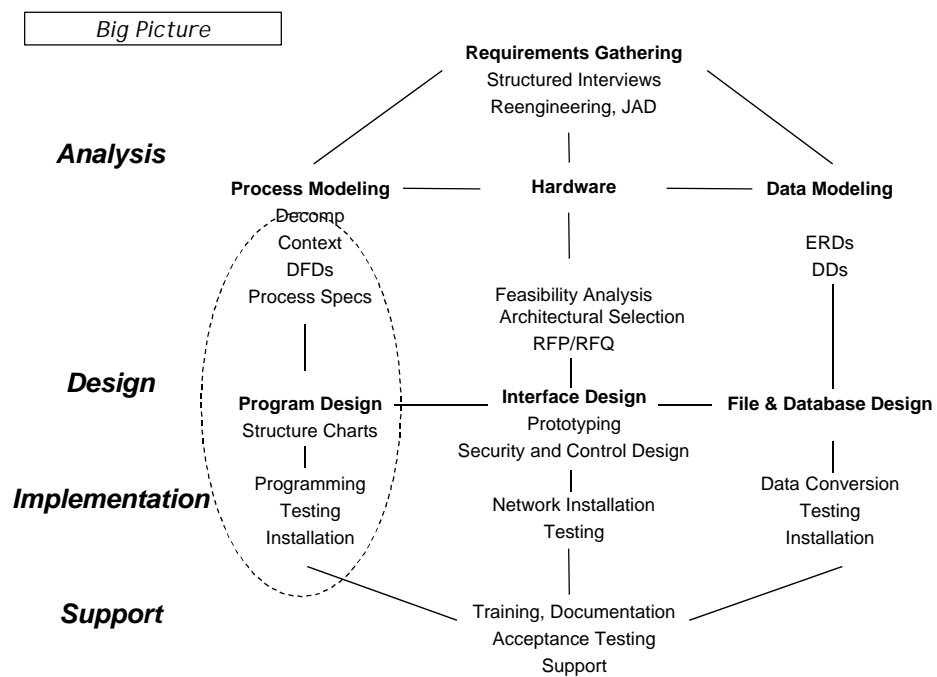
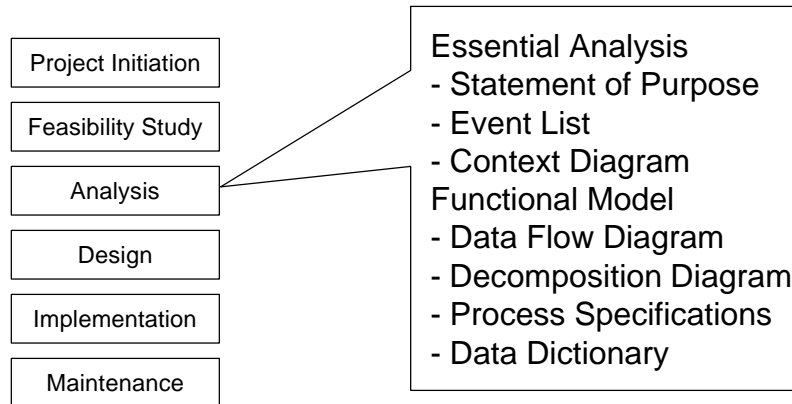


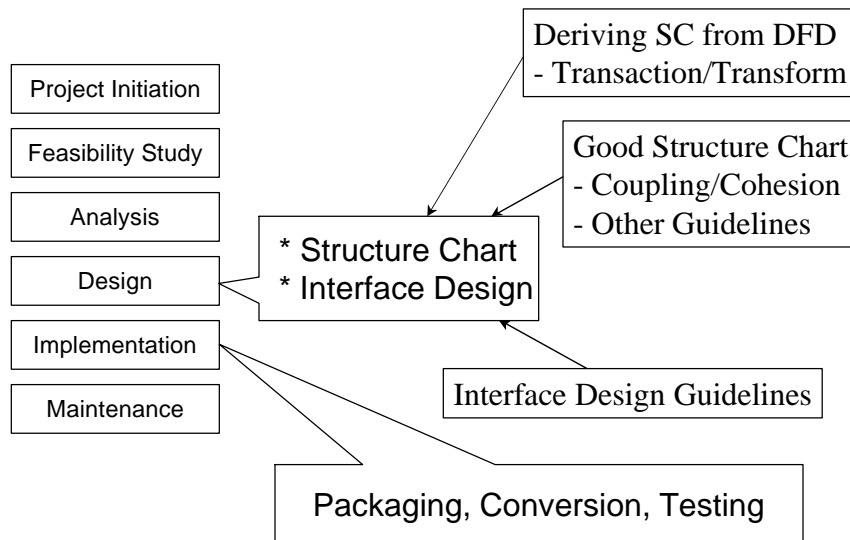
# Structured Process Modeling



## Concept Map of Analysis



## Concept Map of Design



## Data Flow Diagrams

- DFDs model the
  - Sources and destinations of data (external entities)
  - Data inputs and outputs (data flows)
  - Actions that transform inputs into outputs (processes)
  - Data maintained by an information system (data stores)

