Programming Language Design

2015

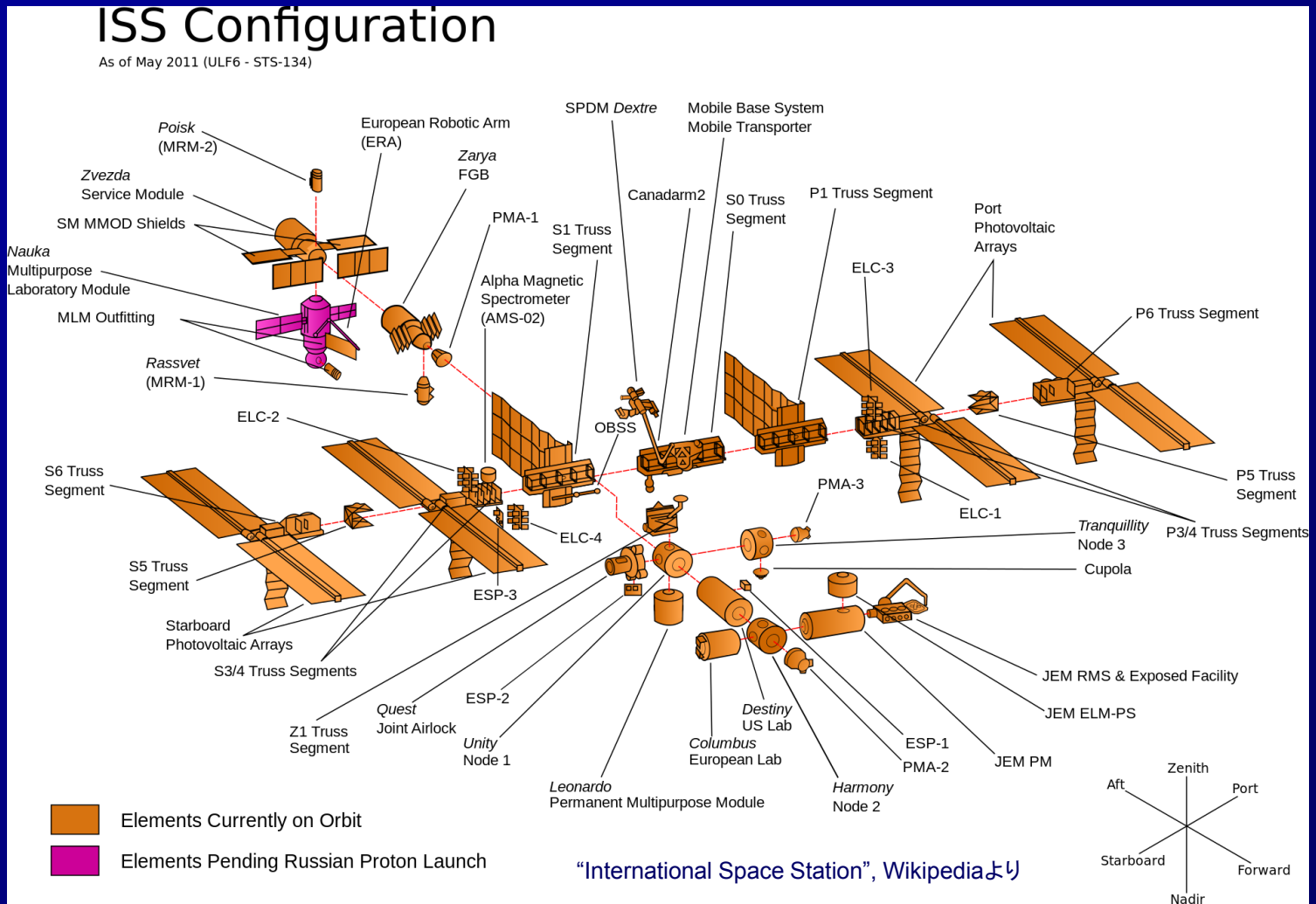
Week #1: Modules

Instructor: Hidehiko Masuhara

Module

- Definition (of a general "module"):
  - one of a set of parts
  - that can be connected or combined
  - to build or complete something (from Merriam-Webster)
- Characteristics:
  - Unit of development, exchange & maintenance
  - Two sides: developers & users
  - Can be nested (i.e., modules in a module)
  - Typically a physically connected unit

An example of a module: International Space Station (ISS)



"International Space Station", Wikipediaより

Example problem: Key Word in Context (KWIC) system [Par72]

Output

:

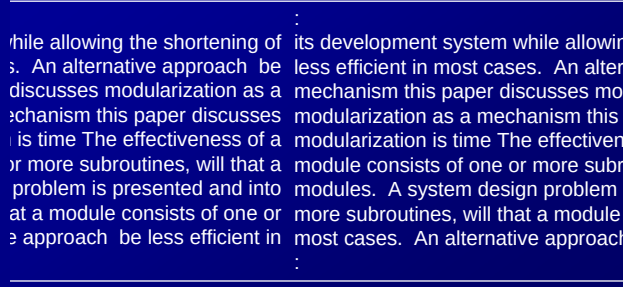
hile allowing the shortening of
s. An alternative approach be
discusses modularization as a
echanism this paper discusses
is time The effectiveness of a
or more subroutines, will that a
problem is presented and into
at a module consists of one or
e approach be less efficient in

:
its development system while allowin
less efficient in most cases. An alter
mechanism this paper discusses mo
modularization as a mechanism this
modularization is time The effectiven
module consists of one or more subr
modules. A system design problem i
more subroutines, will that a module
most cases. An alternative approach

:

例題: Key Word in Context (KWIC)システム

出力



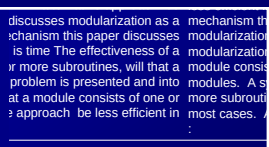
Algorithm

- read a file line by line
- for each line, split it into a sequence words, and generate w sequences by rotating 0..(w-1) words
- sort all the generated sequences in the dictionary order
- output the sorted sequences

Today's quiz (1/3)

- Write your name and student ID on the sheet
- Q1: Design the KWIC system with around five modules. Describe the role of each module and names and roles of functions in each module.

