
  va_dcl
dif

a_list args;
```

```
Pattern/action structure for CASE_COM. */
edef struct pattern_list {
truct pattern_list *next;   /* Clause to try in case this one
ORD_LIST *patterns;     /* Linked list of patterns to test. *
OMMAND *action;     /* Thing to execute if a pattern matches.
nt flags;
ATTERN_LIST;

The C
edef struct case_com {
nt flags;       /* See description of CMD flags. */
nt li                                    rs on */
ORD_DESC *word;     /* The thing to test. */
ATTERN_LIST *clauses;   /* The clauses to test against, or NU
ASE_COM;
```

```
Define if you have a working `mmap' system call. */
fine HAVE_MMAP 1

Define if you have the `munmap' function. */
fine HAVE_MUNMAP 1

Define if you have the `nl_langinfo' function. */
#undef HAVE_NL_LANGINFO */

Defi                py' function. */
```

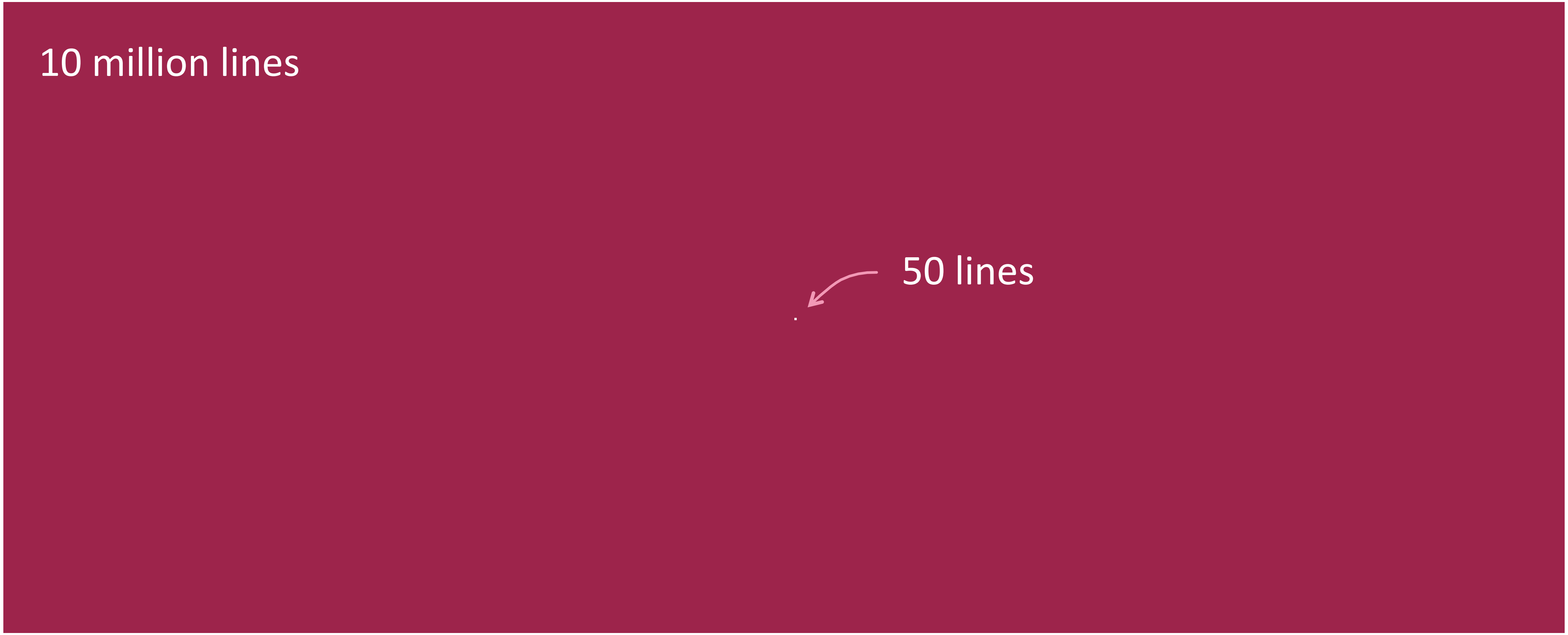
```
     COND_COM *cond;
{
  if (cond)
    {
      if (cond->left)
  dispose_cond_node (cond->left);
      if (cond->right)
  dispose_cond_node (cond->right);
      if (cond->op)
  dispose_word (cond->op);
      free (cond);
    }
}
#endif /* COND_COMMAND */

void
dispose_function_def_contents (c)
     FUNCTION_DEF *c;
{
  dispose_word (c->name);
  dispose_command (c->command);
  FREE (c->source_file);
}

void
dispose_function_def (c)
     FUNCTION_DEF *c;
{
  dispose_function_def_contents (c);
  free (c);
}

/* How to free a WORD_DESC. */
void
dispose_word (w)
     WORD_DESC *w;
{
  FREE (w->word);
  ocache_free (wdcache, WORD_DESC, w);
}

/* Free a WORD_DESC, but not the word conta
void
dispose_word_desc (w)
     WORD_DESC *w;
```

```
{
  hash_flush (hash, 0);
}

int
assoc_insert (hash, key, value)
     HASH_TABLE *hash;
     char *key;
     char *value;
{
  BUCKET_CONTENTS *b;

  b = hash_search (key, hash, HASH_CREATE);
  if (b == 0)
    return -1;
  /* If we are overwriting an existing element's valu
     use the key.  Nothing in the array assignment co
     string, so we can free it here to avoid a memory
  if (b->key != key)
    free (key);
  FREE (b->data);
  b->data = value ? savestring (value) : (char *)0;
  return (0);
}

/* Like assoc_insert, but returns b->data instead of
PTR_T
assoc_replace (hash, key, value)
     HASH_TABLE *hash;
     char *key;
     char *value;
{
  BUCKET_CONTENTS *b;
  PTR_T t;

  b = hash_search (key, hash, HASH_CREATE);
  if (b == 0)
    return (PTR_T)0;
  /* If we are overwriting an existing element's valu
     use the key.  Nothing in the array assignment co
     string, so we can free it here to avoid a memory
  if (b->key != key)
    free (key);
  t = b->data;
  b->data = value ? savestring (value) : (char *)0;
```

But what if that source is large?

10 million lines

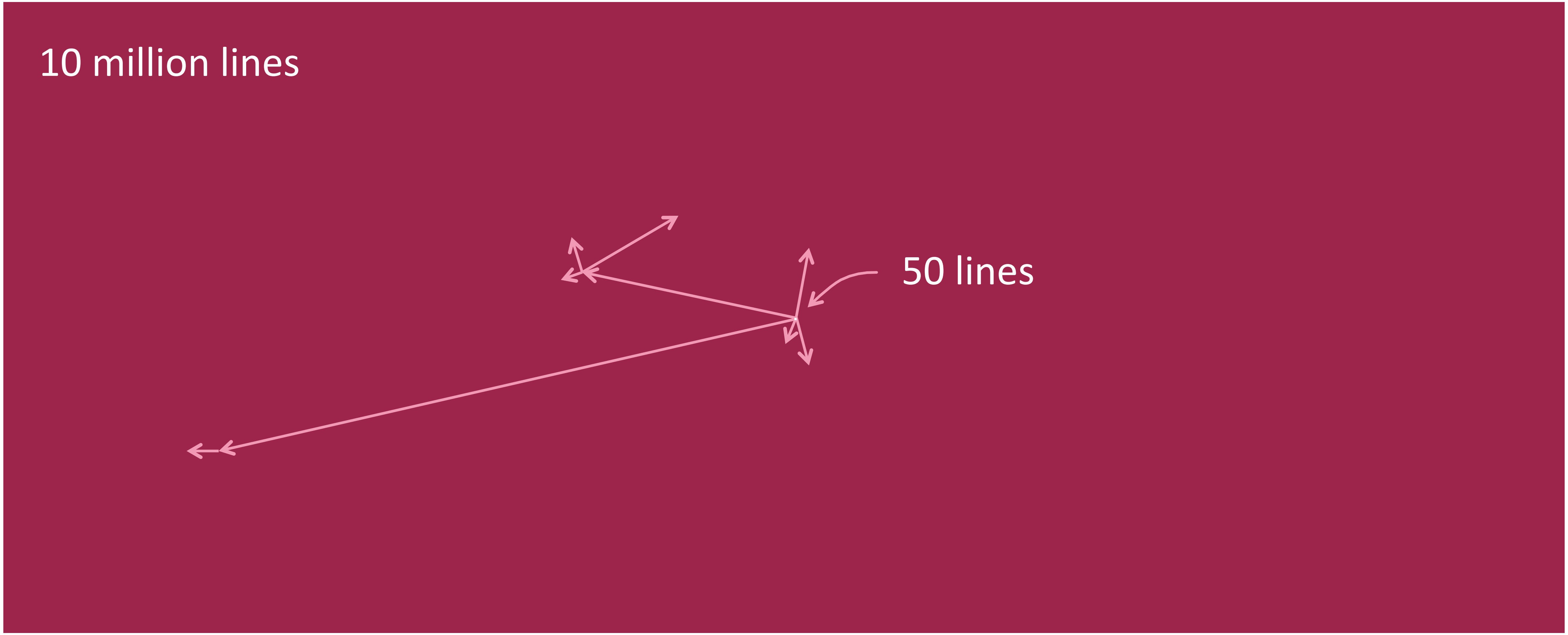
But what if that source is large?



10 million lines

50 lines

But what if that source is large?



Code Comprehension in IDEs

- Go to definition, find uses, class overview, type hierarchy, etc.

- IDEs impractical to use for large C/C++ codebases
  - Imprecise language recognition
  - Issues with custom build systems

Current Practice

For large C/C++ codebases

- Text editors and text-search tools
  - vim, emacs
  - grep, sed, cscope
- Fast and simple
- But imprecise →
  - Symbol types, scopes, linking information, preprocessor
- Low-level focus

```
static VALUE mnew(…) {
    …
    data->id = rid;
    …
}
```

Find definition

```
  method.h:70
  node.h:244
  thread_pthread.c:594
  (+ 17 more)
```

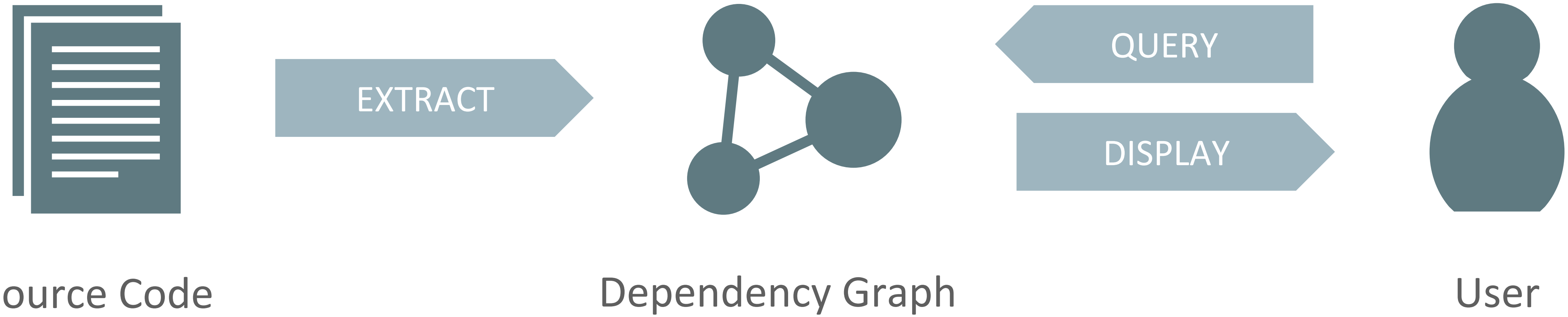
Actual definition (14th)

