Chapter 8
Structuring System Requirements: Process Modeling

Chapter Overview

Chapter 8 introduces students to several process modeling techniques for representing business processes. Although this chapter focuses primarily on data flow diagramming, brief overviews of functional hierarchy modeling and Oracle's process modeler are given.

After a brief introduction to process modeling, data flow diagramming techniques are introduced in a section called "Data Flow Diagramming Mechanics." This section demonstrates the basic DFD symbols, definitions, and rules. The authors use the Gane and Sarson symbol set throughout the book, and these symbols are explained in this section. Hoosier Burger, the food ordering system first mentioned in Chapter 2, is used to illustrate basic data flow diagramming concepts. This section also includes explanations of decomposition and balancing.

Chapter 8's third major section introduces four different types of DFDs: current physical, current logical, new logical, and new physical. Hoosier Burger's inventory control system (which is manual) is used to illustrate the first three types of DFDs. Current practice in using DFDs indicates that very little time should be spent on the current physical DFD.

The fourth major section in this chapter, "Using Data Flow Diagramming in the Analysis Process," introduces guidelines for drawing and using DFDs. This is different from the mechanical rules presented earlier. Topics include completeness, consistency, timing, iterative development, primitive DFDs, and analyzing DFDs for system inefficiencies and discrepancies among DFDs that are supposed to be modeling the same system. A Hoosier Burger example helps illustrate these guidelines.

The "Oracle's Process Modeler and Functional Hierarchy Diagrams" section introduces students to two other process modeling tools. These tools are Oracle Designer's process modeler and functional hierarchy modeling, a tool found in several CASE products. In this section, the authors show how to prepare basic process models and functional hierarchy diagrams. Additionally, the authors compare and contrast Oracle's process models to data flow diagramming.

In the last section of this chapter, the authors' overview process modeling for Internet-based electronic commerce applications. As they explain, process modeling for Internet-based electronic commerce applications does not differ from more traditional applications development projects.

Instructional Objectives

Specific student learning objectives are included at the beginning of the chapter. *From an instructor's point of view, the objectives of this chapter are to*:

1. Show how to logically model processes with data flow diagrams.

2. Teach students data flow diagram symbols and the mechanical rules necessary to create accurate, well-structured process models.

3. Show students how to decompose data flow diagrams into lower-level diagrams.

4. Illustrate the concept of balanced DFDs.

5. Explain and demonstrate the differences among the four levels of DFDs: current physical, current logical, new physical, and new logical.

6. Illustrate how data flow diagrams are used as tools to support systems analysis.

7. Explain and stress the importance of the DFD guidelines: completeness, consistency, timing considerations, the iterative nature of drawing DFDs, and drawing primitive DFDs.

8. Discuss and illustrate Oracle Designer's process modeler.

9. Discuss and illustrate functional hierarchical modeling.

10. Discuss processing modeling for Internet-based electronic commerce applications.

Classroom Ideas

1. Use Figures 8-2 and 8-6 to illustrate the basic DFD symbols and the correct and incorrect ways to draw data flow diagrams. Use Figure 8-3 to demonstrate the problem with trying to include sources/sinks inside the system being modeled.

2. Once you have covered the basics of drawing DFDs, have students complete Problems and Exercises 2 through 4 and 10 as in-class exercises. Once the students have completed these problems, review the problems in class, reinforcing the points that you have made.

3. Use Figures 8-4, 8-5, 8-7, 8-8, and 8-9 in class to teach decomposition. Next, ask students to complete Problems and Exercises 4 and 10; these exercises should be completed in class.

4. Use Figure 8-10 to illustrate unbalanced DFDs.

5. Supplement the chapter material on DFD mechanics, decomposition, and balancing with your own examples, which you can work with your students in class. A good source of such examples is written organizational procedure statements. Modified procedure statements also make good homework problems. See Problems and Exercises 11and 12 for examples. It is best to devote at least one complete class period to working examples. Students can prepare these diagrams outside of class or try preparing them for the first time in class. Many issues arise that are best handled from examples. For example, students often encounter difficulties when:

   - identifying when to show a direct data flow between processes and when to decouple these with a data store (emphasize that data stores allow different processes to work at different rates and at different times)

- deciding what activities to encompass with each process (emphasize the principle of cohesion and the goal of each process being of roughly equal size and complexity)
- distinguishing processes from sinks and sources (emphasize factors such as audience and the ability to change or control in making such distinctions)
- encountering logical inconsistencies or ambiguities in narrative descriptions (emphasize that this is the power of DFDs and the typical interaction between requirements structuring and requirements determination necessary to resolve such ambiguities)

6. The Hoosier Burger inventory control system example demonstrates the differences between current physical, current logical, and new logical DFDs. Working through the entire example in class is an effective way to illustrate the differences in these three types of DFDs. Working through another example from your own experience, or having students come up with their own examples, will supplement the Hoosier Burger example.

7. Use a CASE tool in class to demonstrate ways to model processes other than DFDs. Have students compare and contrast these alternative methods with DFDs.

8. Using a CASE tool that supports DFDs, show in class how the tool provides for decomposition and balancing and how DFDs are linked to the CASE repository. Later, when teaching Chapter 10, you can show how the repository links DFDs and entity-relationship diagrams.

9. Use a CASE tool in class to show how the tool checks for completeness, consistency, and other elements of analysis as discussed in the chapter.

10. Using Oracle Designer's process modeler, show students how to prepare a process model.

11. Using Oracle Designer or another CASE tool, illustrate how functional hierarchy modeling is performed.

12. Have students identify an organization that would benefit from an Internet-based electronic commerce application. Place your students into groups of three or four individuals. Then, have the students prepare a set of data flow diagrams for the Internet-based electronic commerce application. If time permits, ask your students to also prepare process models and/or functional hierarchy diagrams.

Answers to Key Terms

Suggested answers are provided below. These answers are presented top-down, left to right.

3.	Data flow diagram	5.	DFD consistency
1.	Balancing	11.	Level-*n* diagram
10.	Level-0 diagram	13.	Process
14.	Source/sink	6.	Data store
2.	Context diagram	9.	Gap analysis
12.	Primitive DFD	7.	Functional decomposition
4.	DFD completeness	8.	Functional hierarchy diagram

Answers to Review Questions

1. A data flow diagram is a picture of the movement of data between external entities and the processes and data stores within a system. Systems analysts use data flow diagrams to help model the processes internal to an information system and how data from the system's environment enter the system, are used by the system, and are returned to the environment. DFDs help analysts understand how the organization handles information and what its information needs are or might be. Analysts also use DFDs to study alternative information handling procedures during the process of designing new information services.

