Software Design

Ruchita Shah

Goal

- Goal is to produce a model that will be built later,
- 2 major phase
  - Diversification – acquisition of materials such as components, knowledge etc,
  - Convergence – gradual elimination of all but one component, to final product
- Intuition & experience, principles, criteria & a process leads to final design

S/W DESIGN & Software Engineering

- S/w design at kernel of SE
- First activity after analysis & specification
- Elements of analysis model provide info to create four design models
  - data design
  - archi design
  - i/f design
  - component design

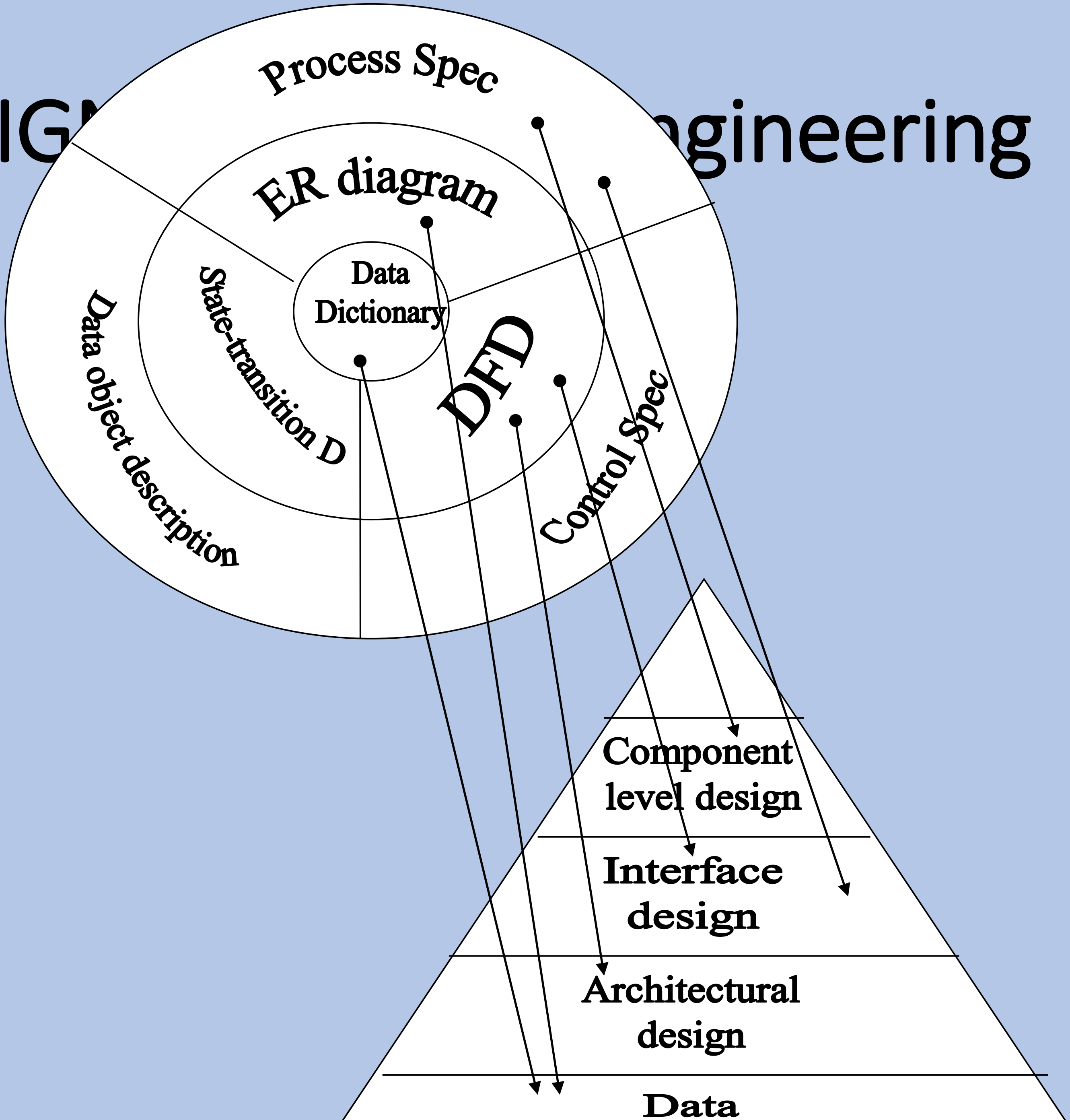
S/W DESIGN & Software Engineering

- Data design transforms into data structures
- Data objects & their relationship defined in ER diagram & details of data in data dictionary forms basis
- Architectural design define relationship b/w major elements
- design pattern to achieve requirements of system
- constraints affecting the archi design are derived from system spec
- I/f design communication of s/w within itself, with system, with human etc implies flow of info
- DFD & control spec forms basis

S/W DESIGN & Software Engineering

- Compo-level design transform structural elements into a procedural description of compo
- info comes from Pspec, Cspec, state- transition data etc
- Decision during design affect success of s/w development & ease with which s/w maintained
- During design quality is fostered
- translation of customer’s requirements into s/w product
- foundation for all SE
- without design unstable product, fail if small change, difficult to test, quality inaccessible

S/W DESIG … ngineering



THE DESIGN PROCESS

- An iterative process
- Requirements translated in s/w
- Iinitially design at a high abstraction i.e. system level
- then detailed – data, functional & behavior
- Design & S/w Quality : During design quality is assessed with FTR
- Three signs to guide a good design (1) design must implement all explicit requirements, accommodate implicit requirements (2) readable, understandable for those who generate code, test & support s/w (3) provide complete picture of s/w, address data, functional & behavioral domains

THE DESIGN PROCESS

- For good design, follow guide lines :

1. Design should exhibit archi structure that(1) created using recognizable design pattern (2) contains compo exhibiting good design characteristics (3) facilitates implementation & testing
2. Modular, logically partitioned that perform specific functions
3. Contain distinct representation of data, archi, i/f & compo
