Glossary/Acronyms

This glossary includes both terms and acronyms. The entry for an acronym shows the expanded phrase; the definition is found under the entry for the phrase. If a definition came from an external source, a citation is provided for the source after the definition.

An entry might contain multiple definitions. If the definitions are merely variations on the same concept, then each definition will reside on its own text line. If the definitions are different, then each definition is numbered and a category is presented in parenthesis. For example:

constraint

1. (SD) Any condition placed on the development of a project, such as what programming language or technologies to use.
2. (UML) A semantic condition or restriction. Certain constraints are predefined in the UML, others may be user defined. (UML v1.4 page B-6)

The set of categories used in this glossary are shown in Table 1

Table 1 Definition Categories

Category Abbreviation	**Category Definition**
Arch	architecture
GUI	graphical user interfaces
Java	Java technology terms
SD	software development
SW	software (general)
UML	Unified Modeling Language

A

abstract class

A class that contains one or more abstract methods, and therefore can never be instantiated. (Sun Glossary)

A class that cannot be directly instantiated. (UML v1.4 page B-2)

(see *abstract method*)
(antonym: *concrete class*)

abstract coupling

Abstract coupling exists when one class depends on either an abstract class or an interface.

abstract method

A method that has no implementation. (Sun Glossary)

abstraction

The essential characteristics of an entity that distinguish it from all other kinds of entities. An abstraction defines a boundary relative to the perspective of the viewer. (UML v1.4 page B-2)

Abstract Window Toolkit

A collection of graphical user interface (GUI) components that were implemented using native-platform versions of the components. (Sun Glossary)

ACM

Association for Computing Machinery

active class

A class whose instances are active objects. (UML v1.4 page B-2)

(see *active object*)

active object

An object that owns a thread and can initiate control activity. An instance of active class. (UML v1.4 page B-2)

activities

The specific actions of one or more workers that produce an artifact. This is the lowest level of organization of the software development process. This describes the who and what of the details of the SD process.

Activity diagram

A UML diagram depicting a flow of activities that might be performed by either a system or an actor.

actors

Users (human or machine) of software.

A coherent set of roles that users of use cases play when interacting with these use cases. (UML v1.4 page B-3)

acyclic dependency principle

The dependencies between packages must form no cycles. (Knoernschild page 27)

ADP

(see *acyclic dependency principle*)

aggregation

A special form of association that specifies a whole-part relationship between the aggregate (whole) and a component part. (UML v1.4 page B-3)

Agile methodologies

“a useful compromise between no process and too much process” (Fowler, `http://www.martinfowler.com/articles/newMethodology.html`) (see also `http://www.agilealliance.org/`)

Analysis model

The model that “refines the use cases in more detail and makes an initial allocation of behavior of the system to a set of objects that provides the behavior.” (Jacobson USPD page 9)

analysis paralysis

The condition in which the development team spends an inordinate amount of time in the Requirements Gathering and Analysis workflows.

API

(see *application programming interface*)

application

A single, deployable component that offers a coherent set of use cases to an end user.

Application layer

In a layered architecture, the Application layer contains the components that are bought or built to support the FRs of the system.

application programming interface

The specification of how a programmer writing an application accesses the behavior and state of classes and objects. (Sun Glossary)

architecturally significant use cases

The use cases of a system that are considered risky (especially, technical risks).

Architecture baseline

The code base that implements the architecturally significant use cases, supports the non-functional requirements, and mitigates the project risks. This is the end result of the Elaboration phase.

Architecture-centric

A software development approach in which the "architecture is defined and validated before the development teams begin the bulk of implementing the system design." (Sun SunTone page 21)

A software development approach in which the "system's architecture is used as a primary artifact for conceptualizing, constructing, managing, and evolving the system under development." (Jacobson USDP page 443)

Architecture model

The model that includes the details of the infrastructure components that are required to support the constraints and NFRs of the proposed system.

Architecture patterns

Software patterns applied at a high-level. These patterns usually solve problems that affect the NFRs of the system.

Architecture template

An abstract model of the detailed Deployment diagram in which the Analysis components are abstracted to a single element.

Architecture workflow

The workflow that models the highest level structure of the system. The architecture model must satisfy the NFRs.

artifact

A physical piece of information that is used or produced by a software development process. (UML v1.4 page B-3)

A tangible product of the development process. (Booch Object Solutions page 303)

assembler paradigm

The programming paradigm in which the software is written in assembler (machine language) for a specific hardware and operating system platform.

association

The semantic relationship between two or more classifiers that specifies connections among their instances. (UML v1.4 page B-3)

association class

A model element that has both association and class properties. (UML v1.4 page B-4)

attribute

A feature within a classifier (such as a class) that describes a range of values that instances of the classifier may hold. (UML v1.4 page B-4)

availability

The system quality that measures the amount of time the system can process a request.