2. The rules for DFDs are listed in Table 8-2 and illustrated in Figure 8-6. Table 8-3 lists advanced rules for data flow diagramming. Processes cannot have only outputs, cannot have only inputs, and must have a verb phrase label. Data can only move to a data store from a process, not from another data store or an outside source. Similarly, data can only be moved to an outside sink or to another data store by a process. Data to and from external sources and sinks can only be moved by processes. Data flows move in one direction only. Both branches of a forked or a joined data flow must represent the same data. A data flow cannot return to the process from which it originated.

3. Decomposition is the iterative process by which a system description is broken down into finer and finer detail, creating a set of diagrams in which one process on a given diagram is explained in greater detail on a lower-level diagram. Balancing is the conservation of inputs and outputs to a data flow diagram process when that process is decomposed to a lower level. You can determine if a set of DFDs are balanced or not by observing whether or not a process which appears in a level-*n* diagram has the same inputs and outputs when decomposed for a lower-level diagram.

4. The highest level DFD is a context diagram. It represents the system as a single process, with all the related entities and the data flows in and out of the system. The next level diagram, called a level-0, decomposes the one process from the context diagram into two to seven high-level processes. Each process in a level-0 diagram can be decomposed if necessary. Each resulting diagram is a level-1. Should processes in a level-1 diagram be decomposed, each resulting diagram is a level-2 diagram. Each one of these processes would be decomposed on a level-3 diagram, and so on.

5. Current physical DFDs often show the people and technology used within a system, illustrating who does what and the media on which data are stored. Current logical DFDs attempt to show the essence of the system without regard to the actual physical implementation.

6. Detailed DFDs for the current physical system often take a great deal of time to prepare and are often tossed aside as the focus shifts from the current to the new system. By not drawing current physical DFDs, or by drawing only cursory ones, analysts save themselves effort and focus from the beginning on what is really important, the new system.

7. DFDs can be used as analysis tools to help determine the completeness of a system model and a model's internal consistency, as a way to focus on when system events occur through analyzing timeliness, and, through iterative use, to develop and check models. Analysts can study DFDs to find excessive information handling, thus identifying areas for possible efficiencies.

8. You stop decomposing a DFD when the following six conditions are satisfied: (1) each process is a single decision or calculation or a single database operation, such as retrieve, update, create, delete, or read; (2) each data store represents data about a single entity, such as a customer, employee, product, or order; (3) the system user does not care to see any more detail, or when you and other analysts have documented sufficient detail to do subsequent systems development tasks; (4) every data flow does not need to be split further to show that different data are handled in different ways; (5) you believe that you have shown each business form or transaction, computer screen, and report as a single data flow; or (6) you believe there is a separate process for each choice on all lowest-level menu options for the system.

9. Sources and sinks are always outside of the system being considered. They are of interest to the system being considered only because they represent sources of data coming into the system and destinations for data leaving the system. If any data processing occurs inside a source or sink, it should be of no interest to the system being modeled. If the processing is of interest, however, or if the identified source/sink has several inputs and outputs to and from the rest of the system, it may be better considered as an internal process.

10. Context diagrams have only one process that represents the entire system being modeled and show only the data flows into and out of the system. The context diagram also includes sources and sinks, which represent the system's environmental boundaries. There are usually no data stores in a context diagram.

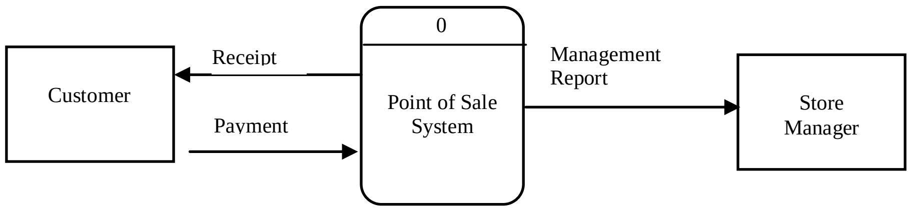
11. Although data flow diagrams are similar to Oracle's process model diagrams, there are several differences. Table 8-4 contrasts these two process-modeling techniques.

12. Both functional hierarchy diagrams and data flow diagrams show a system's decomposition into finer and finer levels of detail. However a FHD's decomposition is shown on the same diagram.

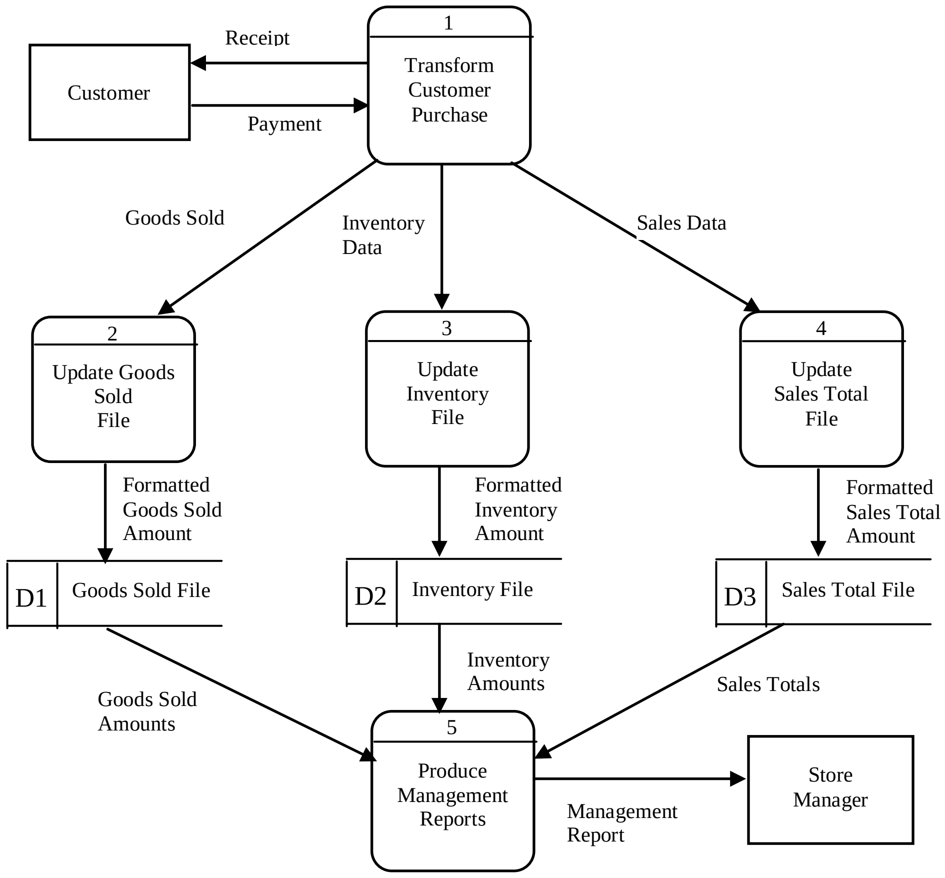
Answers to Problems and Exercises

1. Students can draw their data flow diagrams in several ways, depending on the level of detail they choose to capture. Students should realize that there is not one “right” data flow diagram for this or most other business processes. Relevant data flows include payment information, receipt, goods sold information, and inventory information. Three data stores are a goods sold file, an inventory file, and daily sales total file. Processes include update goods sold file, update inventory file, update daily sales total file, and produce management reports. Sources/sinks include customer and store manager. A sample context diagram and level-0 data flow diagram are provided below. In the level-0 data flow diagram, Transform Customer Purchase, Update Goods Sold File, Update Inventory File, and Update Sales Total File, were selected as processes rather than as sources/sinks because they were deemed to be central to the point of sale process. Point out why these DFDs are balanced.

