Structured design

- Overview of structured design
- Transaction analysis
- Transform analysis
- System integration
9.3 Structured design

- After SSA has produced a set of DFDs, there are three steps to turn the design into structure charts (from which the system can be implemented):

  1. *Transaction analysis*
     
        - divides the system into tractable units

  2. *Transform analysis*

        - converts units into structure charts

  3. *System integration*

        - recombines charts by linking units together
Structured design

(a) Complete system DFD from Analysis

(b) Transaction analysis divides into smaller DFDs

(c) Transform analysis creates structure charts

(d) Structure charts recombined

Transformations of the DFD from analysis into a recombined structure chart via smaller transaction DFDs & their charts.
Step 1: Transaction analysis

- A transaction has five basic components:

1. *event* in the system’s environment that causes the transaction to occur

2. *stimulus* that is applied to the system to inform it about the event

3. *activity* that is performed by the system as a result of the stimulus

4. *response* that is generated in terms of output from the system

5. *effect* that this has on the environment of the system
Step 1: Transaction analysis

- e.g.,
  1. event: student ‘signs up’ for bank account
  2. stimulus: information about student, university, date of opening account, grant source, etc.
  3. activity: add account details to bank records
  4. response: free gifts to student, chequebook, debit card, bank statements
  5. effect: student can spend money

- Later regrouping of transactions in the system structure may be as simple as combining with a case statement.
Step 2: Transform analysis

- Goal is to find a module’s core function:
  - take DFD model of the problem and transform it into hierarchical structure

1. First, identify the central transform in the DFD:
   - central transform lies at the centre of the input and output data flows
   - sometimes the central transform must be created, by adding an extra bubble

2. DFD is ‘picked up’ by central transform so other functions hang from it
   - central transform forms the main body of the module
3. First-cut Structure Chart is formed

- Bubbles replaced by blocks
- Arrows redrawn to show invocations:
  - they describe flow of data in DFD, but they show flow of control (procedure calls) in Structure Chart
- Data flows added as side arrows (data couples)

Illustration of transformation from a simple DFD to a hierarchical structure chart describing the solution.
Transform analysis

- Initial transform makes one-to-one mapping of process bubbles to subprograms

- Central transform is chosen to be the most abstract of the process bubbles

- Least abstract bubbles tend to be closest to the I/O - mapped to the lower levels of the Structure Chart

- Need to create a central transform for a procedure that organizes sequencing of calls to subprocedures (which is not obvious from a single-level DFD)

- First-cut Structure Chart requires further refinement to produce a good design
Step 3: System integration

- Finally, last step combines all separate charts from the transaction and transform analyses:

Structure Chart for transaction servicing, including principal data couples
9 Structured design

• Overview of structured design

• **Transaction analysis** identifies the types of transaction in system, used as components of the design

• **Transform analysis** designs each component separately by extracting a good structure for it

• **System integration** combines components of the system to give a complete solution
9.4 Transaction Analysis

- A transaction has five components:

  (1) **event** within the system's environment

  (2) **stimulus** to the system

  (3) **activity** of the system

  (4) **response** from the system

  (5) **effect** upon the system's environment
Electrical supplier example

(1) event: George wants an electricity service
(2) stimulus: Info about George, his house and installation
(3) activity: Addition of George to customer database
(4) response: A work order to turn service on
(5) effect: George gets electricity

which yields the add_new_customer transaction type:

Three transaction types:

- Turn off customer service
- Add new customer
- Accept customer payment
The structure chart shows how the components for each of the transaction types integrate:
9.5 Transform Analysis

- procedure for converting components of the main DFD isolated by transaction analysis into a structure chart:

1. Draw a Data Flow Diagram of a transaction type
2. Find the central functions of the DFD
3. Convert the DFD into a rough structure chart
4. Refine the structure chart using Structured Design criteria
5. Perform verification
DFD for Produce payroll subsystem

