

Database Design Using the REA Data Model

LEARNING OBJECTIVES

After studying this chapter, you should be able to:

1. Discuss the steps for designing and implementing a database system.
2. Explain the nature and use of Entity-Relationship (E-R) diagrams.
3. Explain the content and purpose of the REA data model.
4. Use the REA data model to design an AIS database.
5. Read an REA diagram and explain what it reveals about the business activities and policies of the organization being modeled.

INTEGRATIVE CASE

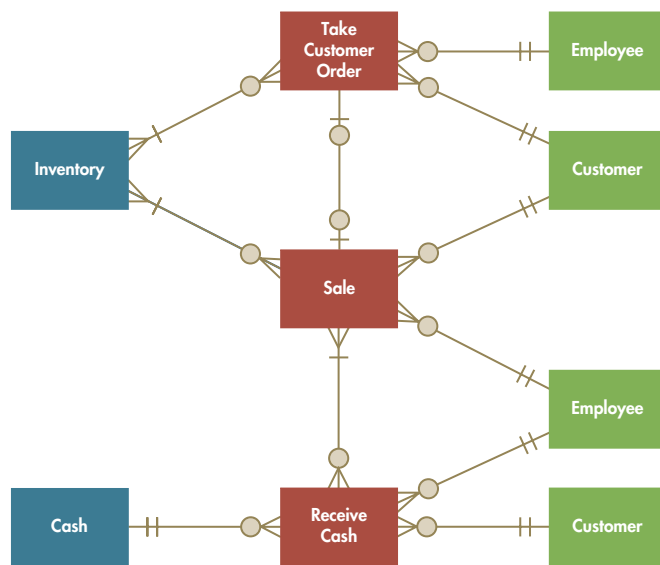
Fred's Train Shop

Fred Smith is frustrated. Business in his model train shop is booming. But the simple accounting software that he uses to run the business has only limited reporting capabilities. Consequently, he often has to manually review transaction data to prepare custom reports. The process is time consuming and prone to error. For example, Fred spent the past weekend poring over sales records for the prior three months to try to identify which combinations of items were most frequently purchased together. He plans to use the information to offer a special sales promotion but is concerned about the quality of his analysis.

At lunch, Fred explains his frustrations to his CPA, Paul Stone. Paul mentions that he has just completed a training course on database design. He suggests that he could create a relational database for Fred that would interface with his accounting software and that would provide Fred with the ability to easily design reports to analyze his business. Fred likes the idea and hires Paul to design a relational database for his train store.

Introduction

Chapter 4 covered the fundamental principles of relational databases. The three chapters in this section will teach you how to design and document a relational database for an accounting information system. Although not all of you may become consultants who, like Paul Stone in the chapter opening case, design a database for clients, every accounting professional needs to understand



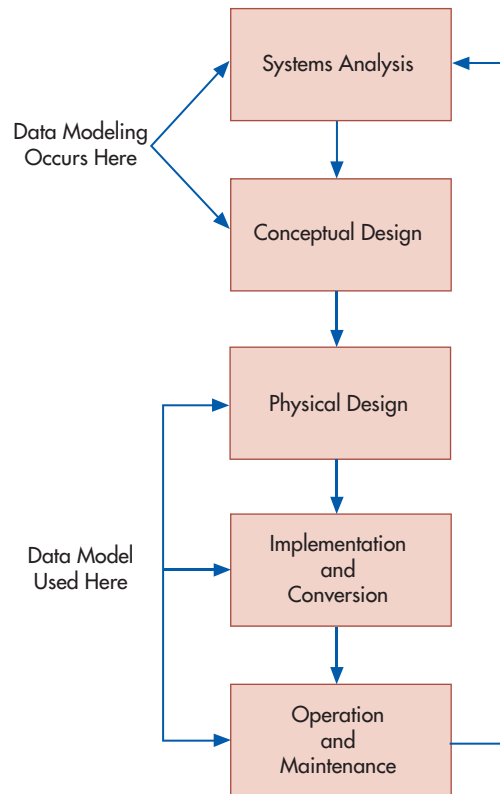
how to document a database and use such documentation as a guide for retrieving information. Auditors (both internal and external) often need to obtain audit evidence from relational databases. Corporate accountants also need to query their organization's databases to retrieve relevant data for cost analysis and tax planning, as well as to produce useful and relevant managerial reports.

This chapter introduces the topic of data modeling. We demonstrate how to use a tool called the REA (resources, events, and agents) data model to design and document an Accounting Information system (AIS). We also explain how the REA data model provides auditors with valuable information about an organization's business activities and policies. Chapter 18 describes how to implement an REA data model in a database management system and how to use it to query the resulting database to retrieve information relevant to managers and auditors. Chapter 19 concludes this three-chapter section by examining a number of advanced data modeling and database design issues.

Database Design Process

Figure 17-1 shows the five basic steps in database design. The first stage (systems analysis) consists of initial planning to determine the need for and feasibility of developing a new system. This stage includes preliminary judgments about the proposal's technological and economic feasibility. It also involves identifying user information needs, defining the scope of the proposed new system, and using information about the expected number of users and transaction volumes to make preliminary decisions about hardware and software requirements. The second stage (conceptual design) includes developing the different schemas for the new system at the conceptual, external, and internal levels. The third stage (physical design) consists of translating the internal-level schema into the actual database structures that will be implemented in the new system. This is also the stage when new applications are developed. The fourth stage (implementation and conversion) includes all the activities associated with transferring data from existing systems to the new database AIS, testing the new system, and training employees how to use it. The final stage is using and maintaining the new system. This includes carefully monitoring system performance and user satisfaction to determine the need for making system enhancements and modifications. Eventually, changes in business strategies and practices or significant new developments in information technology prompt the company to begin investigating the feasibility of developing a new system, and the entire process starts again (note the arrow returning to the systems analysis stage).

Accountants can and should participate in every stage of the database design process, although the level of their involvement is likely to vary across stages. During the systems analysis phase, accountants help evaluate project feasibility and identify user information needs. In the conceptual design stage, accountants participate in developing the logical schemas, designing the data dictionary, and specifying important controls. Accountants with good database skills may directly participate in implementing the data model during the physical design

FIGURE 17-1**Data Modeling in
the Database Design
Process**

stage. During the implementation and conversion stage, accountants should be involved in testing the accuracy of the new database and the application programs that will use that data, as well as assessing the adequacy of controls. Finally, many accountants are regular users of the organization's database and sometimes even have responsibility for its management.

Accountants may provide the greatest value to their organizations by participating in data modeling. **Data modeling** is the process of defining a database so that it faithfully represents all aspects of the organization, including its interactions with the external environment. As shown in Figure 17-1, data modeling occurs during both the systems analysis and conceptual design stages of database design. Next, we discuss two important tools that accountants can use to perform data modeling: entity-relationship diagramming and the REA data model.

Entity-Relationship Diagrams

An **entity-relationship (E-R) diagram**¹ is a graphical technique for portraying a database schema. It is called an E-R diagram because it shows the various *entities* being modeled and the important *relationships* among them. An **entity** is anything about which the organization wants to collect and store information. For example, Fred's Train Shop's database would include entities for employees, customers, suppliers, inventory, and business events such as sales to customers and deliveries from suppliers. In a relational database, separate tables would be created to store information about each distinct entity; in an object-oriented database, separate classes would be created for each distinct entity.