```
  proc.c:21
```

Frappé Aims

- Provide precise dependency information
  - With easy build integration
- Allow users to specify higher-level queries directly
  - Not just defs or refs
- Show users the broader context of the system

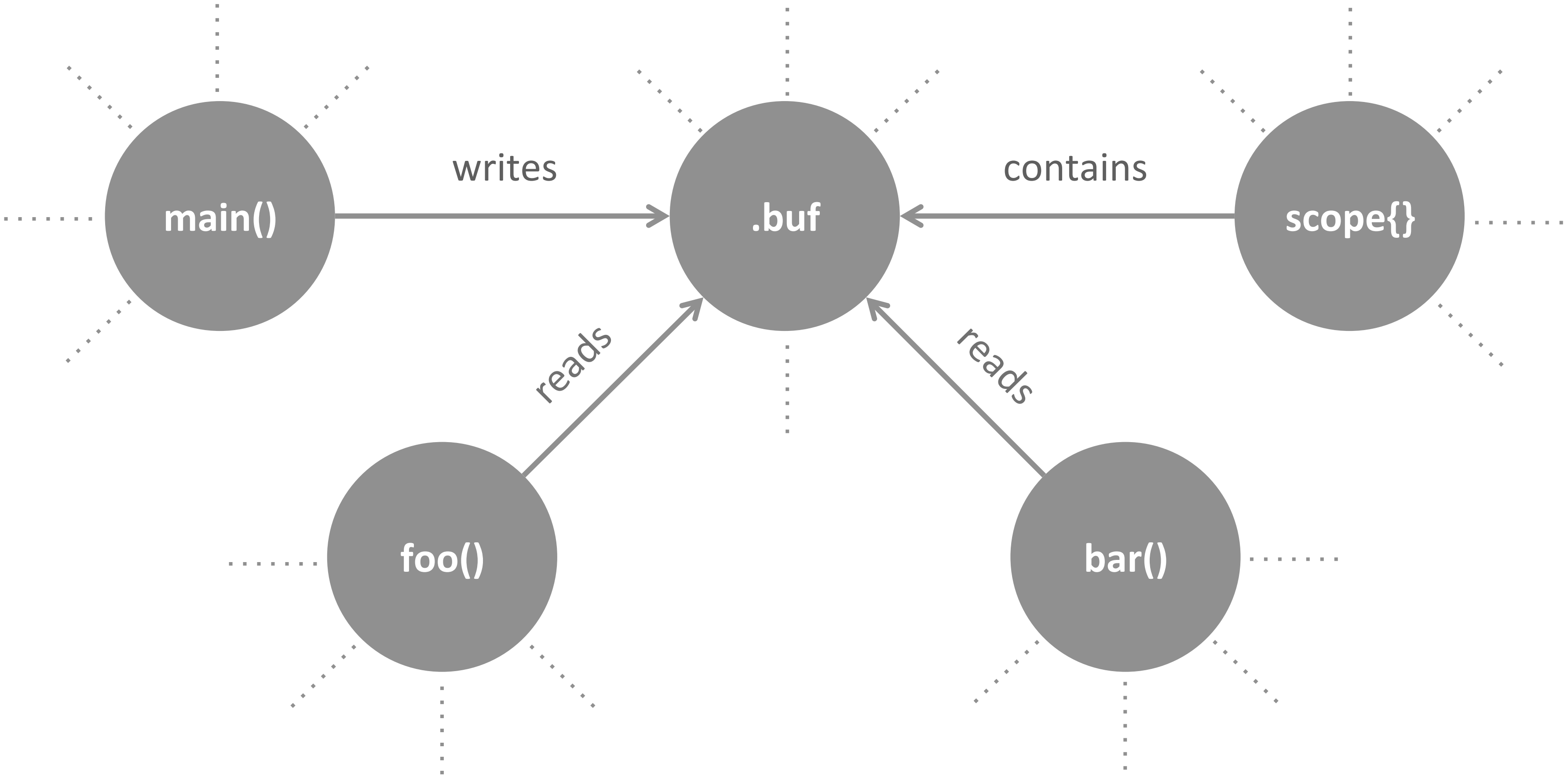
Frappé Overview



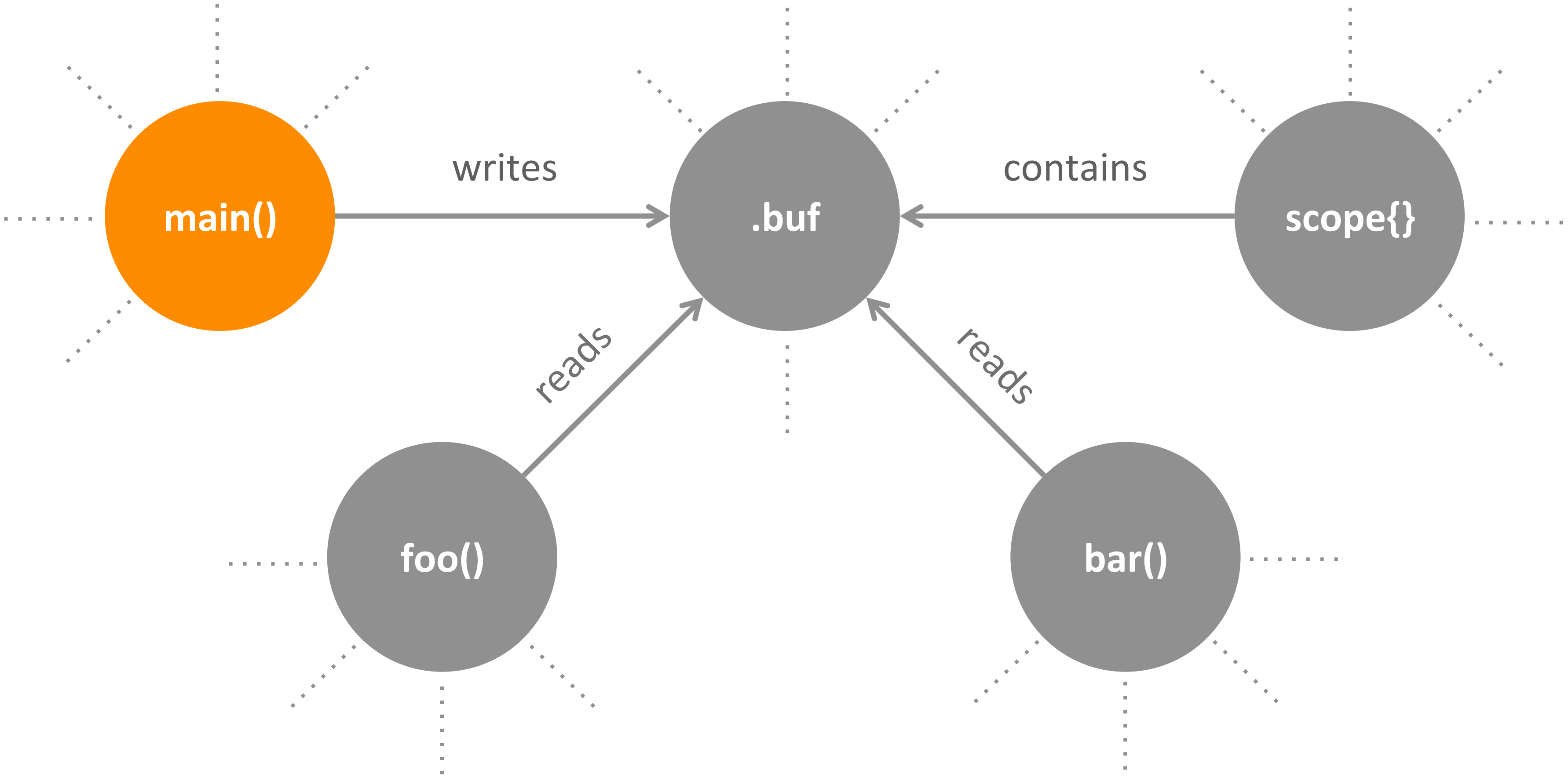
Dependency Graph

- Natural representation of the code
  - Call graphs, type hierarchies, control flow graphs, etc.
- Nodes and edges
  - Build system: modules, files, and linking information between them
  - File system: directories and files
  - Preprocessor: includes, macros, their expansion and interrogations
  - Symbols: functions, locals, types, and relations between them
- High-level questions become graph queries

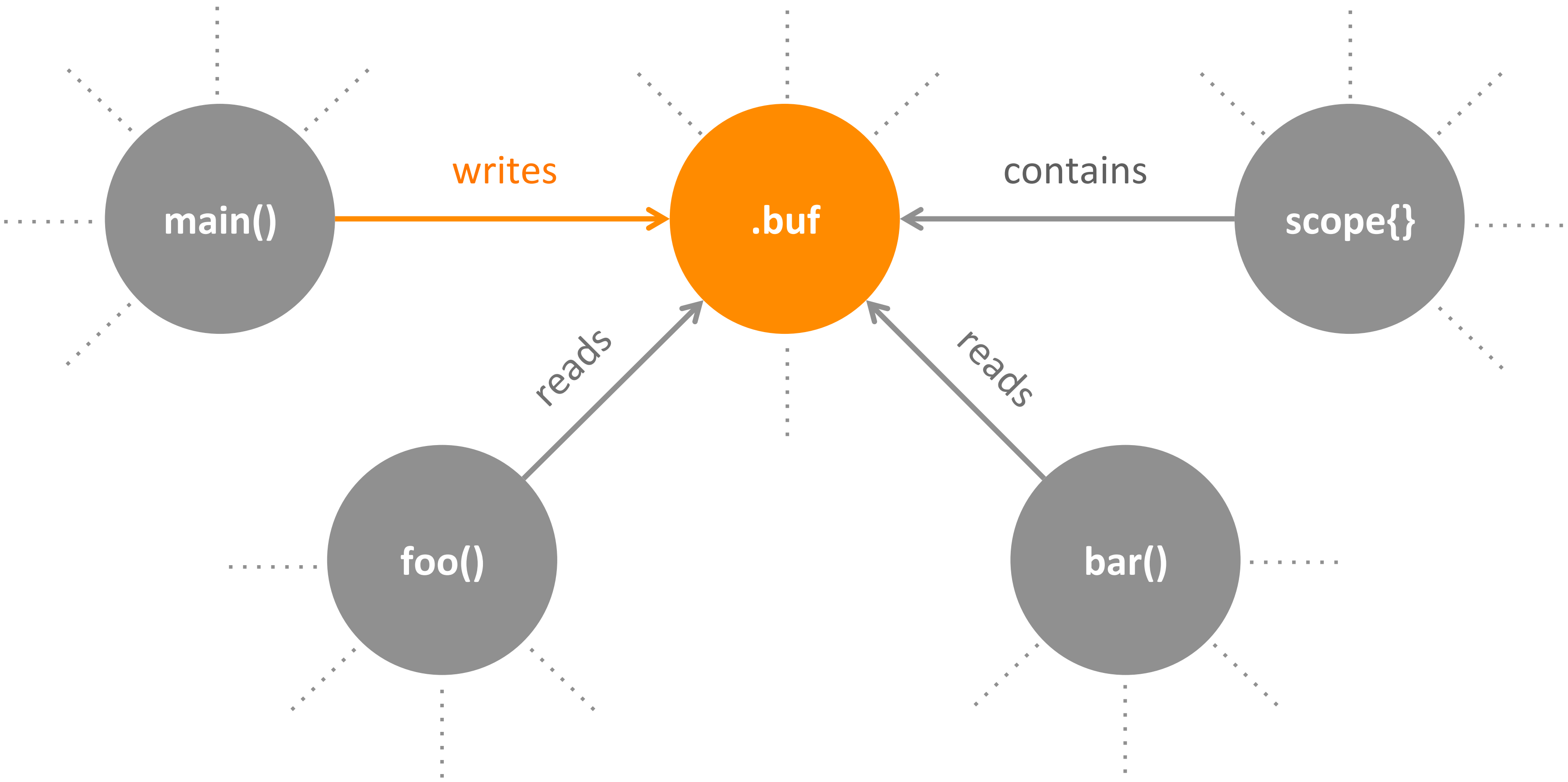
Go to Definition



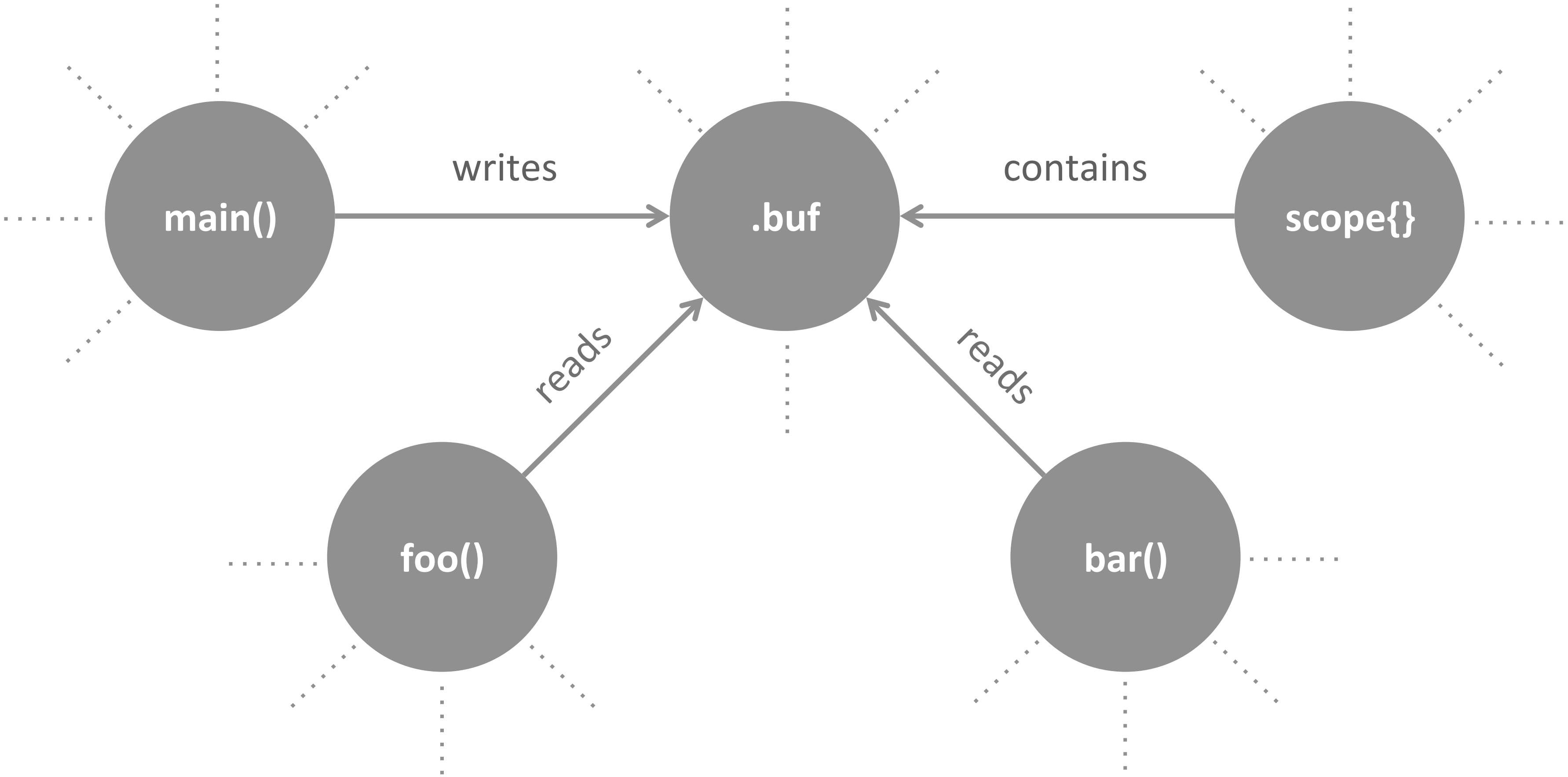
Go to Definition



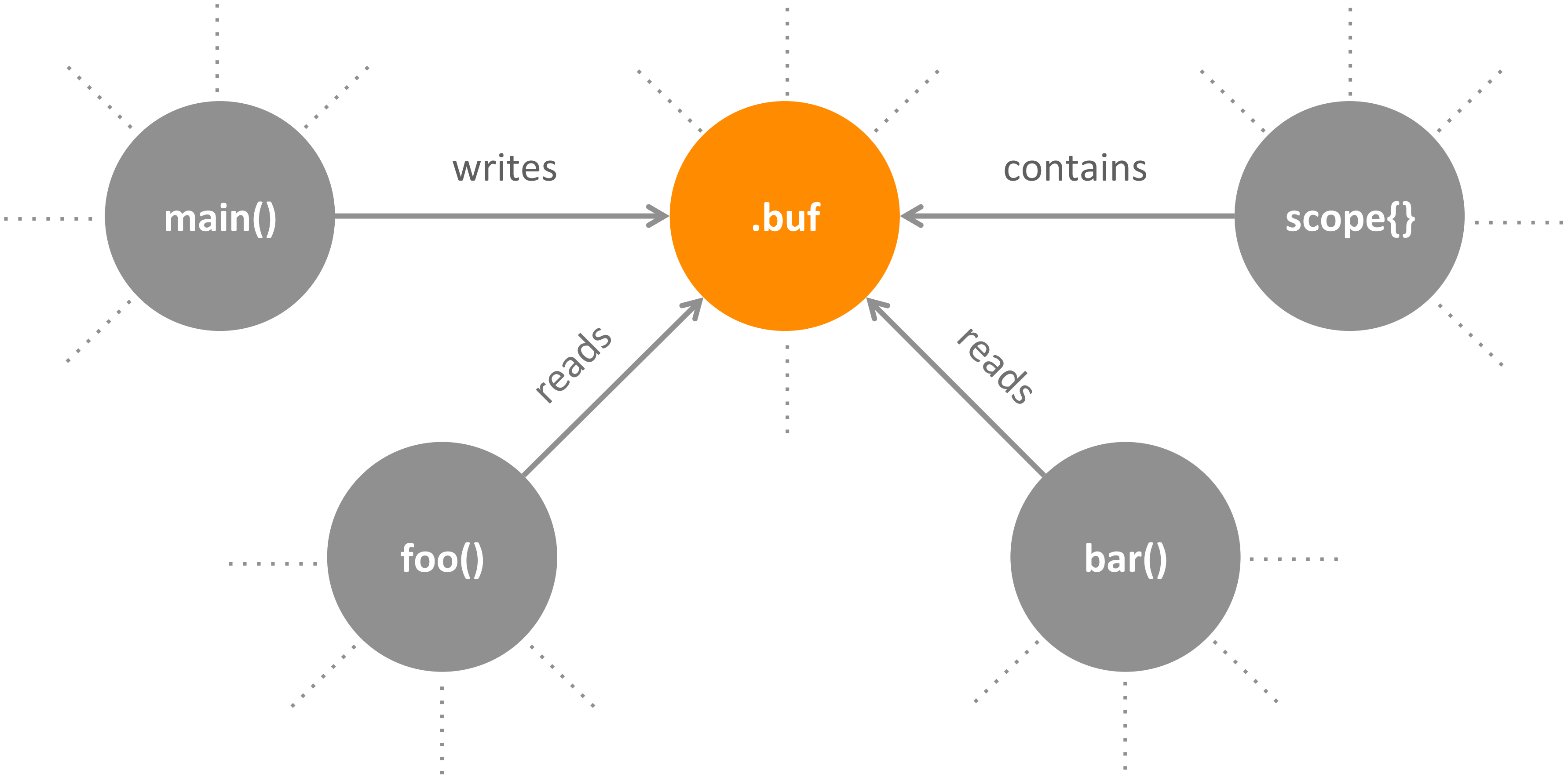
Go to Definition



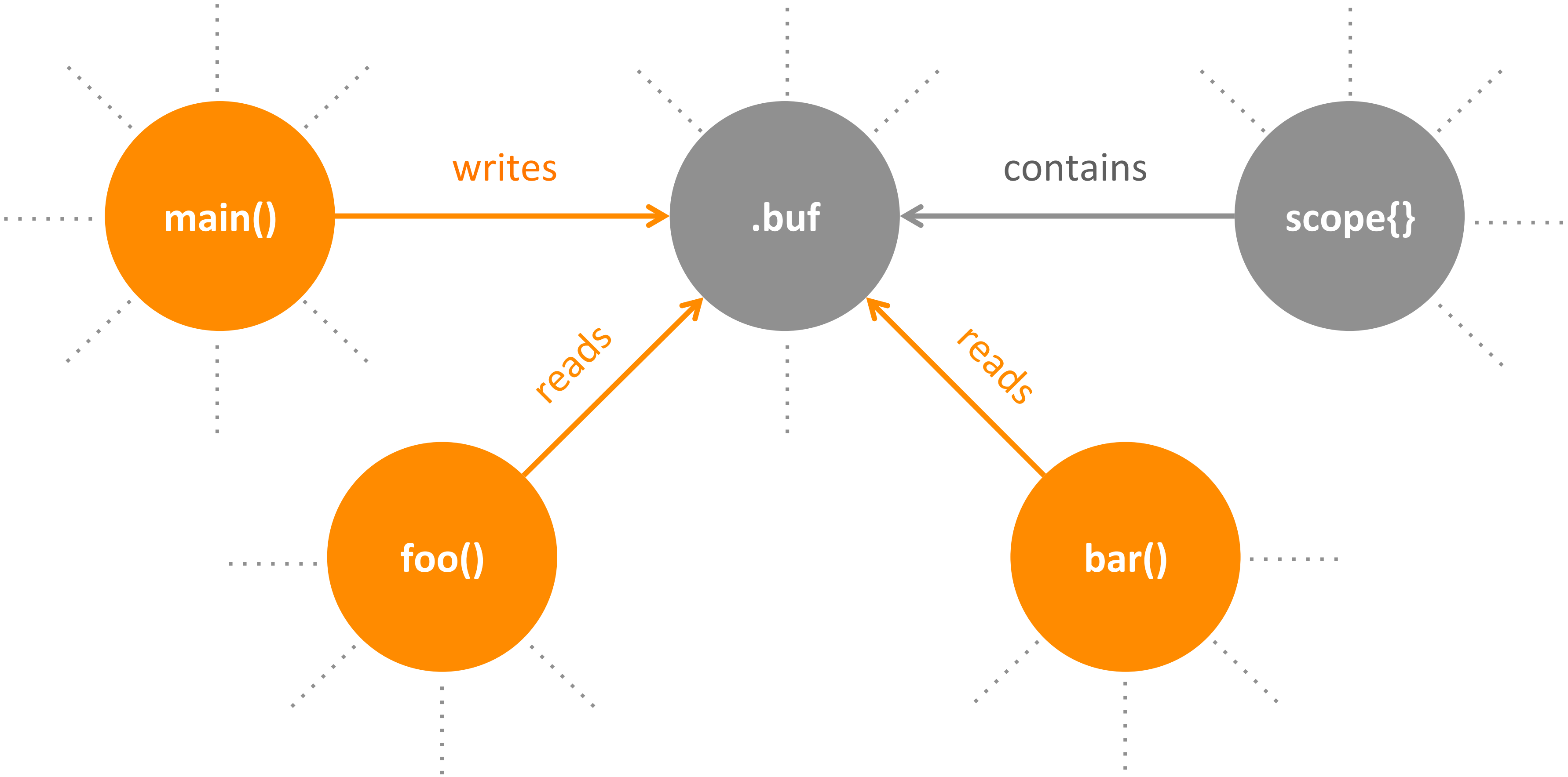
Find References



