Data Flow Diagrams

Outline

Basics

Levels of Detail

Bad & Good DFD Practices

Many further Examples Developed in Class

Data Flow Diagram Notation

Notation:

Source	box
Process (transform)	box with rounded corners
File (store)	box open on right
Sink (destination)	box
Flow	arrow

Sources and sinks may be processes in the larger context, but sources and sinks are beyond the control of your system.

Structural Recursion

- Process nodes that are have internal details may be “exploded”
- Allows top level overview or detailed view of some part
- Flows in/out of node become sources/sinks of detail

  but boxes often are not drawn

© Copyright 1906-10-9, E Robertson

Terminology

The word “process” is overloaded

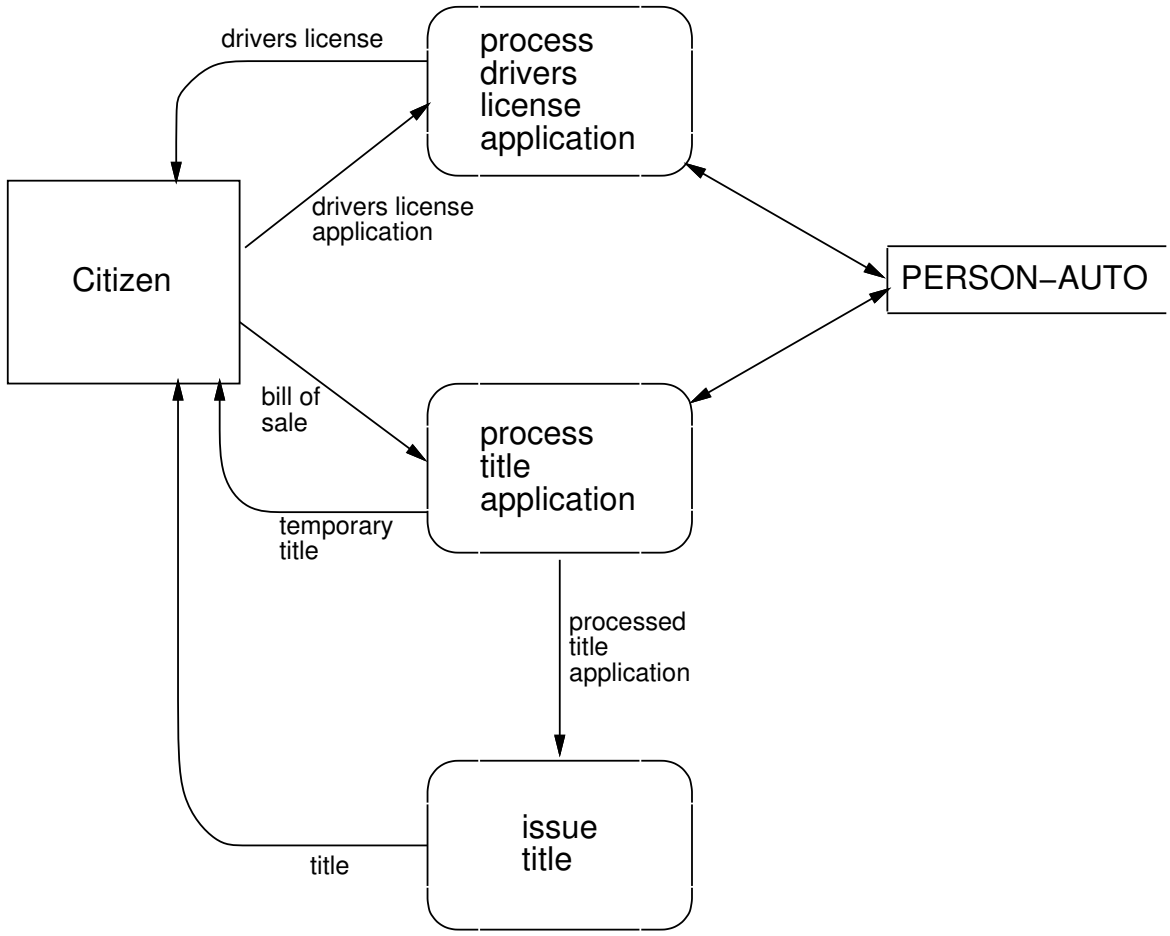
It means:

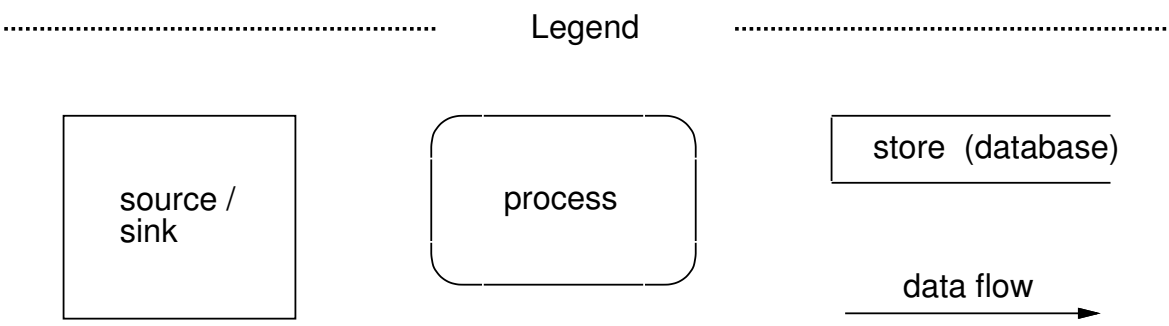
1. entire software engineering activity
2. that aspect of modeling dealing with flows and transformations
3. a node in a DFD

© Copyright 1906-10-9, E Robertson

Example of Notation

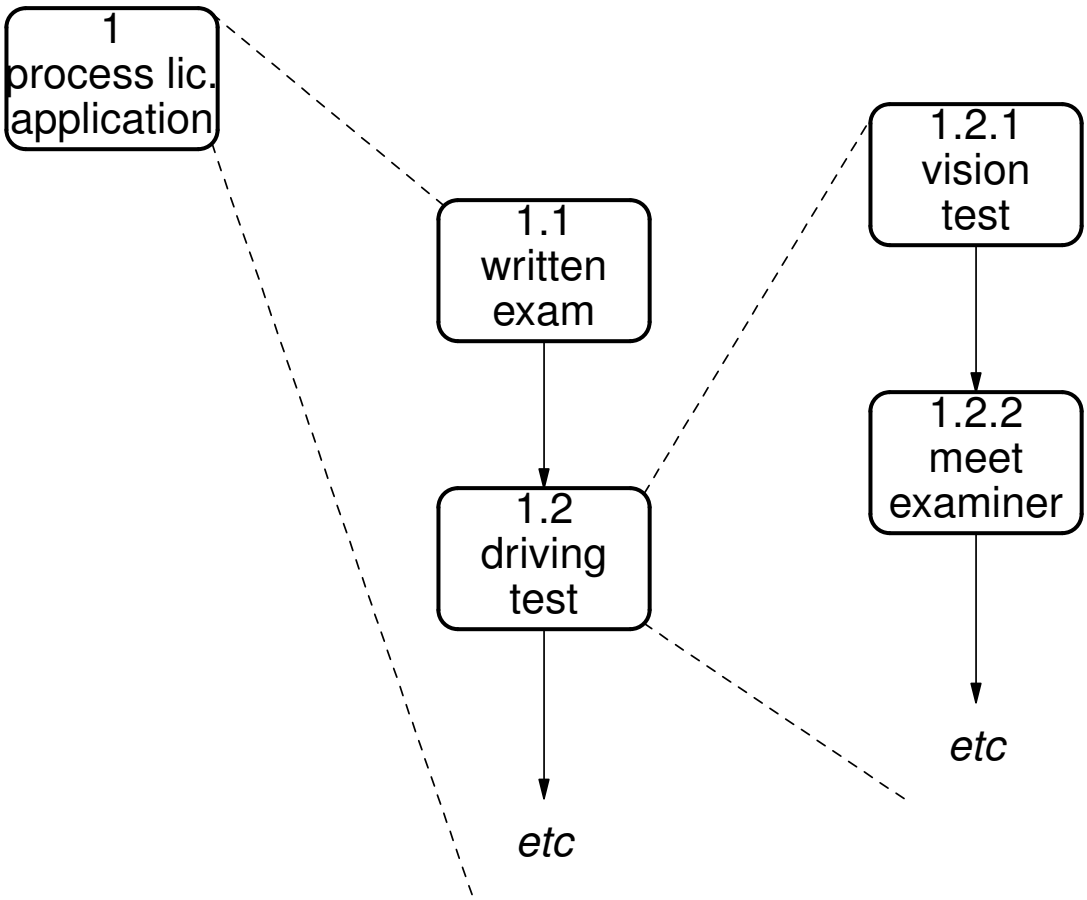
Operation of a “License Branch”