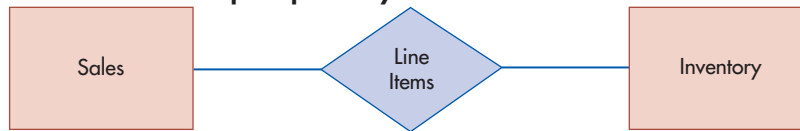
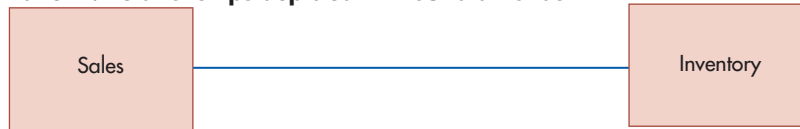
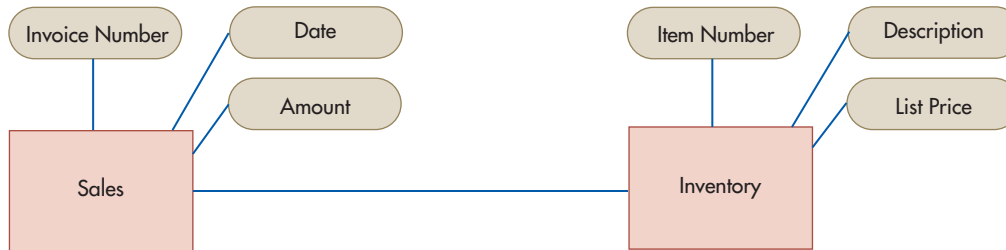
In an E-R diagram, entities are depicted as rectangles. Unfortunately, however, there are no industry standards for other aspects of E-R diagrams. Some data modelers and authors use diamonds to depict relationships (Figure 17-2, panel A) whereas others do not (Figure 17-2, panel B). Sometimes the attributes associated with each entity are depicted as named ovals connected to each rectangle (Figure 17-2, panel C), whereas other times the attributes

data modeling - Defining a database so that it faithfully represents all key components of an organization's environment. The objective is to explicitly capture and store data about every business activity the organization wishes to plan, control, or evaluate.

entity-relationship (E-R) diagram - A graphical depiction of a database's contents showing the various entities being modeled and the important relationships among them.

entity - Anything about which an organization wants to collect and store information.

¹The material in this section is based on P. Chen, "The Entity Relationship Model—Toward a Unified View of Data," *Transactions on Database Systems* (1:1, March 1976): pp. 9–36.

Panel A: Relationships depicted by diamonds**Panel B: Relationships depicted without diamonds****Panel C: Attributes attached to entities****Panel D: Attributes listed in separate table**

Entity Name	Attributes
Sales	Invoice number, date, amount
Inventory	Item number, description, list price

FIGURE 17-2**E-R Diagram Variations**

associated with each entity are listed in a separate table (Figure 17-2, panel D). In this book, we will be creating E-R diagrams with a large number of entities and relationships. Therefore, to reduce clutter and improve readability, we omit the diamonds for relationships and list the attributes associated with each entity in a separate table. Thus, our diagrams look like a combination of panels B and D in Figure 17-2.

E-R diagrams can be used to represent the contents of any kind of database. For example, the E-R diagram of an intramural sports database might include students, teams, and leagues as entities, whereas an E-R diagram for a school might include students, teachers, and courses as entities. In this book, our focus is on databases designed to support an organization's business activities. Consequently, we will show how E-R diagrams can be used not only to design databases but also to document and understand existing databases and to redesign business processes. Business process management is covered in Part V; in this chapter we focus on using E-R diagrams for database design and for understanding the contents of existing databases.

As noted, E-R diagrams can include many different kinds of entities and relationships among those entities. An important step in database design, therefore, entails deciding which entities need to be modeled. The REA data model is useful for making that decision.

The REA Data Model

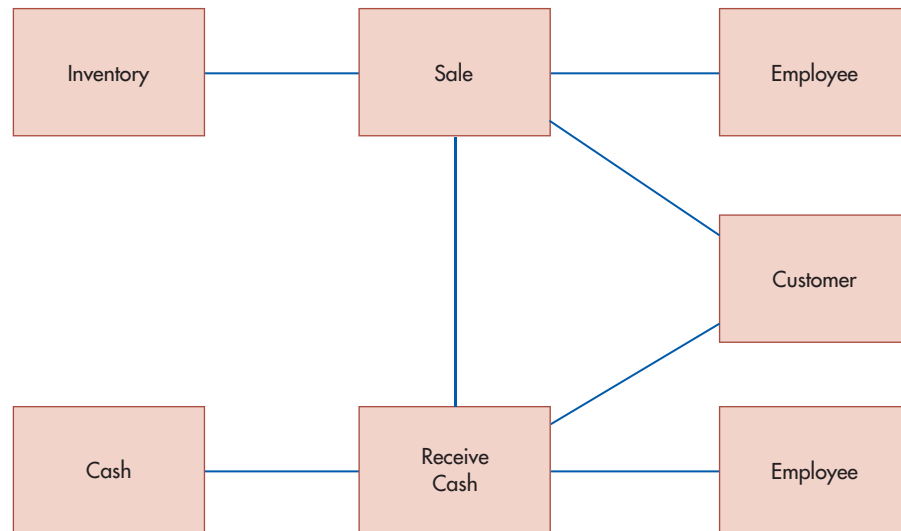
The **REA data model**² was developed specifically for use in designing AIS. The REA data model focuses on the business semantics underlying an organization's value-chain activities. It provides guidance for database design by identifying what entities should be included in the

REA data model - A data model used to design AIS databases. It contains information about three fundamental types of entities: resources, events, and agents.

²The material in this section is adapted from William E. McCarthy, "An Entity-Relationship View of Accounting Models," *The Accounting Review* (October 1979): pp. 667–686; William E. McCarthy, "The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment," *The Accounting Review* (July 1982): pp. 554–578; and Guido L. Geerts and W. E. McCarthy, "An Ontological Analysis of the Primitives of the Extended-REA Enterprise Information Architecture," *International Journal of Accounting Information Systems* (3, March 2002): pp. 1–16.

FIGURE 17-3

Basic Elements of an
REA Diagram



AIS database and by prescribing how to structure relationships among the entities in that database. REA data models are usually depicted in the form of E-R diagrams. Consequently, in the remainder of this chapter and throughout the book, we will refer to E-R diagrams developed according to the REA data model as REA diagrams.

THREE BASIC TYPES OF ENTITIES

The REA data model is so named because it classifies entities into three distinct categories: the **resources** the organization acquires and uses, the **events** (business activities) in which the organization engages, and the **agents** participating in these events.³ Figure 17-3 provides examples of these three types of entities.

Resources are those things that have economic value to the organization. Figure 17-3 includes two resource entities: Cash and Inventory. **Events** are the various business activities about which management wants to collect information for planning or control purposes.⁴ There are two event entities in Figure 17-3: Sale and Receive Cash. **Agents** are the people and organizations that participate in events and about whom information is desired for planning, control, and evaluation purposes. Figure 17-3 includes two types of agent entities: Employees and Customers.

STRUCTURING RELATIONSHIPS: THE BASIC REA TEMPLATE

The REA data model prescribes a basic pattern for how the three types of entities (resources, events, and agents) should relate to one another. Figure 17-4 presents this basic pattern. The essential features of the pattern are as follows:

1. Each event is linked to at least one resource that it affects.
2. Each event is linked to at least one other event.
3. Each event is linked to at least two participating agents.

resources - Those things that have economic value to an organization such as cash, inventory, supplies, factories, and land.

events - Business activities about which management wants to collect information for planning or control purposes.

agents - The people and organizations who participate in events and about whom information is desired.

