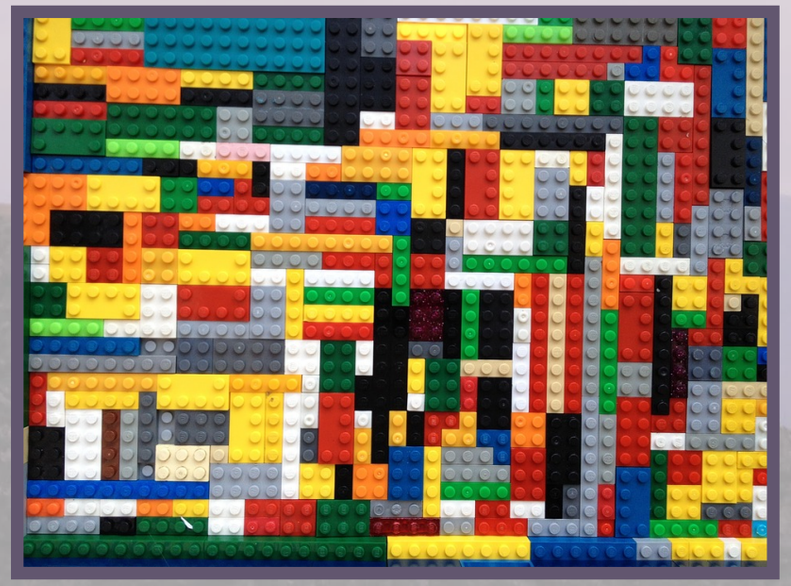
Modularity in Software Design

Anna Eilertsen
anna.eilertsen@uib.no

ICT Research School
Annual Meeting 2020

Modularity in Software Design

- Modularity 101
- Modular Software
- Module-Related Problems



Modularity 101

"Modules are units in a larger system that are structurally independent of one another, but work together."(1)

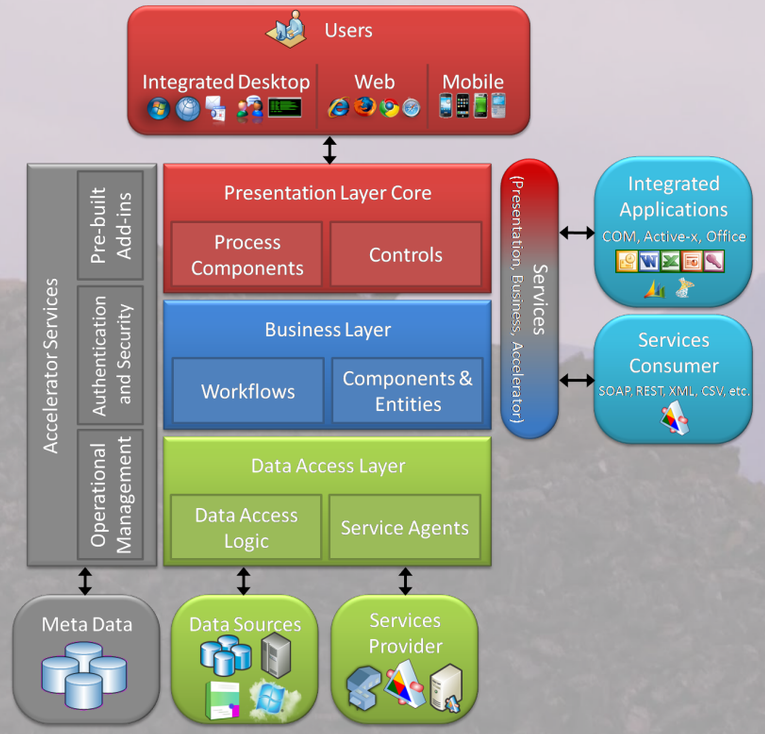
Modularity 101

Benefits from a modular design:

A. Makes complexity manageable;
B. Enables parallel work; and
C. Accommodates future uncertainty. (3)

Modularity in Software Design

- Modularity 101
- Modular Software
- Module-Related Problems

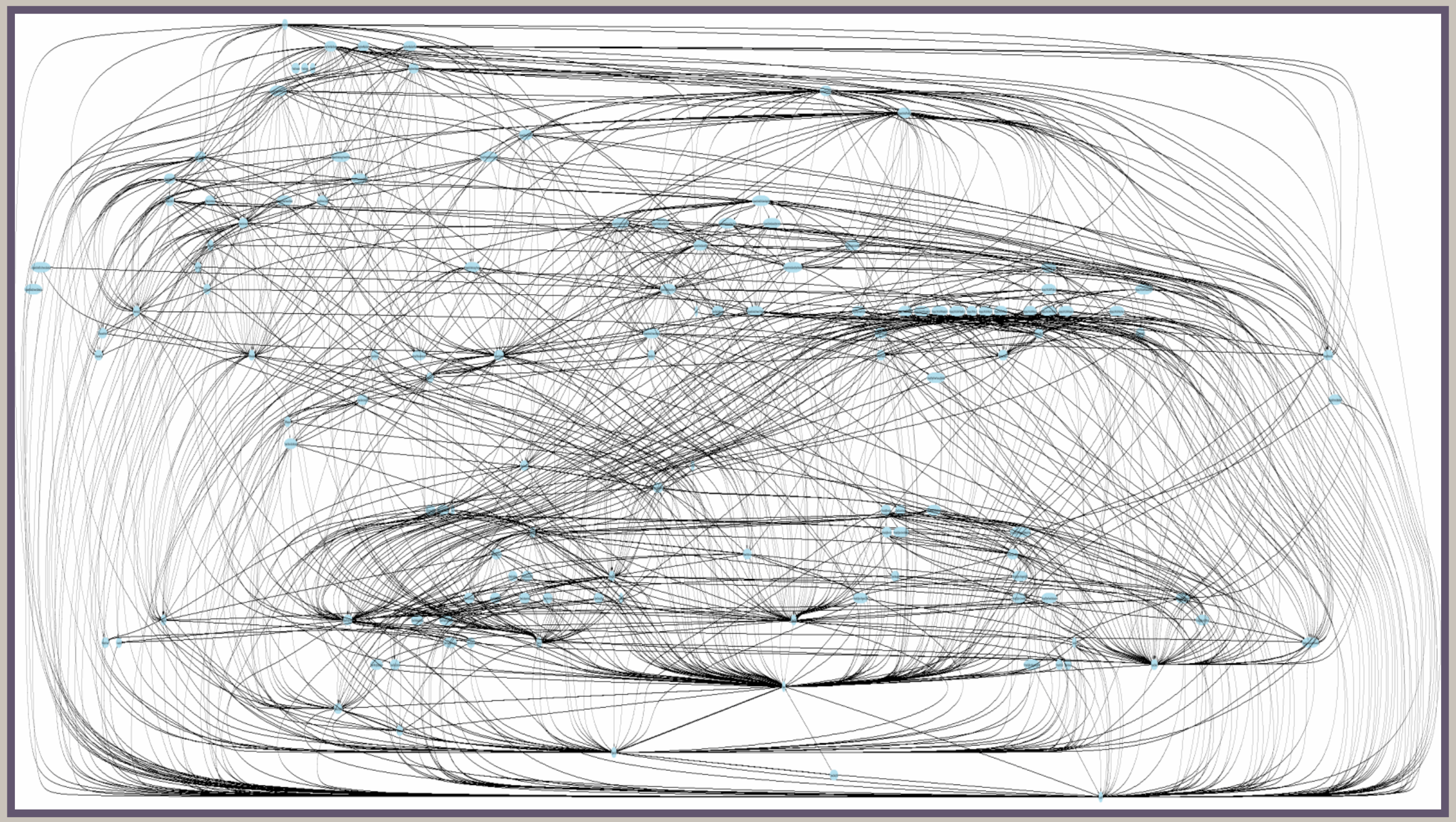


Modular Software

Benefits from a modular design:

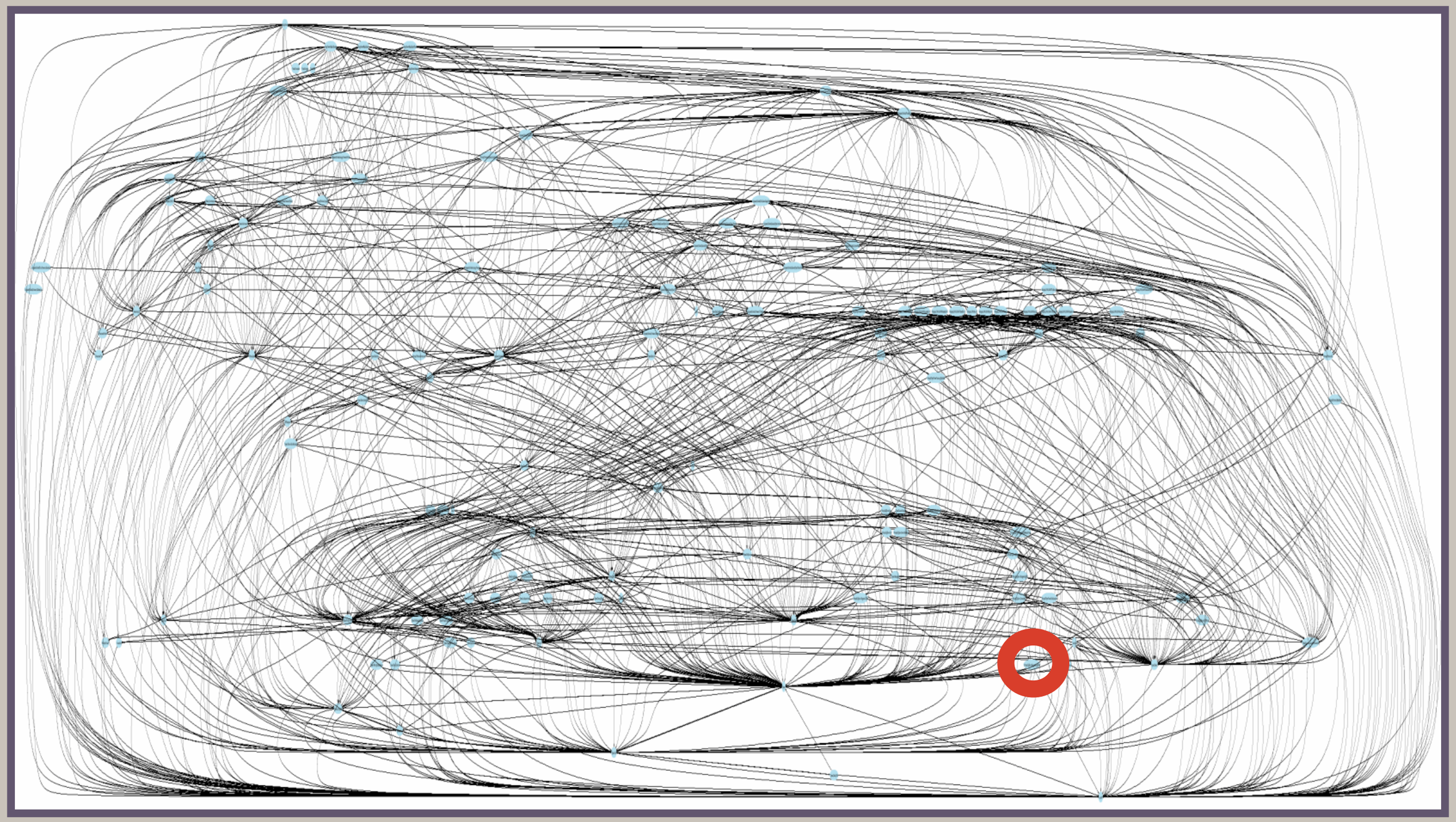
1. “separate groups would work on each module with little need for communication”
2. “it should be possible to make drastic changes to one module without a need to change others”
3. “it should be possible to study the system one module at a time. The whole system can therefore be better designed because it is better understood”(4)

Modular Software



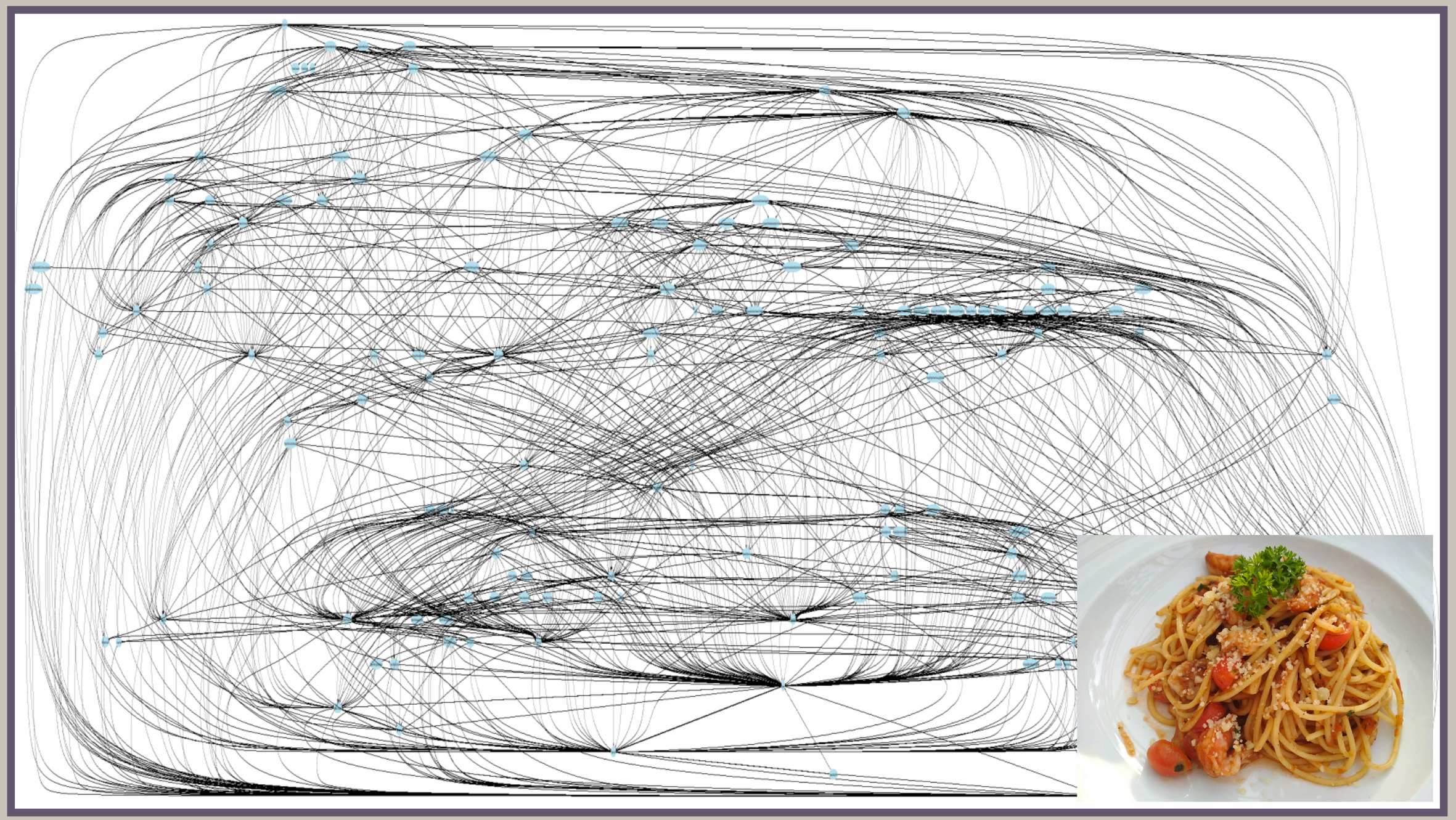
Alex Papadimoulis, THE ENTERPRISE DEPENDENCY
https://thedailywtf.com/articles/The-Enterprise-Dependency

