Modellierung hierarchischer Strukturen in VR-Applikationen

Ekkehard Beier
TU Ilmenau, Fakultät für Informatik und Automatisierung
D-98684 Ilmenau
Ekkehard.Beier@PrakInf.TU-Ilmenau.DE

Zusammenfassung

Die Modellierung hierarchischer Strukturen in 3D/VR-Applikationen ist eine Voraussetzung, um einerseits die Komplexität und die Beziehungen realer Produkte abzubilden, andererseits ein Weg, um intuitive und ergonomische direktmanipulative Benutzerschnittstellen zu realisieren.

Der nachfolgende Beitrag beschreibt diese Problematik z.B. anhand existierender Ansätze und zeigt Möglichkeiten auf, die eine hierarchische Modellierung in derartigen Anwendungen auf praktikable Weise erlauben.

1 Einführung

Der zunehmende Einsatz von 3D/VR-Technologien in eCommerce-Anwendungen erfordert neben der Erfassung und Darstellung geometrischer und visueller Eigenschaften zu präsentierender Produkte auch die entsprechende Repräsentation nichtgeometrischer Eigenschaften wie Klassifizierung und Typisierung, Interaktivität und Produktlogiken. Eine qualitativ und quantitativ hinreichende Erfassung und konsequente Präsentation/Berücksichtigung dieser nichtgeometrischen Merkmale ermöglicht einen qualitativen Sprung von geometrischen Applikationen (Zeichen- und CAD-Programme), die ursprünglich zwei-dimensional arbeiteten, heute jedoch meist drei-dimensional orientiert sind, hin zu semantischen Applikationen, in denen die grafische Darstellung (zwei- oder dreidimensional) von sekundärer Bedeutung ist.

Die Vorteile einer semantischen Applikation bestehen zum einen in der Fähigkeit, die Komplexität und die Beziehungen realer Produkte abzubilden. Diese Wesensmerkmale ergeben sich aus:

- einer typischerweise hierarchischen Struktur,
- einer Menge nichtstrukturverändernder interaktiver Freiheitsgrade zumeist translatorischer oder rotatorischer Art,
- einer Anzahl von expliziten und impliziten Regeln, die beispielsweise die Aggregation zusätzlicher Komponenten oder die Zuweisung von Materialien definieren.

Unter der Annahme, daß eine gegebene Menge von Produkten vollständig und fehlerfrei bezüglich ihrer Semantik und Wechselbeziehungen erfaßt wurde, kann eine hunderprozentige Korrektheit von Planungen und Konfigurationen garantiert werden, die unter Anwendung dieses semantischen Datenmodells auf direktmanipulative Weise in einem 3D/VR-System erstellt worden. Dies ist ein essentielles Kriterium für die Akzeptanz und weitere Verbereitung von 3D/VR-Applikationen im Bereich des eCommerce.

Ein weiterer Aspekt der Anwendung o.g. semantischer Informationen betrifft die Intuitivität und Ergonomie der direktmanipulativen Benutzerschnittstellen im 3D/VR-System. Insbesondere die Aggregation, die ein wesentlicher Aspekt semantischer Informationen ist, erhöht bei

entsprechender Berücksichtigung Intuitivität und Ergonomie, da entsprechende Strukturen inhärent hierarchisch sind und nicht durch explizite Gruppierungsinteraktionen zusammengefaßt werden müssen.

Nachteilig ist jedoch, daß die Erfassung semantischer Informationen quantitativ einen Mehraufwand bedeutet und qualitativ neue Ansprüche stellt.

Im folgenden werden diese Aspekte im Detail diskutiert, wobei die existierenden Ansätze und ein Konzept, das die hierarchische Modellierung priorisiert, besondere Schwerpunkte darstellen. Zunächst sollen jedoch kurz die Merkmale der semantischen Modellierung, zumindest aus Sicht der Produktmodellierung im Rahmen des eCommerce, vorgestellt werden.

2 Semantische Modellierung

Die semantische Modellierung [1] basiert auf Konzepten der Softwaretechnik, insbesondere denen der strukturierten und objektorientierten Analyse und Synthese, und widmet sich der möglichst vollständigen Beschreibung von Objekten/Produkten aus der realen Welt. Dabei kommt den nachfolgenden Konzepten eine besondere Bedeutung zu.