AWT

(see *Abstract Window Toolkit*)

B

B2B

business-to-business

best practice

Recommended techniques that have stood the test of time.

Business Analyst role

The person who gathers requirements from the client stakeholders and analyzes the functional requirements by modeling the enduring business themes of the system.

business entity

(see *Domain entity*)

business logic

The code that supports the workflow of a use case and other functional requirements.

Business Owner role

The lead stakeholder or client of the project. The business owner is responsible for making final decisions about the behavior of the system.

business process

Any coherent collection of activities performed by an person or a system that produces a result of value.

Business tier
The tier whose services "*execute business logic and manage transactions*." (Sun SunTone AM page 15)

C

CACM
Communications of the ACM

CCP
(see *common closure principle*)

CICS
Customer Information Control System, an IBM product

CIO
Chief Information Officer

class
A description of a set of objects that share the same attributes, operations, methods, relationships, and semantics. (UML v1.4 page B-5)

A class is a blueprint or prototype from which objects are created. (The Java™ Tutorials)

Class diagram
A UML diagram depicting a collection of software classes and their inter-relationships.

client
1. (SD) The person or group that is responsible for defining the requirements of the propose system.

2. (SW) The object that uses the services of another object.

client/server
A system that is composed of two physical tiers in which the client tier requests services of a server tier.

Client tier
The tier that contains any "device or system that manages display and local interaction processing." (Sun SunTone AM page 15)

Communication diagram
A UML diagram representing a collection of objects that work together to support some system behavior.

common closure principle

Classes that change together, belong together (in a package). (Knoernschild page 174)

Common Object Request Broker Architecture

A language independent, distributed object model specified by the Object Management Group. (Sun Glossary)

component

A modular, deployable, and replaceable part of a system that encapsulates implementation and exposes a set of interfaces. (UML v1.4 page B-5)

component, Boundary

A component representing a user interface element. This is a primary component in an Analysis model.

component, Controller

A component representing the control aspects of a user interface, such as accepting user input and actions.

Component diagram

A diagram that shows the organizations and dependencies among components. (UML v1.4 page B-6)

component, Entity

A component representing a Domain entity. This is a primary component in an Analysis model.

component, GUI

A component representing an element in a GUI screen, such as a text field or a button.

component, Service

A component which encapsulates a coherent set of business operations, usually representing a use case workflow. This is a primary component in an Analysis model. In the UP method, this component type is called "Control."

component, View

A component representing the visual aspects of a user interface.

composite reuse principle

Favor polymorphic composition of objects over inheritance. (Knoernschild page 17)

Composite Structure diagram

A UML diagram representing the internal structure of a classifier, usually in form of parts, and can include the interaction ports and interfaces (provided or required).

composition

A form of aggregation which requires that a part instance be included in at most one composite at a time, and that the composite object is responsible for the creation and destruction of the parts. (UML v1.4 page B-6)

(see *aggregation*)

concrete class

A class in which all methods have been defined and in which instantiation is permitted.

(see *class*)
(antonym: *abstract class*)

constraint

1. (SD) Any condition placed on the development of a project, such as what programming language or technologies to use.

2. (UML) A semantic condition or restriction. Certain constraints are predefined in the UML, others may be user defined. (UML v1.4 page B-6)

Construction phase

A phase of the Unified Software Development Process focusing on building the software.

constructor

A method-like member of a class that initializes the attributes of an object being instantiated.

container

1. (Arch) A component that exists to contain other components. (UML v1.4 page B-6)

For example, Tomcat is an application-level component that acts as a web container which manages the life cycle of servlet and JSP components within a web application.

2. (SW) An object that holds a collection of other objects, such as an array, list, set, or map.

3. (GUI) A GUI component that groups other GUI components. A GUI container might be a complete window or a panel within a window.

COO
Chief Operational Officer

CORBA
(see *Common Object Request Broker Architecture*)

CTO
Chief Technology Officer

coupling
The degree with which two classes depend upon each other.

CPR
(see *composite reuse principle*)

CPU
central processing unit

CRC analysis
CRC stands for Class-Responsibility-Collaboration. CRC analysis is a technique for identifying key abstractions in the problem domain.

CRUD
This acronym represents the four fundamental database operations: Create, Retrieve, Update, Delete.