Algorithm



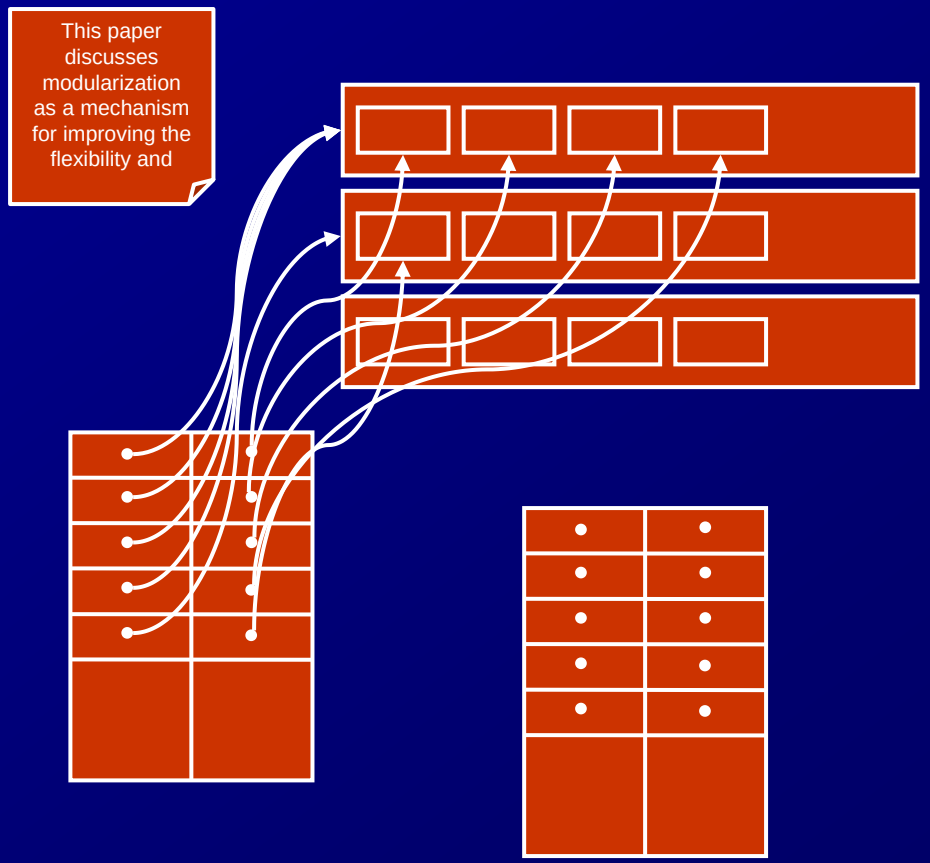
- read a file line by line
- for each line, split it into a sequence words, and generate w sequences by rotating 0..(w-1) words
- sort all the generated sequences in the dictionary order
- output the sorted sequences

Today's quiz (2&3 /3)

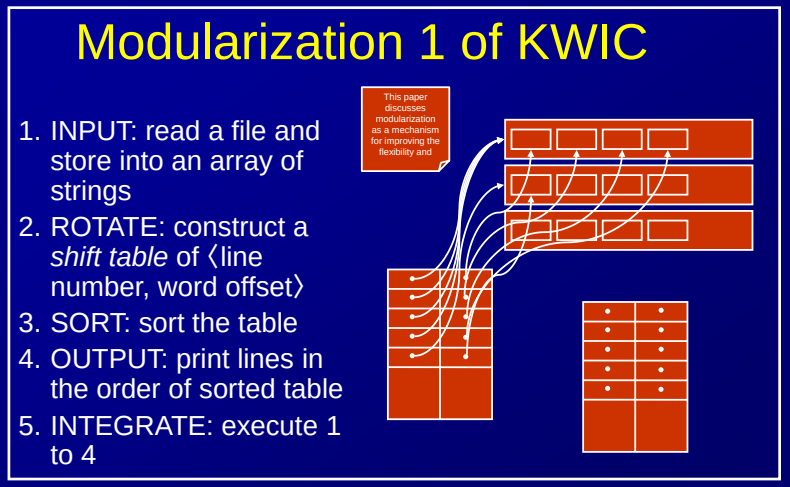
- Q2: Invent a way to compare designs of software with respect to the "better modularization"
- Q3: Which is "better modularized", open source software or proprietary software? Explain with reasons. Assume you are comparing software systems about the same size.

Modularization 1 of KWIC

1. INPUT: read a file and store into an array of strings
2. ROTATE: construct a *shift table* of ⟨line number, word offset⟩
3. SORT: sort the table
4. OUTPUT: print lines in the order of sorted table
5. INTEGRATE: execute 1 to 4



Modularization 1 of KWIC



- Data structures:
  - lines: String[]
  - shift table: int[][]
- Interface
  - INPUT: String[] input(String fileName)
  - ROTATE: int[][] shift(String[] lines)
  - SORT: int[][] sort(String[] lines, int[][] shift)
  - OUTPUT: void output(String[] lines, int[][] shift)
  - INTEGRATE: void main()

Modularization 2 of KWIC

1. LINE STORE
   - CHAR(r,w,c) gives c'th character of w'th word at line r
   - SETCHAR(r,w,c,d), WORDS(r), DELINE, DELWRD ...
2. INIPUT: read lines from a file and store in LINE STORE
3. ROTATE:
   - CSCHAR(l,w,c) gives c'th character of w'th word of l'th rotation
   - CSSETUP
4. SORT:
   - ITH(i) gives the i'th rotation number in a sorted table
   - ALPH
5. OUTPUT
6. INTEGRATE

Modularization 2 of KWIC

1. LINE STORE
   - char getChar(int row, int word, int offset)
   - void setChar(int row, int word, int offset, char c)
   - int numWords(int row)
2. INPUT: input(String fileName)
3. ROTATE:
   - void initialize()
   - char getChar(int shift, int word, int offset)
   - int numWords(int shift)
4. SORT
   - void doSort()
   - int getShift(int i)
5. OUTPUT: void output()
6. INTEGRATE: void main()

Which is better modularized?

Modularization 1 c

- Data structures:
  - lines: String[]
  - shift table: int[][]
- Interface
  - INPUT: String[] input(String fileName)
  - ROTATE: int[][] shift(String[] lines)
  - SORT: int[][] sort(String[] lines, int[][] shift)
  - OUTPUT: void output(String[] lines, int[][] shift)
  - INTEGRATE: void main()

Modularization 2 of KWIC

LINE STORE
- CHAR(r,w,c) gives c'th character of w'th word at line r
- SETCHAR(r,w,c,d), WORDS(r), DELINE, DELWRD ...

