Konzeption einer 3D-CAD-Datenstruktur für eine benutzergerechte Handhabung von sehr großen, komplexen Baugruppen

Vom Fachbereich Maschinenwesen der
Universität Duisburg-Essen, Standort Essen,
zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs
genehmigte Dissertation

Vorgelegt von
Dipl.-Math. Thomas Schneider
Kassel

Referent: PD Dr.-Ing. F. Lobeck
Korreferent: Univ.-Prof. Dr.-Ing. C. Weber

Datum der mündlichen Prüfung: 29.06.2006

Vorwort

Die vorliegende Arbeit ist begleitend zu meiner Tätigkeit in der Forschungs- und Entwicklungsabteilung der CoCreate Software GmbH & Co. KG, Sindelfingen, entstanden. Die akademische Betreuung hat das Institut für Ingenieurinformatik am Fachbereich Maschinenwesen der Universität Duisburg-Essen, Standort Essen, übernommen.

Herrn Priv.-Doz. Dr.-Ing. F. Lobeck, Institut für Ingenieurinformatik der Universität Duisburg-Essen, Standort Essen, danke ich für die fachliche Unterstützung und die Übernahme des Erstgutachtens.

Herrn Univ.-Prof. Dr.-Ing. C. Weber, Lehrstuhl für Konstruktionstechnik / CAD der Universität des Saarlandes, danke ich für das meiner Arbeit entgegengebrachte Interesse sowie die Übernahme des Korreferats.

Herrn Univ.-Prof. Dr.-Ing. H. J. Stracke, Institut für Ingenieurinformatik der Universität Duisburg-Essen, Standort Essen, danke ich für die wertvollen Anregungen, die zum Gelingen dieser Arbeit beigetragen haben.

Herrn Dr. rer. nat. R. Zeifang, Leiter Forschung und Entwicklung Deutschland, CoCreate Software GmbH & Co. KG, Sindelfingen, danke ich für die Unterstützung seitens des Unternehmens. Außerdem danke ich allen Mitarbeitern der Entwicklungsabteilung, die mit zur Umsetzung der Konzepte beigetragen und als Partner für unzählige Diskussionen zur Verfügung gestanden haben.

Herrn Dr.-Ing. U. Hamme, Geschäftsführer und Technischer Direktor, Liebherr-Werk Ehingen GmbH, danke ich für die Erlaubnis, CAD-Modelle aus dem Unternehmen im Rahmen dieser Arbeit untersuchen und die Ergebnisse veröffentlichen zu dürfen, sowie Herrn W. Knehr für seine Mithilfe bei der Beschaffung der Daten.

Meinen Eltern bin ich sehr dankbar für jede Form der Unterstützung und Motivation, die ich von ihnen erhalten habe, um diese Arbeit erfolgreich abschließen zu können.

Ganz besonderer Dank gilt meiner Frau Martina für ihren unschätzbar großen Beitrag zu meiner Unterstützung. Über einen langen Zeitraum hat sie unser Familienleben allein aufrecht erhalten und doch stets voll hinter meinem Vorhaben gestanden.

Böblingen, im April 2006 Thomas Schneider

Für Martina

Carla und Silke

Inhaltsverzeichnis

1 Einleitung

In Zeiten wachsender Globalisierung und immer kürzer werdender Innovationszyklen gewinnen effiziente Konstruktionsprozesse zunehmend an Bedeutung. Sie sind mitentscheidend für den qualitativen und wirtschaftlichen Wert eines Produkts. Mit der Evolution der zur Produktentwicklung eingesetzten Werkzeuge sind auch die Anforderungen weiter gestiegen. Während vor gut einem Jahrzehnt noch die Konstruktion von Einzelteilen und kleinen Baugruppen im Mittelpunkt stand, werden heute komplette, hochkomplexe Produkte virtuell entwickelt und analysiert, bevor die Anfertigung erster realer Prototypen erfolgt. Die Konstruktion von Fertigungsanlagen, Mobilkranen oder Tunnelbohrmaschinen ist ohne den Einsatz moderner CAD- und PDM-Systeme kaum noch denkbar.

Infolge dieses Fortschritts wird jedoch auch der Konstruktionsprozeß selbst immer komplexer. Die Produktentwicklung ist längst zu einer fachübergreifenden Aufgabe geworden, bei der eine immer dichtere Vernetzung von Spezialwissen verschiedener Bereiche stattfindet. Aufgrund dessen ist unter anderem eine zunehmende Integration von Simulations- und Berechnungswerkzeugen in die CAD-Systeme zu beobachten. Dies ermöglicht die frühe Erstellung ganzheitlicher, virtueller Prototypen, mit deren Hilfe Probleme rechtzeitig erkannt sowie Produktions- und Fertigungsprozesse vorab optimiert werden können. Im Ergebnis verringert sich schon bei der ersten Konstruktion das Risiko späterer kostenintensiver Änderungen.

Voraussetzung für die interdisziplinäre Entwicklung ist die Existenz eines gemeinsamen Datenmodells, das sowohl Konstruktionsdaten als auch technologische Produktdaten, beispielsweise Analysen und Berechnungen, umfaßt. Bezogen auf die Produktentwicklung als Ganzes stellt dieses Mastermodell einen wesentlichen Bestandteil des Produktmodells dar. Um seiner Rolle als zentraler Informationsträger

gerecht zu werden, muß das Modell – ebenso wie die Umgebung, in die es eingebettet ist – eine Reihe von Anforderungen wie Integrität und Assoziativität erfüllen. Da unter diesem Dach die Daten verschiedener Anwender und Applikationen zusammenkommen, ist ein gezielter und kontrollierter Zugriff auf einzelne Einheiten erforderlich. Ebenso ist den Abhängigkeiten der Daten voneinander Rechnung zu tragen.

Sowohl die steigende Größe und Komplexität der Baugruppen als auch die fortschreitende Integration von Produkt- und Prozeßdaten in die CAD-Modelle verursachen Datenmengen, die auch moderne High-End Rechner an die Grenze ihrer Leistungsfähigkeit bringen – oder darüber hinaus. Die rapide Weiterentwicklung der Hardware schafft zwar ständig verbesserte Möglichkeiten, größere Datenmengen in kürzerer Zeit zu verarbeiten, auf der anderen Seite wachsen die Anforderungen der Konstrukteure jedoch in mindestens dem gleichen Maße mit. Aus Sicht des Anwenders steht dem stetigen Wachstum der Anforderungen überdies lediglich eine diskrete Verbesserung der technischen Möglichkeiten gegenüber, da Hardware typischerweise nur im mehrjährigen Turnus ausgetauscht wird. Selbst wenn das Moore'sche Gesetz[1] weiterhin gilt, daß sich die Leistungsfähigkeit der Hardware alle zwei Jahre verdoppelt, wird dies allein die Ressourcenprobleme beim Umgang mit großen Baugruppen nicht lösen.

