Most Influential Paper Award
ICPC 2013

An XML-Based Lightweight C++ Fact Extractor
IWPC 2003

Michael L. Collard University of Akron
Huzefa Kagdi Wichita State University
Jonathan I. Maletic Kent State University

The Paper

srcML

- An XML format for the representation of source code
- All original lexical information from the original source code is preserved with the syntactic information marked using XML elements
- A document vs. data format

Code

```
#include "rotate.hpp"

// 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 in Code

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<unit xmlns="http://www.sdml.info/srcML/src" xmlns:cpp="http://www.sdml.info/srcML/cpp" language="C++">
<cpp:include>#<cpp:directive>include</cpp:directive> <cpp:file>"rotate.hpp"</cpp:file></cpp:include>

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

Back to Code

```
#include "rotate.hpp"

// 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;
}
```

Unique Features

- Use of XML as a document format, not as a data format/interchange (e.g., GXL)
- Preservation of all original source code text
- Preservation of preprocessor directives
- Source-code addressing
- Extensible format
- Robust and fast parsing

Parsing

- Wrote own parser for C/C++
- Built on pred-LL(K) parsing toolkit ANTLR 2
- Inspired by Island Grammars [Moonen'01]
  - Separate token streams for white space, comments, preprocessor
  - Selective markup
- For C/C++ takes CFG view, e.g., "a b(c);" assumed to be function declaration

Fact Extraction

- IWPC 2002 Fact Extraction Benchmark [Sim'02]
- Acacia, Columbus, Cppx, TkSee/SN
- XPath queries on srcML using XML tools
- Compared favorably

Historical Context

IWPC 2001

Parsing to Support Comprehension Tasks

- Growing interest in parsing tools
- Hacking g++
- Compiler perspective

IWPC 2002

- Paris, la Sorbonne

Introduced srcML

- Short paper on srcML format
- Session on benchmarks for fact extraction
- Talk of interchange formats for interoperability

Related Technologies

- TXL, Columbus, CPPX, Acacia, ASF+SDF
- JavaML, GXL

IWPC 2003

- Portland Oregon
- co-located with ICSE
- GC: Hausi Muller
- PC: Ken Wong, Rainer Koschke
- 21 full papers, 6 short

Social Events: (Singing and Dancing)

Keynotes

Paul Klint
Understanding vs Compiling

Jim Cordy
Barriers to Tool Adoption

Our Paper

- In session on Fact Extraction
- Techniques with Jim Cordy (Selective Markup)
- Anthony Cox & Charles Clarke (Syntactic Approximation)

Influence

- IWPC 2002 paper
  - Google scholar 94 citations
- IWPC 2003 paper
  - Google scholar 101 citations
- Used in over 25 thesis/dissertation
- Multiple industry users

2003 to 2013

srcML Toolkit

- src2srcml
- srcml2src
- Windows/Mac/Linux
- Feature/enhancements driven by users (industry/research)

src2srcml

- C, C++, Java (limited)
- Input: code fragment, file, directory, source-code archive
- srcML Archive for entire project
- Translation speed: 26KLOC/sec
- Input from URL

Scalability

- Designed for processing large projects
- E.g., src2srcml http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.38.3.tar.bz2 -o linux.xml
- Speed: ~8 minutes to convert entire Linux kernel
- Size: Entire Linux kernel in 1.4 GB file
- Reasonable file size: Typical srcML file is 4.5 times size of source-code text
- Even more important when processing version histories

srcml2src

- Fundamentally, converts srcML back to source code
- Output: file, directory, source-code archive
- Built-in XPath addressing
- Built-in XSLT processing

srcML Elements

Category	srcML Elements
File/Project	unit
Statement	if, then, else, while, do, switch, case, default, for, init, incr, condition, break, continue, comment, name, type, block, index, expr_stmt, expr, decl_stmt, decl, init, goto, label, typedef, asm, macro, enum, empty_stmt, namespace, template, using, extern
Function/Method	function, function_decl, specifier, return, call, parameter_list, param, argument_list, argument
Class	class, class_decl, public, private, protected, member_list, constructor, constructor_decl, destructor, destructor_decl, super, friend
Struct and Union	struct, struct_decl, union, union_decl
Exception	try, catch, throw, throws
C-Preprocessor	cpp:directive, cpp:file, cpp:include, cpp:define, cpp:undef, cpp:line, cpp:if, cpp:ifdef, cpp:ifndef, cpp:else, cpp:elif, cpp:endif, cpp:then, cpp:pragma, cpp:error
Java	extends, implements, import, package
Optional Markup	literal, operator, modifier
Misc	escape

Using srcML

Addressing

- Relatively straightforward to address specific locations in code

