SysML Glossary
(Draft - April 3, 2006)

This glossary is a support document (ad/2006-03-04) to the SysML Specification v1.0 (ad/2006-03-01) developed by the SysML Merge Team (SMT). The document is referenced in Appendix F Terms and Definitions of the SysML Specification.

The following terms and definitions are referred to in the SysML specification and are derived from multiple sources including the UML Superstructure (formal/05-07-04) and the UML for Systems Engineering RFP (ad/03-03-41).

The following conventions are used in the term and definitions below:

- A term followed by [**SysML**] or a phrase preceded by [**SysML**] means that this term is a definition introduced by SysML.
- A term followed by [**UML for SE RFP**] or a phrase preceded by [**UML for SE RFP**] is referenced in the UML for SE RFP (ad/03-03-41).
- A term which is not followed by **[SysML]** or **[UML for SE RFP]** is derived from UML.
- A phrase of the form "See: <term>" refers to a related term that has a similar, but not synonymous meaning.
- A phrase of the form "Contrast: <term>" refers to a term that has an opposed or substantively different meaning.
- A phrase of the form "Synonym: <term>" indicates that the term has the same meaning as another term, which is referenced.
- A phrase of the form "Acronym: <term>" indicates that the term is an acronym. The reader is usually referred to the spelled-out term for the definition, unless the spelled-out term is rarely used.)

abstract class (block)*

A class (or block) that cannot be directly instantiated. Contrast: *concrete class*.

abstract syntax*

abstraction

An abstraction is a relationship that relates two elements or sets of elements that represent the same concept at different levels of abstraction or from different viewpoints.

access*

action

An action is a named element that is the fundamental unit of executable functionality. The execution of an action represents some transformation or processing in the modeled system, be it a computer system or otherwise. [**UML for SE RFP:** A non-interruptible function. Note: An action represents an atomic unit of processing or work. Actions may be continuous or discrete. Discrete actions may or may not be assumed to execute in zero time. See *function.*]

active object

An object that may execute its own behavior without requiring method invocation. This is sometimes referred to as "the object having its own thread of control." The points at which an active object responds to communications from other objects are determined solely by the behavior of the active object and not by the invoking object. This implies that an active object is both autonomous and interactive to some degree. See: *active class, thread.*

activity

An activity is the specification of parameterized behavior as the coordinated sequencing of subordinate units whose individual elements are actions. [**UML for SE RFP**: 1) One or more related actions. 2) [**SysML:** Usage of a function. See *action, function.*]

activity diagram*

A diagram that depicts behavior associated with activities using input/output and control flow.

activity edge
An activity edge is an abstract class for directed connections between two activity nodes.
activity final node
An activity final node is a final node that stops all flows in an activity.
activity node
An activity node is an abstract class for points in the flow of an activity connected by edges.
activity parameter node
An activity parameter node is an object node for inputs and outputs to activities.
activity partition
An activity partition is a kind of activity group for identifying actions that have some characteristic in common.
actor
An actor specifies a role played by a user or any other system that interacts with the subject. (The term “role” is used informally here and does not necessarily imply the technical definition of that term found elsewhere in this specification.)
actual parameter
Synonym: *argument*.
aggregate
A class that represents the “whole” in an aggregation (whole-part) relationship. See: *aggregation*.
aggregation
A special form of association that specifies a whole-part relationship between the aggregate (whole) and a component part. See: *composition*.
allocate [SysML]
A mapping between from one set of model elements (supplier) to another set of model elements (client). The mapping is often performed as part of the design process to refine the design. Typical examples of allocation include allocation of functions to components, logical to physical components, flows to connectors, and software to hardware. The allocation of requirements to components is generally accomplished using the SysML satisfy relationship. See *allocated element, allocate activity partition, allocate behavior, allocate flow, allocate structure*.
allocate activity partition [SysML]
allocated element [SysML]*
A stereotype of an element that is the client or supplier of an allocation with properties allocatedFrom or allocatedTo. See *allocation*.
apply
A relationship that is used to stereotype a model element.
association
An association describes a set of tuples whose values refer to typed instances. An instance of an association is called a link.
atomic flow port [SysML]*
A FlowPort that is typed by an Block, ValueType, DataType or Signal, and not typed by a FlowSpecification.
attribute
A structural feature of a classifier that characterizes instances of the classifier. An attribute represents a declared state of one or more instances in terms of a named relationship to a value or values. See also: *property*
behavior
Behavior is a specification of how its context classifier changes state over time. This specification may be either a definition of possible behavior execution or emergent behavior, or a selective illustration of an interesting subset of possible executions. The latter form is typically used for capturing examples, such as a trace of a particular execution. A behavior can take a number of forms: interaction, statemachine, activity, or procedure (a set of actions).
behavior diagram
A form of diagram that depict behavioral features.

behavioral feature

A behavioral feature is implemented (realized) by a behavior. A behavioral feature specifies that a classifier will respond to a designated request by invoking its implementing method.

binary association

An association between two classes. A special case of an n-ary association.

binding connector [SysML]

bidirectional connector [SysML]

boolean

A boolean type is used for logical expression, consisting of the predefined values true and false.

block [SysML]

A modular unit that describes the structure of a system or element.

block definition diagram [SysML]

A diagram that represents the relationship between blocks and the structural and behavioral features of blocks.

block property [SysML]

A property of a block that enables the unit, dimension, and distribution capabilities to be applied and enforces additional constraints specific to SysML. The Block property is abstract.

boolean expression

An expression that evaluates to a boolean value.

call action

CallAction is an abstract class for actions that invoke behavior and receive return values.

call behavior action

CallBehaviorAction is a call action that invokes a behavior directly rather than invoking a behavioral feature that, in turn, results in the invocation of that behavior.

