Process Modelling
Data Flow Diagrams

Process Modelling aims to graphically represent the processes which capture, manipulate, store and distribute data.

- data flow diagrams
- function decomposition
- structured English
- decision tables and decision trees

Data flow Diagrams

Data flow diagrams model the flow of data into, through, and out of an information system:

- show the processes that change or transform data
- show the movement of data between processes
- represent a system as a network of processes which transform data flowing between them
Data Flow Diagrams

Data flow diagrams (or DFDs):
- a well-known process modelling technique
- easily understood
- a good communication tool
- model both manual and automated processes

Components of a DFD

1. Process
2. Data flow
3. Data store
4. External agent (Source/Sink)
1. Process

• transforms incoming data flows into outgoing data flows:
  the work performed on data which changes it

• each process has a unique number and name

2. Process

• name each process using a verb and a noun phrase
  eg. calculate price
      check customer details
      accept supplier delivery

• the name of a process should describe what the process does

• avoid vague names where possible
  e.g. “process data”

2. Data Flow

• a data flow represents data in motion, moving from one place in the system to another

• describes a “packet” of data or data that moves together:
  a data flow may consist of many individual, related pieces of data that move together to a common destination

loan application
name each data flow using a noun or noun phrase
e.g. customer order
the name of a data flow should describe the contents of the
data "packet"
the name should include as much information as possible
about the data flow e.g. "customer payment" rather than just
"payment"

• validate customer order
• customer order
• valid customer order
• invalid customer order

3. Data Store

• a data store represents a collection of data flows "at rest"
• each data store has a unique name
• the name should describe the contents of the
data store
• a data store may represent many different types of
physical locations of data
• a data store may be a temporary or a permanent
repository of data

The Data Store
different notations
data flows to and from a data store can remain unlabelled if all attributes in the store are moving, i.e. if an entire data packet (or packets) is going into or out of the data store

4. External Agent (Source/Sink)

- an external agent represents an entity with which the system communicates and which is outside the scope of the system
- an external agent is a source if it is an origin of data coming into the system
- an external agent is a sink if it is a destination of data leaving the system

Suppliers

4. External Agent (Source/Sink)

- data flows connecting the external agents to the processes within the system represent the interface between the system and its environment
- external agents are outside the system and define its boundaries
- an external agent may be both a source and a sink

what a sink does with data it receives from the system and how a source produces data which it inputs to the system are outside the boundary of the system and are not shown on the data flow diagram
Guidelines for Drawing DFDs

- each object on a data flow diagram must have a unique name
- each process must have at least one data flow coming in (input) and at least one data flow going out (output)
- the inputs to a process are different from the outputs of that process
- a process must be able to build its outputs using only the information in its input data flows plus any constant information
Guidelines for Drawing DFDs

Data flows are permitted:
- between processes
- from a data store to a process
- from a process to a data store
- from a source to a process
- from a process to a sink

Guidelines for Drawing DFDs

Data flows are NOT permitted:
- between external agents
- between data stores
- from an external agent to a data store

Identify six errors in this abstract DFD
Levelling Data Flow Diagrams

Any “real” system is too large to represent as a single data flow diagram

- the solution is to decompose the system into a hierarchy of levels of processing
- the process model of the system then consists of a set of levellled data flow diagrams
- levelling of DFDs improves their readability and usefulness as a communication tool

Levelling of DFDs

Levelling creates a hierarchical decomposition of the processing within the system

Context diagram
- Level 0 diagram
  - Level 1 diagrams
    - Level 2 diagrams
      - Level n diagrams
Context Diagram

• the highest level data flow diagram is the context diagram
• the context diagram shows the interaction of the system with its environment in terms of data flows
• the context diagram defines the boundary of the system (the scope of the system)
• only the data flows which leave the system and the data flows which come from outside the system are shown

Example Context Diagram
Level Zero Diagram

- The level zero data flow diagram is the diagram at the level immediately below the context diagram.
- It "expands" the single process on the context diagram to show the major, high-level processes (or functions) within the system.

Example Level Zero Diagram

- All external agents (sources and sinks) are included because the level zero diagram, like the context diagram, represents the entire system.
- The number of each process ends in .0 which corresponds to the level of the diagram:
  - e.g., 1.0, 2.0, 3.0 etc.
  - Often just 1, 2, 3 etc. are used.
Level 1 and lower level diagrams

Each Level 1 diagram and diagrams at lower levels (e.g., Level 2, Level 3) show only a part of the processing.

Level 1 processes are numbered 1.1, 1.2, 1.3, and 2.1, 2.2, 2.3 etc. That part is shown in more detail than on the Level zero diagram.

No external agents are shown on a Level 1 or lower diagrams, as the entire system is not being represented.

Level 1 diagrams

A set of data flow diagrams is created at Level 1. There is one Level 1 diagram for each of the processes at Level zero. Each diagram decomposes a Level zero process into several processes.

Levelling DFDs

Context diagram → Level zero diagram → Diagram 3 (level 1) → Further diagrams or process descriptions → Diagram 3 (level 1)
Guidelines for Levelling DFDs

- **numbering:** when a process is decomposed, its diagram is given the same number as that process
- **balancing of DFDs:** all data flows entering and leaving a process must appear on the corresponding diagram which decomposes that process
- **external agents:** are only included on the two diagrams which represent the entire system, i.e. the context and level zero diagrams
Guidelines for Levelling DFDs

balancing data flows:
if data flows are decomposed at lower levels then the contents of the data flows across levels must be balanced

Guidelines for Levelling DFDs

The access to data stores across levels of diagrams must be consistent:
• the direction of accesses must match and all accesses on higher level diagrams must appear on corresponding lower level diagrams
• a data store is first shown on the highest level diagram where it is accessed by more than one process
• it can then appear on all lower level diagrams where it is accessed

Guidelines for Levelling DFDs

partition processes to:
• form cohesive, related groups of activities
• minimise the data flows between them

all parts of the system need not be decomposed to the same level
Guidelines for Levelling DFDs

How many levels should be in a set of DFDs?

- each diagram usually has between 3 and 7 processes
- level the diagrams until bottom level or primitive processes are reached

*primitive processes* cannot be further decomposed as a data flow diagram.

An example - Context Diagram

An example - Level zero diagram
Level 1 - Diagram 1

1.1 Verify for Completeness
1.2 Acknowledge Application
Acknowledged Application

Level 1 - Diagram 2

Screen Applicants

2.1 Screen Applicants
2.2 Unqualified Application
2.3 Decision
2.4 Unsuccessful File

Qualified Applicant

Schedule Evaluation

References
