Lecture # 36

Functional Modeling

Recap

- We had talked about object-oriented static modeling in quite detail
- We had developed a OO static model of the Banking System to gain understanding

Modeling Techniques

- Object-oriented modeling
  - Static and dynamic modeling
- Functional modeling
- Dynamic modeling

Today's Topics

- Today we'll start our discussion on functional modeling or function-oriented modeling
- We'll apply this to our Banking System case study also
- First, let us review the three principles of modeling

Principles of Modeling

- Partitioning
- Abstraction
- Projection

Function-oriented Modeling

- In function-oriented modeling, a hierarchy of functions (also known as processes, transforms, transactions, bubbles, and activities) is created
- At the root of the hierarchy is the most abstract function, while the leaf nodes of the hierarchy are least abstract

Function-oriented Modeling

- Function-oriented modeling is based on the concept of functions or processes, so they become the most important element in this approach
- The functional model describes computations within a system, i.e., what happens

Function-oriented Modeling

- What is a function or a transform or process?
- Each function is a sequential activity that potentially may execute concurrently with other functions
- The functional model specifies the result of a computation without specifying how or when they are computed

Types of Functions

- Asynchronous Function
- Asynchronous State Dependent Function
- Periodic Function
- Periodic State-Dependent Function

Asynchronous Function

- An asynchronous function can be activated by another object or function to perform some action

Asynchronous State-Dependent Function

- An asynchronous state-dependent function is usually a “one-shot” action, which is executed during a transition from one state to another state
- This function is activated by a control transformation

Periodic Function

- A periodic function is activated at regular intervals to perform some action
- The frequency with which a specific function is activated is application dependent

Periodic State-Dependent Function

- A periodic function is activated at regular intervals to perform some action
- The frequency with which a specific function is activated is application dependent
- This function is activated by a control transformation

Functional Modeling

- Non-interactive programs, such as compilers, usually are used for computations. The functional model is the main model for such programs
- On the other hand, databases often have a trivial functional model, since their purpose is to store and organize data, not to transform it

Functional Modeling

- Fundamental to most of techniques used for functional modeling, is some combination of data flow diagrams and data dictionaries

Data Flow Diagrams

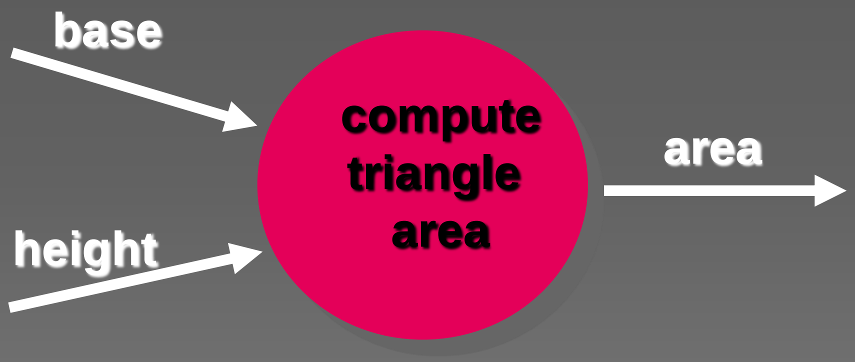
- Data flow diagrams are composed of data on the move, transformations of data into other data, sources and destinations of data, and data in static storage
- Data flow diagrams show the flow of data through a system

Observations About DFDs

- All names should be unique
- A DFD is not a flow chart
- Suppress logical decisions
- Do not become bogged down in details

Data Flow

Data flows through a system, beginning as input and be transformed into output.



