Intro to Software Engineering

Slides

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

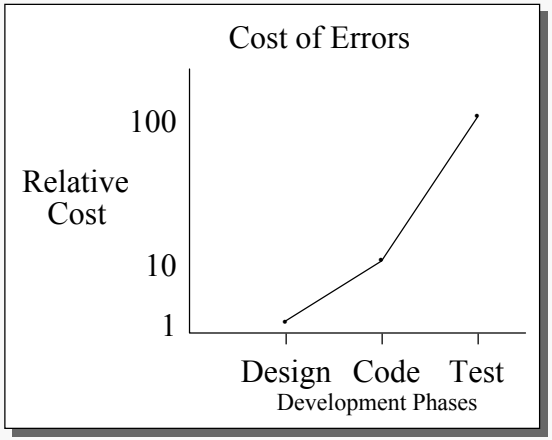
Definitions

Software Systems

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

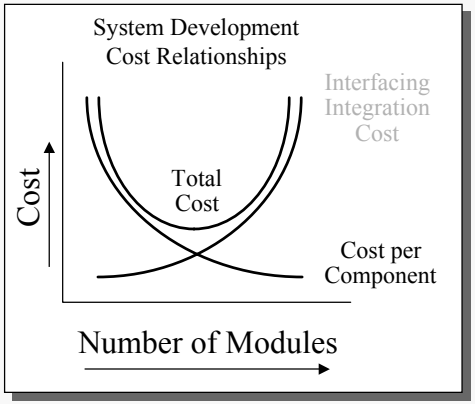
Software Engineering

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



Large System Development Costs

Cost vs. Number of Modules

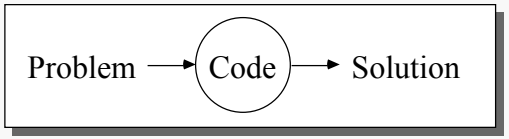


Project	Cost (millions)	Instructions (millions)	Effort (work years)
Apollo Skylab	$209	23	6,000
NASA Satellite Range Monitoring	$ 30	1.25	1,000
FAA Air Traffic Control	$103	1.48	5,000
Safeguard ABM	$120	1.87	3,500

Development Goals

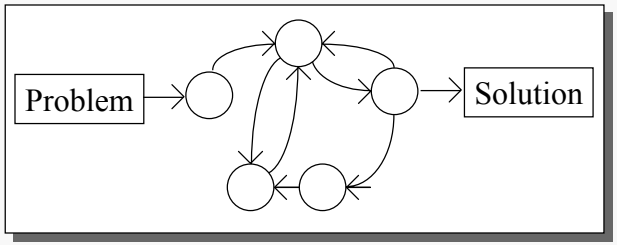
Programming

- Goal: Write Code

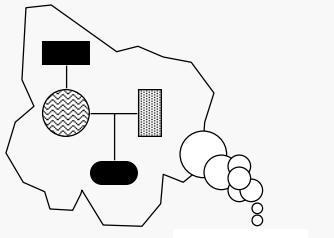


Software Design

- Goals
  - Select components
  - State Functions
  - State Interfaces



Poor Communication



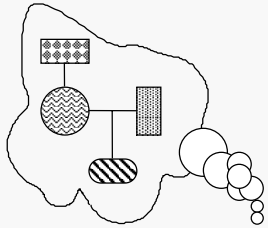
Developer

Code

??

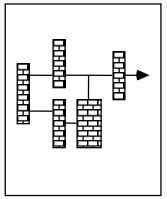
Maintainer / Enhancer

Design Communication



Developer

Idea
(vague)



Design Document

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

Design
Document

Design Elements

A Design should contain
- Components
- Procedures
- Functions

Function of Each Component
- Suggestive names

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

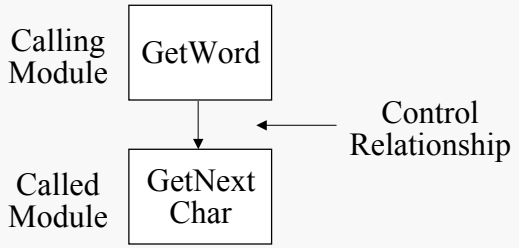
Structure Chart

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



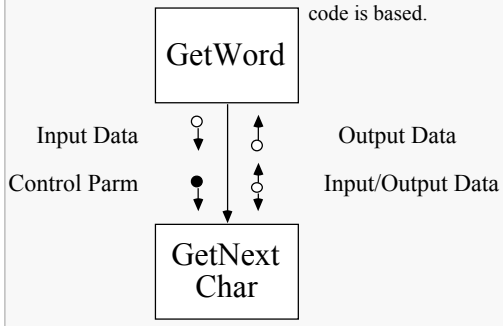
Parameter Notation

Parameter Direction Flow

- 3 Types
  1. Input: Value Parameters & Const Reference Parameters
  2. Output: Reference Parameters (Function changes parameter values independently of parameter's original [passed] value.)
  3. I/O: Reference Parameters (Function changes parameter values dependent upon of parameter's original [passed] value.)

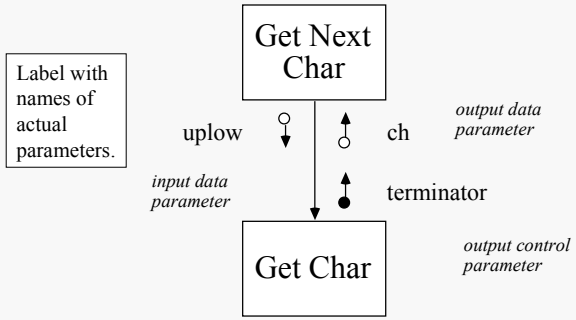
Parameter Classes

- 2 Classes
  1. Data Parameter: Any parameter upon which NO decision in the called module's or calling module's code is based.
  2. Control Parameter: Any parameter upon which a decision in the called module's or calling module's code is based.



