IRiSS – A Source Code Exploration Tool

Denys Poshyvanyk
Andrian Marcus
Yubo Dong
Andrey Sergeyev

Computer Science Department
Wayne State University
Detroit, MI, USA

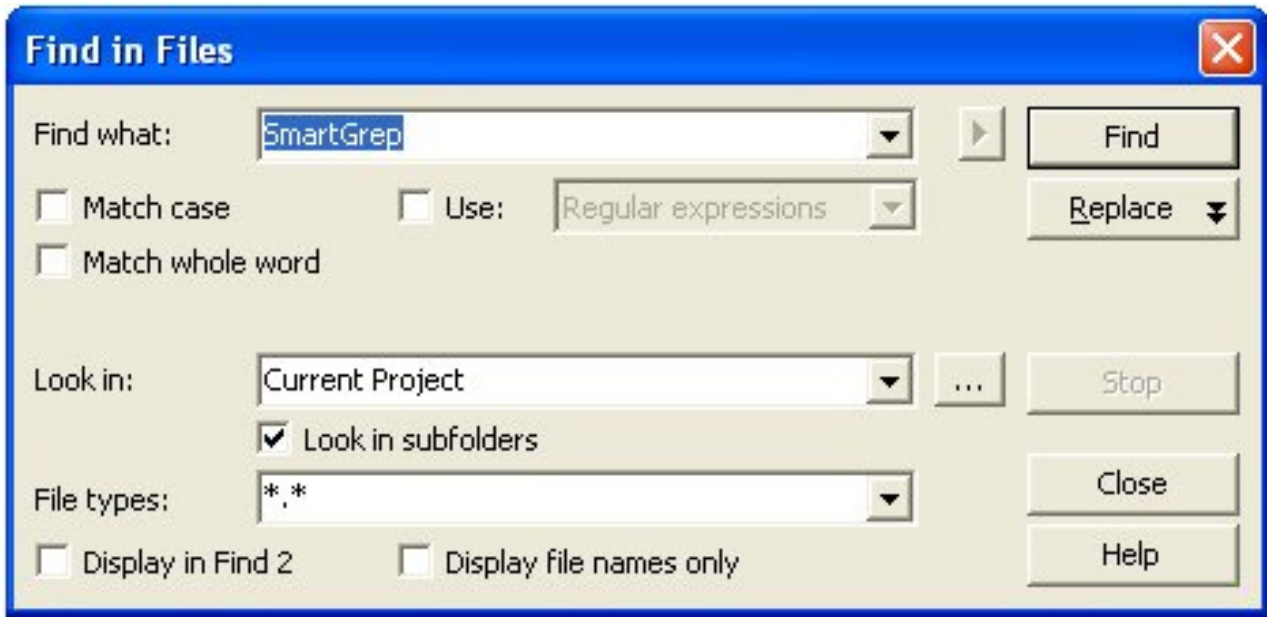
Concept Location

- Locating the implementation of a concept or feature in the source code
- Static
  - Dependency based search [Chen'00]
  - String based search (i.e., grep)
  - IR methods [Marcus'04]
- Dynamic
  - Execution traces - Reconnaissance [Wilde'92]
- Combined (using concept analysis) [Eisenbarth'03]
- Used in incremental change, comprehension, debugging, etc.