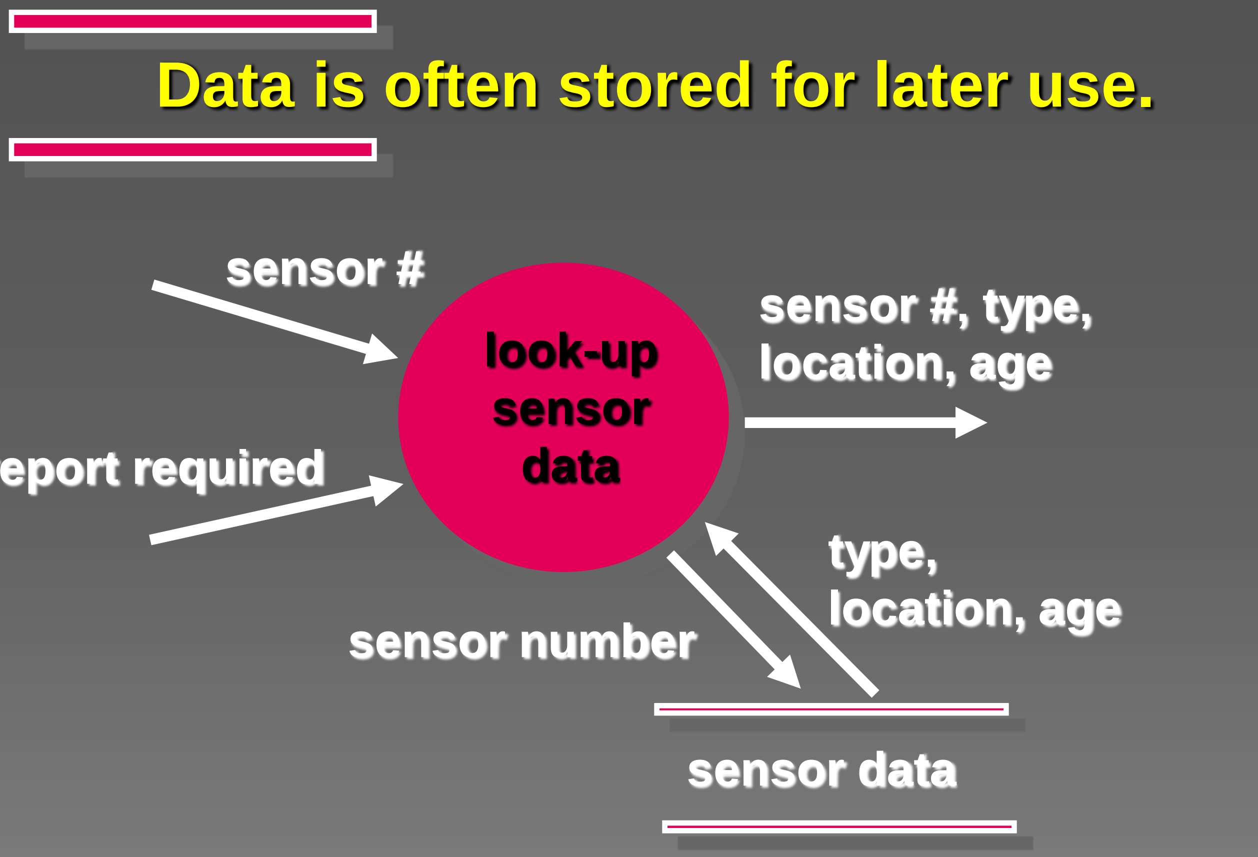
base

height

compute triangle area

area

Data Stores


Data is often stored for later use.

Data Flow Diagramming: Guidelines

- All icons must be labeled with meaningful names
- The DFD evolves through a number of levels of detail
- Always begin with a context level diagram (also called level 0)

