|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 1** |  |
| **Agile Development** | an information systems development process that emphasizes flexibility to anticipate new requirements during development |
| **computer application (app)** | a computer software program that executes on a computing device to carry out a specific function or set of related functions |
| **information system** | a set of interrelated computer components that collects, processes, stores, and provides as output the information needed to complete business tasks |
| **information systems development process** | the actual approach used to develop a particular information system |
| **iterative development** | an approach to system development in which the system is “grown” piece by piece through multiple iterations |
| **project** | a planned undertaking that has a beginning and an end, and that produces some definite result |
| **subsystem** | an identifiable and partitioned portion of an overall system |
| **systems analysis** | those activities that enable a person to understand and specify what the new system should accomplish |
| **systems design** | those activities that enable a person to define and describe in detail the system that solves the need |
| **Systems Development Life Cycle (SDLC)** | the entire process consisting of all the activities required to build, launch, and maintain an information system |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 2** |  |
| **activity diagram** | describes user (or system) activities, the person who does each activity, and the sequential flow of these activities |
| **application architecture** | the organization and construction of software resources to implement an organization’s information systems |
| **client** | person or group that provides the funding for a system development project |
| **closed-ended questions** | questions that elicit specific facts |
| **design constraints** | restrictions to which the hardware and software must adhere |
| **executive stakeholders** | persons who don’t interact directly with the system but who either use information produced by the system or have a significant financial or other interest in its operation and success |
| **external stakeholders** | persons outside the organization’s control and influence who interact with the system or have a significant interest in its operation or success |
| **functional requirements** | the activities that the system must perform |
| **FURPS** | a requirements classification framework (acronym stands for functionality, usability, reliability, performance, and security) |
| **FURPS+** | an extension of FURPS that includes design constraints as well as implementation, interface, physical, and supportability requirements |
| **graphical models** | system models that use pictures and other graphical elements |
| **implementation requirements** | constraints such as required programming languages and tools, documentation method and level of detail, and a specific communication protocol for distributed components |
| **interface requirements** | required interactions among systems |
| **internal stakeholders** | persons within the organization who interact with the system or have a significant interest in its operation or success |
| **mathematical models** | system models that describes requirements numerically or as mathematical expressions |
| **model** | representation of some aspect of a system |
| **nonfunctional requirements** | system characteristics other than the activities it must perform or support |
| **open-ended questions** | questions that encourage discussion or explanation |
| **operational stakeholders** | persons who regularly interact with a system in the course of their jobs or lives |
| **performance requirements** | operational characteristics related to measures of workload, such as throughput and response time |
| **physical requirements** | characteristics of hardware such as size, weight, power consumption, and operating conditions |
| **reliability requirements** | requirements that describe system dependability |
| **security requirements** | requirements that describe how access to the application will be controlled and how data will be protected during storage and transmission |
| **stakeholders** | persons who have an interest in the successful implementation of the system |
| **supportability requirements** | how a system is installed, configured, monitored, and updated |
| **swimlane heading** | activity diagram column containing all activities for a single agent or organizational unit |
| **synchronization bar** | activity diagram component that either splits a control path into multiple concurrent paths or recombines concurrent paths |
| **system requirements** | the activities a system must perform or support and the constraints that the system must meet |
| **technology architecture** | a set of computing hardware, network hardware and topology, and system software employed by an organization |
| **textual models** | text-based system models such as memos, reports, narratives, and lists |
| **Unified Modeling Language (UML)** | standard set of model constructs and notations defined by the Object Management Group |
| **usability requirements** | operational characteristics related to users, such as the user interface, related work procedures, online help, and documentation |