INIPUT: read lines from a file and store in LINE STORE

ROTATE:
- CSCHAR(l,w,c) gives c'th character of w'th word of l'th rotation
- CSSETUP

SORT:
- ITH(i) gives the i'th rotation number in a sorted table
- ALPH

OUTPUT

INTEGRATE

Modularization Principle of Parnas [Par72]

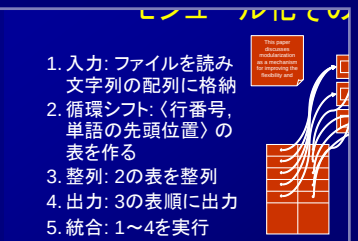
... begins with a list of difficult design decisions or design decisions which are likely to change. Each module is then designed to hide such a decision from the others.

Likely to change decisions in KWIC

- read all lines in memory←→in 2ndary storage
- string representation: String ←→byte sequence
- representation of a rotate: copy strings←→pointer
- range of output array: all at once←→on demand

Modularization 1

- Data structures:
  - lines: String[]
  - shift table: int[][]
- Interface
  - INPUT: String[] input(String fileName)
  - ROTATE: int[][] shift(String[] lines)
  - SORT: int[][] sort(String[] lines, int[][] shift)
  - OUTPUT: void output(String[] lines, int[][] shift)
  - INTEGRATE: void main()



Likely to change decisions in KWIC

- read all lines in memory←→in 2ndary storage
- string representation: String ←→byte sequence
- representation of a rotate: copy strings←→pointer
- range of output array: all at once←→on demand

Modularization 2

LINE STORE
- CHAR(r,w,c) gives c'th character of w'th word at line r
- SETCHAR(r,w,c,d), WORDS(r), DELINE, DELWRD ...

INIPUT: read lines from a file and store in LINE STORE

ROTATE:
- CSCHAR(l,w,c) gives c'th character of w'th word of l'th rotation
- CSSETUP

SORT:
- ITH(i) gives the i'th rotation number in a sorted table
- ALPH

OUTPUT

INTEGRATE

Metrics of modularity

- How can we measure degree of modularity of software design?

In other disciplines: business mgmt.

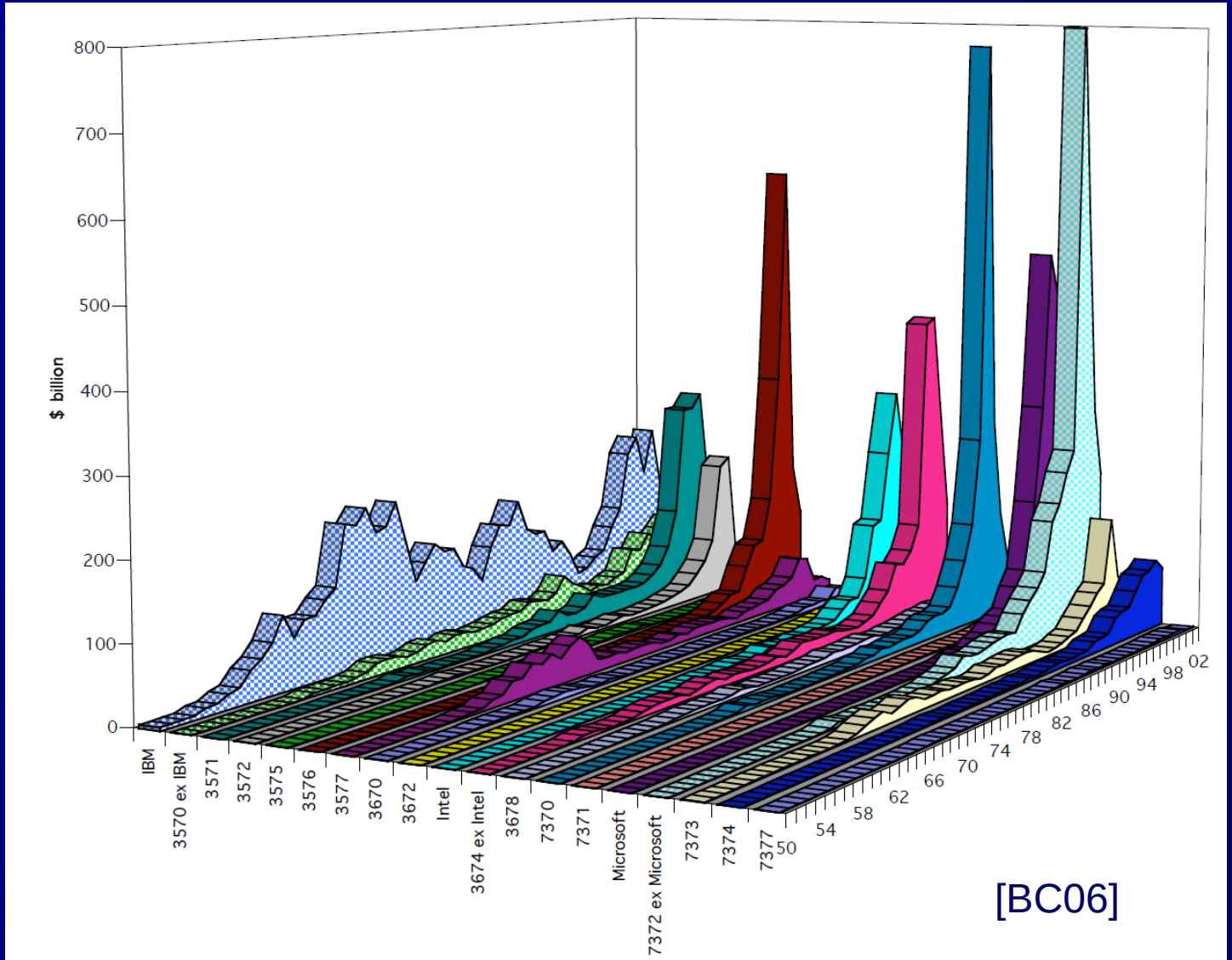
- Market structure in computer industry and modularity [BC06]
- Modularity in the auto and bicycle industries [Fujimoto04]