Modular Software



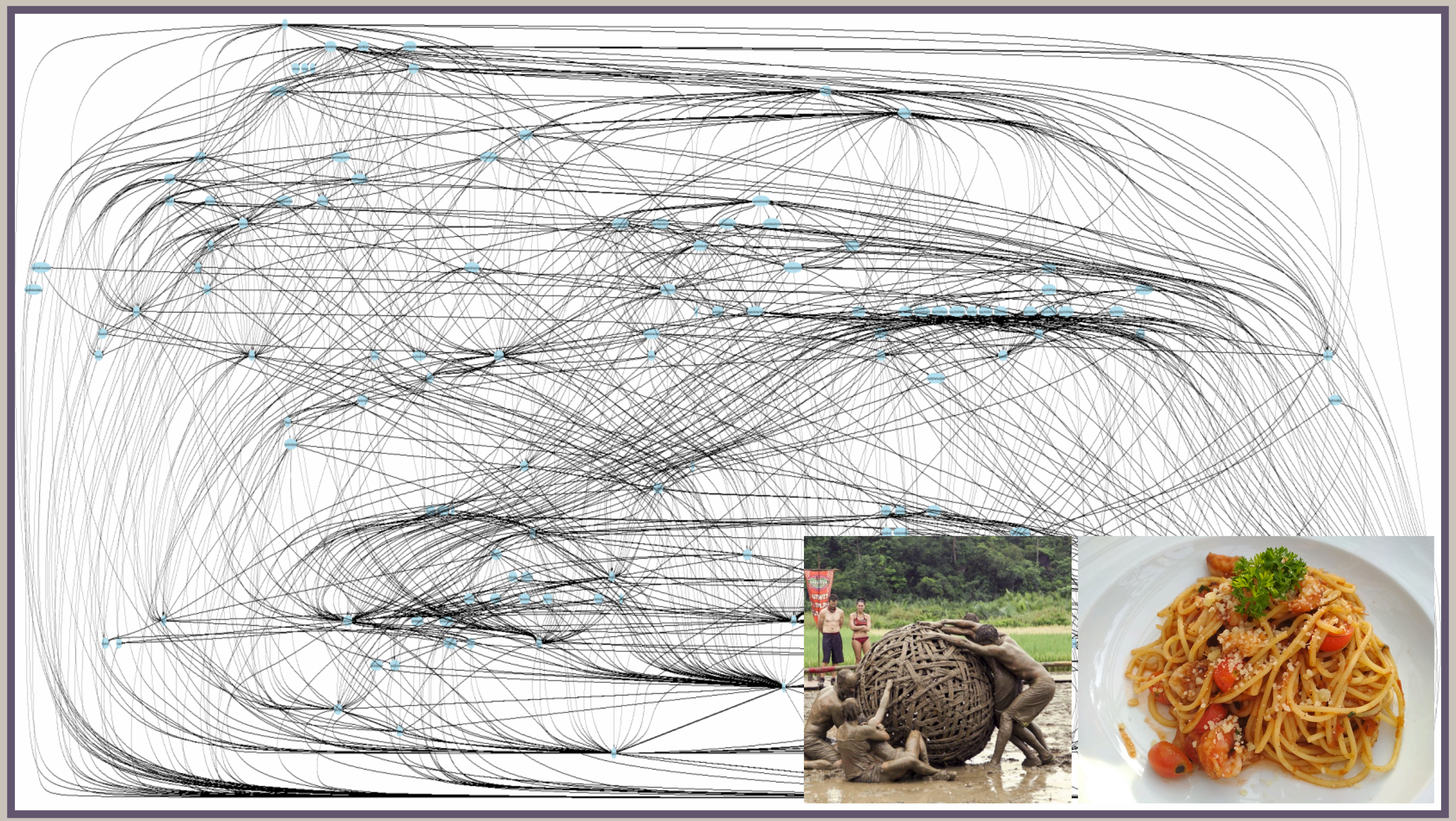
Alex Papadimoulis, THE ENTERPRISE DEPENDENCY
https://thedailywtf.com/articles/The-Enterprise-Dependency

Modular Software



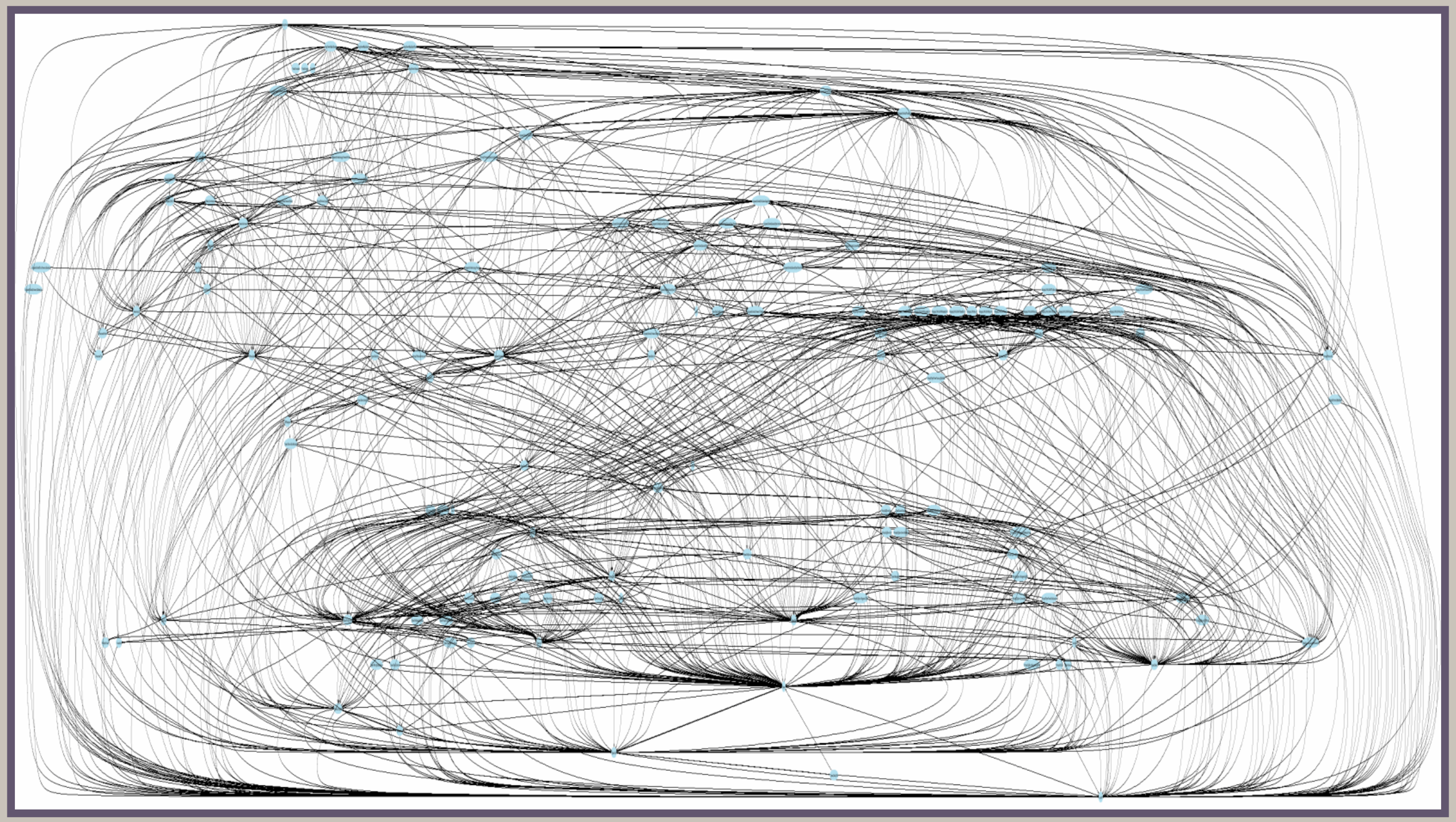
Alex Papadimoulis, THE ENTERPRISE DEPENDENCY
https://thedailywtf.com/articles/The-Enterprise-Dependency

