<SDML>

Stereocode: A Tool for Automatic Identification of Method and Class Stereotypes for Software Systems

Ali Al-Ramadan[1], Joshua Behler[1], Michael Decker[2],
Natalia Dragan[1], Michael Collard[3], Jonathan Maletic[1]

[1]Kent State University, [2]Bowling Green State University, [3]The University of Akron









What are Stereotypes?

- Stereotypes are abstractions that capture the high-level design and behavior of software

- **Method stereotype:** describes the intended behavior and role of a method
- **Class stereotype**: characterizes the overall purpose and responsibility of a class

- Used in various research related to program comprehension and maintenance:
  - Code smell detection
  - Generating commit messages
  - Enhancing feature location
  - Generating natural-language summaries for methods
  - And more!

Method Stereotypes [Dragan et al. ICSM 2006, 2009, 2010]

Stereotype Category	Stereotype	Description
Structural Accessor	**get**	Returns a data member
	predicate	Returns a Boolean value that is not a data member
	property	Returns information about data members (non-Boolean)
	void-accessor	Returns information about data members through method parameters
Structural Mutators	**set**	Modifies a data member
	command	Performs a complex change to the object's state
	non-void-command	
Creational	**constructor**	Creates and/or destroys objects
	copy-constructor	
	destructor	
	factory	
Collaborational	**collaborator**	Works with objects belonging to classes other than itself (parameter, local variable, data member, or return value)
	controller	Changes only an external object's state (not this)
	wrapper	Does not change an object's state. Has at least one free function call
Degenerate	**incidental**	Does not read/change an object's state. No calls to other class methods or to free functions
	stateless	Does not read/change an object's state. Has at least one call to other class methods or to a free function
	empty	Has no statements

Method Stereotypes - HippoDraw

```
// @stereotype get
double AxisModelBase::getFirstTick() const
{
  return m_first_tick;
}

// @stereotype set
void AxisModelBase::setAutoRanging ( bool flag )
{
  m_auto_range = flag;
}

// @stereotype command
void AxisModelBase::setScaleFactor ( double sf )
{
  m_scale_factor = sf;
  m_scaling_on = true;
}
```

Class Stereotypes [Dragan et al. ICSM 2006, 2009, 2010]

Class Stereotype	Description
entity	Encapsulates data and behavior. Keeper of data model and/or business logic
minimal-entity	Special case of Entity. Consists only of get, set, and command methods
data-provider	Encapsulates data and consists mainly of accessor methods
commander	Encapsulates behavior and consists mainly of mutator methods
boundary	Communicator with a large percentage of collaborational methods and a low percentage of controller methods. It also does not have many factory methods
factory	Creator of objects and has mostly factory methods
controller	(aka control) Provides functionality to control external objects. Consists mostly of controller and factory methods
pure-controller	Special case of controller. Consists only of controller and factory methods
large-class	Contains a large number of methods that combine multiple roles, such as Data Provider, Commander, Controller, and Factory
lazy-class	Consists mostly of get, set, and degenerate methods. Occurrence of other methods is low
degenerate	Consists mostly of degenerate methods that do not read/write to the object's state
data-class	Consists only of get and set methods
small-class	Consists only of one or two methods
empty	Has no methods

Class Stereotypes - PowerShell

```
// @stereotype boundary degenerate
public class ResumeController : Controller {

  // @stereotype command
  public async void NoResume();

  // @stereotype controller stateless
  private bool TryGetRangeHeader(out string rangeHeader);

  // @stereotype wrapper
  private void SetResumeResponseHeaders();

  // @stereotype command collaborator
  public async void Index();

  ...
}
```

Challenges in Stereotype Detection

- Existing automated approaches are limited in functionality
  - Perform basic analysis
  - Stereotype one source file at a time
  - Support a single language (C++ or Java)
  - Struggle to identify certain rare stereotypes
  - Difficult to use or install

- Need for accurate and language-agnostic tools to automate stereotype identification at large scale

stereocode

- A static analysis tool engineered to automatically identify, and re-document software systems with method and class stereotypes.