4. Lead to data structures appropriate for objects to be implemented
5. Lead to compo having independent functional characteristic
6. Lead to i/f that reduce complexity of connection b/w modules & external env
7. Derived using repeatable methods

THE DESIGN PROCESS

- The Evolution of S/w Design : Early design concentrated on criteria for developing modular programs & refining s/w structure in top-down manner, then structured programming evolved – procedural aspect of design, then methods for translating DFD or DS into design, newer approach propose OO approach, today emphasis on s/w archi & design pattern
- Many methods, contains common characteristics : (1) mechanism for translation of analysis into design (2) notation of functional compo & their i/f (3) heuristics for refinement & partitioning (4) quality assessment

DESIGN PRINCIPLES

- Design process describe all aspects of s/w
- Requires creative skill, experience, sense for good s/w & commitment to quality
- Design model begins by representing total then refining to provide details
- Design principles enable s/w engineer to navigate design process

1. The design process should not suffer from tunnel vision : consider alternatives, judge each on requirements, resources available & design concepts

DESIGN PRINCIPLES

2. The design should be traceable to the analysis model : single element of design trace to multiple requirements, means of tracking how requirements satisfied in design
3. The design should not reinvent the wheel : Chose design pattern which takes less time, short time & limited resources, design time for new ideas
4. The design should minimize the intellectual distance b/w the s/w & he problem as it exists in the real world : mimic the structure of the problem domain
5. The design should exhibit uniformity & integrity : uniform as one person developed entire system, proper i/f b/w component

DESIGN PRINCIPLES

6. The design should be structured to accommodate change
7. The design should be structured to degrade gently, even when aberrant data, events or operating conditions are encountered : never bomb, accommodate unusual circumstances, if terminate than gracefully
8. Design is not coding, coding is not design : level of abstraction higher in design than code
9. The design should be assessed for quality as it is being created, not after the fact : measures to assess quality
10. The design should be reviewed to minimize conceptual errors : focus on small errors makes on miss big ones, first analyze for major conceptual elements then details