| **workflow** | sequence of processing steps that completely handles one business transaction or customer request |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Glossary** |  | | **Chapter 3** |  | | **<> relationship** | a relationship between use cases in which one use case is stereotypically included within the other use case | | **actor** | an external agent; a person or group that interacts with the system by supplying or receiving data | | **automation boundary** | the boundary between the computerized portion of the application and the users who operate the application but are part of the total system | | **brief use case description** | an often one-sentence description that provides a quick overview of a use case | | **CRUD technique** | an acronym for Create, Read/Report, Update, and Delete; a technique to validate or refine use cases | | **elementary business processes (EBPs)** | the most fundamental tasks in a business process, which leaves the system and data in a quiescent state; usually performed by one person in response to a business event | | **event** | something that occurs at a specific time and place, can be precisely identified, and must be remembered by the system event. | | **event decomposition technique** | a technique to identify use cases by determining the external business events to which the system must respond | | **external event** | an event that occurs outside the system, usually initiated by an external agent | | **perfect technology assumption** | the assumption that a system runs under perfect operating and technological conditions | | **state event** | an event that occurs when something happens inside the system that triggers some process | | **system controls** | checks or safety procedures to protect the integrity of the system and the data | | **temporal event** | an event that occurs as a result of reaching a point in time | | **use case** | an activity that the system performs, usually in response to a request by a user | | **use case diagram** | the UML model used to graphically show use cases and their relationships to actors | | **user goal technique** | a technique to identify use cases by determining what specific goals or objectives must be completed by a user | |  |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 4** |  |
| ***n*-ary association** | an association between n distinct types of things |
| **abstract class** | a class that describes a category or set of objects but that never includes individual objects or instances |
| **aggregation** | a type of whole-part relationship in which the component parts also exist as individual objects apart from the aggregate |
| **association** | a term, in UML, that describes a naturally occurring relationship between specific things, sometimes called a relationship |
| **association class** | an association that is also treated as a class; often required in order to capture attributes for the association |
| **attributes** | descriptive pieces of information about things or objects |
| **binary associations** | associations between exactly two distinct types of things |
| **brainstorming technique** | a technique to identify problem domain objects in which developers work with users in an open group |
| **camelback notation or camelcase notation** | when words are concatenated to form a single word and the first letter of each embedded word is capitalized |
| **cardinality** | a measure of the number of links between one object and another object in a relationship |
| **class** | a category or classification of a set of objects or things |
| **class diagram** | a diagram consisting of classes (i.e., sets of objects) and associations among the classes |
| **composition** | a type of whole-part relationship in which the component parts cannot exist as individual objects apart from the total composition |
| **compound attribute** | an attribute that consists of multiple pieces of information but is best treated in the aggregate |
| **concrete class** | a class that allows individual objects or instances to exist |
| **data entities** | the term used in an ER diagram to describe sets of things or individual things |
| **domain classes** | classes that describe objects from the problem domain |
| **domain model class diagram** | a class diagram that only includes classes from the problem domain |
| **entity-relationship diagram (ERD)** | a diagram consisting of data entities (i.e., sets of things) and their relationships |
| **generalization/specialization relationship** | a type of hierarchical relationship in which subordinate classes are subsets of objects of the superior classes; an inheritance hierarchy |
| **identifier or key** | an attribute the value of which uniquely identifies an individual thing or object |
| **inheritance** | the concept that specialization classes inherit the attributes of the generalization class |
| **multiplicity** | a measure, in UML, of the number of links between one object and another object in an association |
| **multiplicity constraints** | the actual numeric count of the constraints on objects allowed in an association |
| **noun technique** | a technique to identify problem domain objects by finding and classifying the nouns in a dialog or description setting |
| **problem domain** | the specific area (or domain) of the user’s business need (or problem) that is within the scope of the new system |
| **relationship** | a term that describes a naturally occurring association between specific things, sometimes called an association |
| **semantic net** | a graphical representation of an individual data entity and its relationship with other individual data entities |
| **subclass** | the subordinate or more specialized class in a generalization/specialization relationship |
| **superclass** | the superior or more general class in a generalization/specialization relationship |
| **ternary association** | an association between exactly three distinct types of things |
| **unary association** | an association between two instances of the same type of thing |