³Some REA data modelers have proposed a fourth type of entity, which they call locations. Stores and warehouses would be examples of this fourth type of entity. However, such “location” entities are usually also resources controlled by the organization. Therefore, the authors of this text see no compelling reason to create yet another type of entity and model locations as resources. If an organization does not want or need to store information about locations except to identify where an event occurred, location can be an attribute for each event.

⁴The discussion of events in this section is based on the work of Julie Smith David, “Three ‘Events’ That Define an REA Methodology for Systems Analysis, Design, and Implementation,” Working Paper, Arizona State University, August 1997; and Guido L. Geerts and W. E. McCarthy, “An Ontological Analysis of the Primitives of the Extended-REA Enterprise Information Architecture,” *International Journal of Accounting Information Systems* (3, March 2002): pp. 1–16.

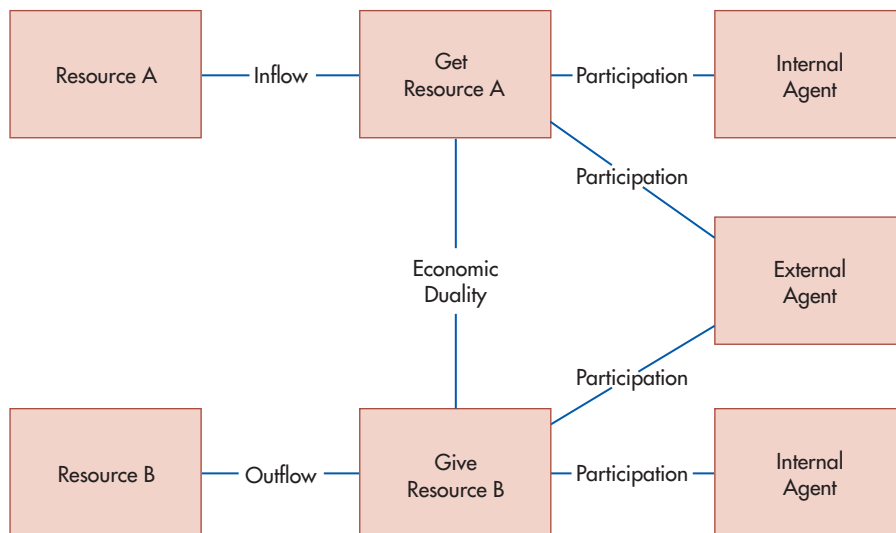


FIGURE 17-4
Standard REA Template

The names on the lines describe the nature of the relationship. Agents participate in events. The economic duality relationship between the “get” event and the “give” event reflects the fact that organizations must give up one resource (e.g., cash) in order to get some other resource (e.g., inventory). The stockflow relationships between an event and a resource represent either inflows or outflows of a resource.

RULE 1: EVERY EVENT ENTITY MUST BE LINKED TO AT LEAST ONE RESOURCE

ENTITY Events *must* be linked to at least one resource that they affect. Some events, such as the one labeled “Get Resource A” in Figure 17-4, increase the quantity of a resource. Common examples of such “Get” events include the receipt of goods from a supplier (which increases the quantity on hand of inventory) and the receipt of payment from a customer (which increases the amount of cash). Other events, such as the one labeled “Give Resource B” in Figure 17-4, directly decrease the quantity of a resource. Common examples of such “Give” events include paying suppliers and selling merchandise, which decrease the amount of cash and quantity on hand of inventory, respectively.

Relationships that affect the quantity of a resource are sometimes referred to as *stockflow* relationships because they represent either an inflow or outflow of that resource. Not every event directly alters the quantity of a resource, however. For example, orders from customers represent commitments that will eventually result in a future sale of merchandise, just as orders to suppliers represent commitments that will eventually result in the subsequent purchase of inventory. For simplicity, Figure 17-4 does not include any such commitment events. Organizations do, however, need to track the effects of such commitments, both to provide better service and for planning purposes. For example, customer orders reduce the quantity available of the specific inventory items being ordered. Sales staff need to know this information to be able to properly respond to subsequent customer inquiries and orders. Manufacturing companies may use information about customer orders to plan production. Later in the chapter we will see how to add commitment events to the basic pattern shown in Figure 17-4.

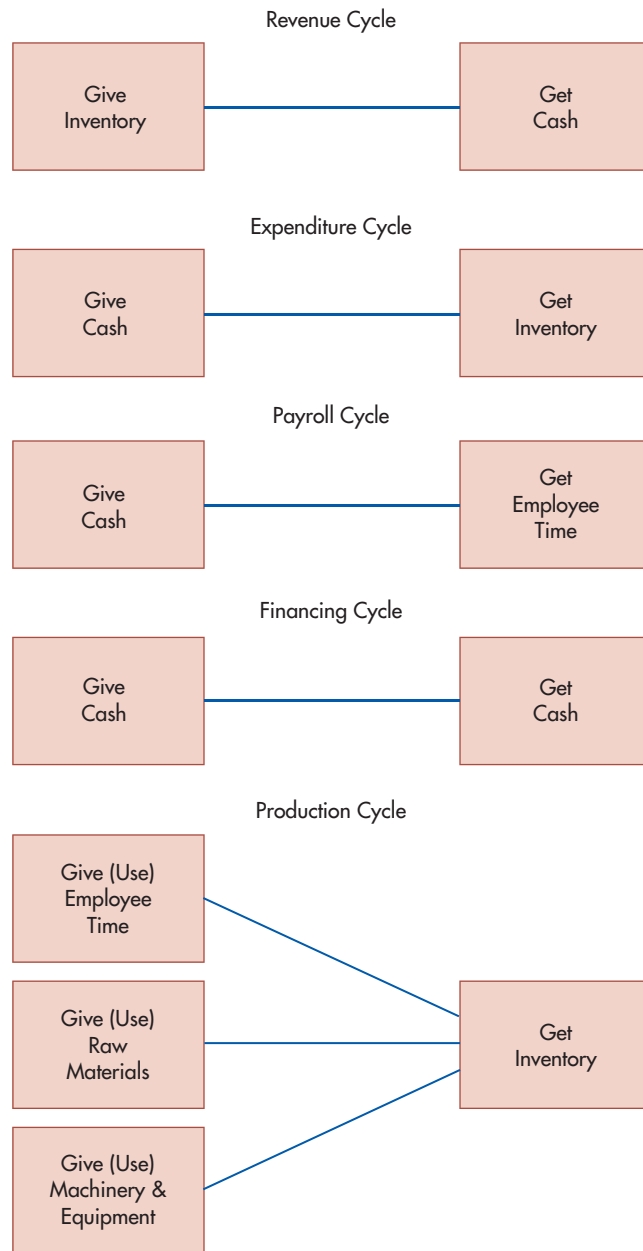
RULE 2: EVERY EVENT ENTITY MUST BE LINKED TO AT LEAST ONE OTHER EVENT

ENTITY Figure 17-4 also shows that the Get Resource A event is linked to the Give Resource B event in what is labeled as an economic duality relationship. Such give-to-get duality relationships reflect the basic business principle that organizations typically engage in activities that use up resources only in the hopes of acquiring some other resource in exchange. For example, the Sale event, which requires giving up (decreasing) inventory, is related to the Receive Cash event, which involves getting (increasing) the amount of cash. Figure 17-5 shows that each accounting cycle can be described in terms of such give-to-get economic duality relationships. The bottom portion of the figure also shows that sometimes one event can be linked to several other events.