call operation action

CallOperationAction is an action that transmits an operation call request to the target object, where it may cause the invocation of associated behavior.

cardinality

The number of elements in a set. Contrast: *multiplicity*.

central buffer node

An object flow for managing flows from multiple sources and destinations.

child

In a generalization relationship, the specialization of another element, the parent. See: *subclass*, *subtype*. Contrast: *parent*.

choice pseudo state

A pseudostate which, when reached, result in the dynamic evaluation of the guards of the triggers of its outgoing transitions.

class

A class describes a set of objects that share the same specifications of features, constraints, and semantics.

classifier

A classifier is a classification of instances, it describes a set of instances that have features in common.

classification

The assignment of an instance to a classifier. See *dynamic classification*, *multiple classification* and *static classification*.

class diagram*

A diagram that shows a collection of declarative (static) model elements, such as classes, types, and their contents and relationships.

client

A classifier that requests a service from another classifier. Contrast: *supplier.*

combined fragment

A combined fragment defines an expression of interaction fragments. A combined fragment is defined by an interaction operator and corresponding interaction operands. Through the use of CombinedFragments the user will be able to describe a number of traces in a compact and concise manner.

comment

A comment is a textual annotation that can be attached to a set of elements.

communication path

An association between a use case and an actor.

compartment

complex [SysML]

A value type that represents a complex number with a real and imaginary part. [**UML for SE RFP:** A number which includes a real and imaginary part.]

compliance

compliance level

composite

A class or block that is related to one or more classes by a composition relationship. See: *composition*.

composite aggregation*

See *composition*.

composite state*

A state that consists of either concurrent (orthogonal) substates or sequential (disjoint) substates. See: *substate*.

composite structure diagram*

A diagram that depicts the internal structure of a classifier, including the interaction points of the classifier to other parts of the system. It shows the configuration of parts that jointly perform the behavior of the containing classifier. The architecture diagram specifies a set of instances playing parts (roles), as well as their required relationships given in a particular context.

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. Composition may be recursive. Synonym: *composite aggregation*.

concrete class

A class that can be directly instantiated. Contrast: *abstract class.*

concrete syntax

concurrency

The occurrence of two or more activities during the same time interval. Concurrency can be achieved by interleaving or simultaneously executing two or more threads. See: *thread.*

concurrent substate

A substate that can be held simultaneously with other substates contained in the same composite state. See: *composite state.* Contrast: *disjoint substate.*

condition

A boolean expression which is true as long as the expression evaluates true, and is false otherwise. See *guard condition*.

connectable element

A ConnectableElement is an abstract metaclass representing a set of instances that play roles of a classifier. Connectable elements may be joined by attached connectors and specify configurations of linked instances to be created within an instance of the containing classifier. See: *connector.*.

connector

Specifies a link that enables communication between two or more instances. This link may be an instance of an association, or it may represent the possibility of the instances being able to communicate because their identities

are known by virtue of being passed in as parameters, held in variables or slots, or because the communicating instances are the same instance. [**UML for SE RFP:** See *connection, connection path, connecting component*.]

connector end

A connector end is an endpoint of a connector, which attaches the connector to a connectable element. Each connector end is part of one connector.

constraint

A constraint is a condition or restriction expressed in natural language text or in a machine readable language for the purpose of declaring some of the semantics of an element.

constraint note

constraint block [SysML]

A block that packages the statement of a constraint so it may be applied in a reusable way to properties of other blocks. A constraint block includes the definition of one or more constraint parameters which can be bound to properties in a specific context where the constraint is used.

constraint parameter [SysML]

A property of a constraint block that is used to bind other properties in a specific context where the constraint is used.

constraint property [SysML]

A Block Property that is typed by a constraint block, and owned by a containing block. A usage of a constraint block in a particular context that can constrain other properties (e.g. value properties).

containment *

See *namespace containment*

containment hierarchy*

A namespace hierarchy consisting of model elements, and the containment relationships that exist between them. A containment hierarchy forms a graph.

continuous rate [**SysML**]

A subclass of rate which enables a parameter value to be sampled at an infinite rate (e.g. the increment of time between parameter values approaches zero). See *rate, discrete rate*.

control flow

A control flow is an edge that starts an activity node after the previous one is finished.

control input

An input that activates or deactivates a function. See *activation, activation/deactivation event*.

control node

A control node is an abstract activity node that coordinates flows in an activity. It is used to coordinate the flows between other nodes. It covers initial node, final node, and its children, fork node, join node, decision node, and merge node.

control operator [SysML]

A behavior that is intended to represent an arbitrarily complex logical operator that can be used to enable and disable other actions.

control pin

control value [**SysML**]

An enumerated value that is used to control the execution of an activity. See *control operator.*

copy

A relationship between a supplier requirement that represents the orignal and a client requirement that represents a copy of the original.

coregion*

creation event

data

A constituent element of information.

data type

A type whose instances are identified only by their value. A DataType may contain attributes to support the modeling of structured data types.

decomposition [**UML for SE RFP**]

A description of a whole in terms of its component parts. See *aggregation*.

decision node

A decision node is a control node that chooses between outgoing flows.

deep history state

A pseudostate that represents the most recent active configuration of the composite state that directly contains this pseudostate (e.g., the state configuration that was active when the composite state was last exited).

delegation

The ability of an object to issue a message to another object in response to a message. Delegation can be used as an alternative to inheritance. Contrast: *inheritance*.

dependency

A dependency is a relationship that signifies that a single or a set of model elements requires other model elements for their specification or implementation. This means that the complete semantics of the depending elements is either semantically or structurallydependent on the definition of the supplier element(s).

