# <srcML>

# srcML 1.0: Explore, Analyze, and Manipulate Source Code

**Michael L. Collard**
collard@uakron.edu
Department of Computer Science
The University of Akron
Ohio, USA

**Jonathan I. Maletic**
jmaletic@kent.edu
Department of Computer Science
Kent State University
Ohio, USA



# srcML

*noun* | src·M·L | \sōrs-em-el\

1 : an infrastructure for the exploration, analysis, and manipulation of source code.

2 : an XML format for source code.

3 : a lightweight, highly scalable, robust, multi-language parsing tool to convert source code into srcML.

4 : a free software application licensed under GPL.

# What does srcML do?

- Convert source code to srcML
- Query code using XML query languages, such as XPath
- Convert srcML back to original source, with no loss of text
- Transform source code while in srcML format
  - src ➡ srcML ➡ (transform) ➡ srcML ➡ src

# History

- Original Motivation: extracting function-level comments (for LSI) and fact extraction
- Published at IWPC’02, IWPC’03, and DocEng’02
- Received "Most Influential Paper" (MIP) Award at ICPC’13 (for IWPC’03 paper)
- Used in over 24 dissertations/theses

# Support


- Supported in part by a grant from CNS 13-05292/05217

  - July, 2013 - July, 2017 funding to enhance infrastructure

- ABB supported srcML early on

# srcML Team

- Michael Collard
- Jonathan Maletic

- Brian Bartman
- Michael Decker
- Drew Guarnera

- Brian Kovacs
- Heather Michaud
- Christian Newman


# The srcML Format

- A document-oriented XML format that explicitly embeds structural information directly into the source text

- Markup is selective at a high AST level (i.e., no sub-expressions)

# Source Code

```
#include "rotate.h"

// rotate three values
void rotate(int& n1, int& n2, int& n3)
{
    // copy original values
    int tn1 = n1, tn2 = n2, tn3 = n3;

    // move
    n1 = tn3;
    n2 = tn1;
    n3 = tn2;
}
```

# srcML

```
<unit xmlns="http://www.srcML.org/srcML/src" xmlns:cpp="http://www.srcML.org/srcML/cpp"
revision="“1.0" language="C" filename="rotate.c">
<cpp:include>#<cpp:directive>include</cpp:directive> <cpp:file>"rotate.h"</cpp:file>
</cpp:include>

<comment type="line">// rotate three values</comment>
<function><type>void</type> <name>rotate</name>
<parameter_list>(<param><type>int&amp;</type> <name>n1</name></param>,
<param><type>int&amp;</type> <name>n2</name></param>,
<param><type>int&amp;</type> <name>n3</name></param>)</parameter_list>
<block>{
    <comment type="line">// copy original values</comment>
    <decl_stmt><decl><type><name>int</name></type> <name>tn1</name> =<init> <expr><name>n1</
name></expr></init>, <name>tn2</name> =<init> <expr><name>n2</name></expr></init>, <name>tn3</
name> =<init> <expr><name>n3</name></expr></init></decl>;</decl_stmt>

     <comment type="line">// move</comment>
     <expr-stmt><expr><name>n1</name> = <name>tn3</name></expr>;</expr-stmt>
     <expr-stmt><expr><name>n2</name> = <name>tn1</name></expr>;</expr-stmt>
     <expr-stmt><expr><name>n3</name> = <name>tn2</name></expr>;</expr-stmt>
}</block></function>
</unit>
```

# srcML Markup

- All original text preserved, including white space, comments, special characters
- Syntactic structure wrapped with tags, making them addressable
- Comments marked in place
- Pre-processor statements unprocessed