Not every relationship between two events represents a give-to-get economic duality, however. Commitment events are linked to other events to reflect sequential cause–effect

FIGURE 17-5

An AIS Viewed as a
Set of Give-to-Get
Exchanges



relationships. For example, the Take Customer Order event would be linked to the Sale event to reflect the fact that such orders precede and result in sales. Similarly, the Order Inventory (purchase) event would be linked to the Receive Inventory event to reflect another sequential cause–effect relationship.

RULE 3: EVERY EVENT ENTITY MUST BE LINKED TO AT LEAST TWO PARTICIPATING AGENTS

For accountability, organizations need to be able to track the actions of employees. Organizations also need to monitor the status of commitments and economic duality exchanges engaged in with outside parties. Thus, Figure 17-4 shows each event linked to two participating agent entities. For events that involve transactions with external parties, the internal agent is the employee who is responsible for the resource affected by that event, and the external agent is the outside party to the transaction. For internal events, such as the transfer of raw materials from the storeroom to production, the internal agent is the employee who is giving up responsibility for or custody of the resource, and the external agent is the employee who is receiving custody of or assuming responsibility for that resource.

Developing an REA Diagram

This chapter focuses on developing an REA diagram for a single business cycle. In the next chapter we will learn how to integrate REA diagrams for individual business cycles to create one enterprise-wide REA diagram.

Developing an REA diagram for a specific business cycle consists of the following three steps:

1. Identify the events about which management wants to collect information.
2. Identify the resources affected by each event and the agents who participate in those events.
3. Determine the cardinalities of each relationship.

Let us follow these three steps to see how Paul developed Figure 17-6 to model the revenue cycle of Fred's Train Shop.

STEP 1: IDENTIFY RELEVANT EVENTS

The first step in developing an REA model of a single business cycle is to identify the events of interest to management. At a minimum, every REA model must include the two events that represent the basic give-to-get economic exchange performed in that particular business cycle (see Figure 17-5). Usually there are other events that management is interested in planning, controlling, and monitoring; they also need to be included in the REA model.

A solid understanding of activities performed in each business cycle (see Chapters 12–16) is needed to identify which events comprise the basic give-to-get economic duality relationships. For example, Chapter 12 explained that the revenue cycle typically consists of four sequential activities:

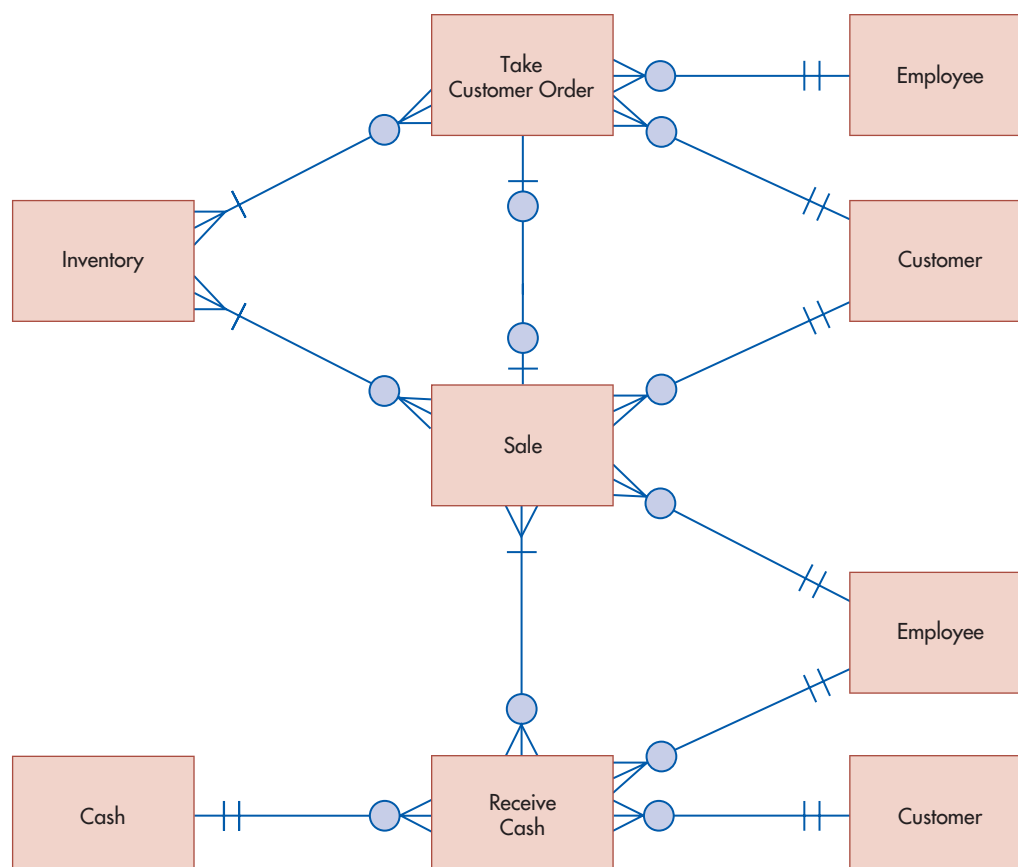


FIGURE 17-6
Partial REA Diagram
for Fred's Train Shop
Revenue Cycle

1. Take customer orders
2. Fill customer orders
3. Bill customers
4. Collect payment from customers

Analysis of the first activity, taking customer orders, indicates that it does not involve either the acquisition of resources from or provision of resources to an external party. It is only a commitment to perform such actions in the future. The second activity, fill customer orders, does reduce the organization's stock of a resource that has economic value (inventory) by delivering it to an external party (the customer). Thus, it represents an example of the prototypical Give Resource event depicted in Figure 17-4. The third activity, billing customers, involves the exchange of information with an external party but does not directly increase or decrease the quantity of any economic resource. Finally, analysis of the fourth activity, collect payments from customers, indicates that it results in an increase in the organization's supply of an economic resource (the entity labeled Cash in Figure 17-6) as a result of receiving it from an external party (the customer). Thus, it is an example of the prototypical Get Resource event depicted in Figure 17-4. Consequently, analysis of the basic business activities performed in the revenue cycle indicates that the basic give-to-get economic exchange consists of two events: fill customer orders (usually referred to as the Sale event) and collect payments from customers (often called the Receive Cash event).

In drawing an REA diagram for a single business cycle, it is useful to divide the paper into three columns, one for each type of entity. Use the left column for resources, the center column for events, and the right column for agents. Readability is further enhanced if the event entities are drawn from top to bottom corresponding to the sequence in which they occur. Thus, Paul begins to draw Figure 17-6 by placing the Sale event entity above the Receive Cash event entity in the center column of the paper.⁵

After the economic exchange events are identified, it is necessary to determine which other business activities should be represented as events in the REA model. This, too, requires understanding what each activity entails because only those activities that involve the acquisition of new information need to be included in the model. Returning to our example, Paul notes that the economic duality of Sale and Receive Cash accurately reflects most in-store sales transactions in which the customer selects one or more items and pays for them. Sometimes, however, customers call the store and ask if specific items can be set aside for pickup later that week. To ensure that he reorders popular items on a timely basis, Fred needs not only to set those items aside but also to record such orders in the system. Therefore, Paul decides to add the commitment event Take Customer Order to the REA diagram, placing it above the Sale event because customer orders precede the Sales event.

