Information Visualization in Software Evolution and Maintenance

MARJANE NAMAVAR
UNIVERSITY OF BRITISH COLUMBIA
INFORMATION VISUALIZATION
FALL 2019
Background

**Software Evolution**: The process of developing software initially, then repeatedly updating it for various reasons

**Software Maintenance**: The modification of a software product after delivery to correct faults or improve performance

**Visualization in Software Evolution and Maintenance**: Mapping from corresponding software artifacts including programs, to graphical representations
Goals

✓ Survey the **existing literature** focusing on the use of visualization for software evolution and maintenance

✓ Analyze the data from empirical experiments under **what/why/how framework**

✓ Abstract gathered information to **categorize** existing approaches
Inclusion Criteria

• 23 papers were gathered
• Design study
• VISSOFT conference
• Under maintenance and evolution categories
• 2003-2019
• A visualization system which is central to that research and has a task
Categories

Task → Data → Data Processing → Representation → Availability
Categories -> Task

- Help to detect code smells
- Help to analyze execution of the program
- Help to perform debugging
- Help to analyze user feedback
- Help to monitor code changes
- Help to monitor developer activities
Categories -> Data

• Source code
• Packages
• Classes and objects
• Test suite
• Bug report
• Events & sequences

• Relationships between code components
• User feedback
• Metadata (such as version information)
Categories -> Data Processing

- Abstract Syntax Tree
- NLP Methods
- Static Analysis
- Dynamic Analysis
Categories -> Representation

- Techniques
- Textual Content (of the artifact being visualized)
Categories -> Availability

• Scalability (supports millions of LOC)
• Integration (with IDE)
CVScan: Visualization of Code Evolution

A multi-view environment including:

• Line-oriented display of the changing code
• Each version is represented by a column
• Horizontal direction is used for time
• Source code
• A large variety of options

CVScan: Analysis

Task

Help to monitor code changes and developer activities
- What code lines were added, removed, or altered and when?
- Which parts of the code are unstable?
- How are changes correlated?
- Who performed these modifications of the code?

Data Processing

Static analysis to compute:
- Line position
- Line status

Data

Source code (lines of code in different versions)
Events and Sequences (sequence of commits)
Metadata (<id,author,date,code> for each version)

Availability

Scalability: Y
Integration: Y
CVScan: Analysis (cont.)

Representation

Textual Content: Y

Techniques:
Encode: 2D plot, Color-map, Position
Reduce: Filter
Facet: Partition into multiform views, Juxtapose views, Linked highlighting, Linked navigation, Overview–detail
Manipulate: Select, Zoom and Pan
Discussion

- The most investigated task is monitoring code changes
- Animations become appealing to researchers
- NLP methods are applied recently
- The main challenge is the large amount of complex data
Questions?