Retail Store
Context Diagram

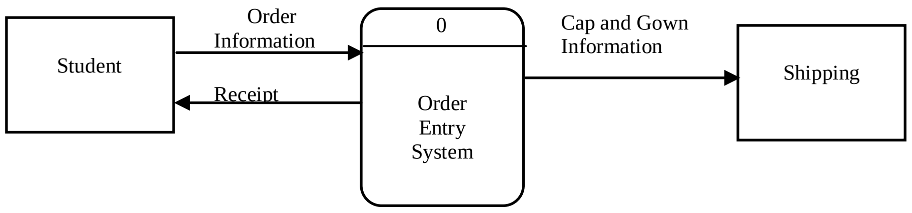


Retail Store
Level-0 Diagram

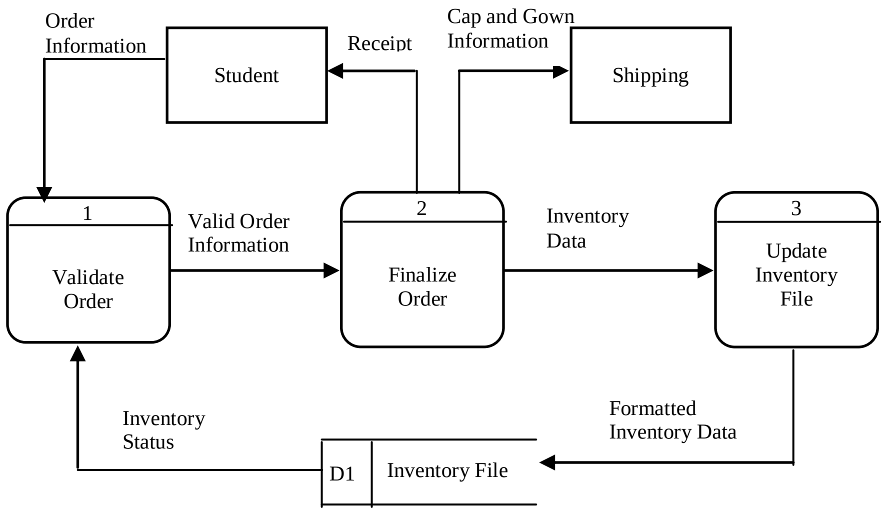


2. Sample context and level-0 data flow diagrams are provided below. As with the previous question, students can draw their data flow diagrams in several ways, depending on the level of detail they choose to capture. Students should realize that there is not necessarily one "right" data flow diagram for this or most other business processes. It is important that the diagrams be balanced, have a clear and purposeful boundary, and obey the rules for drawing DFDs.

Cap and Gown
Context Diagram



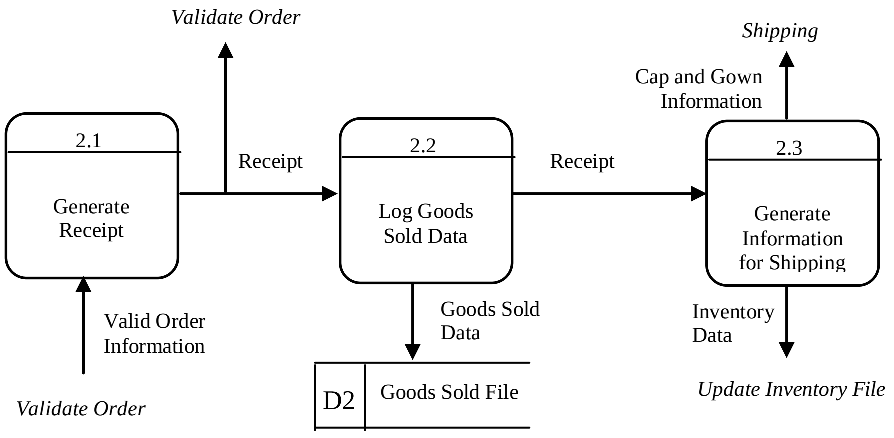
Cap and Gown
Level-0 Diagram



3. Encourage your students to review the rules presented in Table 8-2, Table 8-3, and Figure 8-6 and check each of their data flow diagrams. Alternatively, if the students use a CASE tool to create their data flow diagrams, the CASE tool can automatically check for errors in the diagrams. There are no rule violations in the example DFDs, but we cannot verify that there are no logical problems until we decompose the diagrams to a primitive level. One obvious missing system capability is how to handle invalid orders; typically, processes to handle abnormal conditions, like invalid orders, are shown on primitive or at least low-level diagrams.

4. Your students may choose a variety of situations to use for the *n*th level data flow diagrams for this answer. Basically, students should continue the process of decomposition until they have reached the point where no subprocess can logically be broken down further (i.e., each process meets the definition of a primitive process). See the level-1 data flow diagram for this exercise, which shows a sample decomposition of the process titled Finalize Order from the level-0 data flow diagram provided for Problem and Exercise 2. The (italicized) labels for processes and sources/sinks without borders represent the origin or destination of flows that pass between this subsystem and other system components. Note that the Goods Sold File is a potential black hole, or possibly should be treated as a sink.

Cap and Gown
Level-1 Diagram



5. Some errors and peculiarities in these diagrams include: (1) different names and numbers are used for apparently the same data store on the two diagrams; (2) in the level-0 diagram, the data store, Class Roster, does not have the data flow, Scheduled Classes, flowing into it, rather this data flow connects processes 2 and 3, thus these DFDs are not balanced; (3) Process 1 appears to accomplish nothing since its inflow and outflow are identical; such processes are uninteresting and probably unnecessary; it is possible that this process will become interesting when it is decomposed, where validation and error handling processes might appear; (4) Process 2 does not appear to need Course Request as input in order to perform its function, as implied by its name, and (5) some students may also wonder if Process 3 has input sufficient to produce its output. For example, where are prior class registrations kept so that Process 3 can determine when a course is full?