Paul then considers the other revenue cycle business activity, billing customers. He knows that in-store sales are paid for immediately and, therefore, do not involve a separate "billing" step. But Fred also sells model trains to shopping centers, hotels, and other institutions that want to set up seasonal displays for their customers. Such sales are made on credit, and Fred does subsequently prepare and mail invoices to those customers. However, printing and mailing invoices does not directly increase or decrease any economic resource. Nor does the billing activity represent a commitment to a future economic exchange: The customer's legal obligation to pay arises from the delivery of the merchandise, not from the printing of an invoice. Consequently, as noted in Chapters 12 and 13, many organizations are beginning to realize that billing is a non-value-added activity that can be eliminated entirely. Moreover, the activity of printing an invoice does not add any new information to the database. The prices and quantities of items sold were recorded at the time of the sale, at which time the terms of payment were also agreed upon. Thus, the billing activity is simply an information processing event that merely *retrieves* information from the database, similar to writing a query or printing an internal report. Since such information retrieval events do not alter the contents of the database, they need not be modeled as events in an REA diagram. For all the foregoing

⁵Placement conventions, such as the use of columns and sequential ordering of events, are not *required* to use the REA model to design a database. We suggest these rules only because following them often simplifies the process of drawing an REA diagram and produces REA diagrams that are easy to read.

reasons, Paul realizes that he does not need to include a billing event in his revenue cycle REA diagram for Fred's Train Shop.

But what about accounts receivable? If there is no billing event, how can Fred's Train Shop monitor this balance sheet item? The solution lies in understanding that accounts receivable is merely a timing difference between the two components of the basic economic exchange in the revenue cycle: sales and the receipt of payment. In other words, accounts receivable simply equals all sales for which customers have not yet paid. Consequently, accounts receivable can be calculated and monitored by simply collecting information about Sale and Receive Cash events. The next chapter will illustrate several different ways for extracting information about accounts receivable from a database built using the REA data model.

Finally, notice that there are no events that pertain to the entry of data. The reason for this is that the REA data model is used to design transaction processing databases. The objective is to model the basic value-chain business activities of an organization: what it does to generate revenues and how it spends cash and uses its other resources. Entering data about those events and about the resources and agents associated with them is not usually considered a primary value-chain activity. Thus, just like writing queries and printing reports, data entry activities are not considered important events about which detailed data needs to be collected. Moreover, as discussed in the preceding five chapters, there is a continuous trend to use technology to eliminate routine clerical information processing activities, including data entry. Thus, it is possible to conceive of business events (such as the sale of merchandise) being performed without the need for any separate data entry activities. Indeed, much data entry already occurs as a by-product of performing the business events that are included in the REA diagram. For example, whenever a sale, purchase, receipt of cash, or payment occurs, information about that event is entered in the database. Thus, what gets modeled in the REA diagram is the business event (e.g., the sale transaction) and the facts that management wants to collect about that event, not the entry of that data.

STEP 2: IDENTIFY RESOURCES AND AGENTS


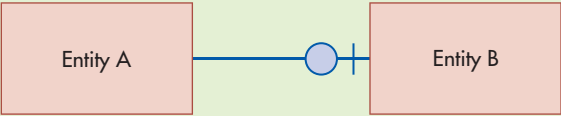
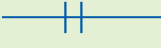


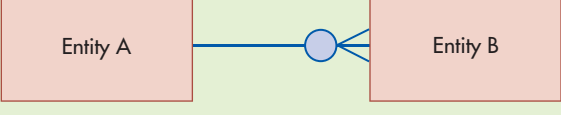

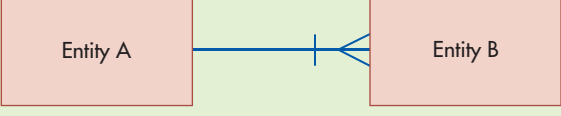
Once the relevant events have been specified, the resources that are affected by those events need to be identified. This involves answering three questions:

1. What economic resource is reduced by the "Give" event?
2. What economic resource is acquired by the "Get" event?
3. What economic resource is affected by a commitment event?

Again, a solid understanding of business processes makes it easy to answer these questions. To continue our example, Paul has observed that the Sale event involves giving inventory to customers and that the Receive Cash event involves obtaining payments (whether in the form of money, checks, credit card, or debit card) from customers. Therefore, he adds an Inventory resource entity to the REA diagram and links it to the Sale event entity. The Inventory entity stores information about each product that Fred sells. Then Paul adds a Cash resource entity to the diagram. Although organizations typically use multiple accounts to track cash and cash equivalents (e.g., operating checking account, petty cash, and short-term investments), these are all summarized in one balance sheet account called Cash. Similarly, the Cash resource contains information about every individual cash account. Thus, in a relational database, the "Cash" table would contain a separate row for each specific account (e.g., petty cash, checking account). Paul then links the Cash resource entity to the Receive Cash event entity. Finally, the Take Customer Order event involves setting aside merchandise for a specific customer. To maintain accurate inventory records, and to facilitate timely reordering to avoid stockouts, each Take Customer Order event should result in reducing the quantity available of that particular inventory item. Therefore, Paul adds a link between the Inventory resource entity and the Take Customer Order event entity in the REA diagram he is developing for Fred's Train Shop's revenue cycle.

In addition to specifying the resources affected by each event, it is also necessary to identify the agents who participate in those events. There will always be at least one internal agent (employee) and, in most cases, an external agent (customer or vendor) who participate in each event. In the case of Fred's Train Shop's revenue cycle, a customer and a salesperson participate

TABLE 17-1 Graphical Symbols for Representing Cardinality Information

SYMBOL	CARDINALITIES	EXAMPLE	MEANING
	Minimum = 0; Maximum = 1		Each instance of entity A may or may not be linked to any instances of entity B, but can be linked to at most one instance of entity B.
	Minimum = 1; Maximum = 1		Each instance of entity A must be linked to an instance of entity B, and can only be linked to at most one instance of entity B.
	Minimum = 0; Maximum = many		Each instance of entity A may or may not be linked to any instances of entity B, but could be linked to more than one instance of entity B.
	Minimum = 1; Maximum = many		Each instance of entity A must be linked to at least one instance of entity B, but can be linked to many instances of entity B.

in each Sale event. The customer and a cashier are the two agents participating in each Receive Cash event. Both the salesperson and the cashier are employees of Fred’s. Thus, both revenue cycle economic exchange events involve the same two general types of agents: employees (the internal party) and customers (the external party). The Take Customer Order event also involves both customers and employees. Therefore, Paul adds both types of agents to the diagram and draws relationships to indicate which agents participated in which events. To reduce clutter, he sometimes links one copy of a particular agent entity to two adjacent event entities.⁶

STEP 3: DETERMINE CARDINALITIES OF RELATIONSHIPS

The final step in drawing an REA diagram for one transaction cycle is to add information about relationship cardinalities. **Cardinalities** describe the nature of the relationship between two entities by indicating how many instances of one entity can be linked to each specific instance of another entity. Consider the relationship between the Customer agent entity and the Sale event entity. Each entity in an REA diagram represents a set. For example, the Customer entity represents the set of the organization’s customers, and the Sale entity represents the set of individual sales transactions that occur during the current fiscal period. Each individual customer or sales transaction represents a specific instance of that entity. Thus, in a relational database, each row in the Customer table would store information about a particular customer, and each row in the Sales table would store information about a specific sales transaction. Cardinalities define how many sales transactions (instances of the Sale entity) can be associated with each customer (instance of the Customer entity) and, conversely, how many customers can be associated with each sales transaction.

No universal standard exists for representing information about cardinalities in REA diagrams. In this text, we use the graphical “crow’s feet” notation style for representing cardinality information because it is becoming increasingly popular and is used by many software design tools. Table 17-1 explains the meanings of the symbols used to represent cardinality information, and Focus 17-1 compares the notation used in this book with other commonly used conventions.