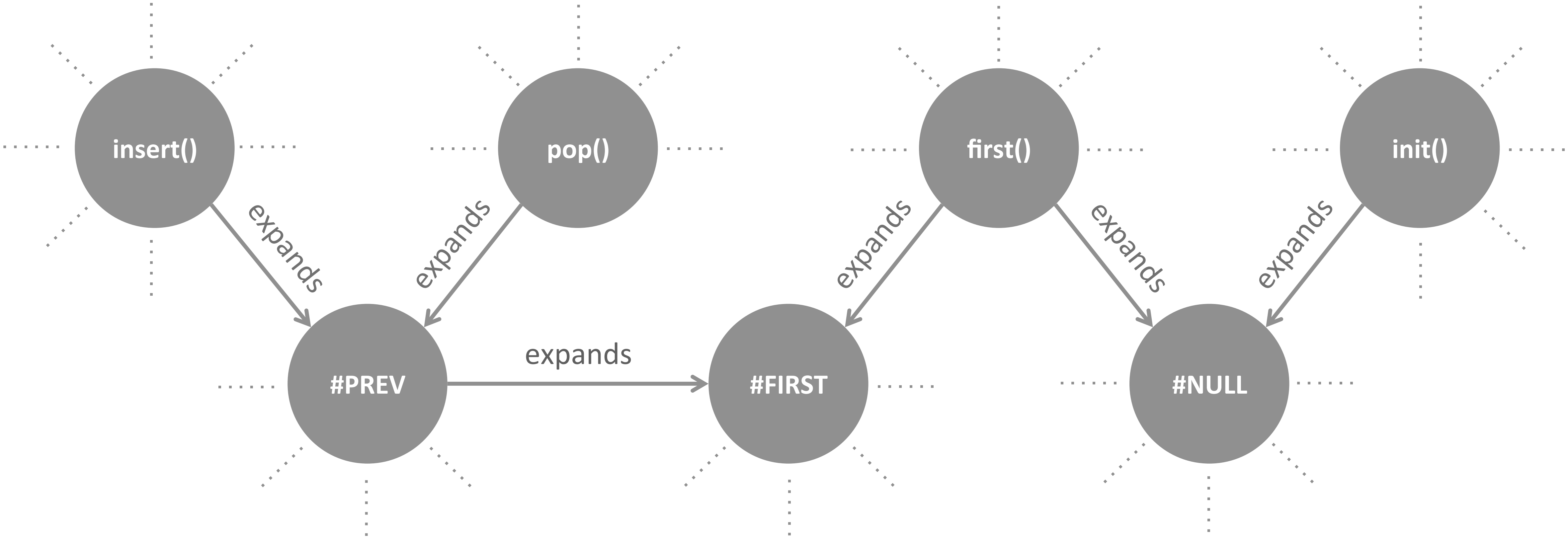
Find References



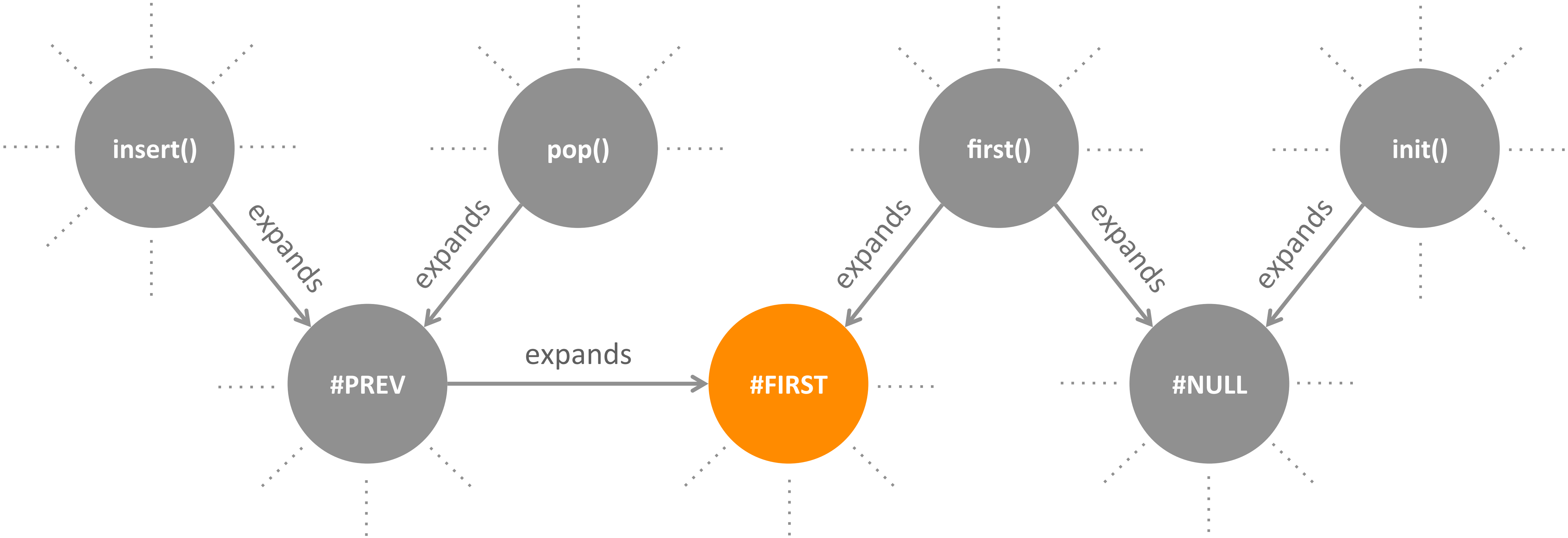
Find References



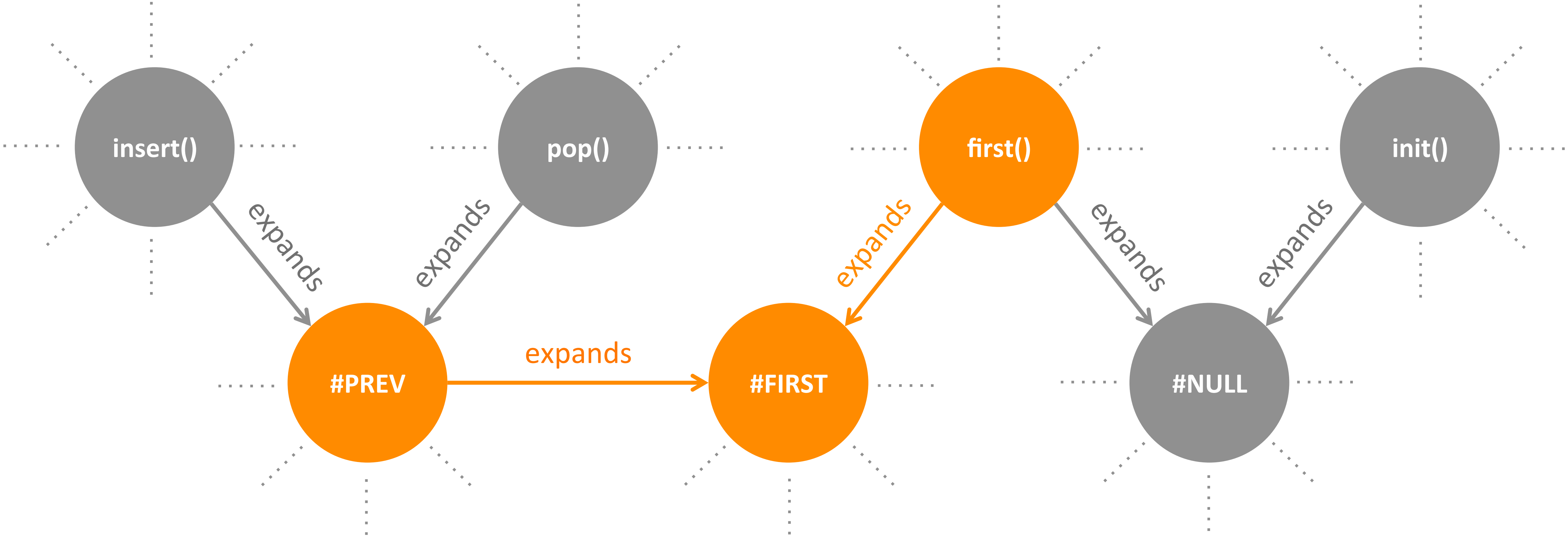
Impact Estimation



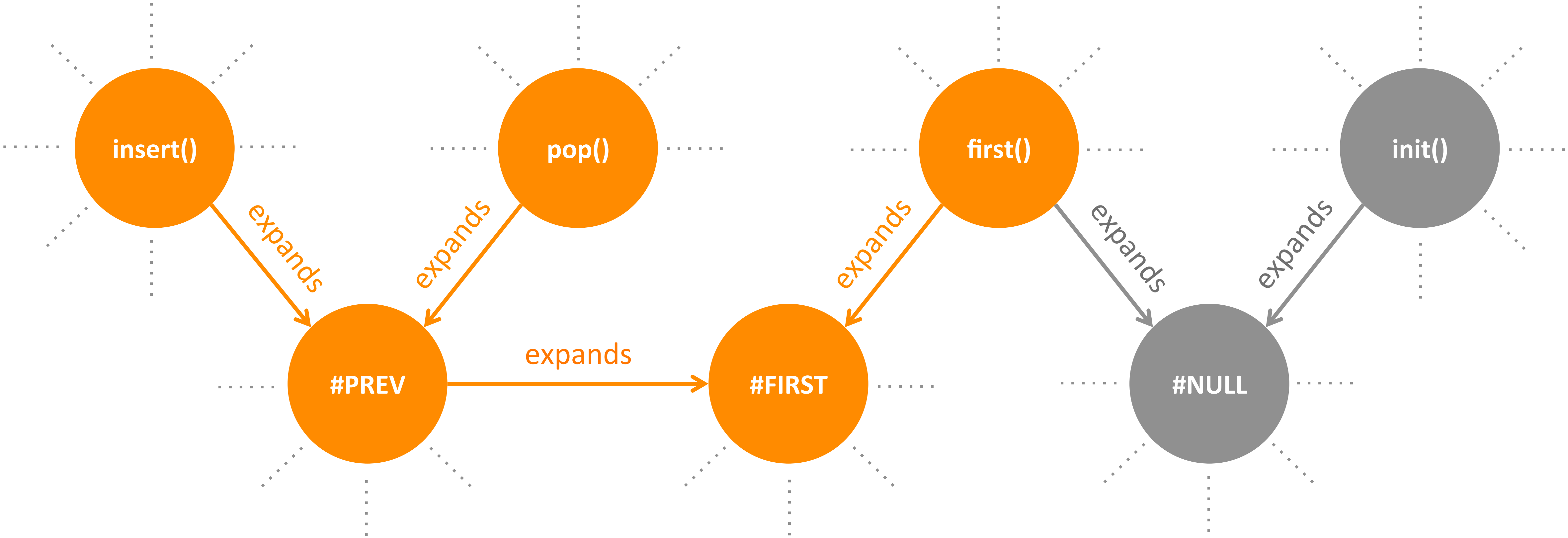
Impact Estimation



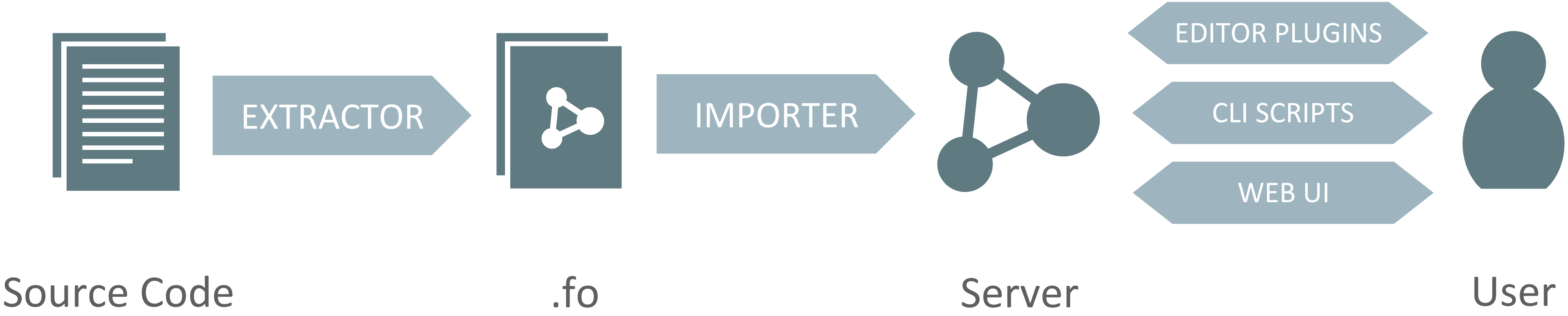
Impact Estimation



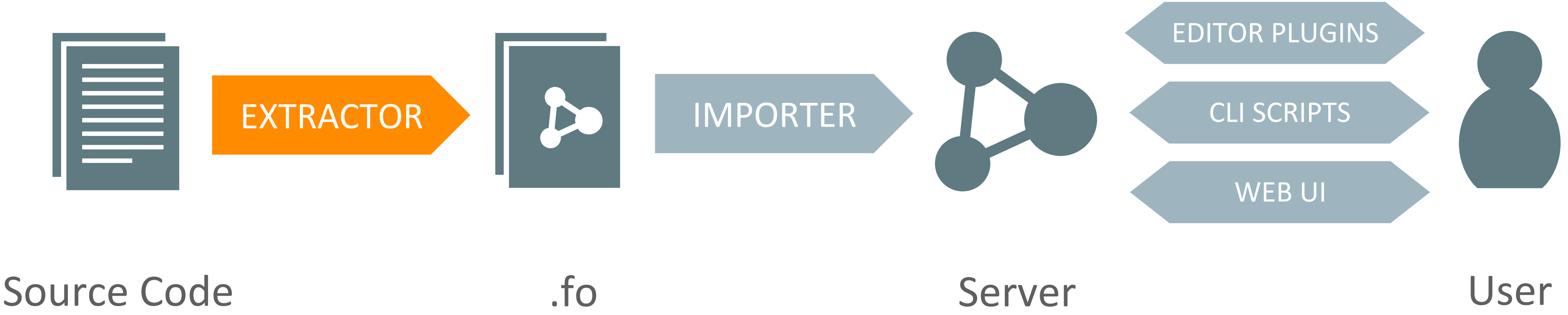
Impact Estimation



Frappé Architecture



Frappé Architecture



Source Code

EXTRACTOR

.fo

IMPORTER

Server

EDITOR PLUGINS

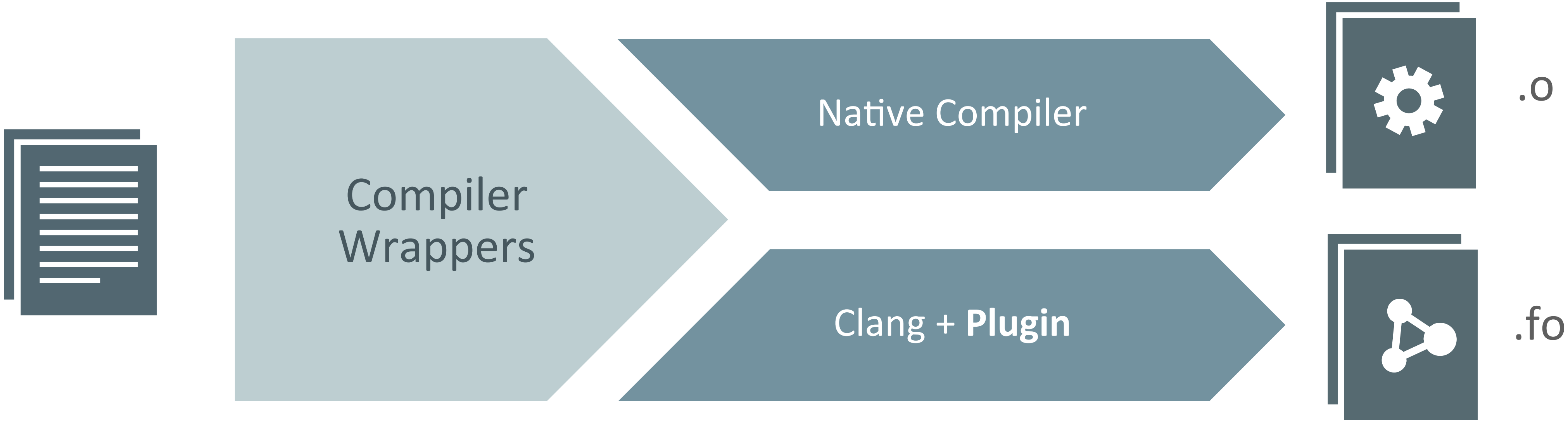
CLI SCRIPTS

WEB UI

User

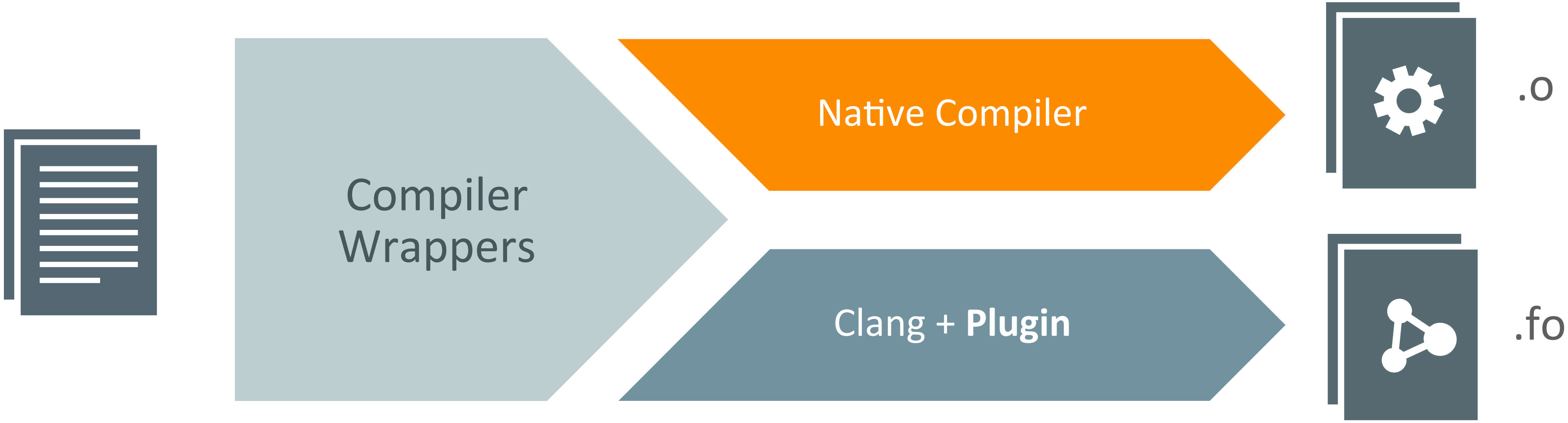
Extractor

Simple Build Integration



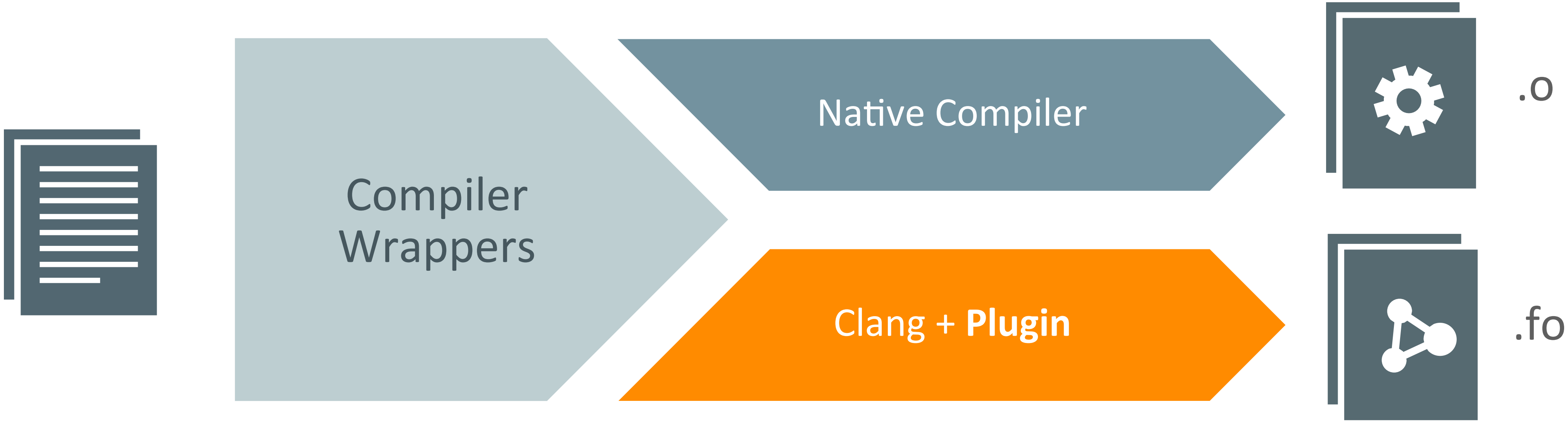
Extractor

Simple Build Integration



Extractor

Simple Build Integration

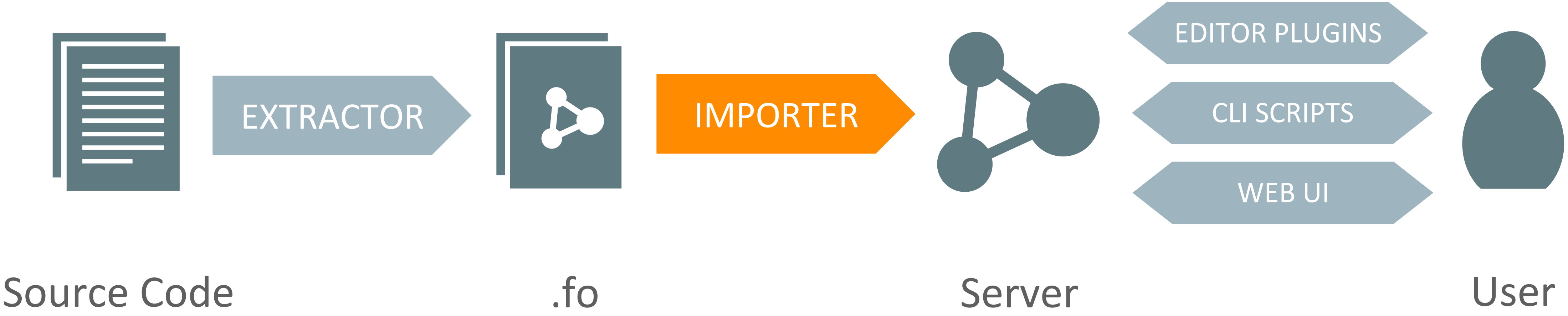


Extractor

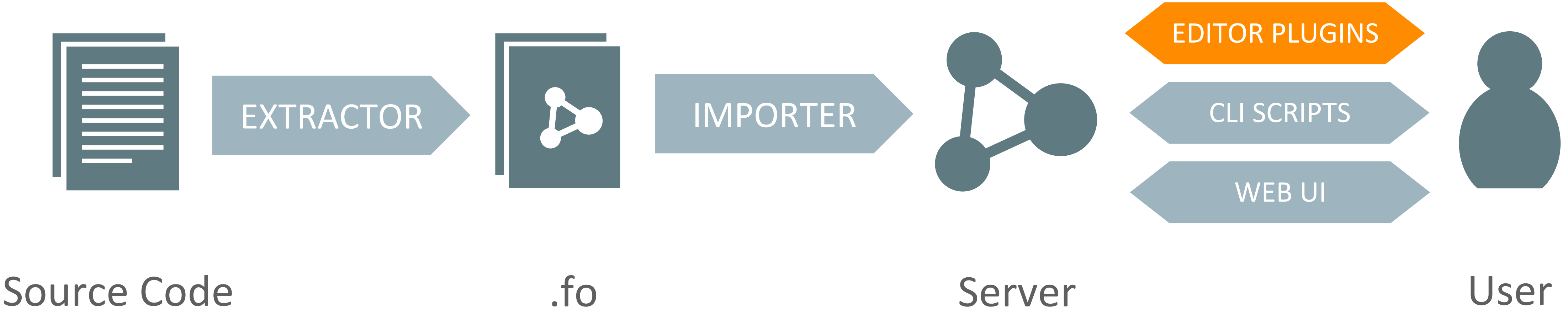
Clang Plugin

- PPCallbacks
  - Includes, macros, their expansions and interrogations
- RecursiveASTVisitor
  - Visit all declarations, types, and expressions
- Easy to use interface
