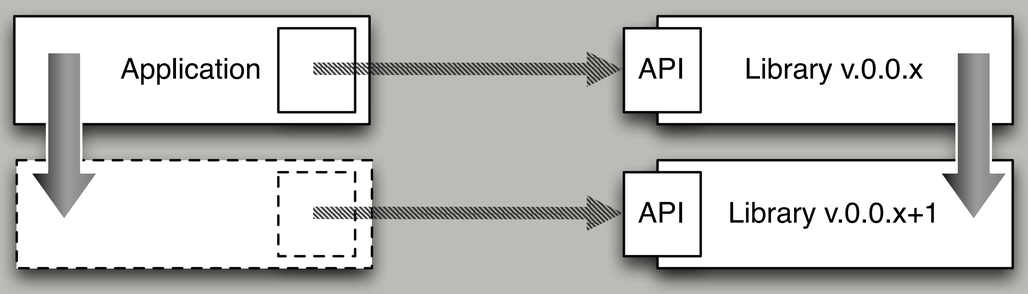
Know Before You Port: The Role of Program Comprehension in Porting Scientific Software

Kostadin Damevski[1], David Shepherd[2], Nicholas Kraft[2], Lori Pollock[3]

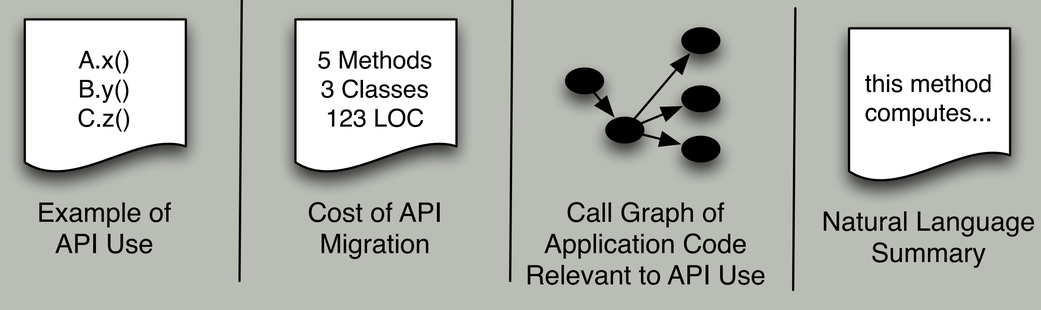
[1]Virginia Commonwealth University
[2]ABB Corporate Research
[3]University of Delaware

