CS360
Business Information Systems Analysis and Modeling
Process Modeling: Data Flow Diagrams
Modeling

the capture of a subset of system characteristics relevant to a level or understanding or detail
System Modeling

- **Data modeling**
  - Entity-Relationship Diagrams (ER, EER)

- **Process modeling**
  - Modeling how data flows between and is transformed within business processes
  - Data Flow Diagrams (DFDs)

- **Logic modeling**
  - Modeling processing logic and timing of events within processes
  - Structured English, decision tables, decision trees, state-transition diagrams and tables
Data Flow Diagramming

Process Model Specification Tool by (Gane & Sarson) focusing on:

- sources of information
- destinations of information
- paths of information flow
- information transforming processes
- repositories of data within the system

Input

Output
**DFD Symbols**  
Gane & Sarson

- **External Entity**
  - originator of data or receiver of information
  - aka “sources” and “sinks”

- **Data Store**
  - a repository for data in the system

- **Process**
  - procedure that operates on data
  - collecting, sorting, selecting, summarizing, analyzing, and reporting
  - abstraction level (general vs. detailed)

- **Data Flow**
  - conduit of data/information between
  - between external entity and process
  - between process and data store
  - between processes
External Entity (sources and sinks)

- External origins and destinations of data
- Every source and sink is a “black box” from the perspective of the system
- Cannot access data stores directly without processes
- They are often organizations, organizational units, individuals, or other systems
- Name with nouns

Diagram:
- Student
- Clerk
- Faculty Member
- General Manager
- Internal Revenue Service
- Customer
Processes

* Describes the actions that are taken to transform, store, or distribute data

* Name with verbs

* Can be either computerized, manual, or both

* For example, Receive customer order, Process customer complaint, Distribute daily sales order report

- Retrieve Account#
- Calculate Sales Tax
- Approve Credit
- Order Catalog Item
- Determine Sale Price
Identifying Processes

- One way to do this: Analysis of Business Events
  - External events – an outside stakeholder interfacing with the system (e.g., customer placing an order; arrival of supplier’s shipment; sales rep calling customer)
  - Decision events – human decision is needed to evaluate input from multiple sources (e.g., hiring decision)
  - Time-based (temporal) events (daily, weekly, monthly, etc. actions)
  - State events (e.g., drop of inventory levels below a certain level triggers action)
Data-Flow

* Data that is in motion
* Data moving together
* Data flow often consists of several components
* Name with nouns
* Data can flow using various media: paper forms, conversation, computer network, electronic media
**Data Stores**

- Data at rest
- Data about people, things, or events
- E.g., customers, sales transactions, discount percentages, etc.
- Name with nouns
- Various physical storage media used: databases and files on computer disks, filing cabinets, desk drawers, notebook (method irrelevant for logical DFDs)
DFD's Bottom Up

- Develop system narrative
- Identify system actions
- List tasks in order
- Keep only data transformations
- Identify cohesive tasks
- Collect all actions under these tasks
- Draw an Input Process Output chart for each cohesive task
DFD’s Top Down

* “What’s the primary task of the system?”
* “What tasks does that break down into?”
* Repeat the decomposition until the tasks are “trivial”
Context Diagram