deployment [**UML for SE RFP**]*

A dependency relationship between components, where one component depends on the hosting component (i.e node) for resources in order to r perform its functions.

derive requirement

A dependency relationship between two requirements in which a client requirement can be generated or inferred from the supplier requirements or additional design information. Derived requirements may refine or restate a requirement to improve stakeholder communications or to track design evolution.

derived [SysML]

A requirement that has been derived from another requirement.

design constraint [**UML for SE RFP**]

A requirement that one or more components of a system must satisfy. Note: This term is sometimes used to refer to a constraint on the design process versus the system.See *requirement.*

destruction event

A DestructionEvent models the destruction of an object.

diagram*

A graphical presentation of a collection of model elements, most often rendered as a connected graph of arcs (relationships) and vertices or nodes (other model elements). UML supports the diagrams listed in Appendix A. See *diagram usage*.

diagram content

diagram description [**SysML**]

A comment that provides standardized information about a diagram.See *reference data*.

diagram element

diagram frame

diagram heading

diagram kind

diagram name [SysML]

diagram usage [**SysML**]

A form of stereotype applied to a diagram to constrain its use. See *diagram.*

dimension [SysML]

A kind of quantity that may be stated by means of defined units. For example, the dimension of length may be mea-

sured by units of meters, kilometers, or feet.See *dimension, unit, value type, value property.*

discrete rate [**SysML**]
A subclass of rate which enables a parameter value to be sampled at a finite rate (e.g. the increment of time between parameter values is non zero). See *rate, continuous rate*.

disjoint substate*
A substate that cannot be held simultaneously with other substates contained in the same composite state. See: *composite state*. Contrast: *concurrent substate*.

distribution definition [SysML]
A probability distribution that may be specified on a Block Property. Synonym *probability distribution.* See *distribution result.*

do action

domain
An area of knowledge or activity characterized by a set of concepts and terminology understood by practitioners in that area.

duration
A duration defines a value specification that specifies the temporal distance between two time expressions that specify time instants.

duration constraint
A DurationConstraint defines a Constraint that refers to a DurationInterval.

duration interval
A DurationInterval defines the range between two Durations.

dynamic classification
The assignment of an instance from one classifier to another. Contrast: *multiple classification, static classification*.

effectiveness measure [**UML for SE RFP**]*
A criterion for system optimization that is critical to the success of the mission. Note: The criterion are often used to support trade studies to select among alternatives well as to optimize a given design. See *measure of effectviness*.
production, deployment, support, and disposal systems.

enhanced functional flow block diagram [**SysML**]*
A restricted form of activity diagram. See *activity diagram*.

entry action
An optional behavior of a state that is executed whenever the state is entered regardless of the transition taken to reach the state. If defined, entry actions are always executed to completion prior to any internal behavior or transitions performed within the state.

entry point
A pseudostate that is an entry point of a state machine or composite state. In each region of the state machine or composite state it has a single transition to a vertex within the same region.

enumeration
An enumeration is a data type whose values are enumerated in the model as enumeration literals.

environment [**UML for SE RFP**]*
A collection of systems and items that interact either directly or indirectly with the system of interest.See *item, system*.*

event
An event is the specification of some occurrence that may potentially trigger effects by an object. [**UML for SE RFP:** A noteworthy occurrence that occurs at the instant of time when a specified expression evaluates true.]

exception
A special kind of signal, typically used to signal fault situations. The sender of the exception aborts execution and execution resumes with the receiver of the exception, which may be the sender itself. The receiver of an exception is determined implicitly by the interaction sequence during execution; it is not explicitly specified.

execution [**UML for SE RFP**]*

The state of the system or model when it is running. For a model, this implies that model computation is occurring.

execution specification

An ExecutionSpecification is a specification of the execution of a unit of behavior or action within the Lifeline.

exit action

An optional behavior that is executed whenever this state is exited regardless of which transition was taken out of the state. If defined, exit actions are always executed to completion only after all internal activities and transition actions have completed execution.

exit point

A pseudostate that is an exit point of a state machine or composite state. Entering an exit point within any region of the composite state or state machine referenced by a submachine state implies the exit of this composite state or submachine state and the triggering of the transition that has this exit point as source in the state machine enclosing the submachine or composite state.

export

In the context of packages, to make an element visible outside its enclosing namespace. See: *visibility.* Contrast: *export* [OMA], *import*.

expression

An expression represents a node in an expression tree, which may be non-terminal or terminal. It defines a symbol, and has a possibly empty sequence of operands that are value specifications.

extend

A relationship from an extending use case to an extended use case that specifies how and when the behavior defined in the extending use case can be inserted into the behavior defined in the extended use case. See *extension point*, *include*.

extension

An extension is used to indicate that the properties of a metaclass are extended through a stereotype, and gives the ability to flexibly add (and later remove) stereotypes to classes.

extension point

An extension point identifies a point in the behavior of a use case where that behavior can be extended by the behavior of some other (extending) use case, as specified by an extend relationship.

facility [**UML for SE RFP**]*

A physical infrastructure that supports use of equipment and other resources. See *component*.

failure [**UML for SE RFP**]*

An inability to satisfy a requirement. See *requirement.*

feature*

A behavioral or structural characteristic of a classifier.

feature support statement

final node

A final node is an abstract control node at which a flow in an activity stops.

final state

A special kind of state signifying that the enclosing region is completed. If the enclosing region is directly contained in a state machine and all other regions in the state machine also are completed, then it means that the entire state machine is completed.

fire

To execute a state transition. See: *transition*.

flow direction [SysML]

flow final node

A flow final node is a final node that terminates a flow.

flow port [SysML]