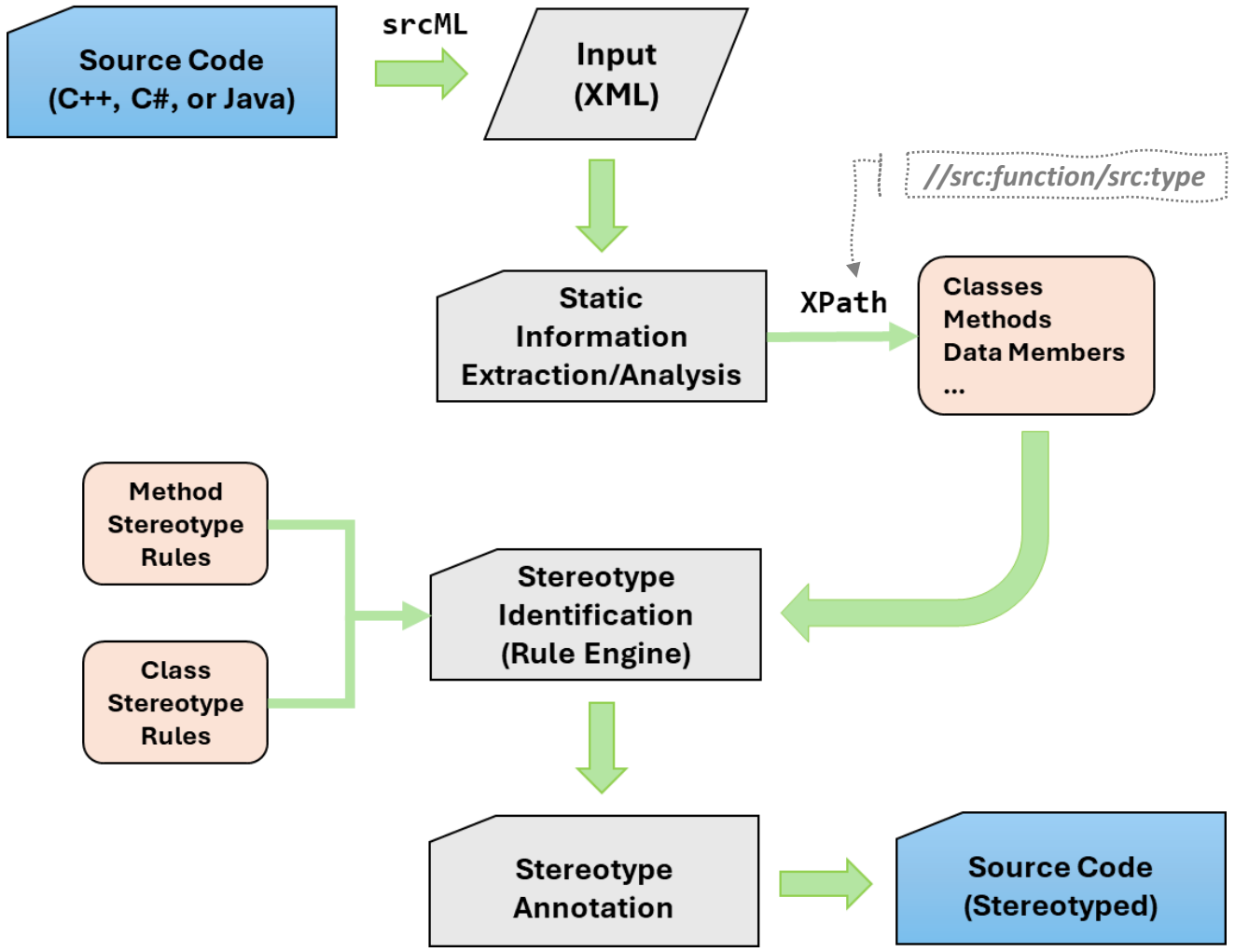
- Leverages srcML [srcML.org]



- Stereocode can identify stereotypes for both individual source files and complete software systems

- Supports C++, C#, and Java systems (including multi-language)