6. Physical data flow diagrams help you better understand the people and/or computer systems that are used in the overall system's processing. Logical data flow diagrams help you better understand the essence of the system, the data and the processes that transform them, regardless of actual physical form. Further, the new logical data flow diagrams can then show any additional functionality necessary in the new system, to indicate which, if any, obsolete components have been eliminated, and any changes in the logical flow of data between system components, including different data stores. The data flow diagrams for the new physical system can then be constructed with the data flow diagrams for the new logical system as a guide. As discussed in the chapter, experts used to recommend that all four levels of data flow diagrams (current physical, new physical, current logical, and new logical) be constructed because: (1) analysts knew little about the business of the user and needed to develop a detailed current physical data flow diagram in order to understand the business, (2) users were not able to work with a new logical data flow diagram right away, and (3) there is not much work in turning current logical data flow diagrams into new logical data flow diagrams.

7. For data flow diagrams to be complete, all the necessary components of a data flow diagram should be included in the diagram and fully described in a project repository. As described in the chapter, with most CASE tools, the CASE tool's repository is linked to the diagram. Whenever you define (or redefine) a process, data flow, source/sink, or data store on a data flow diagram, an entry is automatically created (or updated) in the repository for that element. Figure 8-18 shows a sample report of the contents of a CASE repository entry. It is this tight linkage between diagrams and the CASE repository that creates much of the value of a CASE tool. Further, you cannot have an entry in the repository that is not on some diagram. The repository is also helpful for enforcing DFD rules; for example, during decomposition, the repository remembers what were all of the inflows and outflows of an exploded process. The repository also keeps track of any split or joined data flows.

8. The various views (e.g., process, logic, data) of an information system each have their own unique characteristics and provide the most relevant information to different information system specialists. This variety is best understood, expressed, and managed by using diagrams and documentation that are specifically tailored for each view of the system. For example, data flow diagrams are useful for capturing the flow of data through business processes, but they are not useful for describing the forms and relationships among data. As information systems become larger and more complex, it becomes even more important to use the right tool and technique to develop each component of an information system. One technique that captured all aspects of an information system model on one diagram or in one notation would likely be too complex for systems professionals to handle.

9. Three major errors in Figure 8-27 are: (1) Process 1.0 (P2) has only inputs, making it a black hole; (2) data flow DF5 should not move directly from source E1 to data store DS1 without first going through a process; (3) data flow DF1 should not move directly from source E1 to sink E2 without first going through a process. Other peculiarities (such as Process 1.0 has label P2 and the data store has only a label, not a number) are only that, not errors.

10. There is a general formatting issue with these DFDs since there are no numbers on processes and data stores, but the student should be able to find logical errors as well. Three particular logical errors in Figure 8-28 are: (1) the data store DS1, not DS2, should be represented on the level-1 diagram; (2) data flow DF3 should be an outflow on the level-1 diagram, and

data flow DF6 should not be on the level-1 diagram; (3) process P1.4.2 has no inputs and is thus a miracle process.

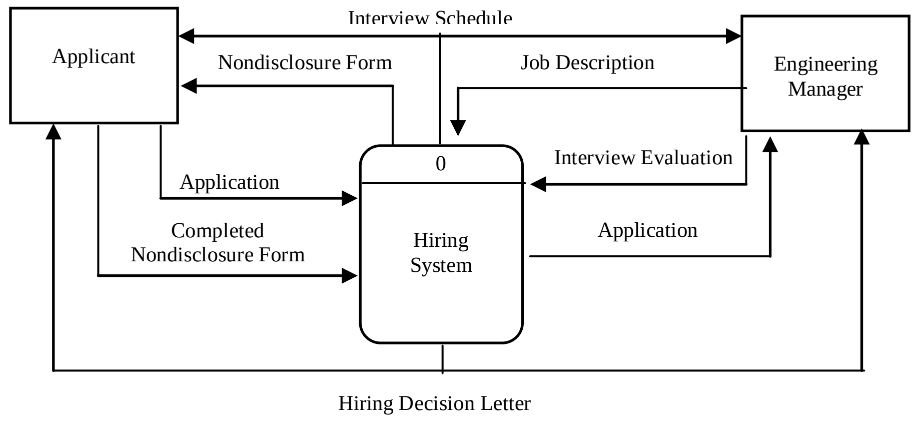
11. The following context and level-0 data flow diagrams represent one way to model the hiring process described in this question. Variations, based on different assumptions, are possible. For example, it is possible to include the processes performed by the Personnel Manager outside the system as a source/sink; in our diagram, we assume the way the Personnel Manager does her work might be studied in more detail, and the way this work is done is subject to change. Also, some students might show Applicant and Hired Employee as separate sources since only Hired Employees provide the information on a nondisclosure agreement.

    Our example assumes that there is always a successful hiring decision (that is, there is no loop between Processes 4 and 3 in the level-0 diagram); we also assume application and job description data flows are accurate and complete, so there is no reason to explode associated processes to show error handling. We also assume that all Hiring Decision Letters result in an accepted hiring offer. You may want to give your students our example answer and ask them to identify additional assumptions we have made, which may be different than what they concluded. Our simplifications mean that it is unlikely that our level-0 diagram needs to be decomposed.

