

Data flows: Note on Data-Driven Process Modeling

[Introduction](#)[A Focus on Data](#)

Data flow diagrams (DFDs)

[Common DFD mistakes](#)[Summary/next steps](#)

Data Flow Diagramming

In this course we will use data flow analysis to understand patterns of data movement within processes. DFDs provide one technique for isolating the data stores used by a process and the major data entities that those stores contain.



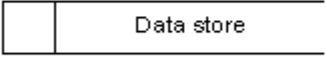
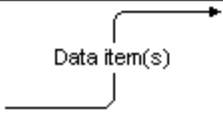
Data flow diagrams (DFDs) offer a graphical technique for summarizing the movement of data between the processing steps that occur within a business process. They isolate the collections of data, or data stores, which accumulate during a process, and identify the sources of data that arise outside process boundaries. Some key characteristics of data flow diagrams are:

- **Two-dimensional summary.** DFDs offer a way to summarize the data flow characteristics of a process on a single page. As such they can provide a useful and concise summary of system-related (e.g., data-driven) process attributes.
- **Completeness.** DFDs offer a way to check the completeness of your process model, particularly as regards your understanding of the data that would be required by an information system (e.g., is all the data that would be needed for input actually available? Does each processing step produce data that could be used by subsequent steps? Is all data generated usable by an information system where necessary?). DFDs can provide a fast way to generate further questions that need to be asked about the process.
- **Processing, not processes.** DFDs refer to "process" steps. It might be more useful to think of DFD "processes" as *processing* steps rather than process activities. In essence, DFDs ask one to refer to the information systems implications of any processing work that occurs during the tasks that comprise a business process. DFD terminology tends to confuse the term "process" in its connotation with *business* process with the term "process" that refers to a *computational* process executing within software (e.g., a software algorithm). Whether this represents the presumption among information engineers that *everything* is just a version of a computational process is a subject for further discussion at a later time (and the winner of the debate receives the Golden Nerd Award); the point here is that it is safer to think of DFD "processes" as *processing* steps.
- **Patterns.** DFDs can provide a shorthand for understanding patterns that exist within the data flows supporting business processes. They can show, for example, where large amounts of data are collected, stored, transferred, generated, used, and delivered. They can highlight areas of potentially extraneous activity, and can suggest process components that do not receive the information support that they deserve (or need).

Symbols used in DFDs

Table 1 summarizes the graphical symbols used in data flow diagrams. The vocabulary used by DFDs is very simple, comprising only four symbols: data flows, processing steps, sources/sinks, and data stores.

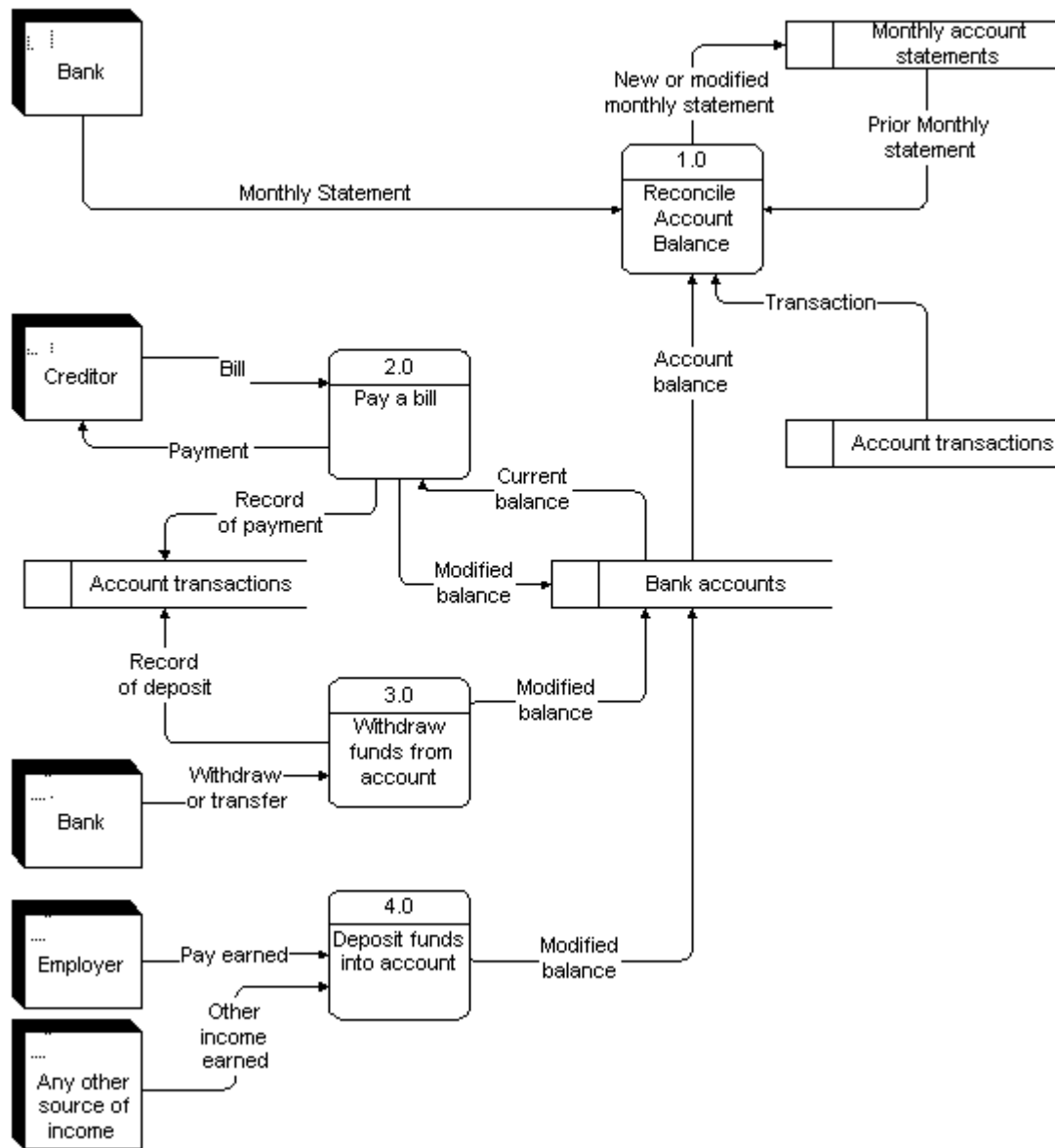
Table 1: Symbols used in Data Flow Diagrams (DFDs)

