End of the Chapter review questions

Review Questions Chapter 5

1. **What are the two key concepts used to begin defining system requirements?**
   Events the system needs to respond to and things that the system needs to store information about.

2. **What is a use case?**
   A use case is an activity the system performs.

3. **What are three techniqus used to identify use cases?**
   The user goal technique, the CRUD technique, and the Elementary Business Process technique.

4. **What is an event and what is an elementary business process (EBP)?**
   An event is an occurance at a specific time and place that should be remembered. An EBP is a task that is performed by one person in one place in response to a business event.

5. **Whare are the three types of events?**
   A temporal event, a state event, and an external event.

6. **Which type of event results in data entering the system?**
   External event.

7. **Which type of event occurs at a defined point in time?**
   Temporal event.

8. **Which type of event does not result in data entering the system but always results in an output?**
   Temporal event.

9. **What type of event would be named *Employee quits job*?**
   An external event because the data entering the system would be data about the termination of the employee (who, when, and why).

10. **What type of event would be named *Time to produce paychecks*?**
    A temporal event because the system knows it is time to produce paychecks at the end of the month.

11. **What are some examples of system controls?**
    Validating user input, requiring user IDs and passwords for logging on to the system, backing up data regularly, encrypting data that is transmitted, and so on.

12. **What does the perfect technology assumption state?**
    The perfect technology assumption states that events should be included during analysis only if the system would be required to respond under perfect conditions—that is, with equipment never breaking down, capacity for processing and storage being unlimited, and people operating the system being completely honest and never making mistakes.

Basically, the perfect technology assumption prevents analysts from worrying about systems controls until later during the design phase.

13. **What are the columns in an event table?**
Event, trigger, source, activity/use case, response, and destination.

14. **What is a trigger? A source? An activity or use case? A response? A destination?**
See margin definitions. A trigger is a data input for an external event and a definition of a point of time for a temporal event. A source is what external agent or actor supplies the data input for an external event. The activity/use case is what the system does when the event occurs (the process). A response is a data output from the system. The destination is the external agent or actor that receives the data output.

15. **What is the difference between a use case and a scenario? Give and example of each.**
A scenario is a particular instance or set of steps for one path through the use case. A use case is a goal oriented process done by the system.

16. **What are the three types of use case descriptions? Whish one is usually sufficient for a simple use case?**
Bried description, Intermediate description, and a Fully Developed description.

17. **What are preconditions and postconditions? Give an exmaple of each.**
Preconditions are conditions that must exist in the system before the use case begins. Post conditions are those that must exist after the use case completes its processing.

18. **What are exceptions conditions? Give an example of each.**
Exception conditions are steps that are outside of the ordinary or normal process of steps for a use case. An example would be out of stock condition.

19. **What communicates back and forth in a two-column flow of activities?**
In a flow of activities, the communication is between an external actor and the automated system. Normally this is some sort of data communications.

20. **What is a "thing" called in models used in the traditional approach?**
A data entity.

21. **What is a "thing" called in the object-oriented approach?**
An object.

22. **What is a relationship?**
A naturally occurring association among specific things, such as an order is placed by a customer, and an employee works in a department.

23. **What is cardinality of a relationship (also called multiplicity)?**
The number of associations that occur between specific things, such as a customer places many orders, and an employee works in one department.

24. **Describe how an entity-relationship diagram shows the minimum and maximum cardinality.**
A circle on the relationship line means zero, and a short slash line means one. A "crow's feet" symbol means many.

25. **What are unary, binary, and n-ary relationships?**

A unary relationship is a relationship between two things of the same type; a binary relationship is a relationship between two things of different types; and an n-ary relationship is a relationship among *n* things or *n* different types.

26. What are attributes and compound attributes?
An attribute is one piece of specific information about a thing, such as the first name of a person. A compound attribute is made up of more than one specific piece of information. For example a *Full name* compound attribute could represent the first, middle, and last names of a person.

27. What is an associative entity?
An associative entity is a data entity that represents a many-to-many relationship between two entities when the relationship includes information needing to be stored.

28. What are the symbols shown on an entity-relationship diagram?
Rectangle for entity, line for relationship, minimum and maximum cardinality symbols including a "crow's feet" symbol are shown on the line of a relationship.

29. What are the symbols shown on a domain model class diagram?
A rectangle with three sections inside for class, a line with a triangle for generalization/ specialization (inheritance), a line with a diamond for aggregation (whole part), and a line with multiplicity written on it for association relationships.

30. How is multiplicity show in a domain model class diagram?
Multiplicity occurs on relationships and is shown by two numbers or symbols separated by two dots, e.g. 0..1 or 1..*

31. What is a generalization/specialization hierarchy?
A hierarchy that structures or ranks classes from the more general superclass to the more specialized subclasses. In OO, the attributes and methods (plus relationships) of the superclass are inherited by a subclass.

32. From what type of class do subclasses inherit?
Superclass.

33. What are two types of whole-part hierarchies?
Aggregation and composition.

34. What does the triangle symbol indicate on a line connecting classes on the class diagram?
The triangle symbol indicates that the class it touches is a superclass and that the class at the other end of the line is a subclass.

35. How is an association class shown on a class diagram?
A class connected to the association line with a dashed line.

36. What types of classes are shown in a domain model class diagram?
A class diagram without methods, created as part of the requirements model. They are classes in the work domain or problem domain of the user.

