UNIVERSITY OF WATERLOO

Software Architecture Series

Architecture Extraction Tools

Matt Armstrong & Christopher Trudeau
{mnarmstr,ctrudeau}@se.math.uwaterloo.ca

CITO'98

slides by *SlickSlides*

Outline

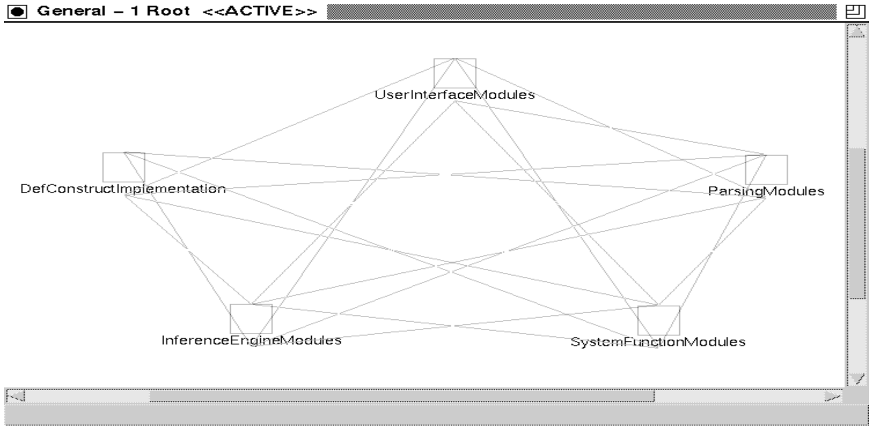
- introduction
- tool overview
- capabilities of tools
- compare and contrast
- conclusions

Introduction

- need to produce architectural designs
- many architecture extraction tools but little comparison work done
- examining five tools:
  1. **Rigi** – University of Victoria
  2. **SNiFF+** – TakeFive Software
  3. **PBS** – University of Toronto
  4. **CIA** – AT&T Research
  5. **Dali** – SEI (CMU)
- comparing tools in three areas:
  1. extraction – extract details into relations
  2. classification – combine relations into more abstract components
  3. visualization – observe conceptual interpretation of source

Rigi

- "robust" C language extractor `rigiparse` (also has C++ & COBOL)
- `rigiedit` graph program
- "collapse" function allows grouping of nodes into hierarchies
- provides "spring" & "Sugiyama" layout algorithms
- filtering based on information in nodes helps classification

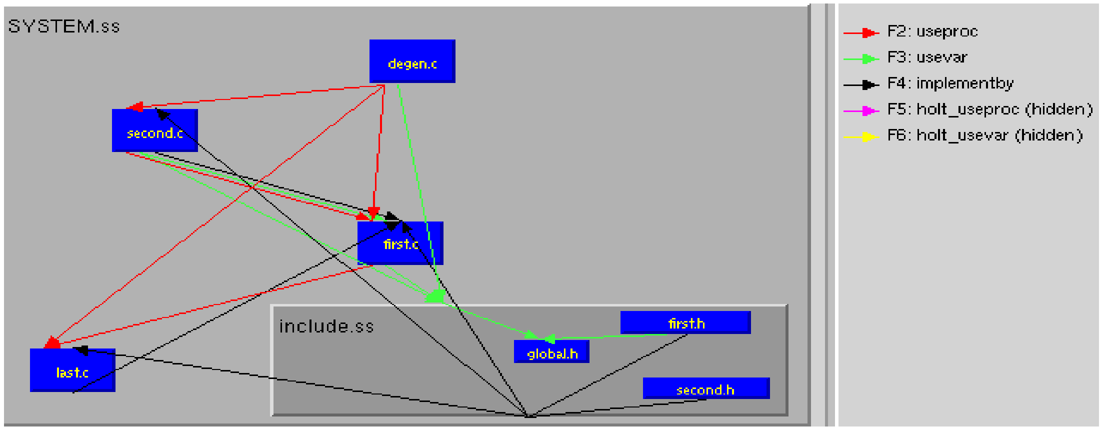


SNiFF+

- commercial product
- superb extractor for C, C++, Java, IDL, Embedded SQL, and Fortran (77 & 90)
- excellent as a source browser
- supports revision control systems, external documentation sources, project configuration & management, and a difference engine
- provides API so custom software can issue queries

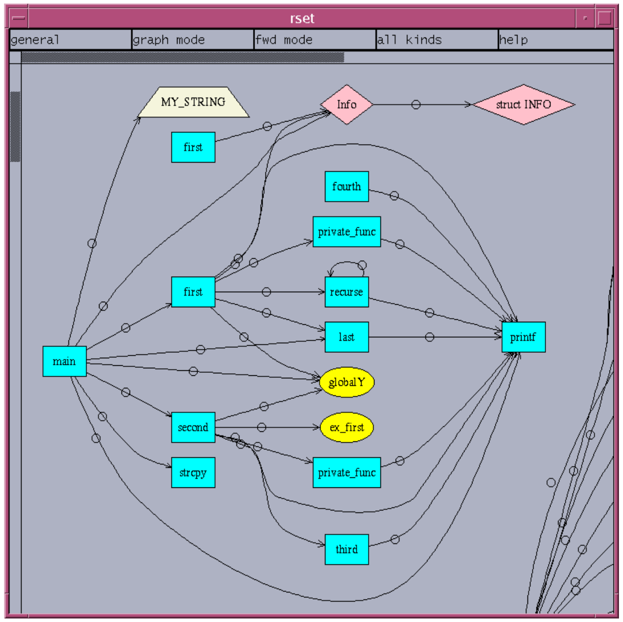
PBS

- CFX (C Fact Extractor) for parsing – requires compilable code
- tuple manipulation tool, Grok, excellent for classification assistance
- Java-based lsedit allows visualization of systems
- can link external documents and hyper-link nodes to other software – typically calls sub-system views



