Reverse Engineering of Legacy Systems

Mitel Corporation
University of Ottawa

Progress Presentation
CASCON November 1998

Outline:

How we discover what software engineers do
 • Synchronized Shadowing
 • Work patterns in UCMs

Work patterns
 • Just in Time Comprehension
 • Search
 • Switching Among Tools
 • Copying and Pasting

Tool Requirements

The WASS methodology

'Work Analysis with Synchronized Shadowing'

1. Learn about basic needs of your users (software engineers)

2. Gather structured data about them using Synchronized Shadowing:
  • Actions
  • Goals, plans tasks

3. Develop Use Case Maps describing patterns of work

4. Derive requirements from all of the above

Synchronized Shadowing

Involves two 'observers'

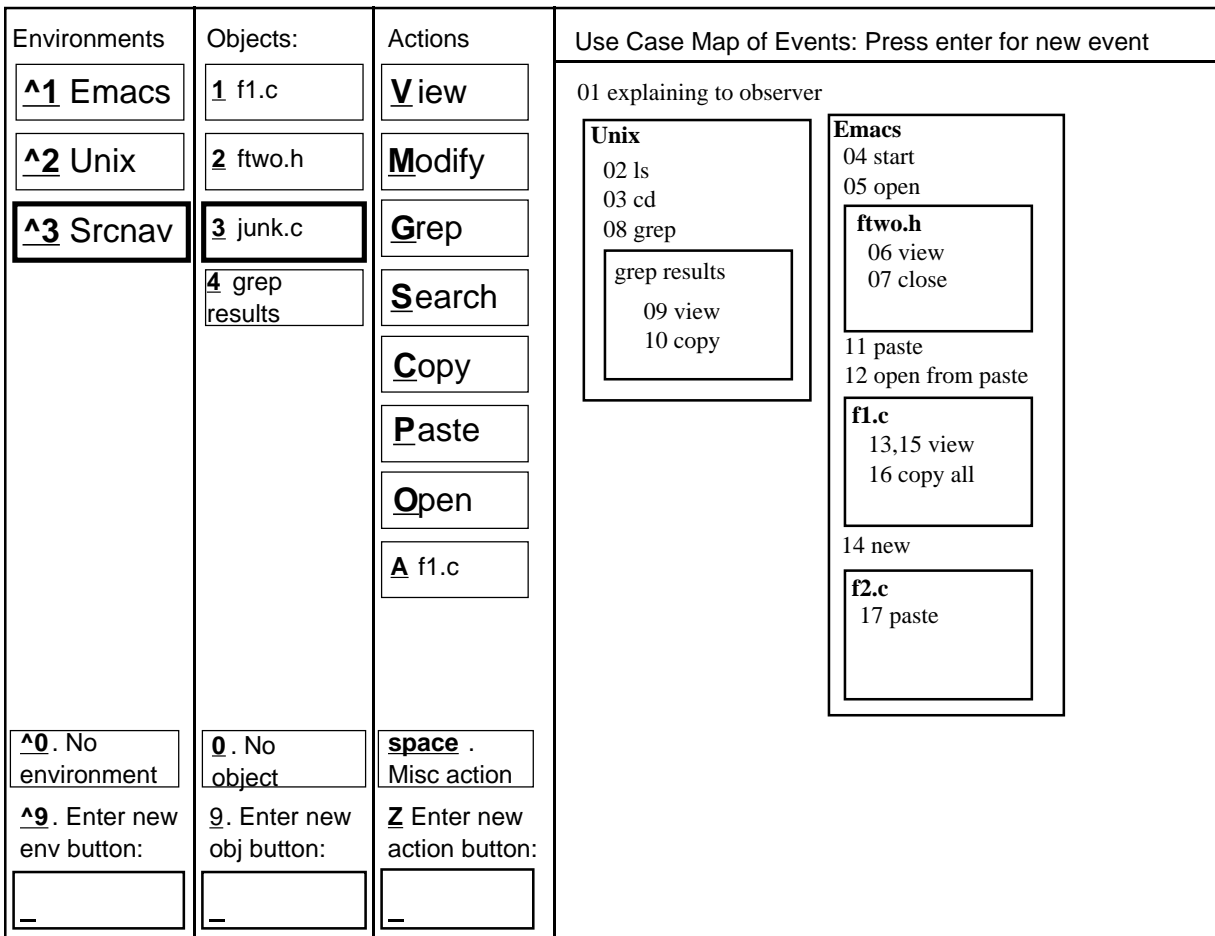Working on laptops

Each observing a different kind of data

Clocks are synchronized

Data is merged later to obtain composite view
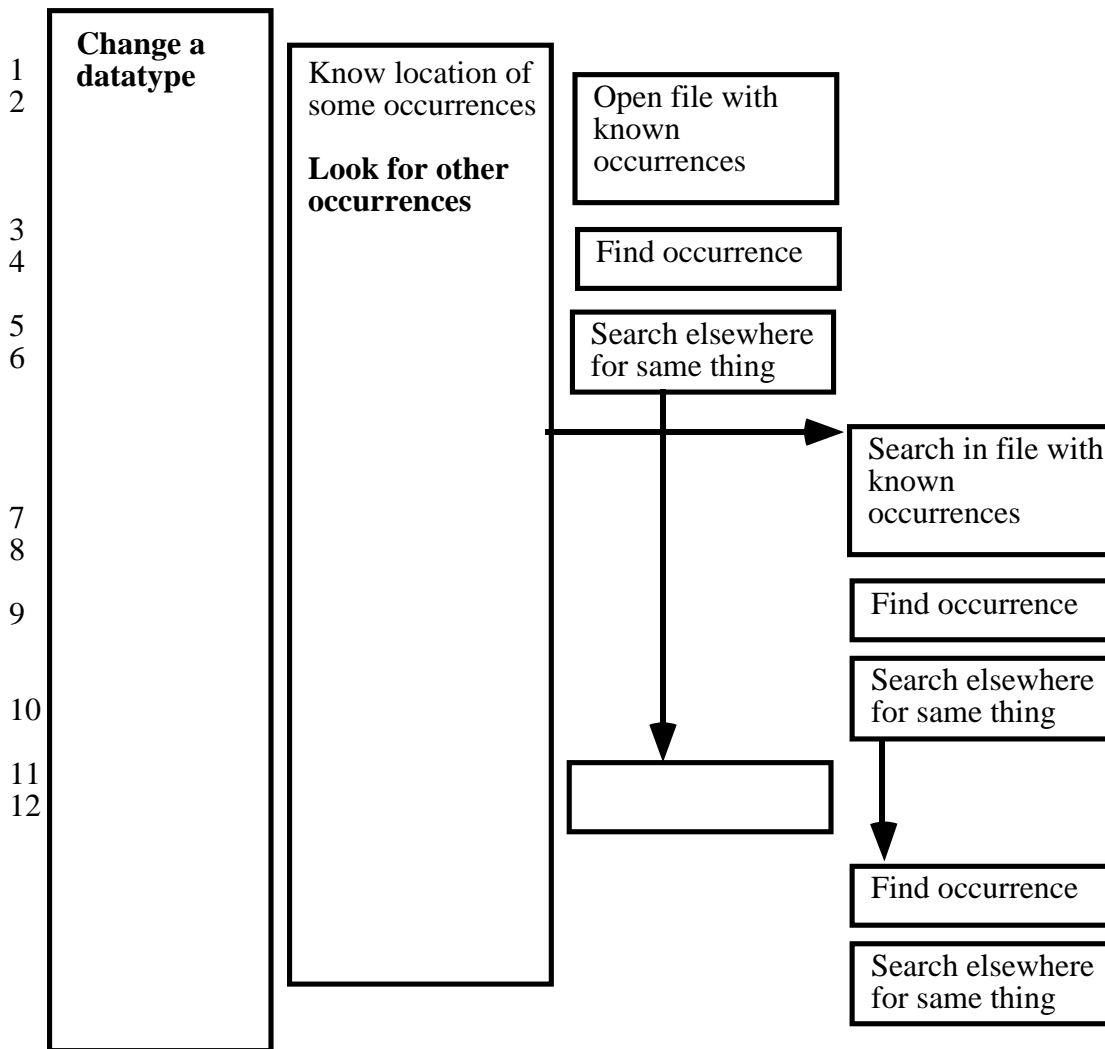
Synchronized Shadowing