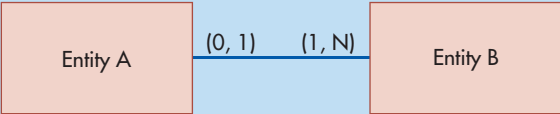


cardinalities - Describe the nature of a database relationship indicating the number of occurrences of one entity that may be associated with a single occurrence of the other entity. Three types of cardinalities are one-to-one, one-to-many, and many-to-many.

⁶Deciding how many copies of the same entity to include in an REA diagram is a matter of personal taste. Including too many copies clutters the diagram with redundant rectangles, but too few copies can result in a confusing tangled web of lines connecting entities to one another.



FOCUS 17-1 Alternative Methods to Represent Cardinality Information

A number of different notations exist for depicting minimum and maximum cardinalities. Some of the more common alternatives to the crow's feet used in this text are shown here.

NOTATION	EXPLANATION	EXAMPLE
(Min, Max)	A pair of alphanumeric characters inside parentheses: (0,1) means minimum = 0, maximum = 1 (1,1) means minimum = 1; maximum = 1 (0,N) means minimum = 0; maximum = many (1,N) means minimum = 1; maximum = many	 <p>Each instance of entity A must be linked to at least one instance of entity B but may be linked to many instances of entity B; each instance of entity B may or may not be linked to an instance of entity A but can only be linked to at most one instance of entity A. Note: Some authors and consultants flip which side of the relationship the cardinality pair appears on! So when you see an REA diagram with cardinality pairs in parentheses, ask which pair refers to which entity.</p>
UML	One or two alphanumeric characters separated by two periods: 0..1 means minimum = 0; maximum = 1 1 means minimum = 1; maximum = 1 * means minimum = 0; maximum = many 1..* means minimum = 1; maximum = many	 <p>Each instance of entity A must be linked to at least one instance of entity B but may be linked to many instances of entity B; each instance of entity B must be linked to an instance of entity A and can only be linked to at most one instance of entity A.</p>
Maximums only (Microsoft Access)	One alphanumeric character to represent the maximum cardinality in that relationship: 1 means 1; the infinity symbol (∞) means many	 <p>Each instance of entity A may be linked to many instances of entity B; each instance of entity B can only be linked to at most one instance of entity A.</p>

As shown in Table 17-1, cardinalities are represented by the pair of symbols next to an entity. The four rows in Table 17-1 depict the four possible combinations of minimum and maximum cardinalities. The **minimum cardinality** can be either zero (0) or one (1), depending upon whether the relationship between the two entities is optional (the minimum cardinality is zero; see rows one and three) or mandatory (the minimum cardinality is one, as in rows two and four). The **maximum cardinality** can be either one or many (the crow's feet symbol), depending upon whether each instance of entity A can be linked to at most one instance (as in the top two rows) or potentially many instances of entity B (as in the bottom two rows).

Let us now use the information in Table 17-1 to interpret some of the cardinalities in Figure 17-6. Look first at the Sale-Customer relationship. The minimum and maximum cardinalities next to the Customer entity are both one. This pattern is the same as that in row two in Table 17-1. Thus, the minimum cardinality of one next to the Customer entity in Figure 17-6 indicates that each sale transaction (entity A) *must* be linked to some specific customer (entity B). The maximum cardinality of one means that each sale transaction can be linked to at most *only one* specific customer. This reflects normal business practices: only one

minimum cardinality - The minimum number of instances that an entity can be linked to the other entity in the relationship. Only two options: 0 and 1.

maximum cardinality - The maximum number of instances that an entity can be linked to the other entity in the relationship. Only two options: 1 or many.

legally identifiable customer (which could be an individual or a business) is held responsible for a sale and its subsequent payment. Now look at the cardinality pair next to the Sale entity. As in row three in Table 17-1, the minimum cardinality is zero, and the maximum cardinality is many. The zero minimum cardinality means that the relationship is optional: A customer does not have to be associated with any specific sale transaction. This allows Fred's Train Shop to enter information about prospective customers to whom it can send advertisements before they have ever purchased anything. The maximum cardinality is many, indicating that a specific customer may, and Fred hopes will, be associated with multiple sale transactions (i.e., become a loyal customer who makes repeated purchases from Fred's Train Shop). Now notice that the cardinality pairs next to the Inventory entity in Figure 17-6 have a minimum of one and a maximum of many for every relationship. This is the same pattern as in row four in Table 17-1. This means that every customer order or sale transaction *must* involve at least one inventory item (you cannot sell "nothing") but *may* involve multiple different items (e.g., a customer could purchase both a locomotive and a rail car in the same transaction). Finally, notice that the cardinality pair next to the Sale entity in its relationship with the Take Customer Order entity is like the pattern in row one of Table 17-1. The minimum cardinality of zero reflects the fact that an order may not *yet* have been turned into an actual sale transaction. The maximum cardinality of one indicates that Fred's Train Shop fills all customer orders in full rather than making a number of partial deliveries.

You should be able to interpret the rest of Figure 17-6 by following the same process just presented by comparing the cardinality pairs next to each entity to the four patterns in Table 17-1. Let us now examine what the various types of relationships mean and what they reveal about an organization's business practices.

THREE TYPES OF RELATIONSHIPS Three basic types of relationships between entities are possible, depending on the *maximum* cardinality associated with each entity (the minimum cardinality does not matter):

1. A **one-to-one (1:1) relationship** exists when the maximum cardinality for each entity in that relationship is 1 (see Figure 17-7, panel A).

one-to-one (1:1) relationship - A relationship between two entities where the maximum cardinality for each entity is 1.

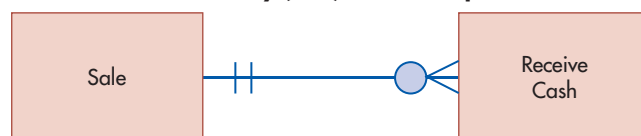
FIGURE 17-7

Examples of Different Types of Relationships

Panel A: A one-to-one (1:1) relationship



Panel B: A one-to-many (1:N) relationship



Panel C: Opposite one-to-many (1:N) relationship (sometimes referred to as N:1)



Panel D: A many-to-many (M:N) relationship



2. A **one-to-many (1:N) relationship** exists when the maximum cardinality of one entity in the relationship is 1 and the maximum cardinality for the other entity in that relationship is many (see Figure 17-7, panels B and C).
3. A **many-to-many (M:N) relationship** exists when the maximum cardinality for both entities in the relationship is many (Figure 17-7, panel D).

one-to-many (1:N) relationship - A relationship between two entities where the maximum cardinality for one of the entities is 1 but the other entity has a maximum cardinality of many.

many-to-many (M:N) relationship - A relationship between two entities where the maximum cardinality of both entities is many.

Figure 17-7 shows that any of these possibilities *might* describe the relationship between the Sale and Receive Cash events. The data modeler or database designer cannot arbitrarily choose which of these three possibilities to use when depicting various relationships. Instead, the cardinalities *must* reflect the organization's business policies. Let us now examine what each of the possibilities depicted in Figure 17-7 means. Figure 17-7, panel A, depicts a one-to-one (1:1) relationship between the Sale and Receive Cash events. The maximum cardinality of 1 associated with the Receive Cash entity means that each Sale event (transaction) can be linked to *at most* one Receive Cash event. This would be appropriate for an organization that had a business policy of not allowing customers to make installment payments. At the same time, the maximum cardinality of 1 associated with each Sale event means that each payment a customer submits is linked to *at most* one sales event. This would be appropriate for an organization that had a business policy of requiring customers to pay for each sales transaction separately. Thus, the 1:1 relationship depicted in Figure 17-7, panel A, represents the typical revenue cycle relationship for business-to-consumer retail sales: Customers must pay, in full, for each sales transaction before they are allowed to leave the store with the merchandise they purchased. Note that it does not matter *how* customers pay for each sales transaction (e.g., with cash, check, credit card, or debit card). Regardless of the method used, there is one, and only one, payment linked to each sales transaction and, conversely, every sales transaction is linked to one, and only one, payment from a customer (payments made by debit and credit cards also involve the card issuer; for simplicity, that transfer agent is not included in Figure 17-6). If management is interested in tracking the frequency of how customers choose to pay, payment method might be recorded as an attribute of the Receive Cash event.