| **whole-part relationship** | a relationship between classes in which one class is a part or a component portion of another class |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 5** |  |
| **action-expression** | a description of the activities performed as part of a transition |
| **alt frame** | notation on a sequence diagram showing if-then-else logic |
| **composite state** | a state containing other states and transitions (that is, a path) |
| **concurrency or concurrent state** | the condition of being in more than one state at a time |
| **destination state** | for a particular transition, the state to which an object moves after the completion of a transition |
| **guard-condition** | a true/false test to see whether a transition can fire |
| **interaction diagram** | either a communication diagram or a sequence diagram that shows the interactions between objects |
| **lifeline or object lifeline** | the vertical line under an object on a sequence diagram to show the passage of time for the object |
| **loop frame** | notation on a sequence diagram showing repeating messages |
| **opt frame** | notation on a sequence diagram showing optional messages |
| **origin state** | for a particular transition, the original state of an object from which the transition occurs |
| **path** | a sequential set of connected states and transitions |
| **postcondition** | what must be true upon the successful completion of a use case |
| **precondition** | a condition that must be true before a use case begins |
| **pseudostate** | the starting point of a state machine diagram, indicated by a black dot |
| **scenarios or use case instances** | unique sets of internal activities within use cases |
| **state** | a condition during an object’s life when it satisfies some criterion, performs some action, or waits for an event |
| **state machine diagram** | a diagram showing the life of an object in states and transitions |
| **system sequence diagram (SSD)** | a diagram showing the sequence of messages between an external actor and the system during a use case or scenario |
| **transition** | the movement of an object from one state to another state |
| **true/false condition** | part of a message between objects that is evaluated prior to transmission to determine whether the message can be sent |
| **use case description** | a textual model that lists and describes the processing details for a use case |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 6** |  |
| **architectural design** | broad design of the overall system structure; also called general design or conceptual design |
| **business logic layer or domain layer** | the part of a three-layer architecture that contains the programs that implement the business rules and processes |
| **client computers** | the computers at which the users work to perform their computational tasks |
| **client-server architecture** | a computer network configuration with user’s computers and central computers that provide common services |
| **cloud computing** | an extension of virtual servers in which the resources available include computing, storage, and Internet access and appear to have unlimited availability |
| **colocation** | a hosting service with a secure location but in which the computers are usually owned by the client businesses |
| **content delivery network (CDN)** | a set of server computers, separate from the hosting computers, used to deliver such static content as images or videos |
| **data layer** | the part of a three-layer architecture that interacts with the data |
| **detail design** | low-level design that includes the design of the specific program details |
| **hosting** | the process of providing physical servers at a secure location and selling those services to other businesses that wish to deploy Web sites |
| **Hypertext Markup Language (HTML)** | the predominant language for constructing Web pages and which consists of tags and rules about how to display pages |
| **Hypertext Transfer Protocol Secure (HTTPS)** | an encrypted form of information transfer on the Internet that combines HTTP and TLS |
| **local area network (LAN)** | a computer network in which the cabling and hardware are confined to a single location |
| **network diagram** | a model that shows how the application is deployed across networks and computers |
| **peer-to-peer connection** | when independent computers communicate and share resources without the need of a centralized server computer |
| **server computer** | the central computer that provides services (such as database access) to the client computers over a network |
| **Service Level Agreement (SLA)** | part of the contract between a business and a hosting company that guarantees a specific level of system availability |
| **three-layer architecture** | a client/server architecture that divides an application into view layer, business logic layer, and data layer |
| **Transmission Control Protocol/ Internet Protocol (TCP/IP)** | the foundation protocol of the Internet; used to provide reliable delivery of messages between networked computers |
| **Transport Layer Security (TLS)** | an advanced version of Secure Sockets Layer (SSL) protocol used to transmit information over the Internet securely |
| **view layer** | the part of the three-layer architecture that contains the user interface |
| **Virtual Private Networks (VPNs)** | a closed network with security and closed access built on top of a public network, such as the Internet |
| **virtual server** | a method to partition the services of a physical Web server so it appears as multiple, independent Internet servers |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 7** |  |