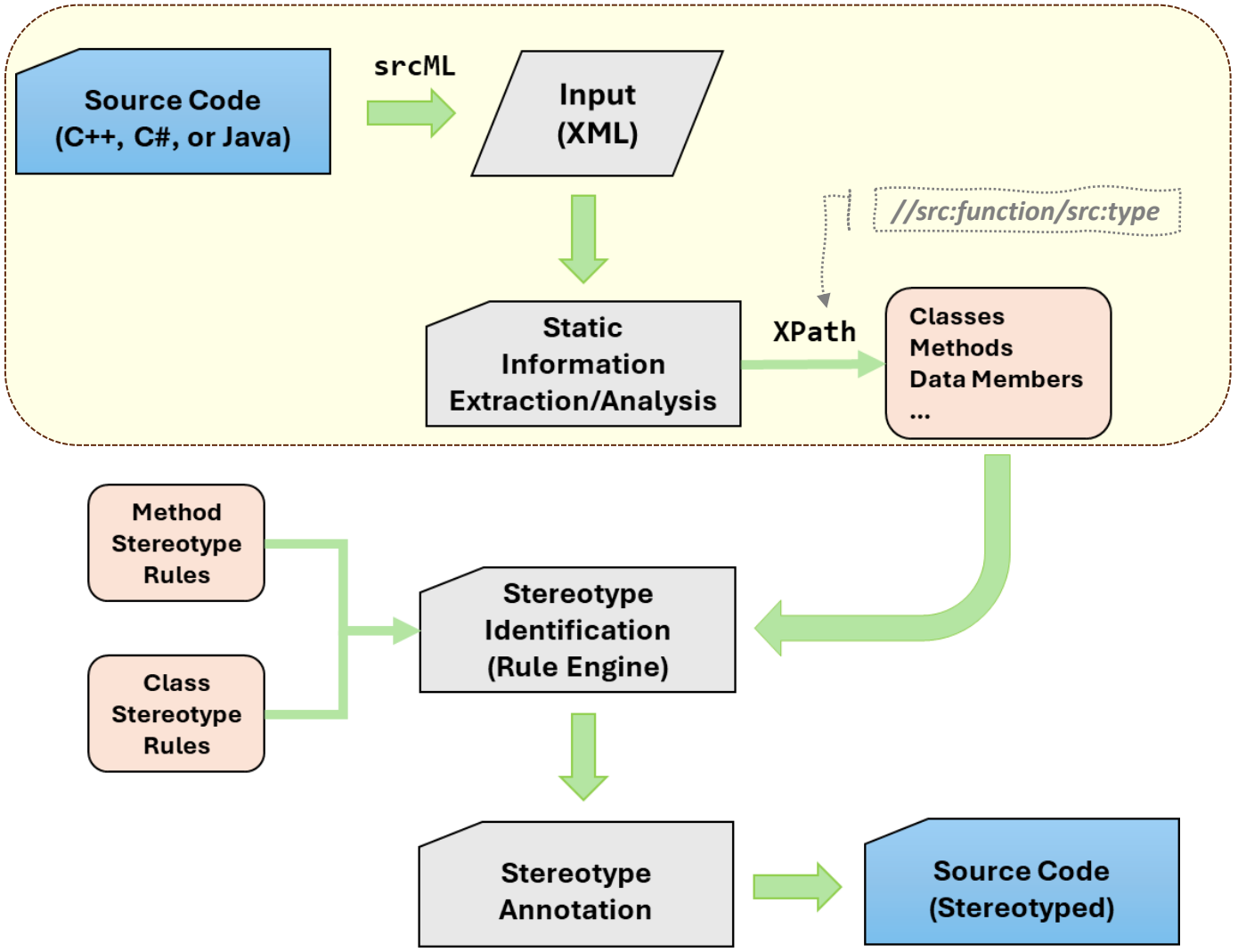
Stereotyping Process



Collect Program Information

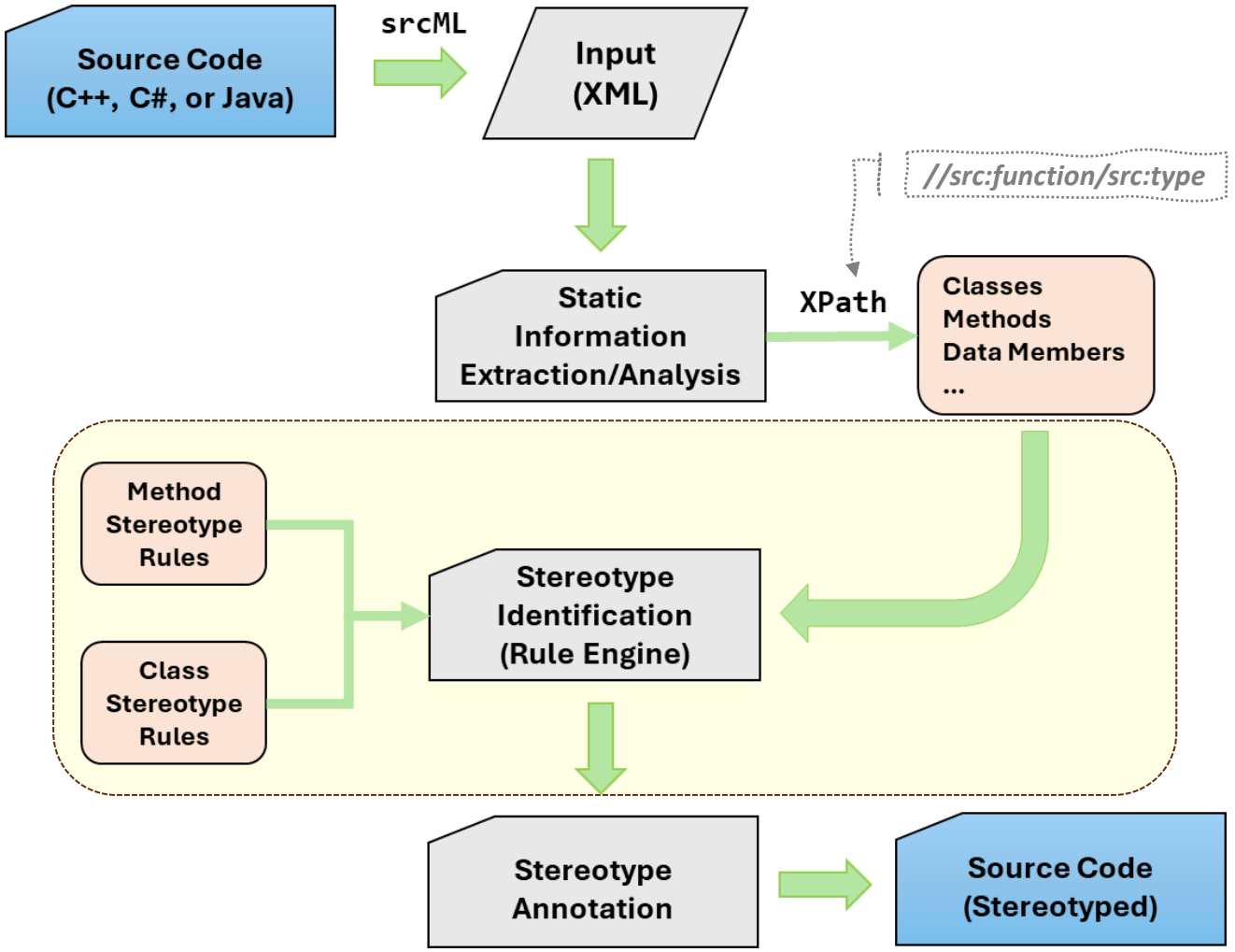
- Convert source code to the srcML format

- Extract detailed program information from code:
  - Classes, methods, attributes, and inherited info