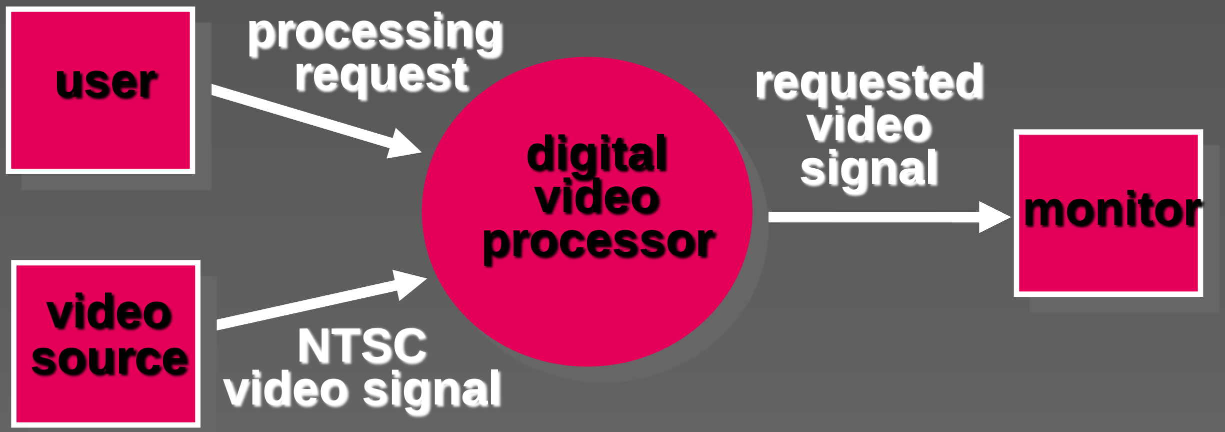
Data Flow Diagramming: Guidelines

- Always show external entities at the context-level or level zero 0
- Always label data flow arrows
- Do not represent procedural logic

Constructing a DFD—I

- Determine data objects and associate operations
- Determine external entities
  - Producers of data
  - Consumers of data
- Create a level 0 DFD

