



Columbus – Reverse Engineering Tool and Schema for C++

Rudolf Ferenc¹, Árpád Beszédes¹, Mikko Tarkiainen²
and Tibor Gyimóthy¹

¹University of Szeged, Hungary

²Nokia Research Center, Finland



Introduction

- Software systems are rapidly growing and are getting more and more complex
- Need to understand the source code of a large system
- Existing legacy code and high number of participants in code development
- 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.]



Introduction (cont.)

- Suitably assembled *set of tools* is required
- To achieve interoperability among these tools we need:
 - a *common schema*,
 - a *front end* that generates data according to the schema and
 - a *framework*
- We present requirements, which should be met when designing a schema, front end and framework



Requirements for a framework

- Project handling
- Extendibility
- Filtering
- Visualization



Requirements for a C++ front end

- Correctness
- Completeness
- Fault tolerance
- Parsing speed
- Preprocessing
- Dialects
- Command line operation



Requirements for a C++ schema

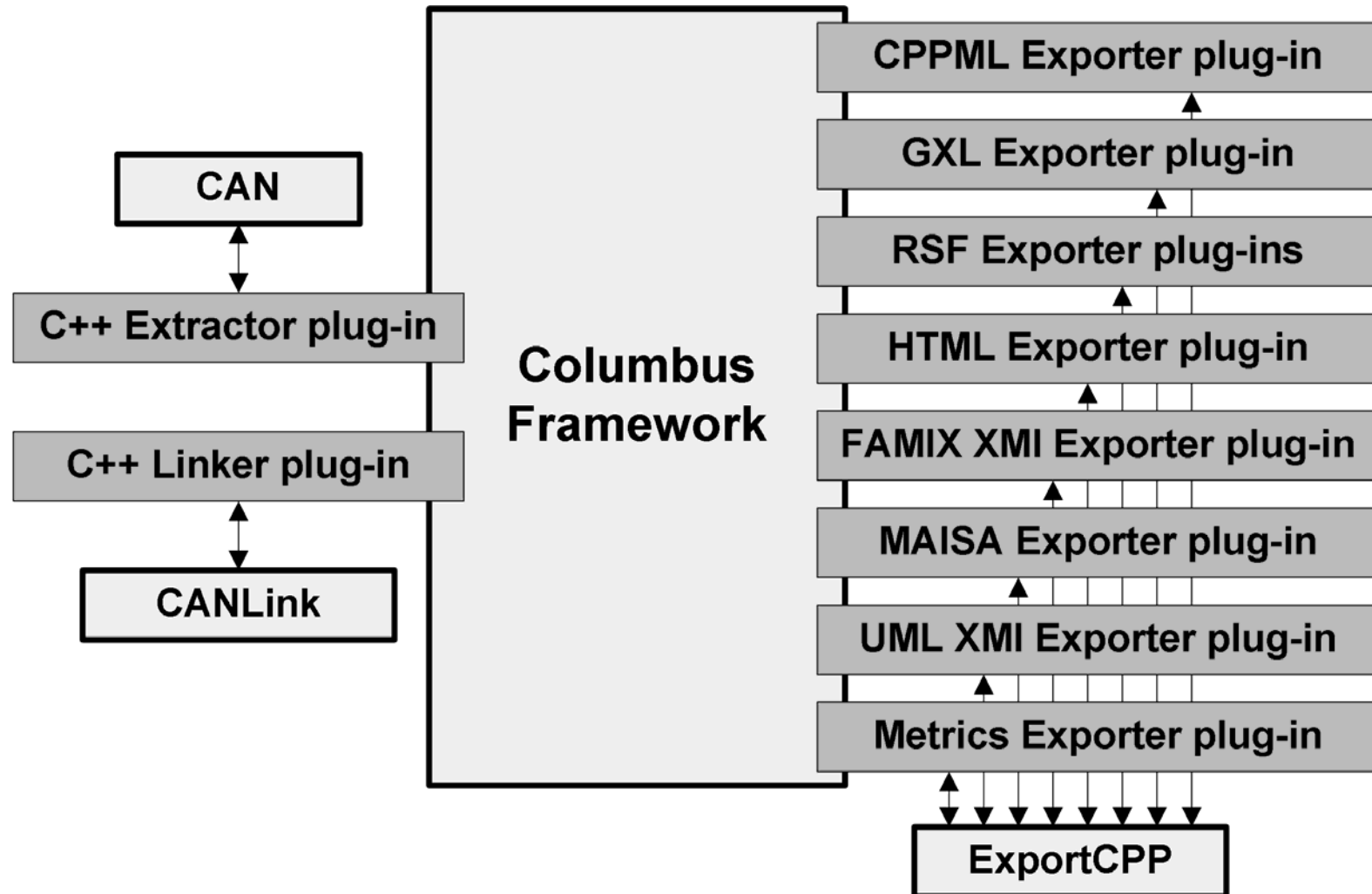
- Level of detail
- Modularity/extendibility
- Instance representation
- Dialects
- Description language