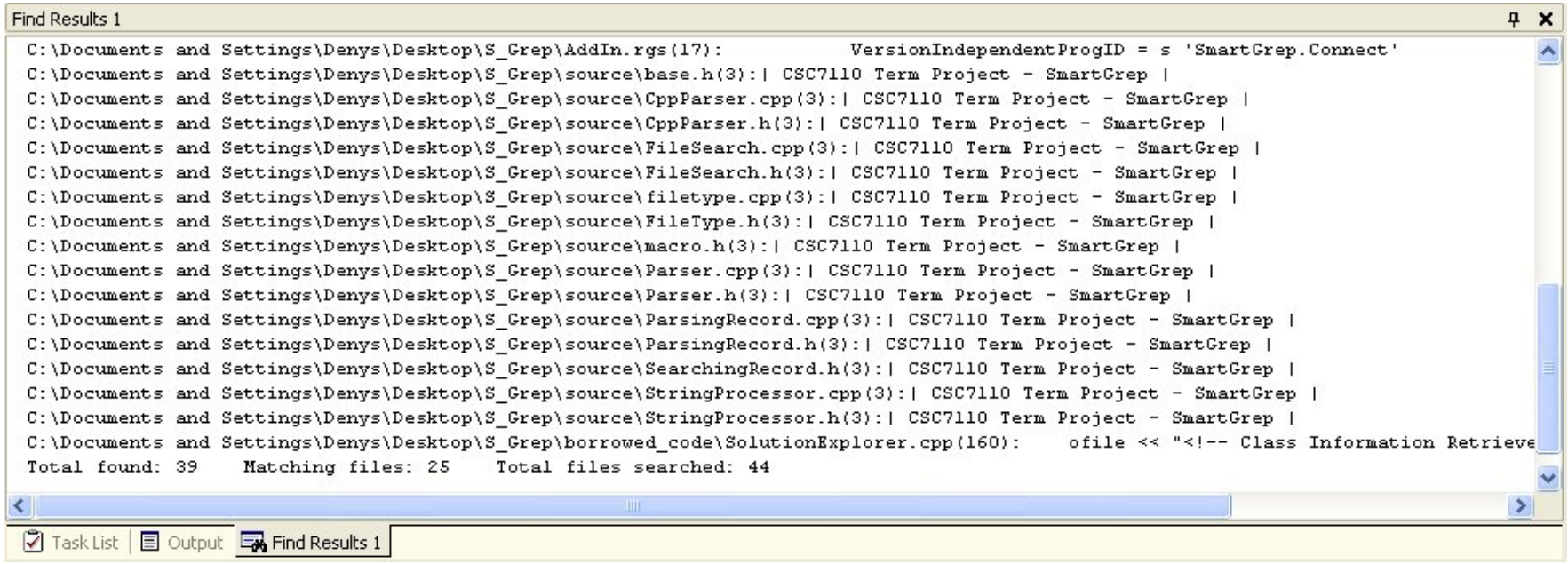
Concept Location

- Locating the implementation of a concept or feature in the source code
- Static
  - Dependency based search [Chen'00]
  - String based search (i.e., grep)
  - IR methods [Marcus'04]
- Dynamic
  - Execution traces - Reconnaissance [Wilde'92]
- Combined (using concept analysis) [Eisenbarth'03]
- Used in incremental change, comprehension, debugging, etc.

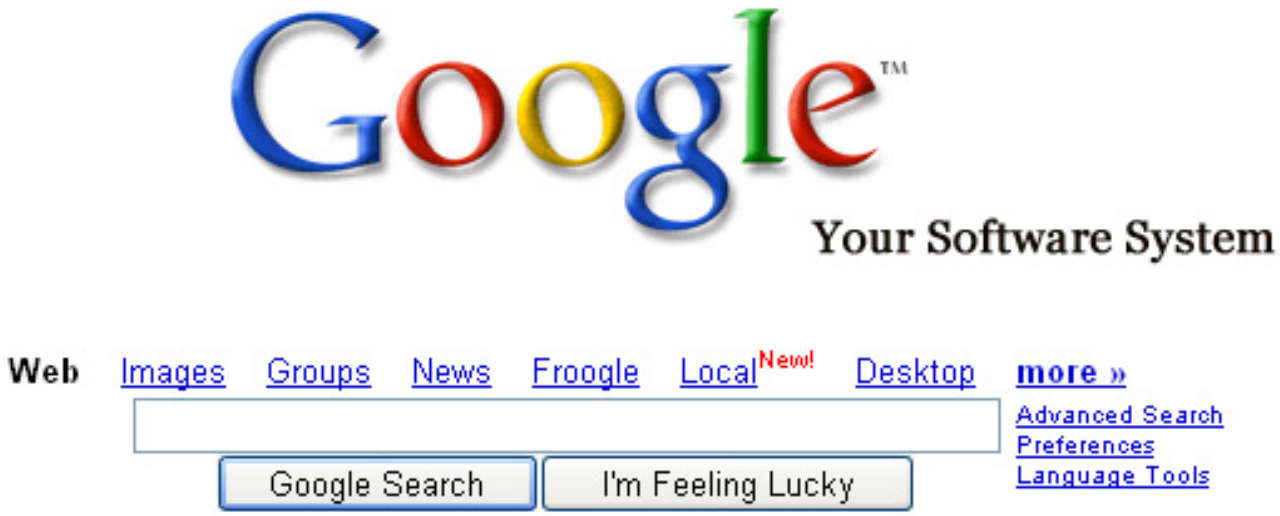
“Find in Files” in .NET?