Die ***Typisierung*** und/oder ***Klassifizierung*** (in der Praxis teilweise orthogonale Konzepte) beschreibt die Zugehörigkeit von Entitäten bzw. Objekten zu einer Menge von Typen und Klassifikationen. Die Begriffe Typ und Klassifikation werden teilweise in der realen Welt verwendet, wenn auch häufig nicht in der Konsequenz, wie sie für die Softwaretechnik typisch ist. Ein Typ ist außerhalb der Softwaretechnik üblicherweise eine schematische Kurzbezeichnung, die in einem gewissen Kontext (z.B. Hersteller- oder branchenspezifisch) eindeutig ist. Der Begriff des abstrakten Typs existiert durchaus in der Praxis, wird aber in der Regel lediglich zur Zusammenfassung einer Menge von Typen durch entsprechende Parametrisierung verwendet.

Die ***Generalisierung***/***Spezialisierung*** ist eine binäre Relation zwischen Typen, die eine Abwärtskompatibilität in Richtung Spezialisierung definiert. Diese bezieht sich zunächst auf die syntaktische Kompatibilität, d.h. der spezialisierte Typ definiert zumindest das Interface des generalisierten Typs, kann aber auch durch entsprechende Mittel (Typinvarianten, Vor- und Nachbedingungen, Rules) auf eine Verhaltenskompatibilität ausgedehnt werden. In letzterem Fall kann der spezialisierte Typ Vorgaben durch den generalisierten Typ bezüglich dem Verhalten nicht umgehen. Das Konzept der Generalisierung/Spezialisierung wird häufig in Kombination mit dem Konzept der (Implementierungs-)***Vererbung*** angewendet. Die wiederholte Anwendung der Generalisierung/Spezialisierung unter der Einschränkung, daß jeder Typ höchstens eine direkte Generalisierung besitzt, ergibt eine strikte Hierarchie von Typen.

Die ***Aggregation*** ist eine binäre Relation zwischen Typen, die das Enthalten sein von Instanzen eines Typs in Instanzen eines anderen Typs beschreibt. Die wiederholte Anwendung der Aggregation ergibt eine strikte Hierarchie von Instanzen.

Der Begriff ***Verhalten*** beschreibt in diesem Kontext die Art und Weise, in der ein Objekt auf eine Anwenderaktion oder ein automatisches Ereignis reagiert. Derartige Anwenderaktionen, die im Bereich der interaktiven Konfiguration von Produkten wesentlich sind, sind:

- die Erzeugung oder das Entfernen eines Objektes, ggf. im Kontext eines übergeordneten Objektes,
- die interaktive Selektion/Deselektion eines Objektes,
- die Bewegung eines Objektes und
- die Modifikation eines Objektes durch das Setzen von Eigenschaften.

Beispiele für Ereignisse, die sich nicht direkt durch eine Anwenderaktion ergeben, sind Timer-Ereignisse und sensorische Ereignisse. Letztere signalisieren einem Objekt die räumliche Annäherung eines anderen Objektes. Die Implementierung von Verhalten erfolgt üblicherweise durch prozedurale Konstrukte, entweder im Geltungsbereich eines Typs oder außerhalb davon.

3 Hierarchische Modellierung

In diesem Abschnitt sollen Ansätze zur hierarchischen Modellierung [1] beschrieben werden. Hierbei liegt der Schwerpunkt allerdings auf der Modellierung der Aggregations-Relation, da die Modellierung der Relation Generalisierung/Spezialisierung keine schwerwiegenden Probleme nach sich zieht.