Modular Software



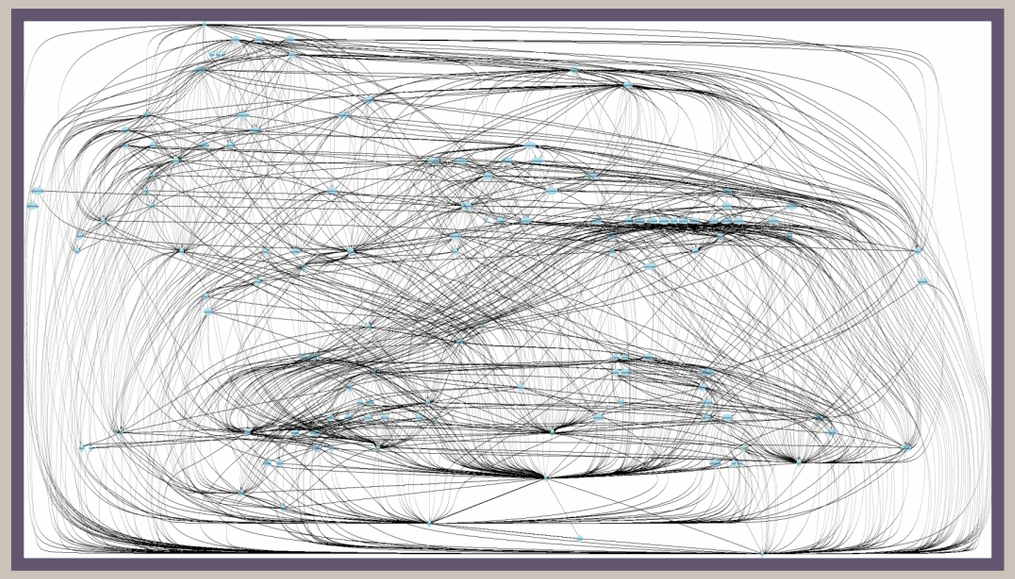
Alex Papadimoulis, THE ENTERPRISE DEPENDENCY
https://thedailywtf.com/articles/The-Enterprise-Dependency

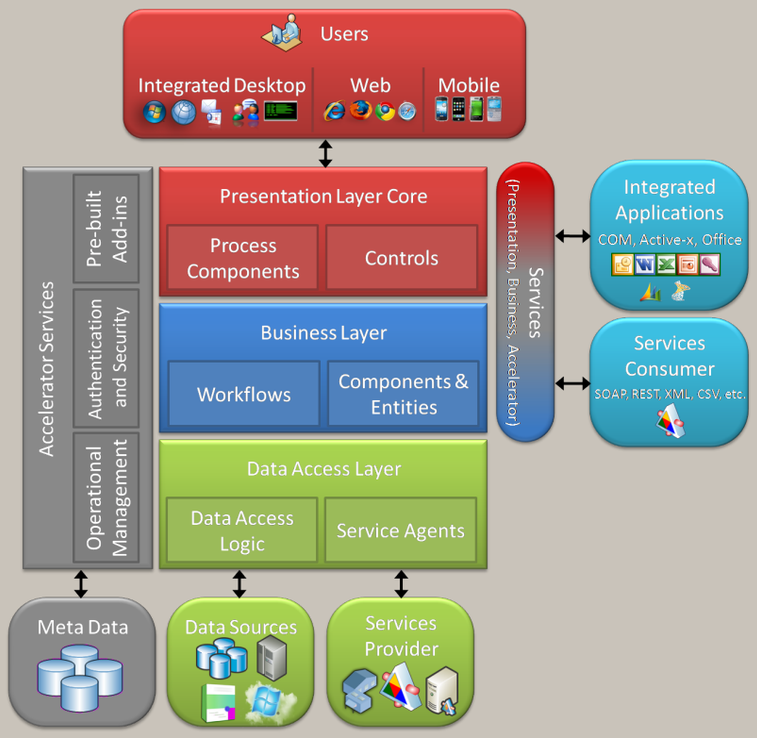
Modular Software



Alex Papadimoulis, THE ENTERPRISE DEPENDENCY
https://thedailywtf.com/articles/The-Enterprise-Dependency

Modular Software





Modular Software

Language	Design	Infrastructure
types	Single Responsibility Principle	Package Managers (OSGi, Maven, NPM)
private / protected	Information Hiding	Plug-in frameworks (e.g. Eclipse)
import / export	Low Coupling	(Micro) Services
interfaces	High Cohesion	Container Frameworks (e.g. Docker)
Polymorphism	Separation of Concerns	
Modules	…	…
…		

Modular Software

Language	Design	Infrastructure
types	Single Responsibility Principle	Package Managers (OSGi, Maven, NPM)
private / protected	Information Hiding	Plug-in frameworks (e.g. Eclipse)
import / export	Low Coupling	(Micro) Services
interfaces	High Cohesion	Container Frameworks (e.g. Docker)
Polymorphism	Separation of Concerns	
Modules	…	…
…		

Modular Software

The KWIC (Key Word in Context) index system accepts an ordered set of lines; each line is an ordered set of words, and each word is an ordered set of characters. Any line may be “circularly shifted” by repeatedly removing the first word and appending it at the end of the line (possibly omitting stop words). The KWIC index system outputs a listing of all circular shifts of all lines in alphabetical order. (4)