  - Provides detailed location information

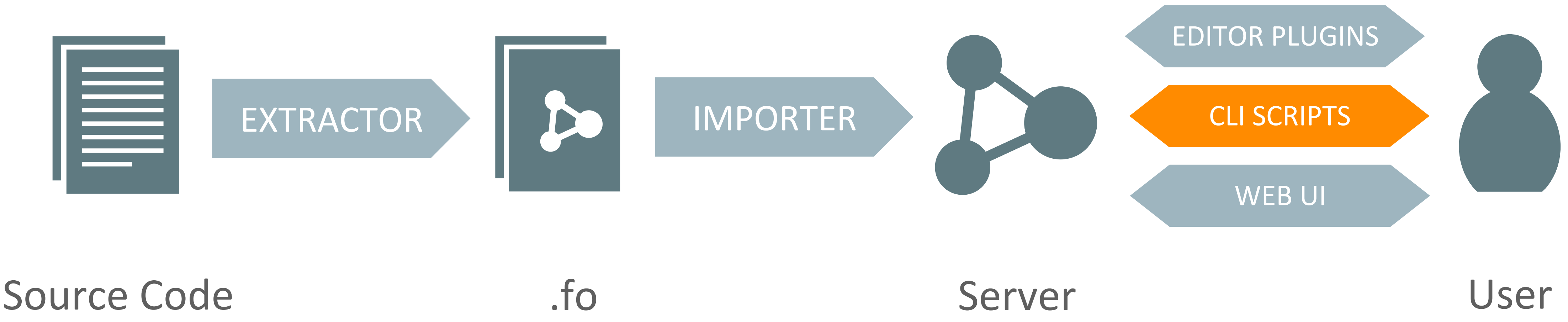
Frappé Architecture



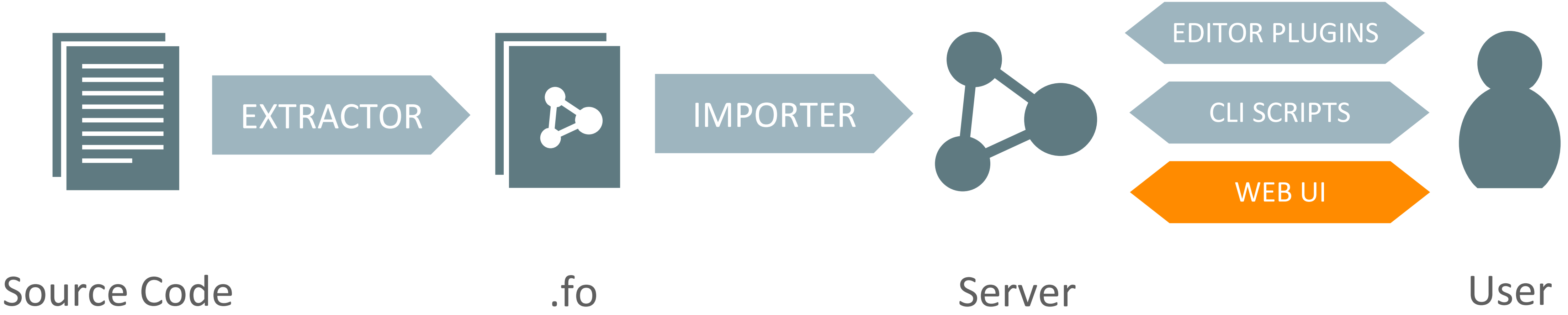
Frappé Architecture

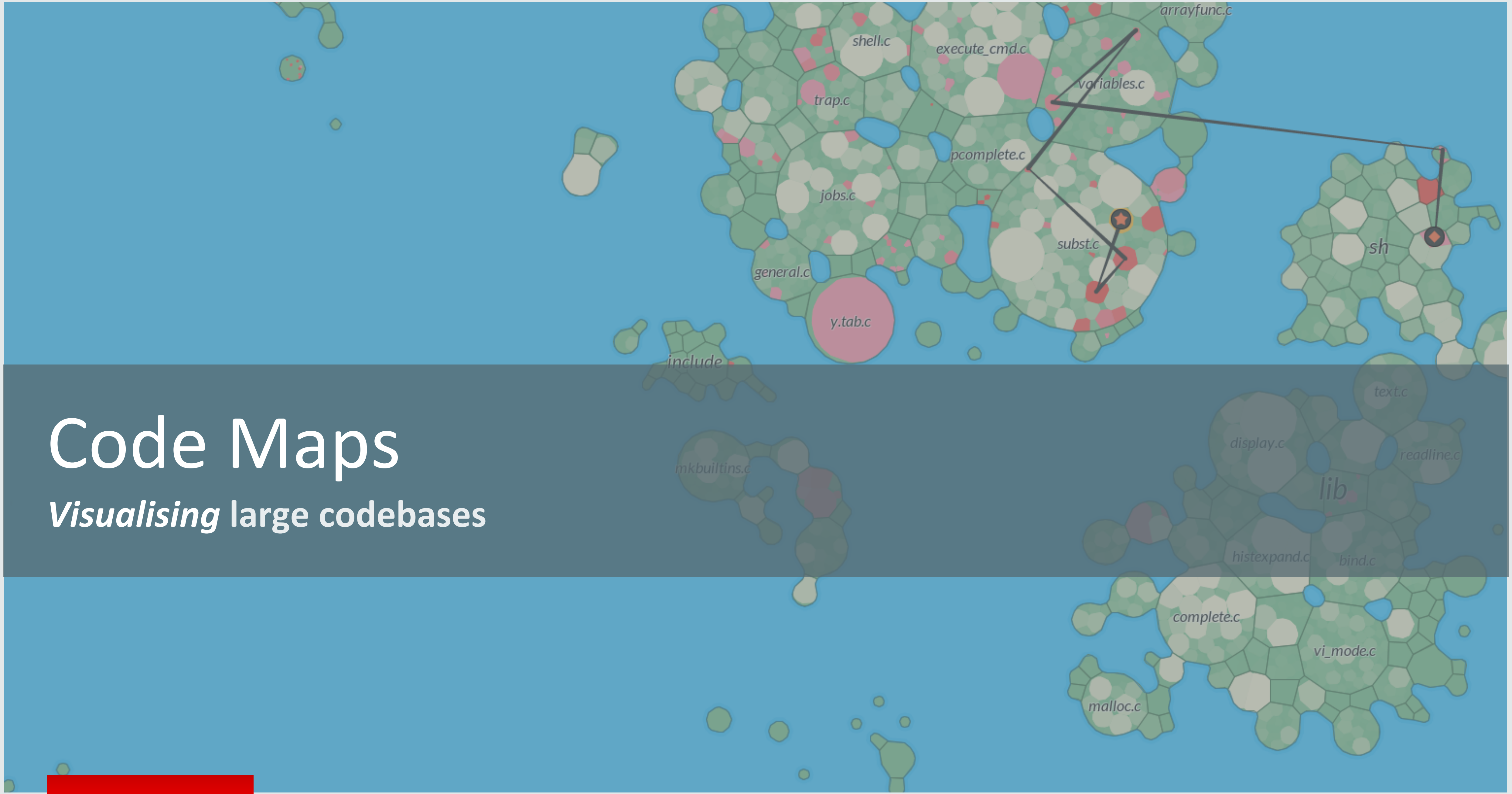


Frappé Architecture



Frappé Architecture



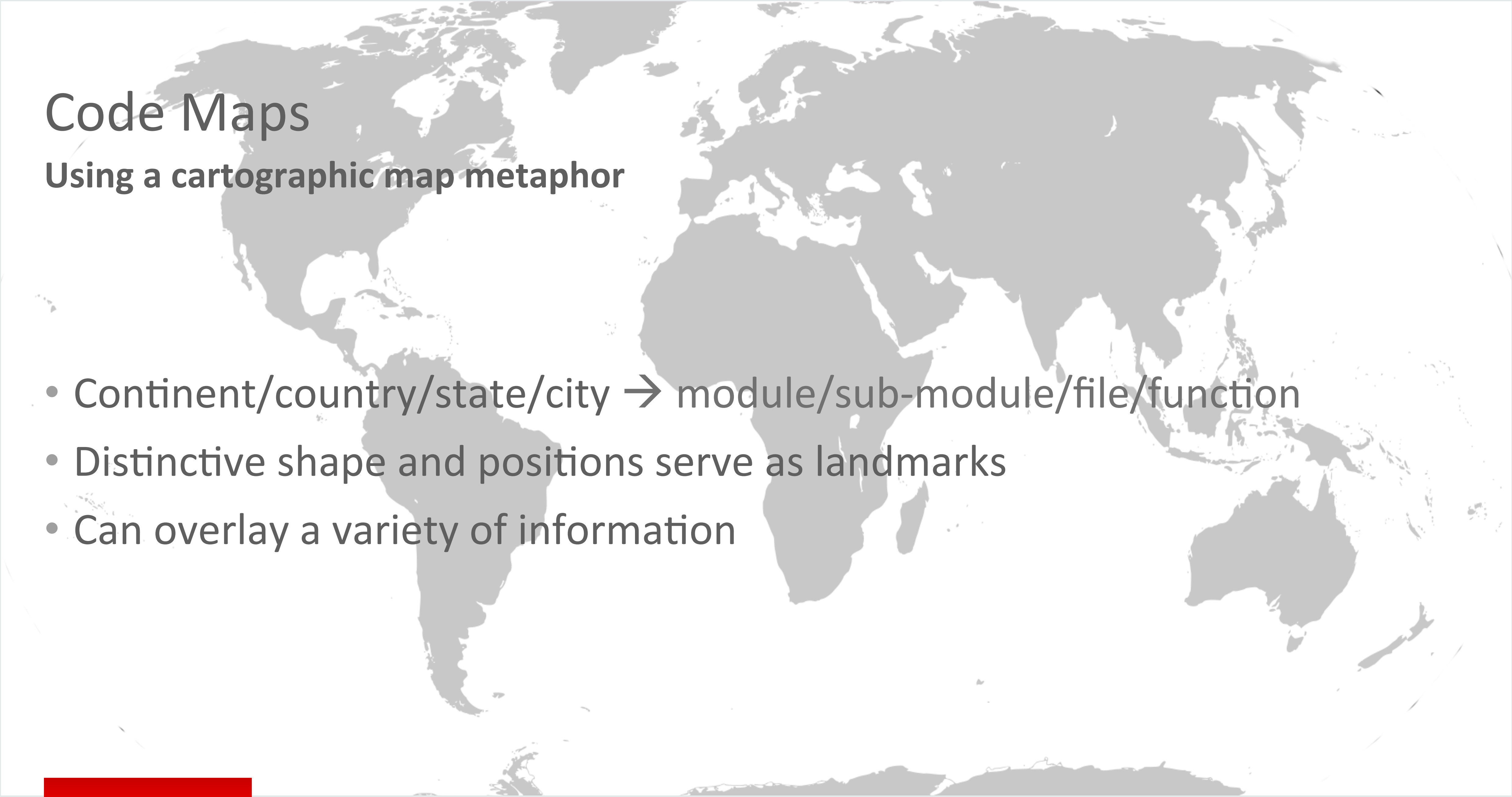


Code Maps

Visualising large codebases

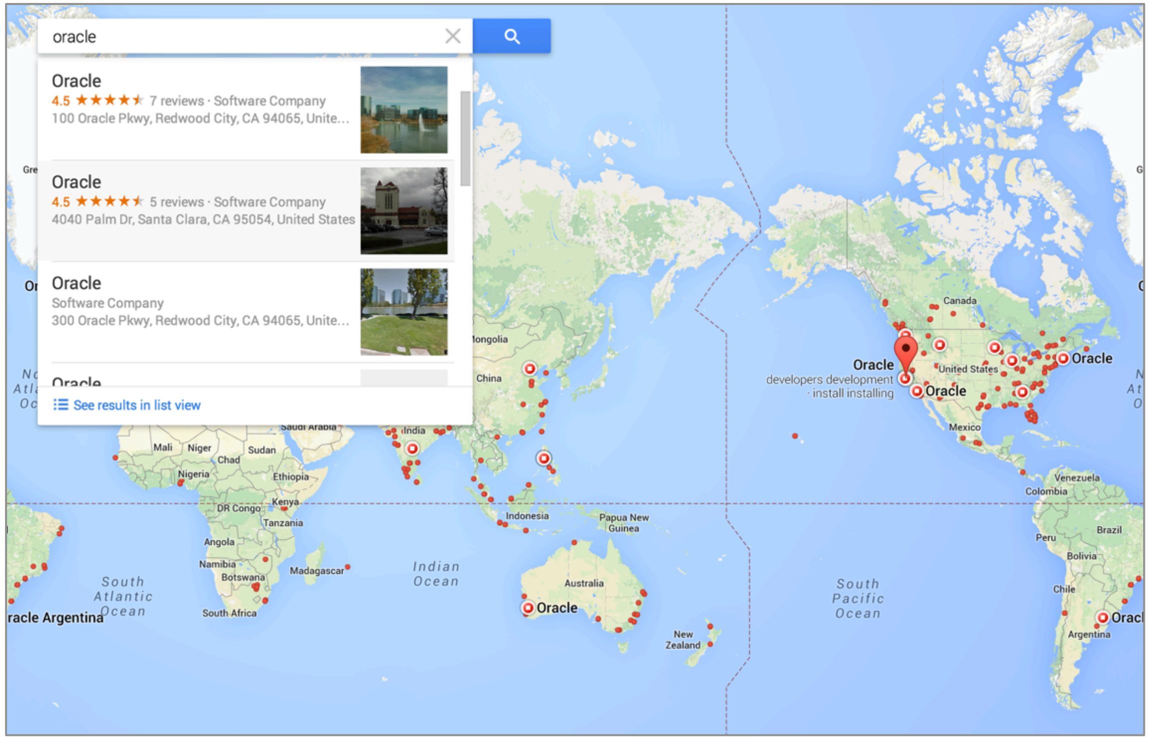
Code Maps

Using a cartographic map metaphor



- Continent/country/state/city → module/sub-module/file/function
- Distinctive shape and positions serve as landmarks
- Can overlay a variety of information

Overlay Search Results



oracle

Oracle
4.5 ★★★★★ 7 reviews · Software Company
100 Oracle Pkwy, Redwood City, CA 94065, Unite...

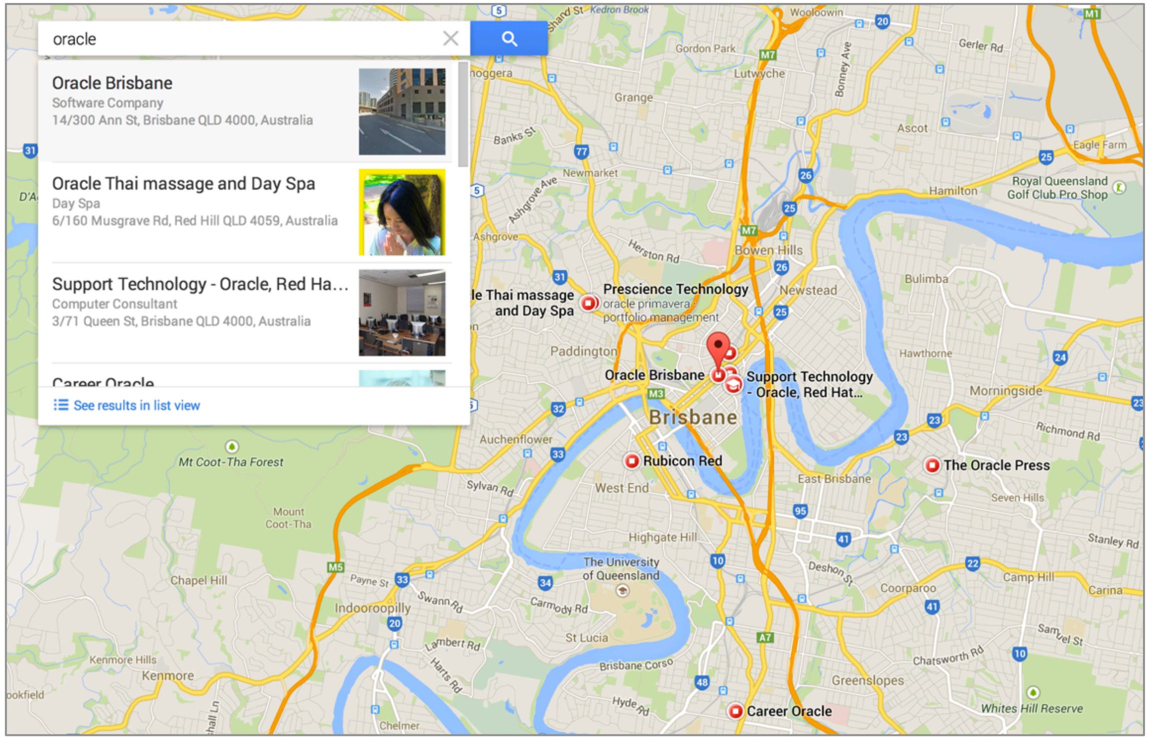
Oracle
4.5 ★★★★★ 5 reviews · Software Company
4040 Palm Dr, Santa Clara, CA 95054, United States

Oracle
Software Company
300 Oracle Pkwy, Redwood City, CA 94065, Unite...

Oracle

See results in list view

Visual filtering



oracle

Oracle Brisbane
Software Company