3.1 Unstrukturierte Modellierung

Ein Modellierungsansatz soll als unstrukturiert klassifiziert werden, wenn es keine expliziten Relationen zwischen Objekten gibt. Diese Art der Modellierung wird in einfachen Zeichenprogrammen und CAD-Systemen angewendet. Hierbei sind alle Objekte in einem gemeinsamen Koordinatensystem definiert. Die Vorteile aus Sicht eines Anwenders sind wie folgt:

- Relationen zwischen Objekten existieren höchstens scheinbar. Folglich können Benutzerinteraktionen, z.B. das Erzeugen, Manipulieren und Zerstören von Objekten, völlig zwangfrei erfolgen. Dies gilt insbesondere für die scheinbare Komposition zusammengesetzter Objekte, die lediglich durch eine gewisse räumliche Kohärenz ausgedrückt ist. Allgemeine Mechanismen zur Beeinflussung einer unstrukturierten Modellierung sind beispielsweise, Rasterisierung, Snap-Positionierung, Kollisionserkennung und -vermeidung, und Simulation von Gravitation.
- (Temporäre) Gruppierungsoperationen können auf einfache Weise ohne zusätzlichen Aufwand erfolgen. Da keine inhärente Strukturierung existiert, können Gruppen beliebig definiert und wieder aufgelöst werden. Die Manipulation einer Gruppe, z.B. eine Verschiebung, ist einfach eine uniforme Anwendung der Manipulation auf die Gruppenelemente.

Ein genereller Nachteil des unstrukturierten Ansatzes besteht darin, daß weder Struktur noch Semantik in das Modell integriert werden können. Die Anwendung dieses Ansatzes sollte sich daher auf triviale Zeichen- und CAD-Programme beschränken. Dennoch fand und findet dieser Ansatz eine breite Verbreitung in Applikationsdomänen, die durch ein hohes Maß an Strukturierung und Semantik gekennzeichnet sind.
Eine Erweiterung dieses Ansatzes könnte die Definition persistenter Gruppen erlauben, deren Elemente elementare Objekte und/oder Gruppen sind.

3.2 Hierarchische Modellierung

Ein hierarchisches Modell ist gegeben, wenn die Objekte des Modells in einem gerichteten, azyklisch organisierten Graphen (DAG) organisiert. Zwischen zwei Objekten *Op* und *Oc* existiert eine Vater-Kind-Beziehung, wenn *Oc* ein Nachfolger von *Op* im DAG ist. Diese Beziehung kann wie folgt charakterisiert werden:

- Ein Objekt *Op* kann eine zeitveränderliche Menge von direkten Kindobjekten Oc_1 bis Oc_n haben.
- Ein Objekt *Oc* kann eine Menge von direkten Vorfahren Op_1 bis Op_m haben. Falls für das gesamte Modell m kleiner oder gleich 1 gilt, so handelt es sich um ein strikt hierarchisches Modell, ansonsten um einen DAG. Die direkten Vorfahren können dem Objekt bekannt sein, oder auch nicht, In einem strikt hierarchischen Modell ist der direkte Vorfahr meistens bekannt. Die Menge der direkten Vorfahren kann nach der Erzeugung eines Objektes geändert werden (re-parenting), oder auch nicht. In einem strikt hierarchischen Modell ist Re-parenting eine explizite Operation, die nicht allgemein zur Verfügung steht. Im Falle eines DAG ist die dynamische Zuweisung von Vaterobjekten eine Standardoperation, die zu minimalen Laufzeitkosten ausgeführt werden kann.
- Die Beziehung zwischen *Op* und *Oc* kann eine gewisse Semantik beinhalten, z.B., das Löschen von *Op* impliziert das Löschen von *Oc*, oder, *Oc* erbt Properties, die an *Op* zugewiesen wurden.
- Der Einfluß, den ein Objekt *O* auf das Modell hat, z.B. durch die Manipulation eines globalen Kontextes kann auf den durch *O* gegebenen Unterbaum begrenzt sein, oder auch nicht. Im letzteren Fall können ist eine Beeinflussung von Geschwistern und Vorfahren von *O* durch *O* möglich. Einige Ansätze erlauben standardmäßig einen unbegrenzten Einfluß, ermöglichen aber eine explizite Begrenzung des Einflusses.
- Kindobjekte können interaktiv zugegriffen werden, oder auch nicht. Der Zugriff auf Kindobjekte ist von Bedeutung in nichttrivialen hierarchischen Modellen und wird daher üblicherweise zur Verfügung gestellt. Dies schließt eine (dynamische) Steuerung der Zugreifbarkeit mit ein.

Im folgenden soll die strikte hierarchische Modellierung diskutiert werden. Dies schließt prinzipiell den DAG mit ein, da jeder DAG konzeptionell auf eine strikte Hierarchie abgebildet werden kann. In der Tat erfolgt diese Abbildung zumindest zur Laufzeit, teilweise aber auch über sekundäre Datenstrukturen, wie sie zur Laufzeitoptimierung von DAGs eingesetzt werden.