| **affordance** | when the appearance of a specific control suggests its function |
| **assistive technologies** | software (such as text-to-speech and voice-recognition utilities) that adapts user interfaces to the special needs of persons with disabilities |
| **cascading style sheets (CSS)** | Web page encoding standard that enables a Web site designer to specify parts of a page that will always look the same and parts that will vary by task or audience |
| **check boxes** | similar to radio buttons, but the user can select multiple items within the group |
| **combo box** | a text box that contains a predefined list of acceptable entries but permits the user to enter a new value when the list doesn’t contain the desired value |
| **desktop metaphor** | metaphor in which the visual display is organized into distinct regions, with a large empty workspace in the middle and a collection of tool icons around the perimeter |
| **detailed reports** | reports that contain specific information on business transactions |
| **dialog metaphor** | metaphor in which user and computer accomplish a task by engaging in a conversation or dialog via text, voice, or tools such as labeled buttons |
| **direct manipulation metaphor** | metaphor in which objects on a display are manipulated to look like physical objects (pictures) or graphic symbols that represent them (icons) |
| **document metaphor** | metaphor in which data is visually represented as paper pages or forms |
| **drill down** | user-interface design technique that enables a user to select summary information and view supporting detail |
| **exception reports** | reports that provide details or summary information about transactions or operating results that fall outside a predefined normal range of values |
| **executive reports** | reports used by highlevel managers to assess overall organizational health and performance |
| **Extensible Markup Language (XML)** | extension of HTML that embeds self-defining data structures within textual Messages |
| **external outputs** | reports or other outputs produced for use by people outside the organization |
| **human-computer interaction (HCI)** | field of study concerned with the efficiency and effectiveness of user interfaces vis-à-vis computer systems, human-oriented input and output technology, and psychological aspects of user interfaces |
| **internal outputs** | reports or other outputs produced for use within the organization |
| **list box** | a text box that contains a list of predefined data values |
| **metaphors** | analogies between features of the user interface and aspects of physical reality that users are familiar with |
| **peer-to-peer connection** | when independent computers communicate and share resources without the need of a centralized server computer |
| **radio buttons** | a group of choices from which the user selects only one; the system then automatically turns off all other buttons in the group |
| **storyboarding** | sequence of sketches of the display screen during a dialog |
| **summary reports** | reports that summarize detail or recap periodic activity |
| **system interfaces** | inputs or outputs that require minimal human intervention |
| **text box** | a rectangular box that accepts text typed on a keyboard or recognized from speech input |
| **turnaround documents** | external outputs that includes one or more parts intended to be returned with new data or information |
| **usability** | degree to which a system is easy to learn and use |
| **user interfaces** | system interfaces that directly involve a system user |
| **user-centered design** | design techniques that embody the view that theuser interface is the entire system |
| **visibility** | when a control is visible so that users know it is available, the control providing immediate feedback to indicate that it is responding to the user |
| **XML tags** | character sequences (such as and ) that define the beginning, end, and meaning of the text that appears between them |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 8** |  |
| **adaptive approach to the SDLC** | an approach that assumes the project must be more flexible and adapt to changing needs as the project progresses |
| **agile development** | a guiding philosophy and set of guidelines for developing information systems in an unknown, rapidly changing environment |
| **agile modeling (AM)** | a guiding philosophy in which only models that are necessary, with a valid need and at the right level of detail, are created |
| **chaordic** | a term used to describe adaptive projects that are chaotic and ordered |
| **data flow diagram (DFD)** | a structured analysis model showing inputs, processes, storage, and outputs of a system |
| **help desk** | the availability of support staff to assist users with technical or processing problems of the information system |
| **incremental development** | an SDLC approach that completes portions of the system in small increments across iterations, with each increment being integrated into the whole as it is completed |
| **integrated development environments (IDEs)** | a set of tools that work together to provide a comprehensive development and programming environment for software developers |
| **object** | a thing in an information system that responds to messages by executing functions or methods |
| **object-oriented analysis (OOA)** | the process of identifying and defining the use cases and the sets of objects (classes) in the new system |
| **object-oriented approach** | system development based on the view that a system is a set of interacting objects that work together |