# srcML Elements

| | |
|---|---|
| **Statements** | <if>, <then>, <else>, <elseif>, <while>, <for>, <do>, <break>, <continue>, <return>, <switch>, <case>, <default>, <block>, <label>, <goto>, <empty_stmt>, <foreach>, <fixed>, <block>, <using>, <unsafe>, <assert> |
| **Specifiers** | <specifier>, <extern> |
| **Declarations, Definitions, and Initializations** | <decl_stmt>, <decl>, <function_decl>, <function>, <modifier>, <typedef>, <init>, <range>, <literal>, <lambda>, <using>, <namespace> |
| **Classes, Struct, Union, Enum, Interfaces** | <struct_decl>, <struct>, <union_decl>, <union>, <enum>, <class>, <class_decl>, <constructor>, <constructor_decl>, <super>, <destructor>, <annotation>, <extends>, <implements>, <static>, <protected>, <private>, <public> |
| **Expressions** | <call>, <name>, <ternary>, <expr>, <operator>, <argument>, <argument_list>, <parameter>, <parameter_list>, <name> |
| **Generics** | <decl>, <class>, <function>, <specifier>, <where>, <name>, <template>, <typename>, <modifier> |
| **Exceptions** | <throw>, <throws>, <try>, <catch>, <finally> |
| **LINQ** | <from>, <where>, <select>, <group>, <orderby>, <join>, <let> |
| **Other (C-based)** | <operator>, <sizeof>, <alignas>, <alignof>, <atomic>, <generic_selection>, <specifier>, <asm> |
| **Other (C#-based)** | <typeof>, <default>, <checked>, <unchecked>, <sizeof>, <attribute> |
| **Other (C++-based)** | <call>, <typeid>, <noexcept>, <decltype> |
| **Other (Java-based)** | <import>, <package>, <synchronized> |

# srcML.org

- Executables: Windows, Fedora, macOS, and Ubuntu
- Source Code - Github
- Bug Reporting
- Documentation
- GPL

# Implementation

- Parsing technology in C++ with ANTLR
- Uses libxml2, libarchive, boost

- Current file speed: ~35 KLOC/second
- srcML to text:  ~4.5 (~1.4 compressed)

- Allows for various input sources, e.g., directories, source archives (tar.gz, etc.)

# srcML Parser

- Custom parser based on modifications to ANTLR parser framework
- Comments and white space in a separate token stream. C-Preprocessor in a separate token stream
- Parser produces token stream with XML tags
- Highly efficient and scalable

# Source Issues

- Source Encoding:
  - Want: UTF-8
  - Get: ASCII, ISO-8859-1 (Latin1), UTF-8 BOM
  - Specify with --src-encoding
- Language Detection:
  - Based on extension
  - Can specify with --language
  - Can register extensions --register-ext

# Language Support

- C11, K&R C
- C++14, Qt extensions
- Java SE 8
- C# Standard ECMA-334
- OpenMP pragmas

# srcML Archive

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<unit xmlns="http://www.srcML.org/srcML/src"  revision="1.0">

<unit xmlns:cpp="http://www.sdml.info/srcML/cpp" revision="1.0" language="C#" filename="main.cs" hash="09…f7">
    <!-- ... -->
</unit>

<unit xmlns:cpp="http://www.sdml.info/srcML/cpp" revision="1.0" language="C" filename="rotate.c" hash="2380…de">
    <!-- ... -->
</unit>

<!-- ... -->

<unit xmlns:cpp="http://www.sdml.info/srcML/cpp" revision="1.0" language="C" filename="rotate.h" hash="1e…35">
    <!-- ... -->
</unit>