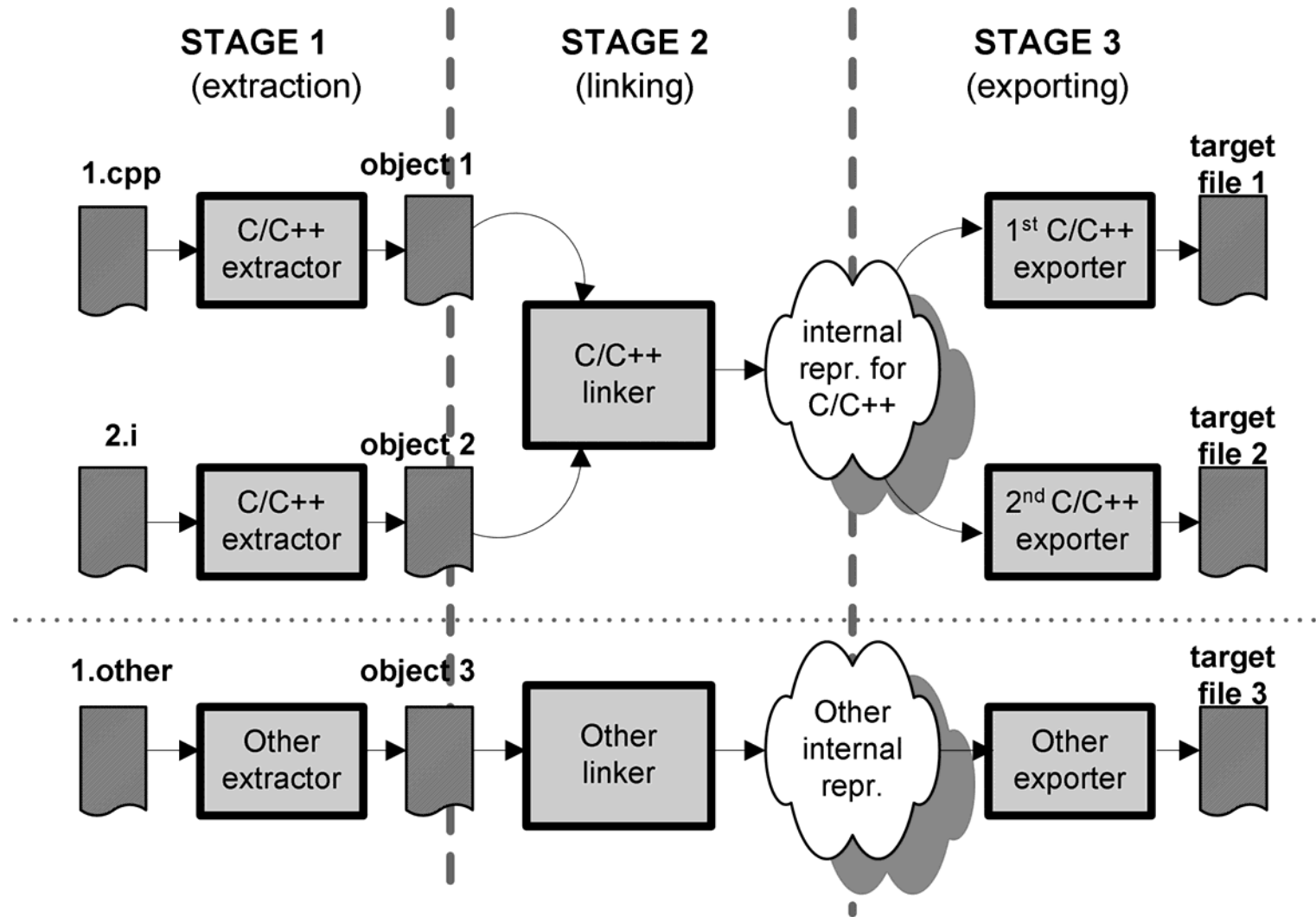
Columbus framework

- Framework for reverse engineering large object oriented systems
- Developed in cooperation between
 - University of Szeged
 - Nokia Research Center
 - FrontEndART Ltd.
- Supports project handling, data extraction, data representation, data storage and filtering
- High extendibility due to the plug-in architecture
- API for writing third party plug-ins
- Integrated with Microsoft Visual Studio 6.0