  - Uses of locals and attributes
  - Calls
  - Data types
  - …

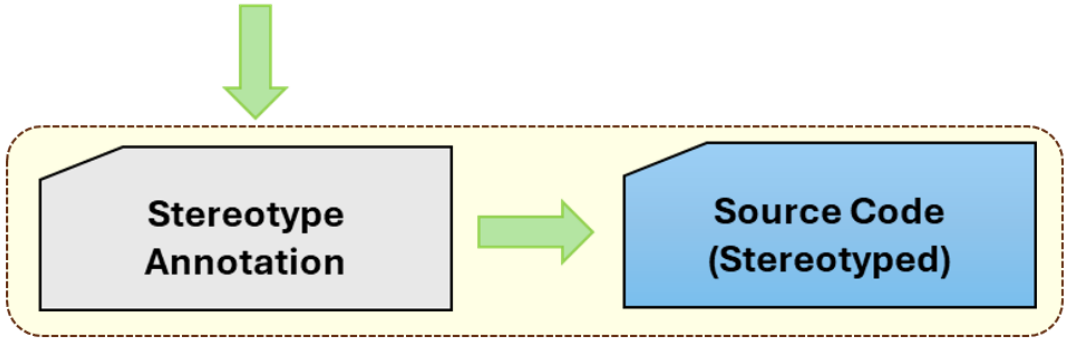


Stereotype Identification

- Method stereotypes are determined by a set of rules
- Class stereotypes are determined by the frequency distribution of their method stereotypes



