1. Intro SE

**Definitions**

**Software Systems**

"An integrated whole composed of diverse, interacting, specialized structures and subfunctions." [IEEE]

**Software Engineering**

Disciplined systematic technological activity for producing and maintaining software products by means of a controlled efficient process.

---

**Cost of Errors**

<table>
<thead>
<tr>
<th>Relative Cost</th>
<th>Design</th>
<th>Code</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

2. Table of Contents
3. Definitions
4. Large System Development Costs
5. Development Goals
6. Poor Communication
7. Design Communication
8. Design Elements
9. Structure Chart
10. Parameter Notation
11. Parameter Example
12. Flow Control: Selection
13. Flow Control: Selection (cont.)
14. Flow Control: Loop
15. Misc. Routine Calls
16. Connector Symbol
17. Interface Diagram
18. Global Data
19. Global X-Reference Charts
Large System Development Costs

1. Intro SE

Cost vs. Number of Modules

- System Development Cost Relationships
- Interfacing Cost
- Total Cost
- Cost per Component

<table>
<thead>
<tr>
<th>Project</th>
<th>Cost (millions)</th>
<th>Instructions (millions)</th>
<th>Effort (work years)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apollo Skylab</td>
<td>$209</td>
<td>23</td>
<td>6,000</td>
</tr>
<tr>
<td>NASA Satellite Range Monitoring</td>
<td>$30</td>
<td>1.25</td>
<td>1,000</td>
</tr>
<tr>
<td>FAA Air Traffic Control</td>
<td>$103</td>
<td>1.48</td>
<td>5,000</td>
</tr>
<tr>
<td>Safeguard ABM</td>
<td>$120</td>
<td>1.87</td>
<td>3,500</td>
</tr>
</tbody>
</table>

Development Goals

1. Intro SE

Programming
- Goal: Write Code

- Problem → Code → Solution

Software Design
- Goals
  - Select components
  - State Functions
  - State Interfaces

- Problem → Solution
1. Intro SE

Poor Communication

Developer

Code

??

Maintainer / Enhancer

Design Communication

Idea (vague)

Design Document
- Concrete and Specific
- Can be analyzed
- Can be compared to implementation
- Traceable through life cycle

Design Document

Developer
1. Intro SE

**Design Elements**

A Design should contain
- Components
- Procedures
- Functions

Function of Each Component
- Suggestive names

Interfaces
- Control
  - Calling Hierarchy
- Data
  - Parameters
  - Global Variables
  - Files

**Structure Chart**

Structure Chart
- A graphic tool used to hierarchically model the design solution of a problem.

Contains:
- Individual modules
- Data passed to/from modules
- Control Interfaces between modules

Does NOT contain a complete representation of the internal structure of individual modules.

Basic Elements

```
Calling Module          GetWord
                        ↓
Called Module          GetNext Char
```

Control Relationship
Parameter Notation

| Parameter Direction Flow
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3 Types</td>
</tr>
<tr>
<td>1. Input: Value Parameters &amp; Const Reference Parameters</td>
</tr>
<tr>
<td>2. Output: Reference Parameters (Function changes parameter values independently of parameter’s original [passed] value.)</td>
</tr>
<tr>
<td>3. I/O: Reference Parameters (Function changes parameter values dependent upon of parameter’s original [passed] value.)</td>
</tr>
</tbody>
</table>

| Parameter Classes
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2 Classes</td>
</tr>
<tr>
<td>1. Data Parameter: Any parameter upon which NO decision in the called module’s or calling module’s code is based.</td>
</tr>
<tr>
<td>2. Control Parameter: Any parameter upon which a decision in the called module’s or calling module’s code is based.</td>
</tr>
</tbody>
</table>

Parameter Example

```c
void GetNextChar ( . . . ) {
    GetChar (ch, uplow, terminator); if (terminator) . . .
}
```

Note: Function return values are treated as output parameters (list variable to which they are assigned).

Calls to multiple functions are usually listed left-right in order of execution.
Flow Control: Selection

What does the function header for DrawBox look like?

void Move(...) {
  ...  
  if (ch == plus)  
    DrawBox(row, col);  
  ... 
}

Conditional Call

Move

ch

row

col

draw Box

Select statements are diagrammed in a similar manner with multiple calls emanating from the decision diamond.
Flow Control: Loop

```c
void Drawbox ( . . . ) {
   ...
   while (ch != plus) {
      ...
      DrawLine ( dir );
      ...
   }
   ...
```

1. Intro SE
13

Misc. Routine Calls

Recursive Routines:
- Routines that call themselves
  - Factorial

Operating System Calls:
- Calendar
- Date

Predefined Module:
- (library routines)
  - Get File
  - fexists

Intro Data Structures & SE
Large designs span many pages

**Connector Symbol**

**Interface Diagram**

**Interface Diagram #3**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Dir</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>data</td>
<td>in</td>
<td>Cartesian coords</td>
</tr>
<tr>
<td>y</td>
<td>data</td>
<td>in</td>
<td>of point</td>
</tr>
<tr>
<td>z</td>
<td>data</td>
<td>in</td>
<td>scale of axis</td>
</tr>
<tr>
<td>scale</td>
<td>data</td>
<td>in</td>
<td>angle degrees (rad)</td>
</tr>
<tr>
<td>angrad</td>
<td>cont</td>
<td>in</td>
<td>angle degrees (rad)</td>
</tr>
<tr>
<td>angle</td>
<td>data</td>
<td>out</td>
<td>polar coords</td>
</tr>
<tr>
<td>dist</td>
<td>data</td>
<td>out</td>
<td>of point</td>
</tr>
</tbody>
</table>
Global Data

Global Data (extern)
- Must be highly justifiable (semi-global)
  separately compiled, module scope

Operation
Assign
Update
Reference

Copy
Get Word
Up Case
Print Word

Assign
Reference

word

Get
Case
Print
Word

Global X-Reference Charts

Global Variables introduce complexities that must be mapped and traceable.

<table>
<thead>
<tr>
<th>Procedures</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetNext</td>
</tr>
<tr>
<td>Advance</td>
</tr>
<tr>
<td>PutNext</td>
</tr>
<tr>
<td>Reset</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Global Structures</th>
</tr>
</thead>
<tbody>
<tr>
<td>⋮</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Form</th>
<th>Setting</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

U : Update  A : Assign  R : Reference

Intro to Software Engineering