DESIGN PRINCIPLES

- If applied properly exhibit high quality
- external quality factors those observed by users – speed, reliability, correctness, usability etc
- internal quality factors imp to s/w engineers
- lead to high quality design

DESIGN CONCEPTS

- Fundamental design concepts provide foundation for sophisticated designs
- Address following
  - Partition of s/w
  - Transformation of concepts into data structure & functions
  - Technical quality

DESIGN CONCEPTS - Abstraction

- For modular solutions many level of abstraction
- highest level - solution stated in broad terms,use language of problem env
- lower level – more procedural detail, language problem oriented plus implementation oriented
- lowest level – solution stated in manner directly implemented
- During system engineering-s/w as an element of computer-based system
- during requirements analysis s/w in terms of problem env
- Abstraction reduces as move to design

DESIGN CONCEPTS - Abstraction

- lowest level of abstraction when code
- At each level of abs procedural & data abstracts created
- Procedural abs – sequence of instructions having limited & specific function ex open a door, open has long sequence of procedure – walk to door, hold know, turn & pull door, step away etc
- Data abs – collection of data describing data objects, ex door, encompass attributes of door – type, swing direction weight etc, procedural abs use info contained in attributes of data abs
- Programming lang allows creation of abstract data type – Ada, C++, third abs is control abs, implies program control mechanism

DESIGN CONCEPTS - Refinement

- A top-down design strategy
- A hierarchy is developed by decomposing functions in stepwise fashion until programming language statement
- process of elaboration starts with statement of function
- no info about internal working or structure
- elaboration of original statement
- abstraction & refinement complementary to each other

DESIGN CONCEPTS - Modularity

- S/w divided in components,
- A large program in a single module is not easy to grasp

 C(x) – defines complexity of problem

 E(x) – efforts to solve problem

 for two problem P1& P2,

$$\text{if } C(p1) > C(p2) \text{ then } E(P1) > E(P2) \quad \text{-------(1)}$$

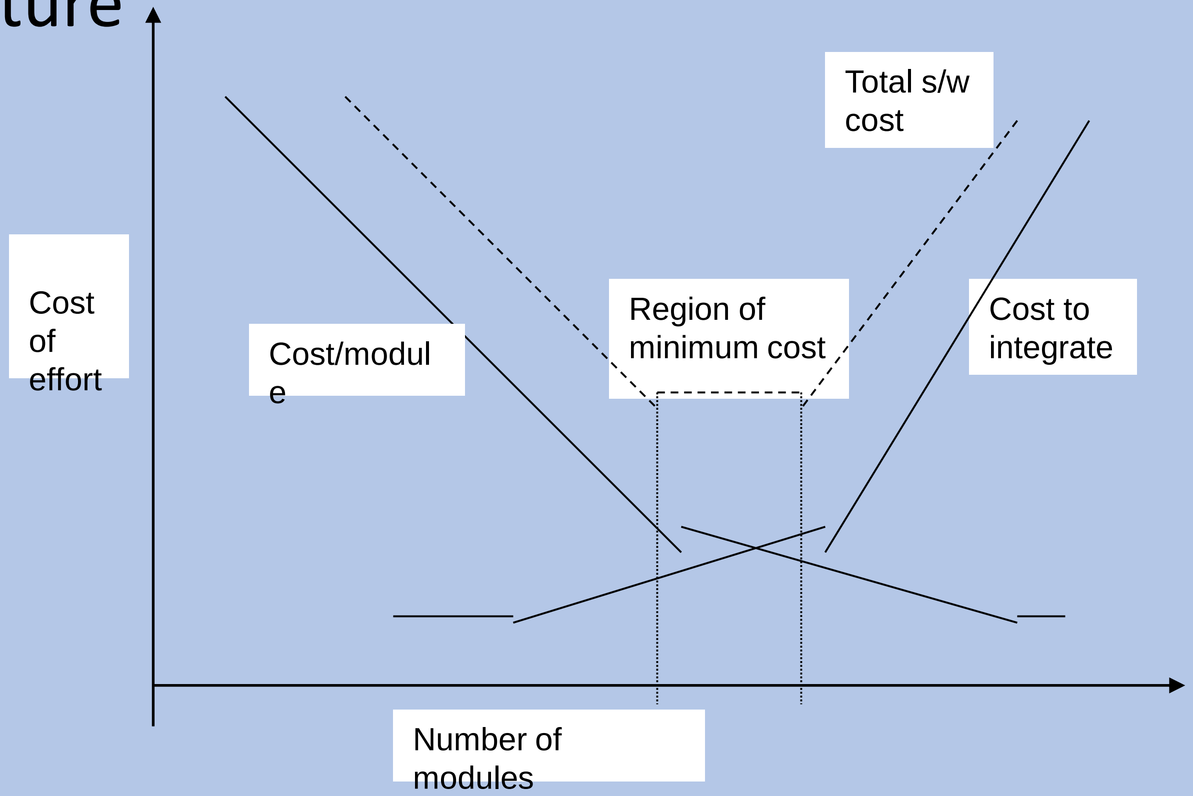
$$C(P1 + P2) > C(P1) + C(P2) \quad \text{-------(2)}$$

