NISTIR 7324

Analysis Catalog for the Focus 3D Telemodeling Tool

Arthur F. Griesser, Ph.D.

National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

NISTIR 7324

Analysis Catalog for the Focus 3D Telemodeling Tool

Arthur F. Griesser, Ph.D.
Semiconductor Electronics Devision
Electronics and Electrical Engineering Laboratory

August 2005

U.S. DEPARTMENT OF COMMERCE
Carlos M. Gutierrez, Secretary
TECHNOLOGY ADMINISTRATION
Michelle O'Neill, Acting Under Secretary of Commerce for Technology
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
William A. Jeffrey, Director

Analysis Catalog for the Focus 3D Telemodeling Tool

Abstract

A high level view of the requirements for the Focus 3D modeling tool is presented. The scope of the project is defined, users of the system are described, an analysis class diagram is presented, and use cases are outlined.

Table of Contents

Overview

The purpose of this catalog is to provide a high level view of the requirements for the Focus 3D telemodeling tool.

The goal of this project is to improve the quality of information transfer standards by providing them with underlying information models. Most standards describe in excruciating detail the format of messages. Frequently missing, however, are models of the information that's sent; even Unified Modeling Language (UML) class models (arguably the most fundamental) are often missing. Well constructed models provide a solid foundation for standards: they show and clarify all the relationships between data (not just those relationships necessary for data transmission). Models can also bridge the communication gaps between standards participants in different domains. Defects in information standards can often be traced to inadequate or missing models.

There are several reasons why standards so frequently omit class models. These models are unfamiliar to most of the volunteer domain experts participating in the standards process; it is difficult to perceive the importance of something that isn't understood. A second problem is lack of tool support for collaborative modeling. Existing modeling tools are controlled by single user. Supplemental tools such as web based desktop sharing can let other experts share the model, but this is so cumbersome as to be useless. A tool that makes it easier for domain experts to participate in collaborative modeling will result in standards with fewer ambiguities, inconsistencies, and holes.

Focus is intended make domain modeling easier by making the models less abstract and more concrete; domain classes would become three dimensional objects that resemble the instances they represent. Users could reach out and pick up a class and move it in the model, or change its properties. Multiple users could simultaneously view and manipulate the model. There are side benefits from collaborative modeling: when people work together in a shared physical space, they use gestures (such as pointing to an object) together with language that differs significantly from the language used when the people are connected only by an audio channel. In fact it has been shown that video conferencing, despite the extra layers of communication that can be conveyed by the video channel, results in language almost identical to that used in audio-only communications. Augmented reality or virtuality environments, on the other hand, result in communications similar to those used when the participants are face-to-face. Furthermore, the large muscle movements required for modeling in a virtual environment should be beneficial to "kinesthetic learners," who constitute about half the population. These factors strongly suggest that augmented environments may be more effective for collaboration than video-conferencing. Similar environments have been designed for physical modeling (of automobile dash boards, for example), but there appears to be no existing environment for domain modeling. The environment is called "Augmented Virtuality" rather than "Augmented Reality" because it is closer to the "Virtual" end of the spectrum than the "Real" end of the spectrum: the only real part of the environment will be the faces of users.