Gone With the Wind
Winds of War
War and Peace

Gone Wind
Peace War
War Peace
War Winds
Wind Gone
Winds War

Modular Software

Design 1

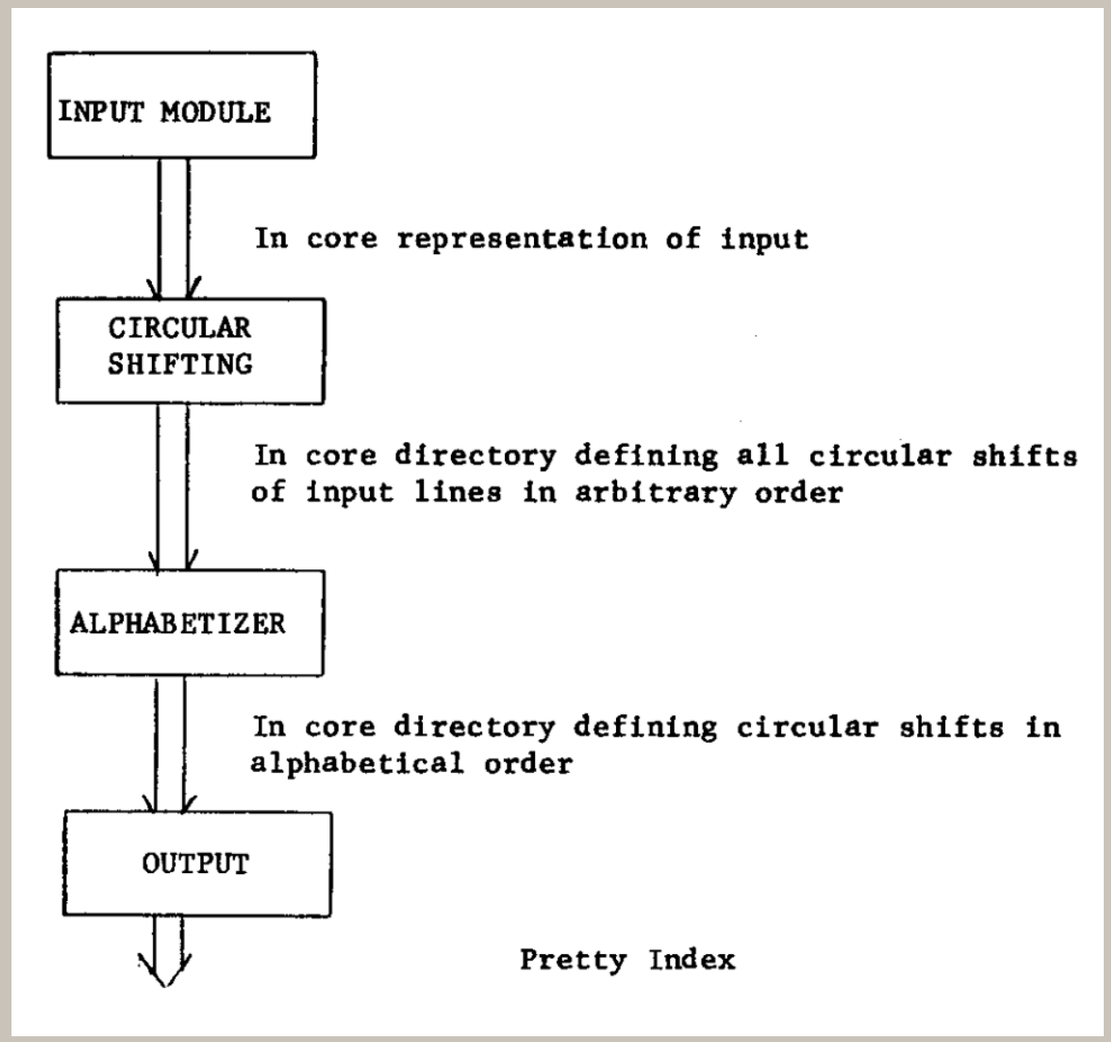
Module 1: Input

Module 2: Circular Shift

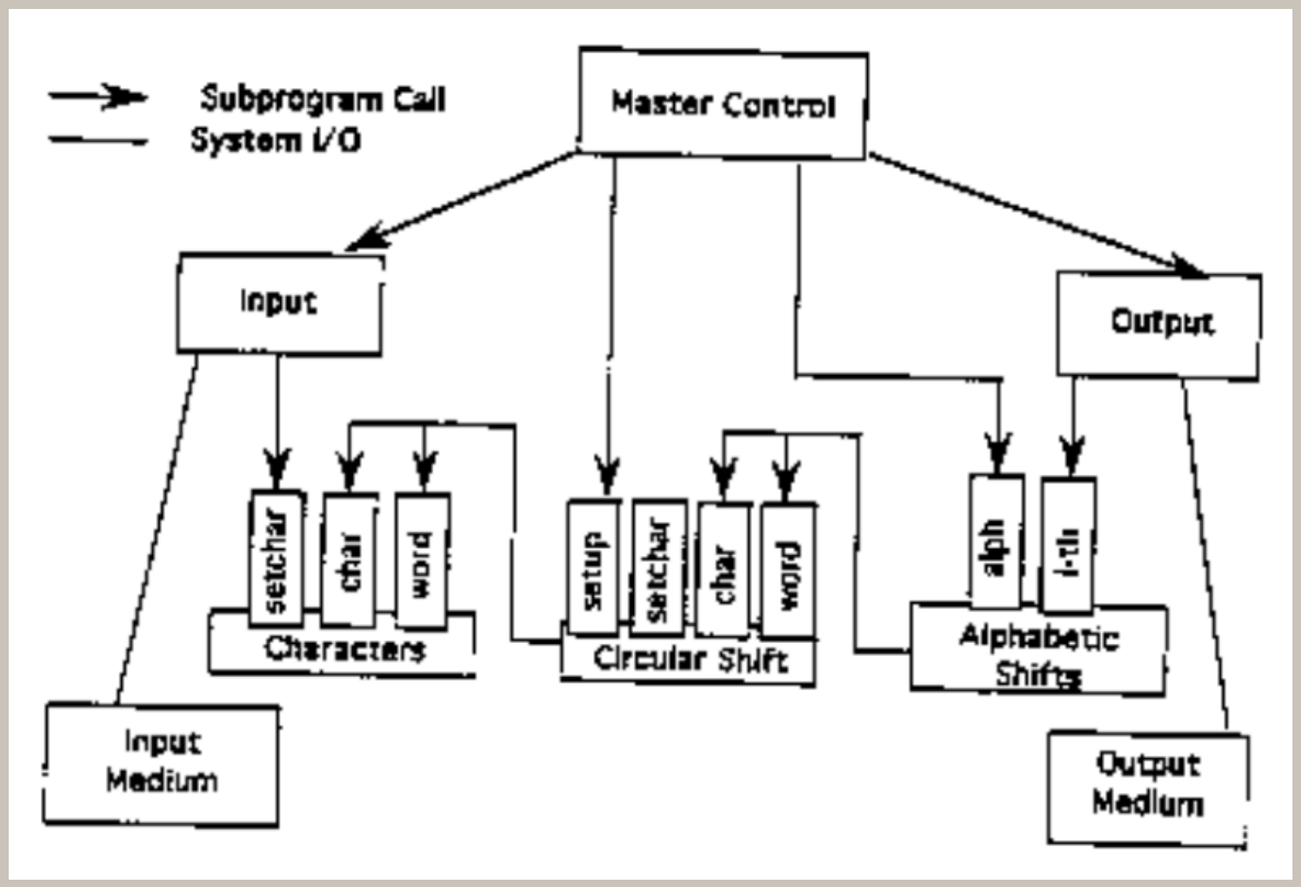
Module 3: Alphabetizing

Module 4: Output

Module 5: Master Control



Modular Software



Design 2

Module 1: Line Storage

