

Columbus - Tool for Reverse Engineering Large Object Oriented Software Systems



*Rudolf Ferenc¹
Ferenc Magyar¹
Árpád Beszédes¹
Ákos Kiss¹
Mikko Tarkiainen²*

¹ Research Group on Artificial Intelligence, University of Szeged

² Nokia Research Center

Introduction

- Software systems are rapidly growing and are getting more and more complex
- As a result of this growth there is a need to understand the relationships between the different parts of a large system
- Existing legacy code and high number of participants in code development also necessitates the use of tools for *reverse engineering*
- Reverse engineering is “the process of analyzing a subject system to (a) identify the system’s components and their interrelationships and (b) create representations of a system in another form at a higher level of abstraction” [Chikofsky et al.]

Assessment of RE tools

- Analysis
 - Parsable source languages
 - Project definition types (ease of project definition)
 - Incremental parsing
 - Fault tolerant parser
 - Parse speed
- Representation
 - Speed of generation
 - Filters, scopes, grouping
 - Sorting
 - Layout algorithms
- Editing/browsing
 - Integrated text editor/browser
 - External editor/browser
- General capabilities
 - Toolset extensibility
 - Storing capabilities
 - Output capabilities

[Bellay, B. and Gall, H.]

Columbus/CAN

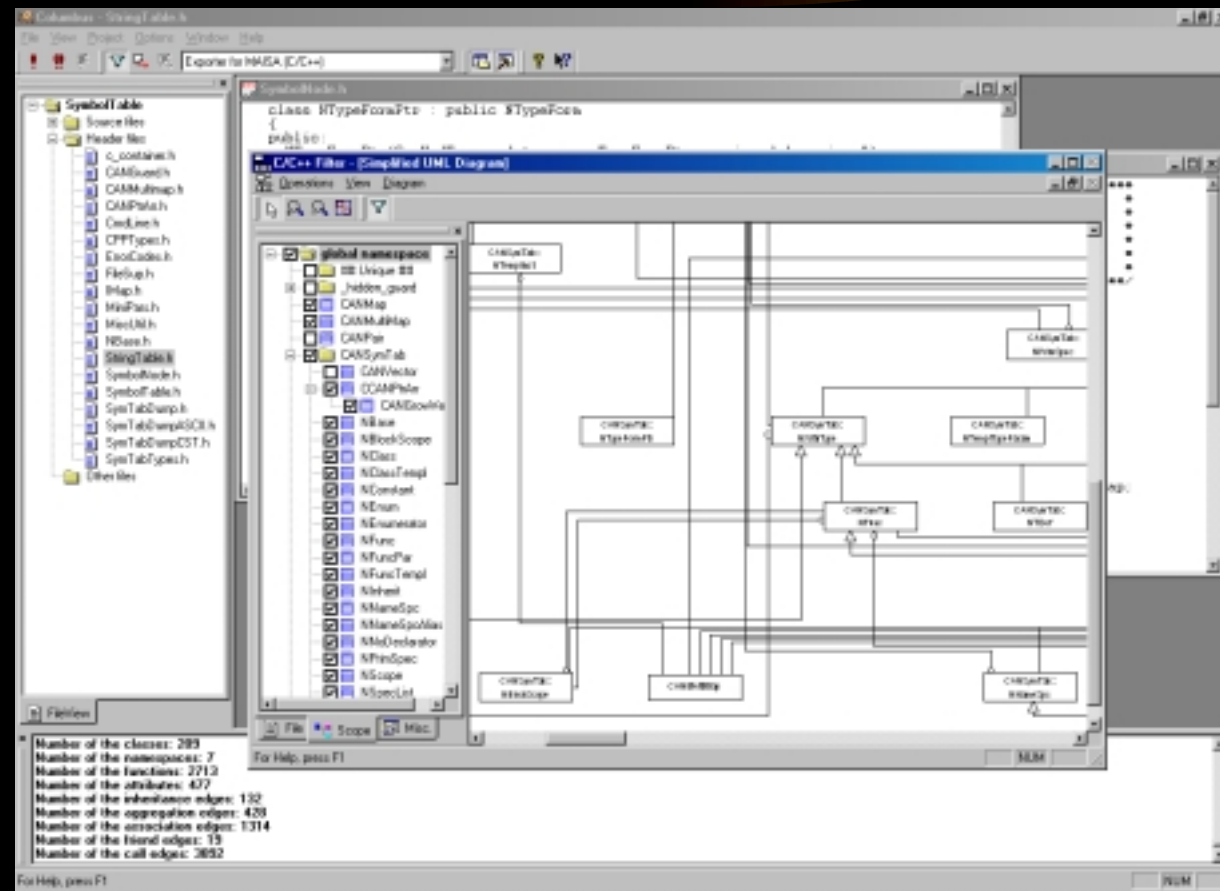
- Tool for *reverse engineering* large object oriented systems
- Developed in a cooperation between the *Nokia Research Center & University of Szeged*
- Fast, fault tolerant C/C++ parser
- Extracts the *UML class diagram* and the *call graph*

Columbus/CAN (cont.)