Zusammenfassend stellt sich die derzeitige Situation so dar, daß einerseits immer mehr Daten von Werkzeugen und Prozessen, die mit der Konstruktion in Verbindung stehen, in das CAD-Modell integriert werden. Dadurch wird eine hohe Assoziativität zum 3D-Modell erreicht und den Anforderungen an eine interdisziplinäre Entwicklung Rechnung getragen, jedoch auch das Datenvolumen erhöht. Dem steht andererseits die Forderung nach gezieltem und kontrolliertem Zugriff auf diese Daten gegenüber. Neben dem Schutz der Daten ist diese – insbesondere bei großen Baugruppen – hauptsächlich durch die Notwendigkeit begründet, jedem Anwender eine Modellsicht bereitzustellen, die den Erfordernissen seiner Aufgabe möglichst optimal entspricht. Damit wird die zu verarbeitende Datenmenge auf das Wesentliche reduziert, was in einigen Fällen die Voraussetzung dafür ist, diese Aufgabe überhaupt bearbeiten zu können.

---

[1] Geäußert von Gordon E. Moore in der Zeitschrift Electronics im Jahre 1965 (!)

Zielsetzung

Existierende Literatur mit Bezug zur Handhabung großer Baugruppen befaßt sich hauptsächlich mit der Datenverwaltung und Produktdatenmodellierung auf der PDM- / PLM-Seite[2] oder nimmt sich des Themas aus der Sicht von Anwendern an[3]. Der Schwerpunkt dieser Veröffentlichungen liegt auf der Beschreibung von Richtlinien und Modellierstrategien, die den Zweck verfolgen, Komplexität weitestgehend zu vermeiden. Knieps [Kni04] entwickelt beispielsweise in seiner Dissertation Konstruktionsrichtlinien zur Generierung komplexer Maschinenbaugruppen. Diese beschäftigen sich, soweit sie die Erstellung neuer Modelle betreffen, in der Hauptsache mit der Organisation von Konstruktionsabläufen mit dem Ziel, möglichst schlanke Modelle zu erhalten.

An dieser Stelle soll ein anderer Lösungsansatz entwickelt werden. Nicht Richtlinien, die der Konstrukteur befolgen muß, um komplexe Baugruppen erstellen zu können, stehen im Mittelpunkt, sondern vielmehr Verbesserungen am CAD-System selbst, die es dem Konstrukteur erlauben, die ihm gestellten Aufgaben einfacher und mit höherer Effizienz erfolgreich zu erfüllen.

Die vorliegende Arbeit verfolgt dabei das Ziel, eine Brücke zwischen der hohen Integration von technologischen Produktdaten einerseits und einem selektiven Zugriff darauf andererseits zu schlagen. Um dies zu erreichen wird ein Ansatz untersucht, ein CAD-Modell so zu strukturieren und auf eine Dateistruktur abzubilden, daß beiden Forderungen gleichermaßen entsprochen wird. In diesem Rahmen erfolgt der Entwurf von Konzepten und Datenstrukturen, die bei der internen Datenverwaltung eines 3D-CAD-Systems zum Tragen kommen.

Auf dieses Ergebnis aufbauend sollen weitere Konzepte ergänzt werden, die die entwickelte Technologie anwenden, um damit einen unmittelbaren Beitrag zur benutzergerechten Handhabung von großen Baugruppen zu leisten. Exemplarisch können dazu die Optimierung der Konstruktionsumgebung sowie die Reduzierung des Speicherbedarfs und der Ladezeiten genannt werden.

---

[2] Vgl. [Schi02] et al.

[3] Vgl. [Kni04], [RuUh00] et al.

Nicht im Fokus sind dagegen explizite Modellveränderungen, wie Vereinfachungen des geometrischen Modells, und Verbesserungen im peripheren Bereich, beispielsweise bei der externen Datenverwaltung in PDM-Systemen. Diese Aspekte bieten sicher auch die Gelegenheit zu einer Betrachtung im Zusammenhang mit großen Baugruppen. Dies würde jedoch den gegebenen Rahmen übersteigen und ist der weiteren Forschung vorbehalten.

Das Vorgehen zur Entwicklung der Konzepte und Datenstrukturen zur Handhabung großer Baugruppen im Rahmen dieser Arbeit soll im folgenden skizziert werden. Dabei wird auf die wesentlichen Inhalte der einzelnen Schritte jeweils kurz eingegangen.

Nach dieser Einleitung, in Kapitel 2, werden zunächst essentielle Grundlagen der CAD-Datenverwaltung und Technologien beschrieben, die eine fundierte Grundlage zur Einordnung der technischen Aspekte bei der Handhabung großer Baugruppen liefern. Der Schwerpunkt liegt dabei auf der Verwaltung und Gruppierung kleinster Informationseinheiten (sogenannter Entities), die für das weitere Vorgehen von zentraler Bedeutung sind.

Im dritten Kapitel werden Anforderungen an 3D-CAD-Systeme zusammengestellt, die sich aufgrund der Analyse von Arbeitsweisen im Konstruktionsbereich verschiedener Anwender ergeben haben. Die Daten wurden hauptsächlich durch die Befragung von Konstruktionsleitern, Konstrukteuren und Systembetreuern gewonnen, die bereits 3D-CAD-Systeme verwenden und mit der Konstruktion sehr großer und komplexer Baugruppen, wie zum Beispiel Mobilkranen oder Verpackungsmaschinen, betraut sind. Ergänzt werden die so gewonnenen Erkenntnisse durch in der Literatur beschriebene Verfahrensweisen, wie beispielsweise in [PaBe03] oder [VDI93].

Im darauffolgenden Kapitel werden die marktbestimmenden CAD-Systeme hinsichtlich ihrer bereits vorhandenen Fähigkeiten zur Unterstützung des Umgangs mit großen Baugruppen untersucht. Ziel dieses Abschnitts ist es, den Status quo der verfügbaren Technik zu ermitteln. Dabei werden wesentliche Konzepte und Technologien jeweils kurz beschrieben und in der Folge den in Kapitel 3 formulierten Anforderungen gegenübergestellt.