D

DAO
(see *Data Access Object*)

Data Access Object
A software pattern that separates the handling of object persistence from the business logic classes.

database
(see *database management system*)

database management system
A system (or embedded subsystem) that provides data persistence facilities.

data type
(see *type*)

DB
database

(see *database management system*)

DBMS

(see *database management system*)

(also see *RDBMS* and *OODBMS*)

DDL

Data Definition Language

dependency

A relationship between two modeling elements, in which a change to one modeling element (the independent element) will affect the other modeling element (the dependent element). (UML v1.4 page B-7)

dependency inversion principle

Depend on abstractions. Do not depend on concretions. (Knoernschild page 12)

Deployment diagram

A UML diagram depicting a collection of components distributed across one or more hardware nodes.

Deployment Specialist role

The person who deploys the implementation onto the production platform.

Deployment workflow

The workflow that puts the implementation into production.

derived attribute

An attribute that can be computed from one or more attributes.

Design pattern

Software patterns applied at a medium-level. These patterns usually solve problems that affect the FRs of the system.

Design workflow

The workflow that generates the Solution model.

Developmental qualities

Developmental qualities are the systemic qualities that are reflected in the immediate development of the system. This category includes such qualities as realizability and planability.

development plan

The schedule of activities for a software project, often based on a prioritized list of use cases.

development team

The group of people that develop the software solution.

DIP
(see *dependency inversion principle*)

discipline
A new term originating from the OMG as a replacement for the term *workflow*.

distributed system
Any system that requires two or more components to communicate across physical hardware nodes.

DLL
dynamically linked library

domain entity
A synonym of *key abstraction*.

Domain model
A Class diagram of the key abstractions of the problem domain space.

DTD
(XML) document type definition

dynamic binding
The ability of the runtime environment to determine which method to call.

E

EBT
(see *enduring business themes*)

EIS
Enterprise Information System

EJB
(see *Enterprise JavaBean*)

Elaboration phase
A phase of the Unified Software Development Process focusing on creating an architecture baseline upon which the rest of the software system will be constructed.

encapsulation
(see *information hiding*)

enduring business themes

The functions of a company that, year after year (decade after decade), do not change.
(see also *key abstraction*)

Enterprise JavaBean

A business-logic component that complies with the EJB specification.

entity

A synonym of *key abstraction*.

entity-relation

The dominate modeling technique for visualizing database schemas.

ER

(see *entity-relation*)

ER diagram

A diagram that represents the entity relationships of the database schema.

ER diagram, logical

An ER diagram in which many details have been left out.

ER diagram, physical

An ER diagram in which the details have been specified. These details include field datatypes, key constraints, indexes, and so on.

event

(GUI) A object that represents a user action, such as moving the mouse, pressing a button, typing on the keyboard, and so on.

event listener

(GUI) A object that responds to user events.

evolutionary prototype

This is the code that is used to prototype and test the Architecture model. This prototype becomes the Architecture baseline after all risks have been mitigated.

Evolutionary qualities

Evolutionary qualities are the systemic qualities that are reflected in the total cost of ownership of the system. This category includes such qualities as scalability, reusability, extensibility, and so on.

extensibility

The system quality that measures the effort saved when adding new functionality.

eXtreme Programming

This methodology enables customers to rank-order features and to change their minds without recrimination from the tech staff. It emphasizes quick-release cycles of code, a focus on keeping the system as simple as possible, and constant testing.

(see also *Agile methodologies*)

F

flexibility

The system quality that measures the effort saved when implementing some change.

forward engineering

The process of creating code (or at least a code skeleton) from a set of models.

FR

(see *functional requirement*)

framework

A collection of classes that provide a set of services for a particular domain; a framework thus exports a number of individual classes and mechanisms that clients can use or adapt. (Booch page 514)

functional paradigm

The programming paradigm in which the software is written as a collection of interacting functions with no side-effects.

functional requirement

The set formal and informal descriptions of the behavior of the system from the user's perspective. This maps closely to the problem model.

G

GANTT chart

A Gantt chart provides a graphical illustration of a schedule that helps to plan, coordinate, and track specific tasks in a project.

generalization

A taxonomic relationship between a more general element and a more specific element. (UML v1.4 page B-10)

In the UML, this relationship is denoted by a solid line with a solid triangular arrowhead.

(antonym: *generalization*)

GoF

Gang of Four. The four authors of the *Design Patterns* book: Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides.

graphical user interface

A user interface that uses windows and GUI components to provide a rich user experience.

guard (condition)

1. (SW) A Boolean expression that permits or prevents flow of control.

2. (UML) *A condition that must be satisfied in order to enable an associated transition to fire.* (UML v1.4 page B-10)

GUI

(see *graphical user interface*)

guillemet

This is a quotation symbol in some European languages such as French. The open guillemet is « and the close guillemet is ». These symbols are used in UML to quote the names of stereotypes, for example «`refines`».