Chapter 7 – The Object-Oriented Approach to Requirements

Solutions to End-of-Chapter Material

Review Questions

1. **What is the OMG?**

   The OMG is the Object Management Group. It is a consortium of over 800 organizations consisting of software vendors, developers and other organizations. The objective of the consortium is to set standards and establish uniformity in object-oriented development, including standard specifications across environments.

2. **What is UML? What type of modeling is it used for?**

   UML stands for Unified Modeling Language. It is the standard object-oriented modeling language that has been accepted by the OMG. It is used for requirements specification and architectural design of object-oriented systems.

3. **What are the four basic parts of a use case model? What is its purpose or objective?**

   The four parts (icons) of a use case model are the use case identifier (an oval), the use case actor (a stick figure), the relationship conneting lines, and the system boundary.

   The objective of the use case model is to identify all of the ways the users will use the system. A use case identifies a specific way, such as completing a business transaction, that the system is used and must support.

4. **What are the two basic parts of the domain model? What is its purpose or objective?**

   The two basic parts are the class identifiers (rectangles with two compartments), and relationships (connecting lines with mutliplicities).

   The object of the domain model is to identify the problem domain objects and the relatinships between those objects.

5. **What is the difference between a use case description and an activity diagram?**

   A use case description identifies several characteristics of the use case. It also includes a textual description of the steps included in the workflow of the use case. An activity diagram is a graphical model that shows the steps in the workflow of the use case.

6. **What is the «includes» relationship used for?**

The «includes» relationship is used to connect two use cases when one use cases invokes the other use case. The "included" use case is like a common subroutine that can be used by other processes or use cases.

7. What is the difference in the focus on the boundary condition of a use case diagram and an event table?

The identification of events for the event table requires a source and a trigger. The source and trigger are those items that initiate an event in the entire system, including the manual system and the automated system. In a use case diagram, the boundary is an automation boundary—that is, it denotes the interface between the environment, where the actors reside, and the internal components of the computer system. The source and the actor are not always the same. The source is where the data comes from.

8. With regard to a use case, what is an activity diagram used for?

An activity diagram documents the workflow in the use case. It shows the external actor in one swimlane, and the system responses in another swimlane. Usually one activity diagram can show the flow of activities for a single use case or scenario.

9. What is the purpose of a system sequence diagram? What symbols are used in a system sequence diagram?

A system sequence diagram (SSD) identifies the input and output messages to the automated system for a use case. The input and output messages describe the inputs and outputs to the system.

The symbols are actors and objects. Actors are depicted by stick figures. Objects are depicted as rectangles with the inside label underlined. Lifelines are attached to actors and objects. Lifelines are depicted by vertical dashed lines beneath the actors and objects. The input and output messages are depicted as arrows between the lifelines.

10. What are the steps required to develop a system sequnce diagram?

There are four steps. In most cases the SSD can be developed from an activity diagram or a flow of events on the use case description.

1. Identify the input messages. This usually occurs on an activity diagram when an arrow crosses the swimlanes from the actor to the system.
2. Describe the input message with name, parameters, and other descriptors.
3. Identify other special conditions for the message, such as repetition or true/false conditions.
4. Add the output return messages.

11. What is the purpose of a state machine diagram?

During analysis, state machine diagrams are used to describe the behavior of complex business objects. The primary focus of a state machine diagram is to identify the various

states of a business object and to describe the ways in which the object moves from state to state.

12. List the primary steps for developing a state machine diagram.

1. Pretend you are the object itself. Identify and model the states and transitions.
2. Make sure you review the state machine diagram and make necessary changes.
3. Make sure you haven't left out any exception conditions.
4. Review the class diagram, and select the classes that will require a state machine diagram.
5. For each selected class in the group, make a list of all the status conditions you can identify.
6. Begin building state machine diagram fragments by identifying the transitions that cause an object to leave the identified state.
7. Sequence these state-transition combinations in the correct order.
8. Review the paths and look for independent, concurrent paths.
9. Look for additional transitions.
10. Expand each transition with the appropriate message event, guard-condition, and action-expression.
11. Review and test each state machine diagram.

13. List the elements that make up a transition description. Which elements are optional?

The elements are transition-name, guard-condition, and action-expression. All of these elements are optional. If a transition has none of these elements, it "fires" automatically when the object has finished any activity with the origin state.

14. What is a composite state? What is it used for?

A composite state is a state containing multiple states and transitions. It represents a higher level of abstraction and can contain nested states and transition paths.

15. What is meant by the term *path*?

A path is a sequential set of connected states and transitions.

16. What is the purpose of a guard-condition?

A guard-condition indicates the condition that must be true before a transition can occur

17. Identify the models explained in this chapter and their relationship to each other.

A use case diagram identifies all of the use cases of the system. A class diagram identifies the domain classes for the system. A CRUD analysis ensures that these two diagrams are consistent with each other. Each use case is described either with a use case description (with various levels of detail) or an activity diagram. The use case description and activity diagram describe the internal flow of activities of the use case. A system sequence diagram shows the inputs and outputs of each use case. The state machine diagram identifies the states or status conditions an object can be in. It shows possible steps in a use case or identifies the need for a use case.

Chapter 8 – Evaluating Alternatives for Requirements, Environment, and Implementation

Solutions to End-of-Chapter Material

Review Questions

1. **What is meant by the *application deployment environment*? Why is it important in the consideration of a development approach?**

   The application deployment environment consists of the computer hardware (platforms) and operating systems that will support the application program. The new application system may have to meet special design requirements to conform to the restrictions of the operating system and the equipment.