© Copyright 1906-10-9, E Robertson

Example of Notation - Blowup



- These should be three distinct diagrams.

© Copyright 1906-10-9, E Robertson

Physical Flow

Context Diagram

- Shows how project fits into larger environment
- Physical flow/communication
  - ◊ paper documents
  - ◊ people
- Showing sources & sinks essential
- Show a (sub)process only if it's externally visible

Other Flow Models

- Many similar methodologies model other kinds of flows
- Nodes correspond to process, location, state, *etc.*
- Examples:
  - ◊ production line
  - ◊ queuing models

© Copyright 1906-10-9, E Robertson

Levels of Detail

Context Diagram

- Main focus is external environment

Level 0 Diagram

- Main focus is top-level internals
- Often omit sources & sinks here and below but still show flow
- Possibly redundant with context diagram (especially for simple projects)

Level *i* Diagrams

- Provide details for nodes at higher level
  that is, level *i*-1
- Use “Dewey decimal” notation to clarify relationships
  node 2.3 at level 1 explodes to 2.3.1, 2.3.2, 2.3.3, . . . at level 2
  viewed as a hierarchy:

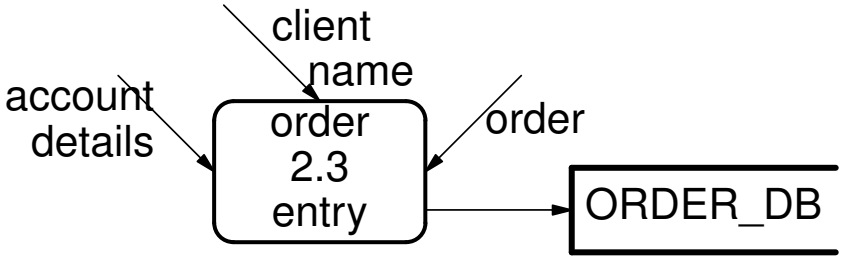
```
2
    . . .
     2.3
         2.3.1
         2.3.2
         2.3.3
       . . .
```