H

Hardware layer

In a layered architecture, the Hardware layer describes the physical characteristics of the machines that support the upper layers.

hierarchy

An ordered classification of classes based on inheritance.

(also know as: taxonomy)

hierarchy, whole-part

(see *whole-part hierarchy*)

HP

Hardware Platform layer

HR

human resources

HRS

Hotel Reservation System

HTML

(see *HyperText Markup Language*)

HTML form
A collection of GUI components generated in a web page which enables the user to enter data and submit requests to the web server.

HTTP
(see *HyperText Transfer Protocol*)

HTTPS
A version of HTTP that uses a secure socket layer (SSL) to provide confidentiality and data integrity for message sent across the Internet.

HVAC
heating, ventilation, and air conditioning

Hypertext Markup Language
This is a file format, based on SGML, for hypertext documents on the Internet. (Sun Glossary)

Hypertext Transfer Protocol
The Internet protocol, based on TCP/IP, used to fetch hypertext objects from remote hosts. (Sun Glossary)

I

ID
identifier

idiom
Software patterns applied at a low-level. These patterns usually solve problems within the scope of a certain computer language, platform, or technology.

IIOP
Internet Inter-ORB Protocol

(see also *CORBA*)

Implementation workflow
The workflow that realizes the solution model. The implementation *model* (which is really just the source code) should have a one-to-one mapping to the solution model. That is to say that the implementation can be coded from the solution model.

Inception phase
A phase of the Unified Software Development Process focusing on understanding the business case for the proposed system.

information hiding

The ability to prevent certain aspects of a class from being accessible to its clients. (Meyer page 1197) Refers to hiding implementation details behind a public interface (one or more methods).

infrastructure

The internal structure of a software system.

infrastructure component

A component that supports the infrastructure of a software system. These components usually do not directly support functional requirements.

inheritance

A mechanism whereby a class is defined in reference to others, adding all their features (members) to its own. (Meyer page 1197)

Integration tier

The tier whose services "abstract and process access to external resources." (Sun SunTone AM page 15)

Interaction Overview diagram

A UML diagram representing a form of activity diagram where nodes can represent interaction diagram fragments. These fragments are usually sequence diagram fragments, but can also be communication, timing, or interaction overview diagram fragments.

interface

A named set of operations that characterize the behavior of an element. (UML v1.4 page B-11)

interface, Java technology

An interface is like a class but has only declarations of it methods. (Arnold, Gosling, and Holmes page 26)

A Java technology interface is roughly equivalent to a C++ class in which all of the methods are empty virtuals.

Internet

An enormous network consisting of literally millions of hosts from many organizations and countries around the world. It is physically put together from many smaller networks and data travels by a common set of protocols. (Sun Glossary)

Internet protocol

The basic protocol of the Internet. It enables the unreliable delivery of individual packets from one host to another. (Sun Glossary)

intranet

A company-wide network.

IP

(see *Internet protocol*)

IT

Information Technology

iterative development

In the context of the software life cycle, a process that involves managing a stream of executable releases. (Jacobson USDP page 446)

Iterative development focuses on growing the system in small, incremental, and planned steps. (Knoernschild page 77)

J

J2EE

(see *Java 2 Platform, Enterprise Edition*)

J2ME

(see *Java 2 Platform, Mobile Edition*)

J2SE

(see *Java 2 Platform, Standard Edition*)

JAR

An archive file that holds a structured collection of Java technology classes and other files. The structure is based on the package hierarchy of the classes.

Java 2 Platform, Enterprise Edition

This is a specification for a platform that supports enterprise application developments. This is a superset of J2SE.

An environment for developing and deploying enterprise applications. The J2EE platform consists of a set of services, application programming interfaces (APIs), and protocols that provide the functionality for developing multi-tiered, Web-based applications. (Sun Glossary)

Java 2 Platform, Mobile Edition

This is a specification for a platform that supports the development of software for small devices, such as palm computers and cell phones. This is a subset of J2SE.

Java 2 Platform, Standard Edition

The core Java technology platform. (Sun Glossary)

JavaBean

A portable, platform-independent reusable component model. (Sun Glossary)

JavaServer Pages™

An extensible Web technology that uses template data, custom elements, scripting languages, and server-side Java objects to return dynamic content to a client. Typically the template data is HTML or XML elements, and in many cases the client is a Web browser. (Sun Glossary)

Java Virtual Machine

A software "execution engine" that safely and compatibly executes the byte codes in Java class files on a microprocessor (whether in a computer or in another electronic device). (Sun Glossary)

JDBC

JDBC stands for Java DataBAse Connectivity. An industry standard for database-independent connectivity between the Java platform and a wide range of databases. The JDBC provides a call-level API for SQL-based database access. (Sun Glossary)

job role

The responsibility of a type of worker.

