

COHESION AND COUPLING

Hans van Vliet, *Software Engineering, Principles and Practice*, 3rd edition,
John Wiley & Sons, 2008. (Section 12.1)

Tuesday's Quote

2

- Exam 1 is on Tuesday 02/20/2017
Dr. Salamah

Outline

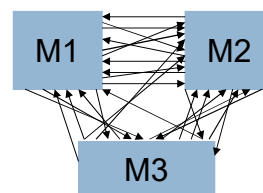
3

- Cohesion
- Coupling

Characteristics of Good Design

4

- Exception identification and handling
- Fault prevention and fault tolerance
- Design for change
 - Component independence
 - High cohesion
 - Low coupling



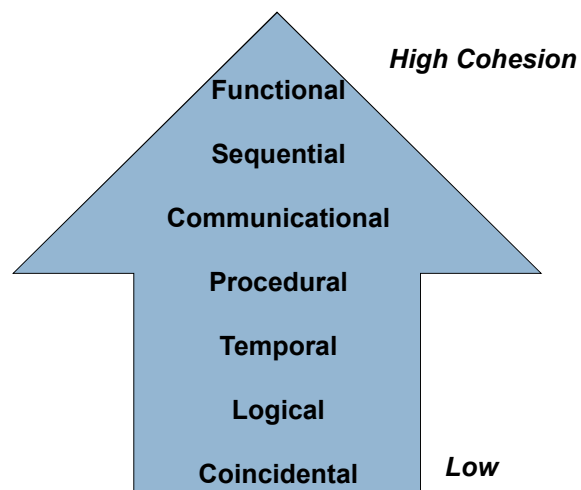
Cohesion

5

- **Definition**
 - ▣ The degree to which all elements of a component are directed towards a single task.
 - ▣ The degree to which all elements directed towards a task are contained in a single component.
 - ▣ The degree to which all responsibilities of a single class are related.
- Internal glue with which component is constructed
- All elements of component are directed toward and essential for performing the same task.

Type of Cohesion

6



Coincidental Cohesion

7

Functional
Sequential
Communicational
Procedural
Temporal
Logical
Coincidental

- Def: Parts of the component are unrelated (unrelated functions, processes, or data)
- Parts of the component are only related by their location in source code.
 - As a result Elements needed to achieve certain functionality are scattered throughout the system.
- Accidental
- Worst form

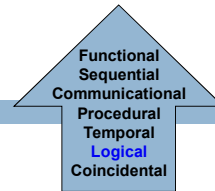
Example

8

1. Print next line
2. Reverse string of characters in second argument
3. Add 7 to 5th argument
4. Convert 4th argument to float

Logical Cohesion

9



- Def: Elements of component are related logically but not functionally.
- Several logically related elements are in the same component and one of the elements is selected by the client component.

Example

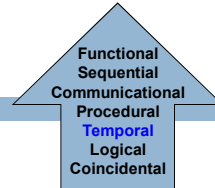
10

- A component reads inputs from tape, disk, and network.
- All the implementation code for these functions are in the same component.
- Operations are related, but the functions are significantly different.

Improvement?

Temporal Cohesion

11

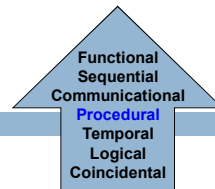


- Def: Elements are related by timing involved
- Elements are grouped by when they are processed.
- Example: An exception handler that
 - Closes all open files
 - Creates an error log
 - Notifies user

Many different activities occur at same period of time

Procedural Cohesion

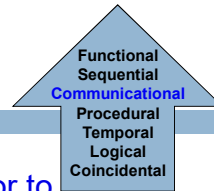
12



- Def: Elements of a component are related only to ensure a particular order of execution.
- Actions are still weakly connected and unlikely to be reusable.
- Example:
 - ...
 - Write output record
 - Read new input record
 - Pad input with spaces
 - Return new record
 - ...