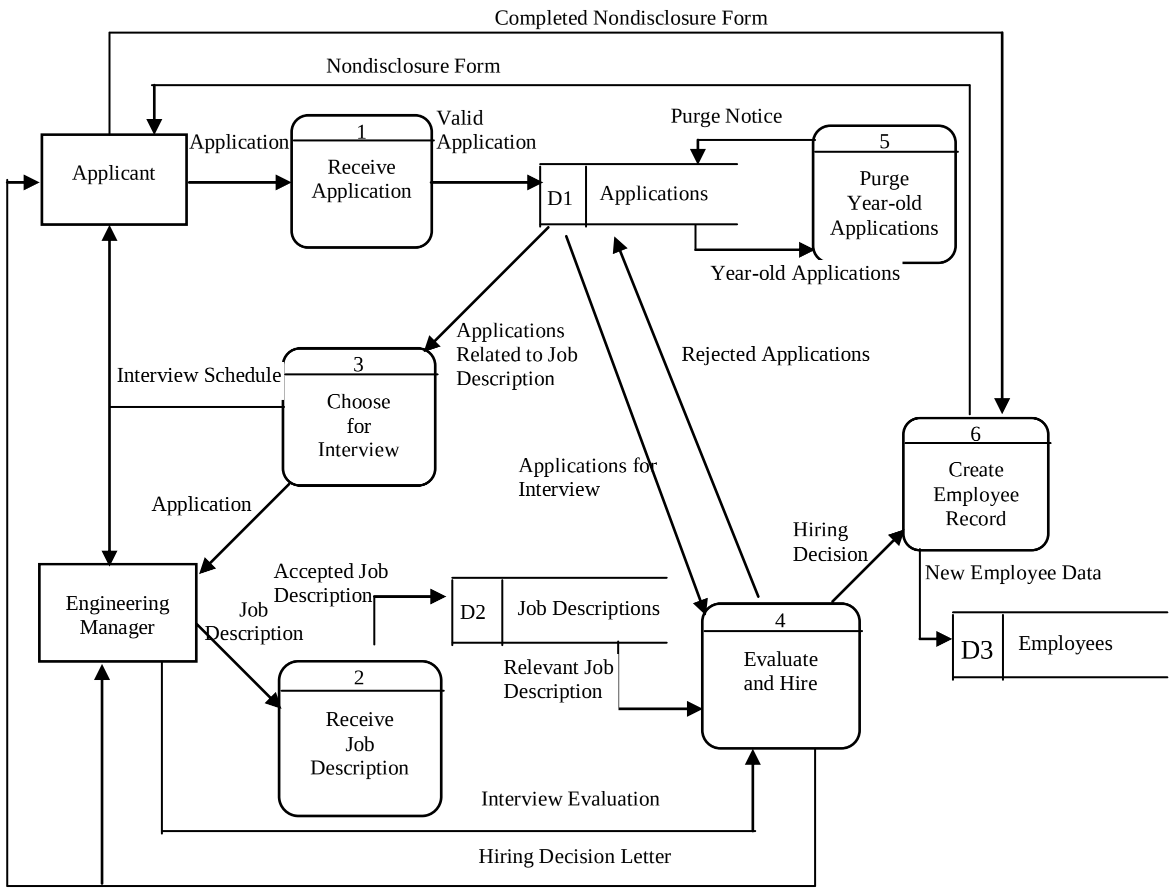
    Our example solution also shows split data flows; be sure to emphasize that a split flow means that the same data flow at the same time, but to multiple destinations. In our example, we also show the Employee data store only on the level-0 diagram, not on the context diagram. Technically, you could show it on the context diagram, but we assume that this is a detail that is necessary to be known only to those looking at level-0 and lower diagrams. A frequent mistake our students have made is to forget to include the purge process (on level-1 or lower diagrams) to get rid of year-old applications. Related to this, since applications are retained for a year and the system operates whenever new jobs are posted (this timing cannot be seen on DFDs), there is a need to have an application data store inside the system (that is, on level-1 and lower diagrams); some students will miss this essential element of the logical system description.

    Students can choose to further decompose the processes; if they do, check that the decompositions are balanced and provide detail needed to show separate processing steps and unique data flows and stores. Some students also make mistakes by trying to use information in the Problem and Exercise that is meaningless for drawing DFDs (for example, there are 500 engineers of different types); this exercise is a good opportunity to emphasize with your students that any given system model, like a DFD, does not model all aspects of an organization, although these facts are relevant (the fact that there are 500 engineers is relevant for physical database design, for example).

Hiring System
Context Diagram



Hiring System
Level-0 Diagram

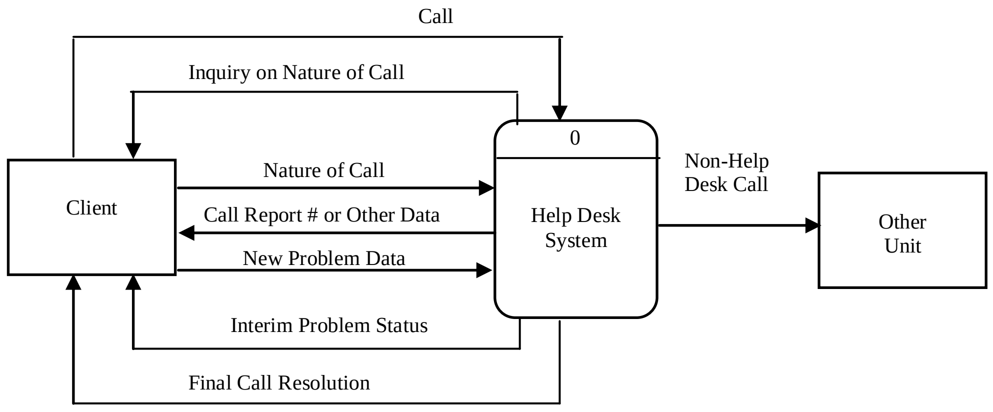


12. The following context and level-0 data flow diagrams represent one way to model the help desk process described in this question. This solution includes the processes performed by the consultants and operators as subsystems within the system rather than as sources/sinks; this adds detail, but allows bottlenecks in these processes to be corrected. Note that data store D1 is repeated in the level-0 diagram, to avoid excessive crossing of data flow lines. Several processes can be exploded further, but the student will need to make many assumptions to do so.

    There are a number of ways that the students can improve this system. For example, with the current system a customer may have to explain their problem and/or question over and over to multiple people: an operator and possibly several consultants. The customer may begin to believe that they are getting the "run-around." One way to avoid this potential problem is to let the initial operator have access to the customer problem database so that when the caller is handed off to a consultant the customer's already opened problem file will go along with him. In addition, the operator could have sufficient information and the option to direct the call to the proper consultant. Alternatively, clients could call the assigned consultant directly on follow-up calls to an initial call for help.