Aus dem Resultat, das der Vergleich der Anforderungen an die CAD-Systeme und der vorhandenen Funktionalität liefert, werden schließlich in Kapitel 5 Konzepte und Datenstrukturen entwickelt, die die bestehende Lücke schließen und ein effizienteres und benutzerfreundlicheres Arbeiten mit großen Baugruppen ermöglichen. Der Schwerpunkt wird dabei auf einer Separierung der CAD-Daten in Einheiten liegen, die eine flexible und anwendungsgerechte Verwaltung der Daten ermöglicht. Gleichzeitig werden Methoden beschrieben, die auf diesen Datenstrukturen aufbauen.

Kapitel 6 ist der exemplarischen Realisierung des Konzepts der Datenseparierung gewidmet. Darin wird die softwaretechnische Umsetzung grundlegender Teile dieses Konzeptes innerhalb eines 3D-CAD-Systems beschrieben. An einem konkreten Beispiel werden Systemparameter wie der Ressourcenbedarf nach bisherigem Entwicklungsstand eines untersuchten CAD-Systems mit denen verglichen, die durch Umsetzung der Konzepte ermöglicht wurden.

Abgerundet wird diese Untersuchung zur Konzeption einer Datenstruktur zur Handhabung großer Baugruppen durch die Zusammenfassung der erzielten Ergebnisse und einen Ausblick auf weitere Forschungsmöglichkeiten in diesem Bereich mit dem abschließenden Kapitel 7.

2 Datenverwaltung eines 3D-CAD-Systems

Die Darstellung eines 3D-Modells basiert auf einer hochkomplexen Datenstruktur, die aus einer sehr großen Anzahl von Objekten besteht. Dabei erfordert der Modellierungsprozeß, daß diese Datenstruktur äußerst flexibel, schnell, sicher und auf generische Art und Weise manipuliert werden kann. Damit die Daten auch über das Ende einer Arbeitssitzung hinaus für weitere Verarbeitungsschritte zur Verfügung stehen, müssen diese dauerhaft auf Dateisysteme oder in Datenbanken gespeichert und später wieder geladen werden können. Außerdem werden vielseitige Schnittstellen benötigt, die es Applikationen erlauben, auf diese Datenstruktur zuzugreifen und sie eigenständig zu verändern oder um eigene Objekte zu erweitern.

Im folgenden soll auf wesentliche Grundzüge einer solchen Datenstruktur näher eingegangen werden. Exemplarisch wird ein Objektnetzwerk aus sogenannten *Entities* betrachtet, welches das Rückgrat des 3D-CAD-Systems OneSpace Designer Modeling von CoCreate darstellt. Kern dieser Beschreibung sind architektonische Aspekte der Basisdatenobjekte und -strukturen, die für die weiteren Untersuchungen im Rahmen dieser Arbeit von Relevanz sind. Darüber hinaus sind allgemeine Grundlagen zur CAD-Technik bereits an diversen Stellen beschrieben worden und können unter anderem in [Enc88], [Grä89], [SpKr84], [Str00] nachgelesen werden.

Ergänzt wird dieses Kapitel durch Erläuterungen zu ausgewählten Technologien zur Verarbeitung der CAD-Daten, die für die Umsetzung der zu entwickelnden Konzepte grundlegend sind.

2.1 Entities

Zur Darstellung eines Knotens in einem komplexen Netzwerk, wie dem eines 3D-CAD-Modells, wird eine Basisdatenstruktur – ein sogenanntes Entity – verwendet (vgl. [BrKu95]). Charakteristisch für ein solches Entity, bzw. einen davon abgeleiteten Repräsentanten, ist die Existenz von eigenen Daten, Beziehungen zu benachbarten Entities und eine lange, in der Regel über die Arbeitssitzung hinausgehende Lebenszeit. Die Daten sind typischerweise gekapselt und nur über Zugriffsfunktionen erreichbar. Möglichkeiten zur Verwaltung der Daten sowie zur internen Konsistenzsicherung sind gewöhnlich vorhanden. Damit entspricht diese Datenstruktur im wesentlichen den in der Literatur[4] an Entities gestellten Anforderungen.

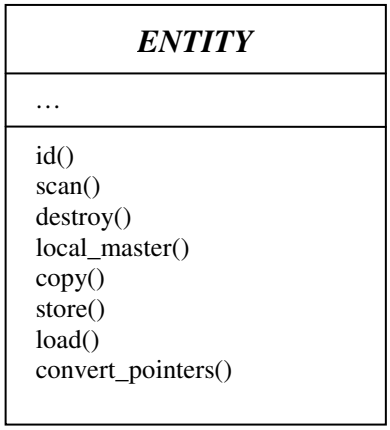


Abbildung 2.1: Die Klasse ENTITY

Weiterhin stellt jedes Entity Grundfunktionen zum Aufbau und zur Verwaltung einer vernetzten Struktur zur Verfügung. Dazu gehört hauptsächlich Funktionalität zur Transaktionsverwaltung (vulgo: „UNDO-System“), für die Persistenz, das Kopieren bzw. das Löschen sowie das Traversieren des Netzes. Im betrachteten Beispiel werden die Datenstrukturen unter Verwendung der Programmiersprache C++ implementiert, so daß diese Funktionalität in Form von virtuellen Funktionen bereitgestellt werden kann. Dadurch ist eine Schnittstelle definiert, die die generische Verarbeitung aller Daten in einem Entity-Netzwerk ermöglicht. Die **Abbildung 2.1** zeigt den Aufbau der Klasse ENTITY in UML-Notation.

[4] Vgl. zum Beispiel [HrRu02]

Alle Applikationen, einschließlich des Modellierkerns, leiten die von ihnen benötigten Klassen von dieser Basis ab. Sie stellen durch Überladen der Schnittstellenfunktionen sicher, daß ihre jeweiligen Daten entsprechende Berücksichtigung finden. In der aktuellen Version von OneSpace Designer Modeling sind auf diese Weise weit über 1.000 verschiedene Entity-Klassen definiert. Die Bandbreite reicht von einfachen Klassen wie Attributen über Geometrie- und Topologieelemente bis hin zu Klassen, die zur Beschreibung komplexer Objekte wie Teile oder Baugruppen benötigt werden. Für häufige Typen von Anwendungsfällen steht eine Reihe von Standardklassen zur Verfügung, die eine für den jeweiligen Zweck erweiterte Grundfunktionalität bereitstellen.

2.1.1 Entity-Netzwerke

Die grundlegende Struktur des gesamten 3D-Modells in OneSpace Designer Modeling besteht aus *einem* Entity-Netzwerk, das von Anwendungsprogrammen und Modellierkern gemeinsam benutzt wird. Die Struktur des Netzwerkes entsteht dadurch, daß jedes Entity Beziehungen zu einem Nachbar-Entity oder mehreren Nachbar-Entities aufbaut und pflegt.