1.1 Validate time report
1.2 Calculate gross pay
1.3 Calculate tax & NI
1.4 Create pay cheque
1.5 Format pay cheque
1.6 Create full pay slip
1.7 Format pay slip

Time sheet records
PAYE & NI tables
Operating system
Employee history
Employee names & addresses
Pay cheque printer
Pay slip printer

Validate time report
Calculate gross pay
Employee ID & current pay slip amounts
Create pay cheque
Format pay cheque
Employee ID &_current pay slip amounts
Create full pay slip
Format pay slip
Pay cheque printer
Pay slip printer

Invalid time report
Employee net pay
Pay cheque details
Formatted pay cheque details
Employee details
Employee ID & current pay slip amounts
Pay slip details
Formatted pay slip details
Pay cheque printer
Pay slip printer
Structure chart for Produce payroll subsystem
Worked example of transform analysis

1. Draw a DFD

   • This should be the result of a structured analysis

2. Identify the Central Transform

   • Part of DFD that contains the essential function
   • Found by inspection, or by “pruning” *afferent* (coming in) and *efferent* (going out) branches in DFD:
     
   (a) Trace each afferent stream from outside to middle, mark the data flow in its most “rich” form

   (b) Trace each efferent stream from outside to middle, mark the data flow in its most “basic” form

   (c) Join all the marks in a closed curve
Update_file transaction type:
3. Produce a first-cut Structure Chart

- Structure chart shows flow of control (whereas DFD shows data flow)
- “Head hunt” for boss module — potential candidates could come from central transform, one that acts as a co-ordinator, or create new boss:

  if there is a good candidate for boss (in central transform),
  then pick up the boss and let all other bubbles hang down;
  else create new boss and hang central transform and each afferent and efferent branch from new boss.
Producing a first-cut structure chart

- Arrow heads removed from data flows — direction of data flow not necessarily same as direction of call

- Arrow heads for calls need to be added and bubbles need to be redrawn as square modules

- Names of modules may not correspond to bubble names — module names depict activities of subordinates too
• Adding read and write library modules gives a first-cut structure chart:
4. Revise the first-cut Structure Chart

To produce better structure, revise the chart as follows:

- Add read and write modules for accessing sources, sinks and files of data
- Factor and reorganize the afferent and efferent modules — keeping the system balanced
- Factor the central transform, if needed.
- Add error handling modules.
- Add initialisation and termination details if required
- Ensure that all modules have names in keeping with their hierarchical roles
- Show all flags that are necessary on a structure chart but not on a DFD: e.g., “end of stream” information
- Check all the design criteria and be prepared to improve the design in keeping with those criteria. Look first at factoring, cohesion, state memory, and correspondence to data structures
• E.g., update_file DFD

(1) Use “Match Transaction With Master Record” as the boss or parent module

(2) Remove data flow arrowheads and names of bubbles, and lift the whole network by the boss module:
(3) Adding detail gives first-cut structure chart:
(4) Module with worst cohesion is “Update and Put Master Record and Report Transaction”

- only has sequential and communicational cohesion
- split it and factor out some functions:
Modifications to design

- All modules are at least sequentially cohesive.

- “Format and Put New Master Record” is sequentially cohesive and trivial — could be removed, except that it is used by another module.

- “Update and Put Master Record” is sequentially cohesive and trivial — remove it, increasing fan-out of boss.

- “Format and Put New Master Record” no longer has fan-in, but removing it would increase fan-out of boss to seven and make boss ‘worry’ about details of “Format new master record” — so leave factored.
• Second-cut Structure Chart:
Summary of Structured design

- Transaction analysis to identify components
  1. Event
  2. Stimulus
  3. Activity
  4. Response
  5. Effect

- Transform analysis to design components
  1. Draw DFD
  2. Identify central transform
  3. Produce first-cut structure chart
  4. Revise and develop design

- System integration to re-combine components