i.e. combine complexity of P1 & P2 is more than sum of complexity of each, so

$$E(P1 + P2) > E(P1) + E(P2) \quad \text{-------(3)}$$

DESIGN CONCEPTS - Modularity

- Easy to solve complex problem by breaking into manageable pieces
- If divide indefinitely effort required will be negligible
- but other factors come in picture



DESIGN CONCEPTS - Modularity

- Effort & cost reduces with increase in total no of modules
- but effort & cost increases for their integration
- unable to decide region of minimal cost
- how much modular s/w is an effective modular design can be defined by criteria
- Modular Decomposability : if design method allows decomposition of problem in sub problem, will reduce complexity

DESIGN CONCEPTS - Modularity

- Modular Composability : if design method allows assembly of reusable components into new system, easier to build
- Modular Understandability : Module understood as standalone unit, easy to build
- Modular Continuity : Change in system requirements lead to change in individual module rather than system change, minimum change impact
- Modular Protection : effect of errors contained within module, minimum side effects

DESIGN CONCEPTS - Modularity

- Certain systems cannot spare minimal speed or memory overheads used by sub programs ex real-time & embedded s/w
- These s/w designed with modularity but developed in-line

DESIGN CONCEPTS - S/w Architecture

- Overall structure of s/w & its integrity
- Hierarchy of program components & the manner in which they interact & structure of data used
- Properties of architectural design are
    - Structural properties – defines components & their interaction
    - Extra-functional properties – how to achieve performance, capabilities security etc
    - Families of related systems – repeatable design, common to similar system, reuse