| **object-oriented design (OOD)** | defining all of the types of objects necessary to communicate with people and devices in the system, showing how objects interact to complete tasks, and refining the definition of each type of object so it can be implemented with a specific language or environment |
| **object-oriented programming (OOP)** | programming using object-oriented languages that support object classes, inheritance, reuse, and encapsulation |
| **phases** | related groups of development activities, such as planning, analysis, design, implementation, and support |
| **predictive approach to the SDLC** | an approach that assumes the project can be planned in advance and that the new information system can be developed according to the plan |
| **spiral model** | an adaptive SDLC approach that cycles over and over again through development activities until completion |
| **structure chart** | a graphical diagram showing the hierarchical organization of modules |
| **structured analysis** | a technique to determine what processing is required and to organize those requirements by using structured analysis models |
| **structured approach** | system development using structured analysis, structured design, and structured programming techniques |
| **structured design** | the design process of organizing a program into a set of modules and organizing those modules into a hierarchical structure |
| **structured programming** | a programming approach where each module has one start point and one end point and uses sequence, decision, and repetition constructs only |
| **support activities** | the activities in the support phase whose objective is to maintain and enhance the system after it is installed and in use |
| **system development methodology** | a set of comprehensive guidelines for the SDLC that includes specific models, tools, and techniques |
| **technique** | guidelines to specify a method for how to carry out a development activity or task |
| **tool** | a software application that assists developers in creating models or other components required for a project |
| **top-down programming** | the concept of dividing a complex program into a hierarchy of program modules |
| **visual modeling tools** | tools that help analysts create and verify graphical models and may also generate program code |
| **walking skeleton** | a development approach in which the complete system structure is built but with bare-bones functionality |
| **waterfall model** | an SDLC approach that assumes the phases can be completed sequentially with no overlap |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 9** |  |
| **break-even point** | the point in time at which dollar benefits offset dollar costs |
| **business benefits** | the benefits that accrue to the organization; usually measured in dollars |
| **ceremony** | the level of formality of a project; the rigor of holding meetings and producing documentation |
| **client** | the person or group that funds the project |
| **cost/benefit analysis** | process of comparing costs and benefits to see whether investing in a new system will be beneficial |
| **critical path** | a sequence of tasks that can’t be delayed without causing the entire project to be delayed |
| **detailed work schedule** | the schedule that lists, organizes, and describes the dependencies of the detailed work tasks |
| **Gantt chart** | a bar chart that portrays the schedule by the length of horizontal bars superimposed on a calendar |
| **intangible benefit** | a benefit that accrues to an organization but that can’t be measured quantitatively or estimated accurately |
| **net present value (NPV)** | the present value of dollar benefits and dollar costs of a particular investment |
| **oversight committee** | clients and key managers who review the progress and direct the project |
| **payback period** | the time period after which the dollar benefits have offset the dollar costs |
| **project iteration schedule** | the list of iterations and use cases or user stories assigned to each iteration |
| **project management** | organizing and directing other people to achieve a planned result within a predetermined schedule and budget |
| **project management body of knowledge (PMBOK)** | a project management guide and standard of fundamental project management principles |
| **retrospective** | a meeting held by the team at the end of an iteration to determine what was successful and what can be improved |
| **system capabilities** | the required capabilities of a new system; part of a System Vision Document |
| **System Vision Document** | a document to help define the scope of a new system |
| **tangible benefit** | a benefit that can be measured or estimated in terms of dollars |
| **users** | the person or group of people who will use the new system |
| **work breakdown structure (WBS)** | the list or hierarchy of activities and tasks of a project; used to estimate the work to be done and to create a detailed work schedule |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 10** |  |
| **abstract class** | a class that can never be instantiated (no objects can be created of this type) |
| **application program interface (API)** | the set of public methods that are available to the outside world |
| **boundary class, or view class** | a class that exists on a system’s automation boundary, such as an input window |
| **class-level attribute** | an attribute that contains the same value for all objects in the system |
| **class-level method** | a method that is associated with a class instead of with objects of the class |
| **cohesion** | a qualitative measure of the focus or unity of purpose within a single class |
| **component diagram** | a type of implementation diagram that shows the overall system architecture and the logical components within it |