Challenges in Porting Scientific Software



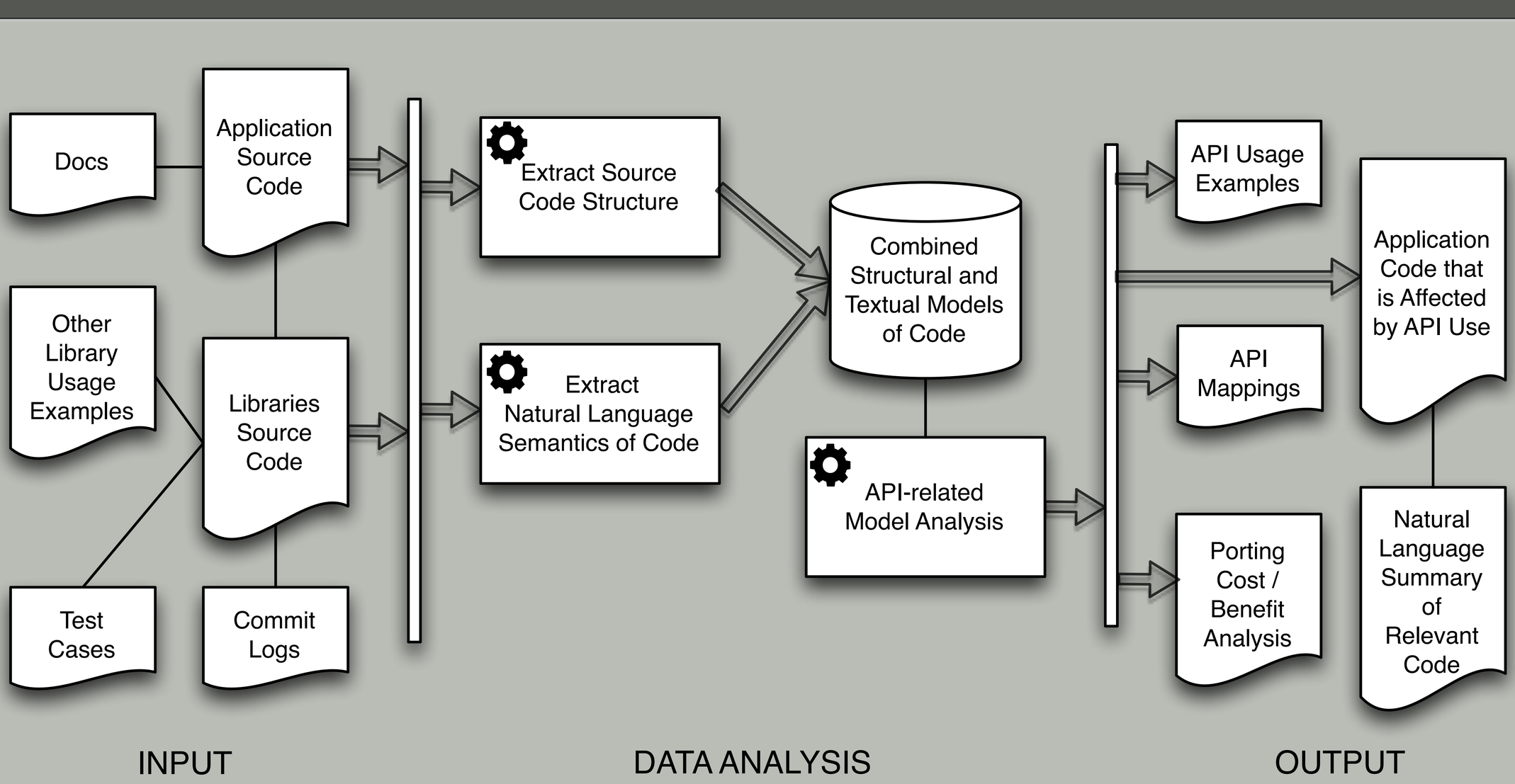
- Scientific and engineering software often outlives the current generation of hardware, and in turn needs to be modified to execute on newer classes of hardware architectures. [Carver et al.]
- Program comprehension is commonly the most time consuming task during software maintenance. [Ko et al.]
- During a software maintenance task, the quality of program comprehension impacts the quality of the result program change. [Robillard et al.]
- *Scientific software is often conservative in the use of external libraries, limiting the impact of software engineering tools that require long-term "buy in".*
- *Few software engineering tools are capable of executing on platforms and for languages relevant to scientific software.*

Key Research Questions



- How can examples of API use, for the purposes of comprehension, be extracted from libraries and other relevant code?
- How can the cost and benefit of API migration be estimated automatically?
- What is the code relevant for comprehension and modification in a particular porting scenario?
- What are relevant natural languages summaries of APIs and their use that can aid rapid comprehension?
- *Are existing natural language dictionaries appropriate for scientific and engineering codes?*
- *Does code with very poor natural language content (in identifiers or comments) lend itself to statistical natural language models?*

Combined Textual and Structural Analysis of Software Artifacts for Program Comprehension in Porting



Relevant Prior Research and Engineering Work

- Source Code Structural Analysis (for C, C++, Java, C#)
  - SrcML – `http://www.srcml.org` – [U. of Akron, Kent State U.]
  - SrcML.NET – `https://github.com/abb-iss/SrcML.NET`
    - Constructs call graphs, def-use chains, and allows for easy querying of AST
- Natural Language Analysis of Source Code
  - Software Word Usage Model (SWUM) – `https://github.com/abb-iss/Swum.NET`
    - Generates natural language semantic information from code
- Existing Comprehension Tools
  - Sando – Feature Location (Code Search) – `https://sando.codeplex.com`
    - Visual Studio extension with over 15,000 downloads
  - Prodet – Call Graph Navigation and Exploration Tool

References

- [Carver et al.] Jeffrey C. Carver, Richard P. Kendall, Susan E. Squires, and Douglass E. Post. *"Software development environments for scientific and engineering software: A series of case studies"*. 29th International Conference on Software Engineering (ICSE'07), pages 550-559. IEEE, 2007.
- [Ko et al.] Andrew J. Ko, Brad A. Myers, Michael J. Coblenz, Htet Htet Aung. *"An exploratory study of how developers seek, relate, and collect relevant information during software maintenance tasks"*. IEEE Transactions on Software Engineering 32(12):971-987 (2006).
- [Robillard et al.] Martin P Robillard, Wesley Coelho, Gail C Murphy. *"How effective developers investigate source code: An exploratory study"*. IEEE Transactions on Software Engineering 30(12):889-903 (2004).

damevski@acm.org