Market structure of computer industry

more modular

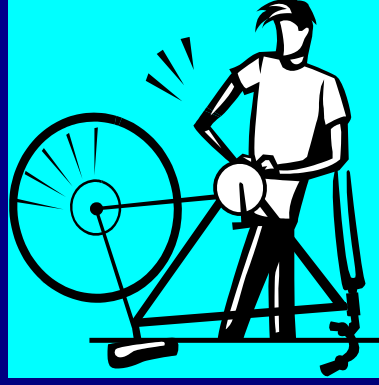
↓

more financial growth

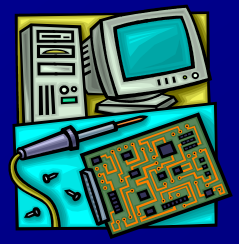


[BC06]

Industry and modularity



- Which is more modular, cars or bicycles?



modular

integral

Metrics of modularity

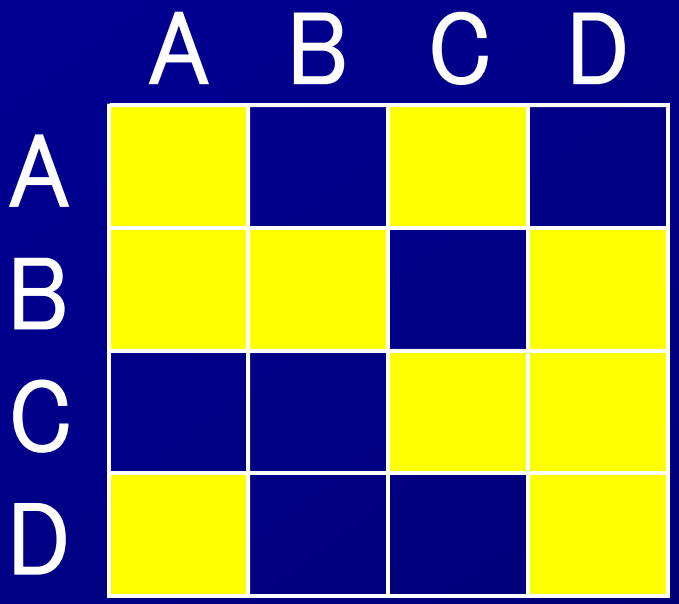
- Parnas' Principle:
  hide difficult/likely to change design decisions
- A metrics: ***change impact***
  - of what change: interface of a module
  - impact: number of modules to be updated

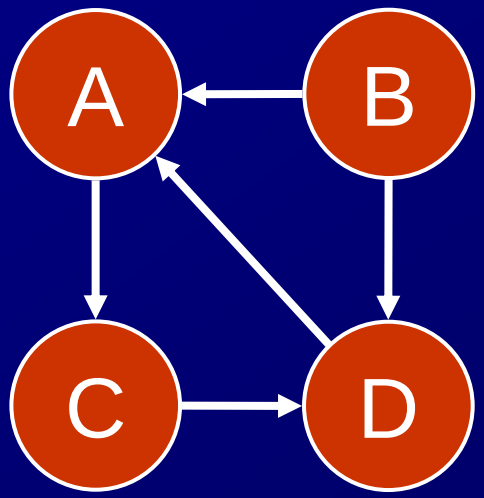
Metrics of modularity

- Design structure matrix
- Change impact analysis

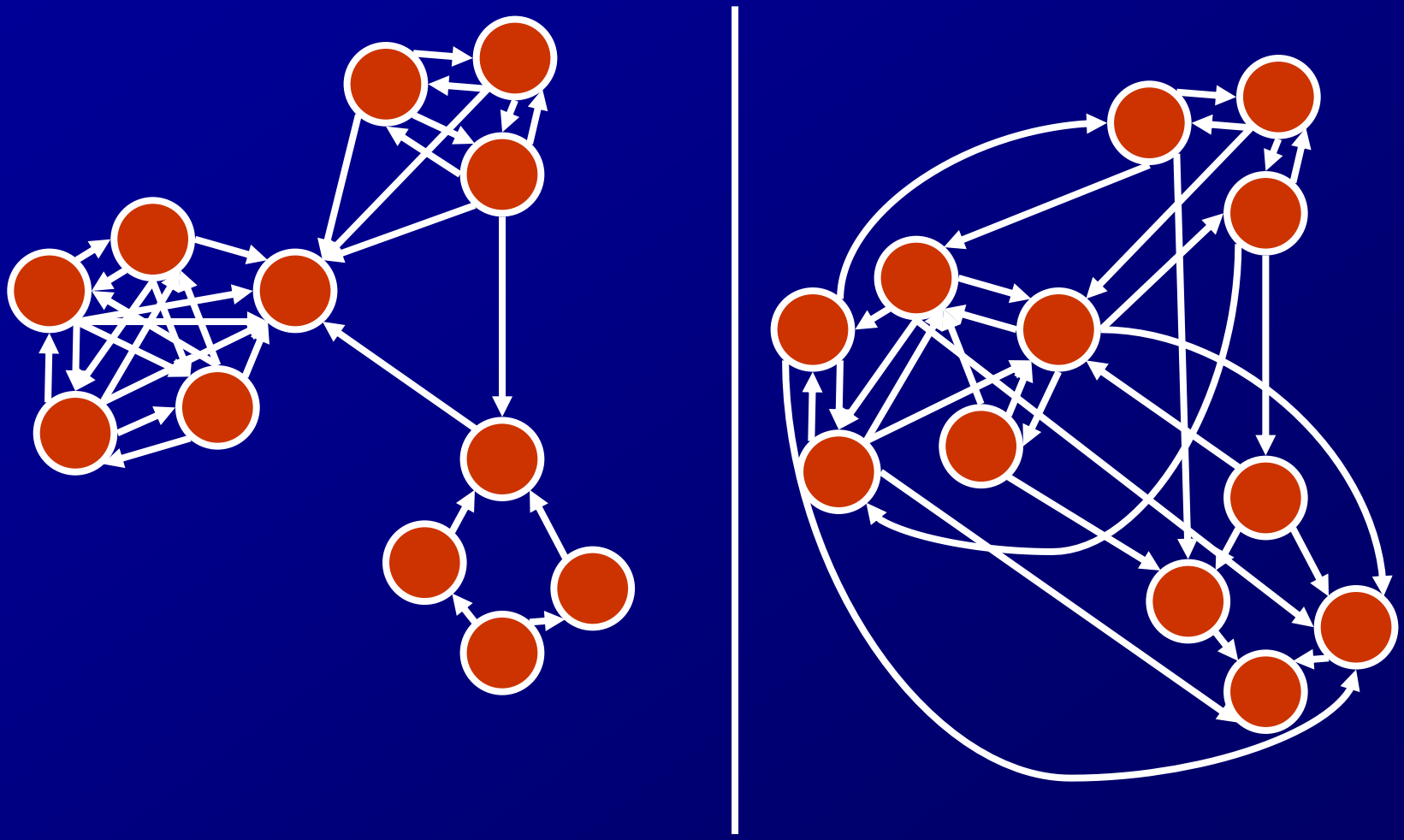
Design structure matrix (graph) [BC06]

