<srcML>

Source code analysis & manipulation via AST

Slides adapted from Michael Collard (srcML.org), Jonathan Maletic (SANER 2020), and Drew Guarnera

What is srcML?

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

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

Community

Current developers

- **Michael L. Collard**
- Jonathan I. Maletic
- Michael Decker
- Christian Newman
- Drew Guarnera
- Vlas Zyrianov

Past contributors

- Brian Bartman
- Paul PJ Leyden
- Mike Weyandt
- Heather Guarnera
- Brian Kovacs
- Patricia Jordan
- Alyssa Myers
- Tessandra Sage
- Kyle Swartz

History

2000 **Good idea!**

2002 IWPC'02 Prototype

2003 IWPC'03 Better Prototype

2004 Initial Release C/C++

2008 ABB Funding

09-12 Applications Bug Fixing New users Java

2013 NSF Funding MIP ICPC

2014 Alpha Version

2015 GitHub Beta Version C#

2016 Documentation

2020 srcML 1.0.0

17-19 Docker Tool Releases Build Management

Extraction via Parsing

- Must parse the source code (compiler)
- Result is an abstract syntax tree and symbol table
- Difficult to map AST (data) back to original source code (document)
- Programmers care about code, not the AST

Example: 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;
}
```

Example: 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 whitespace, comments, special characters
- Syntactic structure wrapped with tags, making them addressable
- Comments marked in place
- srcML element tags include (among many):
  - <if>, <then>, <else>, <elseif>, <while>, <for>, <break>, <continue>, <switch>, <case>, <default>, <block>
  - <specifier>
  - <decl>, <function>, <function_decl>, <typedef>
  - <struct>, <union>, <class>
  - <call>, <name>, <expr>, <operator>, <argument>, <parameter>

Implementation

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

- Current file speed: ~35 KLOC/second
- srcML to text: ~4.5 KLOC/second
- Multithreaded translation for large projects
  - Linux kernel: ~2 minutes

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

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

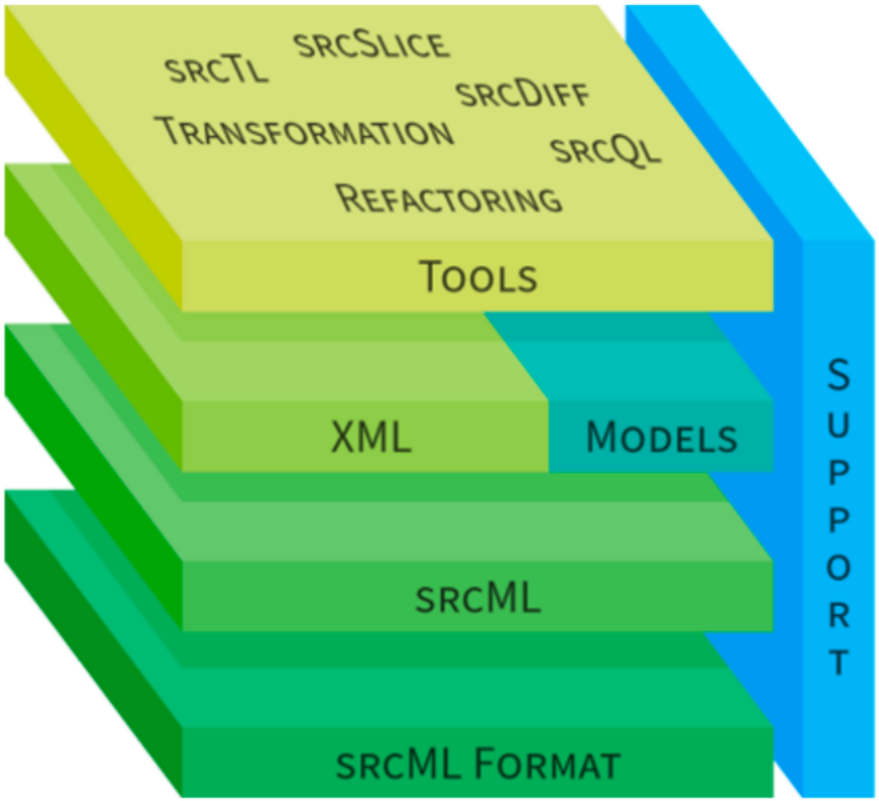
External models of the code such as PDG, UML, call graphs can be built in XML

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.

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
```