Der fundamentale Design-Grundsatz zur Entwicklung einer Entity-Klasse und damit in der Folge zum Aufbau des Entity-Netzwerks ist, Wissen so lokal wie möglich zu halten. Entsprechend kennt jedes Objekt nur seine direkten – oder zumindest allernächsten – Nachbarn. Bei Veränderungen an einem Entity kann es eigenständig geeignete Reaktionen bei seinen Nachbarn auslösen. Das Wissen des Systems steckt somit in der Datenstruktur selbst, es gibt keine zentrale Stelle, die alle Zusammenhänge verwaltet. Dadurch ist eine äußerst hohe Stabilität des Netzwerkes gewährleistet.

Um Informationen über das gesamte Entity-Netz oder Ausschnitte davon zu erhalten, muß die Struktur traversiert werden. Dies geschieht, indem ausgehend von einem Einstiegsknoten dessen Nachbarn besucht werden, die wiederum nach ihren Nachbarknoten befragt werden, um so die Navigation rekursiv fortzuführen. Für jeden auf diese Weise erreichten Knoten wird eine mitgegebene Funktion ausgeführt. Solche Funktionen werden häufig als Task-Funktionen bezeichnet. Interne Logik sorgt

dafür, daß während eines Laufes jeder Knoten nur einmal erreicht wird und damit die Traversierung mit Erreichen aller Knoten terminiert. Task-Funktionen haben darüber hinaus die Möglichkeit, die Navigation zu beeinflussen. Durch Rückgabe geeigneter Parameter können sie veranlassen, die Traversierung der direkten Nachbarn des gerade besuchten Entities oder sogar den gesamten Traversierungsvorgang abzubrechen.

Das Gesamtnetz besitzt einen dedizierten Einstiegsknoten, von dem aus es möglich sein muß, alle anderen Entities durch rekursive Suche zu erreichen. In der Praxis werden allerdings von den verschiedenen Applikationen weitere eigene Einstiegspunkte verwaltet, die einen gezielten und damit schnellen Zugriff auf einen Teilbereich der Daten ermöglichen. Ebenso ist es in der Regel erwünscht, nur Ausschnitte des Netzwerkes zu traversieren. Hierzu können sogenannte *Scouts* verwendet werden, die lokale Zusammenhänge – oft auch nur die Grenzen – einer Wolke von Applikationsdaten kennen und so die Navigation beeinflussen können. Scouts sind letztlich spezielle Task-Funktionen, die ausschließlich zum navigieren benutzt werden, um die eigentliche Task-Funktion gezielt auf Entities einer Applikation – allgemein einer Untermenge des Netzes – anwenden zu können.

Weiterhin ist die Definition logischer Sub-Netzwerke innerhalb des Gesamtnetzes möglich und aus Gründen der Effizienz auch sinnvoll. Ein Konzept hierzu wird im folgenden Abschnitt beschrieben. Zur internen Berechnung komplexer Modellieroperationen ist es zulässig, isolierte Netzwerke temporär aufzubauen, die nach Abschluß der Operation aber wieder in das Gesamtnetz integriert werden müssen.

2.1.2 Cluster-Konzept

Das Entity-Netzwerk einer großen CAD-Baugruppe kann durchaus einige Millionen Entities umfassen. Daraus wird unmittelbar ersichtlich, daß eine gewisse Strukturierung des Netzes dringend erforderlich ist. In OneSpace Designer Modeling werden dazu logische Einheiten – sogenannte *Cluster* – definiert. Dies geschieht zum einen, um die zur Traversierung benötigte Zeit auf ein Minimum zu beschränken, zum anderen aber auch, um für den Benutzer greifbare Objekte wie Teile oder Baugruppen abbilden zu können. Beispiele solcher Einheiten finden sich weiter unten in

der Beschreibung des geometrischen Modells oder des Strukturmodells. Den Design-Richtlinien folgend, gibt es auch innerhalb eines Clusters keine zentrale Stelle, die alle zugehörigen Entities verwaltet. Vielmehr definieren die Entities im Netzwerk selbst, was zu einem Cluster gehört. Diese Vorgehensweise erlaubt es, ohne Änderungen an der Basisfunktionalität jederzeit neue Typen von Clustern zu definieren.

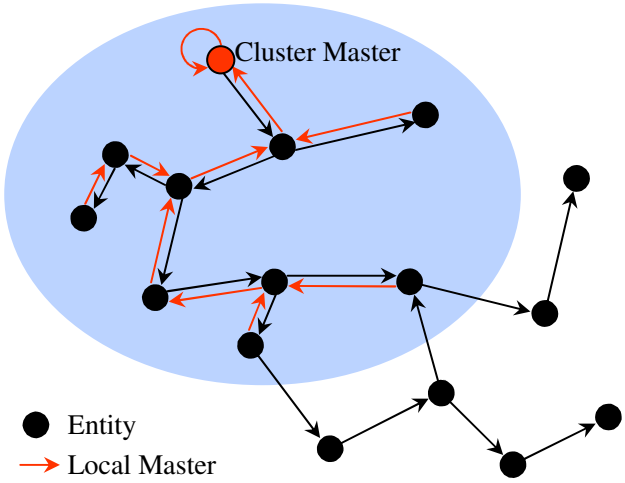


Abbildung 2.2: Ein Cluster innerhalb eines Entity-Netzwerks

Für jeden Cluster existiert ein ausgezeichnetes Entity, der sogenannte *Cluster Master*, der als Repräsentant des Clusters fungiert. Jedes Entity implementiert eine Methode `local_master()`, die auf ein benachbartes Entity verweist, dessen Zugehörigkeit zu einem Cluster das befragte Entity erben möchte. Für den Cluster Master liefert diese Methode einen Verweis auf sich selbst zurück. Das Entity kann also in der Regel nicht direkt mitteilen, zu welchem Cluster es gehört. Es kann aber einen Weg dorthin aufzeigen, indem es auf ein Entity verweist, das einen Schritt näher am Cluster Master ist. Um nun herauszufinden, zu welchem Cluster ein beliebiges Entity gehört, muß, ausgehend vom befragten Entity, rekursiv der Kette aller lokalen Master gefolgt werden, um schließlich am Cluster Master anzukommen. Die **Abbildung 2.2** zeigt einen Cluster innerhalb eines Netzwerks von Entities[5].

