Software Metrices

Complexity metrices for Module relationship

- **The relationship between program modules**
  - affects program complexity
- **Complexity metrices**
  - Degree of Coupling
  - Cohesion
- **Complexity metrices for module size**
  - Software Science Metrices (Halstead)
  - Cyclomatic Number (McCabe)
  - Control Variable Complexity (McClures)

Coupling

- **The degree of interaction between modules (Degree of coupling, D_C)**

  D_C = (amount of data exchanged between modules) / (amount of local processing)

- **Three factors affecting the degree of coupling**
  - The amount of data passed between modules
  - The amount of control data
  - The amount of global data elements shared by modules
- **Two categories**
  - Loosely (or Weakly) coupled
  - Tightly (Strongly) coupled

- **Five levels of coupling possible between modules**
  - Data Coupling :
    - variable or array (parameter passing)
  - Stamp Coupling :
    - composite item (record, struct) but, some of them are only used
  - Control Coupling :
    - flag set and test in other module
  - Common Coupling :
    - shared data (global data)
  - Content Coupling :
    - call or branch to other module
    - ex) one module modifies local data or instr. in another module

Coupling metrics : [Dhama, 1995]

- Component-level design metric으로 다음 3가지로 분류할 수 있다
  - Data & control coupling
    - d_i, d_o, c_i, c_o
    - number of input/output data/control parameters
  - Global coupling
    - g_d, g_c : number of global variables used as data/control
  - Environmental coupling
    - w : number of modules called (fan-out)
    - r ; number of modules calling the module (fan-in)
- Module coupling indicator, $\mathbf{m_c}$ is defined as
  - $\mathbf{m_c = k/M}$
  - where k=1 and $\mathbf{M = d_i + a \times c_i + d_o + b \times c_o + g_d + c \times g_c + w + r}$ where a = b = c = 2
- **The higher the value of m_c, the lower the overall module coupling**
- 그러므로 degree of coupling $\mathbf{D_C = 1 - m_c}$ 가 된다

- Example 1
  - If a module has a single input, output data parameter, access no global d ata and is called by a single module
  - $m_c = 1 /(1 + 0 + 1 + 0 + 0 + 0 + 1 + 0) = 1/ 3 = 0.33$
- Example 2
  - 5 input and 5 output, and equal number of control parameters, access 10 global data, fan-in=3, fan-out=4
  - $m_c = 1/(5 + 2x5 + 5 + 2x5 + 10 + 0 + 3 + 4) = 0.02$

Cohesion

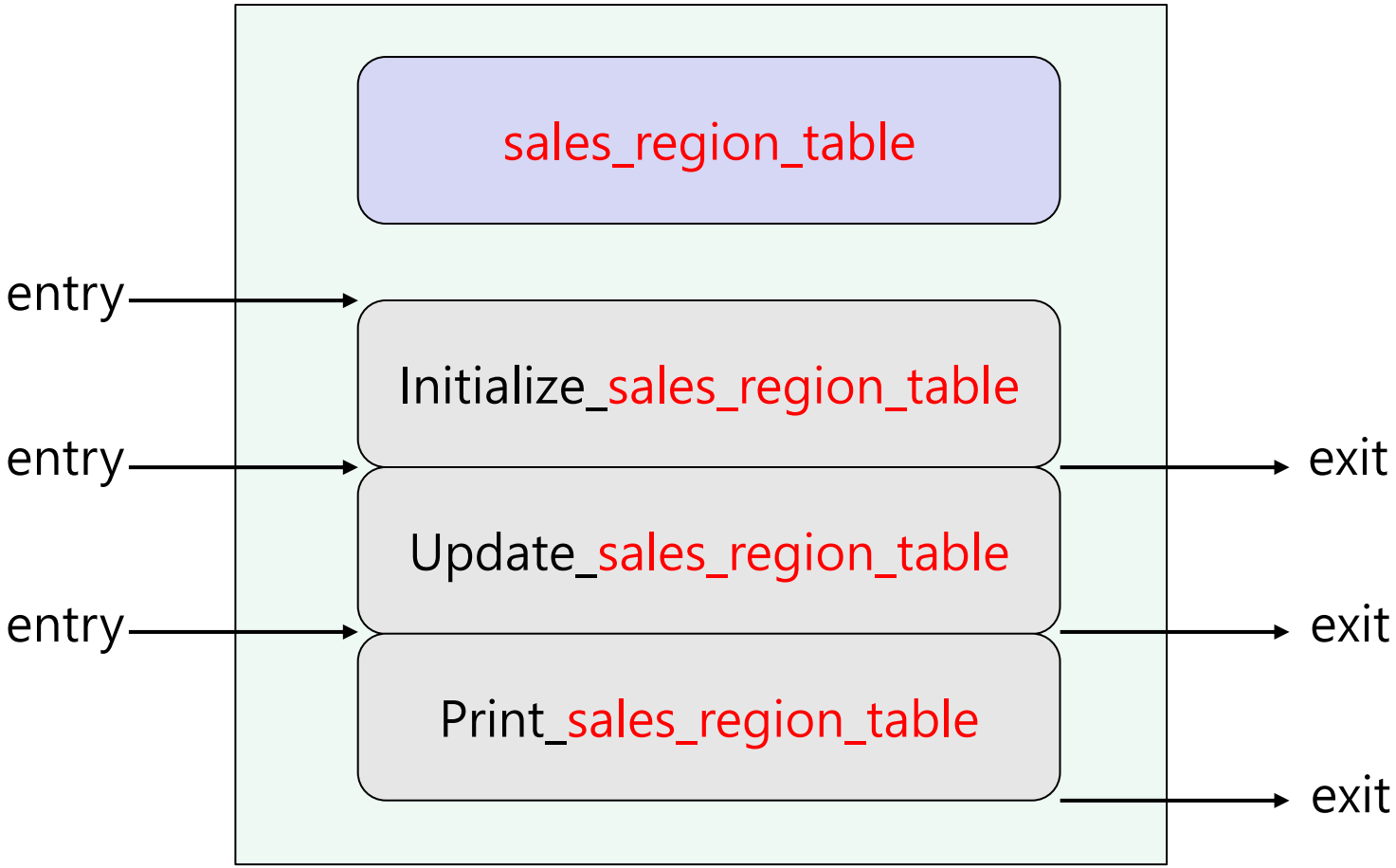
- The degree of dependency among elements within a module (natural expansion of the information hiding)
  - Measures how strongly the elements within a module are related (ideally, a cohesive module should do just one thing)
- 7 levels of Cohesion
  - Functional cohesion
  - Informational cohesion
  - Communicational cohesion
  - Procedural cohesion
  - Temporal cohesion
  - Logical cohesion
  - Coincidental cohesion