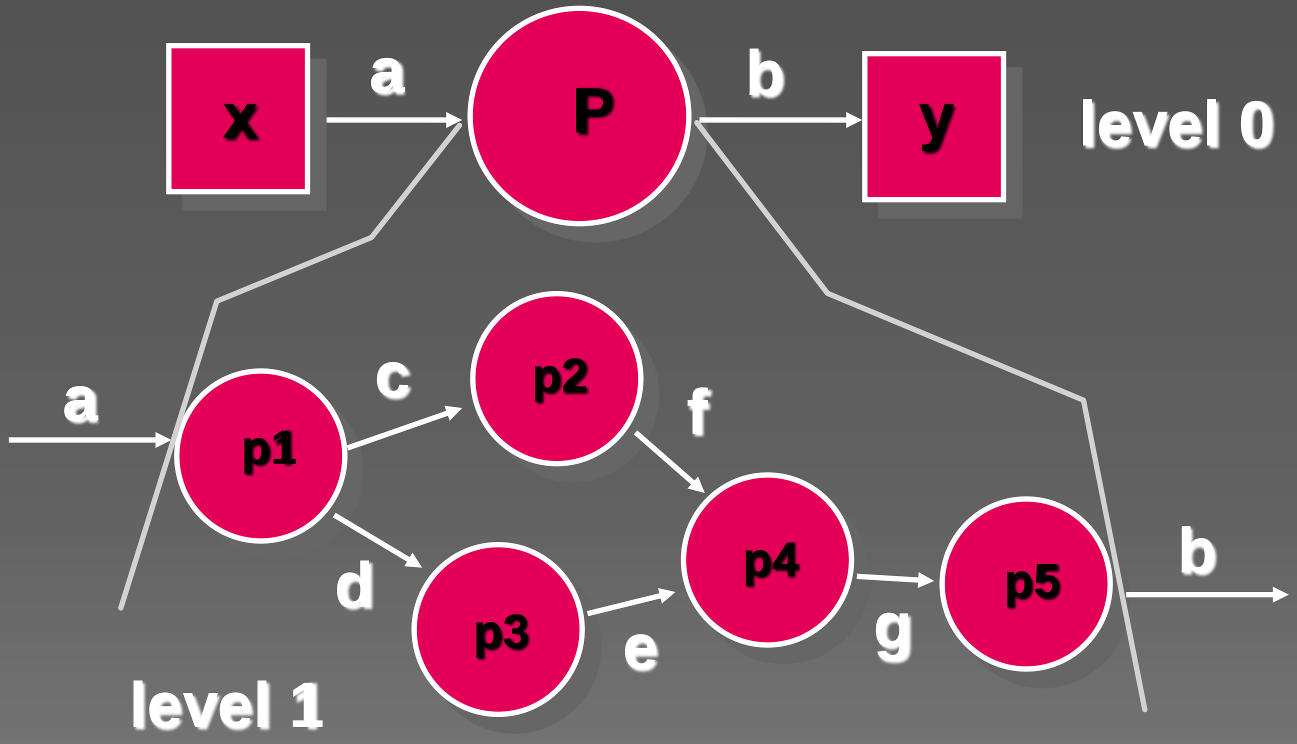
Level 0 DFD Example



Constructing a DFD—II

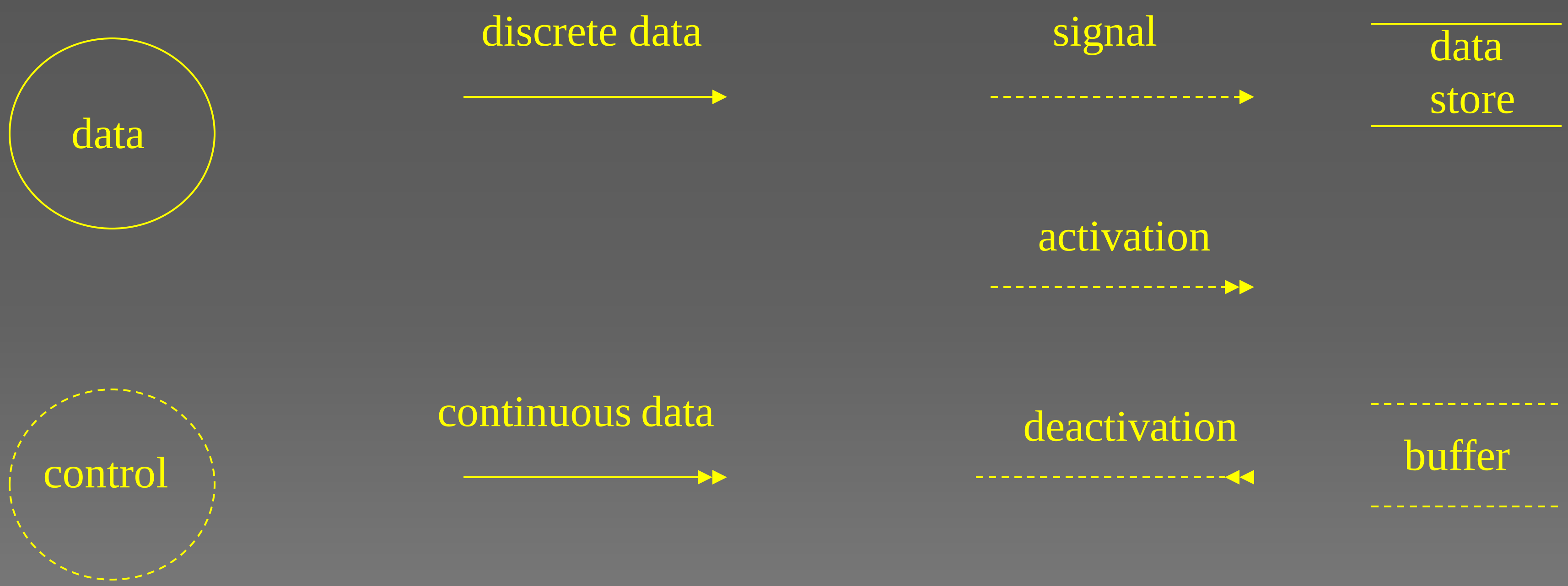
- Write a narrative describing the transform
- Parse to determine next level transforms
- “Balance” the flow to maintain data flow continuity
- Develop a level 1 DFD
- Use a 1:5 (approx.) expansion ratio

The Data Flow Hierarchy



Ward Notations for DFD Extensions

transformations data flows event flows stores



data

control

discrete data

continuous data

signal

activation

deactivation

data store

buffer

Ref: Software Requirements by Alan Davis, PH 1993. Copyright 1986 IEEE

Description of Ward's Extensions

- Discrete data
  - A single item of data
- Continuous data
  - A source of constantly available and perhaps continuously changing data
- Signal
  - Reporting an event

Description of Ward's Extensions

- Activation
  - A direct overt action to initiate another process
- Deactivation
  - A direct overt action to stop another process

Description of Ward's Extensions

- Data transformation
  - This is an example of a process, which is used to transform data values
  - The lowest-level functions are pure functions without side effects
  - A process may have side effects if it contains nonfunctional components, such as data stores or external objects

Description of Ward's Extensions

- Control transformation
  - Control transformations are modeled in the dynamic model, as they deal with sequencing and control issues

Data Dictionaries

- Data dictionaries are simply repositories in which to store information about all data items defined in DFDs

Contents of Data Dictionaries

- Name of the data item
- Aliases
- Description/purpose
- Related data items
- Range of values
- Data flows
- Data structure definition/form

Summary

- Partitioning, abstraction, and projection are three important principals of modeling
- DFDs are an important Functional modeling technique which describes what the system does in terms of functions