2. **List and briefly describe the characteristics that an analyst examines when choosing or defining the deployment environment.**

   Analysts consider the configuration of computer equipment, operating systems, and networks that will exist when the new application system is deployed.

3. **Describe the relationship between the application deployment and development environments.**

   The development environment consists of the programming languages, CASE (computer-assisted software engineering) tools, and other software used to develop application software.

   The application deployment environment, particularly the operating system, DBMS, and distributed software standard, usually limits the choices that are available for the development environment.

   Companies prefer certain languages for system development, and their analysts are familiar with the features of these languages. As newer languages provide additional capabilities, however, companies may consider different languages for their system development. An analyst should consider both the application deployment and development environments when determining how they will work with a particular application.

4. **Explain the fundamentals of facilities management.**

   Facilities management is where an outside company takes over the complete data processing responsibilities of an organization. This normally includes control of all of the hardware, programs, and personnel.

5. **What is the difference between scope and level of automation?**

   Scope is determined by the list of functions that are included in a system. Level of automation is the amount of computer sophistication and support provided for each function.

6. **What is meant by the make-versus-buy decision?**

   The make-versus-buy decision involves determining whether to build a system from the ground up or to buy a packaged solution.

7. **Define a *packaged solution*. Explain what is entailed in the packaged solution approach.**

   A packaged solution is an application program that has been built by a company that specializes in providing computer software for a particular area such as a point-of-sale system. In a strict packaged solution approach, an organization will buy, install, and use the package as is. In a modified approach, the organization may have to modify the package to get it to fulfill the requirements.

8. **What is meant by *ERP*? How does an ERP approach affect acquiring a new solution?**

   ERP stands for enterprise resource planning. ERP takes a strategic, enterprise-wide view of the organization in the planning of new systems. ERP systems have components that support all major functions of a company in a given industry. A complete ERP solution provides new systems that support all company functions as opposed to the functions of individual departments.

9. **What does *outsourcing* mean? How does it impact a project?**

   Outsourcing involves hiring an outside company to provide specific services or products that are usually performed by in-house personnel.

The range of outsourcing alternatives for any given project varies. Alternatives to outsourcing include: contract programmers, project management services, facilities management, systems analyst contractors, purchase of package system, and so forth.

10. Define *benchmark*. Why is it useful in selecting a new system?

A benchmark involves defining standard performance criteria and then evaluating a system against that benchmark. A benchmark is especially useful in the purchase of a new system from an outside vendor. The new system can be benchmarked against the existing system, against the industry average, or against some other identified criteria.

11. What is an *RFP*? Why is it developed at the end of the analysis phase instead of at the beginning?

An RFP is a request for proposal. It is a document that details the information needs and requirements for a new system. An RFP is developed at the end of the analysis phase so that the requirements can be described in detail. These details help the vendor respond to and price a proposal, and they help the requestor to define specific criteria to see if the proposal meets the requirements. Without this detailed information, a proposal and evaluation would be too vague and prone to errors.

12. What is the difference between general requirements, technical requirements, and functional requirements?

General requirements include considerations that are important but that are not directly associated with the computer system itself. General requirements include the track record and reputation of the vendor.

Technical requirements are the technical constraints imposed on the system, such as response time, operating platform, and database system.

Functional requirements identify the specific business functions that must be provided by the new system.

Solutions to End-of-Chapter Material

Review Questions

1. List the components of a DBMS, and describe the function of each.

Application program interface – An interface engine or library of precompiled subroutines that enable application programs (such as those written in C or Java) to interact with the database.

End-user query processor – A program or utility that allows end users to retrieve data and generate reports without writing application programs.

Data definition interface – A program or utility that allows a database administrator to define or modify the content and structure of the database (for example, add new fields or redefine data types or relationships).

Data access and control logic – The system software that controls access to the physical database and maintains various internal data structures (for example, indices and pointers).

Database – The physical data store (or stores) combined with the schema.

Schema – A store of data that describes various aspects of the "real" data, including data types, relationships, indices, content restrictions, and access controls.

Physical data store – The "real" data as stored on a physical storage medium (for example, a magnetic disk).

2. What is a database schema? What information does it contain?

A database schema is a store of data that describes the content and structure of the physical data store (sometimes called metadata—data about data). It contains a variety of information about data types, relationships, indices, content restrictions, and access controls.

3. Why have databases become the preferred method of storing data used by an information system?

Databases are a common point of access, management, and control. They allow data to be managed as an enterprise-wide resource while providing simultaneous access to many different users and application programs. They solve many of the problems associated with separately maintained data stores, including redundancy, inconsistent security, and inconsistent data access methods.

4. List four different types of database models and DBMSs. Which are in common use today?

The four database models are hierarchical, network (CODASYL), relational, and object-oriented. Hierarchical and network models are technologies of the 1960s and 1970s and are rarely found today. The relational model was developed in the 1970s and widely deployed in the 1980s and 1990s. It is currently the predominant database model. The object-oriented database model was first developed in the 1990s and is still being developed today. It is expected to slowly replace the relational model over the next decade.

5. With respect to relational databases, briefly define the terms *row* and *field*.

Row – The portion of a table containing data that describes one entity, relationship, or object.

Field – The portion of a table (a column) containing data that describes the same fact about all entities, relationships, or objects in the table.