JRMP

Java Remote Method Protocol

(see also *RMI*)

JSP

(see *JavaServer Pages*)

JVM

(see *Java Virtual Machine*)

K

key abstraction

A key abstraction is a class or object that forms part of the vocabulary of the problem domain. (Booch OOAD page 162)

key, compound

A database key that is composed of more than one field in the table.

key, foreign

A set of fields in one table that uniquely identifies a single row in another table. A foreign key in the former table is usually the primary key in the latter table.

key, primary
A set of fields in a table that uniquely identifies a single row in the table.

L

layer
The hardware and software stack that hosts services within a give tier. (Sun SuntTone AM page 10)
(also see *tier*)

LDAP
Lightweight Directory Access Protocol

legacy system
An older, potentially moldy system that must be preserved for any number of economic or social reasons, yet must also coexist with newly developed elements. (Booch Object Solutions page 305)

link
A connection between two objects; an instance of an association. (Larman page 617)

Lower Platform layer
In a layered architecture, the Lower Platform layer describes the operating system upon which the upper layers depend.

LP
(see *Lower Platform layer*)

M

maintainability
The system quality that measures the effort saved during revision and correction of design flaws.

maintenance
A revision of software embodying the whole development process.

manageability
The system quality that measures the decrease in effort of performing minor administrative tasks.

Manifest qualities
Manifest qualities are the systemic qualities that are reflected in the execution of the system as experienced by a single user. This category includes such qualities as performance, usability, availability, and so on.

mental model

A model of some system that exists in a person's mind.

message

A specification of the conveyance of information from one instance to another, with the expectation that activity will ensue. A message may specify the raising of a signal or the call of an operation. (UML v1.4 page B-11)

method

A function or procedure that is applied to an object. A method implements a message.

methodology

"A body of methods, rules, and postulates employed by a discipline." Software methodology is the highest level project organization. A methodology puts a repeatable structure into the software development process.

method signature

The specification of the interface to a method. A signature usually includes the name of the method, the parameters, and the return type.

model

A simplification of reality. (Booch UML User Guide page 6)

A description of static and/or dynamic characteristics of a subject area, portrayed through a number of views (usually diagrammatic or textual). (Larman page 617)

Model 2 architecture

A Presentation tier structure that uses a variation on the MVC pattern, in which servlets act as a Controller and JSP pages act as Views.

mouse

1. A computer peripheral device that moves a cursor on the monitor.

2. A small fury rodent.

multiplicity

A specification of the range of allowable cardinalities that a set may assume. Multiplicity specifications may be given for roles within associations, parts within composites, repetitions, and other purposes. (UML v1.4 page B-13)

MVC

Model-View-Controller, an architecture pattern

N

navigation

(UML) The ability to traverse an object association in a specific direction. The direction of navigation is specified by the arrowhead(s) on the association in a Class diagram.

NFR

(see *non-functional requirements*)

non-functional requirements

The set formal and informal descriptions of the systemic qualities that the system must satisfy, as well as the technological constraints that the implementation must follow.

n-tier system

A system that is composed of multiple physical tiers.

O

object

object = state + behavior

An object is a runtime instance of a class that contains attributes and operations.

An entity with a well-defined boundary and identity that encapsulates state and behavior. (UML v1.4 page B-13)

object association

(see *association*)

object database

(see *OODBMS*)

Object diagram

A UML diagram depicting a runtime *snapshot* of software objects and their inter-relationships.

Object Management Group

A not-for-profit organization that promotes the research and development of object technologies. The OMG maintains the specifications for CORBA and UML. (`http://www.omg.org/`)

Object Modeling Technique

An object modeling language and method developed in the late 1980's by James Rumbaugh.

object-oriented

A philosophy of software design in which objects (and not procedures) are the central organizational structure.

object-oriented paradigm

The programming paradigm in which the software is written as a collection of interacting objects passing messages to each other.

OCP

(see *open closed principle*)

OH

Overhead slide book

OMG

(see *Object Management Group*)

OMT

(see *Object Modeling Technique*)

OO

(see *object-oriented*)

OOAD

Object-Oriented Analysis and Design

OODBMS

A database management system that stores entities as objects. (also see *DBMS* and *RDBMS*)

OOSD

Object-Oriented Software Development

OOSE

Object-Oriented Software Engineering

open closed principle

Classes should be open for extension but closed for modification. (Knoernschild page 8)

operating system

The system and set of utilities that provide applications with support to execute code, interact with peripherals, and communicate with other external systems.

operation

Some behavior of an object. This usually co-responds to a method.