- row i col j: i depends on j

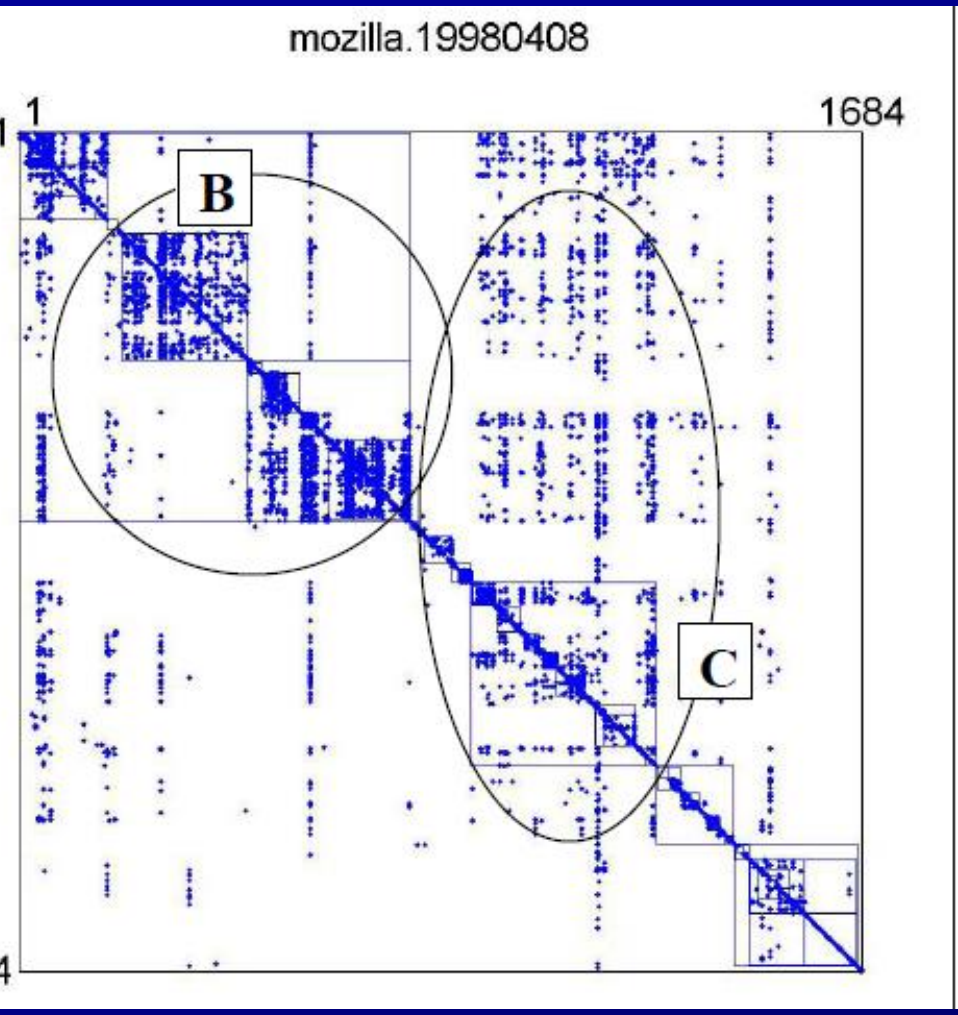


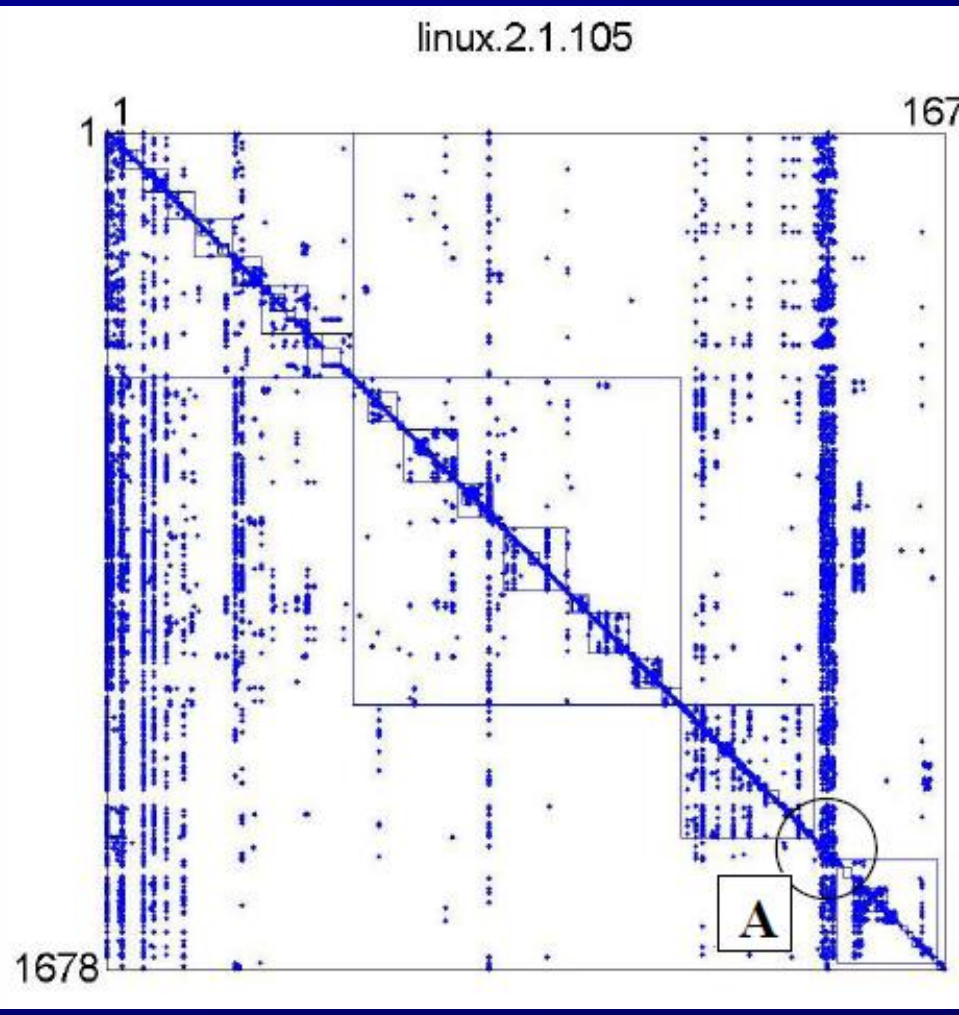


Goodness of modularity through design structure



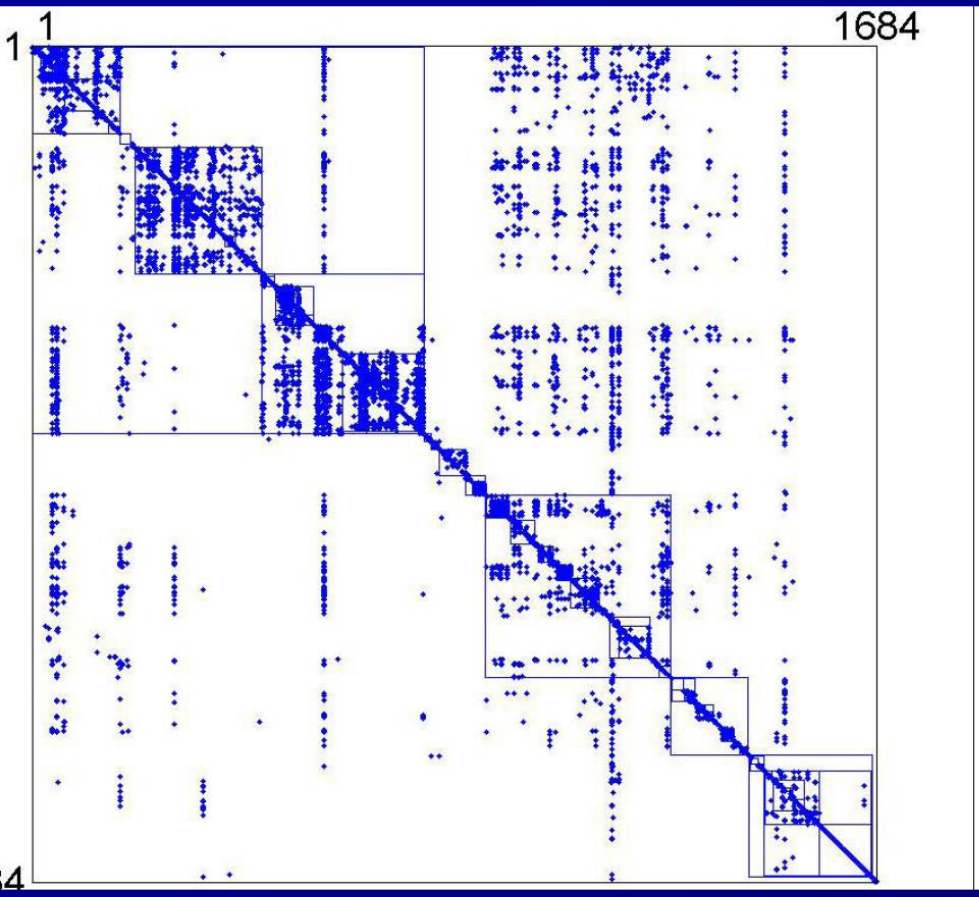
Analysis w/DSM [MRC06]