14/300 Ann St, Brisbane QLD 4000, Australia

Oracle Thai massage and Day Spa
Day Spa
6/160 Musgrave Rd, Red Hill QLD 4059, Australia

Support Technology - Oracle, Red Ha...
Computer Consultant
3/71 Queen St, Brisbane QLD 4000, Australia

Career Oracle

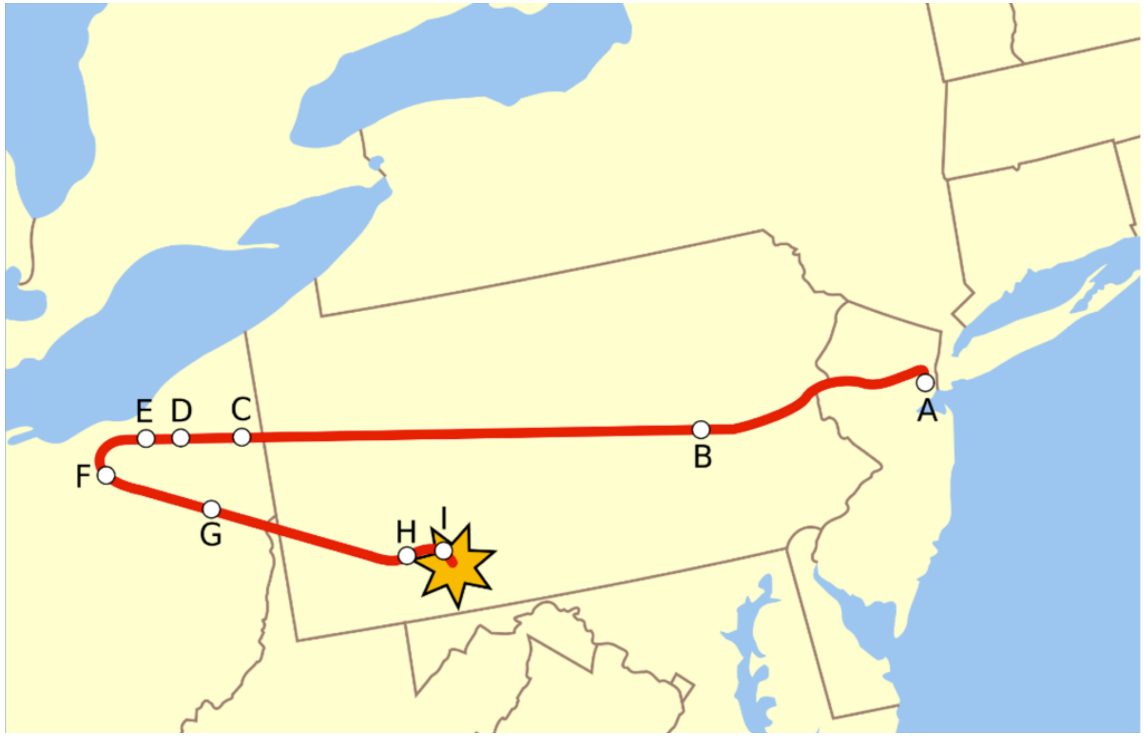
See results in list view

Contextual search

Overlay Paths

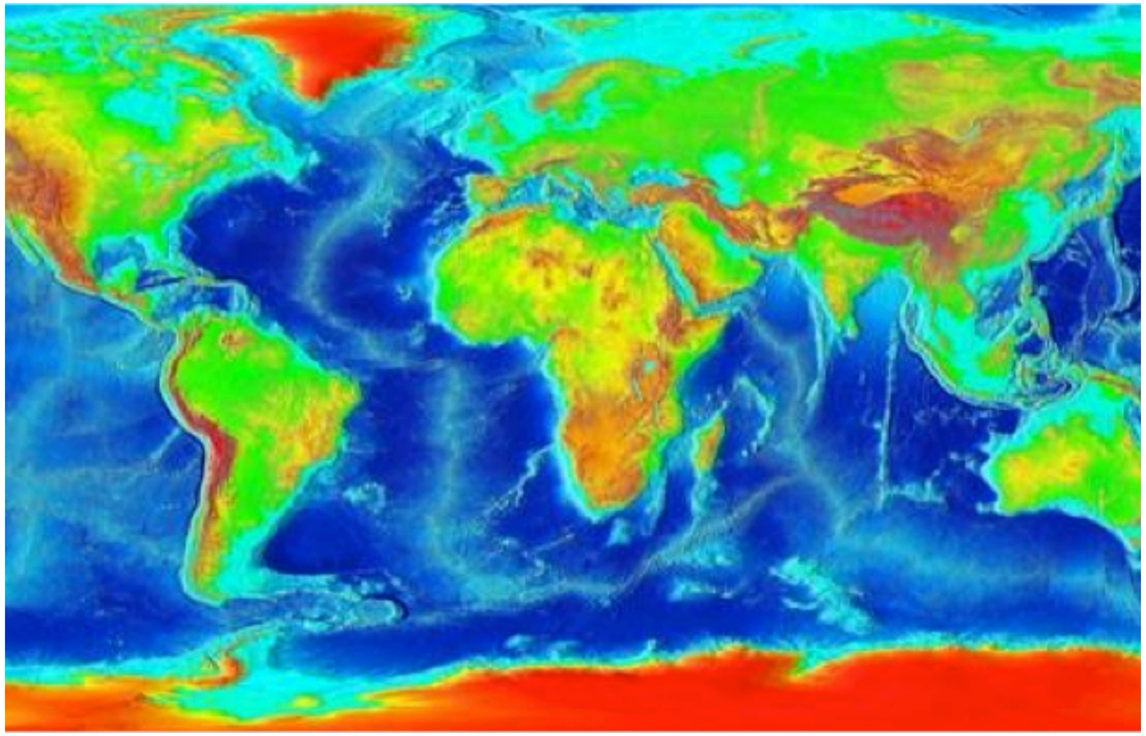
```
java.lang.RuntimeException: bad
  at Foobar.setup(Foobar.java:74)
  at Foobar.launch(Foobar.java:43)
  at Bar.launch(Bar.java:39)
  at Bar.bar(Bar.java:97)
  at Foo.foo(Foo.java:35)
  at Main.main(Main.java:104)
```

Path in stack trace

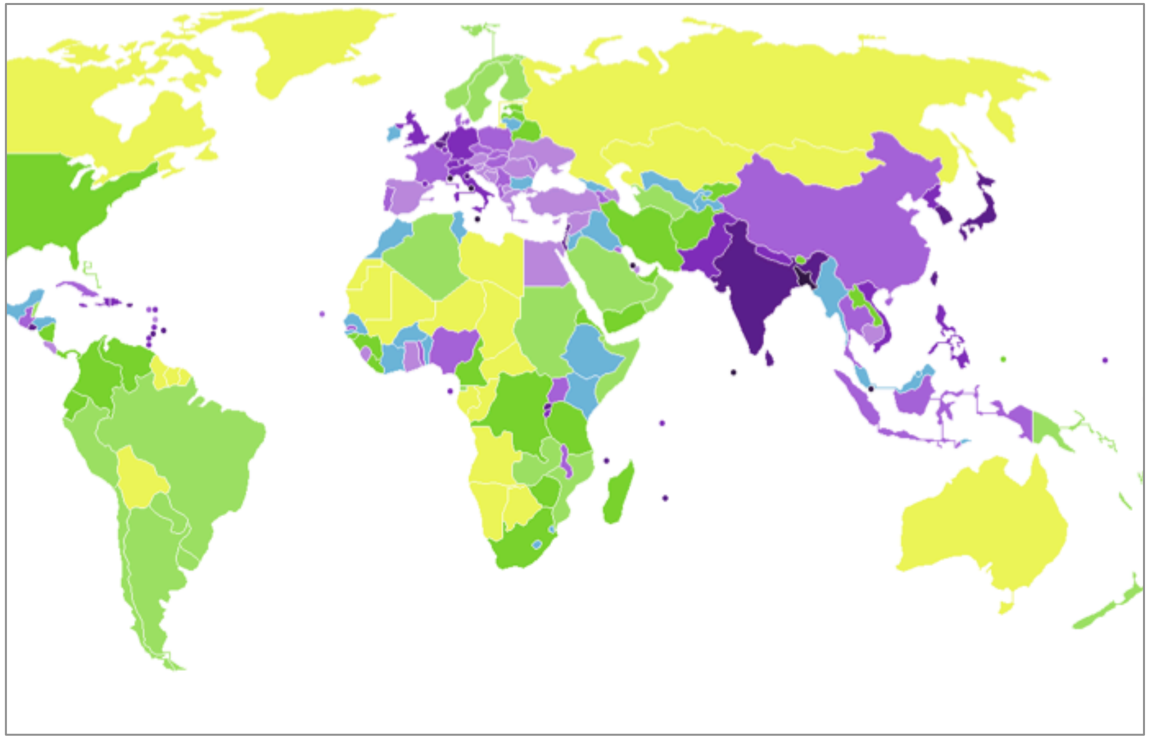


Path on map

Overlay Metrics



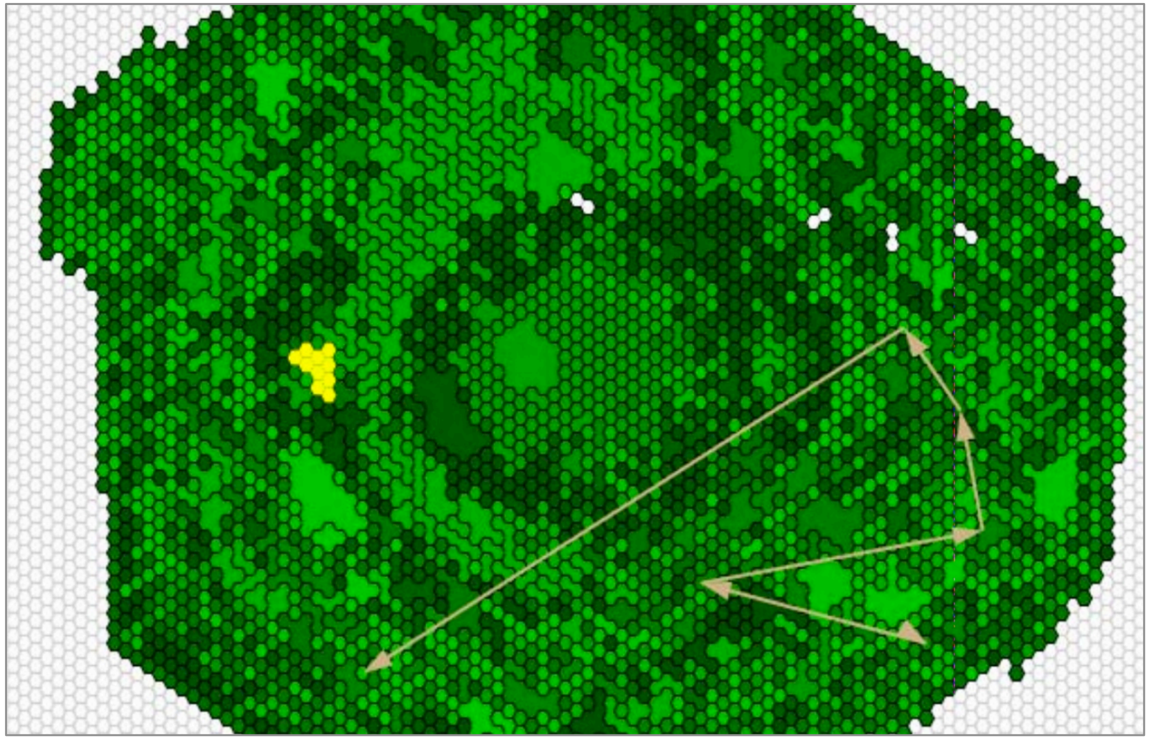
Fine granularity



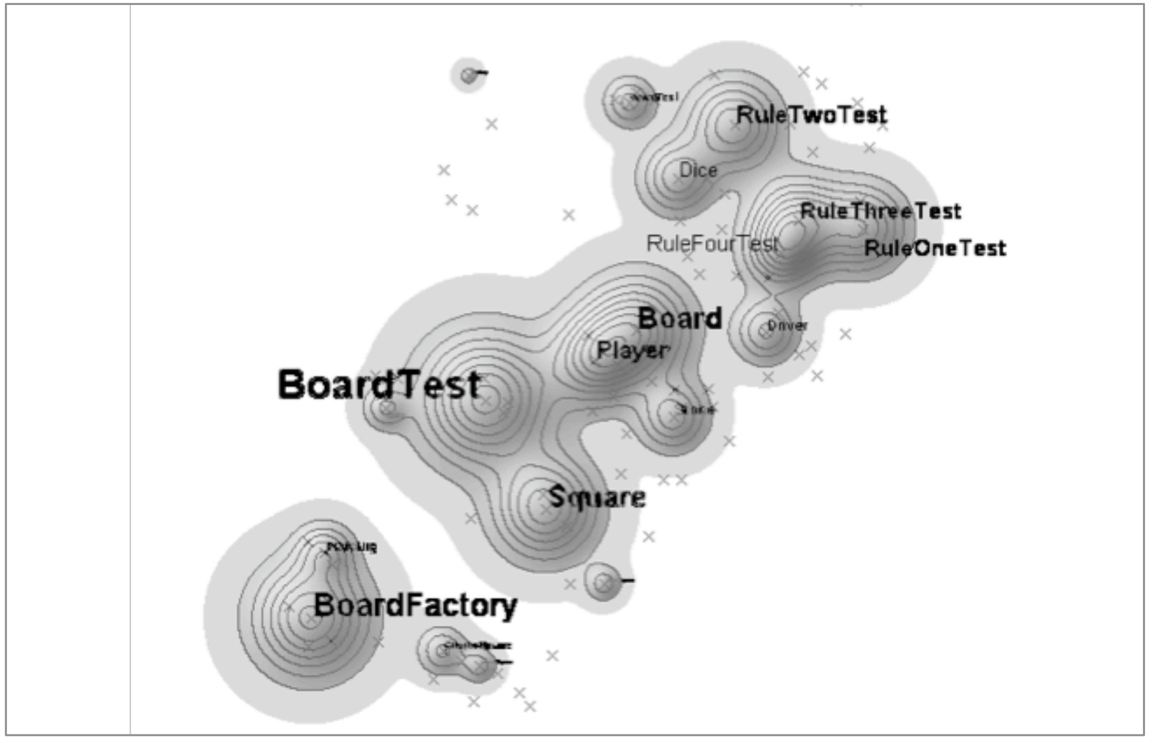
Coarse granularity

But how?

Existing Approaches



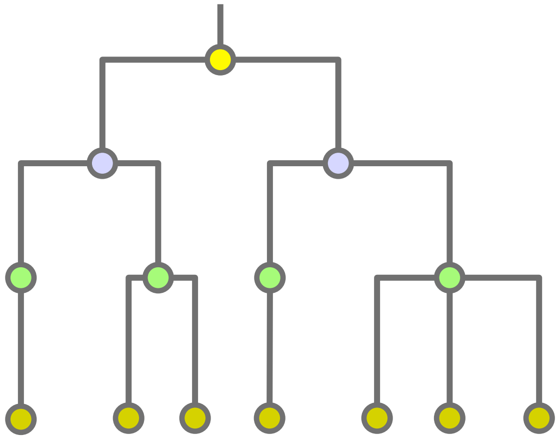
Deline, R. **Staying oriented with software terrain maps** In proc. of the workshop of visual languages and computation, 2005