Communicational Cohesion

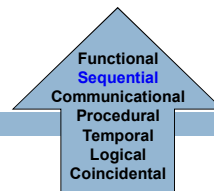
13



- Def: Functions performed on the same data or to produce the same data.
- Examples:
 - Update record in database and send it to the printer
 - Update a record on a database
 - Print the record
 - Fetch unrelated data at the same time.
 - To minimize disk access

Sequential Cohesion

14



- Def: The output of one part is the input to another.
- Data flows between parts (different from procedural)
 - How?
- Occurs naturally in functional programming languages
- Good situation

Functional Cohesion

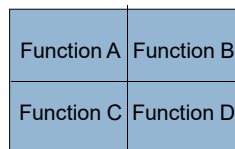
15

Functional
Sequential
Communicational
Procedural
Temporal
Logical
Coincidental

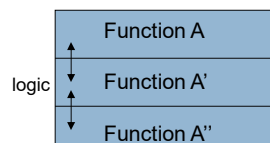
- Def: Every essential element to a single computation is contained in the component.
- Every element in the component is essential to the computation.
- Ideal situation
- What is a functionally cohesive component?
 - ▢ One that not only performs the task for which it was designed but
 - ▢ it performs only that function and nothing else.

Examples of Cohesion

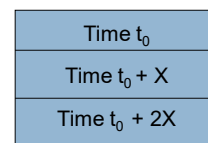
16



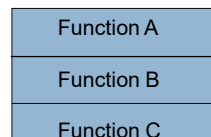
Coincidental
Parts unrelated



Logical
Similar functions



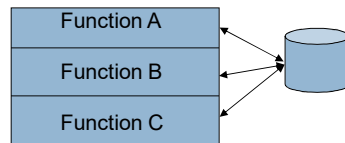
Temporal
Related by time



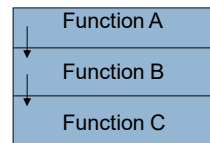
Procedural
Related by order of functions

Examples of Cohesion (Cont.)

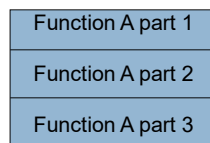
17



Communicational
Access same data



Sequential
Output of one is input to another



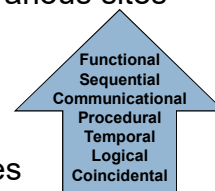
Functional
Sequential with complete, related functions

Quiz: Cohesion for Each Module? (10 mins)

18

- On a sheet of paper
 - Write your name
 - For each of the following modules, specify the type of cohesion

1. Compute average daily temperatures at various sites
2. Initialize sums and open files
3. Create new temperature record
4. Store temperature record
5. Close files and print average temperatures
6. Read in site, time, and temperature
7. Store record for specific site
8. Edit site, time, or temperature field



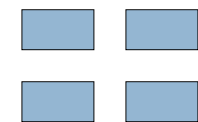
Outline

- ✓ Cohesion
- Coupling

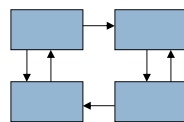
Coupling

20

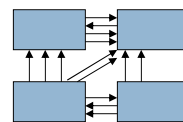
- The degree of dependence such as the amount of interactions among components



No dependencies



Loosely coupled
some dependencies



Highly coupled
many dependencies

Coupling

21

- The degree of dependence such as the amount of interactions among components
- How can you tell if two components are coupled?
- (In pairs, 2 minutes)

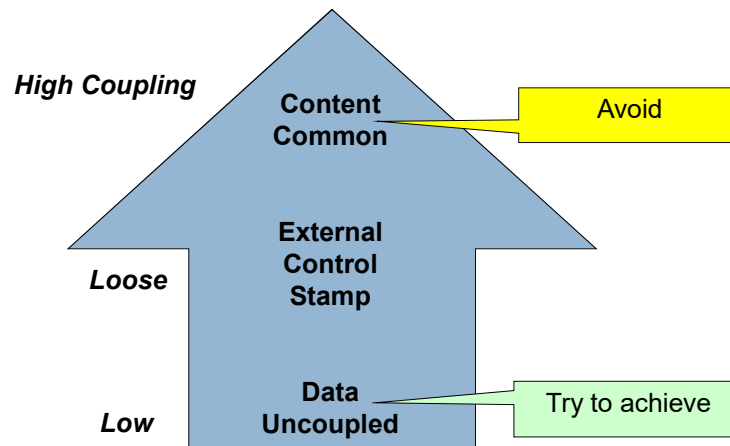
Indications of Coupling

22

- ??