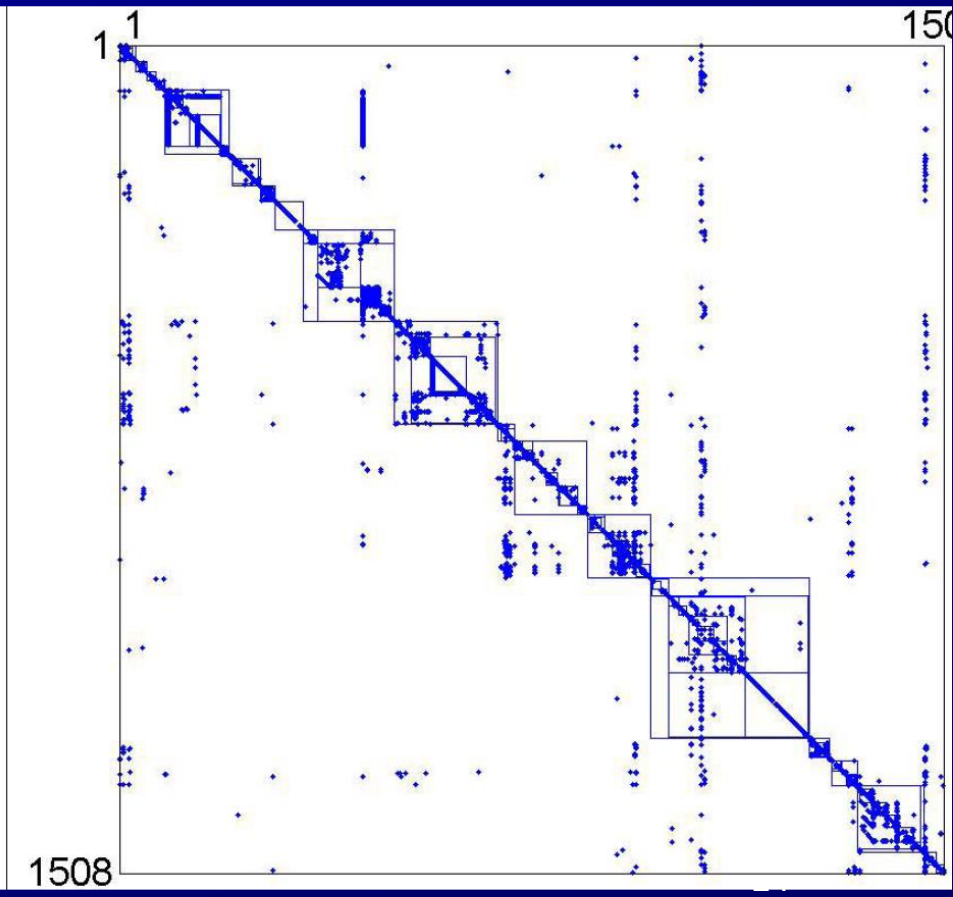


Evolution of DSM

- mozilla.19980408



- mozilla.19981211

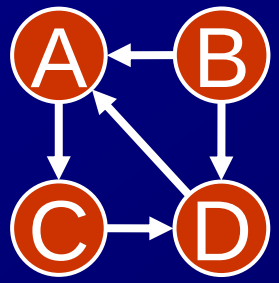


Change impact analysis: measuring a degree of modularization

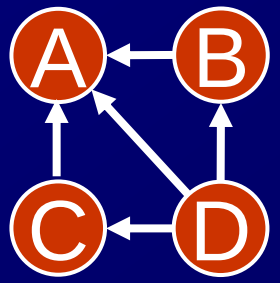
- (many proposals)
- propagation cost
  - M: set of modules
  - d(m): # modules m depends on