Operational qualities

Operational qualities are the systemic qualities that are reflected in the execution of the system. This category includes such qualities as throughput, security, serviceability, and so on.

OS

(see *operating system*)

P

PAC

Presentation-Abstraction-Control, an architecture pattern

package

1. (UML) A general purpose mechanism for organizing elements into groups. Packages may be nested within other packages. (UML v1.4 page B-14)

2. (Java) A hierarchical naming structure that organizes a group of *types*.

Package diagram

A UML diagram depicting a collection of other modeling elements and diagrams.

pattern

(see *software pattern*)

PDA

personal digital assistant

performance

The system quality that measures how quickly the system fulfills a user request.

persistence

The property of an object by which its existence transcends time and space. (Booch OOA&D with Apps page 517)

phases

The highest level of organization of the time-dimension in SD process. UP defines four phases: Inception, Elaboration, Construction, and Transition. Each phase includes one or more iterations.

planability

The system quality that measures the confidence that a system can be planned with appropriate cost estimations.

polymorphism

A concept in type theory, according to which a name (such as a variable declaration) might denote objects of many different classes that are related by some common superclass [type]. (Booch OOA&D with Apps page 517)

portability

The system quality that measures the effort saved when migrating to a different infrastructure (such as a new lower platform).

Presentation tier

The tier whose services "aggregate and personalize content and services into channel-specific user interfaces." (Sun SunTone AM page 15)

procedural paradigm

The programming paradigm in which the software is written as a hierarchy of procedures.

Profile diagram

A UML diagram that might define additional diagram types or extend existing diagrams with additional notations.

project glossary

A glossary that records the terminology of the problem domain. The project glossary is usually included in the SRS document.

Project Manager role

The person who manages aspects of a software development project, such as budget, resources, and schedule.

prototype

A small, coherent collection of code that supports some small-scale goal, such as proving the utility of a certain technology.

PST

Pacific standard time

Q

QA

quality assurance

qualified association

An attribute or tuple of attributes whose values partition the set of objects related to an object across an association.

quality of service

A measure of the qualitative characteristics (such as performance, reliability, and scalability) of a system.

R

RAM

random access memory

Rational Unified Process

A software development process based on the Unified Process, but supported by a tool set from Rational Software, Inc.

RDBMS

A database management system that stores entities as rows in tables. (also see *DBMS* and *OODBMS*)

realizability

The system quality that measures the probability (or confidence) that the proposed system can be built.

realization

A relationship between an interface and the class that implements that interface. In the UML, this relationship is denoted by a dashed line with a solid triangular arrowhead (a "dashed generalization" symbol).

refactoring

A change made to the internal structure of software to make it easier to understand and cheaper to modify without changing its observable behavior. (Fowler Refactoring page 53)

relational database

(see *RDBMS*)

relational schema

The definition of tables, fields, indexes, and so on, for a database.

reliability

The system quality that measures the frequency of correctness in an operation.

requirement

A condition or capability to which a system must conform. (Jacobson USDP page 448)

Requirements Analysis workflow

The workflow that models the problem domain. OR: The workflow that generates the problem model. The problem model is based solely on the functional requirements

Requirements Gathering workflow

The workflow that generates the documentation to describe, in both formal and informal descriptions, the business problem.

Requirements model

The model that defines how the software system is intended to behave. This model includes both FRs and NFRs as well as other constraints on the system.

Resource tier

The tier which includes "legacy systems, databases, external data feeds, specialized hardware devices such as telco switches or factory automation, and so on." (Sun SunTone AM page 16)

reusability

The system quality that measures the effort gained by leveraging existing components for new purposes.

reverse engineering

The process of creating a model from a collection of existing code.

risk

A project variable that endangers or eliminates success for a project. (Jacobson USDP page 448)

RMI

Remote Method Invocation

RMI-IIOP

Remote Method Invocation over IIOP

Robustness analysis

Robustness analysis is a technique for identifying the components of the system that supports one or more use cases.

ROI

return on investment

role

The named specific behavior of an entity participating in a particular context. A role may be static (for example, an association end) or dynamic (for example, a collaboration role). (UML v1.4 page B-16)

row

A single entry in a relational database table.

rule-based paradigm

The programming paradigm in which the software is written as a collection of interacting rules and fact-base. Such systems tend to be goal-driven in which rules are used to deduce new information from existing facts.