Symbol	Description
	<p>Source (or Sink). Source is short for “data source” – a source represents any source of data that is identified as outside the boundary of the process that the DFD is modeling. Similarly, a “sink” is any destination for data that is outside the boundary of the process that the DFD is modeling.²</p>
	<p>Processing step. A DFD <i>process</i> represents an activity that processes data; presumably the processing is important enough to play a significant role in the business process that the DFD is modeling. This processing focus is relevant even when the description of the process does not sound like data processing – e.g., “withdraw funds from account” in a DFD describing banking actually refers to the processing that must accompany the withdrawal.</p> <p>Note that the process is labeled with an identifying number – in this case, 1.0. This number is used in developing levels of DFDs that describe increasingly detailed levels of a process (e.g., process 1.0 breaks apart into three processes labeled 1.1, 1.2, 1.3, etc.).</p>
	<p>Data store. A collection of data needed by the business process. A data store is <i>not</i> the same thing as a “database” – instead, it represents a more abstract way of referring to any accumulation of data that is used by the process. The contents of a data store will very likely suggest some databases that could be derived from that store, but data stores tend to be process-specific rather than database-specific. One of the main purposes for which we will use DFDs is to identify entities within the contents of data stores. Those entities then become the basis for database tables developed using ERDs.</p>
	<p>Data flow(s). In DFDs, data flows are represented by arrows. The arrows are labeled with the data that move along the arrow (the arrowhead indicates the direction of movement, just in case you needed to explain that to anyone).</p>

see also [2]

An example can probably provide the best way to understand how a data flow diagram can summarize knowledge about a process. The diagram on the next page illustrates how a systems analyst might use a DFD to describe data flows, processing steps, sources, sinks, and data stores used to describe an individual's relationship with his or her bank. The DFD describes the processing transactions that take place as the individual and the bank jointly manage a set of customer accounts. The process(es) described include most operational aspects of handling the customer's money, including making deposits, funding withdrawals, paying bills (perhaps a service provided by electronic banking), and reconciling account balances. Some of the data stores can be translated almost directly into database specifications (e.g., “Account transactions”). Others show how the notion of data stores can apply equally well to data that are captured in paper form (e.g., “Monthly account statements”). Throughout, the data flow arrows identify data items and groups of data items that move between sources, data stores, and processing steps. Note that processing steps represent those points at which data from sources and data from data stores come together, resulting to some change in specific data items maintained by the bank.

Figure 4: A sample data flow diagram – bank operations



Source: Adapted from Figure 9.3, p. 351 in Whitten, J. L.; Bentley, L. D.; Barlow, V. M. (1994). *Systems Analysis and Design Methods* (Third Edition). Burr Ridge, IL: Irwin.

What are some of the advantages of using DFD analysis? Here are several:

- **Data flows and process consequences.** Note how this representation of the data characteristics of banking operations enables us to start at any point in the operation (e.g., deposits, withdrawals, or bill payment), and follow the consequences of that activity through to the point where all appropriate account balances have been adjusted and reconciled. Wherever we start in the process, we can understand the processing steps that the bank would need to take to complete the relevant transaction(s) and to inform its constituents of the results.
- **Data inputs and outputs.** The DFD also makes it possible to understand what data are needed to provide appropriate inputs to any processing step. If, for example, we were to build an information system to support this individual's banking activities (in the days before Quicken and/or Microsoft Money), we would need to understand exactly what data items are represented by data flows such as "Monthly Statement", "Pay earned", "Withdraw or transfer", and other arrows shown in the diagram.
- **Simplifying complexity by isolating process components.** Note how the DFD would make it easier to capture the detail of such data flows. By isolating "Withdraw or Transfer" within the larger scheme of the banking process, the DFD makes it possible to consider the details of the data items included in this flow without reference to the flows affecting other processing steps. All of the flows

affecting withdrawals (e.g., processing step 3.0, "Withdraw funds from account") are isolated as entering or leaving processing step 3.0. At the time that DFDs were developed, this shift towards modularizing data flows and processing elements represented a major step forward in enabling systems analysts to add useful structure to process representations rapidly and easily.

Next: [Common DFD mistakes](#)

@ 1999 Charles Osborn