Overview of Component Search System SPARS-J

Tetsuo Yamamoto*, Makoto Matsushita**, Katsuro Inoue**

*Japan Science and Technology Agency
**Osaka University

Motivation

Reuse of Software Components
- A technique of developing new software components by using the components developed in the past.
- Example of reusable components: source code, document ...

However, reuse of components is not utilized effectively.
- A developer doesn’t know existence of desirable components.
- Although there are a lot of components, these components are not organized.

In order to take advantage of reuse, it is required to manage components and search suitable component easily.

Research aim

We have built the system which have functions as follows
- Collects software components eagerly without preserving their inherent structures
- Manages the component information automatically
- Provides component be suitable for User’s request

Targets
- Intranet
- Closed software development inside a company
- Internet
- Large open source software development web site
  - SourceForge, Jakarta Project, etc.

SPARS-J

Software Product Archive, Analysis and Retrieval System for Java

Java Software Product Archiving, analyzing and Retrieving System
- Many components are analyzed automatically.
- A search engine is built based on the analysis information.
- Component: a source code of class or interface

Features
- Keyword search
- Two ranking methods
- Frequency in use of a word
- Use relation
- Analyzed information
  - Components using/used by a component
  - Package hierarchy
Structure of SPARS-J

Component analysis part
- extract components from a file
- store analyzed information to DB
- clustering and rank components using DB

Component retrieval part
- search components in correspondence with query from DB
- rank components based on frequency in use of a keyword
- aggregate two rankings

User interface part
- deliver query to component retrieval part
- show search results

Database
- store analyzed information and component

Ranking search results
- Ranking method
  1. Component suited to a user request
  - Ranking based on frequency in use of a word
    - **Keyword Rank (KR)**
  2. Component used mostly
  - Ranking based on component use relation
    - **Component Rank (CR)**
- We make it high ranking that the component both 1 and 2 are high
- Search results are shown to aggregate two rankings

Outline
- Motivation and research aim
- SPARS-J
  - Outline
  - System architecture
  - Ranking method
  - Each part
    - Analysis part
    - Retrieval part
    - User Interface
  - Experiment
- Conclusion and Future work

Component analysis part
- Extract component and its information from a Java source file
  - The process
    - Extract a component
    - Index the component
    - Extract use relations
    - Clustering similar components
    - Rank components based on use relations (CR method)

Component retrieval part
- search components in correspondence with query from DB
- rank components based on frequency in use of a keyword
- aggregate two rankings

User interface part
- deliver query to component retrieval part
- show search results

Extract and index a component
- Extracting component
  - Find class or interface block in a java source file
  - Location information in the file (start line number, end line number)
- Indexing
  - Extract index key from the component
    - Index key: a word and the kind of it
    - No reserved words are extracted
    - Count frequency in use of the word

Extract use relations
- Extract use relations among components using semantic analysis
- Make component graph from use relations
  - Node: component
  - Edge: use relation

Component graph
- Inheritance
  - Field access
- Method call
  - The kind of use relation

Extract use relations
- Extract use relations among components using semantic analysis
- Make component graph from use relations
  - Node: component
  - Edge: use relation

Component graph
- Inheritance
  - Field access
- Method call
  - The kind of use relation
Similar component

- Similar component is copied component or minor modified component
- We merge similar components into single component
- Merged component have use relations that all component before merging have

Clustering components

- We measure characteristics metrics to merge components
- The difference ratio of each component metrics
  - Metrics
    - Complexity
      - The number of methods, cyclomatic, etc.
    - Token-composition
      - The number of appearances of each token
      - Represent a structural characteristic

Component graph

Clustered component graph

Ranking based on use relation

- Component Rank (CR)
  - Reusable component have many use relation
    - The example of use is much
    - General purpose component
    - Sophisticated component
  - We measure use relation quantitatively, and rank components
    - The component used by many components is important
    - The component used by important component is also important

Propagating weights

- Ad-hoc weights are assigned to each node

Propagating weights

- The node weights are re-defined by the incoming edge weights
- We get new node weights

Propagating weights

• We get stable weight assignment next-step weights are the same as previous ones
• Component Rank : order of nodes sorted by the weight

Component retrieval part

• Search components from database, rank components
• The process
  ▶ Search components
  ▶ Ranking suited to a user request
  ▶ Aggregate two ranks (CR and KR)

Search components

• Search query
  ▶ Words a user input
  ▶ The kind of an index word, package name
• Components contain given query are searched from Database

Calculation of KR value

• Calculate weight $W_c$ with component $c$ word $t$
  ▶ $TF_t$: The frequency with which a kind $i$ of word $t$ occurs in component $c$
  ▶ $IDF_i$: The total number of components / the number of components containing word $t$
  ▶ $w_c = \sum w_{ki} \cdot TF_t \cdot IDF_i$

• KR value is the sum of all word $W_c$
Aggregate two ranks

- Aggregate two ranks KR and CR
- Aggregation method
  - Borda Count method known as a voting system
    - Use for single or multiple-seat elections
    - This form of voting is extremely popular in determining awards
  - SPARS-J
    - Rank components both KR and CR
    - Using KR and CR, the component that best satisfies user's request, reusable and sophisticated

Borda Count method

- There are 10 voters and 5 candidates (from A to E)
- Each voter ranks candidates
- 1 point for last place, 2 points for second from last place ... and N points for first place
- 1st = 5 points, 2nd = 4 points, ...
- A: 15 + 3 + 6 + 4 = 28 points
- B: 38 points
- C: 38 points
- D: 22 points
- E: 26 points

Outline

- Motivation and research aim
- SPARS-J
  - Outline
  - System architecture
  - Ranking method
  - Each part
    - Analysis part
    - Retrieval part
    - User Interface
  - Experiment
  - Conclusion and Future work

User interface

- Receive a user's query and provide the search results through Web browser
  - Microsoft Internet Explore, Mozilla, etc.
- The process
  - Parse query word and the search condition
  - Show rank ordered results
  - Show analyzed information of the component
    - Used by/Using the component
    - Metrics

Analyzed information

A component information are as follows
- Metrics
  - The number of method, variable
  - LOC, cyclomatic
  - Etc. (measurable metrics in the component itself)
- Components used by/using the component
  - Show lists of nodes followed use relation
- Components that are similar to the component
  - Show lists of similar components

Package browsing

- The naming structure for Java packages is hierarchical
  - A user can search lists of components in same package of a component easily
Outline

- Motivation and research aim
- SPARS-J
  - Outline
  - System architecture
  - Ranking method
  - Each part
    - Analysis part
    - Retrieval part
    - User Interface
- Experiment
- Conclusion and Future work

Experiment (1/2)

- Comparison with Google
  - Register about 130,000 components get from Internet
  - Query words ‘calculator applet’ and ‘chat server client’
    - Calculate relevance ratio of 10 rank higher
    - Relevance: The component is reusable source code
  - Google is a web search engine...
    - Add ‘java source’ term to the query words
    - Follow one link from the result web page

Experiment (2/2)

- Example 1:
  - "calculator applet"
  - SPARS-J
    - 9 hits
    - 7 suited components
- Example 2:
  - "chat server client"
  - SPARS-J
    - 69 hits
    - 57 suited components

Using SPARS-J, suited component is high order

Conclusion and Future work

- We developed component search engine SPARS-J
  - Using SPARS-J, retrieval of components used well is enabled easily.
- Future work
  - Morphological analysis of index keyword
  - Collaborative filtering
  - Investigate best ranking method
    - The value of weight
    - Aggregation ranks
  - Evaluation of SPARS-J
    - Usability