Module 2: Input

Module 3: Circular Shifter

Module 4: Alphabetizer

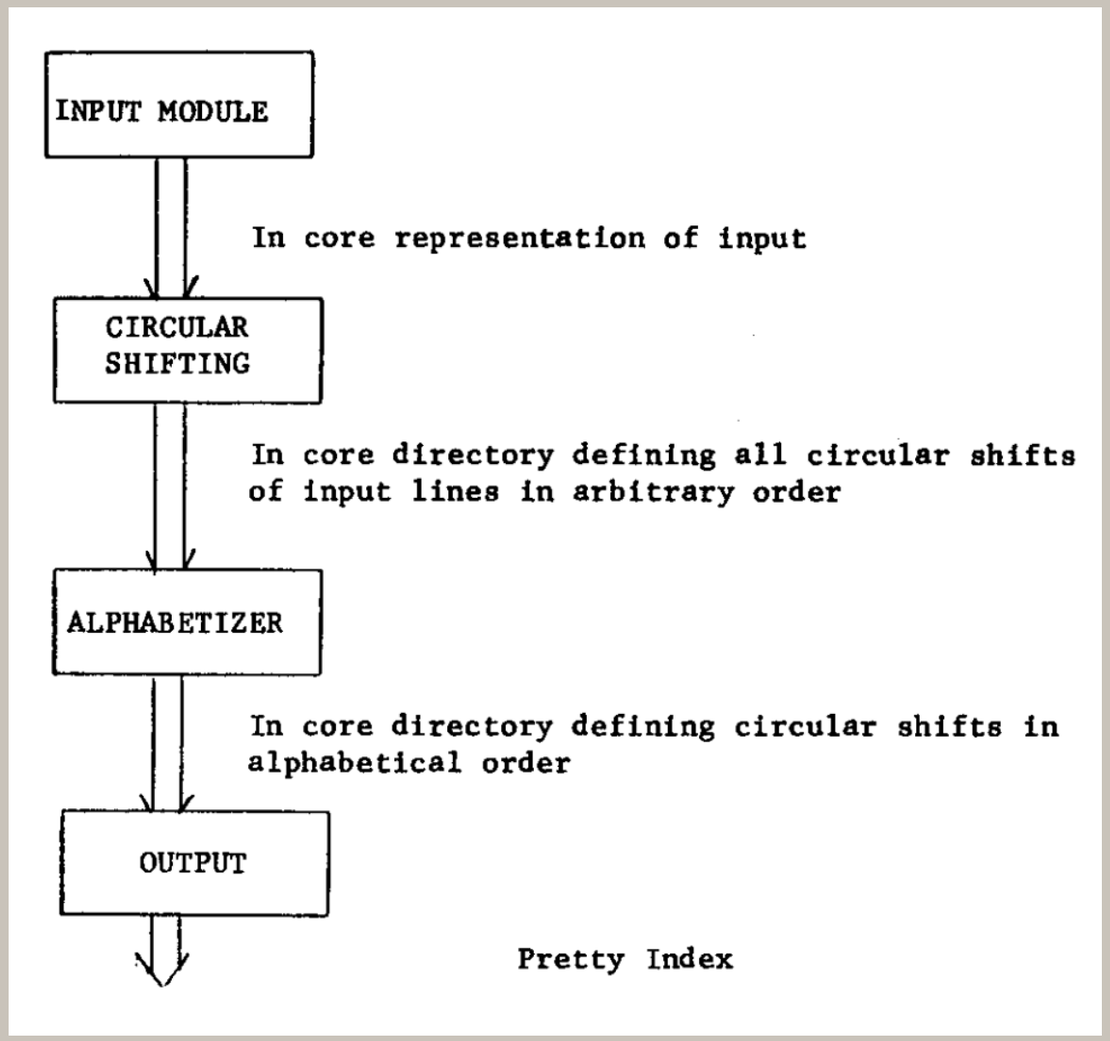
Module 5: Output

Module 6: Master Control

http://www.star.cc.gatech.edu/documents/SpencerRugabear/inClass.pdf

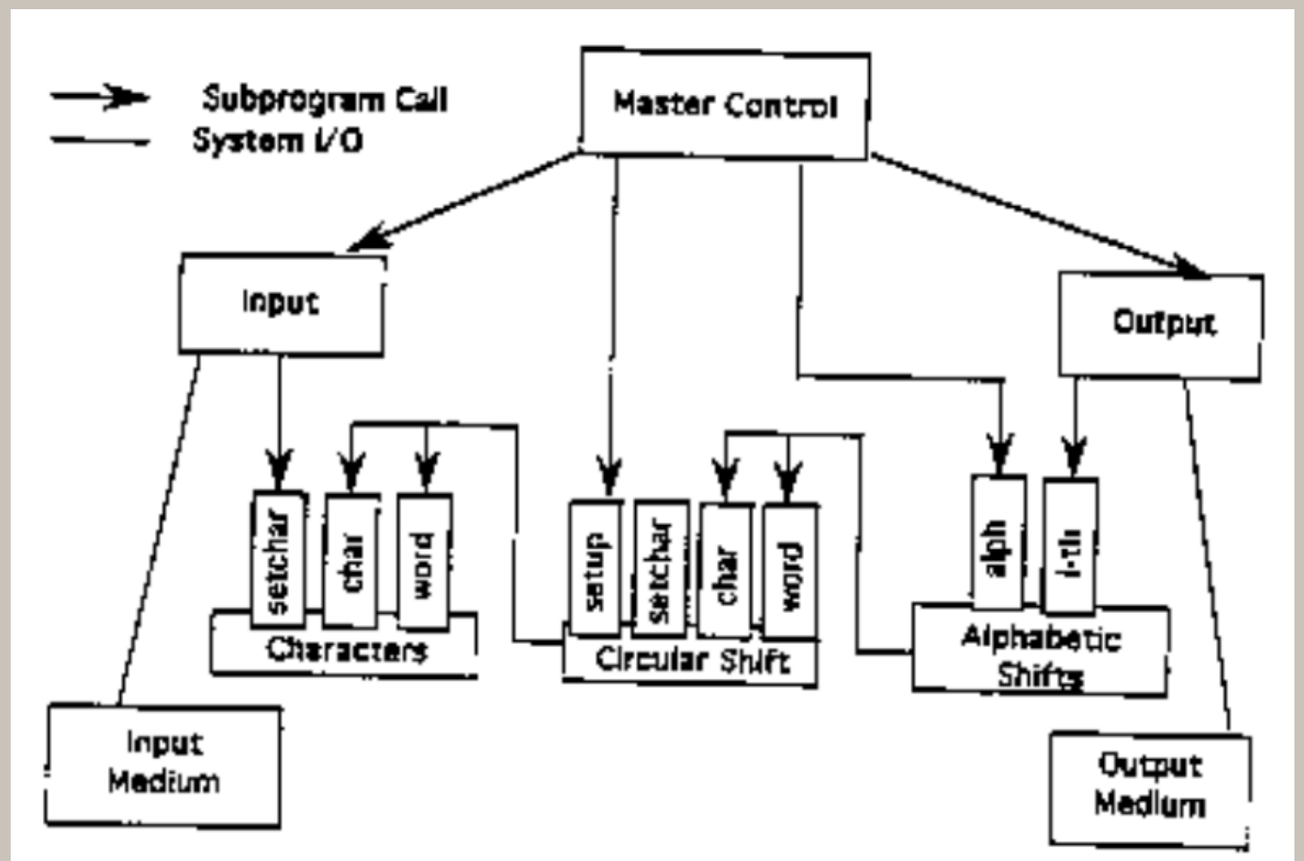
Modular Software

Design 1



Flowchart

Design 2



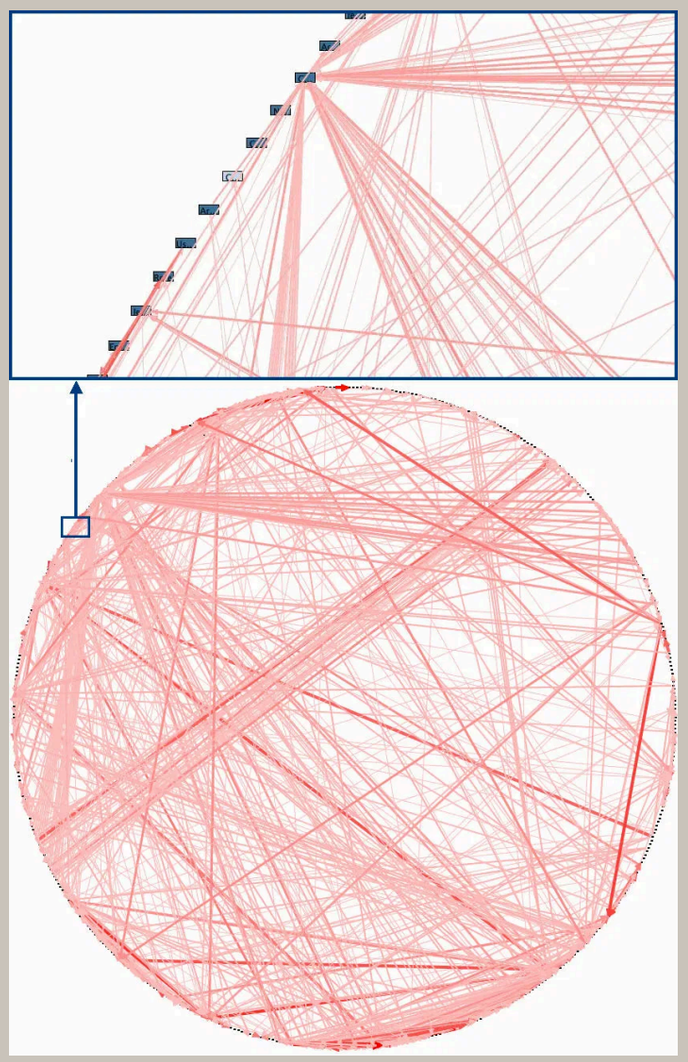
Information Hiding

Modular Software