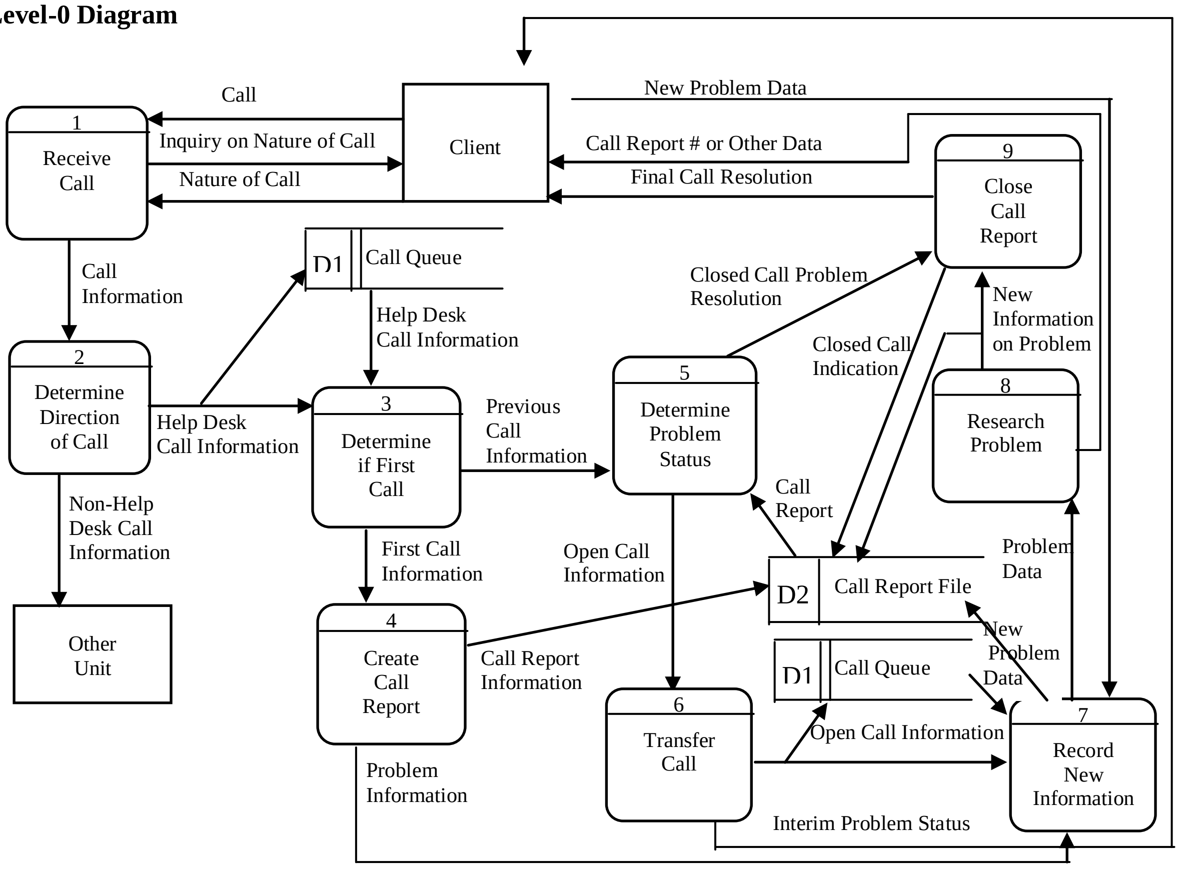
    Ask your students for characteristics of a DFD that imply areas for improvement. Possible answers are: processes that simply collect and pass on information rather than transforming data, collecting the same information into several processes, placing untransformed data into data stores thus causing unknown delays in processing this data, or cycles or loops that have no apparent termination.

Help Desk
Context Diagram



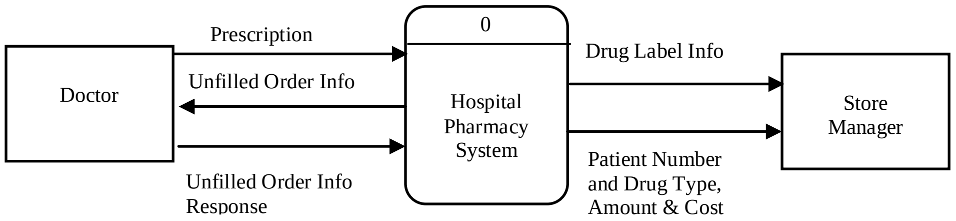
Help Desk
Level-0 Diagram

Level-0 Diagram

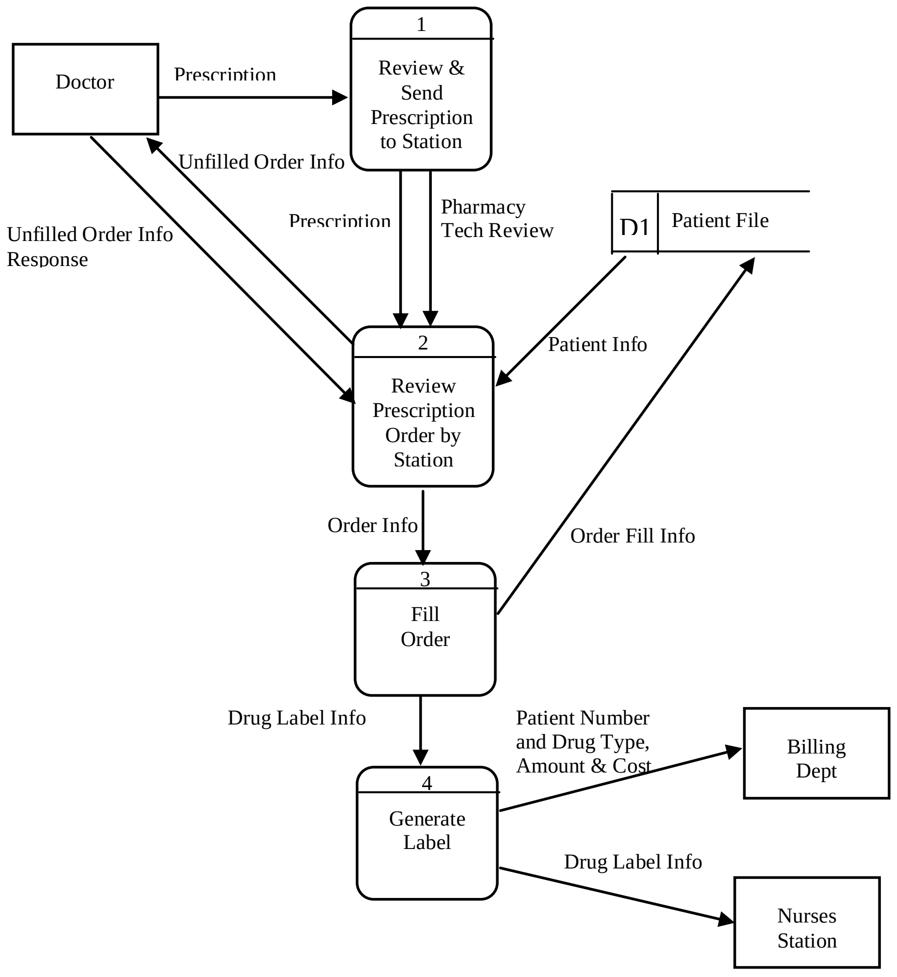


13. A suggested answer is provided below.

Hospital Pharmacy
Context Diagram

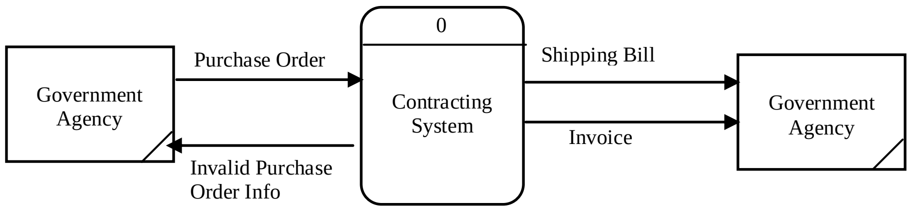

Doctor
0 Hospital Pharmacy System
Store Manager
Prescription
Unfilled Order Info
Unfilled Order Info Response
Drug Label Info
Patient Number and Drug Type, Amount & Cost