The observer of actions

Environments	Objects:	Actions	Use Case Map of Events: Press enter for new event
^1 Emacs	**1** f1.c	**V**iew	01 explaining to observer
^2 Unix	**2** ftwo.h	**M**odify	**Unix**
^3 Srcnav	**3** junk.c	**G**rep	
	4 grep results	**S**earch	
		Copy	
		Paste	
		Open	
		A f1.c	

In the "Unix" box:
Unix
02 ls
03 cd
08 grep

grep results
   09 view
   10 copy

In the "Emacs" box:
Emacs
04 start
05 open

ftwo.h
   06 view
   07 close

11 paste
12 open from paste

f1.c
   13,15 view
   16 copy all

14 new

f2.c
 17 paste

Bottom row:

^0 . No environment	**0** . No object	**space** . Misc action	
^9 . Enter new env button:	**9** . Enter new obj button:	**Z** Enter new action button:	
_	_	_	

Synchronized Shadowing

The observer of tasks, goals etc

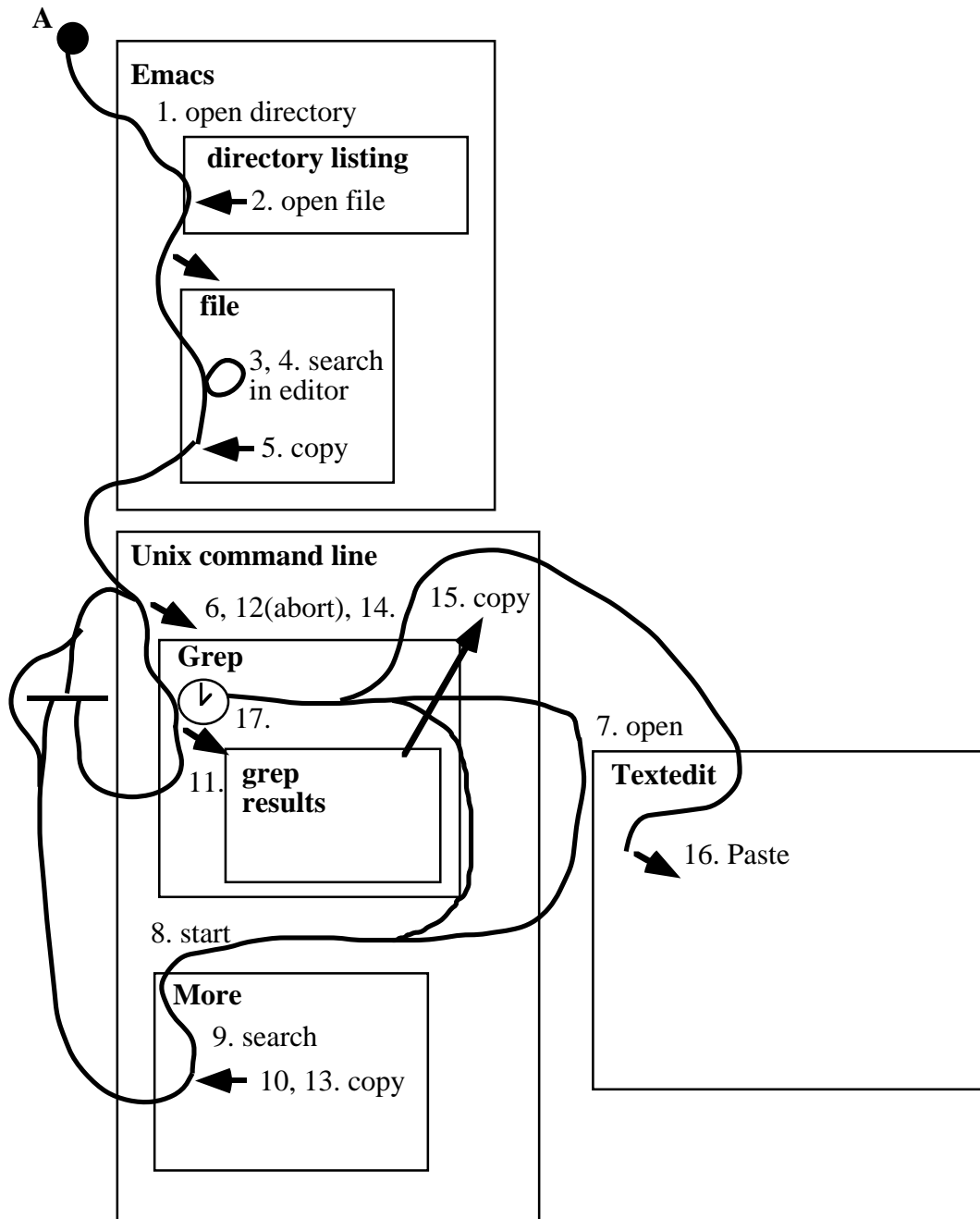|				
1 2	**Change a datatype**	Know location of some occurrences	Open file with known occurrences	
		Look for other occurrences		
3 4			Find occurrence	
5 6			Search elsewhere for same thing	
				Search in file with known occurrences
7 8				
9				Find occurrence
				Search elsewhere for same thing
10				
11 12				
				Find occurrence
				Search elsewhere for same thing