Results for “Find in Files”

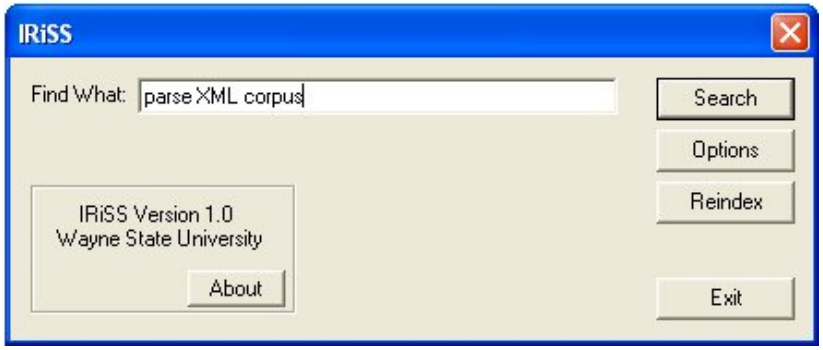


Our Inspiration



How do We Want to Search?

- “Google-like” queries



- Ranked results
- Various granularity of results (i.e., classes, methods, etc.)

IRiSS

Information Retrieval based

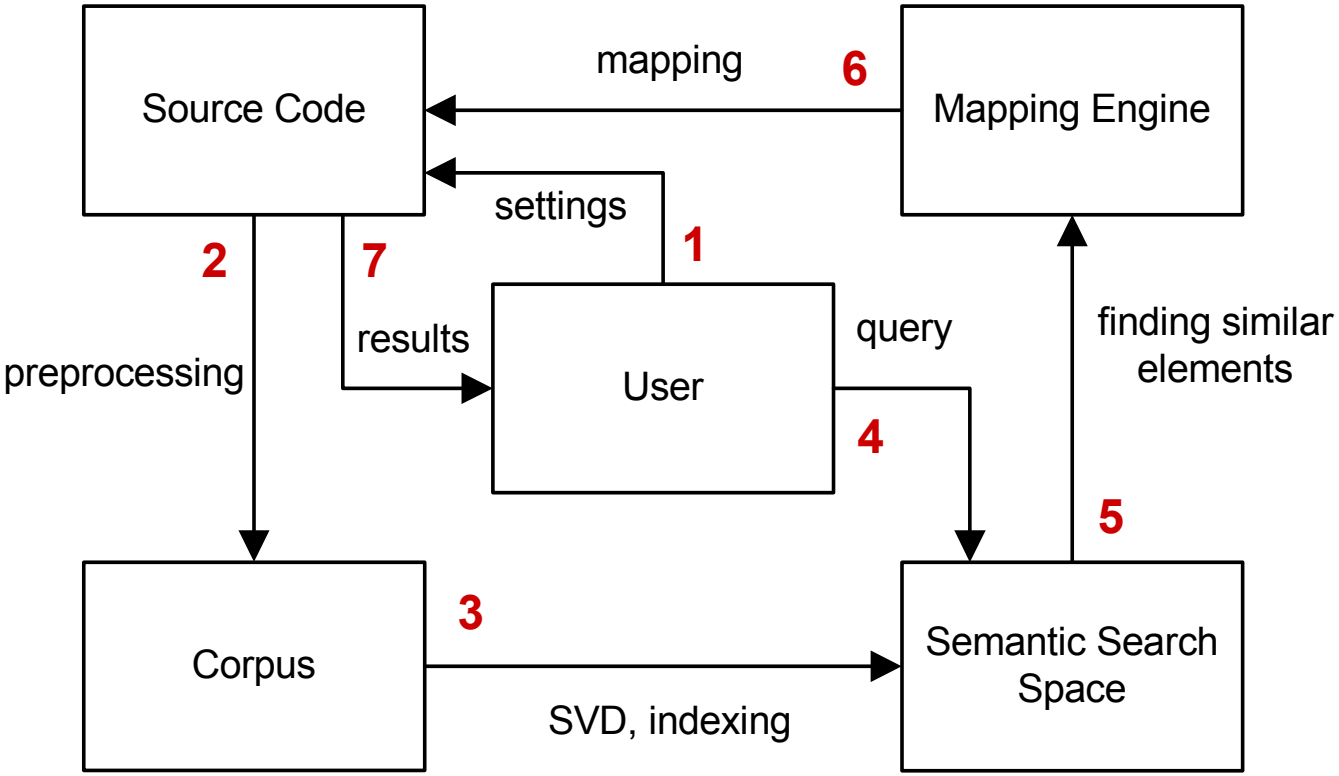
Software Searching

Information Retrieval

- An Information Retrieval System is capable of storage, retrieval, and maintenance of information (e.g., text, images, audio, video, and other multi-media objects) [Kowalski'97]

- IR methods (e.g., signature files, inversion, clustering, probabilistic classifiers, vector space models, etc.) are used often for text retrieval (e.g., libraries, search engine, etc.)

Searching with IRiSS



Corpus Generation

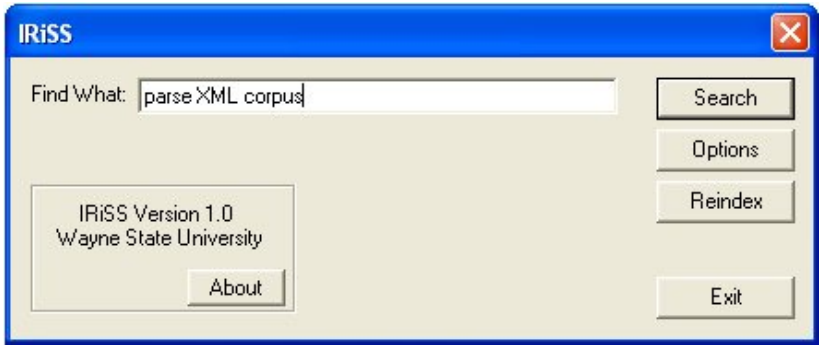
- Parsing to extract semantic information (i.e., comments and identifiers)
- split_identifiers & SplitIdentifiers
- Define source code documents with user-defined granularity (e.g., class, methods, functions, declarations, interfaces, etc.)
- Works on C/C++
- It is easy to extend to other languages

Building the Semantic Search Space

- We use Latent Semantic Indexing (LSI)
- Each source code element is transformed into a vector, based on the words it contains
- A similarity measure between two documents is defined as the cosine between their corresponding vectors

Query Formulation

- User defined queries
  - Most common, based on user experience and domain knowledge, little known about querying patterns



- Only query terms presented in the corpus are considered

Conclusions

- Pros
  - Simple and flexible to use
  - Returns ranked results – advantage over grep
- Cons
  - Misses some data elements (granularity)
  - Depends on the quality of comments and identifiers (grep has the same problem)

