

Example of Defects in a software

Incorrect default parameters

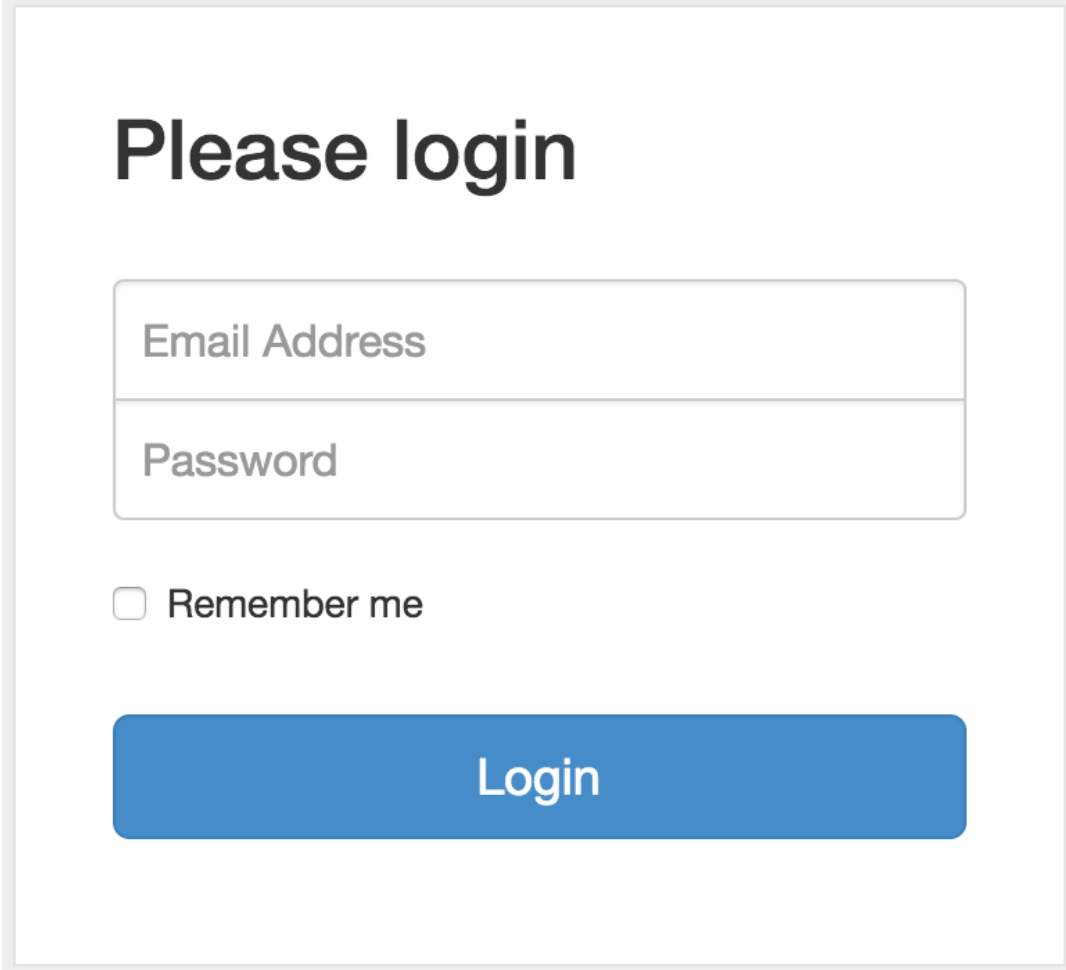
//Javascript (es5)

```
function add(a, b) {  
  // if "a" is 0, 1 is assigned to "a".  
  a=a || 1;  
  b=b || 2;  
  return a + b;  
}
```

```
let result = add(0, 0); // result = 3 X  
console.log(result);
```

Example of Inspection

- Inspect the UI of Login form and find out defects



Please login

Email Address

Password

☐ Remember me

Login

Example of Inspection (cont.)