Resulting Data

1 13:32:40 NEW-GOAL Friday, August 01, 1997 Jane Smith
2 13:39:26 still explaining stuff
3 13:39:52 UNIX ls cd
4 13:40:12 EDITOR srh
5 13:40:22 EDITOR quit
6 13:40:28 GREP in system
7 13:40:40 VIS at results
8 13:41:02 EDITOR open found file
9 13:41:33 EDITOR open empty
10 13:41:45 EDITOR copy
11 13:41:52 EDITOR paste
12 13:42:00 EDITOR save as xxdbllq.c
13 13:42:21 MODIFY part of query text
14 13:44:08 EDITOR save
15 13:44:12 MODIFY func name
16 13:44:38 EDITOR save
17 13:44:59 stop observing

1 13:32:30 NEW-GOAL Friday, August 01, 1997 Jane Smith
2 13:33:21 GOAL-SUB writing new database query
3 13:34:50 EXPLAN will be invoked from two different parts of the UI
4 13:36:20 GOAL-SUB create code for the query
4 13:37:39 PLAN will clone code for similar query
5 13:40:00 HYPOTHESIS suitable query to copy is in file xvdbllq.c
6 13:40:24 FAILURE couldn't find code there
7 13:40:31 GOAL-SUB search in system for suitable code to clone
8 13:40:50 EXPLAN Saw code before somewhere
9 13:41:01 SUCCESS File xwdbllq.c has it
10 13:41:56 SUCCESS New file created
11 13:44:40 GOAL-SUB Add UI to fire off new query
12 13:44:55 end of session

Use Case Maps to Analyze Data

A

Emacs

1. open directory

directory listing

2. open file

file

3, 4. search in editor

5. copy

Unix command line

6, 12(abort), 14.

15. copy

Grep

17.

11.