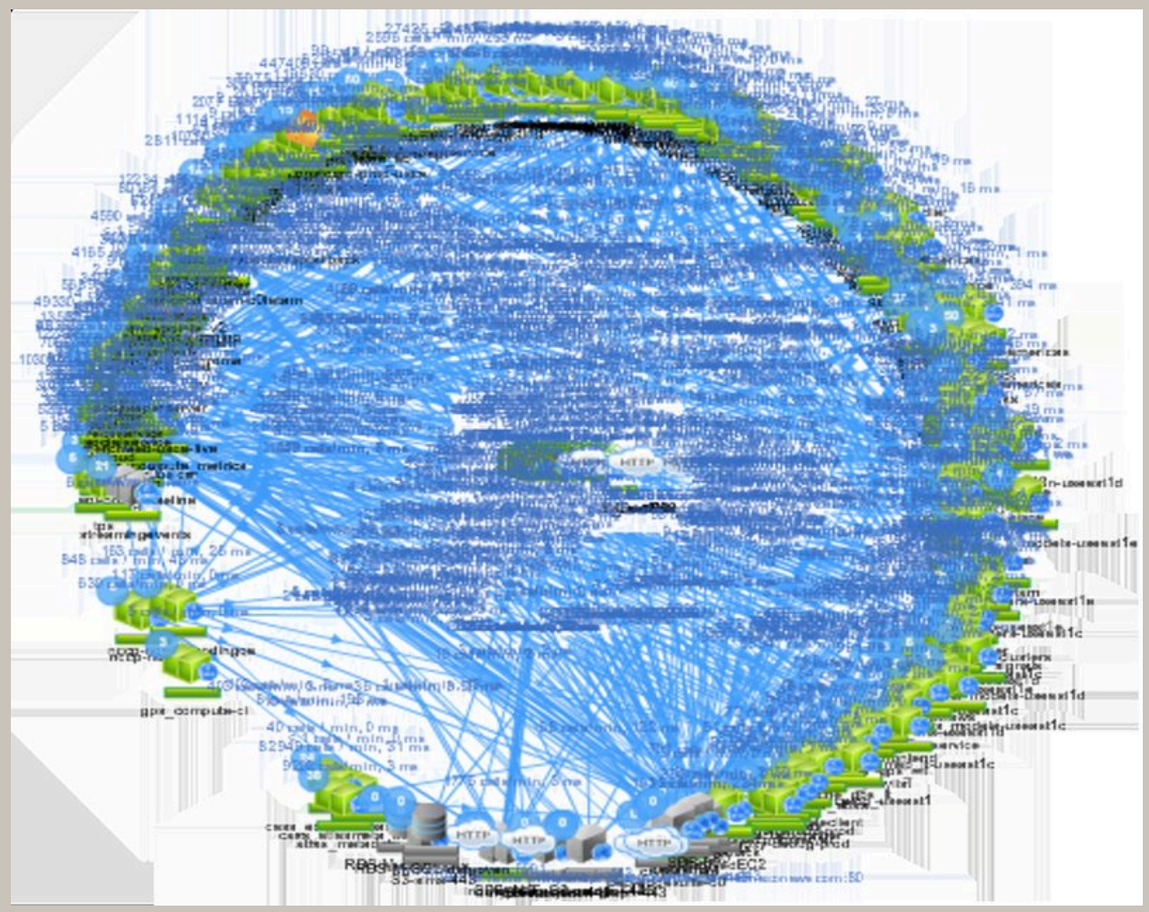
Designing a modular software system (5)

- Identification of the items that are likely to change. These items are termed "secrets."
- Location of the specialized components in separate modules.
- Designing intermodule interfaces that are insensitive to the anticipated changes. The changeable aspects are termed the “secrets" of the modules.