Glossary

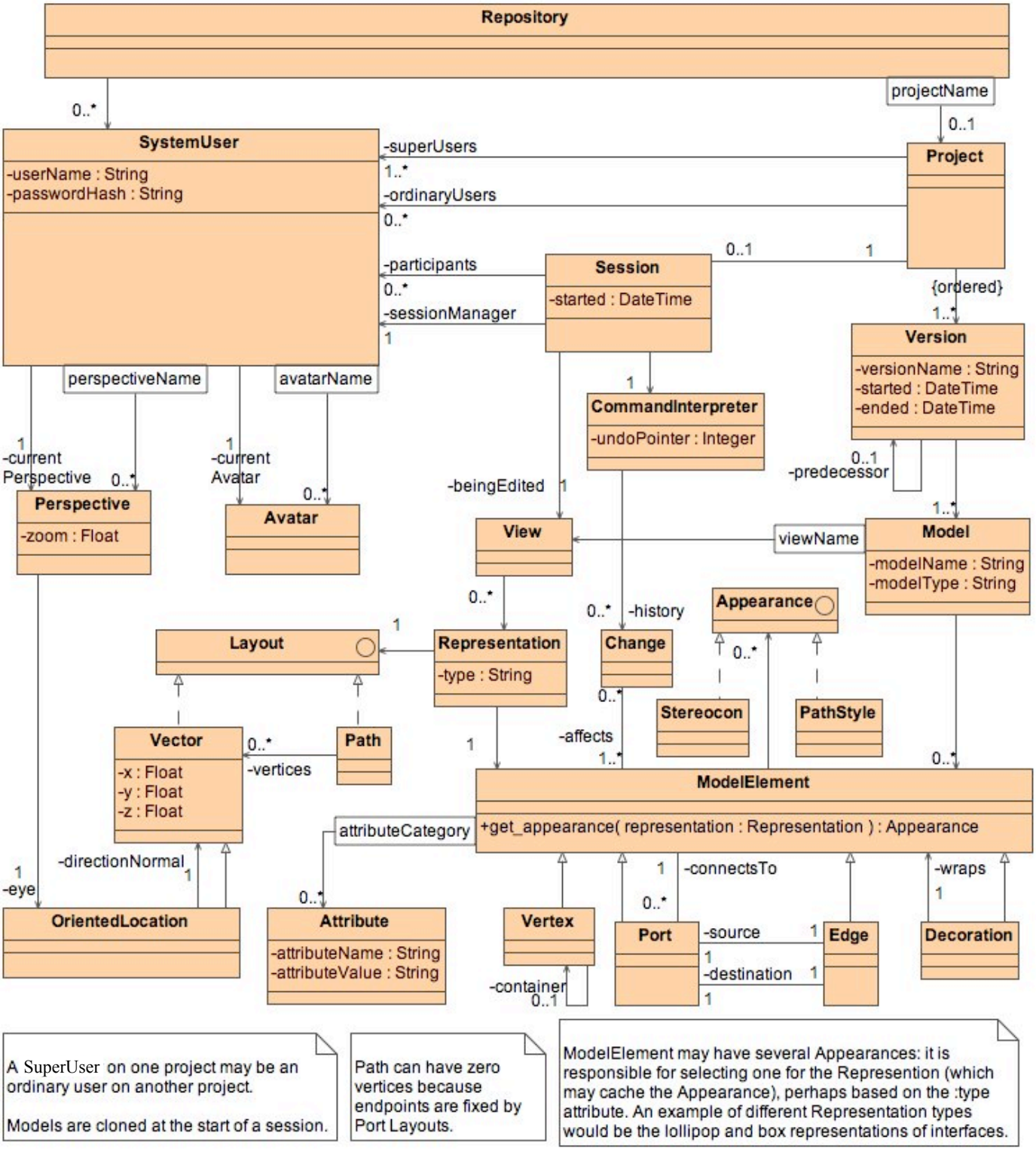
Name	Description
Avatar	A virtual-world surrogate for a SystemUser. Initially Avatars will probably be cartoon-like, but it would be desirable to provide them with real time expressions obtained from their SystemUser. A single SystemUser may have several Avatars, selected at his or her whim.
Change	A modification of a View or Model.
Decoration	Wrapping behavioral delegate from the Decorator Pattern described in *Design Patterns, Elements of Reusable Object-Orented Software*, by Gammma, Helm, Johnson, and Vlissides. For example, a composition diamond is a Decoration that changes the appearance of a Port.
Layout	Spatial positioning information.
Path	Spatial positioning information for an edge.
Perspective	A viewpoint from which a SystemUser regards a View.
Port	A connector that attaches an Edge to a Vertex.
Position	A location and orientation of shape in three dimensional space: spatial positioning information for a Vertex or Port.
Presence	An effective user: the combination of a SystemUser, together with that user's choice of Avatar and Perspective.
Project	Primarily a collection of models. Also a context for a modeling Session, and therefore aware of the potential roles of possible SystemUsers.
Model	A UML-like model. Initially only a model resembling a class model, but it would be desirable to eventually support 3D analogs of all UML model types. A Model by itself does not include Layout information, which resides in a View. A Model may have several associated views, with different levels of detail.
Repository	A global singleton object that holds onto all persistent information.
Session	A system use cycle.
Stereocon	A stereoscopic icon.
Vector	The beloved object from mathematics and physics.
Version	A version of a Project, capturing the complete state of all the Models composing the Project. Aware of the Version from which this Version was derived.
Vertex	A node in a Model. Logically point-like, although the representation may fill some virtual space.
View	A three dimensional arrangement of the ModelElements in a Model. Different Views of the same model may emphasize different aspects of the Model.