Parameter Example


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

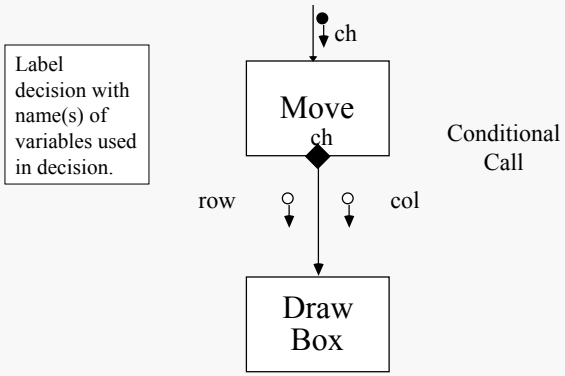



```
void GetChar(char& chact,
             uplowtype uplowCase,
             bool& terminal )
```

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

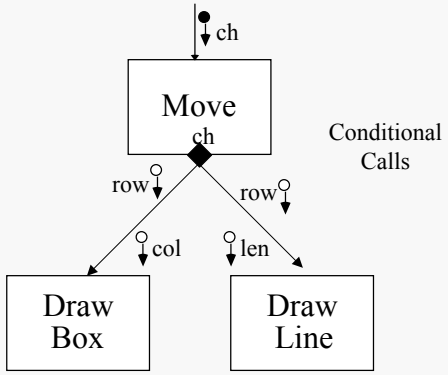
Calls to multiple functions are usually listed left-right in order of execution.


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



What does the function header for DrawBox look like?


```
void Move(. . . ) {
    ...
    if ( ch == plus )
        DrawBox (row, col);
    else
        DrawLine (row, len);
        DrawLine (row, len);
    ...
```



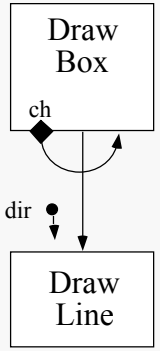
Select statements are diagrammed in a similar manner with multiple calls emanating from the decision diamond.

Flow Control: Loop


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

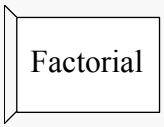
Label with name(s) of variables used in decision.

Conditional Loop

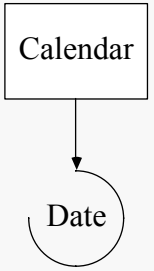


Misc. Routine Calls

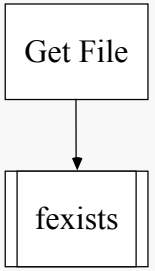
Recursive Routines:
- Routines that call themselves



Operating System Calls:

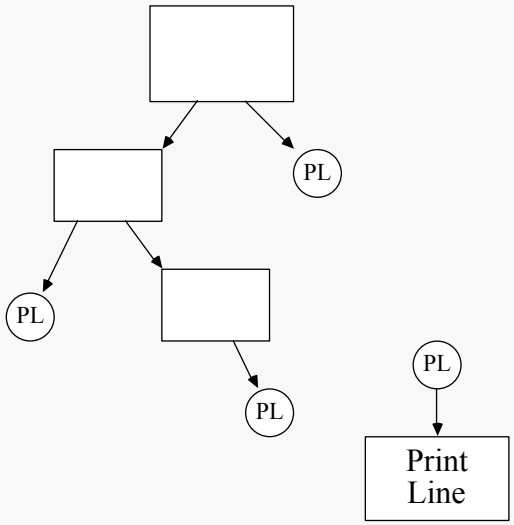


Predefined Module:
- (library routines)

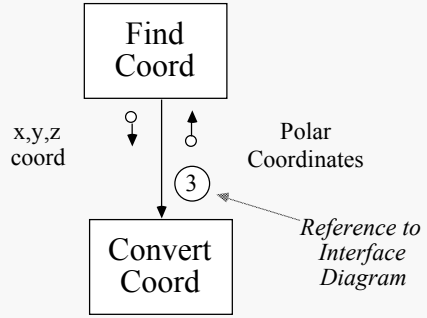


Connector Symbol

Large designs span many pages



Interface Diagram



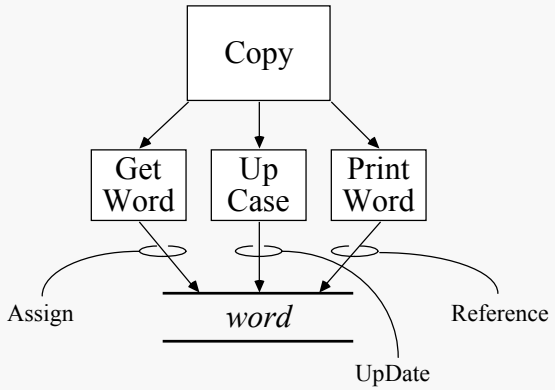
Interface Diagram #3

Parameter	Type	Dir	Description
x	data	in	Cartesian coords
y	data	in	of point
z	data	in	
scale	data	in	scale of axis
angrad	cont	in	angle degrees (rad)
angle	data	out	polar coords
dist	data	out	of point

Global Data

Global Data (extern)

- Must be highly justifiable (semi-global)
  - separately compiled, module scope
- Operation
  - Assign
  - Update
  - Reference



Global X-Reference Charts

Global Variables introduce complexities that must be mapped and traceable.

Global Structures

Procedures	•••	Form	Setting	Date	•••
⋮					
GetNext		R	U		
Advance		U	A	R	
PutNext		A	R		
Reset			A	A	
⋮					

U : Update A : Assign R : Reference