Overview of the Columbus system

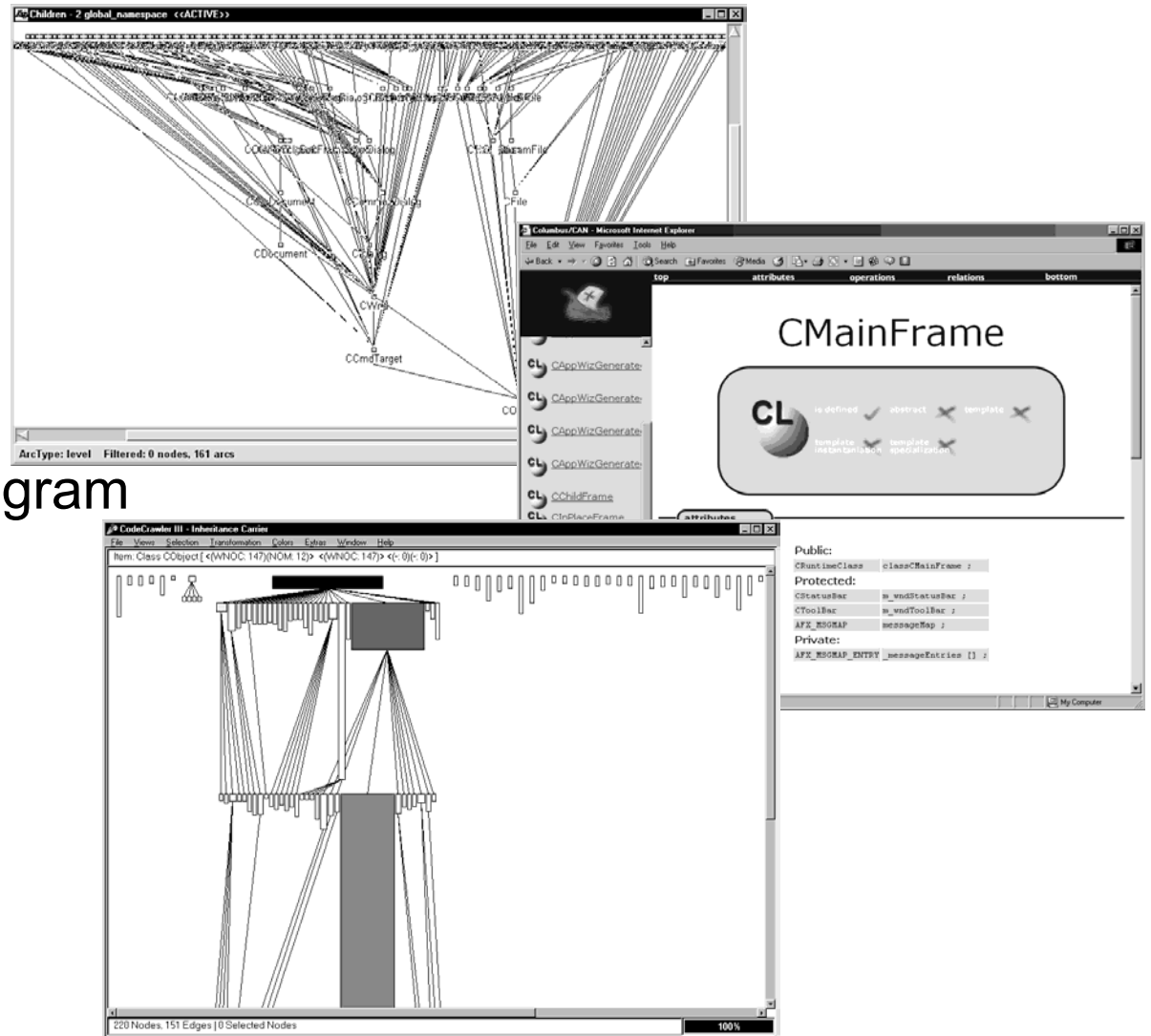


Extraction process

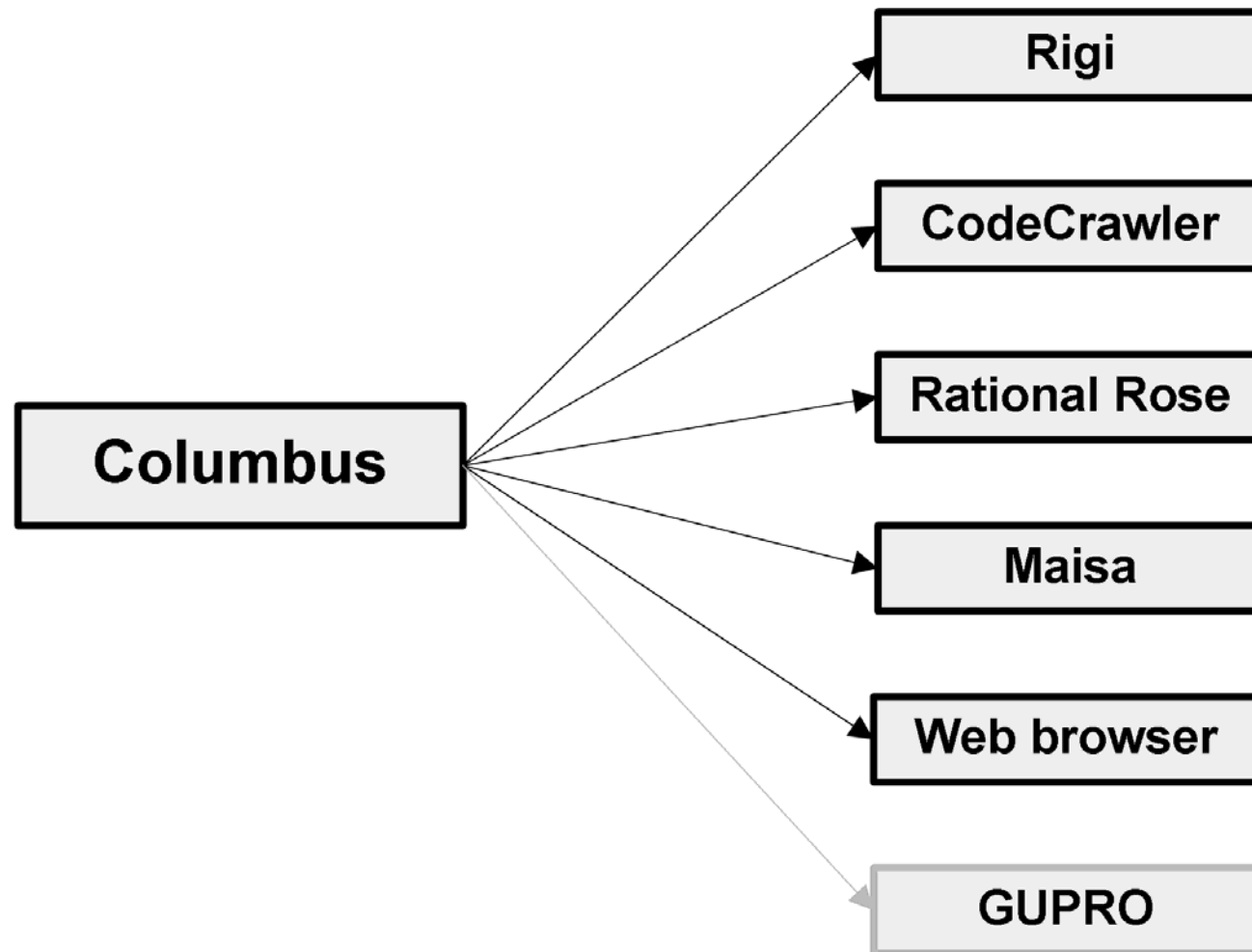


Export formats

- CPPML
- GXL
- RSF
 - Schema
 - Call graph
 - UML Class Diagram
- HTML
- Famix XMI
- Maisa
- UML XMI
- Metrics
 - More than 50



Integration with other tools





CAN – The C++ analyzer

- Front end used by Columbus
- Command-line tool (using in makefiles)
- Fault tolerant (error recovery)
- Use of precompiled headers
- Instantiation of templates at source level
- Supported dialects:
 - ANSI C++
 - Microsoft Visual C++
 - Borland C++ Builder



New features of CAN

- Supported platforms:
 - Windows (Visual C++)
 - Linux (GNU g++)
- Preprocessor
- Analyzes statements and expressions
- Builds the full ASG with name- and overload resolution
- CHA and RTA algorithms for a more precise call graph
- Special version for instrumenting source code (dynamic analysis)

Experiments

| Project infos | No. of files | Size | LOC |