Actor Catalog

Hierarchy

Actor	Description
SystemUser	Anyone who interacts with FOCUS in any way.
Analyst	Someone who develops analysis models, as part of the requirements definition effort. Requirements usually begin as imprecise ideas that define a problem and pose a solution to that problem. These ideas need to be elaborated and scrutinized for consistency and clarity.
Domain expert	Someone with a lot of experience using the information being modeled. Domain experts provide information to analysts and review the accuracy of the analysis models.
Architect	Someone who uses analysis models to create a high level design of a solution to the problem examined by the analyst. This architecture is intended to ensure that the solution meets the functional and non-functional requirements articulated by the analysts.
Designer	Someone who derives design models from the analysis models, subject to the constraints of the architecture.
Developer	Someone who implements the design models.
Tester	Someone who defines tests necessary to ensure that implementations produced by developers actually conform to the requirements articulated by the analysts.
SuperUser	A SystemUser (of any type) with more modeling experience (and better familiarity with FOCUS) than most. SuperUsers can become SessionManagers. When they do, they acquire extra powers and responsibilities.
SessionManager	A role played by one SystemUser per active Session. The SessionManager initiates a collaborative modeling session, controls who participates in the system, etc. This role might be played by one person throughout the modeling session, or it might be passed from one SystemUser to another.

SystemUsers can perform most operations on Models. Some operations, however, are restricted to SessionManagers, who control the session. Some operations are restricted to SuperUsers: these are more complicated, infrequently performed, or potentially dangerous. SuperUsers need to be more familiar with the tool and with modeling techniques than other SystemUsers. Modeling sessions can be held only under the supervision of a SessionManager, who is responsible for the status of the Model. A single user can therefore not modify the model alone, unless he or she is a SessionManager. This may seem restrictive compared to existing techniques of passing around documents. The idea here is to reduce gratuitous changes, and merges, and rework by requiring collaboration.


Repository
projectName
0..*
0..1
SystemUser
-userName : String
-passwordHash : String
-superUsers
1..*
-ordinaryUsers
0..*
Project
-participants
0..*
-sessionManager
1
Session
-started : DateTime
0..1
1
{ordered}
1..*
Version
-versionName : String
-started : DateTime
-ended : DateTime
perspectiveName
avatarName
1
-current Perspective
0..*
1
-current Avatar
0..*
Perspective
-zoom : Float
Avatar
1
CommandInterpreter
-undoPointer : Integer
0..1
-predecessor
-beingEdited
1
View
viewName
Model
-modelName : String
-modelType : String
0..*
0..*
-history
Appearance
Layout
1
Representation
-type : String
Change
0..*
Stereocon
PathStyle
Vector
-x : Float
-y : Float
-z : Float
0..*
-vertices
Path
1
-affects
1..*
ModelElement
+get_appearance(representation : Representation) : Appearance
attributeCategory
-directionNormal
1
1
-eye
0..*
1
-connectsTo
0..*
-wraps
1
OrientedLocation
Attribute
-attributeName : String
-attributeValue : String
Vertex
Port
-source
1
1
-destination
1
1
Edge
Decoration
-container
0..1
A SuperUser on one project may be an ordinary user on another project.
Models are cloned at the start of a session.
Path can have zero vertices because endpoints are fixed by Port Layouts.
ModelElement may have several Appearances: it is responsible for selecting one for the Represention (which may cache the Appearance), perhaps based on the :type attribute. An example of different Representation types would be the lollipop and box representations of interfaces.