Stereotype Annotation

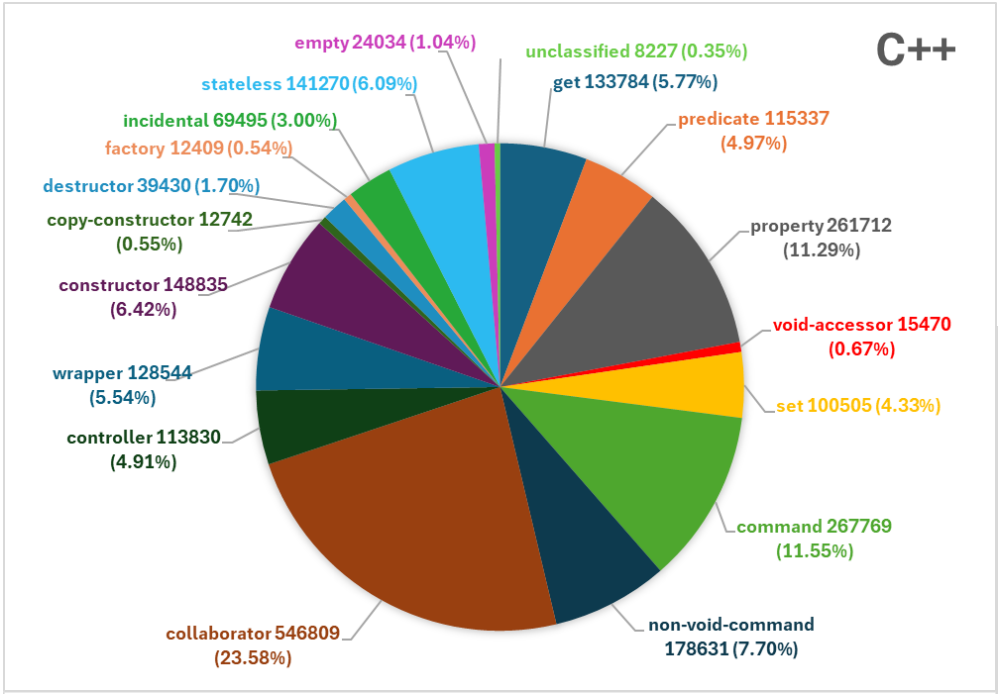


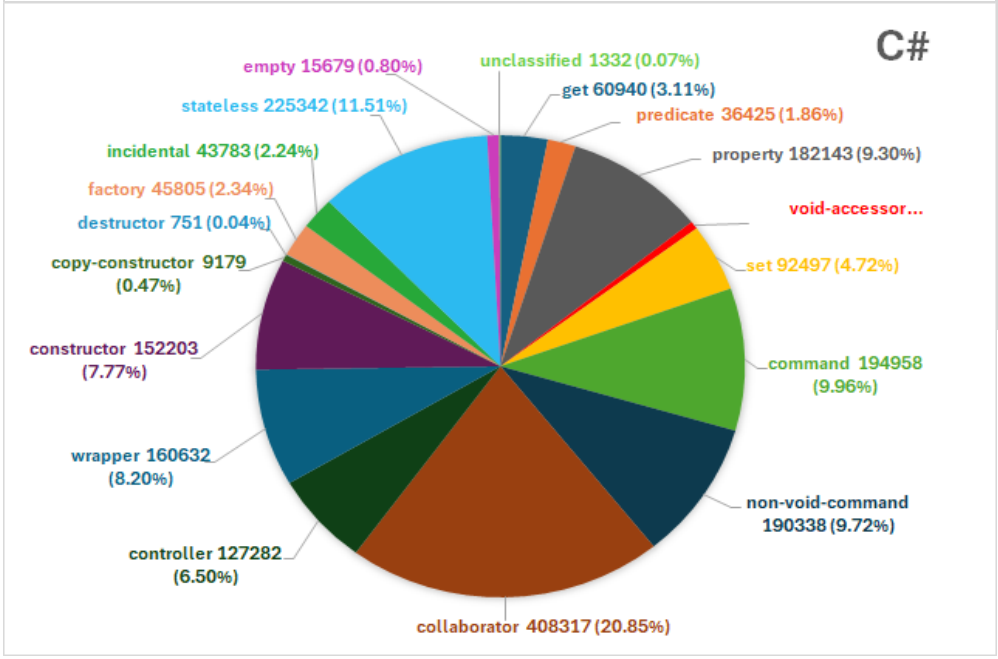

```
<function st:stereotype="get"><type><name>int</name></type>
<name>getID</name><parameter_list>()</parameter_list>
<block>{<block_content>
<return>return<expr><name>id_</name></expr>;</return>
</block_content>}</block></function>
```

