

neither composed of the ATM (as in an aggregation) nor a specific version of the ATM (as in a specialization). The customer briefly associates with the ATM to perform a specific task, the deposit. An instance of the interaction relationship begins at the beginning of the operation and ends at the end of the operation. Process entities can be useful to record the occurrence of this relationship. In this case, a bank transaction model might contain attributes for the customer identity, the ATM identity, the date, and the dollar amount deposited. An instance of the bank transaction holds an instance of the customer, and instance of the ATM, a date value, and a dollar amount value. This data can then be stored in a database to correctly update the customer's balance.

[0135] Metamodels, feature descriptors, and other descriptors are descriptions that represent interfaces to be implemented in a metamodeling tool. Metamodels describe how a model should be implemented. Metamodels are a type of descriptor. Metamodels are comprised of descriptions of smaller parts. These smaller parts are also described using descriptors. Not everything that can be described is a model or part of a model. Descriptors can be used to describe these items too.

[0136] A descriptor defines an implementation. The descriptor contains descriptions of the features and functionality allowed and required in an implementation. The descriptors of the present invention are interfaces that must be implemented to participate in the modeling tool. Different modeling tools may implement the same descriptor interface differently. Different modeling tools may create different implementations based on the descriptors.

[0137] A class is an implementation specific to object-oriented programming. All classes are implementations.

[0138] In one embodiment, the present invention provides a type of meta-implementation called an accessor. An accessor is a meta-implementation containing the descriptor and behaving as an implementation by delegating to a real implementation. No metamodeling equivalent exists for an accessor. A virtual implementation is a meta-implementation containing the descriptor and behaving as an implementation by assembling other implementations to create a larger or more complex implementation. The most basic virtual implementations correspond to the most basic parts of a typical programming language: variables, loops, flow control, and method calls.

[0139] Each descriptor has a name and may have relationships with the following elements as well: a display name, a description, hint instances and role instances. A descriptor has a one-to-one association relationship with a name that provides a name for the mode. Any tool using the model uses this name. The name may be a string or a more complex object. A descriptor has zero or one association relationship with a display name that provides a name to display to a user. The display name may be presented to human users as a more attractive alternative to the name. A descriptor has a zero to one association relationship with a description that provides details about the model for its correct use. A description may be useful for human users and automated documentation. A descriptor has a zero-to-many association relationship with hint instances that are name-value pairings, which add details to the metamodel that cannot adequately be captured anywhere else. A descriptor has a zero-to-many

association relationship with role instances that are each a group of related hints found to be commonly occurring.

[0140] A metamodel describes a model, but does not implement that model or know how to access the various features of a specific model implementation. Each descriptor also has no signals. No additional events are added by the failure descriptor.

[0141] A failure descriptor of the present invention is a descriptor and holds the identity of the descriptor that is the type of error thrown from an operation when an error occurs. Failure Descriptors describe the errors that may occur when an operation is attempted. Failure descriptors do not model the failure; a metamodel describes the failure. A failure descriptor has no operations. In a failure descriptor, a descriptor change event is fired whenever an attribute value is added, changed or removed from the failure descriptor.

[0142] A constraint descriptor of the present invention is a feature descriptor representing a check that results in a true or false condition. If the condition is true, an activity can continue. If false, a failure is thrown to stop the current operation from occurring. A wide variety of constraints can exist including maximum and minimum number of occurrences (occurrence constraint), maximum and minimum values for an attribute (value constraint), possession of security credentials (access constraint), and many more. The constraint descriptor does not attempt to enforce a constraint, only to describe it. The name of the constraint may describe what the constraint enforces. The name of each data instance indicates the purpose of the value as it applies to a particular constraint implementation. The values of each data instance configure the conditions of the constraint instance.

[0143] It is important to note that constraint descriptors are mapped to a "constraint implementation" holding the model implementation that actually implements that constraint. The constraint description needs to be captured and is mapped to a constraint implementation for consistency, but the implementation of a constraint is a model just like any other model in the system.

[0144] A constraint descriptor has a one-to-one association relationship with a configuration that provides the values for each attribute in the model descriptor necessary to setup the constraint. A constraint descriptor describes a constraint, but does not implement that constraint. Also, no additional events are added by the constraint descriptor.

[0145] A feature descriptor of the present invention is a descriptor held by another descriptor. Many of the descriptors described below, such as attributes and methods are feature descriptors. Every feature descriptor has several common characteristics. A feature descriptor has a one-to-one association or aggregation relationship with a parent, the object holding the feature descriptor. A feature descriptor also has a zero-to-many association relationship with access constraint instances that are restrictions to prevent access to the feature implementation unless the user holds certain credentials. A feature descriptor describes an abstraction of various other types useful for describing "features" of a model. It does not perform any operations. Also, no additional events are added by a feature descriptor.

[0146] A datatype descriptor of the present invention is a descriptor describing an item of data. A data descriptor may