grep results

7. open

Textedit

16. Paste

8. start

More

9. search

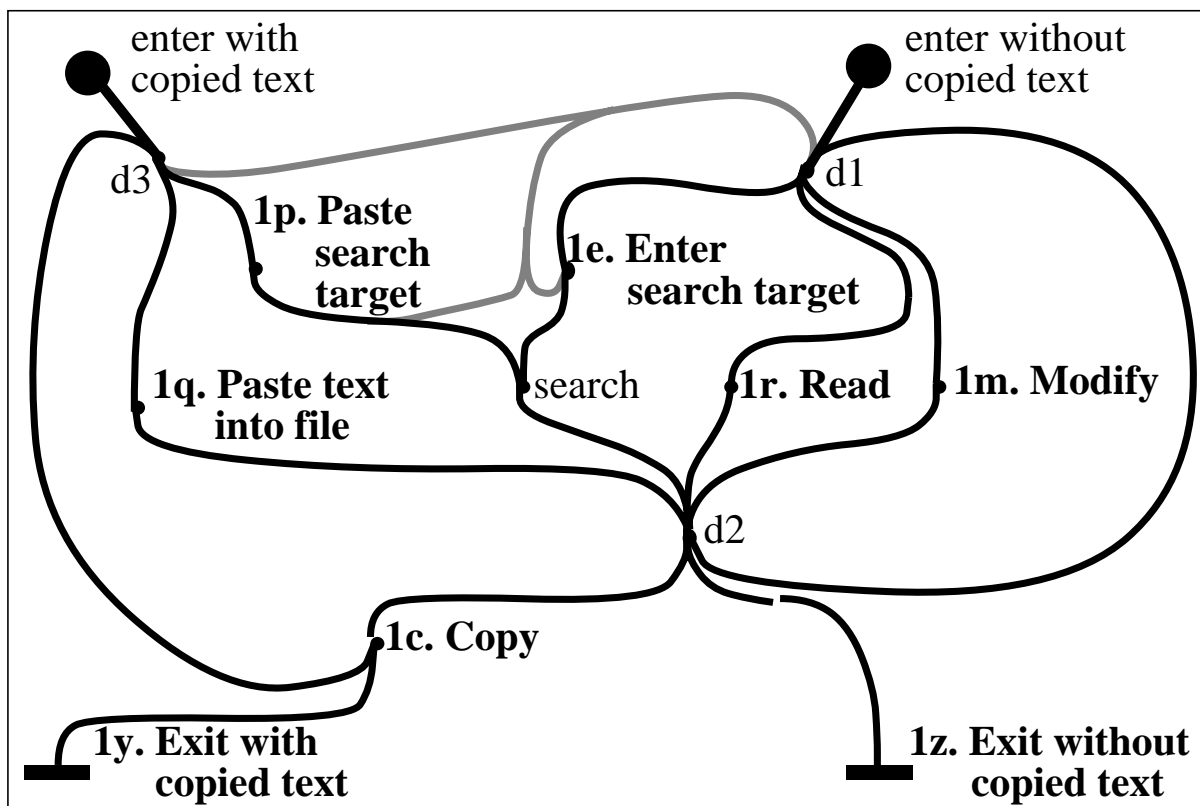10, 13. copy

Work patterns

Just-in-time comprehension

- Software engineers seek to understand just what they need to solve the current problem

- They can't ever hope to learn or remember more than just a small part of the system

Work patterns ...

Use of files

enter with copied text
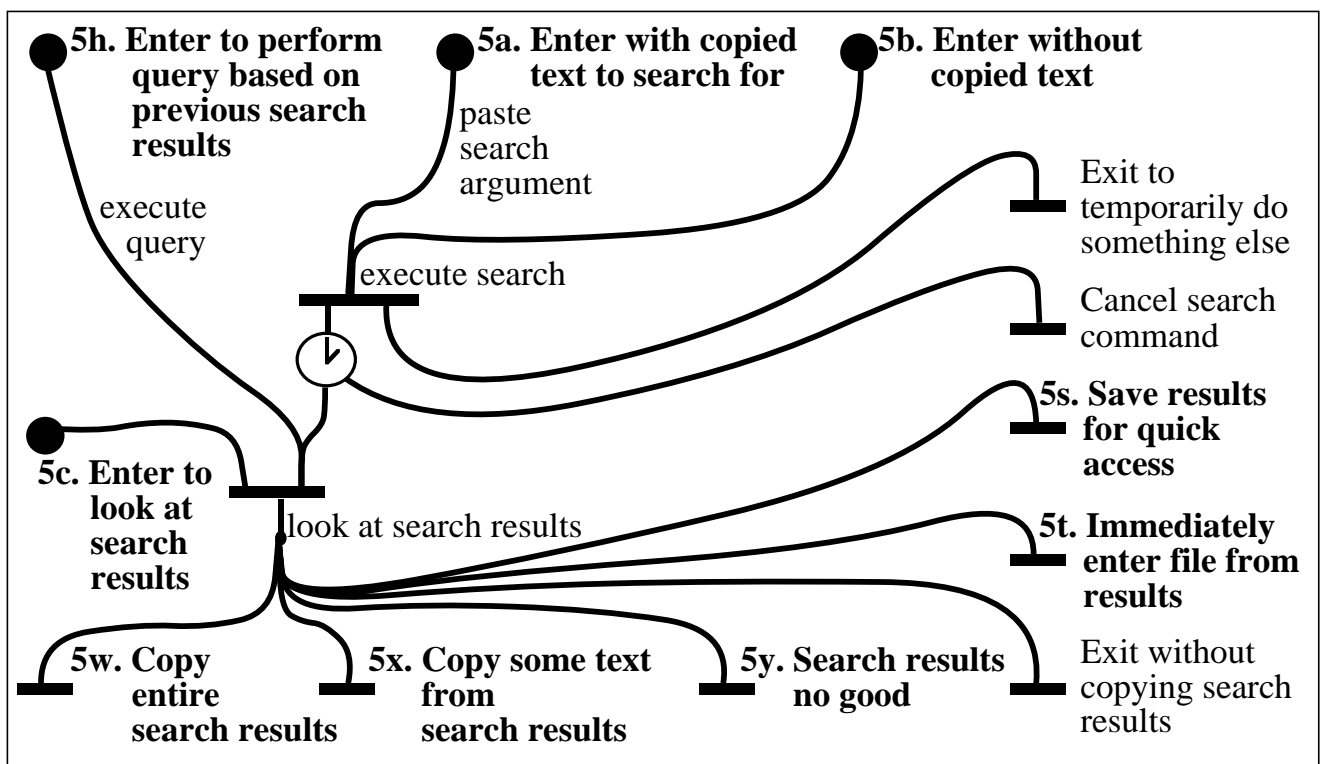
enter without copied text