Use Case Catalog

Use cases split into “System Use Cases,” which have people as Principal Actors, and “Component Use Cases” which describe internal functionality necessary for envisioned operation of the system, but which do not have human principal actors.

All of the use cases in this catalog are high-level “low granularity” use cases. Expanded low-level “high granularity” use cases (when written) are in other documents.

CRUD is an acronym standing for **C**reate **R**etrieve **U**pdate **D**elete. Unless otherwise specified, Retrieve is expected to include both searching and browsing for the item to be retrieved.

The status column (denoted by “s” in the header) contains the following values:

Value	**Description**
<blank>	Use case not yet written, but the use case is expected to be required.
C	Catalog’s description appears to be sufficient; do not expect need to produce more explicit use case.
N	Need explicit use case, but not yet written.
P	Preliminary explicit use case available, but not yet reviewed.
R	Explicit use case reviewed internally.

System Use Cases

Summary Goal	S	User Goal	Principal Actor	Description
Session Management				Control collaborative modeling sessions
	C	CRUD session	SessionManager	Session attributes include session-name, password, allowed users, security options, show grid, snap to grid, and minimum distance between users.
	C	Start/stop session	SessionManager	Initiate or terminate a session. SystemUsers can only participate in active sessions.
	C	Handoff management	SessionManager	The SessionManager gives up this role, conferring it on another SuperUser.
	C	Browse active sessions	SystemUser	SystemUsers find the session they wish to join.
	C	Enter/exit session	SystemUser	SystemUsers begin or stop participating in a session.
	C	Change edit target	SessionManager	Specify the Model and View being edited.
	C	CRUD Project	SessionManager	Manage information about the project.
	C	CRUD Model	SessionManager	Manage information about the model.
	C	CRUD View	SessionManager	View is a way of looking at a model. It includes layouts of ModelElements. Some views might be constrained to 2D.

Edit Model				Changes to the currently active Project, Model, and View.
	N	Browse ModelElements	SystemUser	View a sorted table of ModelElements. ModelElements selected in this table are also selected in the model (and vice-versa).
	N	Perspective modification	SystemUser	Each SystemUser can individually zoom, translate (move), and rotate the View. This is actually a change in the Perspective in use. This use case is different from the R in CRUD Perspective (which lets the SystemUser jump to some other predefined Perspective).
	R	CRUD ModelElements	SystemUser	Expect C,D to be accomplished by gestures. R=Search for model element (by name or attributes).
	R	Layout Representations	SystemUser	Change the position of Stereocons, the position on the Stereocon that the edge is attached to, and the routing of edges.
	N	CRUD Stereotype	SuperUser	Includes rules that determine the appearance of ModelElements (taking Stereotypes into account).
		Assign Stereotype	SystemUser	The ModelElement's appearance may change when the Sterotype is assigned.
		CRUD Stereocon	SystemUser	Stereocon is 3D icon representing vertex.
	R	Select Representation	SystemUser	Presumably by grasping.
	N	Supress/Show detail	SystemUser	Details can be hidden to provide a higher level view, or to make it easier for the user to focus on some other detail.
	N	Undo/Redo	SystemUser	Undo or redo any changes made during the current session. Candidate changes can be selected from a list. Multiple changes can be chosen at one time. The system knows which changes can and cannot commute, and allows only compatible sets of changes.
	N	Automatically layout ModelElements	SuperUser	This is available only to the SessionManager because of the large number of Changes it produces.
	N	Rollback	SuperUser	Revert the current Project, Model, View, or ModelElement to its state at the end of some other Session.
		Get help	SystemUser	Includes search on help text, as well as hints specific to nearby objects.
		Refactor	SystemUser	Reorganize the model by changing the Vertex that an Edge is attached to, the parent that contains a model element (such as the Package that a Class belongs to, the SuperState of a State), etc.