Panels B and C of Figure 17-7 depict two ways that one-to-many (1:N) relationships can occur. Panel B shows that each Sale event may be linked to *many* Receive Cash events. This indicates that the organization has a business policy that allows customers to make installment payments *to the selling organization*. If the customer uses a third-party source of credit, the selling organization receives *one* payment in full from that third party for that particular sales transaction; the customer may be making installment payments to the credit agency, but those payments would not be modeled in an REA diagram for the selling organization. (Think about it: The selling organization has no way of tracking when one of its customers pays a portion of a credit card bill or makes a monthly payment on a bank loan). The situation depicted in Figure 17-7, panel B, does not, however, mean that *every* sales transaction is paid for in installments: The maximum cardinality of N simply means that some sales transactions may be paid in installments. Panel B of Figure 17-7 also shows that each Receive Cash event is linked to *at most* one Sale event. This indicates that the organization has a business policy that requires customers to pay for each sales transaction separately and are not allowed to build up an account balance over a period of time. Thus, Figure 17-7, panel B, represents the revenue cycle of an organization that probably sells big-ticket items. Should a customer return and make another purchase, a separate set of installment payments would be created in order to separately track how much has been paid for each sales transaction.

Figure 17-7, panel C, shows another type of 1:N relationship between the Sale and Receive Cash events. In this case, each Sale event can be linked to *at most* one Receive Cash event. This indicates that the organization has a business policy that does not permit customers to make installment payments. Figure 17-7, panel C, also shows that each Receive Cash event *may* be linked to many different Sale events. This indicates the existence of a business policy allowing customers to make a number of purchases during a period of time (e.g., a month) and then pay off those purchases with one payment. The situation depicted in Figure 17-7, panel C, is quite common, especially for business-to-business sales of nondurable goods.

Figure 17-7, panel D, depicts a many-to-many (M:N) relationship between the Sale and Receive Cash events. It shows that each Sale event may be linked to *one or more* Receive Cash events and that each Receive Cash event may in turn be linked to *one or more* Sale events.

This reflects an organization that has business policies that allow customers to make installment payments and also permits customers to accumulate a balance representing a set of sales transactions over a period of time. Keep in mind, however, that maximum cardinalities of N do not represent mandatory practices: Thus, for the relationship depicted in panel D, some sales transactions may be paid in full in one payment and some customers may pay for each sales transaction separately. The situation depicted in Figure 17-7, panel D, is quite common.

What an REA Diagram Reveals About an Organization

BUSINESS MEANING OF CARDINALITIES

REA diagrams can be used not only to design an AIS but also to understand the organization's business processes.

As noted, the choice of cardinalities is not arbitrary but reflects facts about the organization being modeled and its business practices. This information is obtained during the systems analysis and conceptual design stages of the database design process. Thus, Paul Stone had to clearly understand how Fred's Train Shop conducts its business activities to ensure that Figure 17-6 was correct.

Let us now examine Figure 17-6 to see what it reveals about Fred's Train Shop's revenue cycle processes. First, note that all of the agent–event relationships are 1:N. This is typical for most organizations: A particular agent often participates in many events. For example, organizations expect that over time a given employee will repeatedly perform a particular task. Organizations also desire their customers to make repeat orders and purchases, just as they typically place orders with the same suppliers. However, for accountability purposes, events are usually linked to a specific internal agent and a specific external agent; hence, the maximum cardinality on the agent side of the agent–event relationships in Figure 17-6 is always 1. If, however, a particular event required the cooperation of a team of employees, the maximum cardinality on the agent side of the relationship would be many.

The minimum cardinalities associated with the agent–event relationships in Figure 17-6 also reflect typical business processes followed by most organizations. The figure shows that each event *must* be linked to an agent (a sale must involve a customer, a payment must come from a customer, etc.); hence the minimum cardinality of 1 on the agent side of the relationship. In contrast, Figure 17-6 shows that the minimum cardinality on the event side of the agent–event relationship is 0. There are several reasons why a particular agent need not have participated in any events. The organization may wish to store information about potential customers and alternate suppliers with whom it has not yet conducted any business. Information about newly hired employees will exist in the database prior to their first day on the job. Finally, there is a fundamental difference in the nature of agent entities and event entities. Organizations usually desire to maintain information about agents indefinitely but typically store information only about events that have occurred during the current fiscal year. Thus, agent entities are analogous to master files, whereas event entities are analogous to transaction files. At the end of a fiscal year, the contents of event entities are typically archived, and the next fiscal year begins with no instances of that event. Thus, at the beginning of a new fiscal year, agents are not linked to any current events.

Figure 17-6 depicts M:N relationships between the Inventory resource and the various events that affect it. This is the typical situation for organizations, like Fred's Train Shop, that sell mass-produced items. Most organizations track such inventory by an identifier such as part number, item number, or stock-keeping unit (SKU) number and do not attempt to track each physical instance of that product. When a sale occurs, the system notes which product number(s) were sold. Thus, the same inventory item may be linked to many different sales events. For example, Fred's Train Shop uses product number 15734 to refer to a particular model of a steam locomotive. At a given point in time, it may have five of those locomotives in stock. If, during the course of a weekend, five different customers each purchased one of those locomotives, the system would link product number 15734 to five separate sales events. Hence, the maximum cardinality on the event side of the relationship is many. Of course, Fred's Train Shop, like most organizations, permits (and desires) that customers purchase many different products at the same time. For example, a customer who purchases a steam

locomotive (product number 15734) may also purchase a box of curved track (product number 3265). Thus, the system would link one Sale event to multiple inventory items; hence the maximum cardinality on the Inventory side of the relationship is also many.

But what if an organization sells unique, one-of-a-kind inventory, such as original artwork? Such items can only be sold one time; consequently, the maximum cardinality on the event side of the Inventory–Sale relationship would be 1. The maximum cardinality on the Inventory side of the relationship would still be many, however, because most organizations will be happy to sell as many different one-of-a-kind items as a customer wants and can afford to buy.

The minimum cardinalities on each side of the Inventory–event relationships depicted in Figure 17-6 also reflect typical business practices. Fred’s Train Shop, like many retail organizations, only sells physical inventory. Therefore, every order or sales event *must* be linked to at least one inventory item; hence, the minimum cardinality on the Inventory side of the Inventory–event relationships is 1. The minimum cardinality on the event side of those relationships, however, is 0, for the same reasons that it is 0 in agent–event relationships.

Now consider the relationship between the Cash resource and the Receive Cash event. Figure 17-6 depicts this as being a 1:N relationship, which reflects a best practice followed by most organizations with good internal controls. Each cash receipt from a customer is deposited into one cash account, usually the organization’s general checking account. The treasurer subsequently transfers money from that account to other cash accounts (e.g., payroll, checking, investments) as necessary. The minimum cardinalities on each side of this relationship are also typical. Each customer payment must be deposited into some account; hence the minimum cardinality is 1 on the resource side of the relationship. Conversely, the minimum cardinality on the event side of the relationship is 0 for the same reasons that it is 0 in the agent–event and inventory–event relationships discussed previously.