d3

d1

1p. Paste search target

1e. Enter search target

1q. Paste text into file

search

1r. Read

1m. Modify

d2

1c. Copy

1y. Exit with copied text

1z. Exit without copied text

Work patterns ...

Use of editors

enter editor with
copied text from another program

enter editor with
no copied text

pattern 2:
open or enter
a file preserving
copied text

2a

pattern 2:
open or enter
a file using
any copied
text

enter with
copied text

enter without
copied text

pattern 1:
work on file
in editor

1y. exit with
copied text

exit without
copied text

leave editor with
copied text to use
in another program

leave editor
without copied
text

Work patterns ...

Search

5h. Enter to perform query based on previous search results

execute query

5a. Enter with copied text to search for

paste search argument

5b. Enter without copied text

Exit to temporarily do something else

execute search

Cancel search command

5c. Enter to look at search results

look at search results

5s. Save results for quick access

5t. Immediately enter file from results

Exit without copying search results

5w. Copy entire search results

5x. Copy some text from search results

5y. Search results no good

Work patterns

Copy and pasting of data

Event types	Percent of Total	Standard Deviation	Percent of Subtotals	Standard Deviation	Number of Events
Total number of events					**966**
Copy text	**9.2%**	**6.0%**			**89**
Copy from file (1c, 7c)			66.3%	22.0%	59
Copy from search results (4x, 5x)			34.8%	23.8%	31
Search	**28.3%**	**10.3%**			**273**
Search in editor (1p, 1e, 7p, 7e)			59.3%	17.1%	162
Search in system			40.7%	17.1%	111
Using grep (4a, 4b)			23.8%	15.6%	65
Using other tool (5a, 5b)			16.8%	16.6%	46
Study	**28.5%**	**13.9%**			**275**
Study in editor (1r, 7r, 4t, 5t)			87.6%	7.1%	241
Study search results (4c, 5c)			12.4%	7.1%	34
Paste text	**10.6%**	**7.0%**			**102**
Paste to modify text (1q, 7q)			7.8%	23.5%	8
Paste to search in editor (1p, 7p)			32.4%	22.7%	33
Paste to open file (2a, 8a)			22.5%	16.6%	23
Paste to search in system (4a, 5a)			37.3%	33.0%	38

	Copy from file	Copy from search results
Paste to modify text	5.1%	16.1%
Paste to search in file	27.1%	9.7%
Paste to open file	1.7%	71.0%
Paste as search argument	47.5%	3.2%
Not immediately used	18.6%	0.0%

Requirements for a source browsing tool 1

Strong support for search of all types

Make good use of screen real-estate

Make transitions from one type of information to another easy to do

Allow heterogeneous graph structures in exploration

Allow user to build their own mental model

Requirements for a source browsing tool 2

Allow easy context switching and return

Make it fast

Make it work with multiple languages

Make it USABLE!!!!!

See our Demonstration