6. What is a primary key? Are duplicate primary keys allowed? Why or why not?

A primary key is a field or set of fields, the values of which uniquely identify a row of a table. Because primary keys must uniquely identify a row, duplicate key values aren't allowed.

7. What is the difference between a natural key and an invented key? Which type is most commonly used in business information processing?

A natural key is a naturally occurring attribute of or fact about something represented in a database (for example, a human fingerprint or the atomic weight of an element). An invented key is one that is assigned by a system (for example, a social security or credit card number). Most keys used in business information processing are invented.

8. What is a foreign key? Why are foreign keys used or required in a relational database? Are duplicate foreign key values allowed? Why or why not?

A foreign key is a field value (or set of values) stored in one table that also exists as a primary key value in another table. Foreign keys are used to represent relationships among entities that are represented as tables. Duplicate foreign keys are not allowed within the same table because they would redundantly represent the same relationship. Duplicate foreign keys may exist in different tables because they would represent different relationships.

12. Describe the steps used to transform an ERD into a relational database schema.

13. Create a table for each entity type.
14. Choose a primary key for each table.
15. Add foreign keys to represent one-to-many relationships.
16. Create new tables to represent many-to-many relationships.
17. Define referential integrity constraints.
18. Evaluate schema quality and make necessary improvements.
19. Choose appropriate data types and value restrictions for each field.

17. How is an entity on an ERD represented in a relational database?
Each entity on an ERD is represented as a separate table.

18. How is a one-to-many relationship on an ERD represented in a relational database?

A one-to-many relationship is represented by adding the primary key field(s) of the table that represents the entity participating in the "one" side of the relationship to the table that represents the entity participating in the "many" side of the relationship.

19. How is a many-to-many relationship on an ERD represented in a relational database?

A many-to-many relationship is represented by constructing a new table that contains the primary key fields of the tables that represent each participating entity.

20. What is referential integrity? Describe how it is enforced when a new foreign key value is created, when a row containing a primary key is deleted, and when a primary key value is changed.

Referential integrity is content constraint between the values of a foreign key and the values of the corresponding primary key in another table. The constraint is that values of the foreign key field(s) must either exist as values of a primary key or must be NULL. A valid value must exist in the foreign key field(s) before the row can be added. When a row containing the primary key is deleted, the row with the foreign key must also be deleted for the data to maintain referential integrity. A primary key should never be changed; but in the event that it is, the value of the foreign key must also be changed.

21. What types of data (or fields) should never be stored more than once in a relational database? What types of data (or fields) usually must be stored more than once in a relational database?

Non-key fields should never be stored more than once.

If a table represents an entity, the primary key values of each entity represented in the table are redundantly stored (as foreign keys) for every relationship in which the entity participates.

22. What is relational database normalization? Why is a database schema in third normal form considered to be of higher quality than an unnormalized database schema?

Relational database normalization is a process that increases schema quality by minimizing data redundancy. A schema with tables in third normal form has less non-key data redundancy than a schema with unnormalized tables. Less redundancy makes the schema and database contents easier to maintain over the long term.

23. Describe the process of relational database normalization. Which normal forms rely on the definition of functional dependency?

The process of normalization modifies the schema and table definitions by successively applying higher order rules of table construction. The rules each define a normal form, and the normal forms are numbered one through three. First normal form eliminates repeating groups that are embedded in tables.

Second and third normal forms are based on a concept called *functional dependency*—a one-to-one correspondence between two field values. Second normal form ensures that every field in a table is functionally dependent on the primary key. Third normal form ensures that no non-key field is functionally dependent on any other non-key field.

24. Describe the steps used to transform a class diagram into an object database schema.

25. Determine which classes require persistent storage.
26. Define persistent classes within the schema.
27. Represent relationships among persistent classes.
28. Choose appropriate data types and value restrictions.

29. What is the difference between a persistent class and a transient class? Provide at least one example of each class type.

An object of a transient class exists only for the duration of a program execution (for example, the user interface of the program). An object of a persistent class (for example, a customer object in a billing system) retains its identity and data content between program executions.

30. What is an object identifier? Why are object identifiers required in an object database?

An object identifier is a key or storage address that uniquely identifies an object within an object-oriented database. Object identifiers are needed to represent relationships among objects. A relationship is represented by embedding the object identifier of a participating object in the other participating object.

31. How is a class on a class diagram represented in an object database?

A class on a class diagram is represented "as is" in an object database. That is, each object of the class type is stored in the database along with its data content and methods.

32. How is a one-to-many relationship on a class diagram represented in an object database?

The object identifier of each participating object is embedded in the other participating object. The object on the "one" side of the relationship might have multiple embedded object identifiers to represent multiple participants on the "many" side of the relationship.

33. How is a many-to-many relationship without attributes represented in an object database?

The object identifier of each participating object is embedded in the other participating object. The objects on both sides of the relationship might have multiple embedded object identifiers to represent multiple participants on the other side of the relationship.

34. What is an association class? How are association classes used to represent many-to-many relationships in an object database?

An association class is an "artificial" class that is created to represent a many-to-many relationship among "real" classes. The association class has data members that represent attributes of the many-to-many relationship. Each "real" class implements a one-to-many relationship with the association class.

35. Describe the two ways in which a generalization relationship can be represented in an object database.

Generalization relationships can be represented directly (for example, using the ODL keyword *extends*) or indirectly as a set of one-to-one relationships.