An interaction point through which input and/or output of items such as data, material or energy may flow.

flow property [SysML]

A property of a flow specification or a block that signifies a single flow element to/from a Block.

flow specification [SysML]

A specification of inputs and outputs as a set of flow properties.

focus of control

A symbol on a sequence diagram that shows the period of time during which an object is performing an action, either directly or through a subordinate procedure.

found message.

fork node*

A fork node is a control node that splits a flow into multiple concurrent flows.

formal parameter

Synonym: *parameter*.

function [**UML for SE RFP**]

A transformation of inputs to outputs that may include the creation, monitoring, modification or destruction of elements, or a null transformation.

function port [**UML for SE RFP**]

A binding of an input/output to the arguments of a function. See *argument, input/output, pin*.

function time-line [**UML for SE RFP**]

A representation of the interval of time that one or more functions and/or states are active and inactive.

functional requirement [**UML for SE RFP**]*

A function that must be performed. See *requirement.*

generalization

A generalization is a taxonomic relationship between a more general classifier and a more specific classifier. Each instance of the specific classifier is also an indirect instance of the general classifier. Thus, the specific classifier inherits the features of the more general classifier. See: *inheritance.* [**UML for SE RFP:** See *specialization*]

generalization set

A GeneralizationSet is a PackageableElement whose instances define collections of subsets of Generalization relationships.

graphic edge

graphic node

guard condition*

A condition that must be satisfied in order to enable an associated transition to fire. See *condition.*]

hardware [**UML for SE RFP**]*

A component of a system that has geometric constraints. See *component*.

implementation

A definition of how something is constructed.

operations defined for the type with the same behavior as specified for the type's operations. See also: *type*.

import

An element import identifies an element in another package, and allows the element to be referenced using its name without a qualifier. A package import is a relationship that allows the use of unqualified names to refer to package members from other namespaces. Contrast: *export*.

include

An include relationship defines that a use case contains the behavior defined in another use case. See *extend*.

information flow

An Information Flow specifies that one or more information items circulate from its sources to its targets. Information flows require some kind of “information channel” for transmitting information items from the source to the destination. An information channel is represented in various ways depending on the nature of its sources and targets.

information item
An information Item is an abstraction of all kinds of information that can be exchanged between objects. It is a kind of classifier intended for representing information at a very abstract way, which cannot be instantiated.
inheritance
See *generalization*.
initial node
An initial node is a control node at which flow starts when the activity is invoked.
initial pseudo state
An initial pseudostate represents a default vertex that is the source for a single transition to the default state of a composite state. There can be at most one initial vertex in a region. The initial transition may have an action.
input/output [**UML for SE RFP**]
An item that is subject to a transformation by a function. See argument, *function port, parameter, signature*.*
instance
An entity that has unique identity, a set of operations that can be applied to it, and state that stores the effects of the operations. See: *object*.
instance specification
An instance specification is a model element that represents an instance in a modeled system.
integer [**UML for SE RFP**]
A whole number
interaction
An interaction is a unit of behavior that focuses on the observable exchange of information between ConnectableElements. **[UML for SE RFP:** Emergent behavior that results from two or more dependent behaviors Note: A system or component interacts with other components its environment, to yield an emergent system behavior from the individual component behaviors.]
interaction operator
Interaction Operator is an enumeration designating the different kinds of operators of CombinedFragments. The InteractionOperand defines the type of operator of a CombinedFragment. The literal values of this enumeration are: alt, assert, consider, critical, ignore, loop, neg, opt, par, seq, strict
interface
An interface is a kind of classifier that represents a declaration of a set of coherent public features and obligations. An interface specifies a contract; any instance of a classifier that realizes the interface must fulfill that contract. The obligations that may be associated with an interface are in the form of various kinds of constraints (such as pre- and postconditions) or protocol specifications, which may impose ordering restrictions on interactions through the interface. [**UML for SE RFP:** The inputs, outputs, ports, connections, connecting components (i.e. harness), and associated information that support one or more interactions between systems. Note: The UML definition of interface includes the operations that must be performed in response to the inputs or invocations. An interface in systems engineering is generally more broadly defined consistent with the UML for SE RFP definition.] See *provided interface, required interface.*
interface requirement [**UML for SE RFP**]*
An interface a system must support. See *requirement.*
internal block diagram [SysML]
A diagram that depicts the internal structure of a block, including the interaction points to other parts of the system. It shows the configuration of parts that jointly perform the behavior of the containing block. The diagram specifies a set of instances playing parts (roles) in the context of the enclosing block (context).
internal transition
A transition signifying a response to an event without changing the state of an object.
interruptible activity region

An interruptible activity region is an activity group that supports termination of tokens flowing in the portions of an activity.

item flow [**SysML**]*

Representation of the items that flow across a connector or an association that is a subclass of a directed relationship (Note: more precisely a subclass of information flow) that is realized by a relationship that conveys the item(s). See *item, item property.*

item property [**SysML**]*

A property that relates the instances of the item to the instances of its enclosing class. See *item, item flow, property.*

iteration loop [**UML for SE RFP**]

A specialized loop where the loop repeats a specified number of times.

join node

A join node is a control node that synchronizes multiple flows.

junction pseudo state

Junction pseudo states are semantic-free vertices that are used to chain together multiple transitions. They are used to construct compound transition paths between states.

leaf function [**UML for SE RFP**]*

A function which is not further decomposed. See *function, action.*

lifeline

A lifeline represents an individual participant in the Interaction.

link

An instance of an association. See: *association*.

lost message*

manual procedure [**UML for SE RFP**]

A set of operations that provide instructions for a user to perform. See *procedure*.

mean [**UML for SE RFP**]

The expected value associated with a probability distribution.