| **concrete class** | a class that can be instantiated (objects can be created of this type) |
| **control class** | a class that mediates between boundary classes and entity classes, acting as a switchboard between the view layer and domain layer |
| **coupling** | a qualitative measure of how closely the classes in a design class diagram are linked |
| **CRC cards** | a brainstorming technique for designing interactions in use cases by assigning responsibilities and collaborations for classes |
| **data access class** | a class that is used to retrieve data from and send data to a database |
| **enterprise-level system** | a system that has shared resources among multiple people or groups in an organization |
| **entity class** | a design identifier for a problem domain class |
| **indirection** | a design principle in which an intermediate class is placed between two classes to decouple them but still link them |
| **instantiation** | creation of an object based on the template provided by the class definition |
| **method signature** | a notation that shows all the information needed to invoke, or call, the method |
| **navigation visibility** | a design principle in which one object has a reference to another object and thus can interact with it |
| **object responsibility** | a design principle in which objects are responsible for carrying out system processing |
| **persistent class** | an entity class whose objects exist after a system is shut down |
| **protection from variations** | a design principle in which parts of a system that are unlikely to change are segregated from those that will |
| **stereotype** | a way of categorizing a model element by its characteristics, indicated by guillemets (<< >>) |
| **visibility** | a notation that indicates (by plus or minus sign) whether an attribute can be directly accessed by another object |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 11** |  |
| **activation lifeline** | a representation of the period during which a method of an object is alive and executing |
| **communication diagrams** | type of interaction diagram that emphasizes the objects that send and receive messages for a specific use case |
| **dependency relationship** | a relationship between packages, classes, or use cases in which a change in the independent item requires a change in the dependent item |
| **design patterns** | standard design techniques and templates that are widely recognized as good practice |
| **persistent classes** | problem domain classes that must be remembered between program executions (i.e., require storage in a database) |
| **separation of responsibilities** | a design principle that recommends segregating classes into separate components based on the primary focus of the classes |
| **sequence diagrams** | type of interaction diagram that emphasizes the sequence of messages sent between objects for a specific use case |
| **use case realization** | the process of elaborating the detailed design with interaction diagrams for a particular use case |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 12** |  |
| **access control** | a control that restricts which persons or programs can add, modify, or view information resources |
| **access control list** | a list attached or linked to a specific resource that describes users or user groups and the nature of permitted access |
| **asymmetric key encryption** | encryption method that uses different keys to encrypt and decrypt the data |
| **attribute** | one vertical group of data attribute values in a table |
| **attribute value** | the value held in a single table cell |
| **authentication** | the process of identifying users who request access to sensitive resources |
| **authorization** | the process of allowing or restricting a specific authenticated user’s access to a specific resource based on an access control list |
| **certifying authority** | a widely accepted issuer of digital certificates |
| **cloud-based database server architecture** | use of a cloud computing service provider to provide some or all database services |
| **completeness control** | a control that ensures that all required data values describing an object or transaction are present |
| **complex data type** | combinations of or extensions to primitive data types that are supported by programming languages, operating systems, and DBMSs |
| **complex update control** | a control that prevents errors that can occur when multiple programs try to update the same data at the same time or when recording a single transaction requires multiple related database updates |
| **data type** | the storage format and allowable content of a program variable, class attribute, or relational database attribute or column |
| **data validation control** | a control that ensures that numeric fields that contain codes or identifiers are correct |
| **database (DB)** | an integrated collection of stored data that is centrally managed and controlled |
| **database management system (DBMS)** | a system software component that manages and controls one or more databases |
| **database synchronization** | updating one database copy with changes made to other database copies |
| **decryption** | the process of converting encrypted data back to their original state |
| **digital certificate** | an institution’s name and public key (plus other information, such as address, Web site URL, and validity date of the certificate) encrypted and certified by a third party |
| **digital signature** | a technique in which a document is encrypted by using a private key to verify who wrote the document |
| **encryption** | the process of altering data so unauthorized users can’t view them |
