

ference between a component assembly and a model is duration. The component assembly is generally used for a shorter period of time to perform its specific task and then disassembled. If a component assembly is used for a longer period of time, it should be formally examined and a model implementation created.

[0122] Persistence engines provide a mechanism for storing attribute data and state information for components to a storage device like a hard drive or database. The data from the storage device can then be used to recreate the components at a later time. The storage device “persists” the data. A persistence engine is a software component that accepts a software component or piece of data and stores that data to the storage device. The persistence engine also knows how to retrieve that data from the storage device. Advanced persistence engines may support queries to retrieve specific components or sets of components from the storage device and may also support different transaction levels to ensure proper synchronization across multiple users accessing data simultaneously.

[0123] Although the component integration engine depicted in FIG. 3 includes component assemblies, persistence engines, and flow chart assemblies, depending on the application, a component engine of the present invention may not include component assemblies, persistence engines and flow chart assemblies. Also, it should be understood that not all component integration engines of the present invention include all of the types of instances of the component integration engine shown in FIG. 3

[0124] An enumeration instance is a datatype instance holding a single value selected from the choices listed in the enumeration.

[0125] Other element instances or “future element instances” may be created by other element implementations described by other element descriptors. Instances of these future elements will represent a specific item following the implementation described by the element’s implementation. That implementation will be described by a descriptor for the future element type. These other element instances will operate in a manner appropriate to the new technology paradigm or technology that required the definition of a new element type.

[0126] FIG. 4 illustrates that metamodels of the present invention extend other metamodels of the present invention, and metamodels describe models. Models of the present invention may also extend other models described by the model’s parent metamodel. A dog metamodel of the present invention extends an animal metamodel, and a dog model of the present invention extends an animal model.

[0127] In FIG. 4 the base model attribute of the Dog metamodel is a pointer to the Animal metamodel. It is very important to note the metamodel does not contain an implementation of the model like a class does in object-oriented programming. A metamodel clearly and exactly describes a model, but the details of implementing the model are not included.

[0128] A complete metamodeling tool contains the ability to accurately detail relationships between models. A relationship is a link or connector between models. A relationship is described as occurring between two models in the same way that a model describes the item it represents. In the

same way that an instance of a model represents one occurrence of that model, an instance of a relationship represents one occurrence of the relationship. An instance of a relationship occurs between two or more instances of models. Some type of relationships used in the present invention: association (“has-a”, “uses”, “knows about”), aggregation (“has-a”, “part-of”), generalization/specialization (“is-a”, “extends from”), and interaction (“communicates-with”).

[0129] An association is a relationship of one model that is aware of another independent model. This type of relationship is frequently defined through the use of attribute descriptors. A model instance holding an attribute “aware-of” the instance value assigned to that attribute. If an instance of a model holds an instance of another independent model in an attribute, then the relationship is an instance of an association relationship. When two models have an association that is not captured by either model, an association relationship may still be created. In order to capture the relationship, a model is created to represent that relationship. This model is an association model containing one attribute for each model participating in the association relationship. Static attributes create these association relationships for all instances of the model. Instance attributes create association relationships for a single instance of the model.

[0130] An aggregation is a relationship between a model and one of its parts. This type of relationship is frequently defined through the use of attribute descriptors. A model instance holding an attribute “has-a” value assigned to that attribute. An aggregation differs from an association in that the part is not independent.

[0131] A generalization/specialization relationship exists when a model extends a base model.

[0132] An interaction relationship is a type of relationship that can be described as “communicates with”, “calls upon”, or “makes use of”. Operation descriptors serve to describe interaction relationships. An instance of the model holding the operation interacts with instances of those models passed as parameters. Interaction relationships may also be established by an attribute holding a value later used by an operation. In this case, the distinction between an association relationship and an interaction relationship becomes blurred.

[0133] Interaction relationships may also be modeled by creating a process entity model. A process entity model does not represent a real-world object, but instead represents the execution of a process or the occurrence of an event. The process entity contains an attribute to hold each of the items associated by the process. Student enrollment, banking deposits, product orders, and line items are examples of process entities. In the figures described below, interaction relationships are shown by a line between the two entities. The cardinality, or number of occurrences allowed, is shown on each end of this line. To distinguish between an interaction and other relationships, the interaction line has an arrow from the model initiating the interaction to the other model.

[0134] An example of the interaction relationship is a bank deposit or withdrawal where a customer makes use of an automated teller machine (ATM). The customer exists independently of the ATM but is not constantly aware of the ATM (as in an association relationship). The customer is