36. Does an object database require key fields or attributes? Why or why not?

Key fields aren't required because they aren't needed to represent relationships. However, they are usually included because they are useful for a number of reasons, including guaranteeing unique object content and searching or sorting database content.

37. Describe the similarities and differences between an ERD and a class diagram that models the same underlying reality.

Each entity on an ERD corresponds to one class on a class diagram. The one-to-one, one-to-many, and many-to-many relationships among those classes are the same as those on the ERD.

38. How are classes and relationships on a class diagram represented in a relational database?

A class is represented as a table.

A one-to-many relationship among classes is represented in the same way as a one-to-many among entities (see answer #11).

A many-to-many relationship among classes is represented in the same way as a many-to-many among entities (see answer #12). Note that the table that represents the relationships serves the same purpose as an association class.

39. What is the difference between a primitive data type and a complex data type?

A primitive data type (for example, integer, real, or character) is directly supported (represented) by the CPU or a programming language. A complex data type (for example, record, linked list, or object) contains one or more data elements constructed using the primitive data types as building blocks.

40. What are th advantages of having an RDBMS provide complex data types?

Providing complex data types in the RDBMS allows a wider range of data to be represented. It also minimizes compatibility problems that might result from using different programming languages or hardware.

41. Does an ODBMS need to provide predefined complex data types? Why or why not?

No. A required complex data type can be defined as a new class.

42. Why might all or part of a database need to be replicated in multiple locations?

Database accesses between distant servers and clients must traverse one or more network links. This can slow the accesses due to propagation delay or network congestion. Access speed can be increased by placing a database replica close to clients.

43. Briefly describe the following distributed database architectures: replicated database servers, partitioned database servers, and federated database servers. What are the comparative advantages of each?

Replicated database servers – An entire database is replicated on multiple servers, and each server is located near a group of clients. Best performance and fault tolerance for clients because all data is available from a "nearby" server.

Partitioned database servers – A database is partitioned so that each partition is a database subset used by a single group of clients. Each partition is located on a separate server, and each server is located close to the clients that access it. Better performance and less replication traffic than replicated servers if similar collocated clients use only a subset of database content.

Federated database servers – Data from multiple servers with different data models and/or DBMSs is pooled by implementing a separate (federated) server that presents a unified view of the data stored on all the other servers. The federated server constructs answers to client queries by forwarding requests to other servers and combining their responses for the client. Simplest and most manageable way to combine data from disparate DBMSs into a single unified data store.

44. What additional database management complexities are introduced when database contents are replicated in multiple locations?

Replicated copies are redundant data stores. Thus, any changes to data content must be redundantly implemented on each copy. Implementing redundant maintenance of data content requires all servers to periodically exchange database updates.

Chapter 14 – Designing the User Interface

Solutions to End-of-Chapter Material

Review Questions

11. Why is interface design often referred to as dialog design?

The user interface involves communication between the user and the computer, which is much like a dialog.

12. What are the three aspects of the system that make up the user interface for a user?

Physical, perceptual, and conceptual aspects.

13. What is the term generally used to describe the study of end users and their interaction with computers?

Human-computer interaction (HCI).

14. What are some examples of physical aspects of the user interface?

Keyboard, mouse, touch screen, reference manuals, documents, data entry forms.

15. What are some examples of perceptual aspects of the user interface?

Data on the screen, shapes, lines, numbers, words, beeps, clicks, menus, dialog boxes, icons, drawings.

16. What are some examples of conceptual aspects of the user interface?

Conceptual aspects of the user interface include everything the user knows about using the system, including all of the problem domain "things" in the system the user is manipulating, the operations that can be performed, and the procedures followed to carry out the operations.

17. What collection of techniques places the user interface at the center of the development process?

User-centered design.

18. What are the three important principles emphasized by user-centered design?

Focus early on users and their work, evaluate designs to ensure usability, use iterative development.

19. What term refers to the degree to which a system is easy to learn and use?

Usability.

20. What is it about the "human factor" that engineers find difficult? What is the solution to human factors problems?

The human factor is hard to predict. The solution is to design the machine to accommodate the user, rather than trying to change the user to accommodate the machine.

21. What are some of the fields that contribute to the field of human-computer interaction?

See Figure 14-2 on page 535 for a list of fields.

22. What research center significantly influenced the nature of the computers we use today?

Xerox Palo Alto Research Center (Xerox PARC).

23. What are the three metaphors used to describe human-computer interaction?

Direct manipulation metaphor, document metaphor, dialog metaphor.

24. A "desktop" on the screen is an example of which of the three metaphors used to describe human-computer interaction?

Direct manipulation (of objects on the desktop).

25. **What type of document allows the user to click on a link and jump to another part of the document?**
Hypertext.

26. **What type of document allows the user to click on links to text, graphics, video, and audio in a document?**
Hypermedia.

27. **What is the name for general principles and specific rules that must always be followed when designing the interface of a system?**
Interface design standards.

28. **What two key principles are proposed by Norman that ensure good interaction between a person and a computer?**
Visibility (controls should be visible and provide feedback to indicate the control is responding) and affordance (the appearance of a control should suggest its functionality).

29. **List the eight golden rules proposed by Shneiderman.**
See the list of rules in Figure 14-7.

30. **What is the technique that shows a sequence of sketches of the display screen during a dialog?**
Storyboarding.

31. **What UML diagram can be used to show how the interface objects are plugged in between the actor and the problem domain classes during a dialog?**
Sequence diagram and collaboration diagram.