* Shows the entire system as a single process

```
<table>
<thead>
<tr>
<th></th>
<th>Data Flow Line</th>
<th>Data Flow Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;A&quot;</td>
<td>Data Flow Line</td>
<td>Data Flow Line</td>
</tr>
<tr>
<td>&quot;B&quot;</td>
<td>Data Flow Line</td>
<td>Data Flow Line</td>
</tr>
<tr>
<td>&quot;C&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;D&quot;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

The Whole System
Context Diagram (Level zero)

* “The context level diagram is intended to identify the system boundary with regard to its relationship to any source or sink entities that may interact with it. As such, the context diagram contains only one process, labeled with the name of the system and assigned a zero as its identifier.” (Marakas, 2001, p. 127)

* “An overview of an organizational system that shows the system boundaries, external entities that interact with the system, and the major information flows between the entities and the system.” (Hoffer & al, 2002, p. 243)
The context diagram depicts the most general view of system connections to the “outside world.”
Decomposition “drilling down”

* Dividing a system/process into subsystems/processes that all perform a well-defined function
* Several levels of decomposition in a large system (subsystems/processes can be further divided into smaller units)
* Continues until no subprocess can logically be broken any further
"Telescoping Levels of Abstraction"
“Balancing” DFD’s

- Inputs to and outputs from a process in a data flow diagram should remain the same when the process is decomposed.
- A data flow can be split into component data flows that are input for different subprocesses, but the content of the data flow has to remain the same.
Identifier in DFD’s

* The context level diagram gets the number 0 (zero)
* Level-0 diagram processes are numbered from 1.0 to n.0 (where n = the total number of processes)
* In Level-1 diagrams, the processes are numbered with the number of the Level-0 diagram process that is begin decomposed and a running number for each of the processes (E.g., 2.1, 2.2, etc., if the Level-1 represents the process 2.0 from Level-0)
* The data stores are identified with a letter-number combination; most commonly used letters are S and D (E.g., S1, S2, etc.)
DFD Tree

“Context Diagram Level”

Level Zero

Level One

Level Two

1. Process Name
2. Process Name
3. Process Name
4. Process Name
5. Process Name

2.1. Process Name
2.2. Process Name
2.2.1. Process Name
2.2.2. Process Name
2.2.3. Process Name

5.1. Process Name
5.2. Process Name
5.3. Process Name
DFD Syntax

- A: No process can have only outputs
- B: No process can have only inputs
- C: Processes are named with verbs
- D: Data cannot move directly from one data store to another data store
- E: Data cannot move directly from an external source to a data store (a process is required)
- F: Data cannot move directly from a data store to an external sink (a process is required)
DFD Syntax . . .

- G: A data store is named with a noun
- H: Data cannot move directly between two external entities (e.g., from a source to a sink)
- I: A source and a sink are named with a noun phrase
- J: A data flow is unidirectional
- K: A fork means that exactly the same data flow goes to two different destinations
DFD Syntax . . .

- **L**: A join means that exactly the same data comes from two different locations to the same destination (very rare)

- **M**: Loops are not permitted

- **N**: A data flow is named with a noun phrase. Several nouns can be used as part of the same label if several flows move together as a package
Shorthand DFD Syntax Rules

External Entity → Data Flow Line → External Entity

External Entity → Data Flow Line → Data Store

Data Store

Process #

At Least One!!

At Least One!!

At Least One!!

At Least One!!
“Logical” vs. “Physical” DFD’s

**Logical DFD**
- Ignores implementation specifics
  - computer configuration
  - data storage technology
  - communication or message passing methods
- Focuses on the functions performed by the system
  - data collection
  - data to information transformation
  - information reporting

**Physical DFD**
- Reports/Prescribes implementation specifics
  - data entry devices
  - data storage capacities and technologies
  - network and data transfer modes and protocols
- Describes the information processing “hardware”
  - processing modes (batch, online, interactive)
  - user interface details (forms, report media)
  - connectivity of information processing to user “world”
Producing DFD's

- Start with “Top-Down” or “Bottom-Up”
- Sketch some “system context” ideas
- Identify “next level of detail” process steps below the “previous level”
  - Define inputs from ones available at previous level
  - Define any data stores needed to hold “working” data
  - Define the processes feeding to the previous layer outputs
- Repeat steps 2 thru 4 until process blocks are trivial to describe

CASE and drawing tools can take a lot of the pain out of “reworking” and polishing any DFD with more than a few processes and more than one level. That’s why it’s worth learning a CASE tool like Visible Analyst or Enterprise Architect!!
Guidelines...

- Identify external entities (source and sinks; sets system boundary)
- Identify inputs and outputs for each entity
- Ignore timing considerations -- draw system as if it never started or will never stop
- Iterate and refine -- usually takes at least 3 drafts of each DFD for each level
- Decompose and balance processes until they are highly cohesive
- Try to keep diagrams to 7-12 symbols
Guidelines...

- **Completeness**
  - Have all components been included and fully described?

- **Consistency**
  - Between levels

- **When to stop?**
  - Once you have reached the level of details suitable for primitive DFDs
Some Additional DFD Comments

- DFDs tend to be relatively unstable
- DFDs alone are not enough to describe the requirements at a sufficient level
- Logic modeling tools required to describe the internal logic of the primitive level DFDs
- Some authors have suggested that use cases (a requirements analysis tool developed for object-oriented analysis) can be used together with DFDs
  - For example, utilization of use cases to describe the behavior within low-level processes