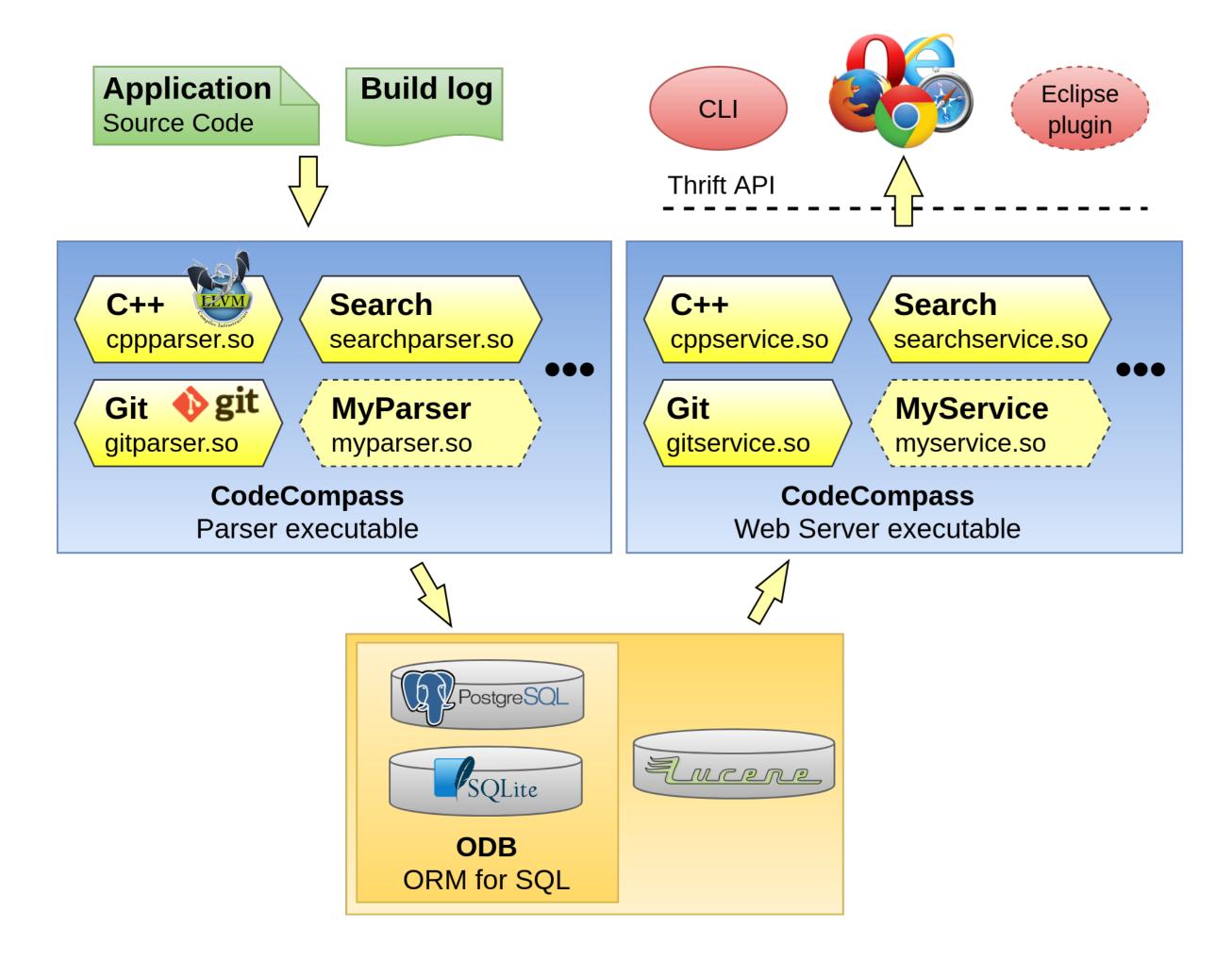




## 1. Introduction

CodeCompass is an open source framework that supports code comprehension. The information is gathered from all available resources: the source code, build commands, Git repositories, software metrics, etc. It provides wide scale textual and visual information displayed on a common WEB based GUI or queried by a Thrift API. Its functionality is extensible by plugins which may provide: a parser, a service layer, a database model and WEB GUI modules.