RUP

(see *Rational Unified Process*)

S

SAP

(see *stable abstractions principle*)

scalability

The system quality that measures the ratio of load growth required to the cost to implement that capacity.

SD

(see *software development*)

SDK

(see *software development kit*)

SDP

(see *stable dependencies principle*)

security

The system quality that prevents undesired use (misuse or abuse) of the system.

Separation of Concerns

An architectural principle in which different components are created to support different purposes within the software. For example, MVC is a pattern that supports the separation of the business logic (Model) components from the user interface (View and Controller) components.

Sequence diagram

A UML diagram depicting a time-oriented perspective of an object collaboration.

serviceability

The system quality that measures the effort required to update or repair the system.

servlet

A Java program that extends the functionality of a Web server, generating dynamic content and interacting with Web clients using a request-response paradigm. (Sun Glossary)

Simple Object Access Protocol

SOAP is a uses a combination of XML-based data structuring and HTTP to define a standardized method for invoking methods in objects distributed in diverse operating environments across the internet. (Sun Glossary)

SOAP

(see Simple Object Access Protocol)

software

The set of instructions that directs the actions of a computer system.

Software Architect role

The person who defines the architecture of the system, leads the development of the architectural baseline during the Inception and Elaboration phases, analyzes the non-functional requirements, and identifies project risks and creates a risk mitigation plan.

Software Designer role

The person who creates the solution model of the system based on the functional requirements (use cases) within the framework of the architecture.

software development

The set of activities that support the creation of software.

software development kit

A set of tools (and possibly one or more frameworks) that enable an engineer to construct software systems.

software pattern

A description of communicating objects and classes that are customized to solve a general design problem in a particular context. (Gamma, Helm, Johnson, Vissides page 3)

A repeatable solution to a recurring problem in a given context.

Software Programmer role

The person who implements the software solution.

Solution model

The model that describes the software components that satisfy both the functional and non-functional requirements of a use case.

specialization

(antonym: *generalization*)

SRS

(see: *System Requirements Specification*)

stable abstractions principle

Stable packages should be abstract packages. (Knoernschild page 31)

stable dependencies principle

Depend on the direction of stability. (Knoernschild page 29)

State Machine diagram

A UML diagram depicting a set of states that an object might experience and the triggers that transition the object from one state to another state.

stakeholder

Any party (person or group) that has an interest in the project.

stereotype

A new type of modeling element that extends the semantics of the metamodel. (UML v1.4 page B-18)

Structured Query Language

The standardized relational database language for defining database objects and manipulating data. (Sun Glossary)

SQL

(see *Structured Query Language*)

state

1. (UML) A condition or situation during the life of an object during which it satisfies some condition, performs some activity, or waits for some event. (UML v1.4 page B-17)

2. (SW) The configuration of attribute values in an object.

state transition

The process of changing state within an object.

subclass

A class that is derived from a particular class, perhaps with one or more classes in between. (Sun Glossary)

subsystem

A grouping of model elements that represents a behavioral unit in a physical system. (UML v1.4 page B-19)

A collection of modules, some of which are visible to other subsystems and other of which are hidden. (Booch OOAD page 519)

subtype

If type X extends or implements type Y, then X is a subtype of Y. (Sun Glossary)

SunTone Architecture Methodology

A software development process based on "a step-by-step process for creating dot-com (n-tier and web-centric) architectures." (Sun SunTone page 4)

superclass

A class from which a particular class is derived, perhaps with one or more classes in between. (Sun Glossary)

supertype

The supertypes of a type are all the interfaces and classes that are extended or implemented by that type. (Sun Glossary)

system

A collection of interacting software and hardware components usually at the application-level.

system boundary

In a UML Use Case diagram, the box that separates the use cases for a system from the outside of the system (the actors).

System Requirements Specification

A document that defines all requirements (both FRs and NFRs) for a system.

systemic qualities

The term used by the SunTone architecture methodology for non-functional requirements.
(see also *quality of service*)

systemic-quality-driven

An software development approach that "places a critical emphasis on identifying, ranking, and quantifying the systemic qualities as requirements." (Sun SunTone page 23)

T

table

A coherent collection of rows (all with the same fields) in a relational database.

tag

A syntactic structure in an SGML (XML or HTML) file that identifies a specific piece of information.

TCP

(see *transmission control protocol*)

testability

The system quality that measures the effort required to identify and isolate a fault or error in the system.

Test Engineer role

The person who tests the implementation to verify that the system meets the requirements (both functional and non-functional).