Hospital Pharmacy
Level-0 Diagram

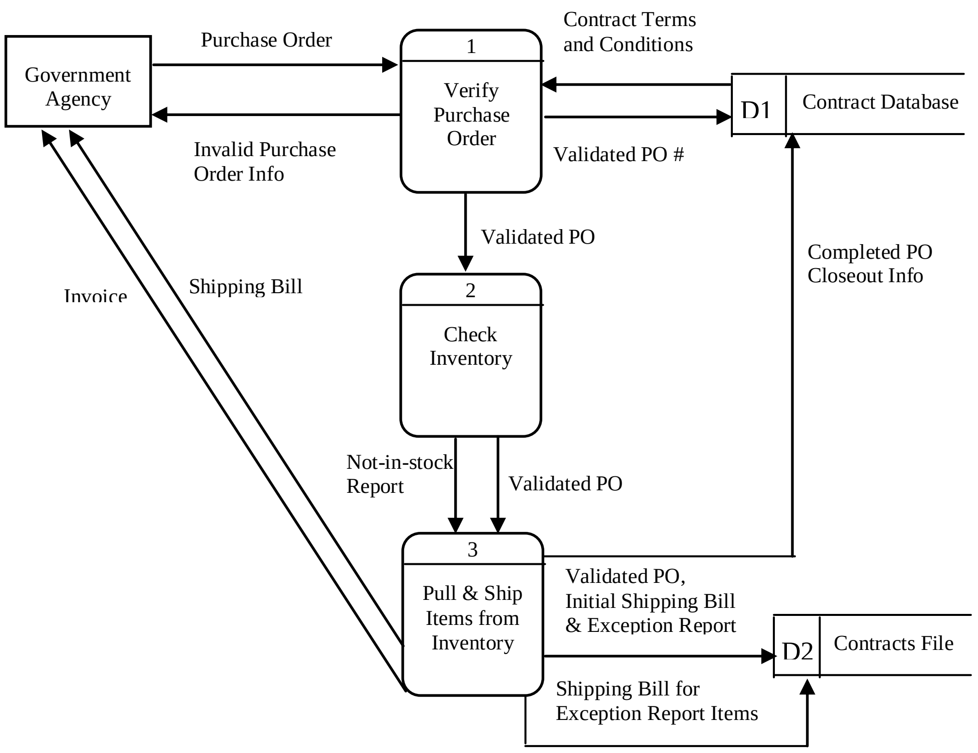


14. A suggested answer is provided below.

Contracting System
Context Diagram

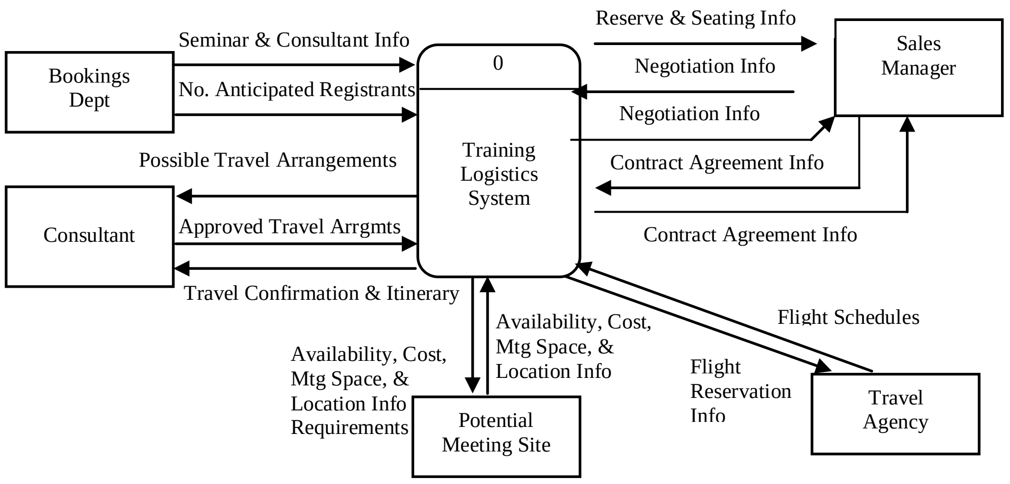


Contracting System
Level-0 Diagram

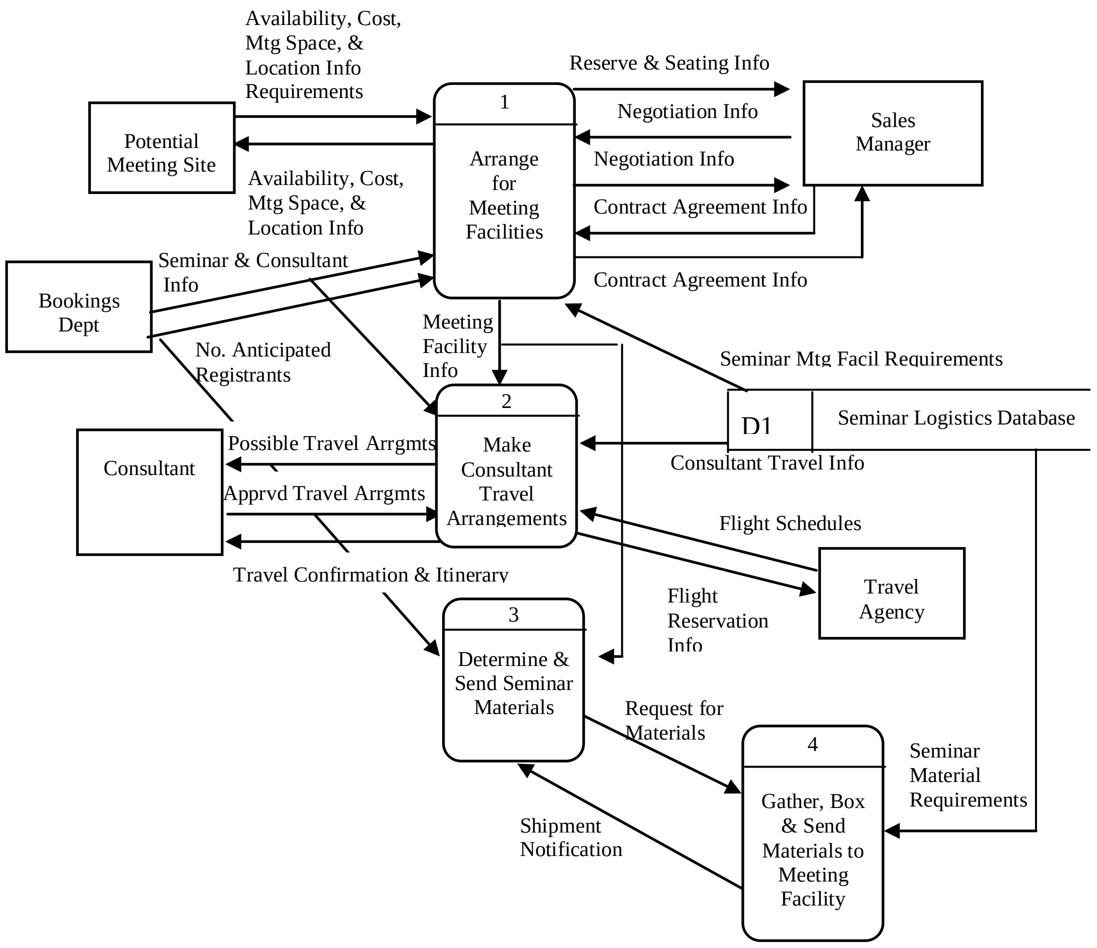

Government Agency
Purchase Order
Invalid Purchase Order Info
1
Verify Purchase Order
Contract Terms and Conditions
Validated PO #
D1
Contract Database
Validated PO
2
Check Inventory
Invoice
Shipping Bill
Completed PO Closeout Info
Not-in-stock Report
Validated PO
3
Pull & Ship Items from Inventory
Validated PO, Initial Shipping Bill & Exception Report
D2
Contracts File
Shipping Bill for Exception Report Items

15. A suggested answer is provided below.

Training Logistics System
Context Diagram


Bookings Dept
Seminar & Consultant Info
No. Anticipated Registrants
0
Training Logistics System
Reserve & Seating Info
Negotiation Info
Negotiation Info
Sales Manager
Contract Agreement Info
Contract Agreement Info
Possible Travel Arrangements
Consultant
Approved Travel Arrgmts
Travel Confirmation & Itinerary
Availability, Cost, Mtg Space, & Location Info Requirements
Availability, Cost, Mtg Space, & Location Info
Potential Meeting Site
Flight Schedules
Flight Reservation Info
Travel Agency

Training Logistics System
Level-0 Diagram


Availability, Cost, Mtg Space, & Location Info Requirements
Potential Meeting Site
Availability, Cost, Mtg Space, & Location Info
1
Arrange for Meeting Facilities
Reserve & Seating Info
Negotiation Info
Negotiation Info
Contract Agreement Info
Contract Agreement Info
Sales Manager
Bookings Dept
Seminar & Consultant Info
No. Anticipated Registrants
Meeting Facility Info
Seminar Mtg Facil Requirements
D1
Seminar Logistics Database
Consultant Travel Info
2
Make Consultant Travel Arrangements
Consultant
Possible Travel Arrgmts
Apprvd Travel Arrgmts
Travel Confirmation & Itinerary
Flight Schedules
Flight Reservation Info
Travel Agency
3
Determine & Send Seminar Materials
Request for Materials
4
Gather, Box & Send Materials to Meeting Facility
Seminar Material Requirements
Shipment Notification

16. This is an excellent question to demonstrate the integration capabilities among different diagramming tools. In order to prepare the process model, your students will need access to Oracle's process modeler. Your students' diagrams will vary, since there is not any one correct answer. Use Problem and Exercise 13's scenario to demonstrate the ease with which data flow diagrams, process models, and a functional hierarchy diagram can be built.

Guidelines for Using the Field Exercises

1. Because data flow diagrams are a popular tool, students are likely to find analysts who are using them. Students are also likely to find that data flow diagrams for real information systems are much larger and more complex than those discussed in this chapter. This is likely to help students see why data flow diagrams are so useful. Without them, it would be difficult to make sense of large, complex business processes and information systems. Students will also likely discover different notations are in use than depicted in this text. Some systems analysts may prefer object-oriented modeling rather than DFDs or may state that their organization is reducing the use of DFDs in favor of entity-relationship diagrams and other notations. It would also be interesting to discover what type of drawing or CASE tool analysts use for developing and maintaining DFDs.