CIA

- `cia` for parsing – requires compilable code
- entity, relationship, and directory based
- graphical tool `dotty`
- `ciao` – query-by-example interface for finding information in code



Dali

- an open prototype workbench comprising external extractors, POSTGRES data repository, and the Rigi user interface
- SQL query interface for classifying and manipulating information
- provides a rich set of Perl scripts to aid in extraction combination and transformation into the POSTGRES database

Extraction

	SNiFF+	Rigi	PBS	CIA	Dali
Hides system library calls	√		√		√
Extern globals	√	√	√		√
Non-extern globals	√	√	√	√	√
Contents of structures	√				
Handles address of fn.	√			√	
Fn. ptr. deref'd as a call					
Conditional compilation		√	√	√	√
Macros	√		√		
Recursion	√	√	√	√	√
Multiple static functions	√		√	√	
Local variables	√				
Multiple prototypes	√	√		√	√

Classification

- *CIA* and *SNiFF+* support information queries only
- *Rigi*:
  - pattern matching utilities
  - manual groupings through the "collapse" function
  - *Tcl/Tk* interface – custom manipulation of Rigi
- *PBS*:
  - `Grok` scripts provided for file and subsystem level extraction
  - subsystem classification must be done before hand in data files – not in GUI
  - no integration of classification and visualization
- *Dali*:
  - augments *Rigi*'s capabilities
  - provides SQL interface to query information from the POSTGRES database
  - several SQL scripts provided for classification up to file level, but none for subsystem

Visualization

	SNiFF+	Rigi	PBS	CIA
Nodes type	t	c	c	cs
Move nodes		√	√	
Node hierarchies		√	√	
Bidirectional edges			√	
Edge type		c	c	
Edges use inflection pts				√
Avoids cross over	√			√
Scroll	√	√	√	√
Zoom	√	√	√	
Annotation		√	√	
Multiple views		√	√	
Saving	√	√	√	

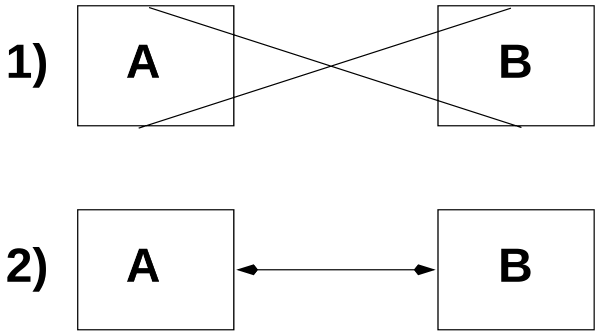
Extraction

- *SNiFF+* and *Rigi* provide the most robust parsing capabilities
- only *SNiFF+* and *Rigi* support multiple languages
  - *CIA* has a C++ version coming
  - lack of support for typical legacy languages
- information overload in *PBS* and *CIA*
- compilation requirement in *CIA* and *PBS*
- false positives in *Rigi*, *PBS*, and *Dali*'s use of *SNiFF+*
- all relatively easy to use; *SNiFF+* is the only one that uses a GUI

Classification

- problem of classification requires both domain knowledge and specific expertise with the software in question – very difficult to automate
- *CIA* and *SNiFF+* provide query info only – mostly useless for real systems
- file level abstraction tools provided by *PBS* and *Dali*
- *Rigi* provides pattern matching which is only useful in a limited set of cases
- *PBS*'s classification is done at the data file level – can not manipulate subsystems in the GUI
- *Rigi* provides an interface for using *Tcl/Tk*
- *Dali* requires knowledge of SQL
- both *PBS* and *Dali* provide some automated scripts
- *Dali* does not currently provide subsystem decomposition scripts

Visualization



- all can scroll, *SNiFF+* can't zoom
- none take advantage of bidirectional relationships
- all have static anchor points
- lines can cross through nodes in everything but *SNiFF+*
- *CIA* uses curved lines with 2 inflection points – nice diagrams, but this may be one of the reasons it is so slow
- static layout in *CIA* and *SNiFF+*
- document linking in *SNiFF+*, *Rigi* and *PBS*
- *Rigi* provides layout algorithms
- both *Rigi* and *PBS* allow hierarchical diagrams
  - *Rigi* only has one type of "collapsed" node and arc
  - *PBS* can not manipulate hierarchy in the GUI

Conclusions

- *PBS* and *Dali* provide the most features for architecture abstraction
- almost all of the visualization systems need work – an amalgamation of good features from all of the tool sets would be beneficial
- *SNiFF+* and *CIA* are poor tools for architecture extraction, but *SNiFF+* is very good at what it was intended
- *Rigi* provides a rich graphical tool set but is lacking when it comes to classification
- *PBS* and *Dali* have equivalent classification capabilities