Testing workflow

The workflow in which you test the implementation against the expectations as defined by the requirements.

thread

The basic unit of program execution. A process can have several threads running concurrently, each performing a different job, such as waiting for events or performing a time-consuming job that the program does not need to complete before going on. (Sun Glossary)

threadsafe

The condition of a method (or set of methods in a component) in which there

throughput

The system quality that measures the amount of work done by the system, measured in operations per unit time.

tier

A logical or physical organization of components into an ordered chain of service providers and consumers. (Sun SunTone AM page 10)

Timing diagram

A UML diagram representing changes in state (state lifeline view) or value (value lifeline view). It can also show time and duration constraints and interactions between timed events.

transaction

An atomic unit of work that modifies data. A transaction encloses one or more program statements, all of which either complete or roll back. Transactions enable multiple users to access the same data concurrently. (Sun Glossary)

(abbreviated: txn)

Transition phase

A phase of the Unified Software Development Process focusing on readying the software for production.

transmission control protocol

This is an Internet protocol that provides for the reliable delivery of streams of data from one host to another. (Sun Glossary)

type

A stereotyped class that specifies a domain of objects together with the operations applicable to the objects, without defining the physical implementation of those objects. (UML v1.4 page B-20)

A class or interface. (Sun Glossary)

type, primitive

A data type that is not part of the object types of a computer language. These usually include support for integers, real numbers (called floating-point numbers), characters, Boolean values, and so on.

U

UC

(see *use case*)

UI

(see *user interface*)

UML

(see *Unified Modeling Language*)

UML tool

An tool that supports the creation and maintenance of UML diagrams for a project.

Unified Modeling Language

A standard modeling language for software – a language for visualizing, specifying, constructing, and documenting artifacts of a software-intensive system. (Jacobson USDP page 449)

Unified Software Development Process

A software development process based on the UML that is iterative, architecture-centric, use-case-driven, and risk-driven. (Jacobson USDP page 449)

UP

(see *Unified Software Development Process*)

Upper Platform layer

In a layered architecture, the Upper Platform layer contains the systems that implement the Virtual Platform specifications. The systems in this layer are usually containers for components in the Application layer.

US

United States

usability

The system quality that measures the ease by which a user can accomplish some goal.

USDP

(see Unified Software Development Process)

use case

The activities performed by software for actors to support a single system function. Often called a "user story."

Use Case diagram

An UML diagram depicting the set of high-level behaviors the system must perform for a given actor.

Use-Case-driven

In the context of the software life cycle, meaning that use cases are used as a primary artifact for establishing the desired behavior of the system and for communicating this among the stakeholders of the system. (Jacobson USDP page 450)

An software development approach in which "every attempt is made to prioritize design and development around the realization of complete, end-to-end use cases." (Sun SunTone page 21)

use case form

A text form that records the analysis of a use case.

use case scenario

A text story of a specific instance of a use case.

user interface

The boundary components that interact with the user.

V

view

A projection of a model, which is seen from a given perspective or vantage point and omits entities that are not relevant to this perspective. (UML v1.4 page B-21)

Virtual Platform layer

In a layered architecture, the Virtual Platform layer contains the specifications and APIs upon which the Application layer components depend.

visibility

An enumeration whose value (public, protected, or private) denotes how the model element to which it refers may be seen outside its enclosing name space. (UML v1.4 page B-21)

Vision document

A document that records the main ideas of a proposed software system that usually includes the business case for building such a system.

VP

virtual platform

VPN

virtual private network

W

W3C

World Wide Web Consortium

WAE

Web Application Extension

Waterfall

A software development process based on the a single iteration through the SD workflows.

web application

A coherent collection of dynamically-generated web pages and forms that supports one or more use cases of a system.

web container

A container that provides the network services over which requests and responses are sent, decodes requests, and formats responses. All servlet (web) containers must support HTTP as a protocol for requests and responses, but may also support additional request-response protocols such as HTTPS. (Sun Glossary)

WebMVC

Web model-view-controller

web server
A host machine that supports one or more Web sites or web applications.

web services
A distributed remote procedure communication mechanism.

Web site
A coherent collection of web pages at a single web server.

Web tier
(synonym: *Presentation tier*)

WebUI
(see *web user interface*)

web user interface
A UI that is presented in a Web browser screen.

whole-part hierarchy
An ordered classification of objects based on the breakdown of parts and subparts.

window
(GUI) A rectangular portion of the computer monitor that contains user interface components.

worker
A person that performs an activity.

workflow
1. The middle level of organization of activities within an iteration. The classical SD workflows are: requirements gathering, requirements analysis, architecture, design, implementation, test, and deployment.

2. A business process.

X

XML
eXtensible Markup Language

XP
(see *eXtreme Programming*)