#### 2. Database model

Parsers collect information about the project and store them into a relational database in an extremely compressed form. Storing all AST nodes in the database would be very inefficient (~1:1000 ratio between source code and DB size). By storing only the ones with names, as these are interesting, we achieve a  $\sim 1:30$  size ratio considering C++ related tables.

Parsing the LLVM code base (320MB) itself results in a 21GB database.

Language related persistent information is available via an ORM tool. Its simplified schema is shown to the right.

# CodeCompass An Open Software Comprehension Framework

Zoltán Porkoláb, Tibor Brunner, Márton Csordás, Máté Cserép, Anett Fekete, Endre Fülöp, Gábor Horváth Eötvös Loránd University, Budapest Ericsson Ltd.

https://github.com/Ericsson/CodeCompass (live demo is available)

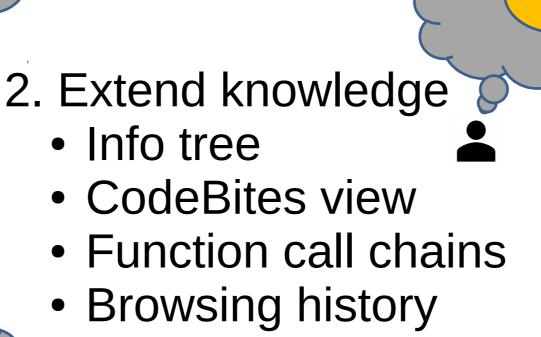
#### 3. Incremental parsing

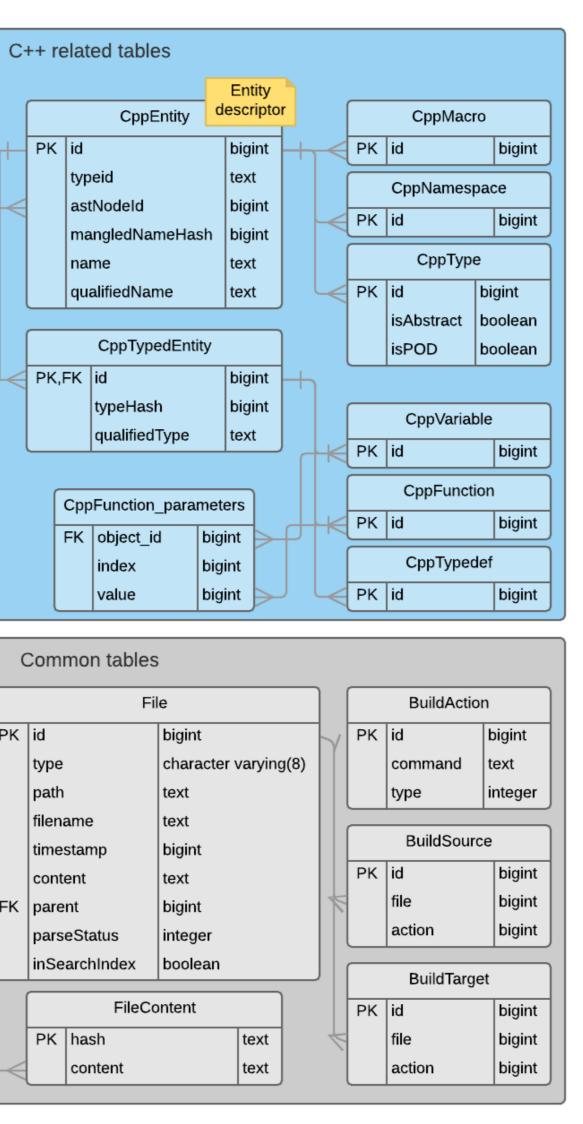
Parsing is the most time consuming action. However, code modifications are detected and only the affected translation units should to be reparsed. Individual users may have small code changes. These code editions can be modeled as database layers over a baseline parsing.