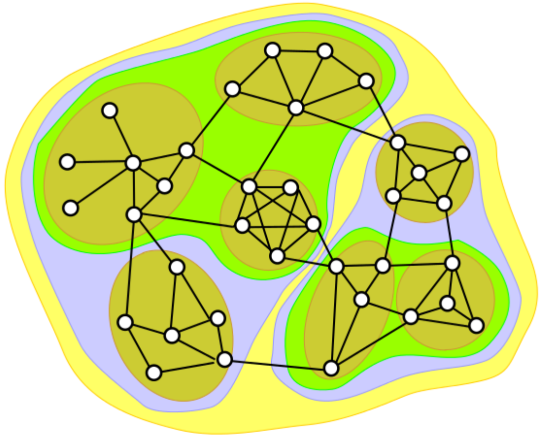
Kuhn, A.; Erni, D.; Loretan, P.; Nierstrasz, O. **Software cartography: thematic software visualization with consistent layout** Journal of Software Maintenance and Evolution, 2010

Input

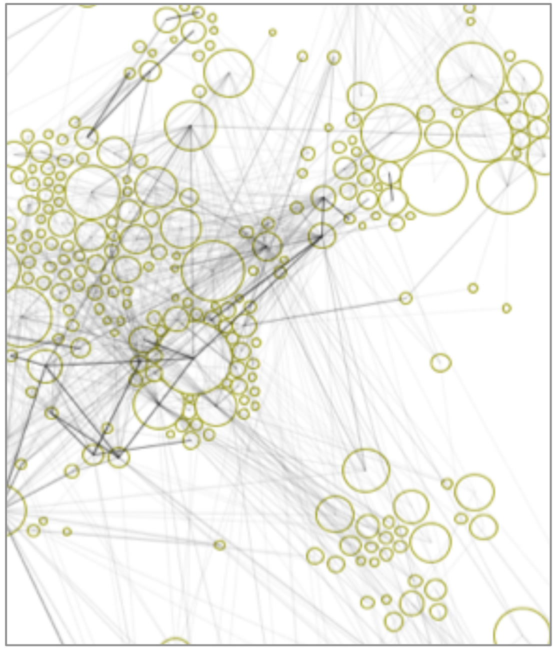
- Abstraction hierarchy
  - Abstracts files into higher level groupings
  - Use directory structure by default



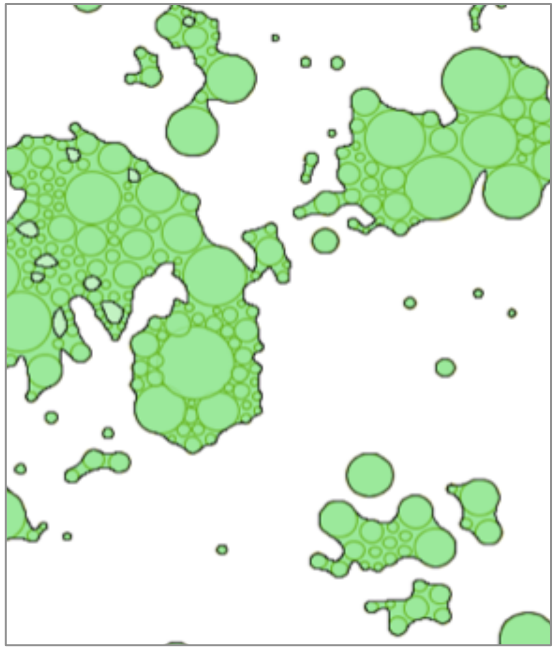
- Dependency graph
  - Represents dependencies between files as a weighted edge
  - Use references



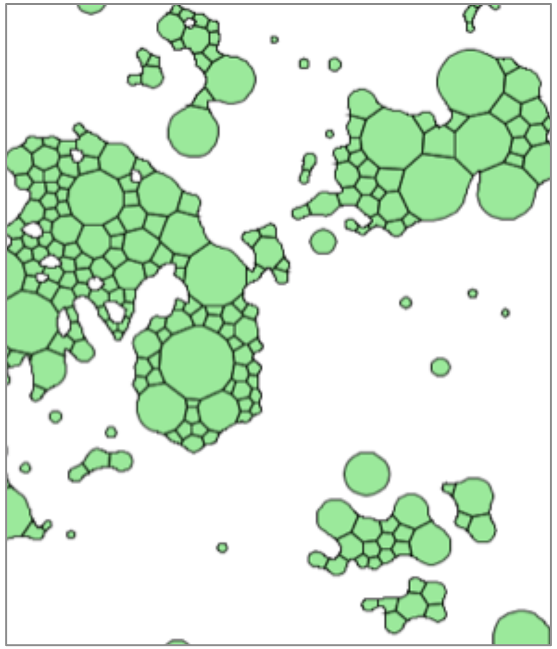
Map Generation



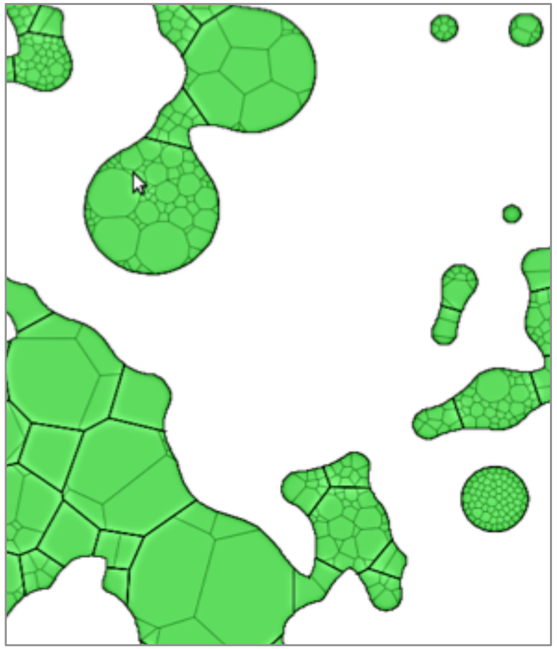
Graph layout



Implicit surface generation

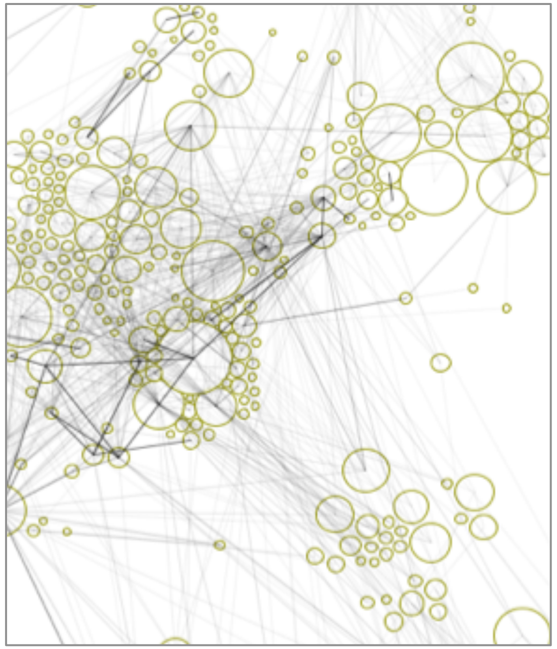


Surface subdivision

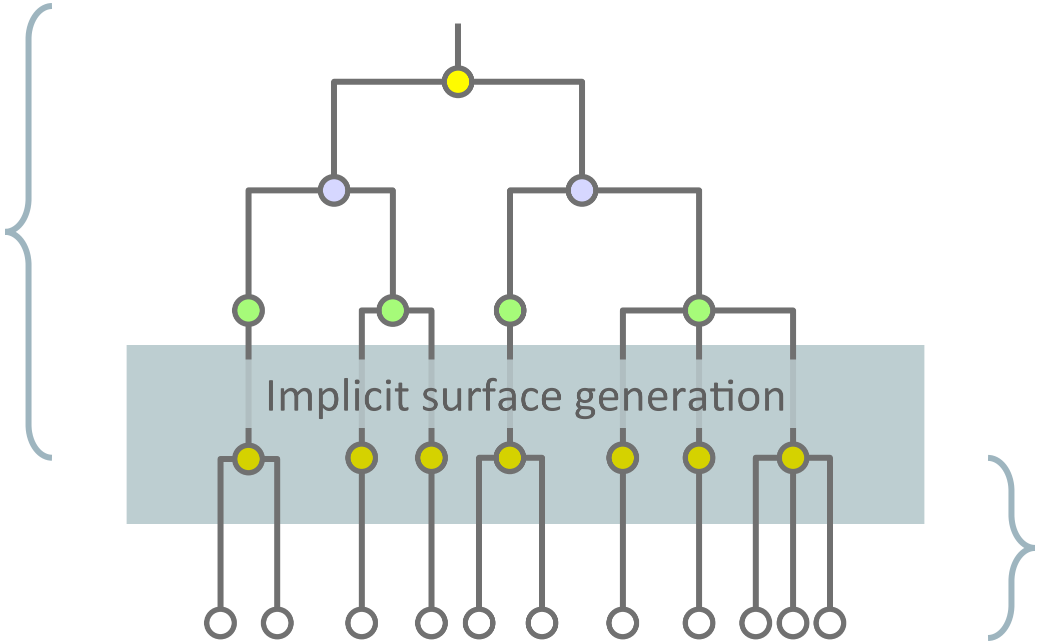


Recursive subdivision

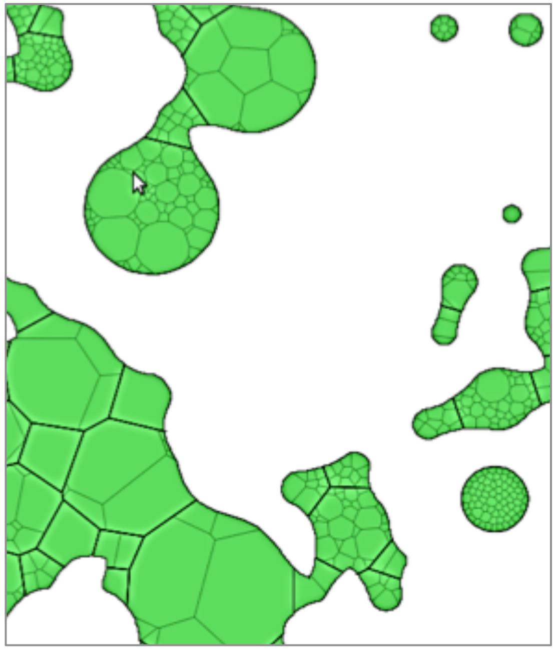
Map Generation



Graph layout



Implicit surface generation



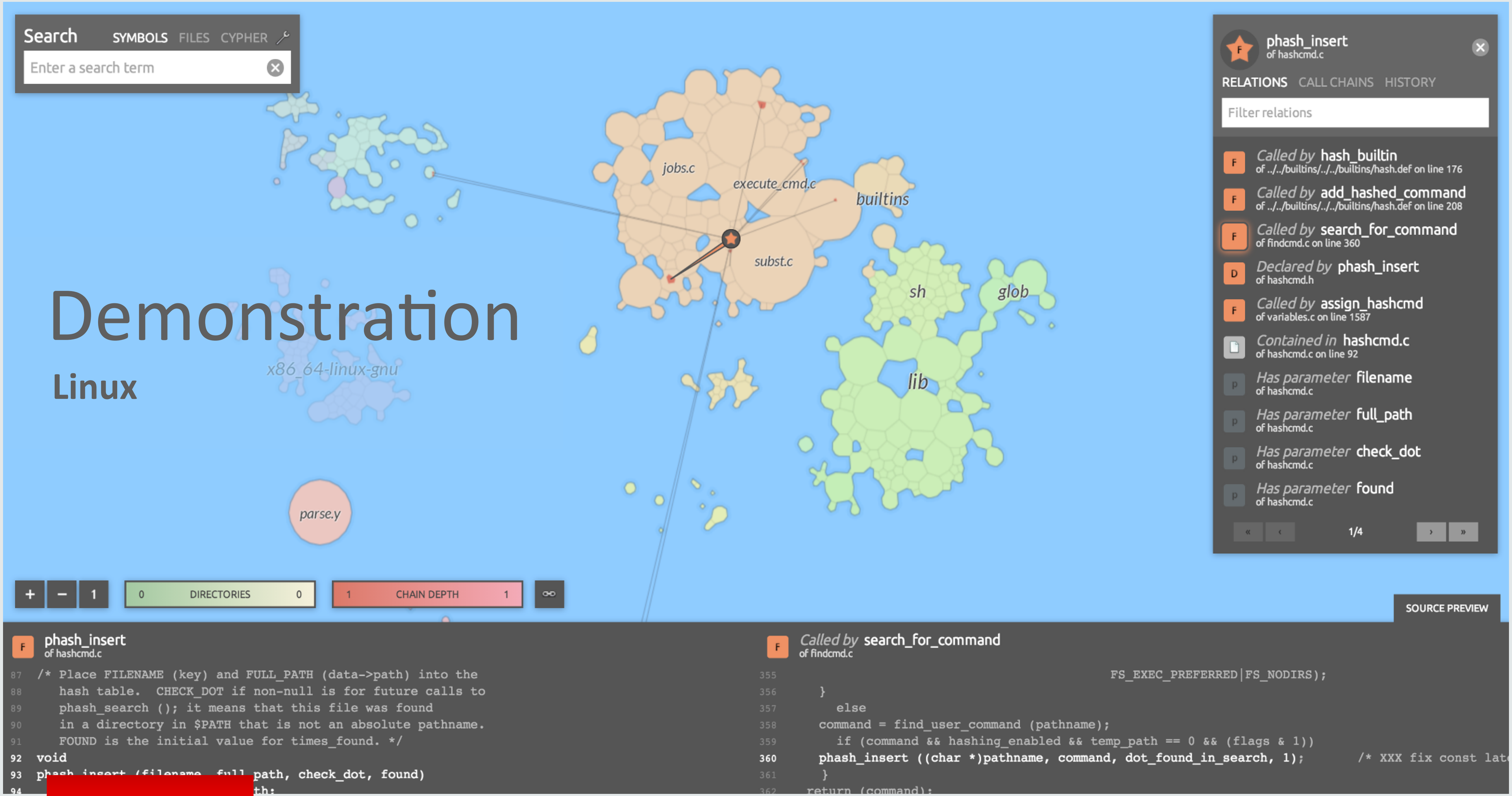
Recursive subdivision

Noack, A. & Lewerentz, C. **A space of layout styles for hierarchical graph models of software systems** Proceedings of the 2005 ACM symposium on Software visualization, ACM, 2005, 155-164

Nocaj, A. & Brandes, U. **Computing Voronoi Treemaps: Faster, Simpler, and Resolution-independent** Computer Graphics Forum, Blackwell Publishing Ltd, 2012, 31, 855-864

Demonstration

Linux



phash_insert
of hashcmd.c

RELATIONS CALL CHAINS HISTORY

Filter relations

F *Called by* hash_builtin
of ../../builtins/../../builtins/hash.def on line 176

F *Called by* add_hashed_command
of ../../builtins/../../builtins/hash.def on line 208

F *Called by* search_for_command
of findcmd.c on line 360

D *Declared by* phash_insert
of hashcmd.h

F *Called by* assign_hashcmd
of variables.c on line 1587

Contained in hashcmd.c
of hashcmd.c on line 92

p *Has parameter* filename
of hashcmd.c

p *Has parameter* full_path
of hashcmd.c

p *Has parameter* check_dot
of hashcmd.c

p *Has parameter* found
of hashcmd.c

« ‹ 1/4 › »

F **phash_insert**
of hashcmd.c