- Improvement after inspection

Please login

Email Address

Password

☐ Remember me

Login

Cancel

Register

Forget Password

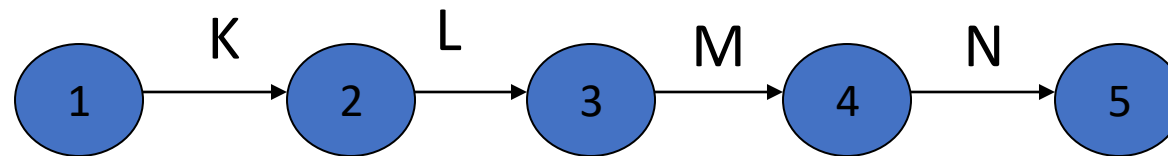
Create a test case for ping-pong ball selection

- A machine must count only orange ping-pong ball

Test case	Example Input	Actual Result
Correct color-Correct size	Orange ping-pong ball	Count
Incorrect color- Correct size	White, red , blue, green ping-pong ball	Not count
Correct color-Incorrect size	Orange basketball, orange volleyball, orange box,	Not count
Incorrect color-Incorrect size	Multi-color basketball, non-orange box	Not count

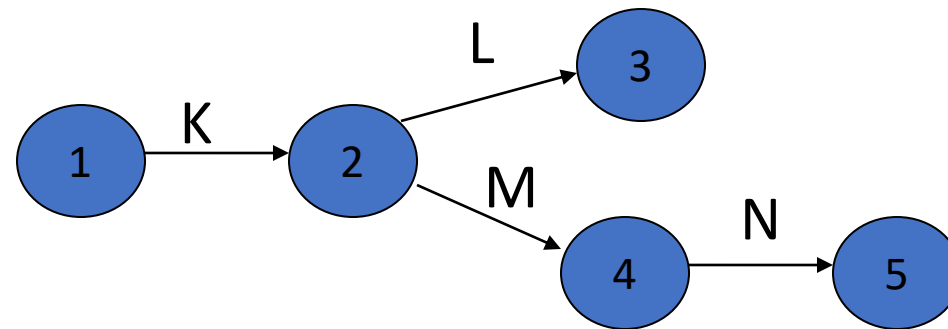
Example I

- K – integration testing
- L – Install Software
- M – Write Manual
- N – Train Users



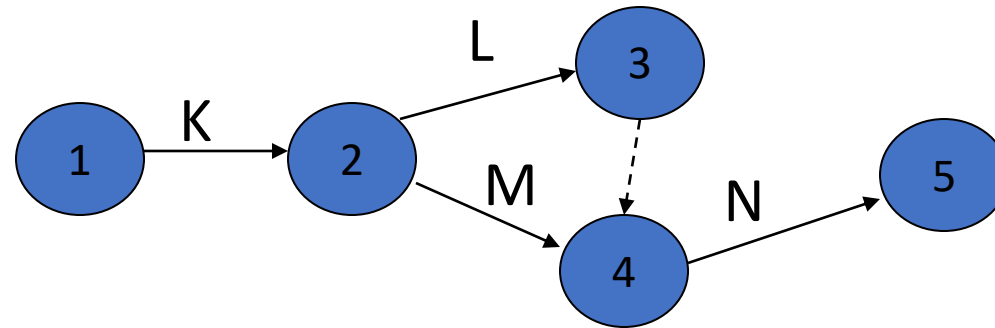
Example I (cont.)

- K – integration testing
- L – Install Software
- M – Write Manual
- N – Train Users



Example I (Cont.)

- K – integration testing
- L – Install Software
- M – Write Manual
- N – Train Users



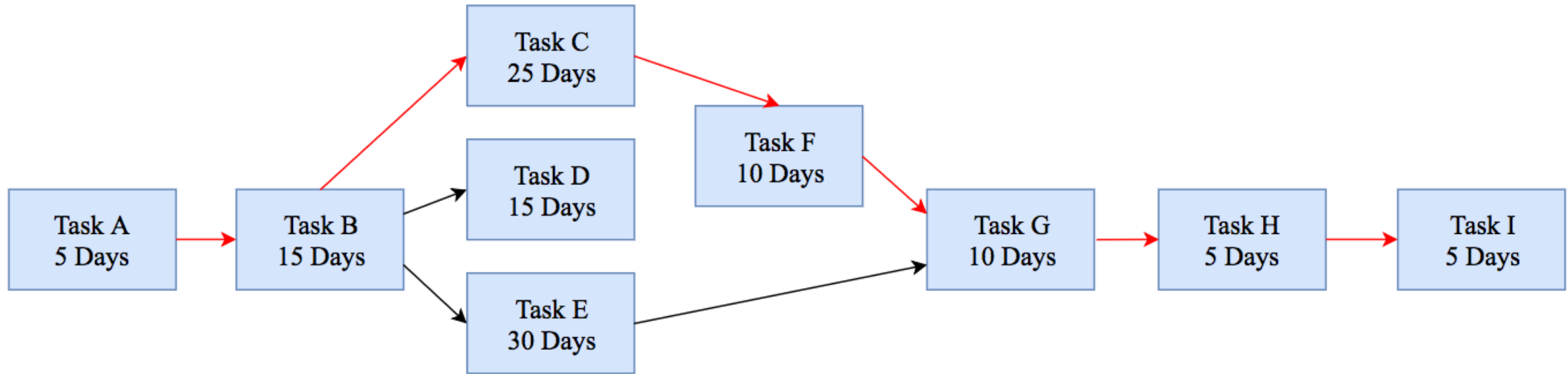
Dashed line is a *dummy activity*.