```
//src:comment[contains(.,'Collard')]

//src:if[
    src:then//src:call[src:name='rotate']
]
```

Redocumentation

- Analyze functions/methods to determine stereotypes, e.g., @get, @set, @predicate, @command, etc.
- Redocument methods with stereotypes
- [Dragan, Collard, Maletic'06]

Method Stereotype Pattern

```
src:function[
    src:specifier='const' and
    src:type/src:name='bool' and
    (src:data_members() or
        (src:real_call()[src:is_pure_call() and
            (not(src:name/op:operator='::')
             or src:name/src:name[1]=src:class_name())
        or src:calling_object()[src:is_data_member()]])
    ) and src:return()[.='false' or .='true' or
             src:call[1]
             or count(src:name)!=1 or
             *[2][self::op:operator] or
             src:primary_variable_name(src:name)
                 [src:is_declared()]][1]
]
```

API Migration

[Collard, Maletic, Robinson'10]

```
CNICmdFactory *cmdFactory = new CNICmdFactory;
```

```
CNICmdFactory *cmdFactory;
try {
   cmdFactory = new CNICdmFactory;
} catch (...) {
   cmdFactory = NULL;
}
```

Transformation

```
<xsl:template match="src:decl_stmt[src:decl/src:init/src:expr/op:operator='new']">

  <!-- Copy the declaration, without any part of the initialization -->
  <xsl:copy-of select="src:exclude(., src:decl/src:name/following-sibling::node())"/>

  <!-- Wrap a try catch around the initialization of the variable -->
  <xsl:variable name="trycatch">
try {
  <xsl:copy-of select="src:decl/src:name | src:decl/src:name/following-
sibling::node()"/>;
} catch (...) {
  <xsl:value-of select="src:decl/src:name[1]"/> = NULL;
}</xsl:variable>

  <!-- Copy the generated try catch with the indentation of the original statement -->
  <xsl:copy-of select="src:indent(src:indentation(.), $trycatch)"/>