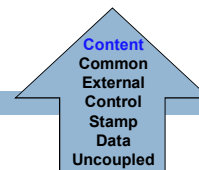
Type of Coupling

23



Content Coupling

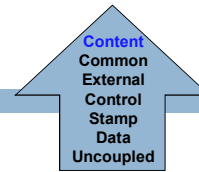
24



- Def: One component modifies another.
 - Component directly modifies another's data

Content Coupling

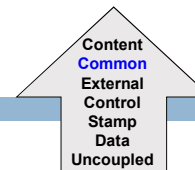
25



- Example:
 - ▣ Object A handles lookup for customer.
 - ▣ If the customer object C does not exist, A takes it upon itself to Add C by directly modifying the contents of the data structure containing customer data.
- Why is this even a problem?
- How can we improve this situation?

Common Coupling

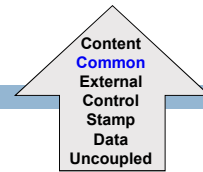
26



- Def: More than one component share data such as global data structures
- Usually a poor design choice because
 - ▣ Lack clear responsibility for the data
 - ▣ Reduces readability
 - ▣ Makes it difficult to determine all the components that affect a data element (reduces maintainability)
 - ▣ Reduces reusability of components
 - ▣ Reduces ability to control data accesses

Common Coupling

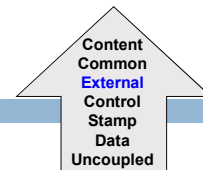
27



- Example
 - Process control component maintains current data about state of operation.
 - Component gets data from multiple sources.
 - Supplies data to multiple sinks.
 - Each source process writes directly to global data store.
 - Each sink process reads directly from global data store.
- Why is this a problem?
- How can we Improve this situation?

External Coupling

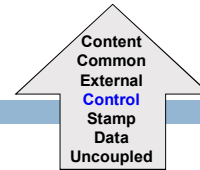
28



- Def: Two components share something externally imposed, e.g.,
 - External file
 - Device interface
 - Protocol
 - Data format
- Improvement?

Control Coupling

29

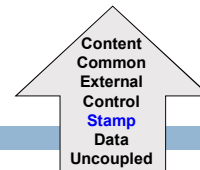


- Def: Component passes control parameters to coupled components.
- May be either good or bad, depending on situation.
 - ▢ Bad, if parameters indicate completely different behavior
 - ▢ Good, if parameters allow factoring and reuse of functionality
- Good example: sort that takes a comparison function as an argument.
 - ▢ The sort function is clearly defined: return a list in sorted order, where type of sorting is determined by a parameter.

29

Stamp Coupling

30

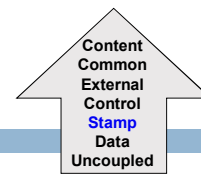


- Def: Component passes a data structure to another component that does not have access to the entire structure.
- Requires second component to know how to manipulate the data structure (e.g., needs to know about implementation).
- The second has access to more information than it needs.
- May be necessary due to efficiency factors: this is a choice made by insightful designer, not lazy programmer.

30

Stamp Coupling

31

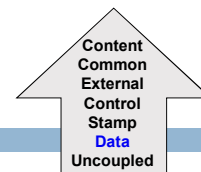


- Example:
 - ▣ Customer Billing System
 - ▣ The print routine of the customer billing accepts customer data structure as an argument,
 - ▣ parses it, and prints the name, address, and billing information.
- Why is this even a problem?
- How can we Improve this situation?

31

Data Coupling

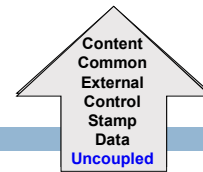
32



- Def: Component passes data (not data structures) to another component.
- Every argument is simple argument or data structure in which all elements are used
- Good, if it can be achieved.
- Example: Customer billing system
 - ▣ The print routine takes the customer name, address, and billing information as arguments.

Uncoupled

33



- Completely uncoupled components are not systems.
- Systems are made of interacting components.

Consequences of Coupling

34

- Why does coupling matter? What are the costs and benefits of coupling?
- (pairs, 3 minutes)

Consequences of Coupling

35

- High coupling
 - ▣ Components are difficult to understand in isolation
 - ▣ Changes in component ripple to others
 - ▣ Components are difficult to reuse
 - Need to include all coupled components
 - Difficult to understand
- Low coupling
 - ▣ May incur performance cost
 - ▣ Generally faster to build systems with low coupling

In Class

36

In your groups:

- P1: What is the effect of cohesion on maintenance?
- P2: What is the effect of coupling on maintenance?

36