Example III

- Draw Pert diagram for the scheduling table below
- Show the critical path and calculate duration of critical path
- Draw Gantt chart

Task	Description	Duration (Working Days)	Predecessor/s
A	Requirement Analysis	5	
B	Systems Design	15	A
C	Programming	25	B
D	telecoms	15	B
E	Hardware Installation	30	B
F	Integration	10	C, D
G	System Testing	10	E, F
H	Training/Support	5	G
I	Handover and Go-Live	5	H

Pert Diagram



Lines of Code (LOC)

- What's a line of code?
 - The measurement was the first propose when programs were typed on cards with one line per card
 - How does this correspondent to statements as in Java which can span several lines or where there can be several statements on one line
- What programs should be counted as part of the system?
- This model assumes that there is a linear relationship between system size and volume of documentation

Lines of Code (cont.)

- **Physical LOC** is a count of lines in the text of the program's source code excluding comment lines.
- **Logical LOC** is to count the number of executable "statements".

Example of LOC

- How many lines of codes for (1) (2) and (3)

1.

```
1. main() {  
2.     printf("Hello World\n");  
3. }
```

2.

```
1. main()  
2. {  
3.     printf("Hello World\n");  
4. }
```

3.

```
1. main()  
2. {  
3.     // Print Hello World  
4.     printf("Hello World\n");  
5.  
6. }
```

Section 1

- Physical Lines of Code 3
- Logical Lines of Code 2
- Comment 0

Section 2

- Physical Lines of Code 4
- Logical Lines of Code 2
- Comment 0

Section 3

- Physical Lines of Code 4
- Logical Lines of Code 2
- Comment 1

Example of LOC

- How many lines of codes for (1) and (2)

1.

```
1. for (i = 0; i < 100; i++) printf("hello"); /* How many  
   lines of code is this? */
```
2.

```
1. /* Now how many lines of code is this? */  
2. for (i = 0; i < 100; i++)  
3. {  
4.     printf("hello");  
5. }
```

Section 1

- Physical Lines of Code 1
- Logical Lines of Code 2
- Comment 1

Section 2

- Physical Lines of Code 4
- Logical Lines of Code 2
- Comment 1

Data Functions (Data Element and Record Element)

■ Internal Logical Files

- ILF is a user identifiable group of logically related data or control information that resides entirely within the application boundary.

■ External Interface Files

- EIF is a user identifiable group of logically related data or control information that is used by the application for reference purposes only. The data resides entirely outside the application boundary and is maintained in an ILF by another application.