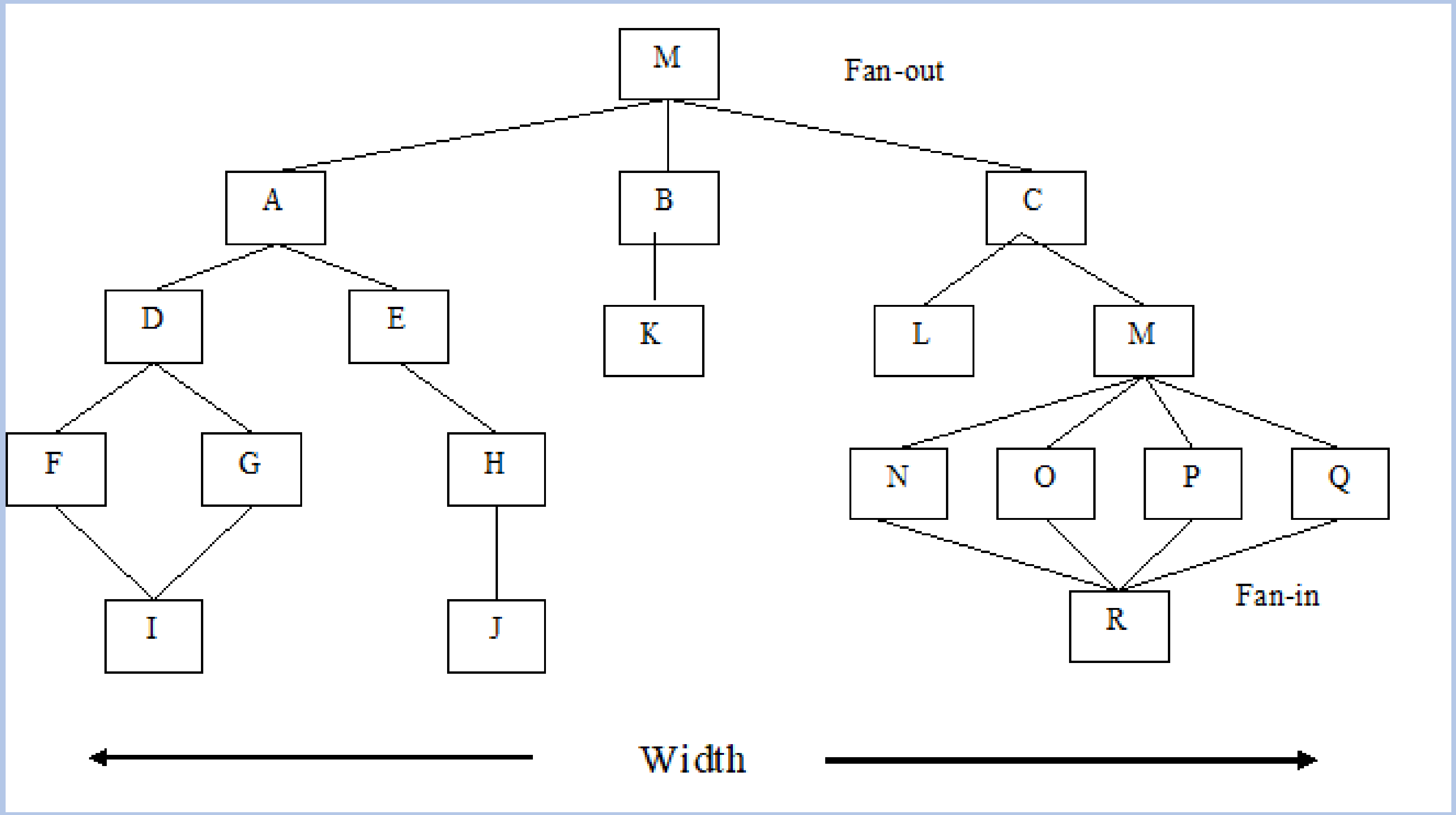
DESIGN CONCEPTS - S/w Architecture

- No of models
- Structural model – archi as organized collection of components
- Framework model – level of design abstraction, repeatable archi design framework
- Dynamic model – address behavior of program archi
- Process model – focus on design of business or technical process
- Functional models – functional hierarchy

DESIGN CONCEPTS - Control Hierarchy

- Program structure
- Represent organization of program components & hierarchy of controls
- Do not specify sequence of process, order etc
- Number of ways to represent control hierarchy
- Common is tree like diagram
- Represent control for call & return