- **Functional cohesion**
  - When it performs exactly one action or achieves a single goal
  - can be easily reused in other contexts
  - easy maintenance by fault isolation
  - easy extension
- Examples
  - Compute_Square_Root(…)
  - Obtain_Random_Number

Informational Cohesion

- When it performs a number of actions, each **with its own entry point**, with **independent code for each action**, all performed on **the same data structure**
  - 그러므로 Multiple entry and exit points를 가지게 된다
  - Implementation of an abstract data type(혹은 클래스 객체)
- All advantages of using an ADT(혹은 객체) are gained when a module with informational cohesion is used
- In a module with information cohesion, each action is completely independent

일반적인 ADT 혹은 객체의 구성



Communicational Cohesion

- When it performs a series of actions related by the sequence of steps to be followed by the product, and in addition all the actions are performed **on the same input or output data**
- Actions are more closely connected
- Flow chart cohesion

[Examples]

- (1) Update record in DB
- Write it to the audit trail
- (2) Calculate new trajectory
- Send it to the printer

Problems

- procedural cohesion보다는 좋은 응집력을 보인다. 왜냐하면 동일한 입력 혹은 출력 데이터 상에서 동작하기 때문이다
- 그러나 모듈의 재사용이 어려운 것은 동일하다

Procedural Cohesion

- When the elements of a module are related and must be executed in a specific order
- The case of modularizing a part of flow chart
- [Examples]
  - Read_Part_Number from DB
  - Update_Repair_Record on Maintenance_File

Temporal Cohesion

- When it performs a series of actions related in time
- The actions in that module are weakly related to one another but more strongly related to actions in other modules

- [Examples]

  Open_File

  Create_File

  Print_File

  Initialize_Table

  Read_Record, . . .

- [Problem]
  - 해당 모듈의 수정은 다른 모듈의 변경을 초래한다
  - logical cohesion처럼 다른 product에서는 reusable하지 않다

Logical Cohesion

- When it performs a series of related actions, one of which is selected by the calling module
  - control parameter for action selection
- A certain class of logically related operations
  - ex) I/O routines, Editor, math. library

- [Example]
  - A module performing all input and output
  - A module performing editing of insertions, deletions, and modifications of master file records
- [Problem]
  - The interface is difficult to understand
  - The code for more than one action may be intertwined, leading to sever e maintenance problems (즉, output 장치가 하나 추가 되던지 혹은 변경 될 때, 그와 관련된 모든 routine들이 수정되어야 한다)

- **Coincidental Cohesion**
  - unrelated elements (일반적으로 프로그램을 임으로 분할할 때)
- [Example]
  - Print next line
  - Reverse the string of characters comprising the second paramenter
  - Add 7 to the fifth parameter
  - Convert the fourth parameter to floating point
- [Problems]
  - Degrade the maintenability of the product
  - Lack of reusability

Cohesion Example