- Supports project handling, data extraction, data representation, data storage, filtering and visualization
- Follows the standard C++ compilation model
- High extensibility due to the plug-in architecture
- Easy to use API for writing third party plug-ins

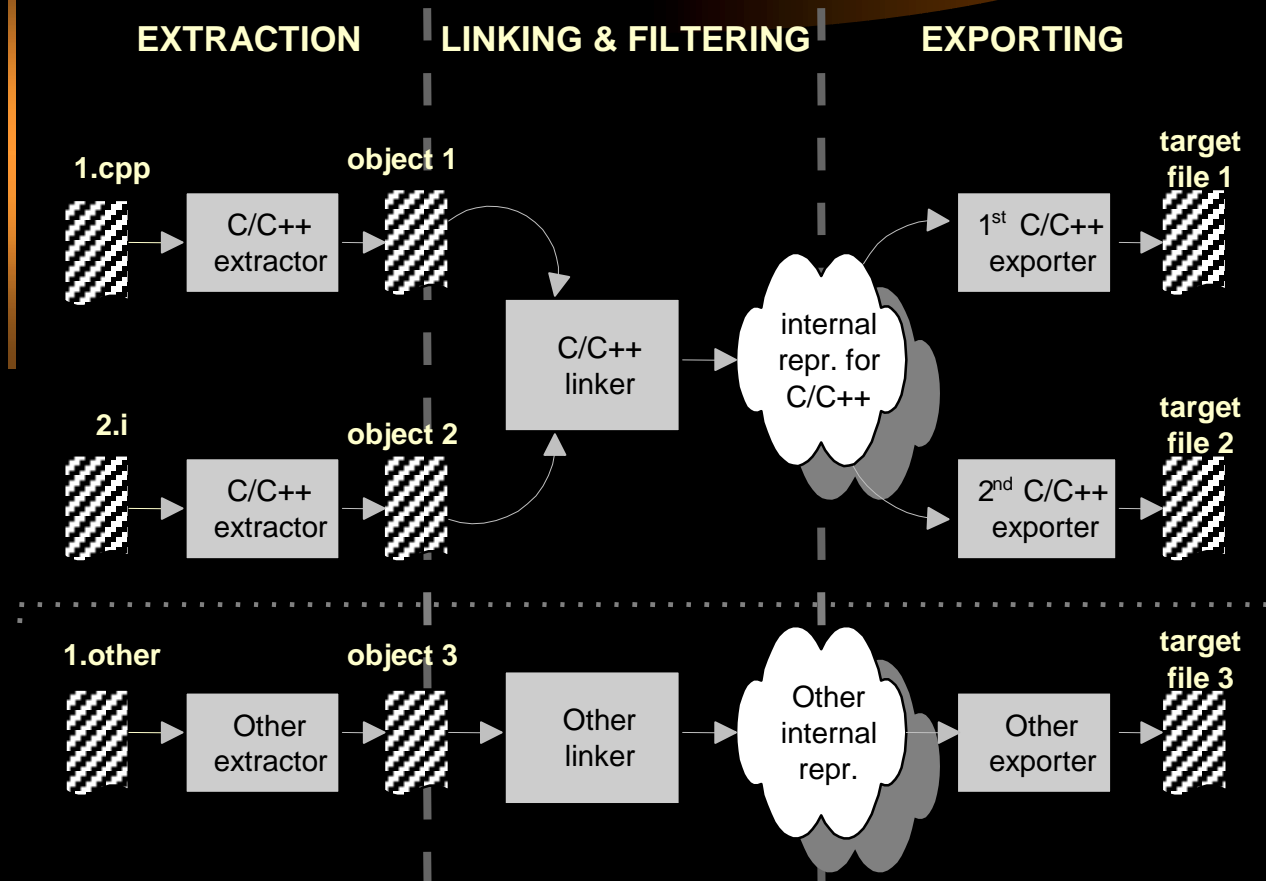
User Interface

- IDE like user interface



Overview of the Columbus System

- Extractor-,
Linker-
and
Exporter
plug-in-s



CAN – the C++ Analyzer

- Extractor plug-in for Columbus
- Command-line tool (can be integrated into makefiles)
- Fault tolerant (error recovery)
- Use of precompiled headers
- Fast: about *6800 non-empty LOC/sec!* (PIII-800 machine)
- Two-pass parser for template-handling
- Instantiation of templates at source level

Instantiation of Templates

- Small example:

```
template <class T>
class A {
    T a;
};

char c;
A<int> var;
```

```
class _CTC20B70D45F2;

char c;
class _CTC20B70D45F2 {
    int a;
};
_CTC20B70D45F2 var;
```

Producing Comprehensible Diagrams

- Reverse engineered code can produce huge amount of extracted data
- Different filtering methods are available
 - Filtering by input source files
 - Filtering according to scopes
 - Filtering using class dependencies
 - Filtering “by hand”
 - Diagram Completing

Experiments

- Five publicly available C++ programs

Project	No. of files	Size (MB)	Extract. time	Mem. (MB)	No. of classes	No. of methods	No. of attributes
<i>jikes</i>	76	3.6	00:00:58	20	275	3.464	1.719
<i>boost</i>	281	2.4	00:04:46	27	1.712	5.187	1.197
<i>leda</i>	505	3.1	00:05:31	51	1.562	1.562	8.310
<i>StarCalc</i>	8.527	61.5	01:59:11	110	3.983	47.747	18.098
<i>StarWriter</i>	9.449	69.2	02:10:08	128	4.995	60.515	23.929

(Machine: PIII-800; 256MB RAM)

Conclusion

- We presented the functionalities of the Columbus toolset with respect to its reverse engineering capabilities
- Current version is able to analyze C/C++ projects (supporting Java is under development)
- Powerful C/C++ extraction
- Direct access to the extracted information (API)
- Extensibility
- Various output formats (Mermaid, TED, Rose, MS Jet, html, XML, ASCII)