DESIGN CONCEPTS - Control Hierarchy

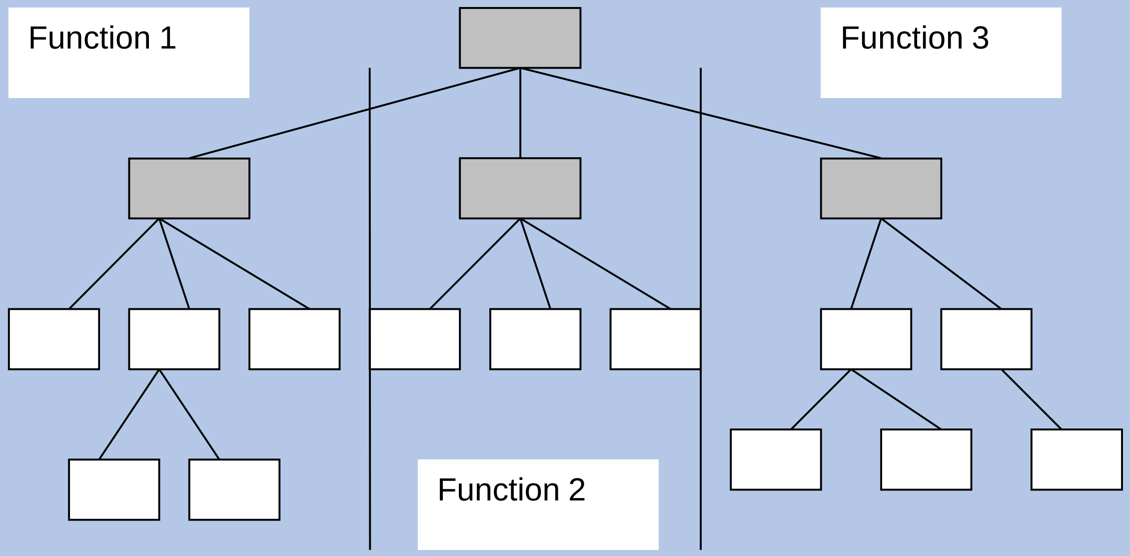


DESIGN CONCEPTS - Control Hierarchy

- Depth is levels of control & width is span of control
- Fan-out – no of modules directly controlled by given module
- Fan-in – no of module directly control a given module
- Super ordinate – a module that control another module
- Subordinate – module controlled by other
- Control hierarchy also represent two characteristics of s/w archi
  - visibility – set of programs that may be invoked by a given component, even indirectly
  - connectivity – set of components directly invoked by given component