[5] Vgl. [BrKu95]

Der folgende Algorithmus beschreibt eine Möglichkeit, alle zu einem Cluster gehörenden Entities zu traversieren:

1. Starte mit dem Cluster Master
2. Bearbeite das Entity gemäß Aufgabenstellung
3. Besuche jedes Nachbar-Entity und stelle fest, zu welchem Cluster es gehört
    a. Wenn es zum selben Cluster gehört, das heißt, wenn sein Cluster Master mit dem Startpunkt der Suche übereinstimmt, führe den Algorithmus mit der Bearbeitung des Entities fort (2.)
    b. Wenn es zu einem anderen Cluster gehört, hat die Suche die Cluster-Grenze überschritten. Dieses Entity wird nicht weiter prozessiert und die Suche wird von diesem Knoten aus nicht weitergeführt.

Dieses Vorgehen beruht auf einigen Annahmen, die von den Datenstrukturen und Methoden zur Traversierung erfüllt werden müssen. So muß sichergestellt sein, daß jeder Knoten während der Suche nur einmal erreicht wird, um eine Terminierung des Algorithmus zu gewährleisten. Jedes Entity muß eindeutig einem Cluster zugeordnet sein. Außerdem muß es möglich sein, jedes Entity des Clusters vom Cluster Master aus zu erreichen. Das allein ist zwar notwendig, jedoch noch nicht hinreichend. Es muß überdies der gesamte Pfad vom Cluster Master zum Entity innerhalb des Clusters liegen.

Der beschriebene Algorithmus ist eine typische Aufgabe der in Kapitel 2.1.1 eingeführten Scout-Funktion. Dadurch wird es einer Applikation ermöglicht, auf effiziente Weise die Bearbeitung von Entities auf einen definierten Cluster zu beschränken, ohne das gesamte Entity-Netzwerk traversieren zu müssen. Die beschriebene Vorgehensweise zur Definition eines Clusters ist vollständig objektorientiert. Durch Implementieren einer einzigen Methode im Entity ist es möglich, daß die Objekte selbst die Größe, Struktur und Gestalt eines Clusters festlegen können. Auch neue, in der Zukunft zu erstellende Entities können so in einen Cluster integriert werden, ohne die bestehende Logik oder den vorhandenen Programm-Code ändern zu müssen.

2.1.3 Beziehungen zwischen Clustern

Das vorliegende Cluster-Konzept beschreibt eine Unterteilung des gesamten Netzwerks in disjunkte Abschnitte. Das allein ist aber für eine sinnvolle Strukturierung eines CAD-Modells noch nicht ausreichend. Vielmehr ist es erforderlich, auch den Aufbau von Beziehungen zwischen den Clustern zu unterstützen. In diesem Zusammenhang finden integrative und hierarchische Beziehungen ihre Anwendung.

Abhängig von der jeweiligen Applikation beschränkt sich die Sicht auf die Daten eines Clusters auf eine in sich logisch abgeschlossene Teilmenge desselben. Integrative Beziehungen zwischen Clustern setzen eine weitere Unterteilung eines Clusters nach demselben Konzept voraus, bei der die betrachtete Teilmenge ihrerseits den Anforderungen an einen Cluster genügt und in den übergeordneten Cluster integriert ist. Einzig die Forderung nach eindeutiger Zuordnung aller Entities zu einem Cluster wird in abgeschwächter Form angewandt. Für die Unterteilung des gesamten Netzwerks besteht zunächst die Forderung, daß alle Entities eindeutig einem Cluster zugeordnet sein müssen. Bei der weiteren Unterteilung eines Clusters dagegen müssen nicht mehr alle Entities eindeutig einem untergeordneten Cluster zugerechnet werden können. Vielmehr bleiben Entities, die zu keinem untergeordneten Cluster gehören, automatisch in der Obhut des übergeordneten Cluster Masters. Die weitere Aufteilung eines Clusters muß also nicht vollständig sein. Die Forderung nach konsistenten Pfaden vom Cluster Master zum Entity bleibt allerdings auch in diesem Fall bestehen.

Die Integration eines untergeordneten Clusters wird dadurch realisiert, daß dessen Cluster Master seinen lokalen Master dynamisch den Anforderungen anpassen kann. Im Kontext des Gesamtmodells – dies soll als Normalfall angesehen werden – verweist seine `local_master()` Methode auf ein benachbartes Entity innerhalb des übergeordneten Clusters. Der Cluster Master des untergeordneten Clusters gibt dadurch seine Stellung als Master auf, verhält sich wie ein gewöhnliches Entity und integriert somit sich und alle zu ihm gehörenden Entities in den übergeordneten Cluster. Im Kontext einer speziellen, auf dieser Datenstruktur basierenden Applikation kann dieses Entity aber wieder die Rolle des Cluster Masters übernehmen, indem es in seiner `local_master()` Methode temporär auf sich selbst verweist.

Die **Abbildung 2.3** zeigt eine integrative Beziehung zwischen Clustern. Solange der integrierte Master ein Entity des übergeordneten Clusters als lokalen Master definiert, sind der Master und alle Entities in seiner Obhut – in der Abbildung durch den hell unterlegten Bereich hervorgehoben – in den übergeordneten Cluster integriert. Dies betrifft somit alle Entities, deren Kette von lokalen Mastern den integrierten Master erreicht. Sobald eine Applikation den lokalen Master des integrierten Repräsentanten temporär auf sich selbst verweisen läßt, erfüllt dieses Entity alle Kriterien eines Cluster Masters und die Applikation kann die so definierte Untermenge als eigenständigen Cluster bearbeiten.

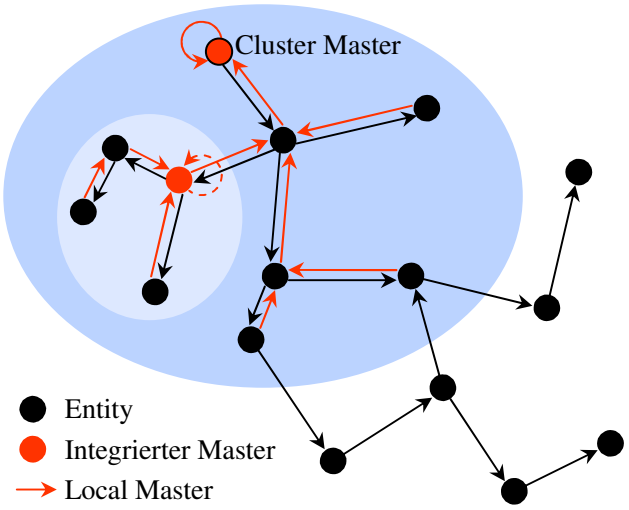


Abbildung 2.3: Integrative Cluster-Beziehung

Integrative Beziehungen werden beispielsweise zur Verwaltung von Geometrie-Daten innerhalb eines Teils verwendet, wie sie in Abschnitt 2.2 beschrieben wird. Sie werden aber auch bei der Entwicklung neuer Konzepte in Kapitel 5 eine grundlegende Rolle spielen.