Finally, let us examine the event–event relationships depicted in Figure 17-6. Fred’s Train Shop ships each business customer order individually and waits until all items are in stock before filling an order. Thus, each order is linked to only one sales transaction, and each sales transaction is related to only one order. Therefore, Paul has modeled the relationship between the Take Customer Order and Sale events as being 1:1. The minimum cardinality on the Sale side of the relationship is 0, meaning that orders may exist which are not linked to sales. This reflects the temporal sequence between the two events: Orders precede sales, so at any given point in time, Fred’s Train Shop may have orders that it has not yet filled. Fred’s Train Shop does not, however, require that every sale be preceded by an order; indeed, while many sales to corporate customers are preceded by orders, walk-in sales to consumers are not. Therefore, Paul Stone has modeled the minimum cardinality on the Take Customer Order side of the Sale–Take Customer Order relationship as 0.

Paul also has learned that Fred’s Train Shop extends credit to its business customers and mails them monthly statements listing all unpaid purchases. He also has found out that many business customers send Fred one check to cover all their purchases during a given time period. Thus, one Receive Cash event could be linked to many different Sale events. However, Fred’s Train Shop also allows its business customers to make installment payments on large purchases; thus, a given Sale event could be connected to more than one Receive Cash event. That is why Paul has modeled the relationship between the Sale and Receive Cash events as being many-to-many.

Because Fred’s Train Shop extends credit to some of its customers, at any point in time there can be Sale events that are not yet linked to any Receive Cash events. Therefore, Figure 17-6 shows the minimum cardinality on the Receive Cash side of the relationship as 0. Paul also has learned that Fred’s Train Shop never requires customers to pay in advance for special orders. Thus, every Receive Cash event must be linked to a previous Sale event; consequently, Figure 17-6 shows the minimum cardinality on the sales side of the Sale–Receive Cash relationship is 1.

UNIQUENESS OF REA DIAGRAMS

The preceding discussion indicates that each organization will have its own unique REA diagram. At a minimum, because business practices differ across companies, so will relationship cardinalities. In fact, an REA diagram for a given organization will have to change to reflect changes to existing business practices. For example, if Fred’s Train Shop decides to begin making partial shipments to fill customer orders, then Figure 17-6 would have to be changed to show the relationship between the Take Customer Order and Sale events as being 1:N,



FOCUS 17-2 Why Should Users Participate in Data Modeling?

Data modeling is not an easy task, as Hewlett-Packard learned when it began designing a new database for its accounting and finance function. A major problem was that the same term meant different things to different people. For example, accounting used the term *orders* to refer to the total dollar amount of orders per time period, whereas the sales department used the term to refer to individual customer orders. Moreover, such confusions existed even within the accounting and finance function. For example, the reporting group used the term *product* to refer to any good currently sold to customers. Thus, the primary key for this entity was product number. In contrast, the forecasting group used the term *product* to refer to any good that was often still in the planning stage and had no product number assigned yet.

To solve these problems, Hewlett-Packard asked the different user groups to actively participate in the data modeling process. The first step was to convince all users of the need for and benefits of creating a data model for their function. Then it was necessary to carefully define the scope of the modeling effort. Hewlett-Packard found

that the time invested in these early steps was worthwhile, because it facilitated the activities of clarifying definitions and developing attribute lists that took place later in the process. The latter activity was an iterative affair that included many revisions. Documentation was critical to this process. Each member of the modeling team and user groups had copies of the proposed lists, which made it easier to spot inconsistencies in definitions.

Hewlett-Packard credits the data modeling approach as contributing significantly to the project's overall success. Data modeling allowed the participants to concentrate first on understanding the essential business characteristics of the new system, instead of getting bogged down in specifying the contents of relational database tables. This helped them to identify and resolve conflicting viewpoints early in the process and paved the way for eventual acceptance of the resulting system. The key step, however, was in getting the different user groups to actively participate in the data modeling process. Otherwise, the resulting data model would not have been as accurate or widely accepted.

instead of the 1:1 relationship currently depicted. Similarly, if Fred's Train Shop also decided to adopt a policy of combining several orders from one customer into one large shipment, then Figure 17-6 would have to be modified to depict the relationship between those two events as being M:N. Sometimes, differences in business practices can result in different entities being modeled. For example, if Fred's Train Shop only made sales to walk-in customers and did not take any orders from businesses, then Figure 17-6 would not need to include the Take Customer Order commitment event.

Although the development of the REA diagram for Fred's Train Shop's revenue cycle may seem to have been relatively straightforward and intuitive, data modeling is usually a complex and repetitive process. Frequently, data modelers develop an initial REA diagram that reflects their understanding of the organization's business processes, only to learn when showing it to intended users that they had omitted key dimensions or misunderstood some operating procedures. Thus, it is not unusual to erase and redraw portions of an REA diagram several times before finally producing an acceptable model. One common source of misunderstanding is the use of different terminology by various subsets of the intended user groups. Focus 17-2 highlights the importance of involving the eventual users of the system in the data modeling process so that terminology is consistent.

Summary and Case Conclusion

The database design process has five stages: systems analysis, conceptual design, physical design, implementation and conversion, and operation and maintenance. Because of their extensive knowledge of transaction processing requirements and general business functions, accountants should actively participate in every stage.

One way to perform the activities of systems analysis and conceptual design is to build a data model of the AIS. The REA accounting data model is developed specifically for designing a database to support an AIS. The REA model classifies entities into three basic

categories: resources, events, and agents. An REA model can be documented in the form of an entity-relationship (E-R) diagram, which depicts the entities about which data are collected as rectangles and represents the important relationships between entities by connecting lines. The cardinalities of the relationships depicted in REA diagrams specify the minimum and maximum number of times an instance of one entity can be linked to an instance of the other entity participating in that relationship. Cardinalities also provide information about the basic business policies an organization follows.

Developing an REA diagram involves three steps. First, identify the basic events of interest (any activity about which management wants to collect information in order to plan, control, and evaluate performance). Second, identify the resources affected by and the agents who participate in those events. Third, use knowledge about the organization's business practices to add relationship cardinality information to the diagram.

Paul Stone followed these steps to develop an REA diagram for Fred's Train Shop's revenue cycle. He interviewed Fred to understand the store's business policies and used his general knowledge of revenue cycle activities to draw Figure 17-6. Paul showed the diagram to Fred and explained what each portion represents. Fred indicated that the diagram correctly reflects his store's revenue cycle activities. Paul then explained that he will proceed to use the model to design a relational database that Fred can use to automate the analyses he currently does by hand.

KEY TERMS

data modeling 528	events 530	one-to-one (1:1) relationship 538
entity-relationship (E-R) diagram 528	agents 530	one-to-many (1:N) relationship 539
entity 528	cardinalities 536	many-to-many (M:N) relationship 539
REA data model 529	minimum cardinality 537	
resources 530	maximum cardinality 537	

AIS in Action

CHAPTER QUIZ

- Accounts Receivable would appear in an REA diagram as an example of which kind of entity?
 - resource
 - event
 - agent
 - none of the above
- Which of the following is NOT likely to be depicted as an entity in the REA data model?
 - customers
 - sales
 - invoices
 - delivery trucks
- In most cases, the relationship between agent entities and event entities is _____.
 - 1:1
 - 1:N
 - M:N
 - 0:N
- If customers pay for each sales transaction with a separate check and are not permitted to make installment payments on any sales, then the relationship between the Sale and Receive Cash events would be modeled as being which of the following?
 - 1:1
 - 1:N
 - M:N
 - 0:N