I/O				Generate or consume related information.
	N	Generate report	SystemUser	Report on metrics such as complexity, problems with the model (such as containment cycles), effort to implement, etc.
	N	Generate code	SystemUser	Use templating system to support multiple computer languages. Code should include XML Schemas as well as executable code.
	N	Reverse engineer code	SuperUser	Automatically build and layout models that will recreate the input code when "Generate code" is invoked.
		Print	SystemUser	Print a View of the Model as seen from the requesting user's Perspective
	N	Export	SystemUser	Export Project, Model, or ModelElement to some other format, such as XML Metadata Interchange (XMI).
	N	Import	SuperUser	Import a Project, Model, or a fragment from some other format, such as XMI.
User Preferences				User maintains his or her unique options.
		CRUD Perspective	SystemUser	Includes Zoom, translate (move), and rotate. R includes switching to a saved Perspective and browsing Perspectives: the model is shown (from some default perspective), with Perspectives represented by Vectors. Can show perspectives in use by other SystemUsers. Can only switch to a Perspective that is not too close to a perspective in use by another SystemUser, or within the model. These constraints are present in the physical world and easily implemented through cylindrical approximations. They prevent, for example, a perspective inside the head of a different Avatar. This is not intended to prevent a user from borrowing the Perspective of another. When this is done, the Avatar of the borrower should be modified (for example, it might become transparent) to indicate she is "out of body." The Avatar of the lender should be modified as well.
		CRUD Avatar	SystemUser	SystemUser can adjust the appearance of his own Avatar and can also override the appearance of the Avatars of the other users.

Component Use Cases

Summary Goal	S	User Goal	Description
Virtuality			To the SystemUser, ModelElements appear to exist as tangible three dimensional objects.
	C	Head tracking	The system needs to know the SystemUser's head position in order to compute the video to feed into the head mounted display.
	C	Hand tacking	The system needs to know the SytemUser's hand position to determine which objects the user is intending to manipulate and to determine the avatar's hand position.
	C	Projection	The current View is projected on to the SystemUser's Perspective to obtain video to feed into a headmounted display.
Reality			Aspects of the physical world are present in modeling space.
		Facial video	Video of each SystemUser's face is superimposed on that SystemUser's Avatar.
		Environmental video	ModelElements may appear to be superimposed upon each SystemUser's phyical space.
Collaboration			Multiple SystemUsers collaborate to construct and improve the Model.
	C	Change distribution	When one SystemUser makes a change to the model, the change needs to be distributed to other SystemUsers participating in this session. Changes to the SystemUser's Perspective and video of the user's face are also distributed so that the user's Avatar can be updated.
	C	Scene update	Each SystemUser's Scene must be updated with changes distributed by other other users and with changes to the Avatars of the other users.
Object Management			Control over persistent objects.
	C	Authenticate user	Verify the identity of users logging onto the system: associate the user with the appropriate SystemUser object.
	C	Abstract CRUD	Generalized persistence services.