Das allgemeine Problem eines strikt hierarchischen Modells besteht darin, daß es eine mehr oder weniger statische Topologie beschreibt, in welcher jedes Objekt seine feste Position hat. Die dominierende Relation ist, wie oben geschrieben, die Aggregationsrelation. Diese Relation, die häufig auf natürliche Weise auf die reale Welt bzw. Modelle von dieser abgebildet werden kann, ist unzureichend, falls die topologische Zuordnung eines Objekts mehrdeutig oder hochgradig temporär ist.

Das erste Problem (Mehrdeutigkeit) kann beseitigt werden, in dem mehrdeutige bzw. alternative Abstraktionsstufen entfernt werden und nur temporär existieren. Beispielsweise könnten in einem Modell von Rubiks Würfel für jeden atomaren Würfel drei Ebenen als direkte Vorfahren identifiziert werden. Statt dessen wird die tatsächliche Ebene temporär bestimmt, z.B. bei Bewegung eines Teilwürfels, und auf diese die entsprechenden Operationen angewendet.

Das zweite Problem (temporäre Zuordnung) ergibt sich, wenn aggregative Relationen zur Simulation physikalischer Kräfte verwendet werden sollen. Man stelle sich einen Tisch vor, auf dem ein Buch liegt. Die Verwendung der Aggregation mit den oben beschriebenen Merkmalen würde dazu führen, daß sich das Buch bei einer Bewegung des Tisches entsprechend mitbewegt. Diese Abhängigkeit müßte natürlich beseitigt werden, wenn das Buch vom Tisch fortbewegt wird. Gegebenenfalls müßte eine Abhängigkeit zu einem anderen Objekt aufgebaut werden. Eine

derartige Operation muß allerdings explizit durchgeführt werden und wird von heutigen Grafiksystemen nicht unterstützt.

Die einfache hierarchische Modellierung erlaubt die willkürliche Gruppierung von Objekten, z.B. durch entsprechende Interaktionen. Auf diese Weise können unsinnige Gruppen definiert werden, z.B. solche, die Objekte derselben Abstraktionsstufe auf unterschiedlichen topologischen Ebenen verwenden.

Des weiteren ist die Definition von temporären Gruppen, orthogonal zum hierarchischen Modell, nicht trivial, da die entsprechenden hierarchischen Relationen berücksichtigt werden müssen. Eine diesbezüglich sinnvolle Beschränkung würde darin bestehen, nur die Gruppierung von solchen Objekten zuzulassen, die einen gemeinsamen Vater haben.

3.3 Kontrollierte Hierarchische Modellierung

Eine kontrollierte hierarchische Modellierung soll gegeben sein, wenn die Menge der Objekte, für die eine Vater-Kind-Beziehung zu einem gegebenen Objekt *Op* aufgestellt werden kann, begrenzt ist, wie folgende Beispiele illustrieren:

- Lediglich Instanzen einer bestimmten Menge von Typen können Kinder von *Op* sein.
- Nur solche Objekte können Kinder von *Op* werden, die bestimmte geometrische Beziehungen erfüllen, z.B. sie überschreiten nicht ein gegebenes Begrenzungsvolumen, sie kollidieren weder mit Op selbst noch mit ihren (zukünftigen) Geschwistern.

Die kontrollierte hierarchische Modellierung stellt eine zusätzliche Schicht zwischen der interaktiven Applikation und der vorher definierten einfachen hierarchischen Modellierung dar. Diese Schicht kann verwendet werden, um den Realismus interaktiver Modellierung zu erhöhen, da falsche oder unsinnige Konfigurationen vermieden werden, z.B. die Durchdringung von Objekten. Die direkte Integration einer solchen Schicht in das hierarchische Modell erfordert Konzepte wie Callback-Prozeduren oder Regeln. Eine explizite Definition dieser Regeln für jeden Typ erscheint zunächst aufwendig, kann aber durch geeignete Mittel wie Abstraktion und Vererbung weitgehend redundanzfrei erfolgen.