measure of effectiveness [**SysML**]

See *effectiveness measure, objective function*

merge node

A merge node is a control node that brings together multiple alternate flows. It is not used to synchronize concurrent flows but to accept one among several alternate flows.

message

A Message defines a particular communication between Lifelines of an Interaction. [**UML for SE RFP:** See *control input, triggering input*.]

metaclass

A class whose instances are classes. Metaclasses are typically used to construct metamodels.

meta-metamodel

A model that defines the language for expressing a metamodel. The relationship between a meta-metamodel and a metamodel is analogous to the relationship between a metamodel and a model.

metamodel

A model that defines the language for expressing a model.

metamodel reference

metaobject

A generic term for all metaentities in a metamodeling language. For example, metatypes, metaclasses, metaattributes, and metaassociations.

method

The implementation of an operation. It specifies the algorithm or procedure associated with an operation.

model

It is an abstraction of the physical system, with a certain purpose. This purpose determines what is to be included in the model and what is irrelevant.

model element
A constituent of a model that is an abstraction drawn from the system being modeled.

model interchange [**UML for SE RFP**]
The ability to exchange model information.

model library
A stereotyped package that contains model elements that are intended to be reused by other packages. A model library differs from a profile in that a model library does not extend the metamodel using stereotypes and tagged definitions. A model library is analogous to a class library in some programming languages.

multiple inheritance
A semantic variation of generalization in which a type may have more than one supertype. Contrast: *single inheritance*.

multiplicity element
A multiplicity is a definition of an inclusive interval of non-negative integers beginning with a lower bound and ending with a (possibly infinite) upper bound. A multiplicity element embeds this information to specify the allowable cardinalities for an instantiation of this element. Contrast: *cardinality*.

name
A string used to identify a model element.

named element
A named element is an element in a model that may have a name.

namespace
A namespace is an element in a model that contains a set of named elements that can be identified by name. See: *name*.

namespace containment
See: *containment*.

natural object [**UML for SE RFP**]*
An *item* that is not engineered, and may be part of a system or environment.See *item.**

need [**UML for SE RFP**]
A desired requirement of a stakeholder. See *requirement.*

nested connector end [SysML]
A connector end that allows a connected property to be identified by a multi-level path through a part (property) hierarchy.

no-buffer [SysML]
stereotype that is applied to object nodes, so that tokens arriving at the node are discarded, if they are refused by outgoing edges, or refused by actions for object nodes that are input pins.

non-streaming parameter*
A non-streaming parameter specifies that the parameter value can only be accepted at the beginning of execution and produced at the end of execution. Contrast: *streaming*.

note
See: *comment*.

notation [**UML for SE RFP**]
The graphical depiction of a model construct.

object
An instance of a class. See: *class, instance*.

object diagram*
A diagram that encompasses objects and their relationships at a point in time. An object diagram may be considered a special case of a class diagram. See: *class diagram.*

object flow
An object flow is an activity edge that can have objects or data passing along it.

object node
An object node is an abstract activity node that is part of defining object flow in an activity.
objective function [SysML]
See *effectiveness measure*
operation
An operation is a behavioral feature of a classifier that specifies the name, type, parameters, and constraints for invoking an associated behavior. [**UML for SE RFP:** See *activity, function*.]
operational requirement [**UML for SE RFP**]*
A requirement which is associated with the operation of a system, and typically includes a combination of functional, interface, and performance requirements. See *requirement.*
optional parameter [SysML]
A stereotype of a parameter with lower multiplicity equal to zero indicating that the parameter is not required for the activity to begin execution. Otherwise, the lower multiplicity must be greater than zero, which is call “required”. (See required parameter, activity parameter node).Synonym *non-triggering input*.
overwrite [SysML]
a stereotype that is applied to object nodes, so that a token arriving at a full object node replaces the ones already there (a full object node has as many tokens as allowed by its upper bound).
package
A package is used to group elements, and provides a namespace for the grouped elements.
package containment
See *namespace containment*.
package diagram
A diagram that depicts how model elements are organized into packages and the dependencies among them, including package imports and package extensions.
parameter
A parameter is a specification of an argument used to pass information into or out of an invocation of a behavioral feature. Synonyms: *formal parameter*. Contrast: *argument*.
parameter set
A parameter set is an element that provides alternative sets of inputs and outputs that a behavior may use.
parametric diagram [SysML]
A diagram that represents a network of constraints on properties to support engineering analysis such as performance, reliability and mass properties analysis.
parent
In a generalization relationship, the generalization of another element, the child. See: *subclass*, *subtype*. Contrast: *child*.
part
An element representing a set of instances that are owned by a containing classifier instance. (See *role.)* Parts may be joined by attached connectors and specify configurations of linked instances to be created within an instance of the containing classifier. [Note: This is referred to as a part property in SysML which is an owned property of a block.] (See *property*)
participate
The connection of a model element to a relationship or to a reified relationship. For example, a class participates in an association, an actor participates in a use case.
partition
A grouping of any set of model elements based on a set of criteria.
1. activity diagram: A grouping of activity nodes and edges. Partitions divide the nodes and edges to constrain and show a view of the contained nodes. Partitions can share contents. They often correspond to organizational units in a business model. They may be used to allocate characteristics or resources among the nodes of an activity.
2. architecture: A set of related classifiers or packages at the same level of abstraction or across layers in a layered architecture. A partition represents a vertical slice through an architecture, whereas a layer represents a horizontal

slice. Contrast: *layer*.

performance property [**UML for SE RFP**]
A measure of the transformation or response of a function or behavior (i.e response time, etc.).

performance requirement [**UML for SE RFP**]*
A performance property a system must satisfy. See *requirement.*

persistent object
An object that exists after the process or thread that created it has ceased to exist.

