

Welcome to SENG 371

Software Evolution Spring 2013

A Core Course of the BEng Program

Hausi A. Müller, PhD PEng
Professor, Department of Computer Science
Associate Dean Research, Faculty of Engineering
University of Victoria

Announcements

- Final exam
 - Sat, April 13 — 7:00 -10:00 pm
- Marking
 - Midterm and A1 graded
 - Marks posted
- Course website
 - <http://www.engr.uvic.ca/~seng371>
 - Lecture notes posted
 - Lab slides and activities are posted
- Assignment 2
 - Due March 11 — revised
 - Reverse engineering and program understanding
 - Part I—Summarize three papers
 - Part II—Define terms
 - Part III—Reverse engineer a C program (gawk)
 - Rigi demo
 - Cite your sources
 - Submit by e-mail to seng371@uvic.ca

Reading assignments

- Chikofsky, Cross: Reverse Engineering and Design Recovery: A Taxonomy, IEEE Software 7(1):13-17 (1990)
http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=43044
- Kienle, Müller: Rigi—An Environment for Software Reverse Engineering, Exploration, Visualization, and Redocumentation, Science of Computer Programming 75(4):247-263, Elsevier, Apr. 2010.
<http://www.sciencedirect.com/science/article/pii/S016764230900149X>
- Müller, Jahnke, Smith, Storey, Tilley, Wong, Reverse Engineering: A Roadmap, in The Future of Software Engineering, ICSE 2000 Millennium Celebration, 2000.
<http://dl.acm.org/citation.cfm?id=336526>


Rigi—Mac using Windows 7 VM & under Windows 7 by Mackenzie Marshall

- Create Windows 7 VM
- Run a Windows 7 virtual machine
 - <http://www.parallels.com/ca/>
 - Any Windows 7 VMs should work
- Download Windows XP/2000 version of Rigi
 - <http://www.rigi.cs.uvic.ca/Pages/download.html>
 - Use Rigi-12-Jan-2003.zip version
 - Extract zip file to a folder
- Add system variables to the windows VM
 - Go to Computer → System Properties → Advanced System Settings → Environment Variables
 - Click "New" button under the "System variables" section
 - Add the following system variables:
 - RIGI "the path to your extracted rigi folder"
 - TCL_LIBRARY %RIGI%\lib\tcl8.4
 - TK_LIBRARY %RIGI%\lib\tk8.4
 - path %path%;%RIGI%\bin
- Installation instructions
 - readme.txt of the zip file or in <http://www.program-transformation.org/Transform/Rigi/GettingStarted>
 - Add .rsf files
 - RSF files you wish to parse add under "your rigi folder" \Rigi\rb\gawk
 - Run Rigi
 - Go to "your rigi folder" \bin
 - Double click "rignedit.exe"
 - Use "Refresh" from "Window" menu to redraw the windows.
 - Explore RCL commands to automate tasks.
- The same steps work for native Windows 7**

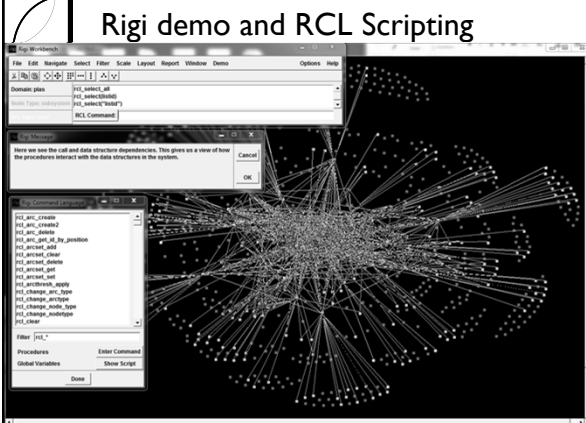
Rigi Manual

- Link to Rigi manual
 - <http://www.rigi.cs.uvic.ca/downloads/rigi/doc/rigi-5.4.4-manual.pdf>
- Explore RCL commands
 - Click the RCL Command: button in the Workbench window

Rigi User's Manual
Version 5.4.4



Rigi demo and RCL Scripting



The screenshot displays the Rigi Workbench interface. On the left, there are several panels: 'Domain plan' showing a list of RCL commands like 'rcl_add', 'rcl_remove', 'rcl_change', etc.; 'Procedures' showing a list of RCL commands; and 'Global Variables'. The main area shows a complex dependency graph with numerous nodes and connecting lines. A dialog box is open in the foreground, displaying a list of RCL commands and their dependencies.

Assignment 2 Part III

How to get started?