Neben den integrativen Beziehungen zwischen Clustern finden auch hierarchische Beziehungen Anwendung bei der Strukturierung eines CAD-Modells. Eine derartige Beziehung entsteht dadurch, daß ein Entity eines Clusters auf den Master eines anderen Clusters verweist. **Abbildung 2.4** stellt eine solche Beziehung dar.

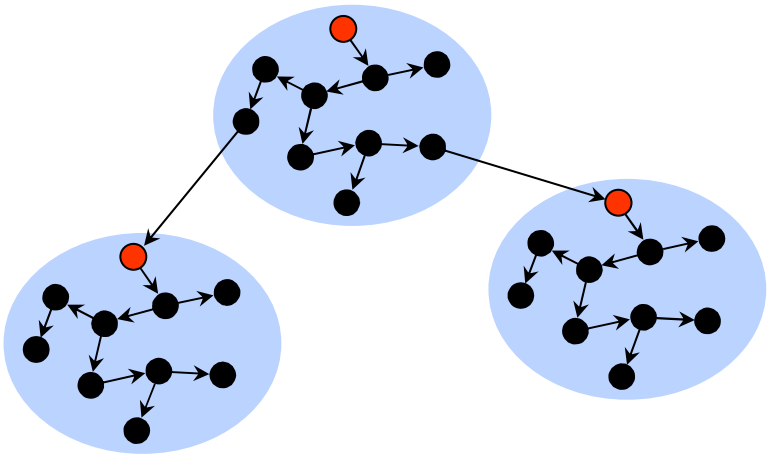


Abbildung 2.4: Hierarchische Cluster-Beziehung

Die Möglichkeit, daß ein Entity auf mehrere Cluster Master verweist, soll ebenso zugelassen werden wie die Möglichkeit, einen Cluster Master von mehreren Entities aus zu referenzieren. Zyklische Verweise, auch über mehrere Stufen hinweg, sollen jedoch ausgeschlossen werden. Dies erlaubt den Aufbau eines gerichteten, azyklischen Graphen von Clustern. In Abschnitt 2.3 wird beschrieben, wie unter Ausnutzung hierarchischer Cluster-Beziehungen das Strukturmodell abgebildet wird.

2.1.4 Persistenz von Clustern

CAD-Daten haben typischerweise eine lange Lebensdauer. Sie werden gewöhnlich über viele Arbeitssitzungen hinweg immer wieder benötigt, teils verändert und danach in weiteren Schritten wiederverwendet. Dabei kommt es häufig vor, daß verschiedene Arbeitsschritte auf verschiedenen Rechnern durchgeführt werden. Dies macht die Forderung nach Persistenz dieser Daten unabdingbar.

Mit Hilfe des Cluster-Konzepts wurde gezeigt, wie das gesamte Netzwerk der CAD-Daten in logische Untereinheiten aufgeteilt werden kann. Die dadurch gewonnene Strukturierung soll auch beim Speichern der Daten auf ein Speichermedium erhalten bleiben. Dem wird Rechnung getragen, indem ein Cluster in die Lage versetzt wird, seinen Inhalt in eine Datei zu schreiben, bzw. eine Funktionalität zur Verfügung gestellt wird, mit deren Hilfe aufgrund der abgespeicherten Information ein Cluster wiederhergestellt werden kann. Die auf Clustern basierende Strukturierung auch auf Dateiebene bietet die Möglichkeit, daß nur der Teil des Gesamtmodells, der tatsächlich verändert resp. neu erzeugt wird, abgespeichert werden muß. Ein weiterer

Vorteil dieser Vorgehensweise ist, daß in späteren Arbeitsschritten nicht immer das gesamte Modell geladen werden muß, sondern das Laden der für die Bearbeitung relevanten Teilmenge der Daten ausreichend ist.

Das Speichern eines Clusters in eine Datei erfolgt auf generische Art und Weise. Aus Effizienzgründen werden alle Daten in einem kompakten Binärformat geschrieben. Dieses Format ist plattformunabhängig, so daß der Austausch der Dateien über Betriebssystemgrenzen hinweg problemlos möglich ist. Um Kompatibilität zwischen verschiedenen Dateiversionen zu wahren, wird eine entsprechende Informationseinheit an den Beginn der Datei geschrieben. Jedes Entity ist selbst dafür verantwortlich, seine Daten und Beziehungen zu anderen Entities persistent zu machen. Lediglich die benötigte Infrastruktur, wie das Öffnen von Dateien, Traversieren des Entity-Netzes oder Binär-Kodieren von atomaren Datenelementen, wird von zentraler Stelle zur Verfügung gestellt.

Der in Abschnitt 2.1.2 beschriebene Algorithmus wird benutzt, um alle Entities des Clusters zu erreichen. Für jedes Entity wird zunächst ein Entity-Header geschrieben, aus dem der Typ des Entities hervorgeht sowie Angaben, die die eindeutige Identifikation des Entities zulassen. Der Typ wird benötigt, um bei einer späteren Ladeoperation die richtige Lademethode bestimmen zu können, die Kennung dient zum Wiederaufbau der Beziehungen zwischen den Entities. Anschließend ist es Aufgabe des Entities, in seiner `store()` Methode seine Daten unter Verwendung der bereitgestellten Methoden zur Binärkodierung zu sequentialisieren. Dabei wird zur Kodierung von Verweisen auf andere Objekte deren Kennung verwendet. Abgeschlossen wird das Speichern eines Entities durch einen Entity-Trailer. Dadurch ist es auf unterster Ebene möglich, die Daten eines unbekannten Entities zu überlesen, was durchaus erforderlich sein kann, wenn beispielsweise Dateien gelesen werden sollen, die Daten einer Zusatzapplikation enthalten, die zum Zeitpunkt des Lesens nicht aktiv ist.

Während das Speichern eines Entities durch eine virtuelle Objektmethode realisiert werden konnte, ist dies – mangels Objekt – beim Laden nicht möglich. Daher muß für jede Entity-Klasse eine statische Funktion zum Lesen der Daten beim System angemeldet werden. Wird nun ein Entity-Header aus einer Datei gelesen, kann die

entsprechende Ladefunktion mit Hilfe der darin enthaltenen Typ-Information identifiziert werden. Durch ihren Aufruf wird ein Entity angelegt und die folgenden Daten können gelesen und dekodiert werden, um die Werte den Datenfeldern des Entities zuzuweisen. An dieser Stelle kann auch die in der Datei abgespeicherte Versionsinformation ausgenutzt werden, die die Anzahl und Reihenfolge der zu lesenden Datenelemente individuell für jede Entity-Klasse festlegt. Wenn eine Ladefunktion gefunden werden konnte, ist der Entity-Trailer ohne Bedeutung und wird überlesen.