physical property [**UML for SE RFP**]
A physical characteristic of a system or element (i.e. weight, color).

physical requirement [**UML for SE RFP**]
A physical property a system must satisfy. See *requirement.*

pin
A pin is an object node for inputs and outputs to actions. [**UML for SE RFP:** See *function port*., input/output.]

port
A port is a property of a classifier that specifies a distinct interaction point between that classifier and its environment or between the (behavior of the) classifier and its internal parts. [**UML for SE RFP:** The part of a system or component that provides access between a system's behaviors and properties, and its environment. Note: this is sometimes referred to as an interaction point.]

postcondition
A constraint epresses a condition that must be true at the completion of an operation.

precondition*
A constraint expresses a condition that must be true when an operation is invoked.

primitive type
A primitive type defines a predefined data type, without any relevant substructure (i.e., it has no parts). A primitive datatype may have an algebra and operations defined outside of UML, for example, mathematically.

probability [**SysML**]
Note: used in activities.

probability distribution [**UML for SE RFP**]*
A mathematical function which defines the likelihood of a particular set of outcomes. See *expression*.

problem [**UML for SE RFP**]*
A deficiency, limitation, or failure to satisfy a requirement or need, or other undesired outcome. Note: A problem may be associated with the behavior, structure, and/or properties of a system or element at any level of the hierarchy (i.e. system of system level, down to a component/part level). See *need, requirement.*

problem cause [**UML for SE RFP**]
The relationship between a problem and its source problems (i.e. cause). Note: This cause ***e***ffect relationship is often represented in fishbone diagrams, fault trees, etc.**

procedure
A set of actions that may be attached as a unit to other parts of a model, for example, as the body of a method. Conceptually a procedure, when executed, takes a set of values as arguments and produces a set of values as results, as specified by the parameters of the procedure.

profile
A profile defines limited extensions to a reference metamodel with the purpose of adapting the metamodel to a specific platform or domain.

property
A property is a structural feature. [**SysML**] A usage of a class that relates the instances of the enclosing class to the instances of the class that types the property.

property path [SysML]*

provided interface

pseudo-state

A pseudostate is an abstraction that encompasses different types of transient vertices in the state machine graph.

rate [SysML]

a specification of the number of objects and values that traverse the edge per time interval, that is, the rate they leave the source node and arrive at the target node. See *discrete rate, continuous rate.*

rationale [**SysML**]

An element that documents the principles or reasons for a modeling decision, such as an analysis choice or a design selection. It provides or references the basis for the modeling decision, and can be attached to any model element.

real [SysML]*

A value type that represents a real number which can have a value from negative infinity to infinity.

realization

Realization is a specialized abstraction relationship between two sets of model elements, one representing a specification (the supplier) and the other represents an implementation of the latter (the client). Realization can be used to model stepwise refinement, optimizations, transformations, templates, model synthesis, framework composition, etc.

receive

The handling of a stimulus passed from a sender instance. See: *sender, receiver*.

receive signal action

receiver

The object handling a stimulus passed from a sender object. Contrast: *sender*.

reception

A declaration that a classifier is prepared to react to the receipt of a signal.

reference

1. A denotation of a model element.
2. A named slot within a classifier that facilitates navigation to other classifiers. Synonym: *pointer*.

reference association [SysML]

reference link [SysML]

reference property [SysML]

A property of a block that refers to another element of a system or system description, but is not owned by the block.

refine*

A relationship that represents a fuller specification of something that has already been specified at a certain level of detail. For example, a design class is a refinement of an analysis class.

region

A region is an orthogonal part of either a composite state or a state machine. It contains states and transitions.

relationship

Relationship is an abstract concept that specifies some kind of relationship between elements.

replicate function [**UML for SE RFP**]

A function which represents the same transformation, but is implemented by separate resources. See *function*.

required interface*

Contrast: *provided interface*.

required parameter [SysML]

A parameter that is required for the activity to begin execution. The lower multiplicity must be nonzero. See *optional parameter, activity parameter node*.

requirement [SysML]

a capability or condition that must (or should) be satisfied.

requirement containment [SysML]

requirement traceability [**UML for SE RFP**]*

The relationship between a source requirement and the derived requirements needed to satisfy the source

requirement. See *requirement, derive, trace.*

requirement type [**UML for SE RFP**]*

A category of requirement. Note: This includes functional, interface, performance, etc.

requirements diagram [SysML]

A diagram that represents requirements and their relationships.See *requirement.*

resource [**UML for SE RFP**]*

Any lement that is needed for the execution of a function. See *resource constraint.*

role*

The named set of features defined over a collection of entities participating in a particular context.
Collaboration: The named set of behaviors possessed by a class or part participating in a particular context.
Part: a subset of a particular class which exhibits a subset of features possessed by the class
Associations: A synonym for association end often referring to a subset of classifier instances that are participating in the association. [**UML for SE RFP:** See *part, system role*.]

run time

The period of time during which a computer program or a system executes.

satisfy [SysML]

A dependency relationship between a requirement and a model element that fulfills the requirement. See: *derive*, *verify*.

scalable [**UML for SE RFP**]

A measure of the extent to which the modeling language (or methodology, etc.), can be adapted to an increase in scope and/or complexity.

scenario

A specific sequence of actions that illustrates behaviors. A scenario may be used to illustrate an interaction or the execution of a use case instance. See: *interaction.*

semantic variation point

A point of variation in the semantics of a metamodel. It provides an intentional degree of freedom for the interpretation of the metamodel semantics.

semantics [**UML for SE RFP**]*

The meaning of a model element. Note: a precise meaning should be able to be expressed mathematically.