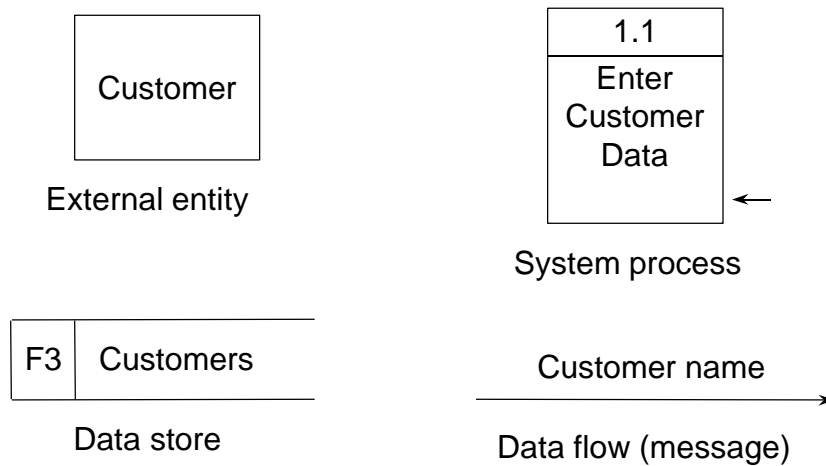
## Logical vs. Physical DFDs

- Logical DFDs model processes and data stores at logical level
  - Does not specify who and how
- Physical DFDs provide details about physical implementation
  - Who performs what processes
  - How processes are performed
  - Where data are stored

## Logical and Physical DFDs

- Traditionally, we modeled the existing physical system, then converted it to a logical model of the existing system
- Changes were made to logical model to reflect requirements of new system, then new physical model was developed

## DFD Symbols



## Rules for Constructing DFDs

- Sequence processes from left to right, top to bottom
- External entities and data stores may be repeated to avoid intersecting lines
- Every data flow must begin or end at a process
- Avoid black holes and magic processes

## Functional Decomposition

- DFDs support top-down analysis and functional decomposition
- Leveled DFDs show successive levels of decomposition
- Successive levels show children of the parent process
  - All data flows in parent process must also appear in child diagrams (vertical balancing)

## Turning Text Into DFD

- Read text and identify candidate processes
- Reread text to identify input and output data flows
- Reread text to identify external entities and data stores
- Finally, put the pieces together

## Design Definition

- Design
  - an outline, sketch, or plan of the form or structure of work
- Computer System Design
  - the activity of transforming a statement of what is required to be accomplished into a plan for implementing that requirement on an electronic automaton.

## Structured Design

## Structured Design

- Uses problem definition to guide solution
  - Transform the problem description (DFD) into a solution description (SC) via design strategies.
- Manages complexity with “black boxes”
  - Modules which can be reasoned about solely on their input and output.
- Uses graphic tools (structure chart)
- Offers strategies for deriving solutions
  - Transform and transaction analysis
- Offers design quality criteria

## Problem Definition Guides Solution

- If your only tool is a hammer, everything looks like a nail
  - Often, solutions are predetermined, not by the problem definition but by the designs preconceived notions how systems should be
  - “All systems should ...”
    - Be event processing
    - Be batch driven
    - Have one input and one output module
- Structure Design attempts to eliminate such predetermined design

## Simplifying a System

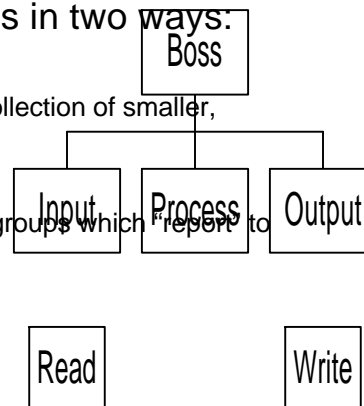
- Structure design seeks to manage the complexity of large systems in two ways:

- partitioning

- divide a large system into a collection of smaller, interrelated modules

- hierarchical organization

- arrange the modules into subgroups which report to (fewer) high-level modules





## Partitioned “Black Boxes”

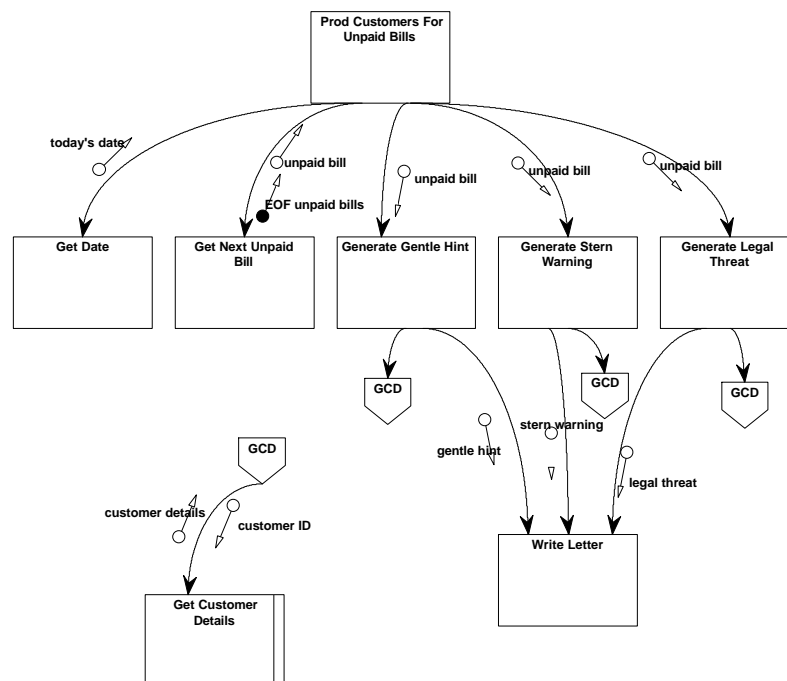
- Easily constructed
  - Compose modules based only on I/O
- Easily tested
  - Narrow search for faults based on test of I/O only
- Easily corrected
  - Repair modules in isolation to give correct I/O
- Easily understood
  - Lesser need to understand details of other modules
- Easily modified
  - Add to system functionality by adding modules

## Hierarchical Organization

- Consider hierarchical business organization as analogy to design
  - A manager should have  $\leq 7$  subordinates
  - Work and management should be separated
  - Every department should have a well defined function
  - Every job should be allocated to the proper department
  - Reports to upper management should be meaningful
  - A manager should give only as much information to a subordinate as that person needs in order to get the job done (more meaningful for design)

# Structured Design Tools

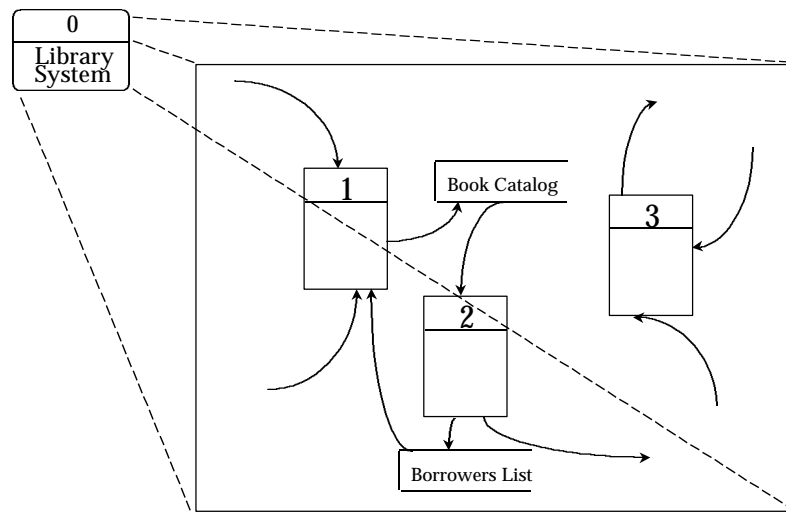
- Structure Chart Shows
  - Partitioning
    - each box is a module
  - Hierarchy
    - Managing modules are shown above with arrow pointing downward
  - Communication
    - Small “flags” indicate control, data, and descriptive information as it is passed from one module to another
- Structure Charts Provides
  - A semi-formal view of system or program structure
  - Documentation and blue-prints for programmers and maintainers



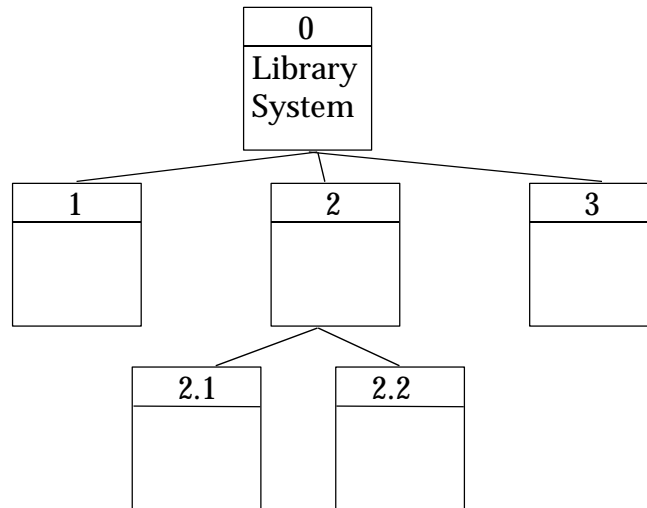
## Structured Design Tools (cont.)