</unit>
```

# srcML Infrastructure

## Tools

Tools provided and custom built are used to query, extract data, and transform source code.

## Models

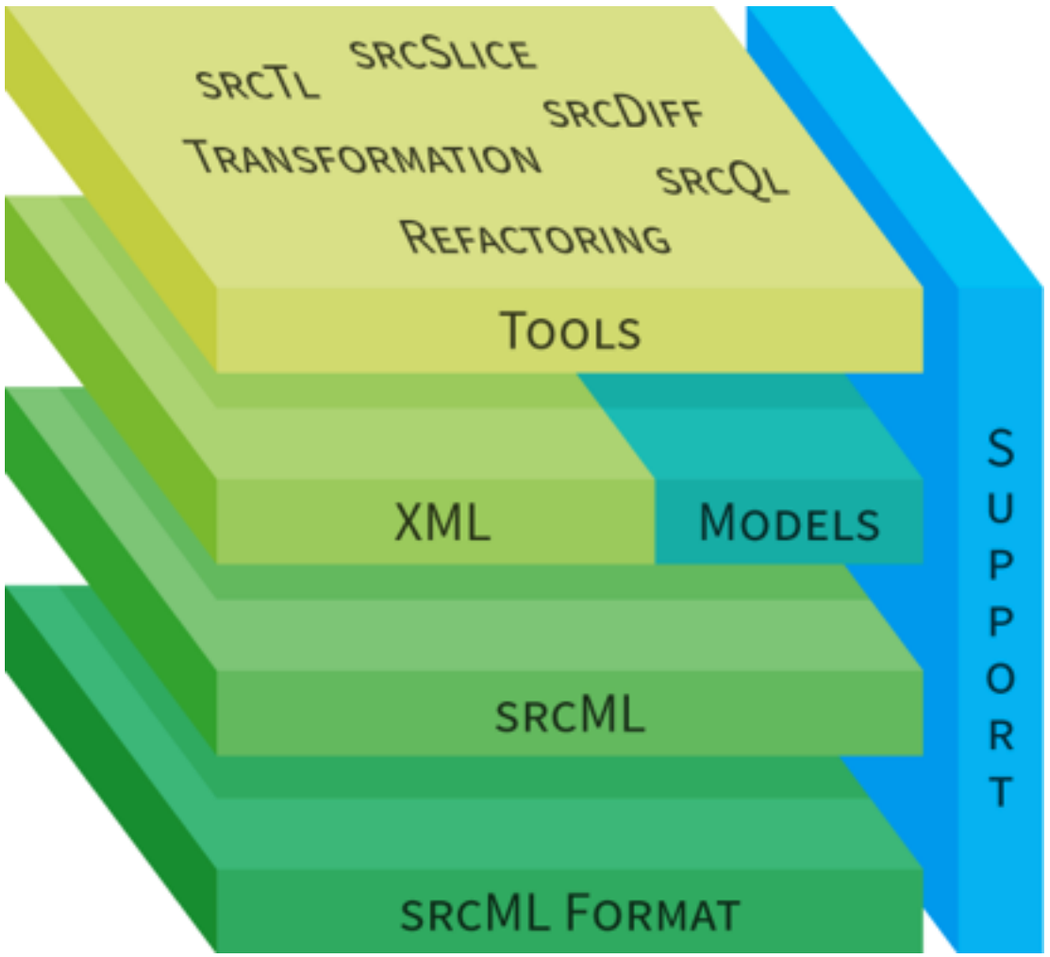
External models of the code such as PDG, UML, call graphs can be built in XML

## XML

The full range of XML technologies can be applied to the srcML format.

## srcML

The srcml CLI is used to convert entire projects from and to source code and the srcML format. Languages supported include C, C++, Java, and C#.




## srcML Format

The srcML format represents source code with all original information intact, including whitespace, comments, and preprocessing statements.

## Support

A multi-university team currently supports the infrastructure.

# Applications of srcML

- Fact extraction, analysis, computing metrics
- Refactoring, Transformation
- Syntactic Differencing
- Slicing
- Reverse engineering UML class diagrams, method/class stereotypes
- C++ preprocessor analysis
- Reverse engineering C++ template parameter constraints

# *srcml 1.0*

- (New) client srcml with C API libsrcml
- Freeze and version srcML tags
- Cross-linked documentation
- Multithreaded translation for large projects:

```
%srcml linux-3.16.tar.xz –o linux-3.16.xml.gz
```

  - Macbook Air: ~7 minutes
  - Mac Pro 6 Core: ~2 minutes

# Simple Examples

```
%srcml -l C++ --text "a = a + 1;"

%srcml foo.cpp -o foo.cpp.xml

%srcml linux-3.16.tar.xz -o linux-3.16.xml.gz
```

# Developing with srcML

- foo.cpp ➡ srcml ✚ XPath
- foo.cpp ➡ srcml ➡ foo.cpp.xml ➡
  - XML Tools (e.g., XSLT, XPath)
  - your code ✚ libxml2
  - srcSAX
- foo.cpp ➡ your code ✚ libsrcml ➡
  - XML Tools (e.g., XSLT, XPath)
  - your code ✚ libxml2
  - srcSAX

# Query srcML with XPath

- Names of all functions that include a direct call to malloc():

```
%srcml --xpath="//src:function[.//src:call/
src:name='malloc']/src:name" linux-4.0.3.tar.xz.xml -o
function_names.xml
```

- Result: srcML Archive with <unit> for each function name
- Good for collecting results in isolation
- Also able to mark in context with a specified attribute or element

# Tools (beta release)

- srcSlice - highly scalable forward static slicer
- stereoCode - method/class stereotypes
- srcType - type resolution
- srcYUML - generates UML from source

# Tools (in the works)

- srcMX - GUI for working with srcML
- srcDiff - syntactic differencing
- srcQL - source code query language
- incremental call graph generator
- pointer analysis
- source code POS tagger

# Future

- Domain Specific Languages (DSLs)
- Objective-C, Swift
- Full internal pipeline

srcML.org