Modular Software



Ball of Yarn Design

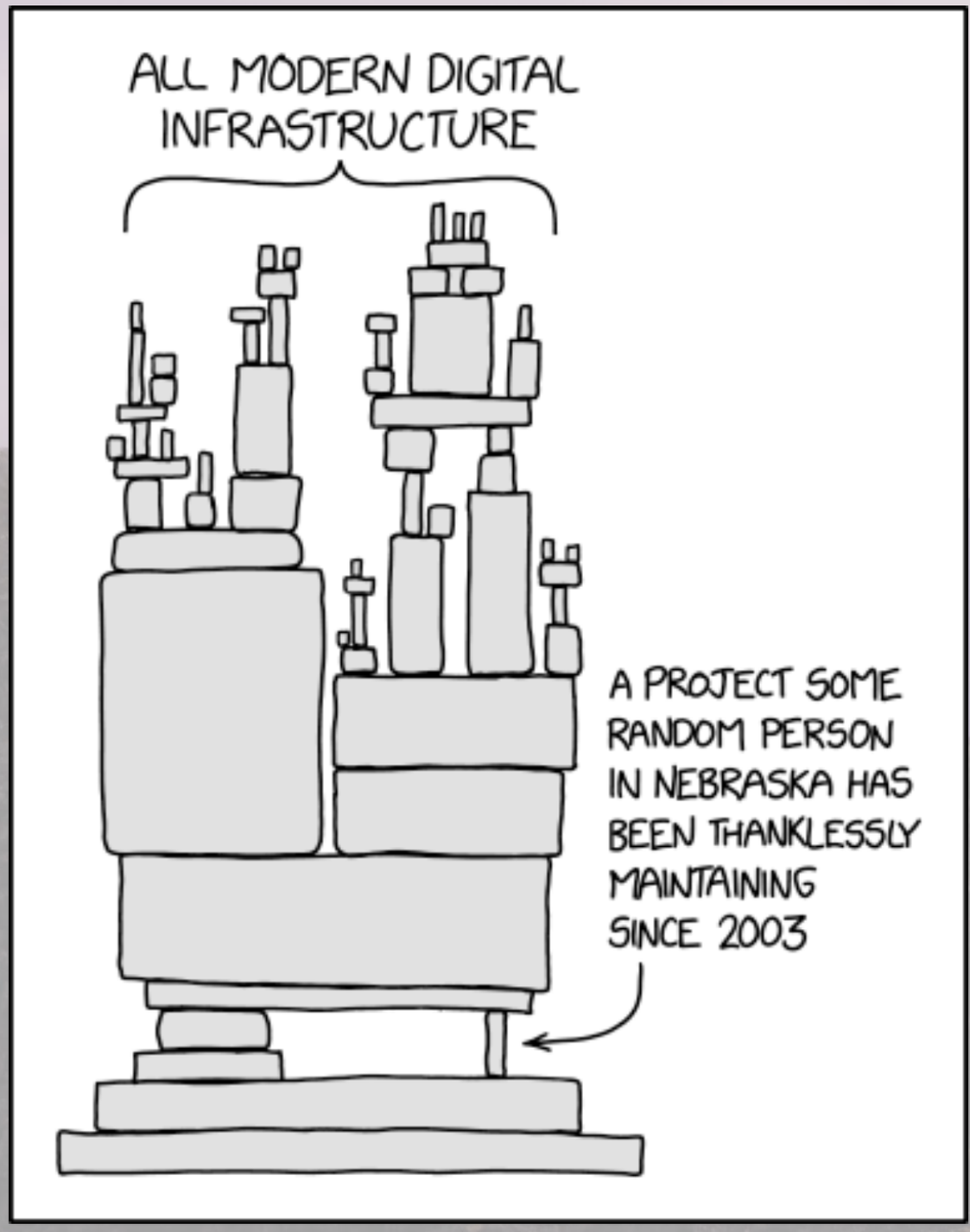


Netflix's Death Star

http://thedailywtf.com/articles/Enterprise-Dependency-Big-Ball-of-Yarn
http://codeobsession.blogspot.com/2018/02/architecture-at-different-levels-of.html

Modularity in Software Design

- Modularity 101
- Modular Software
- Module-Related Problems



https://xkcd.com/2347/

Module-Shaped Problems

- JS package manager
  (client + repository)
- Anyone can publish
- 2019: 1’000’000 packages
- 1 week: 10.9 billion
  downloaded packages
- Estimated users:
  11 million developers

Module-Shaped Problems

left-pad (March, 2016)

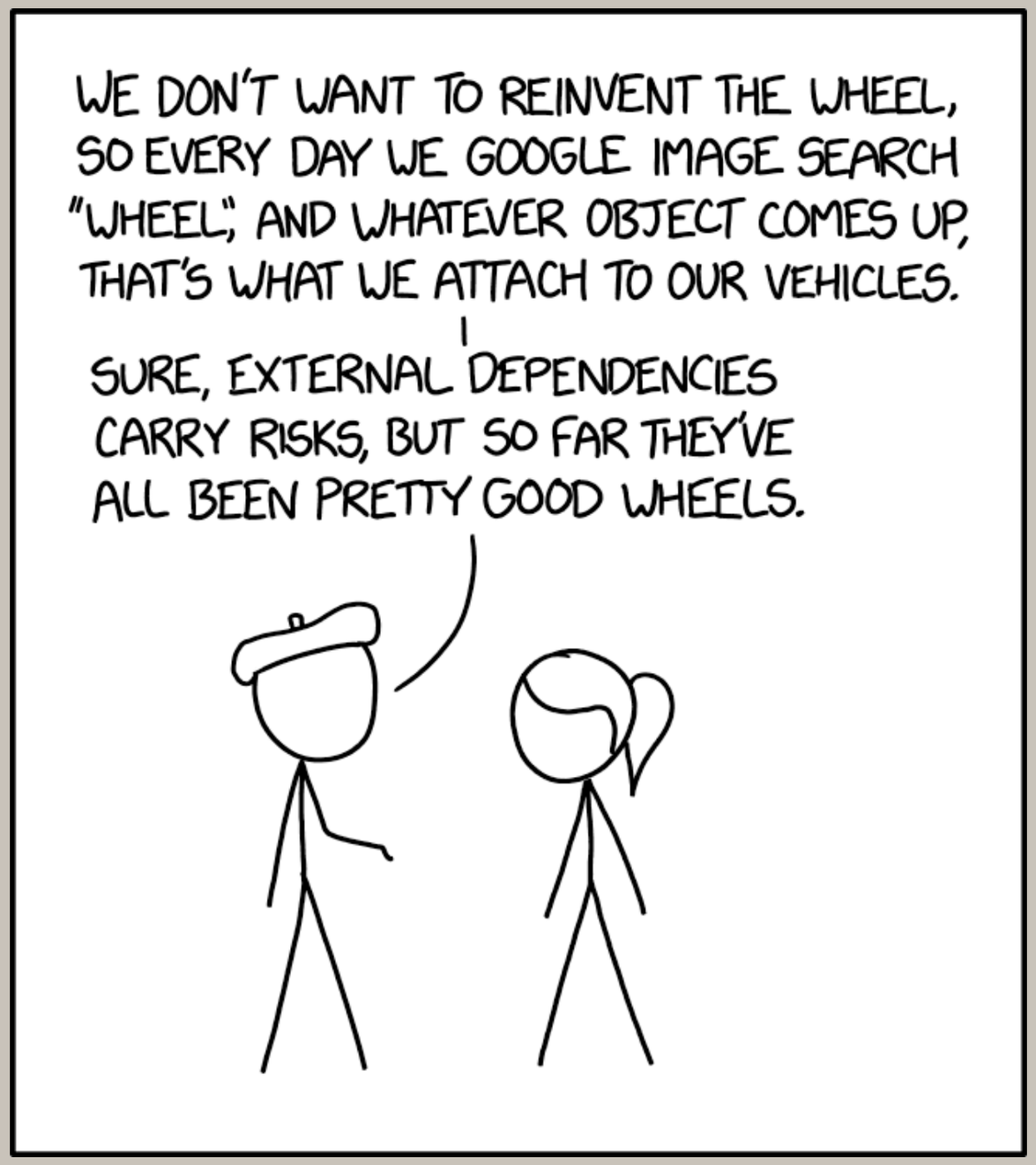
leftpad.js | package.json

```
1  module.exports = leftpad;
2  function leftpad (str, len, ch) {
3    str = String(str);
4    var i = -1;
5    if (!ch && ch !== 0) ch = ' ';
6    len = len - str.length;
7    while (++i < len) {
8      str = ch + str;
9    }
10   return str;
11 }
12
13
14
15
16
17
18
19
20
```

- 1 week:
  575 000 downloads
- March 22 2016

```
npm ERR! 404 'left-pad' is not in the npm registry.
```

Module-Shaped Problems



https://xkcd.com/2140/

References

(1) Baldwin, Carliss Young, Kim B. Clark, and Kim B. Clark. Design rules: The power of modularity. Vol. 1. MIT press, 2000.

(2) Alex Papadimoulis, THE ENTERPRISE DEPENDENCY https://thedailywtf.com/articles/The-Enterprise-Dependency

(3) Baldwin, Carliss Y., and Kim B. Clark. "Modularity in the design of complex engineering systems." Complex engineered systems. Springer, Berlin, Heidelberg, 2006. 175-205.

(4) Parnas, David L. "On the criteria to be used in decomposing systems into modules." Pioneers and Their Contributions to Software Engineering. Springer, Berlin, Heidelberg, 1972. 479-498.

(5) David L. Parnas. 1978. Designing software for ease of extension and contraction. In Proceedings of the 3rd international conference on Software engineering (ICSE '78). IEEE Press, 264–277.

(6) Sullivan, Kevin J., et al. "The structure and value of modularity in software design." ACM SIGSOFT Software Engineering Notes 26.5 (2001): 99-108.