## 4. Typical comprehension workflow

- 1. Fast feature location Textual search Log-base search Info tree
- 3. Validate knowledge
- All references
- Architectural info
- Version control info

Occurrence of the entit CppAstNode ocation range start line cation range end line ocation range end columr PK,FK id mangledNameHash astType oleInSourceCode CppHeaderInclusion PK id intege includer included bigint value Common tables Other plugins Metrics PK id bigint file metric integer BuildLog type integer Statistics PK id barer bigint parseStatus group text location\_range\_start\_line inSearchIndex key text value integer



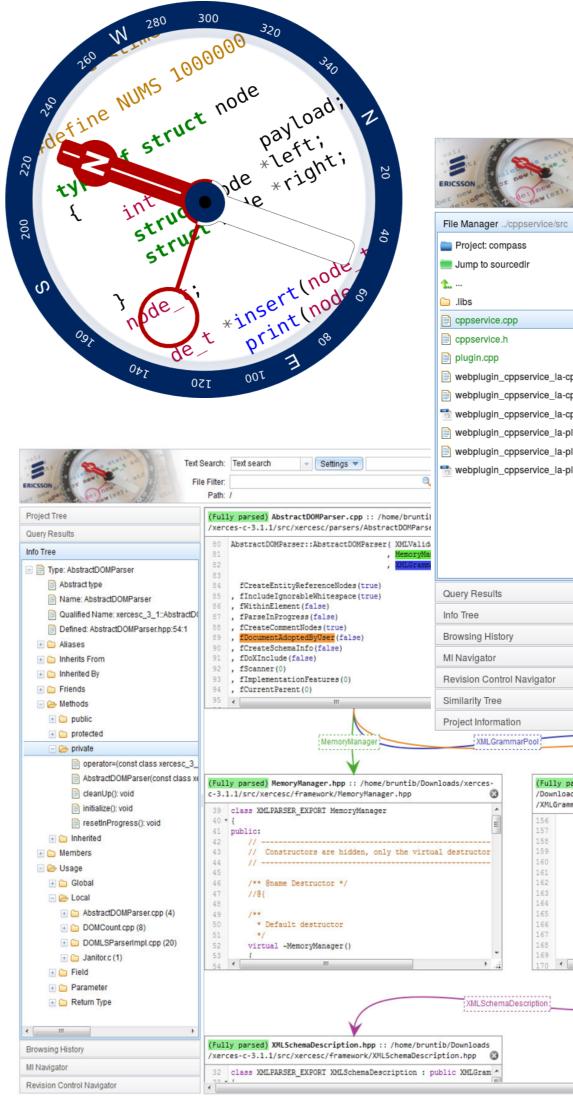


## 5. Features

- Macro expansions
- Symbol search
- Log output fuzzy searcl
- Entity info collector
- Function call diagram
- Class diagram
- C/C++ module diagram header providers, object users, directory relations

## 6. Future plans

CodeCompass currently provides a snapshot view i.e. one parsed version of the project is stored in a read-only database that can be accessed via a Thrift API. Language Server Protocol (LSP) is a Microsoft initiative for connecting an editor with a language smartness provider, like CodeCompass, by a similar interface. We are going to implement the LSP protocol for CodeCompass.





:h	<ul> <li>Git history browser</li> <li>Pointer analysis</li> <li>Displaying compiler generated functions</li> <li>Call chain via function</li> </ul>
	pointers
n:	<ul> <li>Metrics</li> </ul>
ct	<ul> <li>CodeBites</li> </ul>
IS	