© Copyright 1906-10-9, E Robertson

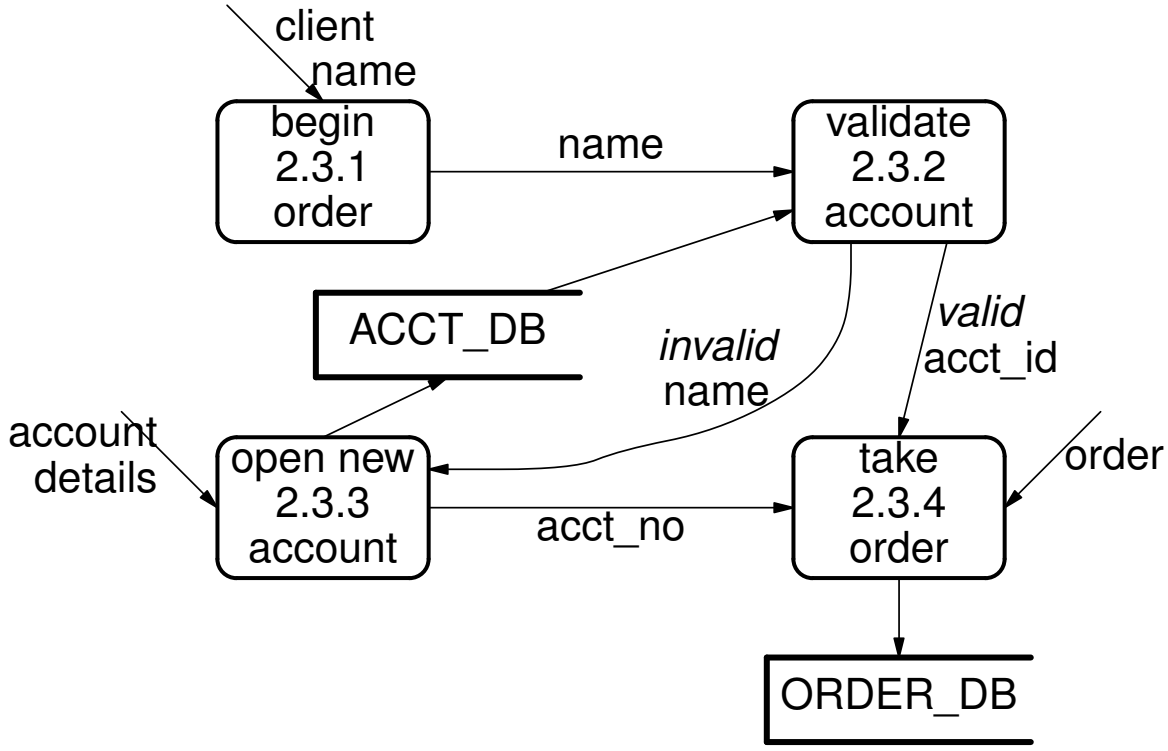
Levels and Perspectives

View of processes depends upon position:

- process as seen by manager



- process as seen by order-entry clerk



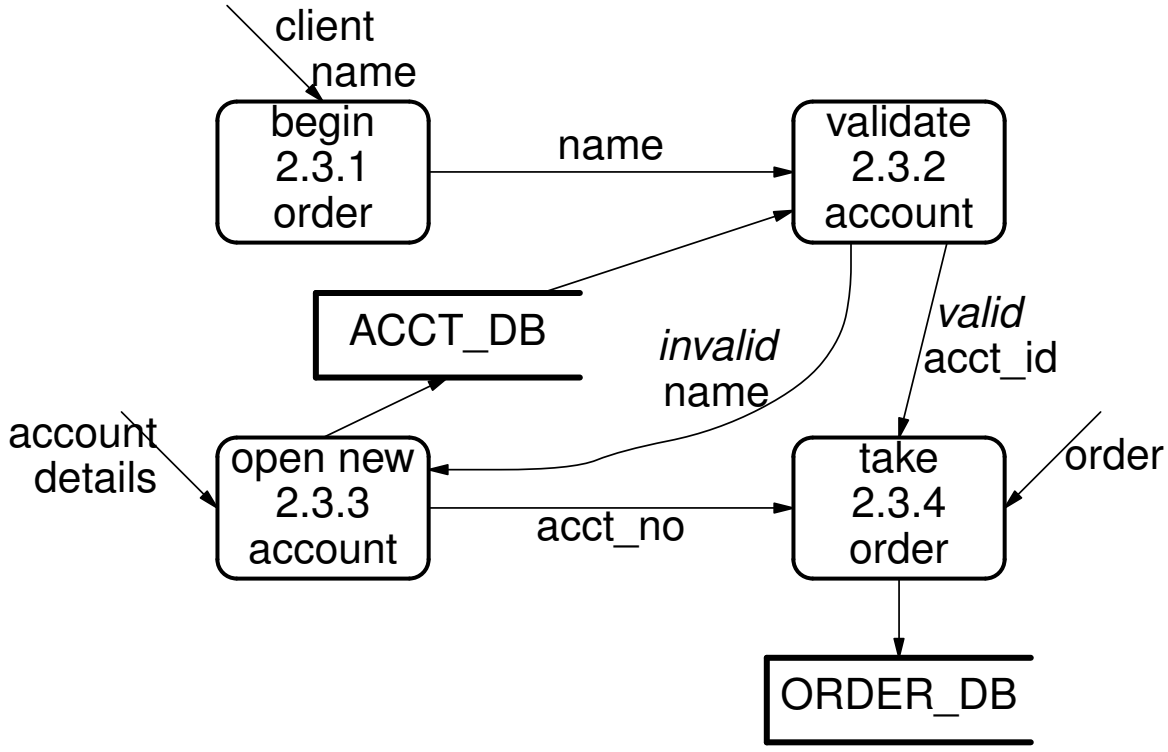
© Copyright 1906-10-9, E Robertson

How Far to Expand?

- a process almost always *can* be expanded into subprocesses

? *should* it be expanded?

- remember the intent is to account for *information flow*
- therefore detail of ‘order entry’ necessary because ‘account details’ are conditional

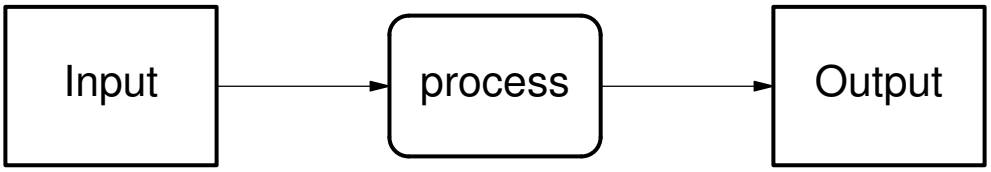


- further expansion may be required
  if ‘order’ is complex, ‘order entry’ should expand

© Copyright 1906-10-9, E Robertson

The Trivial DFD

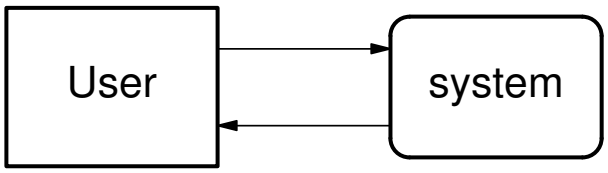
- *Every* information system has the same simplistic model:



- If that's all you have to say, don't bother!

- The purpose is to *identify* and *distinguish* specific sources and sinks.

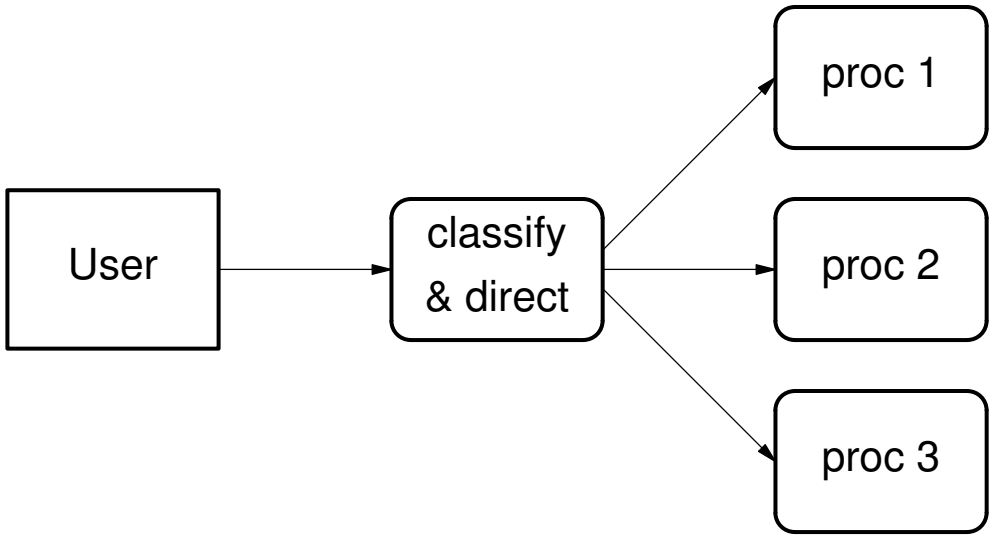
- In interactive systems, the trivial model is:



© Copyright 1906-10-9, E Robertson

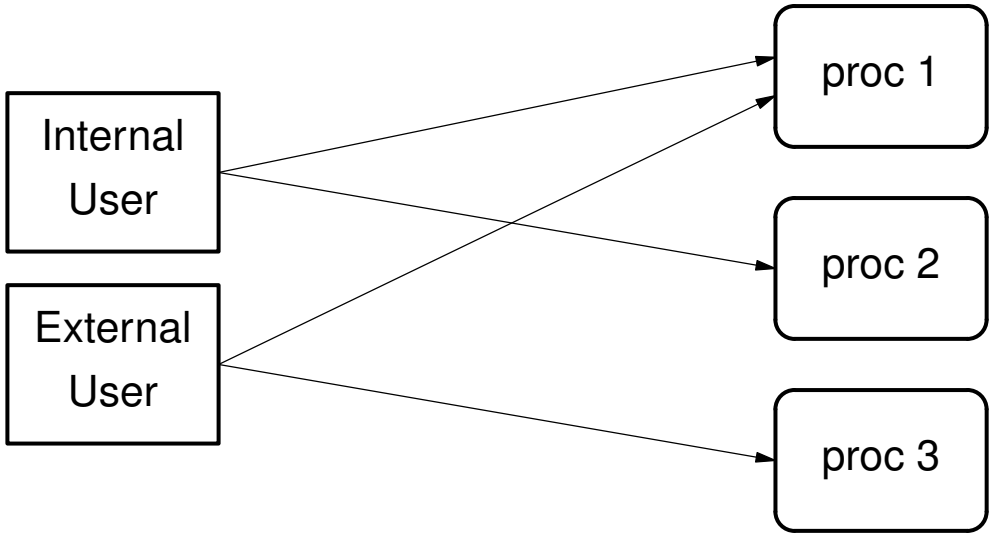
Avoid Monolithic “User”

Mistake recommended by Mynatt & others:



Correct by

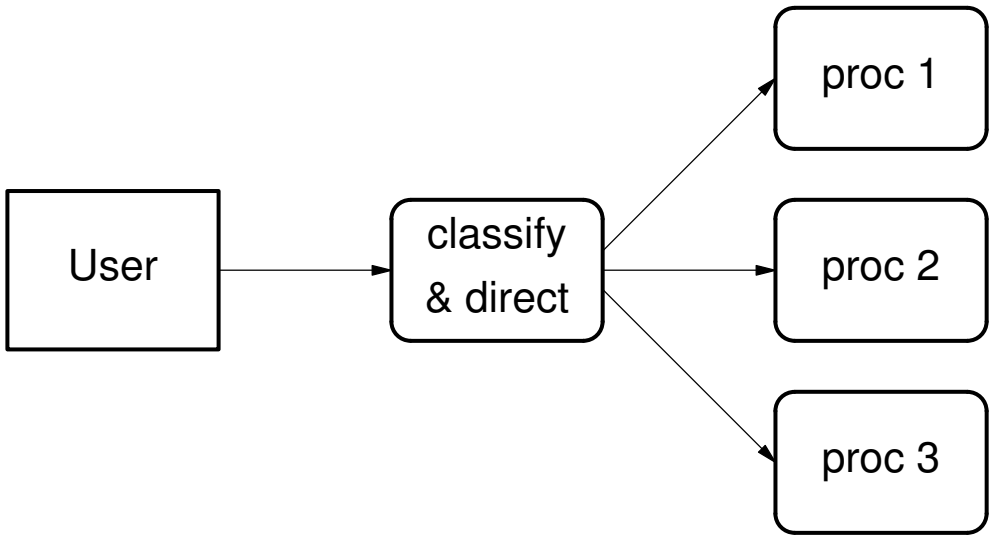
- distinguishing user *roles*
- separating user input at source



© Copyright 1906-10-9, E Robertson

Don't Confuse Model and Implementation

Recall previous mistake:



This is an attempt to capture menu selection in DFD.