- Module Specification
  - a.k.a. Pseudocode, Structured English
  - Informal flexible design description
- Example
  - module prod customer for unpaid bills
  - open unpaid bill file, customer detail file
  - get today's date
  - repeat
    - » call get next unpaid bill, getting unpaid bill and EOF
  - until EOF
  - set days overdue to today's date - bill date
  - case days overdue
    - » > 90: call generate legal threat using unpaid bill
  - ...

## First Level Data Flow Diagram

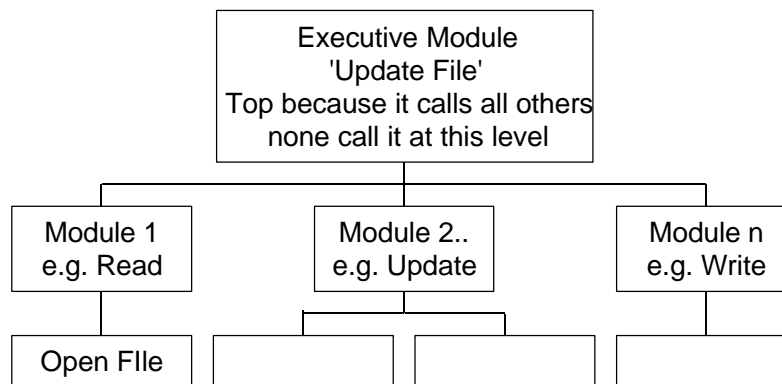


## Functional Decomposition



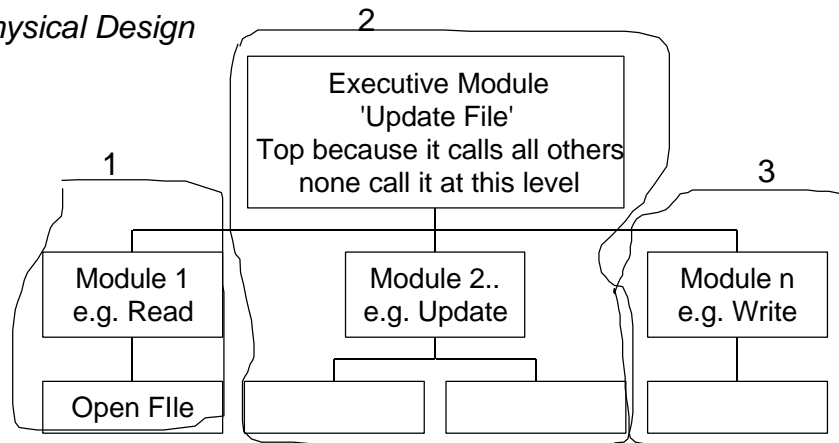
## Structure Chart

### *Logical Design*

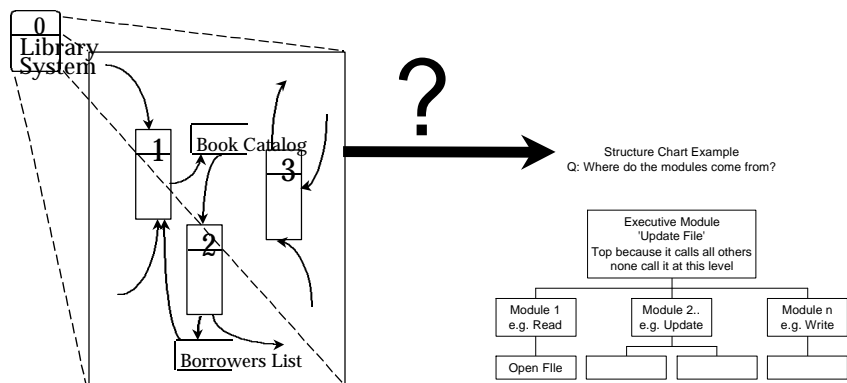


## Structure Chart

*Physical Design*



## DFD to Structure Charts?



## Design Goals

- To Create Programs which are:
  - High Quality
  - Error-free
  - Maintainable, extensible

## Design Guidelines

- Cohesion
  - the degree modules are sufficient to carry out one, single, well-defined function
- Coupling
  - Module independence preferred to tight interdependence
- Module Size (reasonable size)
- Span of Control (known and limited)
- Scope of Effect/Control (clear path of relationships in the hierarchy)