| **encryption algorithm** | a complex mathematical transformation that encrypts or decrypts binary data |
| **encryption key** | a binary input to the encryption algorithm—typically a long string of bits |
| **field combination control** | a control that reviews combinations of data inputs to ensure that the correct data are entered |
| **first normal form (1NF)** | restriction that all rows of a table must contain the same number of columns |
| **foreign key** | an attribute that duplicates the primary key of a different (or foreign) table |
| **fraud triangle** | model of fraud that states that opportunity, motivation, and rationalization must all exist for a fraud to occur |
| **functional dependency** | a one-to-one association between the values of two attributes |
| **input control** | a control that prevents invalid or erroneous data from entering the system |
| **integrity control** | a control that rejects invalid data inputs, prevents unauthorized data outputs, and protects data and programs against accidental or malicious tampering |
| **IP Security (IPSec)** | an Internet standard for secure transmission of low-level network pack |
| **key** | an attribute or set of attributes, the values of which occur only once in all the rows of the table |
| **multifactor authentication** | using multiple authentication methods for increased reliability |
| **normalization** | a formal technique for evaluating and improving the quality of a relational database schema |
| **output control** | a control that ensures that output arrives at the proper destination and is accurate, current, and complete |
| **partitioned database server architecture** | multiple distributed database servers are used and the database schema is partitioned, with some content on only one server and some content copied on all servers |
| **physical data store** | database component that stores the raw bits and bytes of data |
| **primary key** | the key chosen by a database designer to represent relationships among rows in different tables |
| **primitive data type** | a data type supported directly by computer hardware or a programming language |
| **privileged user** | a person who has access to the source code, executable program, and database structure of the system |
| **public key encryption** | a form of asymmetric key encryption that uses a public key for encryption and a private key for decryption |
| **referential integrity** | a consistent state among foreign key and primary key values |
| **referential integrity constraint** | a constraint, stored in the schema, that the DBMS uses to automatically enforce referential integrity |
| **registered user** | a person who is authorized to access the system |
| **relational database management system (RDBMS)** | a DBMS that organizes data in tables or relations |
| **remote wipe** | security measure that automatically deletes sensitive data from a portable device when unauthorized accesses are attempted |
| **replicated database server architecture** | complete database copies are hosted by cooperating DBMSs running on multiple servers |
| **row** | one horizontal group of data attribute values in a table |
| **schema** | database component that contains descriptive information about the data stored in the physical data store |
| **second normal form (2NF)** | restriction that a table is in 1NF and that each non-key attribute is functionally dependent on the entire primary key |
| **Secure Hypertext Transport Protocol (HTTPS)** | an Internet standard for securely transmitting Web pages |
| **Secure Sockets Layer (SSL)** | a standard set of methods and protocols that address authentication, authorization, privacy, and integrity |
| **security control** | a control that protects the assets of an organization from all threats, with a primary focus on external threats |
| **single database server architecture** | one or more databases are hosted by a single DBMS running on a single server |
| **symmetric key encryption** | encryption method that uses the same key to encrypt and decrypt the data |
| **table** | a two-dimensional data structure of columns and rows |
| **third normal form (3NF)** | restriction that a table is in 2NF and that no non-key attribute is functionally dependent on any other non-key attribute |
| **transaction logging** | a technique by which any update to the database is logged with such audit information as user ID, date, time, input data, and type of update |
| **Transport Layer Security (TLS)** | an Internet standard equivalent to SSL |
| **unauthorized user** | a person who isn’t allowed access to any part or functions of the system |
| **value limit control** | a control that checks numeric data input to ensure that the value is reasonable |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 13** |  |
| **alpha version** | a test version that is incomplete but ready for some level of rigorous integration or usability testing |
| **beta version** | a test version that is stable enough to be tested by end users over an extended period of time |
| **bottom-up development** | a development order that implements low-level detailed modules first |
| **build and smoke test** | a system test that is performed daily or several times a week |
| **direct deployment or immediate cutover** | a deployment method that installs a new system, quickly makes it operational, and immediately turns off any overlapping systems |
| **driver** | a method or class developed for unit testing that simulates the behavior of a method that sends a message to the method being tested |
| **input, process, output (IPO) development order** | a development order that implements input modules first, process modules next, and output modules last |
