GLOSSARY

abstract class	A class which cannot be instantiated, for example, **ActiveValue**.
active value	The mechanism that carries out access-oriented programming for variables in LOOPS. Active values send messages as a side effect of having an object's variable referenced.
activeValue	The previous implementation of the active value concept.
ActiveValue	An abstract class that defines the general protocol followed by all active value objects.
annotatedValue	A special Interlisp-D data type that wraps each **ActiveValue** instance.
AnnotatedValue	An abstract class that allows an annotatedValue to be treated as an object.
browser	A window that allows you to examine and change items in a data structure.
class	A description of one or more similar objects; that is, objects containing the sames types of data fields and responding to the same messages.
class inheritance	The means by which a class inherits variables, values, and methods from its super class(es).
class lattice	A network showing the inheritance relationship among classes.
class variable (CV)	A variable that contains information shared by all instances of the class. A class variable is typically used for information about a class taken as a whole.
inheritance	The means by which you can organize information in objects, create objects that are similar to other objects, and update objects in a simplified way.
Inspector	A Lisp display program that has been modified to allow you to view classes, objects, and active values.
instance	An object described by a particular class. Every object within LOOPS is an instance of exactly one class.
instance variable (IV)	A variable that contains information specific to an instance.
instantiate	To make a new instance of a class.
lattice	An arrangement of nodes in a hierarchical network, which allows for multiple parents of each node.
Masterscope	A Lisp Library Module program analysis tool that has been modified to allow analysis of LOOPS files.
message	A command sent to an object that activates a method defined in the object's class. The object responds by computing a value that is returned to the sender of the message.
metaclass	Classes whose instances are classes or abstract classes.
method	What an object applies to the arguments of a message it receives. This is similar to a procedure in procedure-oriented programming, except that here, you determine the message to send and the object receiving the message

	determines the method to apply, instead of the calling routine determining which procedure to apply.
mixin	A class that is used in conjunction with another class to create a subclass. Mixins never have instances, and hence have **AbstractClass** as their metaclass.
object	A data structure that contains data and a pointer to functionality that can manipulate the data.
property list	A place for storing additional information on classes, their variables, and their methods.
selector	Part of a message that is sent to an object. The object uses the selector to determine which method is appropriate to apply to the message arguments.
self	A method argument that represents the receiver of the message.
specialization	The process of creating a subclass from a class, or the result of that process.
subclass	A class that is a specialization of another class.
super class	A class from which a given class inherits variables, values, and methods.
Tofu	An acronym for Top of the universe, which is the highest class in the LOOPS hierarchy.
Unique Identifier (UID)	An alphanumeric identifier that LOOPS uses to store and retrieve objects. Objects do not have UIDs unless they are named, are instances of indexed objects, or are instances printed to a file.
wrap	Objects have fields that can contain data. Some **ActiveValue** can be added so this data is stored within it. When this occurs, the **ActiveValue** wraps the data.