32. **What are the three basic types of windows and browser forms used in business systems?**
Input forms, output forms, input/output forms.

33. **What are some of the input controls that can be used for selecting an item from a list?**
Listboxes, combo boxes, spin boxes.

34. **What two types of input controls are included in groups?**
Checkboxes and radio buttons (option buttons).

35. What are three requirements for usability of a direct customer access Web site beyond those of a windows interface used by employees?

Provide even more information, be even more flexible, and be easier to learn and use.

26. What is a popular analogy used for direct customer access with a Web site when customers shop online?

Shopping cart analogy.

27. What are three principles of Web design that guide designers?

A Web design book by Joel Sklar suggests that the designer should focus on three broad aspects of Web design: designing for the computer medium, designing the whole site, and designing for the user.

28. What are four of the 10 good deeds of Web design?

See pages 553 for a list of Nielsen's "Ten Good Deeds in Web Design."

Chapter 15 – Designing System Interfaces, Controls, and Security

Solutions to End-of-Chapter Material

Review Questions

1. **What does XML stand for? Explain how XML is similar to HTML. Also, discuss the differences between XML and HTML.**

   XML stands for eXtensible Markup Language.

   Like HTML, XML uses markup codes that are identified with brackets (< >) and that are embedded in an ASCII text file. The markup codes as well as the text file are in human-readable format.

   Unlike HTML, the markup codes in XML are not predefined. The language is extensible because the markup codes are self-defining. An additional file, called the Document Type Definition (DTD) file, is needed to define the meaning of the markup codes.

2. **Compare the strengths and weaknesses of using a DFD to define inputs with using a sequence diagram to define inputs. Which do you like the best? Why**?

   DFDs: The arrows on a DFD describe information flows. As such, they define logical inputs of data. The design of the inputs and outputs might be more detailed, requiring several screens.

   Sequence diagrams: The messages on a sequence diagram identify interactions. The parameters define the data that is passed. If the developer does not carefully define the parameters, the message might give an incomplete design. A sequence diagram also describes the various steps that are required to complete a business transaction, which might help define the set of required screens.

   Preferences will vary.

3. **Explain the system boundary. Why was one used on a DFD but not used on a system sequence diagram?**

   The system boundary divides the automated system from the manual system. It is used on a DFD because a DFD includes both manual and automated processes. To identify which data flows cross the boundary, a boundary must be superimposed on the diagram.

   The flow of information from actors to objects is by definition a flow of data from external to internal.

4. **What additional information does the structure chart provide that is not obtained from a DFD in the development of input forms?**

   A structure chart gives more detailed information about access to individual input screens or files. A structure chart provides more details about the set of individual forms that might be required to support an input data flow.

5. **How are the data fields identified using the structured approach?**

   Data fields primarily come from the data flows that cross the system boundary. These data flows can be checked by ensuring that adequate input data is available to support the fields in the data stores.

6. **How are the data fields identified using UML and the object-oriented approach?**

   The primary source of data fields is the class diagram. Fields are identified in the classes. Input messages must contain enough data fields to support the required modifications to data fields in the internal classes.

7. **Explain four types of integrity controls for input forms. Which have you seen most frequently? Why are they important?**

   36. Field combination controls verify that the data in one field is based on the data in another field or fields.
   37. Value limit controls identify when a value in a field is too large or too small.
   38. Completeness controls ensure that all necessary fields on an input form have been entered.
   39. Data validation controls validate the input data for correctness.

   Answers will vary.

8. **What protection does transaction logging provide? Should it be included in every system?**

   Transaction logging provides an audit trail to track the changes that were made, when they were made, and who made them. It also provides an effective method for backup and recovery in case the primary data file is destroyed.

Transaction logging should be included in every system that contains financial information. Other types of systems that are not as critical do not require logging.

9. What are the different considerations for output screen design and output report design?

Output screens are more dynamic but have limited information available at one time. It is harder to view multiple pages at the same time with screen output. However, dynamic features, such as drill down, can be provided so that summary information does not have to stand alone.

Printed output is more permanent. Consequently, it should always include identifying fields, such as date printed. Because reports are not dynamic, they must be self-contained and include all necessary information to be understandable.

10. What is meant by *drill down*? Give an example of how you might use it in a report design.

Drill down is a technique that links a summary field to its supporting detail and enables users to view the detail dynamically.

In financial reports, totals or summary amounts can include drill-down links to supporting accumulations. Textual reports can include drill-down links to more detailed explanations of critical terms.

11. What is the danger from information overload? What solutions can you think of?

Information overload can cause users to miss important facts, such as exception conditions. Users can also become discouraged when they are unable to find the information they need within reams of unimportant data.

Solutions generally include identifying the information that is important and highlighting it using color or graphics, or by visually separating it from the other data.

12. Describe the kinds of integrity controls you would recommend to place on all output reports. Why?

Date printed, processing date, page numbers, titles, form numbers, routing information, end-of-report notification, and control totals and footings.

It is usually easy to understand the report and its data when the report is first printed. However, reports are often long-term, and reviewing a report that was printed a week or month ago requires this control information. Frequently, there are multiple copies of the same type of report, and this control information is necessary to distinguish one day's report from another day's report.

13. **What are the objectives of integrity controls in information systems? Explain what each of the three objectives mean. Give an example of each.**
    - Ensure that only appropriate and correct business transactions occur. This objective ensures that no erroneous or fraudulent transactions are entered. Example: A control to ensure that a clerk does not request a check for a service that was never provided.