| **integration test** | test of the behavior of a group of methods, classes, or components |
| **maintenance release** | a system update that provides bug fixes and small changes to existing features |
| **parallel deployment** | a deployment method that operates the old and the new systems for an extended time period |
| **performance test or stress test** | an integration and usability test that determines whether a system or subsystem can meet time-based performance criteria |
| **phased deployment** | a deployment method that installs a new system and makes it operational in a series of steps or phases |
| **production release, release version, or production release** | a system version that is formally distributed to users or made operational for long-term use |
| **production system** | the version of the system used from day to day |
| **response time** | the desired or maximum allowable time limit for software response to a query or update |
| **source code control system (SCCS)** | an automated tool for tracking source code files and controlling changes to those files |
| **stub** | a method or class developed for unit testing that simulates the behavior of a method that hasn’t yet been written |
| **system documentation** | descriptions of system requirements, architecture, and construction details, as used by maintenance personnel and future developers |
| **system test** | an integration test of an entire system or independent subsystem |
| **test case** | a formal description of a starting state, one or more events to which the software must respond, and the expected response or ending state |
| **test data** | a set of starting states and events used to test a module, group of modules, or entire system |
| **test system** | a copy of the production system that is modified to test changes |
| **throughput** | the desired or minimum number of queries and transactions that must be processed per minute or hour |
| **top-down development** | a development order that implements top-level modules first |
| **unit testing** | test of an individual method, class, or component before it is integrated with other software |
| **usability test** | a test to determine whether a method, class, subsystem, or system meets user requirements |
| **user acceptance test** | a system test performed to determine whether the system fulfills user requirements |
| **user documentation** | descriptions of how to interact with and use the system, as used by end users and system operators |

|  |  |
| --- | --- |
| **Glossary** |  |
| **Chapter 14** |  |
| **browser-based application** | a client/server application that executes locally within the control of a Web browser |
| **device-top application** | a computer application that is built to execute on a local computer (such as a desktop) without requiring a client/server connection |
| **free-standing Internet application** | a client/server application that is self-contained and doesn’t require the use of a Web browser |
| **gadget** | another term for a widget—often used for widgets that reside on a desktop |
| **HTML5** | the new HTML specification that standardizes RIA specifications for built-in browser delivery |
| **Internet backbone** | the primary data routes between large, strategically interconnected networks and routers on the Internet |
| **last mile** | the final leg that delivers connectivity from the Internet network to the customer |
| **mashup** | a type of Web site that combines the functionality of several other Web sites through the use of predefined APIs |
| **on-demand software** | another term for SAAS |
| **open-source software** | a method of developing, delivering, and licensing software that makes the application source code freely available to any interested developer or client |
| **pair programming** | XP practice in which two programmers work together on designing, coding, and testing software |
| **plug-ins** | a software component that adds specific capabilities to a larger software application |
| **product backlog** | a prioritized list of user requirements used to choose work to be done in a Scrum project |
| **product owner** | the client stakeholder for whom the system is being built |
| **refactoring** | revising, reorganizing, and rebuilding part of a system so it is of higher quality |
| **Rich Internet Applications (RIAs)** | a type of Web site that provides active user interaction as well as delivers rich multimedia |
| **Scrum master** | the person in charge of a Scrum project—similar to a project manager |
| **software as a service (SAAS)** | a software delivery model similar to a utility, in which the application and its associated data are accessed via the Internet without locally installed programs |
| **sprint** | a time-controlled mini-project that implements a specific portion of a system |
| **theme** | a type of add-on to an application that allows the look and feel, such as colors and layout, to be changed |
| **toolbar** | a type of add-on usually comprised of iconic menu items that access the capabilities of the application or plug-ins in a userfriendly fashion |
| **UP discipline** | a set of functionally related activities that combine to enable the development process in a UP project |
| **Web 2.0** | a loosely defined, nonstandard term used to refer to Web sites that permit user-generated content and user interaction, such as social networking sites |
| **Web mini-app** | a software application that provides a complete set of functions but that must be executed within the confines of another application |
| **widget** | a type of plug-in that focuses on enhancing the user interface with additional capability |