Nachdem alle Daten gelesen wurden, sind durch die Ladeoperation alle Entities des Clusters als solche wiederhergestellt. Einzig die Beziehungen zwischen den Entities müssen noch aktualisiert werden. Dazu wird die eindeutige Kennung verwendet, die in den Entity-Header geschrieben wurde. Zu Beginn der Ladeoperation wird eine Tabelle aufgebaut, zu der beim Laden eines jeden Entities ein Eintrag hinzugefügt wird, der die aktuelle Speicheradresse der Kennung des Entities zuordnet. Allen Verweisen innerhalb eines Entities wird beim Laden zunächst Metainformation zugegewiesen, aus der die Kennung des referenzierten Objekts hervorgeht.

Nachdem alle Entities des Clusters gelesen wurden, das heißt, nachdem jeder Kennung die Speicheradresse des zugehörigen Objekts zugeordnet werden kann, wird für jedes geladene Entity dessen Methode `convert_pointers()` aufgerufen. Diese Methode wertet die Metainformation zu jeder Referenz aus, erfragt die zur Kennung des referenzierten Objekts gehörige Speicheradresse und ersetzt schließlich die Referenz in situ, so daß sie wieder direkt auf das Zielobjekt verweist. Nach Anwendung dieser Operation auf alle geladenen Entities ist der Cluster insgesamt wiederhergestellt.

Da alle Entities eines Clusters im Moment des Speicherns im Hauptspeicher sind und gemeinsam abgespeichert werden, ist deren Speicheradresse zu diesem Zeitpunkt als eindeutige Kennung für Verweise innerhalb des Clusters, sogenannten *internen Referenzen*, ausreichend. Sie kann daher als Grundlage für die eben beschriebene Konvertierung dienen – die trotzdem notwendig ist, da im allgemeinen nicht davon ausgegangen werden kann, daß die Entities nach dem Laden wieder dieselben Speicheradressen haben werden.

Für Verweise zwischen Clustern, sogenannten *externen Referenzen*, stellt sich die Lage anders dar. Insbesondere dann, wenn Cluster aufeinander verweisen, die in unterschiedlichen Arbeitssitzungen abgespeichert wurden, sind Speicheradressen als eindeutige Kennung offensichtlich untauglich. Diese Situation kommt zum Beispiel dann vor, wenn ein referenzierter Cluster in einer späteren Sitzung verändert und nochmals abgespeichert wird.

Zur Lösung des Problems wird hier die Kennung um einen global eindeutigen Wert, die *SysID*, erweitert. In diesen Wert gehen Parameter wie Hardware-ID, normalisierte Startzeit der Arbeitssitzung, in der das Entity erzeugt wurde, sowie die laufende Nummer des Entities in dieser Sitzung ein. Beim Abspeichern einer externen Referenz wird neben der erweiterten Kennung des Entities, auf das verwiesen wird, auch die Kennung und der Dateiname des Clusters, in dem sich dieses Entity befindet, mit aufgenommen. Dies ermöglicht das automatische Nachladen von referenzierten Clustern.

Der Prozeß der Konvertierung externer Referenzen verläuft im Prinzip genauso wie oben für interne Referenzen beschrieben. Da es erforderlich ist, daß zur Konvertierung vorher jeder Kennung eine Speicheradresse zugeordnet worden ist, können externe Verweise erst dann aktualisiert werden, wenn alle referenzierten Cluster geladen worden sind. Zur Effizienzsteigerung kann ein mehrstufiges Verfahren zum Einsatz kommen, bei dem zunächst je Cluster alle internen Referenzen vorkonvertiert werden, bevor schließlich nach Abschluß des gesamten Ladevorgangs auch die externen Referenzen nachgezogen werden.

2.2 Geometrisches Modell

Einer der grundlegendsten Bestandteile eines jeden 3D-CAD-Systems ist die modellhafte Beschreibung der konstruierten Teile im dreidimensionalen Raum. Der dafür verwendeten Datenstruktur und den Algorithmen zu ihrer Verarbeitung kommt eine Schlüsselrolle im CAD-System zu. Diese Struktur muß eine möglichst realitätsnahe Repräsentierung der realen Bauteile ermöglichen.

Über dieses 3D-Modell hinaus, auf dessen Darstellung im folgenden Abschnitt näher eingegangen wird, läßt sich der Umfang des geometrischen Modells noch um das Zeichnungsmodell (2D-Modell) sowie das Bildmodell erweitern, die beide aus dem 3D-Modell abgeleitet sind (vgl. [Kal97]). Das Zeichnungsmodell besteht dabei hauptsächlich aus Projektionen und Ansichten des Bauteils. Zusätzlich enthält es weitere Elemente einer normgerechten Zeichnung, wie Maße, Beschriftungen oder Toleranzangaben. Darüber hinaus ist das Zeichnungsmodell in den meisten Systemen assoziativ mit dem 3D-Modell verbunden, damit Änderungen in einem Modell automatisch im anderen nachgeführt werden können. Das Bildmodell enthält alle zur graphischen Darstellung des Bauteils benötigten Elemente. Diese werden in der Regel durch Approximation aus der Oberfläche des 3D-Modells abgeleitet.

Die Qualität des geometrischen Modells und seine Genauigkeit sind für die spätere Realisierung der konstruierten Teile von herausragender Bedeutung. Am Bildschirm während der Konstruktion quasi unsichtbare Unzulänglichkeiten wie Messerkanten, extrem kurze Kanten, Flächenüberschneidungen oder Löcher können leicht dazu führen, daß ein Teil später nicht zu fertigen ist oder nachgeschaltete Berechnungen, wie zum Beispiel NC-Programme, keine sinnvollen Ergebnisse liefern bzw. abbrechen.

2.2.1 3D-Modell

Allgemein betrachtet lassen sich 3D-Modelle in drei voneinander unabhängige Systemklassen unterteilen. Basierend auf der Art der verwendeten Grundelemente unterscheidet man in der Literatur[6] zwischen Draht-, Flächen- und Volumenmodellen. Bedingt durch ihre charakteristischen Eigenschaften, die im folgenden kurz betrachtet werden, eignen sich die Modelle unterschiedlich gut für verschiedene Anwendungsbereiche. Im Bereich der 3D-CAD-Systeme haben sich Volumenmodelle weitestgehend durchgesetzt. **Abbildung 2.5** gibt einen Überblick über die verschiedenen Klassen, der auf [Grä89] zurückgeht. Dieser wird ergänzt durch eine weitere Unterteilung des Volumenmodells in verschiedene Modelltypen.