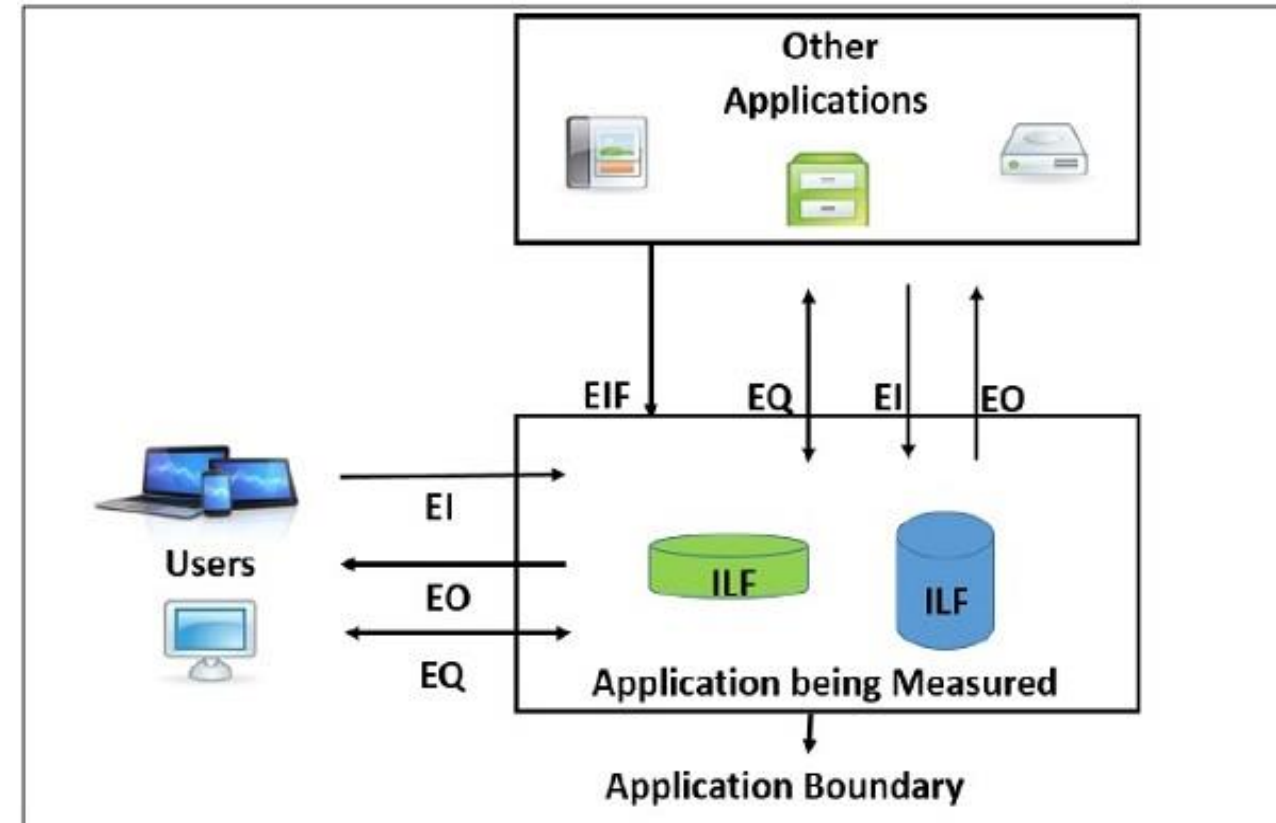


Figure 1: Application Boundary, Data Functions, Transaction Functions

Transaction Functions (Data Element and File Type Reference)

■ External Inputs

- EI is a transaction function in which Data goes “into” the application from outside the boundary to inside. T

■ External Outputs

- EO is a transaction function in which data comes “out” of the system.

■ External Inquiries

- EQ is a transaction function with both input and output components that result in data retrieval