send [a message]

The passing of a stimulus from a sender instance to a receiver instance. See: *sender, receiver.*

send signal action*

sender

The object passing a stimulus to a receiver instance. Contrast: *receiver*.

sequence diagram

A diagram that depicts an interaction by focusing on the sequence of messages that are exchanged, along with their corresponding event occurrences on the lifelines.
Unlike a communication diagram, a sequence diagram includes time sequences but does not include object relationships. A sequence diagram can exist in a generic form (describes all possible scenarios) and in an instance form (describes one actual scenario). Sequence diagrams and communication diagrams express similar information, but show it in different ways. See: *communication diagram*.

shallow history state

signal

A signal is a specification of type of send request instances communicated between objects.

signature

The name and parameters of a behavioral feature. A signature may include an optional returned parameter. [**UML for SE RFP**: See *input/output*.

simple state [**UML for SE RFP**]*

A state that does not have nested states.See *state, composite state.*

slot

A slot specifies that an entity modeled by an instance specification has a value or values for a specific structural feature.

software [**UML for SE RFP**]
A component of a system that specifies instructions which are executed by a computer. See *component*.

source requirement [**UML for SE RFP**]
The requirement which is the basis for deriving one or more other requirements.

spatial representation [**UML for SE RFP**]
A geometrical relationship among elements. See *geometric model*.

specialized requirement [**UML for SE RFP**]
A requirement that is not explicitly addressed by the default requirement types. Note: This may include safety, reliability, maintainability, producibility, usability, security, etc.

specification*
A set of requirements for a system or other element.

stakeholder [**UML for SE RFP**]*
Individuals, groups, and/or institutions which may be impacted by the system throughout its life cycle, including acquisition, development, production, deployment, operations, support, and disposal.

standard port [SysML]
An interaction point through which a Block provides and requires a set of services to and from its environment. Synonym: UML Port.

state
A state models a situation during which some (usually implicit) invariant condition holds. [UML for SE RFP: A condition of a system or element, as defined by some of its properties, which can enable system behaviors and/or structure to occur. Note: The enabled behavior may include no actions, such as associated with a wait state. Also, the condition that defines the state may be dependent on one or more previous states.]

state-based behavior [**UML for SE RFP**]
Behavior which is described by states and transitions between states.

state invariant
A stateInvariant is a runtime constraint on the participants of the interaction.

state list

state machine diagram
A diagram that depicts discrete behavior modeled through finite state-transition systems. In particular, it specifies the sequences of states that an object or an interaction goes through during its life in response to events, together with its responses and actions. See: *state machine*.

state machine
State machines can be used to express the behavior of part of a system. Behavior is modeled as a traversal of a graph of state nodes interconnected by one or more joined transition arcs that are triggered by the dispatching of series of (event) occurrences. During this traversal, the state machine executes a series of activities associated with various elements of the state machine.

stereotype
A stereotype defines how an existing metaclass may be extended, and enables the use of platform or domain specific terminology or notation in place of, or in addition to, the ones used for the extended metaclass.

stereotype property

stimulus
The passing of information from one instance to another, such as raising a signal or invoking an operation. The receipt of a signal is normally considered an event. See: *message*.

storage device [**UML for SE RFP**]
A component of a system that is used to store a stored item. Note: this may include memory device, a battery, or a tank. See *component*, *stored item*. *

store requirement [**UML for SE RFP**]
A stored item a system must store. See *requirement, stored item*. *

stored item [**UML for SE RFP**]

An item that persists over time, which may be depletable or non-depletable. Note: Non-depletable stores may include data store in computer memory, and depletable stores may include energy in a battery, or fluid in a tank. Physical stores obey the conservation laws (only take out what is put in). A non-depletable store, such as a data store, is not constrained by the conservation laws. The stored item should be differentiated from the storage device, which stores the item. See *central buffer node, storage device*. Note: This was previously a system store. *

streaming
A property of a parameter that specifies whether its values can be accepted or produced by an action while executing. A non-streaming parameter specifies that the parameter value can only be accepted at the beginning of execution and produced at the end of execution. Contrast: *nonstreaming*.

string
A string is a sequence of characters in some suitable character set used to display information about the model. Character sets may include non-Roman alphabets and characters.

structural feature
A structural feature is a typed feature of a classifier that specifies the structure of instances of the classifier.

structure [**UML for SE RFP**]
The relationships between the components that contribute to the properties of the whole, and enable them to interact (inter-relate).

structural diagram
A form of diagram that depicts the elements in a specification that are irrespective of time. Class diagrams and component diagrams are examples of structure diagrams.

subclass
In a generalization relationship, the specialization of another class, the superclass. See: *generalization*. Contrast: *superclass*.

subject

submachine state
A state in a state machine that is equivalent to a composite state but whose contents are described by another state machine.

substate
A state that is part of a composite state. See: *concurrent state, disjoint state*.

subpackage
A package that is contained in another package.

subsystem
A unit of hierarchical decomposition for large systems. A subsystem is commonly instantiated indirectly. Definitions of subsystems vary widely among domains and methods, and it is expected that domain and method profiles will specialize this construct. A subsystem may be defined to have specification and realization elements. [**UML for SE RFP**:A logical or physical partitioning of a system. See *system, logical component, physical component*.]

subtype
In a generalization relationship, the specialization of another type, the supertype. See: *generalization*. Contrast: *supertype*.

superclass
In a generalization relationship, the generalization of another class, the subclass. See: *generalization.* Contrast: *subclass*.

supertype
In a generalization relationship, the generalization of another type, the subtype. See: *generalization*. Contrast: *subtype*.

supplier
A classifier that provides services that can be invoked by others. Contrast: *client*.