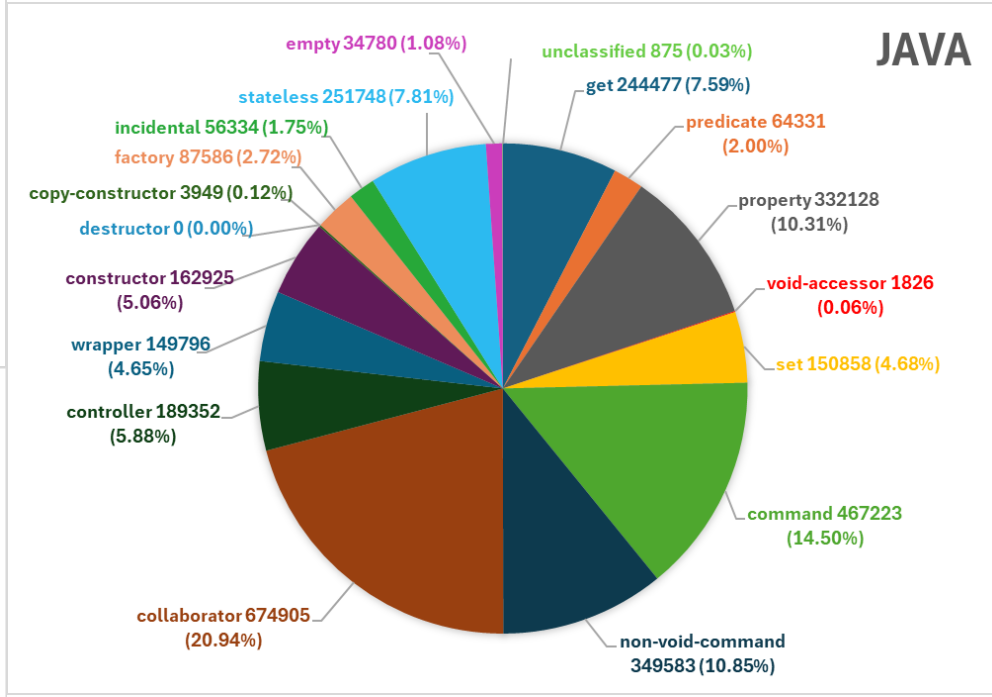
Scalability

- Stereocode applied to 1050 open-source software systems
  - 350 each for C++, C#, and Java
- Systems are converted to srcML format in less than a second
- Stereotyped 60% of systems in under 6 seconds
- Average stereotyping time is 19 seconds for all 1050 systems

Distribution of Method Stereotypes







srcML.org | Discord server



Running *Stereocode*

- *Stereocode* is a command-line tool that runs on a srcML formatted input
- The input, either an individual source code file or a full system, is converted to srcML
- Then, *Stereocode* is run on that srcML:

```
srcml main.cpp -o main.cpp.xml
stereocode main.cpp.xml -o stereotypes.cpp.xml
```

- *Stereocode* needs to be compiled
  - It is developed in C++ and built via CMake
  - Instruction to compile and run are in the GitHub repo[1]
- *Stereocode* runs on Mac, Windows, and Linux

