

COMPONENT INTEGRATION ENGINE

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the priority of U.S. Provisional Patent Application No. 60/484,251, entitled "Component Integration Engine," filed Jul. 2, 2003 and the entire disclosure and contents of this provisional patent application are hereby incorporated by reference.

BACKGROUND

[0002] 1. Field of the Invention

[0003] The present invention relates to a computer method and system for creation of computer software.

[0004] 2. Related Art

[0005] Object-oriented programming distributes data and functionality among different objects. Dividing a system into objects tends to increase reusability of code. Many programs that require the same groupings of features and functionality can reuse a model. When programs require similar groupings of features and functionality, object-oriented techniques like model inheritance and method overriding allow most of the code to be reused while adding new methods and attributes or varying existing methods and attributes. Inheritance does not allow the removal of methods or attributes.

[0006] Objects connect to other objects by attribute references, parameter references, and method calls into the other objects. Having many connections between objects tends to reduce the reusability of code. The higher the number of connections between objects, the more the object requires the other objects in order work correctly. When objects require other objects in order to perform properly, they behave more like a single, larger object, working against the goal of dividing a system into smaller objects.

[0007] Many differences exist even between programs of a similar nature. These differences require the addition and variation of methods and attributes through inheritance. These small differences lead to a proliferation of models, each of which is only slightly different from the others. This proliferation increases costs associated with software maintenance since new models need to be documented, tested, debugged, and maintained.

[0008] In order to increase code reuse and reduce model proliferation, resulting in lower cost of development and maintenance, an approach is needed to reduce the number of connections between models and reduce the number of attributes and number of methods that needs to be changed through inheritance.

SUMMARY

[0009] According to a first broad aspect of the present invention, there is provided a meta-implementation layer comprising: a metamodel repository containing a plurality of descriptors; a plurality of implementations for providing access to software components described by the plurality of descriptors; a metamodel repository including a plurality of metamodel descriptors for describing the descriptors and a plurality of metamodel implementations for describing the implementations, wherein the meta-implementation

layer provides access to an implementation of the plurality of implementations to thereby allow a user to have access to the software components of a software program.

[0010] According to a second broad aspect of the invention, there is provided a component integration engine comprising: a meta-implementation layer for allowing a user to have access to components of a software program; a plurality of component integration instances for providing access to software component instances to thereby allow the software component instances to be assembled in the software program; communication means for allowing a user to communicate with the component integration engine; and assembly means for assembling the component integration instances to build the software program.

[0011] According to a third broad aspect of the invention, there is provided a computer system having a meta-implementation layer stored therein, wherein the meta-implementation layer comprises: a metamodel repository containing a plurality of descriptors; a plurality of implementations for providing access to software components described by the plurality of descriptors; a metamodel repository including a plurality of metamodel descriptors for describing the descriptors and a plurality of metamodel implementations for describing the implementations, wherein the meta-implementation layer provides access to an implementation of the plurality of implementations to thereby allow a user to have access to the software components of a software program.

[0012] According to a fourth broad aspect of the invention, there is provided a storage medium including instructions stored thereon that when executed by a computer system produce a meta-implementation layer, wherein the meta-implementation layer comprises: a metamodel repository containing a plurality of descriptors; a plurality of implementations for providing access to software components described by the plurality of descriptors; a metamodel repository including a plurality of metamodel descriptors for describing the descriptors and a plurality of metamodel implementations for describing the implementations, wherein the meta-implementation layer provides access to an implementation of the plurality of implementations to thereby allow a user to have access to the software components of a software program.

BRIEF DESCRIPTION OF THE DRAWINGS

[0013] The invention will be described in conjunction with the accompanying drawings, in which:

[0014] FIG. 1 is a block diagram illustrating a meta-implementation of the present invention;

[0015] FIG. 2 is a block diagram illustrating the metamodel repository of the meta-implementation layer of FIG. 1;

[0016] FIG. 3 is a component integration engine including the meta-implementation layer of FIG. 1;

[0017] FIG. 4 is a block diagram showing relationships between metamodels, other metamodels, and models used with the meta-implementation layer and component integration engine according to one embodiment of the present invention;

[0018] FIGS. 5-34 are a block diagram, in multiple sections, illustrating how a component integration engine according to one embodiment of the present invention operates;