    - Ensure that the transactions are recorded and processed correctly. This objective ensures that the system processes and stores the data completely. Example: a control to ensure that a double-entry bookkeeping entry always processes both entries.

    - Protect and safeguard the assets (including information) of the organization. This objective ensures that information is not lost due to theft, fire, or some other mishap. Example: Storing backup data periodically offsite.

14. **What are four types of input controls used to reduce input errors? Describe how each works.**

    Field combination control: An integrity control that verifies that the data in one field is based on the data in another field or fields.

    Value limit control: An integrity control that identifies when a value in a field is too large or too small.

    Completeness control: An integrity control that ensures that all necessary fields on an input form have been entered.

    Data validation control: An integrity control that validates the input data for correctness and appropriateness.

15. **Explain what is meant by update controls for a database management system.**

    Update controls prohibit multiple programs from simultaneously updating the same fields or records in the database, which can result in overwriting or destroying data. Update controls maintain database integrity by ensuring that either all or none of the updates are completed.

16. **What is the basic purpose of transaction logging? Microsoft Access does not have automatic transaction logging. Is this a deficiency, or is it not really an important consideration in database integrity?**

Transaction logging takes every update to the database and logs exactly how it happened (sometimes with an image of the transaction). It is extremely important for audit trails and for recovery in case something goes wrong.

Transaction logging is usually not performed in databases that are small and not important to the business. For this reason, MS Access should not be used for important or mission-critical databases.

17. **On a printed output report, what is the difference between the date the report was printed and the date of the data?**

Every report should have a date and time stamp, both for the time the report was printed and for the date of the underlying data. It is important to distinguish between the two dates. A report that is printed in November, for example, might contain data from August, September, or October. Therefore, if you did not know the date of the underlying data in the report, you might mistakenly assume that it was from November, the date the report was actually printed.

18. **What are the two primary objectives of security controls?**

Maintain a stable, functioning operating environment for users and application systems (usually 24 hours a day, seven days a week).

Protect information and transactions during transmission outside the organization (public carriers).

19. **Explain the three categories of user access privileges. Is three the right number, or should there be more or fewer than three? Why or why not?**

Unauthorized, registered, and privileged. Unauthorized users do not have any access rights to the system. Registered users have different levels of access rights to the system. Some users might have rights to update data fields while others can only see them. Managers might have access to sensitive information that is not available to clerks. Privileged users are those who can access the security system and other control systems. There are also various levels of privileged users.

These three categories cover all the needs. The registered and privileged user categories can include several different levels of access privileges, making it unnecessary to create additional categories.

20. **How does single-key (symmetric) encryption work? What are its strengths? What are its weaknesses?**

A single key is used to encrypt and decrypt a message. Both parties must have the key. Its strength is that it is simple and fast. Its weaknesses are that it might be easy to break the encryption and that it is difficult to distribute the key in a secret fashion to all the authorized participants.

21. How does public-key (asymmetric) encryption work? What are its strengths? What are its weaknesses?

A public-key encryption has two keys, a public one that is widely distributed and a private one that is secret. To send data to the owner of the keys, someone uses the public key. The data can then only be decrypted with the private key. So, the owner is the only one who can decrypt the data. After the message is encrypted, it can only be decrypted with the private key.

22. What is a digital certificate? What role do certifying authorities play in security systems?

A digital certificate is an institution's name and public key (plus other information such as address, Web site URL, and validity date of the certificate) that is encrypted and certified by a third party.

Certifying authorities are companies that are very well known so that everybody knows for sure what their public keys are. These certifying authorities sell digital certificates to other companies (that are not as well known) so that these companies can convince their customers that they are legitimate.

23. What is a digital signature? What does it tell a user?

A digital signature is a technique in which a document is encrypted using a private key to verify who wrote the document. If you have the public key of an entity, and that entity sends you a message with its private key, you can decode it with the public key. You know that the party is the one you want to communicate with because that entity is the only one who can encode a message with that private key.

Chapter 17 – Current Trends in System Development

Solutions to End-of-Chapter Material

Review Questions

1. **Identify the five important principles and practices that are driving many of the current trends in software development. Briefly explain each.**

    - **Abstraction** – The process of extracting and distilling core principles from a set of facts or statements. Abstraction is important in the field of computing, especially for the development of metamodels, which help to generalize and standardize the construction of models.
    - **Models and modeling** – Abstractions of something real, with a particular set of properties, such as a business process or thing.
    - **Pattern** – A standard solution to a given problem, or a template that can be applied to a problem.
    - **Reuse** – The principle of developing and using standard solutions and components that can be used over and over.
    - **Methodology** – A process, including rules, guidelines, and techniques, that defines how systems are built.

2. **What are the driving forces that are moving many companies to adopt more adaptive approaches to system development?**

    The historically low success rate of system development projects and the volatility of today's business environment.

3. **Explain the difference between a predictive control process and an empirical control process.**

    A predictive control process defines the steps to monitor a process in great detail and works well when more planning is possible and can provide more detail. An empirical control process describes processes that are variable and unpredictable and are best controlled by handling each variation as it occurs and determining the best way to correct the deviation on the fly.

4. List the six fundamental characteristics of adaptive projects.

- Less emphasis on up-front analysis, design, and documentation.
- More focus on incremental development.
- More user involvement in project teams.
- Reduced detailed planning, which is used for near-term work phases only; downstream phases might have high-level plans.
- Tightly controlling schedules by fitting work into discrete time boxes.
- More use of small work teams that are self-organizing.