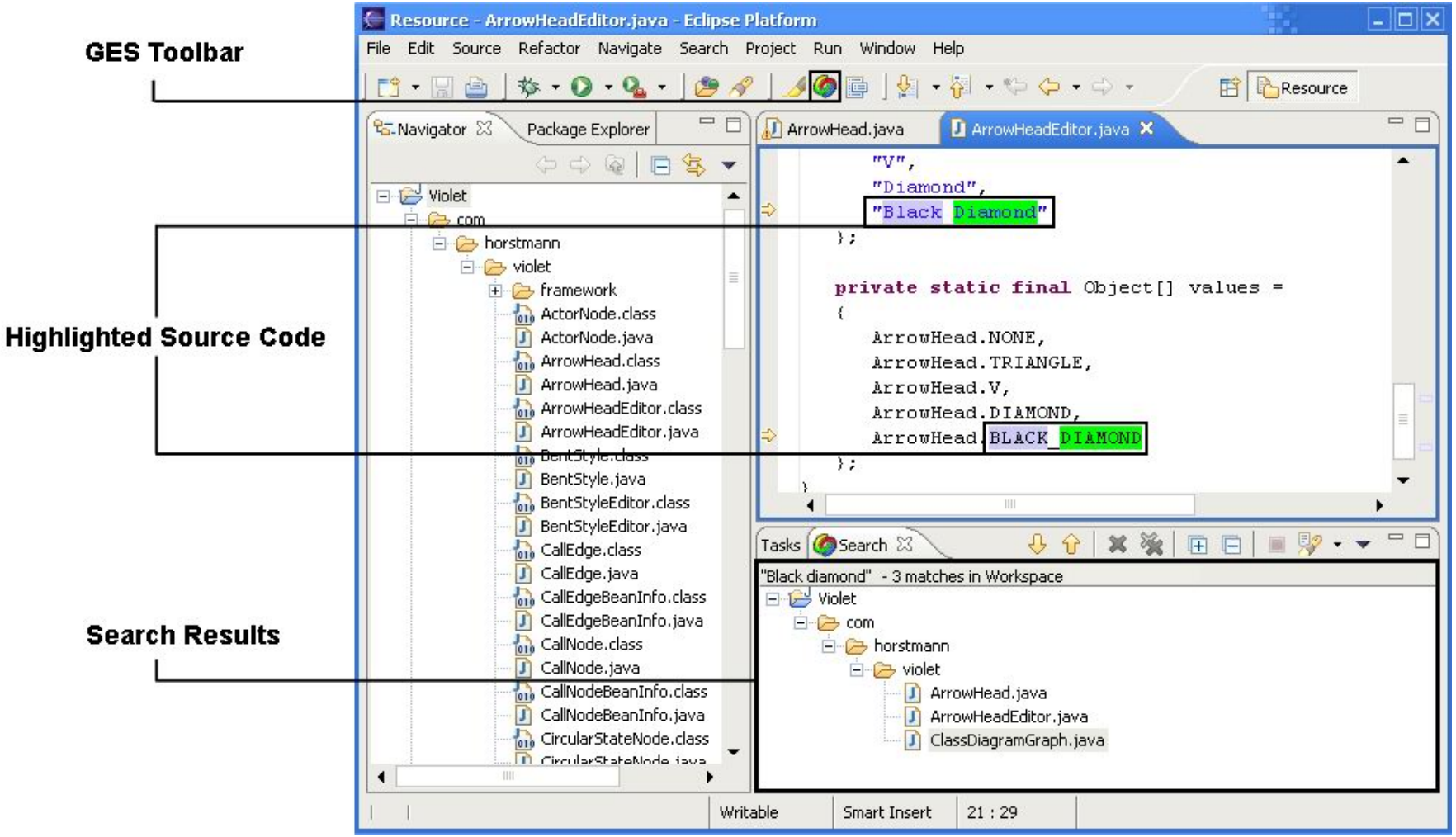
- Using IR methods for concept location and source code browsing is promising

Current Work

- Address the query refinement issue
  - Identify misspelled words
  - Suggest additional words/elements based on first set of results – combine them with `AND`, `OR`, `XOR`, `NOT`
  - Define the semantic signature of a source code element
  - Deal with scalability issues
- Combine LSI with dependency graph based search, dynamic methods, concept analysis
- Implement the tool as plug-in to Eclipse as well

Google Eclipse Search

Google Desktop Search + Eclipse



Future Work – Result Clustering

DEMONSTRATION