|---------------|--------------|--------|-----------|
| Jikes | 77 | 3.5MB | 94 611 |
| Leda | 508 | 2.9MB | 116 752 |
| StarWriter | 9 449 | 66.5MB | 1 764 574 |

| Extraction | Time* | Memory |
|------------|----------|--------|
| Jikes | 00:01:02 | 19MB |
| Leda | 00:05:50 | 49MB |
| StarWriter | 01:55:09 | 139MB |

- Machine:
PIII- 800 – 256MB RAM
- * Without
statements & expressions

| Statistics | Classes | Namespaces | Functions | Attributes |
|------------|---------|------------|-----------|------------|
| Jikes | 275 | 1 | 3 471 | 1 643 |
| Leda | 1 563 | 1 | 10 802 | 8 287 |
| StarWriter | 4 988 | 99 | 61 553 | 23 862 |



Columbus Schema for C++

- To accomplish data interchange, a common format – *schema* is needed
- No standard schema for C++ exists
- Active research topic
 - e.g. GXL, DATRIX, GUPRO, Bauhaus
- We propose a new C++ schema as a candidate for data exchange



Columbus Schema for C++ (cont.)

- Satisfies some important requirements of an exchange format. It reflects the
 - low-level (AST) structure of the code, as well as
 - higher level semantic information (e.g. semantics of types)
- Models the “clean” C++ language syntax (preprocessed source code)
- Described with standard UML Class Diagrams



Our mission

- Not just a front end but a useful framework
 - Integration with other useful tools
 - Calculating additional information from the schema for the RE community like
 - Metrics
 - UML Class Diagrams
 - Design Patterns
 - Call graph
 - ...
- Free for scientific and educational purposes



Future

- New version 3.5 at end of October
 - Statements/expressions
 - GNU dialect
 - Metrics, UML XMI, Famix XMI, more precise call graph
- Future plans (version 4.0)
 - More general platform independent framework
 - Integration with even more tools
 - Data flow graph, control flow graph, ...
 - Java front end and schema
 - Integration with Microsoft Visual Studio.NET
- Download: www.frontendart.com