Chapter 6

Implementation / Integration

I. Language classes (available language)

II. Choice of language

III. Style / Standards

IV. Integration strategies

V. Team organization

I. Language classes

i) FGL (~50’s)
   Assembly (machine)

ii) 2nd GL (late 50’s ~ 60’s)
    FORTRAN, COBOL, Algol, BASIC

iii) 3rd GL (late 60’s ~ 80’s)
    - general purpose language: PL/I, Pascal, C. Ada
    - OOL: G++, Smalltalk
    - Specialized: LISP, Prolog, Forth (Firmware)

iv) 4 GL
    SQL, Powerbuilder. Formal spec languages
    → good for prototyping

II. Choice of a PL

i) Demand by the customer

ii) Based on the application area
    AI → LISP, Smalltalk, Snobol
    Business → COBOL
    Military → Jovial, PL/I
    Scientific → FORTRAN, ALGOL, PL/I

iii) Cost-benefit
    - Decision by the management
    - Least expensive

iv) S. Engineering characteristics
    a. Ease of design to implementation translation
       Ex: “bit manipulation” – C, C++, Assembly
    b. Available SE tools
       -VB
       -compiler, debugger, GUI
       -language directed editor

v) Language features:
    a. Readability of source
       - naming of objects, types; identifiers
    b. Dynamic memory management
       - static mm. – cobol BASIC, Fortran
       - limited Dmm – Pascal, C. C++
       - Dynamic mm. – LISP
    c. Type checking
       - no type checking, Basic
- Strong type checking Ada
- Semi-type checking: Fortran, Pascal, C.

Fortran:

```
Mpoint = MP0int + 5;
MP created, and initialized to 0
Mpoint = 5
```

d. Exception Handling - patient monitor system
   - military system

When an exceptional condition occurs. (run-time)
Divide by 0 language: VB, Java, C++, ADA

III. Style and standards
1. Style: (personal choice)
2. Standard: (imposed by the co)

   - Style
   a. Use of consistent and meaningful names
      Meaningful: a - average
      Consistency:
      Frequency – aver-frequency

   b. Use one statement per line avoid unnecessary complexity.
      Unnecessary complexity n=5;
      Printf (“% d %d”, ++n, n+2)
      Depends on compiler (6, 7) or (6, 8)

   - Standard:
   a. A function should have < 50 lines. (IBM)
   b. Nesting <=3 more readable
   c. Not use of “goto”

Review by SQA team
/*Article*/
fundamental “theorem” of formatting with an example.

IV. Integration Strategies

AD:

```
Logic modules
S

A  B  C

D  E  F

H  I

J  K

L

M  N
```
1. Top-down integration
   test complete thread
   - vertical approach
   - horizontal approach
   - $\rightarrow$ S A B C D E F H I J K L M N
   Adv:
   - logic first
   - Fault isolation is better
   - Writing stubs is easy
   Disadv: work modules are not tested enough

2. Bottom-up integration
   - horizontal $\rightarrow$ L M N D E F H I J K A B C S
   - vertical: L D A S E F H B I M J L N K
   Adv: work modules are tested more
   Disadv: the logic is tested late

3. Sandwich integration
   Combination of 1 & 2
   - The logic modules are implemented Top-down
   - The work modules are implemented bottom-up

V. Team Organization:
   How to organize the team
1. Weinborg (Democratic Team)
2. Chief-programmer’s Team
3. Hybrid Model

1. Weinborg Team (Democratic team)
   - All the members have the same power & responsibility
     * Egoless Environment
   - No leader
   - Direct commu
   - Same power / respon $\rightarrow$ more interesting
     * You are likely to admit your mistakes. Better Discovery of fault.
   Disadv:
   - the commu. Structure is not clear
   - emotionally bound $\rightarrow$ if one person leaves, all may go.
     Good for small group, and research.
     (less than 10)

2. Chief programmer (IBM)
- Chief: - in charge of the project has power / responsibility  
  - in technical & management  
- Backup Programmer: - equal knowledge / power / responsibility as the chief.  
- Program Secretary: - Secretarial work

- Good for small project

Adv:  
- communication structure is clear  
- No communication needed to other programmers  
- Backup is available  
- Each person has clear function

Disadv:  
- more age  
- Hard to find chief programmer  
- To find a backup is worse

Variation:

Very large: >50 - 100