- What are the major static components and relationships?
 - Graph model
- Nodes and arcs
 - What are the entities?
 - What are the relationships?
- Graphs
 - Call graph (functions and function calls)
 - Module graph (files and file dependencies—calls, uses)
 - Abstract data types (data types and access functions)
- The next few slides provide ideas for identifying entities and relationships or nodes and arcs in the graph model

7

Ideas

- Apply everything you learned about software structure over the past four years
- Don't settle on the first idea you come up with
- Use the diagrams suggested in the resources, but you need legends and explanations (prose) to go with the diagrams
 - SEI views
 - Siemens views
 - Rational views
- Exploit file and directory structure (i.e., build subsystems)
- Form abstract data types (i.e., build classes)
- Call graph (i.e., function-function relationships)
- Entity relationship diagrams (ER)
 - Identify node categories (entities)
 - Identify arc categories (relationships)

8


More ideas

- Form abstraction hierarchies
- Encapsulate control, data, control & data (objects)
- Summarize graphs to build hierarchies
- Compose views
 - emphasize important aspects
 - de-emphasize immaterial components
- Recognize and match design patterns
- Separate concerns
- Extract UML (class) diagrams
- Recognize and identify APIs and interfaces
- Mine configuration management system for
 - product lines
 - versions, releases
- Use visualization techniques
 - SHriMP
 - Rigi

9

How do you document software architecture?

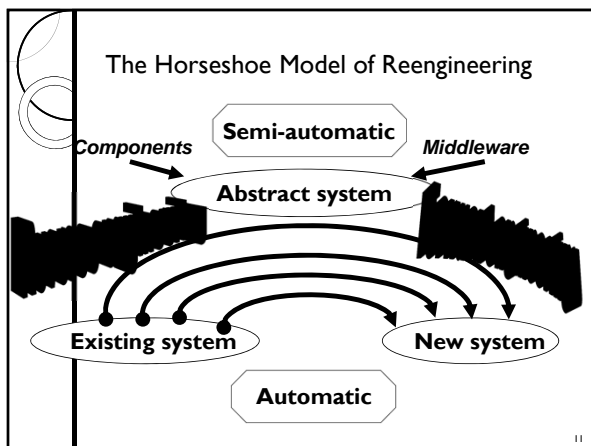
- Box and arrow diagrams
- UML diagrams
- Class diagrams in Rational Rose



Multiple views

- Software architectures are complicated—typically too complicated to view all at once

10



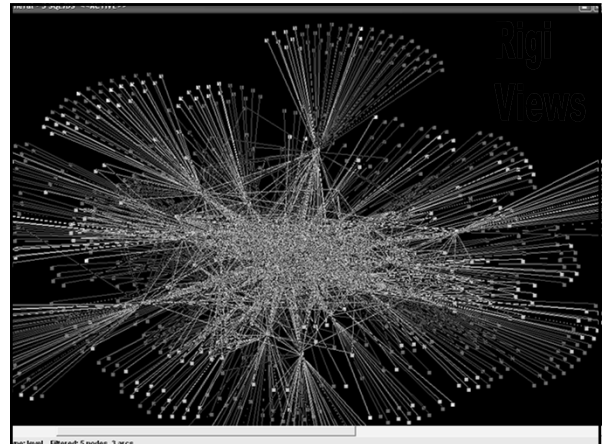
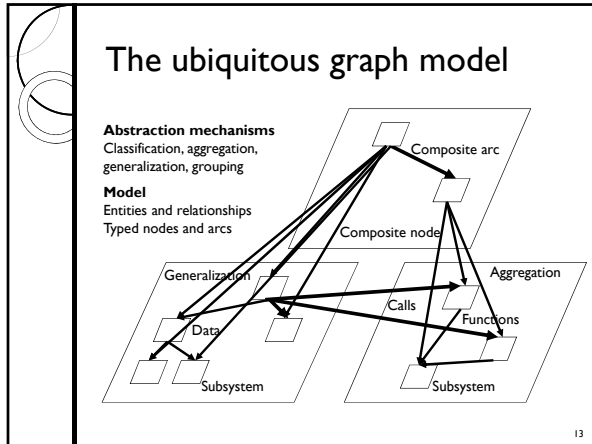
Organizational axes

Abstraction hierarchies

- Aggregation hierarchies
 - part-of relationships
- Generalization / specialization hierarchies
 - is-a relationships
 - inheritance
- Grouping
 - arbitrary
- Classification
 - category, instances
 - type, variables
 - class, objects



12



- ### Rigi System
- Website
 - <http://www.program-transformation.org/Transform/RigiSystem>
 - Installation
 - <http://www.program-transformation.org/Transform/RigiInstall>
 - Publications
 - <http://www.program-transformation.org/Transform/RigiPublications>
- 15