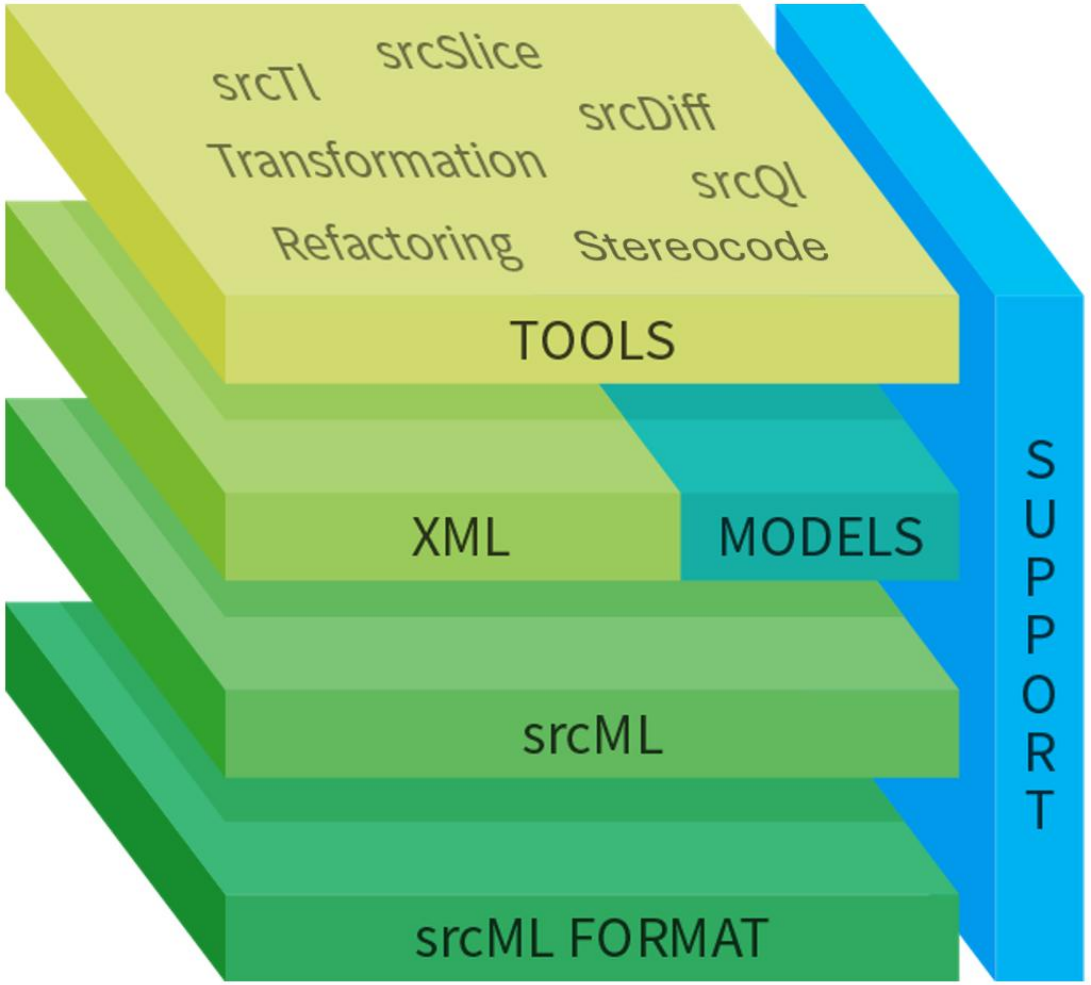
[1] https://github.com/srcML/stereocode

srcML

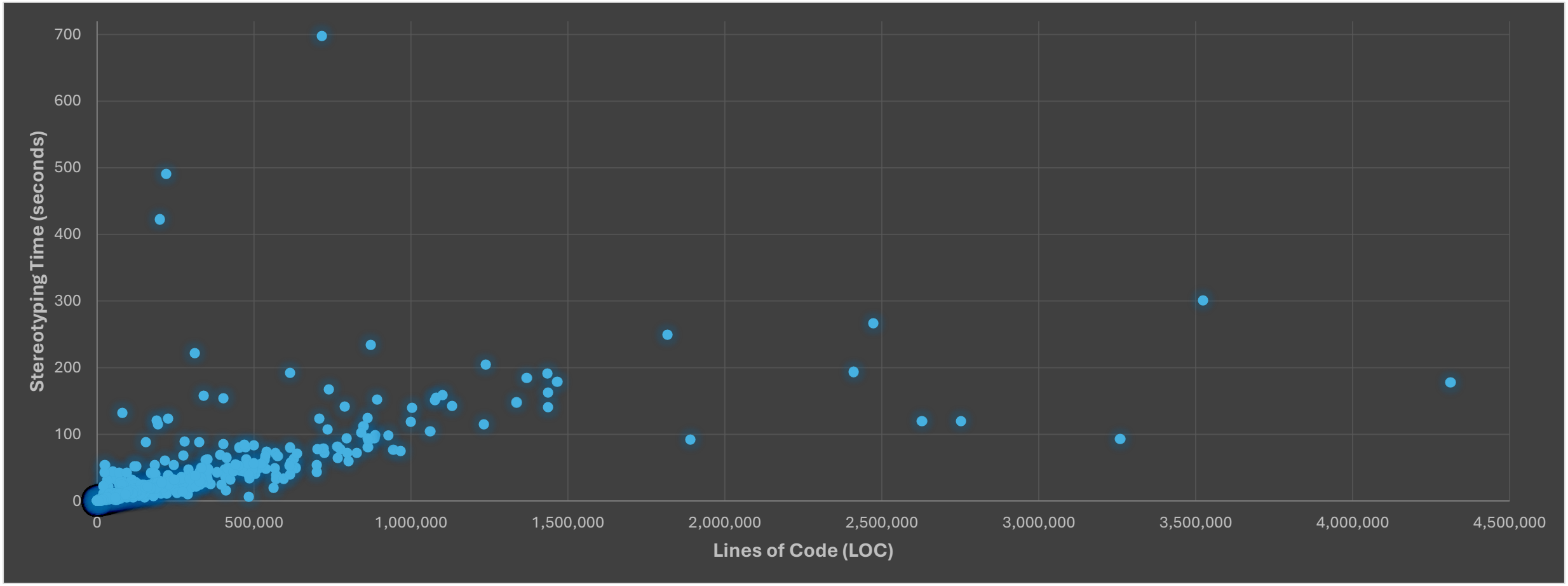
- srcML format is an XML representation of the source code that preserves all original information