Ein wesentlicher Nachteil kontrollierter hierarchischer Modellierung ist durch ihren expliziten und statischen Charakter gegeben. Sofern die Regeln über Typen definiert sind, ist es normalerweise unmöglich, sie zur Lebenszeit von Objekten zu ändern.

4 Objektorientierte Hierarchische Modellierung

In diesem Abschnitt soll ein Ansatz zur hierarchischen Modellierung beschrieben werden, der objektorientierten Konzepte auf homogene Weise integriert. Hierbei sind insbesondere folgende Konzepte relevant:

- Die Vererbung stellt eine erweiterte Semantik für die Vater-Kind-Beziehung dar. Kinder erben Merkmale des Vaterobjektes, z.B. die räumliche Modellierung.
- Die Einkapselung bestimmt, ob und wie Objektmerkmale, z.B. Kinder, außerhalb des Objektes zugreifbar sind.
- Polymorphie ist gegeben, wenn verschiedene Objekte in uniformer Weise behandelt werden können.

Eine offensichtliche Kombinationen der hierarchischen Modellierung mit objektorientierten Konzepten besteht darin, die Knoten eines hierarchischen Modells als Objekte zu betrachten. Folglich ist es eine inhärente Eigenschaft eines jeden Objekts, eine Menge von Kindobjekten zu verwalten. Entsprechend dem objektorientierten Paradigma sollen Kindobjekte abgeschirmte Komponenten eines Objektes sein. Die eindeutige Zuordnung eines Kindobjektes zu seinem Vaterobjekt resultiert in einem hierarchischen Namensraum.

Der direkte Zugriff auf Kindobjekte widerspricht dem objektorientierten Paradigma, da die interne Struktur unsichtbar sein sollte. Andererseits ist der Zugriff auf verschiedene Ebenen eines hierarchischen Modells zwingend erforderlich. Eine pragmatische Auflösung dieses Konflikts, auch hinsichtlich dem mentalen Modell der interaktiven Computergrafik [2], besteht darin, den Zugriff auf Kindobjekte durch das Vaterobjekt zu kontrollieren, d.h., das Erzeugen, Manipulieren und Entfernen von Kindobjekten könnte vom Vaterobjekt erlaubt, verhindert oder in irgendeiner Weise beschränkt werden.

Das Konzept der Attribute, wie es aus der strukturierten und objektorientierten Programmierung bekannt ist, soll wie folgt in das objektorientierte hierarchische Modell übernommen werden:

- Ein Attribut (Wert oder Referenz) ist eine private Komponente des entsprechenden Objektes.
- Der direkte Einfluß eines Attributes soll auf das zugehörige Objekt beschränkt bleiben, was allerdings die Kinderobjekte mit einschließt, da sie ebenfalls zum Objekt gehören. Eine Vorbild für diese Semantik findet sich z.B. in ISO PHIGS [3].
- Der Typ, das Format und die Semantik von Attributen sind prinzipiell nicht beschränkt. Beispiele für die Semantik von Attributen ist die Vererbung eines Attributwertes an die Kinderobjekte oder die Beschränkung des Wertebereiches eines Attributes.
- In Klassen-basierten Systemen sind Attribute statisch, da ihre Anzahl und Spezifikation nicht zur Lebenszeit der Objekte geändert werden kann. Dies ist ein offensichtlicher Unterschied zum dynamischen Wesen der Kinderobjekte.

In Entsprechung zum objektorientierten Paradigma soll ein direkter Zugriff auf Attribute nicht möglich sein. Statt dessen soll der Zugriff über Setter/Getter-Methoden erfolgen, die eine zusätzlich Semantik besitzen können, z.B.:

- Verwaltung multipler Referenzen zur Optimierung der Speichereffizienz,
- Delegation, z.B. bei Einkapselung eines entsprechenden Objektes,
- Abwärtstraversierung für die Vererbung von Eigenschaften,
- Caching zur Optimierung der Laufzeiteffizienz,
- Kompilierung einer alternativen internen Repräsentation ebenfalls zur Verbesserung der Laufzeiteigenschaften,
- Beziehungen und Constraints, die lokal oder global definiert sind und
- Re-Traversierung, z.B. zur Erhöhung der Speichereffizienz.