```
/* Place FILENAME (key) and FULL_PATH (data->path) into the
   hash table.  CHECK_DOT if non-null is for future calls to
   phash_search (); it means that this file was found
   in a directory in $PATH that is not an absolute pathname.
   FOUND is the initial value for times_found. */
void
phash_insert (filename, full_path, check_dot, found)
                                    th;
```

F *Called by* **search_for_command**
of findcmd.c

```
                                                 FS_EXEC_PREFERRED|FS_NODIRS);
      }
        else
      command = find_user_command (pathname);
        if (command && hashing_enabled && temp_path == 0 && (flags & 1))
      phash_insert ((char *)pathname, command, dot_found_in_search, 1);       /* XXX fix const lat
      }
    return (command);
```

Future Work

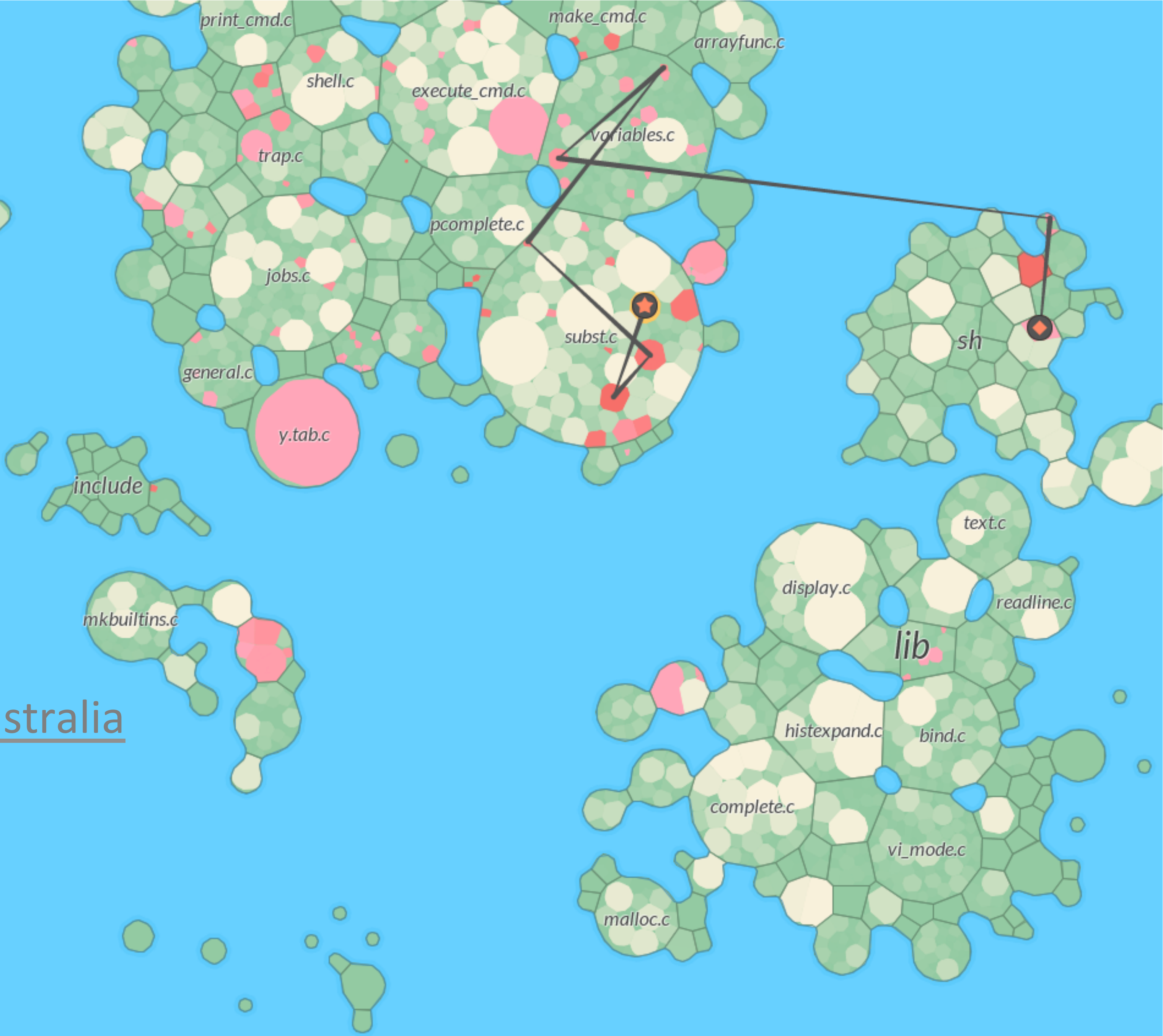
- More detailed dependency graph
  - Find calls where third argument is macro `FLAG`
  - Find all functions the pointer `fptr` could point to
- More overlays
  - Test coverage, profiling data
- Store multiple versions
  - Impact estimation
  - Code map evolution (stability)

Frappé

Nathan Hawes and Ben Barham
nathan.hawes@oracle.com
ben.barham@oracle.com

Oracle Labs Australia
http://labs.oracle.com/locations/australia

Research Director
cristina.cifuentes@oracle.com



|

ORACLE®