

# Cross-Language Program Analysis and Refactoring

**Dennis Strein, Hans Kratz**

Omnicores, Germany

**Welf Löwe**

Växjö University, Software Technology  
Group, MSI, Sweden



# Introduction

- “Real” software systems are **mixed-language**
    - e.g. Web-application: HTML + JavaScript + C# + VB
    - e.g. Java program: Java + Java Byte-code
  - Tools are usually designed for **single-language** systems
    - e.g. a Java refactoring tool
    - e.g. a C++ visualization tool
- We need better tool support for mixed-language systems

# Code Analysis and Refactoring Tools

- e.g. *“Plot a call-graph of a program”*

mixed-language software → cross-language graph

- e.g. *“Rename method A to B and update all calls”*

mixed-language software → method A may be used across languages

# Example

```
// An ASP web page file
<%@ Page language="c#" ClassName="WebForm">
<html>
  <head>
    <script language="c#" runat="server">
      public Button myButton;
      private void Init(string text) {
        myButton.Text = text;
      }
    </script>
  </head>
  <body>
    <asp:button id="myButton" runat="server">
    </asp:button>
  </body>
</html>
```

```
// A VisualBasic file
Public Class VBClient
  Sub SetText(f As WebForm, text As String)
    f.myButton.Text = text
  End Sub
End Class
```

- ASP.Net web-application example
- Mixes: ASP, HTML, C#, VisualBasic
- “myButton” is used in C#, VB and ASP source code

→ model syntactic language mixes

→ model semantic cross-language relations

# Real-world Example

- Project “ITextDotNet”, 251.225 lines of code
- Programming languages - C#, J# and VB
- Rename method `com.lowagie.text.Document.add(Element element)`

→ Find correct **886** of **2739** occurrences of “add”

|                    | Method Declarations |            | Call Expressions |            | Unrelated Textual Occurrences (irrelevant) | Total |
|--------------------|---------------------|------------|------------------|------------|--------------------------------------------|-------|
|                    | relevant            | irrelevant | relevant         | irrelevant |                                            |       |
| J#                 | 7                   | 80         | 420              | 795        | 479                                        | 1781  |
| C#                 | 0                   | 0          | 374              | 249        | 136                                        | 759   |
| VB                 | 0                   | 0          | 85               | 66         | 48                                         | 199   |
| Whole system (sum) | 7                   | 80         | 879              | 1110       | 663                                        | 2739  |

# Architecture

---

- **Common meta-model:** Captures cross-language whole-system models
- **Language-specific front-ends:**
  - Implement language-specific analysis, i.e. parsing and semantic analysis
  - Store information in the common model
  - Semantic analysis based on the common model to access other languages
- **High-level analysis and Refactorings:** Use the common model as a source of information

# Common Model

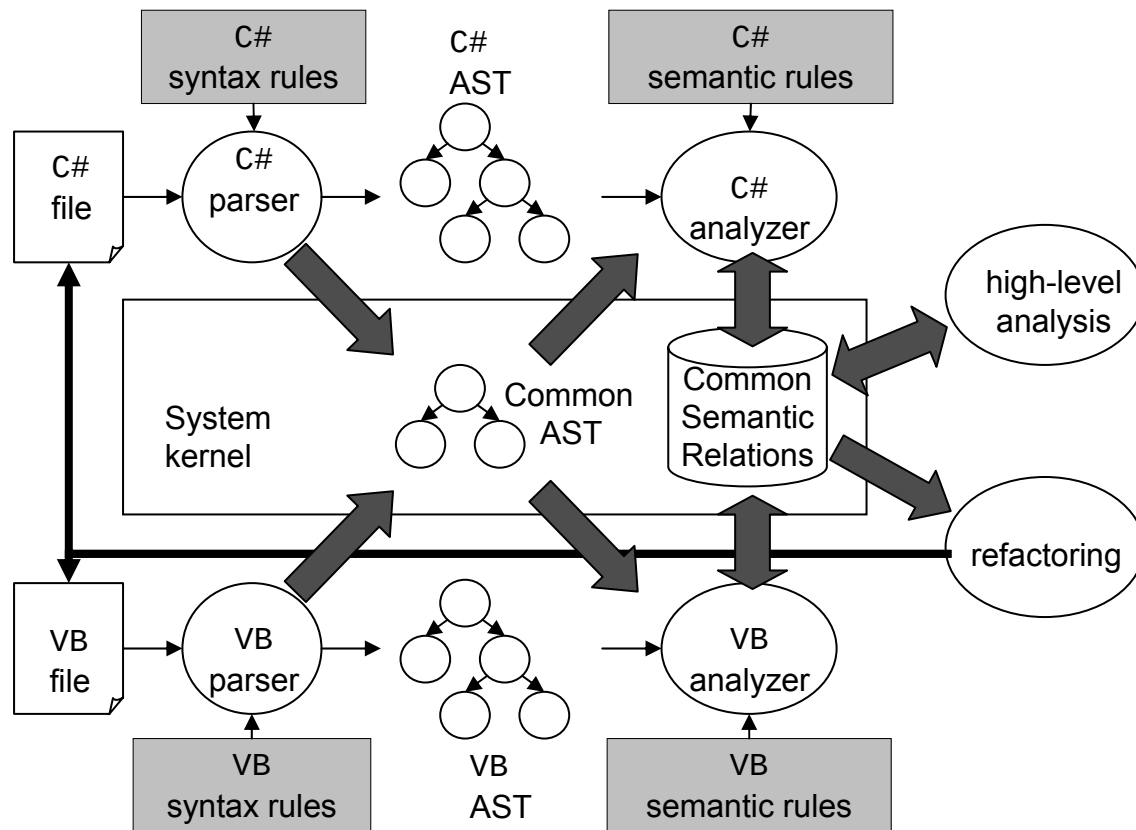
---

Very hard (impossible?) to get a fixed language-independent model

→ Common model must be extensible

Compare ICSM 06 Paper “An Extensible Meta-Model for Program Analysis”

# Architecture





# X-develop

---

- Mixed-language development environment
- Developed by Omnicore Software, Germany ([www.omnicore.com](http://www.omnicore.com))
- Current front-ends: Java, Java Byte-code, C#, VisualBasic.Net, J#, .Net Byte-code, HTML, XML, ASP, JSP, JavaScript
- “Instant” code analysis

**Questions?**