- `srcml` is a robust tool to convert source code into XML (srcML) format

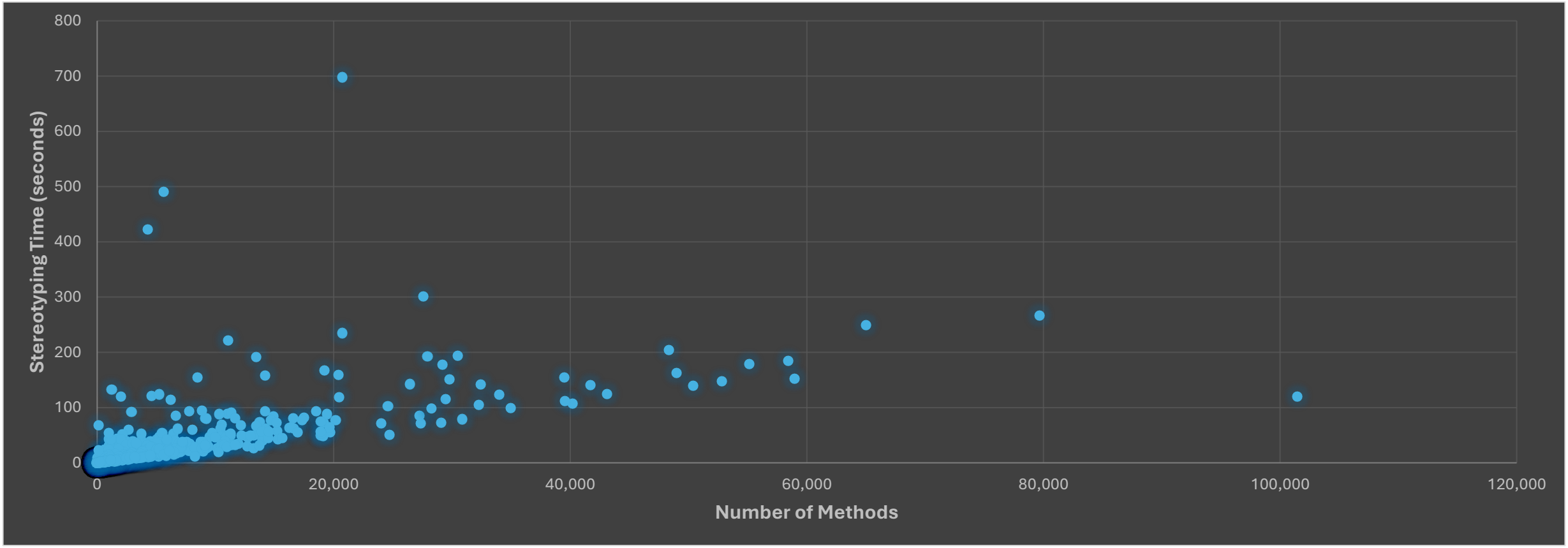
- srcML currently supports C, C++, C#, and Java



Stereotyping Time vs Code Size



Stereotyping Time vs Method Count



Metrics

Mean	Median	Mode	90th Percentile	Variance	Standard Deviation
19.65307848	5.4942	0.6228 0.217 1.9803	49.53	1981.120435	44.5097791