Im Bereich der Constraint-basierten Applikationen stellt sich ein Zugriff über o.g. Setter/Getter-Methoden problematisch dar, da globale Constraint-Solver direkt mit Attributwerten operieren. Ein Zugriff über Methoden führt neben der Verringerung der Laufzeiteffizienz und der Erhöhung des Speicherbedarfs aufgrund der redundanten Datenhaltung auch zu dem konzeptionellen Problem, daß aufgrund der internen Semantik (z.B. lokale

Constraints) eine globale Lösung nicht akzeptiert werden kann. In diesem Fall sind die Constraints entweder nicht erfüllt oder eine Kommunikation zwischen Constraint-Solver und Objekt wird erforderlich, deren Zeitkomplexität allerdings exponentiell für diskrete Domänen oder höher für kontinuierliche Domänen ist [4]. Ansätze zur Lösung des Problems werden z.B. in [5] und [6] diskutiert.

Ein weiterer Aspekt im Kontext eines objektorientierten hierarchischen Modells ist die Wahrung des Lokalitätsprinzips [2]. Ein Applikationsprogrammierer erwartet intuitiv eine gewisse Integrität von Entitäten, die er auf visuelle Weise identifiziert. Diese Integrität ist mehr oder weniger gegeben, wenn das Prinzip der Lokalität gewährleistet ist. In den allermeisten Ansätzen im Bereich der 3D-Modellierung ist dieses jedoch verletzt, da die verschiedenen Komponenten, die zusammen eine Entität im Sinne des mentalen Modells repräsentieren, über das hierarchische Modell hinweg verteilt sind. Eine derartige Verletzung führt zu Problemen bei der interaktiven Modifikation von hierarchischen Modellen, da keine eindeutige Interpretation der Modifikation möglich ist. Diese Mehrdeutigkeit begründet sich nicht zuletzt in einer unscharfen Verwendung der Begriffe Klasse und Instanz. In einem objektorientierten Modell - wie oben beschrieben - existiert diese Mehrdeutigkeit nicht, da derartige Komponenten stets eindeutig zu einem Objekt gehören. Die scheinbare Ineffizienz eines solchen Ansatzes kann durch Setter/Getter-Methoden beseitigt werden, die Komponenten mehrfach referenzieren, ohne dabei das Lokalitätsprinzip aus Sicht des Anwendungsprogrammierers zu verletzen [7].

5 Explizite Modellierung

Der Begriff der expliziten Modellierung stellt eine alternative Sichtweise zur vorher beschriebenen objektorientierten hierarchischen Modellierung dar. Ein Modell *M* soll als explizit bezeichnet werden, wenn jeder Typ *T* in *M* bezüglich der Möglichkeiten von *M* vollständig beschrieben ist. In einem Produktmodellierungssystem, das auf einem hierarchischen Ansatz basiert, schließt dies die folgenden Informationen ein:

- eine Menge von aggregativen Beziehungen, die beschreiben, wie, d.h. an welchen relativen Positionen und unter welchen Bedingungen, Instanzen von *T* in anderen Objekten aggregiert werden können,
- eine Menge aggregativer Beziehungen, die beschreiben, welche anderen Objekte in Instanzen von *T* aggregiert werden können,
- interaktive Möglichkeiten, z.B. die Bewegbarkeit von Instanzen oder ihrer Komponenten, sowie
- Manipulationen, die über dem Objekt ausgeführt werden können.

Ein explizites Modell bildet die reale Welt in einem hinreichenden Maß ab (zumindest in Bezug auf eine bestimmte Applikationsdomäne) und erlaubt somit eine authentische Modellierung realer Vorbilder durch ihre virtuellen Modelle. Unter der Annahme, daß ein vollständiges und fehlerfreies explizites Modell erfaßt wurde, wurde die Grundvoraussetzung für die Realisierung robuster und korrekter Applikationen geschaffen, die aufgrund der Tatsache, daß sie das jeweilige domänenspezifische Wissen vollständig enthalten, auch von Laien korrekt bedient werden können. Dies entspricht der zunehmenden Orientierung des eCommerce auf Direktvertrieb bzw. Endkunden.