DESIGN CONCEPTS - Structural Partitioning

- Hierarchical system partitioned vertically & horizontally

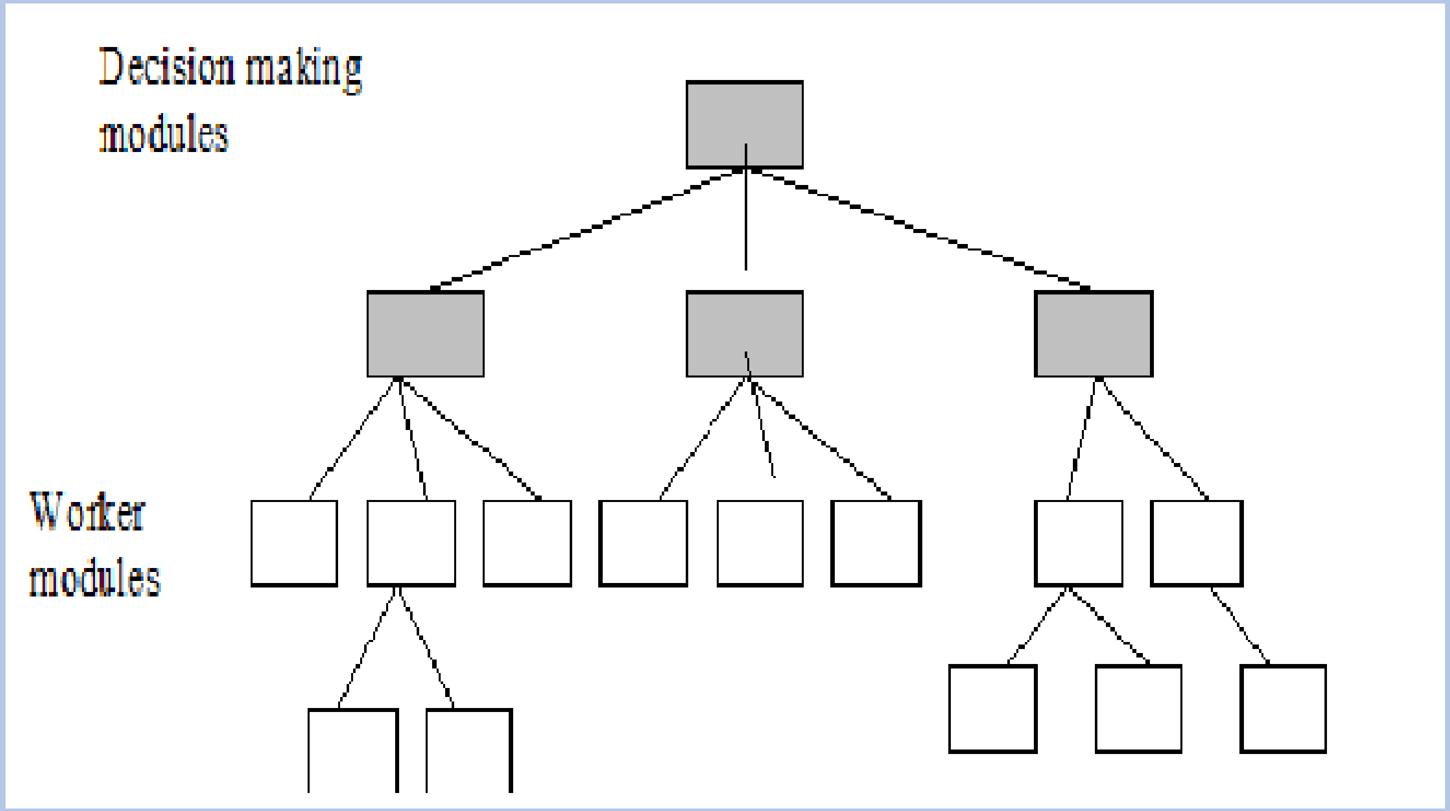


DESIGN CONCEPTS - Structural Partitioning

- Horizontal partitioning
  - separate branches for each major functions
  - control modules coordinate communication & execution of functions
  - simplest ex is input, processing & output
  - benefits are: S/w easier to test, easier to maintain, propagation of fewer side effects, easier to extend
  - Major functions decoupled
  - change less complex
  - extension without side effects
  - Causes more data to be passed across modules
  - control of program flow complex

DESIGN CONCEPTS - Structural Partitioning

- Horizontal partitioning



DESIGN CONCEPTS - Structural Partitioning

- Vertical partitioning – factoring
  - Control & work distributed top-down
  - Top modules perform control function with little processing
  - Lower modules worker, perform i/p, process & o/p
  - Useful when change in program structure
  - change in control module lead to higher propagation of side effects
  - lower module change – less side effects, generally change in i/p, processing or o/p, more maintainable

DESIGN CONCEPTS - Data Structure

- Logical relationship b/w individual data
- Structure of info affect procedural design
- Dictates organization, methods of access, degree of associativity & processing alternatives
- Scalar items – simplest DS, single element, addressed by identifier, size & format language dependent

DESIGN CONCEPTS - Data Structure

- Group of scalar items form vector, have dimensions
- scalar & vectors in variety of formats – linked list – creates a list, nodes
- Hierarchical DS using multilinked structures
- different level of abstraction ex stack

DESIGN CONCEPTS - Software Procedure

- Focus on processing of each module
- Provide precise specification of process
  - Sequence of events
  - Decision points
  - Repetitive operations etc
- A relationship b/w procedures which may include subordinate modules

DESIGN CONCEPTS - Information Hiding

- Decision made within models hide from all others
- Information contained within module inaccessible to other modules
- Hiding & effective modularity by defining independent modules
- Necessary communication
- Defines access constraints within module
- Benefit are easy to test & maintain