5. What are the elements of the "Manifesto for Agile Software Development?" Explain what each means.

37. **Responding to change over following a plan** – Accept that software development is unpredictable and should be approached with flexibility.
38. **Individuals and interactions over processes and tools** – Rely on teams to plan and execute their work as the project progresses.
39. **Working software over comprehensive documentation** – The ultimate aim of the project is working software, and this aim should always be kept in sight.
40. **Customer collaboration over contract negotiation** – Customers must continually be involved with the project team, and contracts should take more of a collaborative track.

6. What does *chaordic* mean? What implications does it have for development projects?

Chaordic is a term used to describe adaptive projects, which are both chaotic and ordered. Developers need to accept the chaos, but they also need to use specific methodologies to organize and impose order.

7. What are the four UP phases, and what is the objective of each?

1. **Inception** – Develop an approximate vision of the system, make the business case, define the scope, and produce rough estimates for cost and schedule.

2. **Elaboration** – Refine the vision, identify and describe all requirements, finalize the scope, design and implement the core architecture and functions, resolve high risks, and produce realistic estimates for cost and schedule.

3. **Construction** – Iteratively implement the remaining lower-risk, predictable, and easier elements and prepare for deployment.

4. **Transition** – Complete the beta test and deployment so users have a working system and are ready to benefit as expected.

8. What are the six UP development disciplines?

Business modeling, requirements, design, implementation, test, and deployment.

9. What are the three UP support disciplines?

Configuration and change management, project management, and environment.

10. List the basic principles of Agile Modeling.

1. Develop software as your primary goal.
2. Enable the next effort as your secondary goal.
3. Minimize your modeling activity—few and simple.
4. Embrace change, and change incrementally.
5. Model with a purpose.
6. Build multiple models.
7. Build high-quality models and get feedback rapidly.
8. Focus on content rather than representation.
9. Learn from each other with open communication.
10. Know your models and how to use them.
11. Adapt to specific project needs.

11. Why is the word *extreme* included as part of Extreme Programming?

The intent of XP is to extend the best practices of software development "to the extreme."

12. List the core values of XP.

- Communication
- Simplicity
- Feedback
- Courage

13. List the XP practices.

1. Planning
2. Testing
3. Pair programming
4. Simple designs
5. Refactoring the code
6. Owning the code collectively
7. Continuous integration

8. On-site customer
9. System metaphor
10. Small releases
11. Forty-hour week
12. Coding standards

14. What is the product backlog used for in a Scrum project?

The product backlog is a list of all the things the system should include and address, including user functions, features, and technology. The product backlog is continually being prioritized and is controlled by the product owner.

15. Explain how a Scrum sprint works.

A Scrum sprint is a firm 30-day time box with a specific goal or deliverable. At the beginning, the team gathers for a one-day planning session, during which they decide on the major goal for the sprint, drawing from the prioritized product backlog. The scope of that sprint is then frozen. Team members meet every day to report progress. At the end of the sprint, the agreed-upon deliverable is produced.

16. Explain the difference in project time management and project scope management for projects using agile methods.

Project time management changes because the team does not attempt to make a complete, detailed project schedule. Instead, time and the schedule are managed incrementally, through iterations, sprints, and so on. Project scope management changes because users or clients become part of the team, and they are given the responsibility of defining the project scope.

17. What is a PIM? What is a PSM? How are they related?

A PIM is a platform-independent model that describes system characteristics that are not specific to any deployment platform. An example is a UML class diagram. A PSM is a model that describes system characteristics that include deployment platform requirements. PIMs can be used to define PSMs through the design discipline under MDA. PSMs can also be used at some point to move from code to a PIM.

18. What are the potential benefits of Model-Driven Architecture?

MDA provides a mechanism by which organizations can extract critical features and information about each of their current systems and combine them into a PIM. This capability allows an organization to analyze the combined PIM to determine where duplication, inconsistencies, and conflicts in technologies exist. In addition, new systems can be designed to conform to existing systems.

19. What is a metamodel? How is a metamodel used?

A metamodel is a model that describes the characteristics of another model. It is used to abstract the properties and components of a model such as a class diagram.

20. What is an object framework? How is it different from a library of components?

An object framework is a library of foundation classes designed for maximal reuse via instantiation or inheritance. It differs from a component library in that object frameworks are source code, and components are executable code.

21. For which layers of an OO program are off-the-shelf components most likely to be available?

Components are most readily available for the view layer (or user interface).

22. What is a software component?

A software component is a module of executable code that can be easily integrated with other software modules to form a complete system.

23. Why have software components only recently come into widespread use?

The technology and standards to support widespread component use and integration have been created and widely deployed only in the last decade.

24. In what ways do components make software construction and maintenance faster?

Systems can be assembled from previously developed components, thus eliminating many custom-software development steps. Maintenance is also faster because a single component can be replaced without rewriting, recompiling, relinking, or redistributing the entire application.

25. What is a service? How does a service differ from a component? How are services similar to components?

A service is a computer function that an application interacts with through the Internet. Components differ in that they are dynamically or statically linked to an application before execution. They are similar in that they share the goal of reuse to shorten development time.

26. On what standards is service-oriented architecture based?

The Microsoft .NET service standard is based on SOAP. The Java 2 Web Services (J2WS) standard extends SOAP to create a Java-specific implementation