8	Search: Text search	Cocungs	h expression, like "foo AND bar" (see question mark icon for 🚽 🕗 Menu 🔻
10 1 E	File Filter: File name filter regex	(.*cpp) 🔍 🄇	Directory Filter: Path filter regex (click on a dir below) Compass
and the second	Copen file: cppservice.cpp		>
MC Style 🔻	(Fully parsed) cppservic	:e.cpp :: /ssd/whispe	erity/CodeCompass_Discovery/ccSrcMaster/service/cppservice/src/cppservice.cpp
	Find in this file:	2013 00 17 11.3/	(Use /re/ syntax for regexp search) < prev next >
	13 Zoltán Borók-Nagy	2013-10-14 16:52	<pre>#include "model/cxx/cppastnode.h"</pre>
	14 Zoltán Borók-Nagy 15 Krupp	2013-07-30 15:59 2013-05-14 10:32	//for now we only support one workspace
	16 Krupp	2013-05-14 10:32	#define WORKSPACE_ID 1
	– 17 Zalán Szűgyi 18 Zalán Szűgyi	2014-01-12 22:24 2014-01-12 22:24	namespace cc
	19 Zalán Szűgyi	2014-01-12 22:24	* {
	20 Zalán Szűgyi 21 Zalán Szűgyi	2014-01-12 22:24 2014-01-12 22:24	namespace service * {
service.d	22 Zalán Szűgyi	2014-01-12 22:24	namespace language
service.lo	23 Zalán Szűgyi 24 Krupp	2014-01-12 22:24 2013-05-16 08:49	* 1
oservice.o	25 Krupp 26 Zoltán Borók-Nagy	2013-05-16 08:49 2013-05-09 10:06	<pre>CppServiceHandler::CppServiceHandler(const CppServiceHelper&amp; helper) : helper(helper)</pre>
gin.d	27 Krupp	2013-05-16 08:49	{
gin.lo	28 Krupp 29 Krupp	2013-05-16 08:49	}
gin.o	30 Krupp	2013-05-16 08:49 2013-05-16 08:49	<pre>void CppServiceHandler::getAstNodeInfoByPosition(AstNodeInfo&amp; _return,</pre>
ginto	31 Zalán Szűgyi 32 Krupp	2014-01-12 22:24 2013-05-16 08:49	<pre>const core::FilePosition&amp; fpos, const std::vector<std::string> &amp; filters)</std::string></pre>
	33 Zoltán Borók-Nagy	2013-05-09 10:06	* {
	34 Zalán Szűgyi 35 Krupp	2014-01-12 22:24 2013-05-16 08:49	<pre>SLog(util::DEBUG)&lt;&lt;"getting astno return = helper.getAstNodeInfoByl ====================================</pre>
	36 Zoltán Borók-Nagy	2013-05-09 10:06	_return = helper.getAstNodeInfoBy( const std::vector & filters ); }
	37 Krupp 38 Tibor Brunner	2013-05-16 08:49 2015-02-17 12:52	<pre>void CppServiceHandler::getInfoBox(InfoBox&amp; return,</pre>
	39 Tibor Brunner	2015-02-17 12:52	<pre>const core::AstNodeId&amp; astNodeId)</pre>
	40 Tibor Brunner 41 Tibor Brunner	2015-02-17 12:52 2015-02-17 12:52	<pre>* {     _return = helper.getInfoBox(astNodeId);</pre>
	42 Tibor Brunner	2015-02-17 12:52	}
	43 Tibor Brunner 44 Zoltán Borók-Nagy	2015-02-17 12:52 2013-07-09 13:20	void CppServiceHandler::getInfoBoxByPosition(InfoBox
	45 Zalán Szűgyi	2014-01-12 22:24	const core::FilePosition& fpos,
	46 Zoltán Borók-Nagy 47 Zoltán Borók-Nagy	2013-07-09 13:20 2013-07-09 13:20	<pre>const std::vector<std::string> &amp; filters) * {</std::string></pre>
	48 Zoltán Borók-Nagy	2013-07-09 13:20	_return = helper.getInfoBoxByPosition(fpos, filter File diagrams
	49 Zoltán Borók-Nagy 50 Zoltán Borók-Nagy	2013-07-09 13:20 2013-07-09 13:20	/ Metrics 🔻
	51 Krupp 52 Zalán Szűgvi	2013-05-16 08:49 2014-01-12 22:24	<pre>void CppServiceHandler::qetAstNodeInfo(AstNodeInfo&amp; return,</pre>
fDocumentAdo	optedByUser	2014 01 12 22.24	
sed) XMLGrammarPoo	1.hpp :: /home/bruntib	(Fully parsed) Abst	tractDOMParser.hpp :: /home/brunti
ds/xerces-c-3.1.1/src/xercesc/framework		/Downloads/xerces-c	c-3.1.1/src/xercesc/parsers
marPool.hpp		/AbstractDOMParser.	fDocumentAdopte
* createDTDDescri	iption	1700 2001	recommenced approximation appro
•/			
virtual XMLDTDDesc: **	ription* createDTDDescrip		
* createSchemaDes *	scription		
*/	escription* createSchemaDes(		
	eacription Createschemabes		
/@}			
//	•		