$$\frac{\sum_{m \in M} |d(m)|}{|M|^2}$$

assuming transitivity of change impact

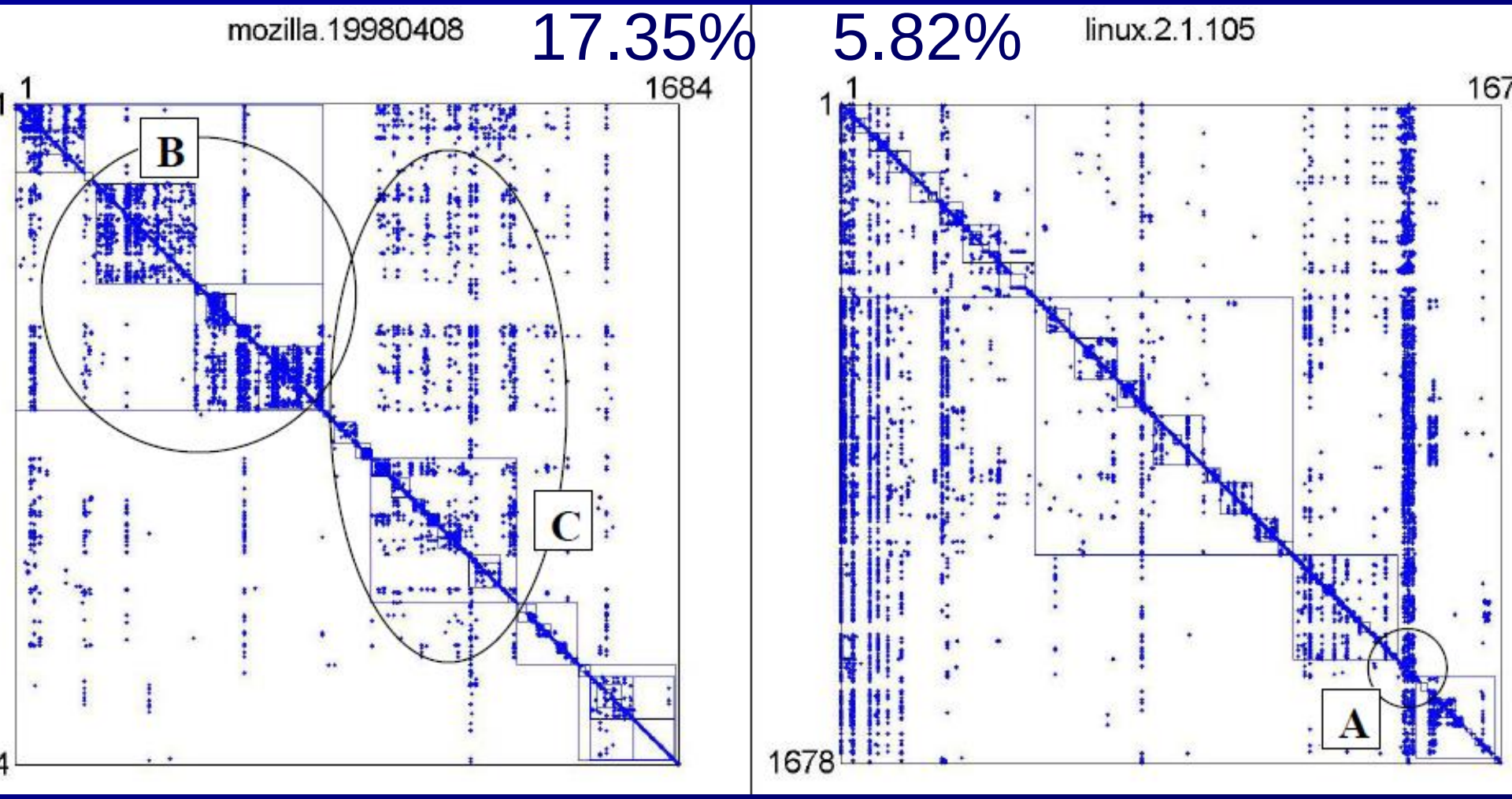


13/16=81%



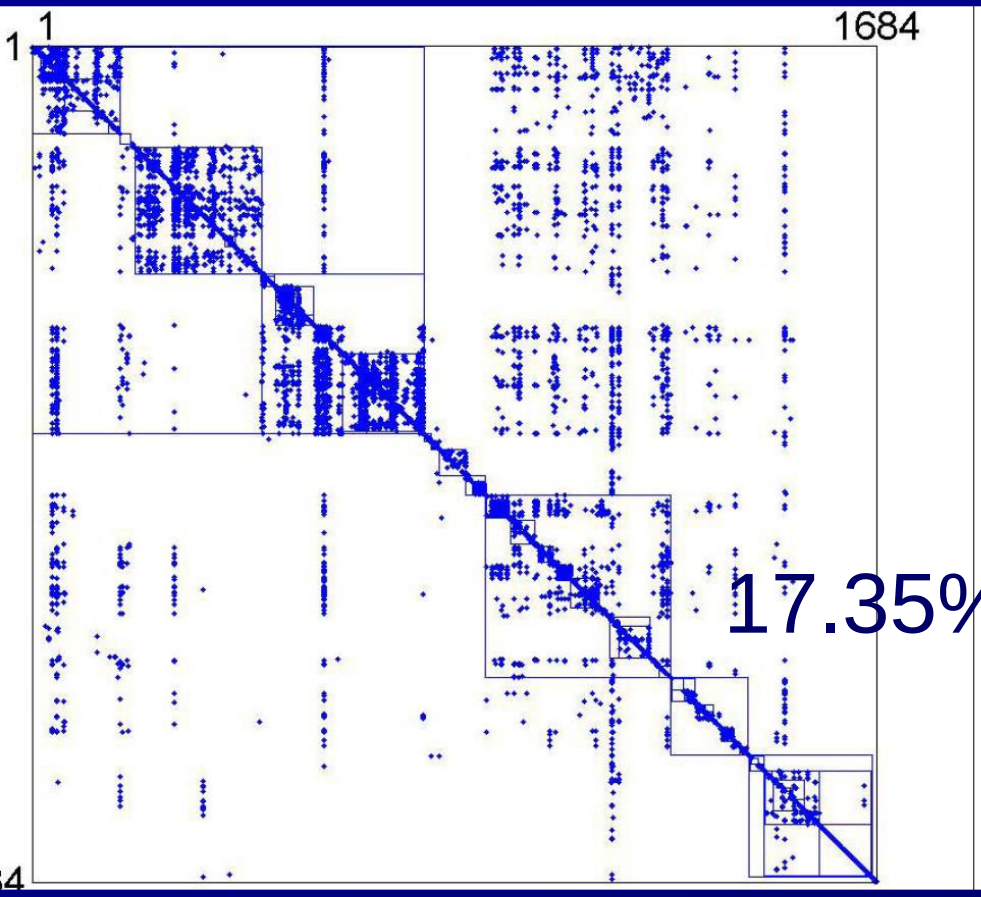
5/16=31%

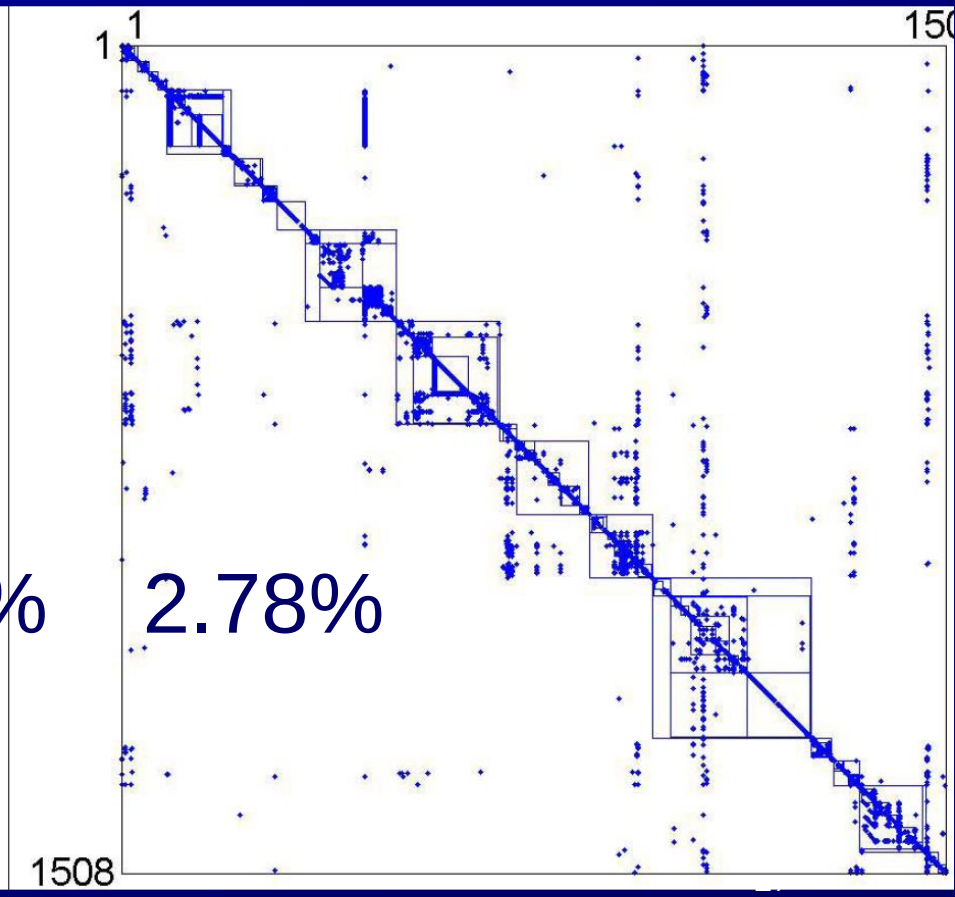
Change impact analysis



Change impact analysis

- mozilla.19980408
- mozilla.19981211





Observations

Compared approx. same size systems

- open source software
- proprietary software

- OSS is more modular
- Refactoring makes a system more modular

Open question: why?

References

[BC06] Carliss Y. Baldwin and Kim B. Clark, “Modularity in the Design of Complex Engineering Systems,” in *Complex Engineered Systems: Science Meets Technology*, Springer, 2006.

[MRC06] Alan MacCormack, John Rusnak and Carliss Y. Baldwin, “Exploring the Structure of Complex Software Designs: An Empirical Study of Open Source and Proprietary Code,” in *Management Science* 52(7), 2006.

[Par72] D. L. Parnas, “On the Criteria to be Used in Decomposing Systems into Modules”, in *Communications of the ACM* 15(12)1053-1058, 1972.

[Fujimoto04] 藤本隆宏, 日本のもの造り哲学, 日本経済新聞社, 2004