</xsl:template>
```

XPath

```
.//src:if[src:condition/op:operator='new']
```

LINQ

```
var ifStmts = from stmt in doc.Descendants
(SrcML.SRCNS + "if")
where stmt.Element(SrcML.SRCNS +
"condition"). Descendants (SrcML.OPNS +
"operator").Any(x => x.Value == "new")
select stmt;
```

The Future

Realities

- Very hard to build real software in an academic setting
- Students graduate (hopefully)
- Heroics of individuals

srcML - a community infrastructure

Extended Language Support

- C+ - 11
- Java - Full support for Java 5, 6, and 7
- C# - full support, including LINQ

C++11

- rvalue references
- template alias
- initializer lists
- explicit defaulting and deletion
- lambda functions
- new function declaration syntax
- base class constructor inheritance
- user-defined literals
- variadic templates

Language Evolution

- Maintenance of existing languages is problematic
- Want support for other languages, including Domain Specific Languages
- Need a new architecture

Plugin Grammars

- Current architecture is heavily hacked ANTLR
- Define DSL in ANTLR grammar, with some metadata
- Specific srcML support is created by using an ANTLR srcML target
- Build plugin for this grammar

Tools!!

- Exploration
- Analysis
- Manipulation

Exploration

- srcQuery
- srcQL

Analysis

- Type resolution
- Slicing
- Call graph
- Reverse engineering (e.g., UML)

Manipulation

- Refactoring
- Transformation

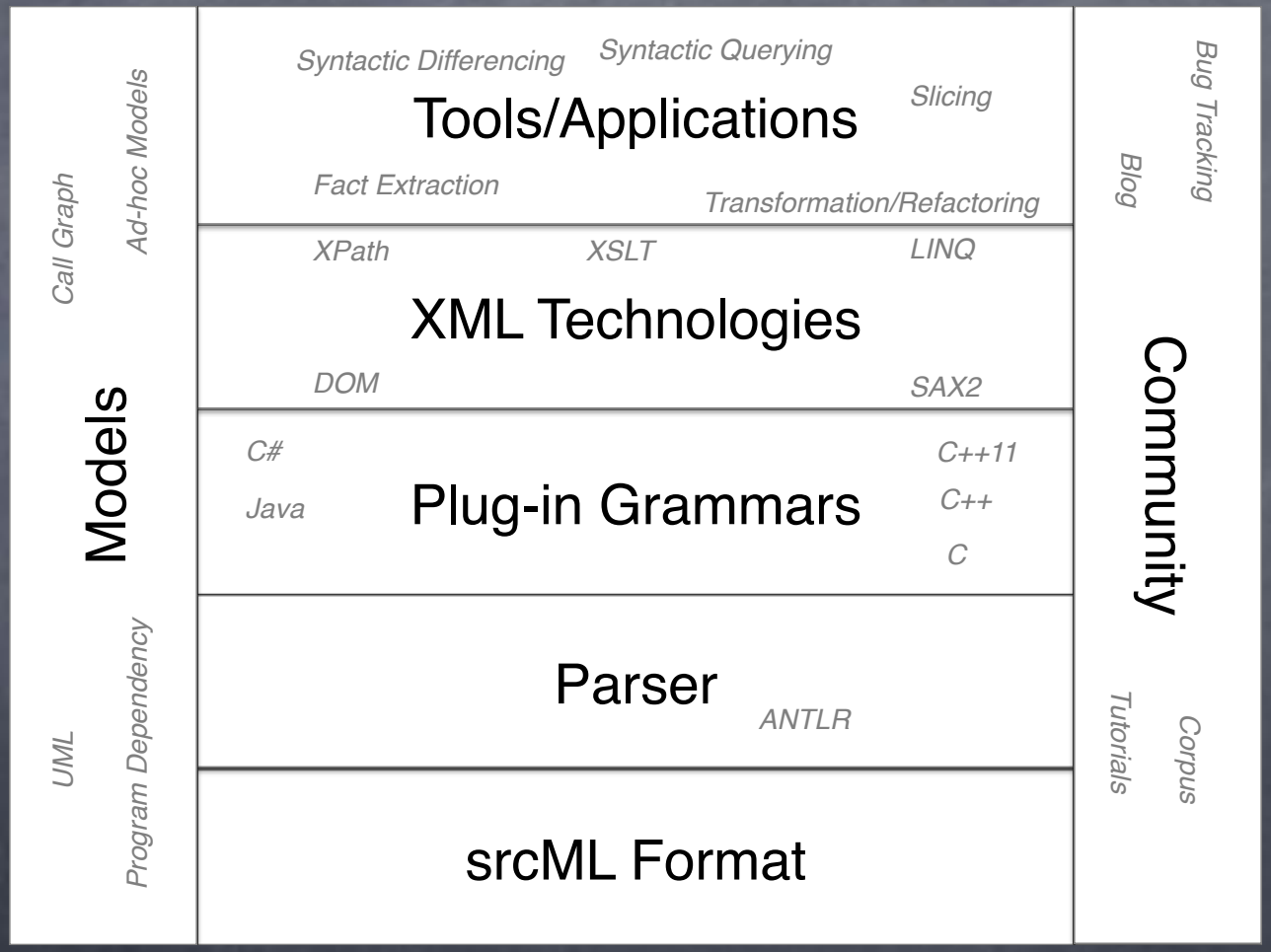
Syntactic Differencing

- srcDiff

Community

- Open source
- Bug tracking
- Tutorials
- Demonstrations
- Documentation

srcML


Models
Call Graph
Ad-hoc Models
UML
Program Dependency
Syntactic Differencing
Syntactic Querying
Tools/Applications
Slicing
Fact Extraction
Transformation/Refactoring
XPath
XSLT
LINQ
XML Technologies
DOM
SAX2
C#
Java
Plug-in Grammars
C++11
C++
C
Parser
ANTLR
srcML Format
Community
Bug Tracking
Blog
Corpus
Tutorials

Other Contributors

- ABB Inc., Tira Wireless
- Michael Decker, Andrew Sutton, Paolo Tonella, Giulio Antoniol, Andi Marcus, Brian Robinson, Dave Shepherd

srcML.org