Leveraging srcML

- ICSM'04 - Syntactic differencing with Collard
- SET'04 - Refactoring using XSLT with Collard
- WCRE'05 - Reverse engineering UML class models with Andrew Sutton
- TEFSE'05 - Traceability with Bonita Sharif
- ICSM'06 - Reverse engineering method stereotypes with Natalia Dragan
- SCAM'06 - Factoring differences with Collard, Huzefa Kagdi
- ICSM'07 - C preprocessor analysis with Sutton
- ICSM'08 - C++ template analysis with Sutton
- ICPC'09 - Code to design traceability with Maen Hammad
- ICSM'10 - Reverse engineering class stereotypes with Dragan, Collard
- ICSM'10 - Transformations for large scale adaptive changes with Collard, Robinson

Others using srcML

- Birrer ’04 - XWeaver aspect weaver
- Binkley ’07 - Identifier analysis
- Stefik ’07 - Accessibility (for the blind)
- Hill ’07 - Program exploration
- Marcus ’08 - Metrics computation
- Tonella, Abebe ’08 - code quality
- Abebe ’09 - Source code vocabulary analysis
- Cleland-Huang ’09 - Traceability
- Jens ’09 - Quality assurance
- Corazza ’11 - Lexical information analysis
- Gethers ’12 - Information retrieval and traceability

Applications of srcML

- Fast extraction, analysis, computing metrics
- Refactoring, transformation
- Syntactic differencing
- Slicing
- Reverse engineering UML class diagrams, method/class stereotypes

Query srcML with XPath

Obtain names of all functions that include a direct call to malloc():

```
$ srcml --xpath="//src:function[.//src:call/
src:name='malloc']/src:name" linux.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 specific attribute or element

Example Application: srcDiff

Original	Modified
`KisPenWidget *m_optionsWidget;`	`KisPenWidget *m_options;`
srcDiff	
`KisPenWidget *~~m_optionsWidget~~m_options;`	

Original	Modified

`if (shape.has("anchor")) {`
`  string type = shape.get("anchor");`
`  if(type != "page") {`
`  Anchor *anchor = new`
`  Anchor(shape);`
`  anchor->loadShape(element);`
`  }`
`}`
	`string type = shape.get("anchor");`
	`if(type != "page") {`
	`  Anchor *anchor = new`
	`  Anchor(shape);`
	`  anchor->loadShape(element);`
	`}`

srcDiff	

`~~if(shape.has("anchor")) {~~`
`~~——~~string type = shape.get("anchor");`
`~~——~~if(type != "page") {`
`~~——~~Anchor *anchor = new`
`~~——~~Anchor(shape);`
`~~——~~anchor->loadShape(element);`
`~~——~~}`
`~~}~~`	

Example Application: srcQL

- Query language for source code that is syntactically aware
- Loosely modeled on SQL
- Example:
  - Find all functions
    ```
    FIND src:function
    ```
  - Find all functions that contain a call to new
    ```
    FIND src:function CONTAINS $T = new $X
    ```
  - Find all functions with a new and delete
    ```
    FIND src:function
         CONTAINS $T = new $X
         CONTAINS delete $T
    ```

Example Application: iTrace

```
public static void main(String[] args) {
  Scanner input = new Scanner(System.in);

  System.out.print("Enter a string: ");
  String s = input.nextLine();

  int[] counts = countLetters(s.toLowerCase());

  for (int i = 0; i < counts.length; i++) {
    if (counts[i] != 0)
      System.out.println((char)('a' + i) + " appears " +
        counts[i] + ((counts[i] == 1) ? " time" : " times"));
  }
}

public static int[] countLetters(String s) {
  int[] counts = new int[26];

  for (int i = 0; i < s.length(); i++) {
```

More Tools & Applications

- **srcSlice** – highly scalable forward static slicer
- **stereoCode** – method/class sterotypes
- **srcYUML** – generates UML from source code
- **srcMX** – GUI for working with srcML
- **srcDiff** – syntactic differencing
- **srcQL** – syntactic aware query language
- **srcTL** – transformation language
- **srcNLP** – parts of speech tagger for identifiers
- **iTrace** – eye tracking software

Interested?

Home page: https://www.srcml.org/

GitHub: https://github.com/srcML/srcML