Flow is inherently parallel

- DFD cannot model "focus" or state
- program "flow charts" couldn't answer "*What is flowing?*"

© Copyright 1906-10-9, E Robertson

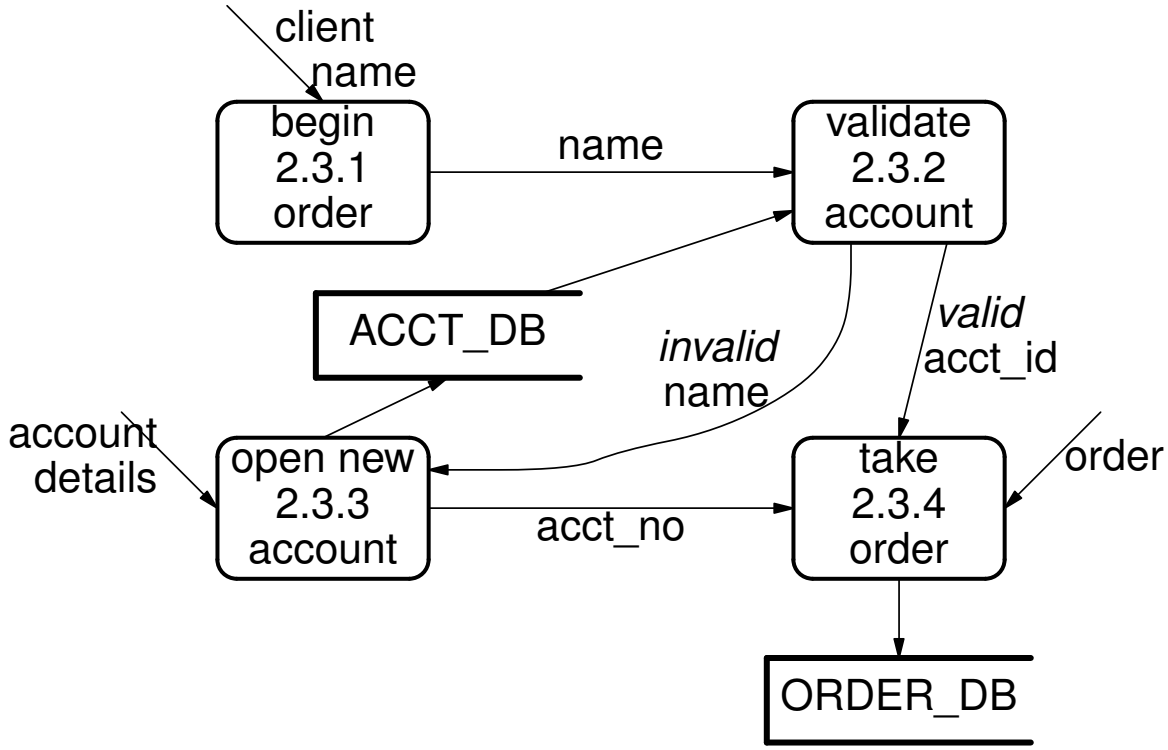
Datafbw Guides User Interface

DFD does look forward to implementation

As much as possible, interface should

- guide user through steps
- avoid presenting meaningless choices

Recall previous ‘order entry’ example



⇒ automatically move to ‘open new account’ if account is invalid

© Copyright 1906-10-9, E Robertson

Wrapup

- Model fbw

  Implement processes

Thus DFDs are an excellent illustration of modeling as the *bridge* between specification and implementation.

© Copyright 1906-10-9, E Robertson