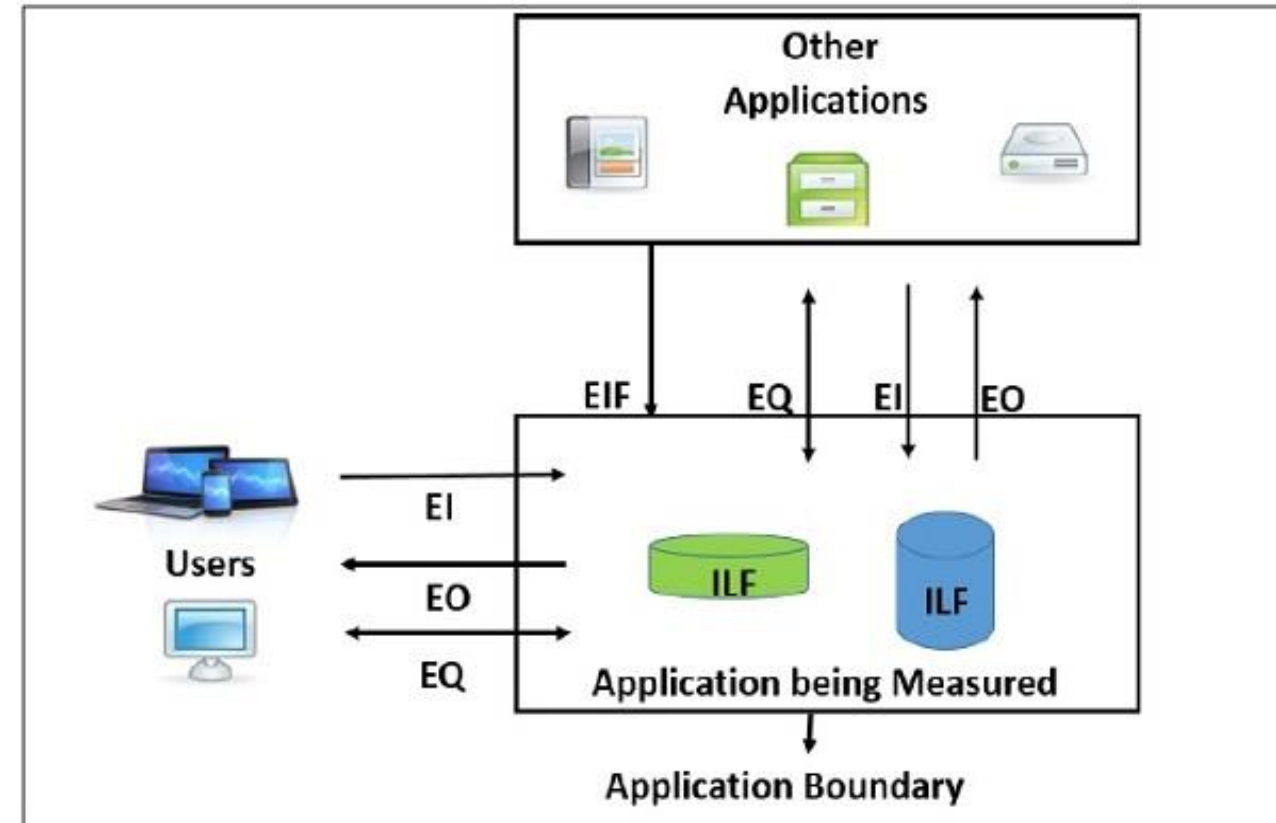


Figure 1: Application Boundary, Data Functions, Transaction Functions

Types of Elements

Record Element Type

- A Record Element Type (RET) is the largest user identifiable subgroup of elements within an ILF or an EIF. It is best to look at logical groupings of data to help identify them.

Data Element Type

- Data Element Type (DET) is the data subgroup within an FTR. They are unique and user identifiable.

File Type Referenced

- File Type Referenced (FTR) is the largest user identifiable subgroup within the EI, EO, or EQ that is referenced to.

Reference: [TutorialsPoint](#)

Practice

No. of Functions	Description	Weight
2	Read text files	Average
3	Write data to csv files	Low
3	Query data from database	High
1	Get name and address from user	Low
3	Print reports on the screen	Average
2	Print reports to the printer	Average
2	Read data from sensors	Low

1. Find function points
2. Find estimated LOC
3. Find cost of software
4. Find Effort

- TDI: 48
- Language: Visual C++
- LOC/P-month: 360
- \$/LOC: 24

Practice - Solution

No. of Functions	Description	Weight	Type
2	Read text files	Average	EIF
3	Write data to csv files	Low	EIF
3	Query data from database	High	ILF
1	Get name and address from user	Low	EI
3	Print reports on the screen	Average	EO
2	Print reports to the printer	Average	EO
2	Read data from sensors	Low	WQ

1. Find function points: 122.04
2. Find estimated LOC: 4,149.36
3. Find cost of software: 99,584.64
4. Find Effort: 11.5

- TDI: 48
- Language: Visual C++
- LOC/P-month: 360
- \$/LOC: 24

Practice

- $PM = A \times \text{Size}^B \times M$
- $A = 2.94$
- Size = 24,000 lines of codes = 24 KLOC
- $B = 1.01 + 0.01 \times \sum SF(i), i=1, \dots, 5$
 - Precedence - new project (4), Development flexibility - no client involvement - Very high (1), Architecture/risk resolution - No risk analysis - V. Low (5), Team cohesion - new team - nominal (3), Process maturity - some control - nominal (3)
 - Scale factor $B = 1.17$
- $M = PERS \times RCPX \times RUSE \times PDIF \times PREX \times FCIL \times SCED = 2 \times 3 \times 2 \times 4 = 48$
- $PM = 2.94 \times 24^{1.17} \times 48$