2. Students are likely to have a difficult time constructing data flow diagrams for a real business process and information system. As with previous questions, there are a number of ways that students can draw their data flow diagrams for the system under question. Students should realize that there is not necessarily one "right" data flow diagram for this or most other business processes. Even if the students have not had much practice in constructing data flow diagrams, they can still construct diagrams that will be interesting and useful to the people who have developed or who are using and/or maintaining the system under question. Remind students to start with a context diagram and use decomposition to create refined, focused diagrams of different subsystems. It is difficult to draw DFDs because it can be difficult to get an unambiguous explanation from one or several people. Also, it may be hard to determine from business documents how a system works (or is supposed to work). Real business situations are typically messy, and unequivocal descriptions are not easy to obtain. This point is important for your students to learn and reveals the value of reasonably precise notations (models) such as DFDs.

3. Most CASE tools have capabilities for automatically checking for, reporting on, and warning of rule violations in data flow diagrams. Many of the CASE tools with graphical user interfaces let the user simply click on a button to run various tests of the integrity of the diagrams. Some CASE tools automatically do not allow a user to commit errors while constructing the diagram. For example, the CASE tools display an error message in a pop-up window when a user makes a mistake, such as attempting to place a data flow directly between two sources/sinks.

4. Students are likely to find that various types of software packages have elements in them that can be used to construct data flow diagrams. For example, a drawing package might include data flow diagram symbols. Alternatively, a database management package might include some data dictionary features. However, none of these will be as effective and easy for constructing data flow diagrams as will a CASE tool with features for this purpose. If students do not believe this, have them construct a simple data flow diagram using a drawing package, and then have them move sources or processes around on the screen. Chances are

that the drawing package does not automatically move all the connected data flows while these objects are moved around the screen. With a CASE tool, when an object is moved, its connecting data flows move with it. Similarly, with a CASE tool you can often move to a different, related data flow diagram by clicking on a button or by double clicking on a particular process. Also, drawing packages do not know DFD rules, so no automatic error checking can be done and decomposition is not a function of the package. In addition, since a drawing package has no repository, symbols on different diagrams probably cannot be linked (e.g., when details on a higher-level diagram change, these are automatically reflected on nested diagrams).

5. Students might find that these people draw their business process using either a data flow diagram, a flow chart, a series of input/output models, or some combination of these techniques. People will draw the diagram using the techniques with which they are familiar. If they are not familiar with any techniques for modeling processes, then they will probably draw pictures that represent physical reality (e.g., a piece of paper moving from one person to another person, or a piece of paper moving from one person to a file cabinet). For people who are not familiar with data flow diagrams, the students should find that it is relatively easy to show them that data flow diagrams are a better way to model processes. Chances are that this person's original picture already has many of the elements of a standard data flow diagram anyway. Research has found that process modeling is a very natural activity for most people, even when they are not formally trained in this technique.

Guidelines for Using the Broadway Entertainment Company Cases

Guidelines and answers for using the Broadway Entertainment Company cases are available on this textbook's companion Web site. Please visit www.prenhall.com/hoffer to access this information.