synch state
A vertex in a state machine used for synchronizing the concurrent regions of a state machine.

system
An organized set of elements functioning as a unit. [**UML for SE RFP:** An *item*, with structure, that exhibits

observable properties and behaviors. See *block, part, component, item*. *]

system (component) boundary [**UML for SE RFP**]*

The set of all ports, which connect the system (component) to its environment.

system hierarchy [**UML for SE RFP**]

A decomposition of a system and its components.

system interconnection [**UML for SE RFP**]

The connection between systems and between components. See *connector*.

tagged value

The explicit definition of a property as a name-value pair. In a tagged value, the name is referred as the tag. Certain tags are predefined in the UML; others may be user defined. Tagged values can be properties of a stereotype. See: *constraint, stereotype.*

terminate node

test case [SysML]

A method that is used to verify a requirement has been satisfied.See *requirement, satisfy, verify*. [Note: A test case in SysML is intended to be consistent with a test case in the UML testing profile].

text-based requirement [**UML for SE RFP**]*

One or more requirements specified in text. See *requirement*.

thread [of control]

A single path of execution through a program, a dynamic model, or some other representation of control flow. Also, a stereotype for the implementation of an active object as lightweight process. See *process*. [**UML for SE RFP:** A process with no concurrent functions, and represents a single path of execution.]

time constraint

A TimeConstraint defines a Constraint that refers to a TimeInterval.

time event

A TimeEvent specifies a point in time. At the specified time, the event occurs. See: *event*.

time expression

A TimeExpression defines a value specification that represents a time value.

time property [**UML for SE RFP**]

A property of the model that represents a local or global time, which other properties may depend on. . Note: The property can support continuous or discrete-time models. This variable should not be confused with the measured or computed time that an actual system uses, which depends on a number of implementation specific factors related to clocks, synchronization, etc. See *property*.

time reference [**SysML**]

The time property from which other time properties are derived. See *time property*.

timing diagram

An interaction diagram that shows the change in state or condition of a lifeline (representing a Classifier Instance or Classifier Role) over linear time. The most common usage is to show the change in state of an object over time in response to accepted events or stimuli.

token

topology [**UML for SE RFP**]

A graph of nodes and arcs.

trace

A dependency that indicates a historical or process relationship between two elements that represent the same concept without specific rules for deriving one from the other. Trace dependencies are used to track requirements and changes across models.

trade-off analysis [**UML for SE RFP**]

An evaluation of alternatives based on a set of evaluation criteria.

transition

A transition is a directed relationship between a source vertex and a target vertex. It may be part of a compound transition, which takes the state machine from one state configuration to another, representing the complete response of the state machine to an occurrence of an event of a particular type. [**UML for SE RFP:** Response to

events/conditions, which triggers a behavior.]

transition action

trigger

A trigger relates an event to a behavior that may affect an instance of the classifier.

triggering input [**UML for SE RFP**]

An input which is required for a function to be activated. See *required parameter.*

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. A type may not contain any methods, maintain its own thread of control, or be nested. However, it may have attributes and associations. Although an object may have at most one implementation class, it may conform to multiple different types. See also: *implementation class* Contrast: *interface*.

type expression

An expression that evaluates to a reference to one or more types.

unidirectional connector

unit [**SysML**]*

A standard for expressing a quantity in terms of the magnitudes of other quantities that have the same dimension. A unit often relies on precise and reproduceable ways to measure the unit. For example, a unit of length such as a meter may be specified as a multiple of a particular wavelength of light. See *dimension, value property*.

usage

A dependency in which one element (the client) requires the presence of another element (the supplier) for its correct functioning or im*plementation.

user [**UML for SE RFP**]

An individual or group of individuals that use a system. See *actor.*

use case

A use case is the specification of a set of actions performed by a system, which yields an observable result that is, typically, of value for one or more actors or other stakeholders of the system. See: *use case instances.*

use case diagram

A diagram that shows the relationships among actors and the subject (system), and use cases.

value property [SysML]

A property of a block that holds a value.See *value type*.

value type [SysML]

A type that defines values.

variance [**UML for SE RFP**]

A measure of the distribution about the mean of a probability distribution. Refer to the mathematical definition associated with a probability distribution.

vector [**UML for SE RFP**]

A data type, which specifies a magnitude and direction.

verdict [**SysML**]

The outcome of executing one or more test cases or verification procedures. See *test case, verification procedure, verification result, verify*. Note: This term is borrowed from the testing profile.

verification [**UML for SE RFP**]

The process for demonstrating a system satisfies its requirements.

verification procedure [**UML for SE RFP**]

The functions needed to support execution of a test case. Note. This may include generating an input stimulus and monitoring an output response. See *procedure, manual procedure*.

verification result [**UML for SE RFP**]

The outcome of executing one or more test cases.* See *verdict.*

verification system [**UML for SE RFP**]

The system that implements the verification procedures.

verified [SysML]
A named element that verifies a requirement
verify [SysML]
A relationship between a requirement and a test case that can determine whether a system satisfies the requirement.See *requirement, test case, verdict.*
vertex
A source or a target for a transition in a state machine. A vertex can be either a state or a pseudo-state. See: *state, pseudo-state.*
view [SysML]
A view is a representation of a whole system from the perspective of a single viewpoint. See *model*.
viewpoint [SysML]
A viewpoint is a specification of the conventions and rules for constructing and using a view for the purpose of addressing a set of stakeholder concerns.
visibility
An enumeration whose value (public, protected, or private) denotes how the model element to which it refers may be seen outside its enclosing namespace.
well-formedness rule [**UML for SE RFP**]
A rule which specifies the allowable relationships and constraints among model elements.
XMI