Issues

Name	Description		Resolution
2D > 3D	Some aspects of UML are 2D specific. It is not immediately obvious how to represent this information in 3D.		
	Problem	**Possible solutions**	
	Class Attributes, Operations	Pop-up cluster surrounding the stereocon, or popup table.	
	Arrowheads, Composition, Aggregation, Qualifier, multiplicity, labels	Same as 2D UML, but different for each user, or use new 3D representations (perhaps derived from Entity Relation Diagram [ERD] multiplicity representations).	
	Notes	Stereocon (notebook?) that expands to 2D note.	
	Distinguish a class from instance	Perhaps by size, brightness, internal illumination, transparency, color, texture, aura, halo, crown, elevation.	
	Abstract domain objects (not clear what 3D shape should be used)	Morph together the concrete classes Stereocons? Reserve a Stereocon (an amorphous blob?) for abstract classes. Of course this could be overridden.	
	Nested states	Translucent outer states	
Stereocon name	There must be some better name for “stereocon.”		
Stereocon definition	It would be disruptive to have to create a new stereocon for each class. Perhaps use a simple geometric object until users decide on a representation.		
Update	The brief descriptions in version 0.3 need to be updated for those use cases that now have complete definitions.		

Revision history

(last entry applies to the current document)

Version	Date	Developer	Change
0.1	12/30/04	AFG	First cut
0.2	1/5/05	AFG	• Changed name from AVUML to Focus. • Corrected scope names. • Expanded Glossary definition of Decoration. • Rejected request that SuperUser not be more experienced with modeling and/or Focus. SuperUser is defined by extra powers, which require extra knowledge. • Accepted request that SessionManager not require more modeling experience than SuperUser. Changed Actor from SessionManager to SuperUser for: CRUD_Stereotype, Automatically_Layout_ModelElements, Rollback, I/O:Reverse_engineer_code, I/O:Import (all user goals within summary goal, unless otherwise indicated). • Rejected request to drop distinction between icon and stereocon (icons **look** 3D, but are actually planar, such as some Croquet icons, whereas stereocons actually **are** 3D and have underlying 3D models). The program needs to handle these differently. • Rejected request to add 5-change block-undo bookmark. • Added list of undo/redo candidates, knowledge of change commutativity. • Clarify EditModel.Refactor (more details will be added to full use case). • Rejected request to drop perspective proximity constraint (but added explanation of why this is important). • Accepted request to share perspectives. • Added explanation of ObjectManagement:AbstractCRUD.

0.3	8/26/05	AFG	• Changed meaning of blank status from " Use case not yet written, but the use case is expected to be required" to "Need for explicit use case not yet determined."
			• Added N "Need explicit use case, but not yet written" status.
			• Updated use case statuses.
			• Added "or within the model" and " and easily implemented through cylindrical approximations" to UserPreferences.CRUD_Perpective.
			• Renamed Edit_Model.Select_ModelElements to Edit_Model.Select_Representation.
			• Improved analysis model:
			  • Merged Presence into SystemUser by giving a currentPerspective and currentAvatar to SystemUser.
			  • Renamed Position to OrientedLocation so that name conveys directional information.
			  • Made OrientationLocation a subclass of Vector to give it translation and rotation behaviors.
			  • Version's predecessor is optional because the first Version does not have a predecessor.
			  • Gave Session, a CommandInterpreter, and moved undoPointer there.
			  • Made Location an interface.
			  • Change allowed to affect arbitrary number of ModelElements (required by "move," for example)
			  • Made association from Model to ModelElement non-navigable and added multiplicity at ModelElement end.
			  • Decoration made a ModelElement and association reversed to correspond more closely to gang-of-four pattern. Glossary entry updated to correspond.
			  • Clarified the role of Layout adding an explicit Representation class and making Layout a purely geometric associate of Representation (instead of an association class with the responsibilities of Representation).
			  • Added Appearance interface, ModelElement given the responsibility of selecting Appearance based on its properties and those of the Representation.
			  • Stereocon made an Appearance, PathStyle added.
			  • Added Attribute.
			  • Port now connects to ModelElement instead of Vertex (needed to represent association classes, attachment of notes).