Ein explizites Modell stellt gegenüber herkömmlichen Ansätzen einen qualitativen Sprung dar, sowohl was die Mächtigkeit betrifft, als auch bezüglich des Erfassungsaufwandes. Letzteres scheint aber durch entsprechende Mechanismen wie Abstraktion und Vererbung, und

Modellierungswerkzeuge beherrschbar, so daß die Anwendung expliziter Modelle letztlich der erfolgversprechende Weg für die Zukunft zu sein scheint.

6 Die Modellierungssprache AFML

Die semantische 3D-Modellierungssprache AFML [8][1] setzt die oben eingeführten Paradigmas objektorientierte hierarchische Modellierung und explizite Modellierung konsequent um. Unter Anwendung eines integrierten Ansatzes werden die verschiedenen semantischen Aspekte mit geometrischen und visuellen Eigenschaften direkt zusammen, wodurch sich eine kompakte Beschreibungsform ergibt, die Redundanzen vermeidet. Die Beschreibung der sichtbaren Merkmale ist dabei von sekundärer Bedeutung. Dies zeigt sich zum einen darin, daß sich physische Objekte fast ausschließlich in den Blättern der hierarchischen Modelle befinden. Weiterhin ist die direkte geometrische Modellierung nur minimal im Vergleich zu etablierten geometrischen Modellieren ausgeprägt. Statt dessen unterstützt AFML die direkte und somit effiziente Integration externer geometrischer Modelle, was insbesondere hinsichtlich der Migration existierender Datenbestände in das AFML-Format vorteilhaft ist.

AFML, in Verbindung mit einer kommerziellen Laufzeitumgebung, befindet sich momentan im flächendeckenden Einsatz in der deutschen Möbelindustrie. Diese Anwendungsdomäne ist von einem hohen Maß an Varianten und strukturellen Relationen geprägt und eignet sich daher besonders für die Anwendung eines expliziten hierarchischen Modellierungsansatzes. Zu diesem Zeitpunkt sind bereits eine Reihe AFML-basierter Applikationen verfügbar, die den Nachweis erbracht haben, daß ein der AFML zugrundeliegende Ansatz in der Tat Endkunden befähigt, nichttriviale Planungen, wie z.B. im Küchenbereich, eigenständig und fehlerfrei erstellen zu können.

7 Ausblick

Die aktuellen Anstrengungen im AFML-Kontext zielen auf

- die Definition einer hierarchischen Menge von Bibliotheken, die, durch entsprechende abstrakte Typen, die Interoperabilität der semantischen Daten über Hersteller und Domänen hinweg erlauben,
- die Realisierung einer Modellierumgebung, die die direktmanipulative Erstellung semantischer Daten ermöglicht, und
- die Definition von genormten Schnittstellen, um existierende Daten, die z.B. in relationalen Datenbanken vorliegen, direkt integrieren zu können.

Literatur

[1] E. Beier. *AFML - A Conceptual Modeling Language*. Manuskript, 1998/1999.

[2] P. Wißkirchen. *Object-Oriented Graphics - From GKS and PHIGS to Object-Oriented Systems.* Springer-Verlag 1990.

[3] ISO. Information Processing Systems - Computer Graphics - Programmer's Hierarchical Interactive Graphics System (PHIGS). ISO/IEC 9592:1989, International Organisation of Standardization, 1989, Parts 1 - 3.

[4] R. Veltkamp und R. Kelleners. *Information Hiding and the Complexity of Constraint Satisfaction*. In R. Veltkamp and E. Blake, *Programming Paradigms in Graphics*. Springer-Verlag, 1996.

[5] Q. Hoole und E. Blake. OOCS - *Constraints in an Object-Oriented Environment*. In 4th *EuroGraphics Workshop on Object-Oriented Graphics*, 1994. Sintra, Portugal.
[6] B. N. Freeman-Benson und A. Borning. *Integrating Constraints with an Object-Oriented Language*. In ECOOP'92. Springer-Verlag, 1992.
[7] E. Beier. *Issues on Hierarchical Graphical Scenes*. In R. Veltkamp and E. Blake, *Programming Paradigms in Graphics*. Springer-Verlag, 1996.
[8] E. Beier. *AFML - An Object-Oriented 3D Metafile Format. In P. Wißkirchen, Object-Oriented and Mixed Programming Paradigms in Graphics*, Budapest, 1997.