[6] Vgl. u. a. [Grä89], [Pah90], [SpKr84], [Str00]

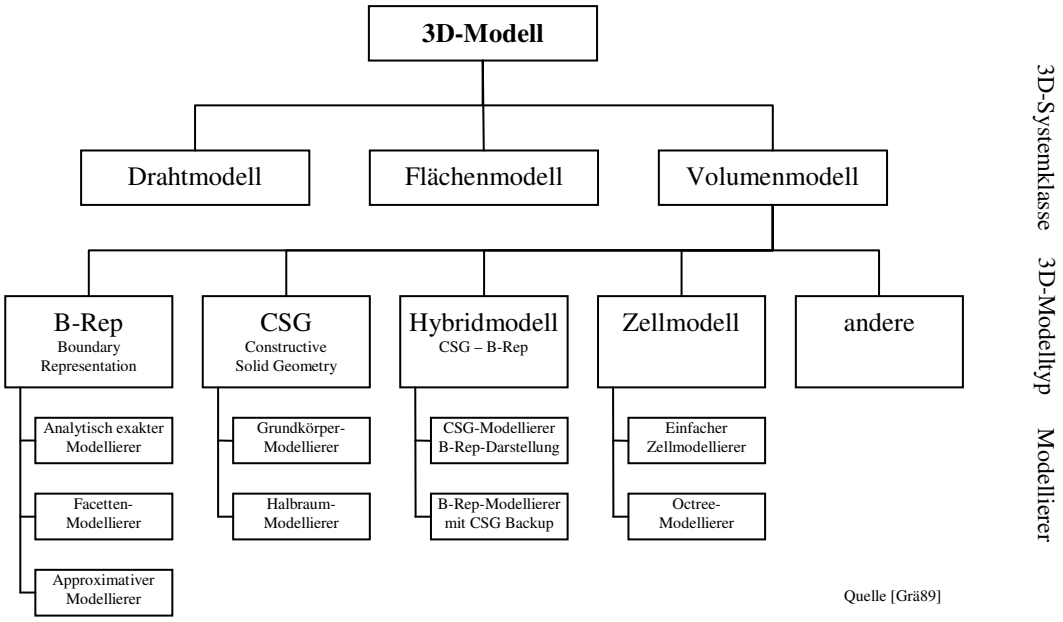


Abbildung 2.5: Überblick über 3D-Modelle

Drahtmodelle

Diese Modellklasse zeichnet sich durch ihre geringe Komplexität aus und ist daher eine häufig anzutreffende Form des 3D-Modells. Ein Bauteil wird in diesem Modell durch seine Kanten und Eckpunkte beschrieben. Dadurch entsteht ein gewisser räumlicher Eindruck des Objektes. Außerdem ist es möglich, Projektionen abzuleiten.

Seine Einfachheit ist zugleich auch der größte Nachteil eines Drahtmodells in bezug auf die Verwendbarkeit in einem 3D-CAD-System. So ist zum Beispiel die Darstellung eines Teils allein durch seine Begrenzungskanten nicht notwendigerweise eindeutig. Durch fehlende Flächeninformation sind weder Kollisionsberechnungen möglich, noch kann über die Sichtbarkeit von Kanten entschieden werden. Überdies können aus dem Modell keine physikalischen Eigenschaften wie das Volumen, der Schwerpunkt oder Trägheitsmomente abgeleitet werden.

Flächenmodelle

Eine weitere 3D-Modell-Klasse basiert auf der Darstellung von Flächenverbänden. Bei der Verwendung dieser Klasse eines 3D-Modells steht die Beschreibung von Oberflächen eines Bauteils, wie etwa der Außenhaut eines Schiffskörpers, im Vordergrund. Oft sind in diesem Zusammenhang Flächen anzutreffen, deren Form allein

durch analytische Geometrien nicht mehr darstellbar ist, sondern die Verwendung von Freiformgeometrien erfordert. Körperhafte Eigenschaften spielen hierbei eine untergeordnete Rolle. Systeme, die auf Flächenmodellen basieren, finden eine weite Verbreitung in Bereichen wie der Strömungsberechnung oder im Formenbau.

Während Kollisions- und Sichtbarkeitsberechnungen auf Flächenbasis durchgeführt werden können und physikalische Eigenschaften zumindest eingeschränkt abgeleitet werden können, fehlen weiterhin einige wesentliche Eigenschaften, die von einem Modell gefordert werden, das als Grundlage für ein 3D-CAD-System fungieren kann. So ist weder eine Richtung für das Material definiert, das heißt, es ist unklar, was das Innere des Körpers ist, noch ist die räumliche Integrität des Körpers gesichert – der Flächenverband muß nicht notwendigerweise abgeschlossen sein, außerdem sind mögliche Flächendurchdringungen nicht ausgeschlossen.

Volumenmodelle

Die volumenhafte Repräsentation dreidimensionaler Objekte ist schließlich Kernstück der Volumenmodelle. Auf der Basis dieser Klasse von 3D-Modellen ist es nun möglich, Bauteile in vollständiger und eindeutiger Weise zu beschreiben, deren Materialsituation zudem algorithmisch klar erfaßbar ist. Dabei ist Vollständigkeit in dem Sinne zu verstehen, daß es nicht möglich ist, Körper mit fehlenden Kanten oder Flächen zu definieren.

Die Eindeutigkeit der Darstellung gewährleistet, daß sich für jeden beliebigen Punkt des Raumes exakt bestimmen läßt, ob er im Inneren des Körpers liegt – das heißt dem Material des Körpers zuzurechnen ist. Dies hat zur Folge, daß das Modell jetzt automatisch interpretiert werden kann, zum Beispiel zur Bestimmung physikalischer Eigenschaften wie Volumen und Schwerpunkt, aber auch durch nachgeschaltete Berechnungsprogramme. Eine visuelle Interpretation, die gegebenenfalls nur unter Berücksichtigung von Zusatzinformationen möglich ist, ist somit nicht länger erforderlich. Ebenso wird es durch die Eigenschaften des Volumenmodells möglich, Berechnungen zur Sichtbarkeit bzw. Kollisionsüberprüfung in eindeutiger Weise durchzuführen.

In der Konsistenz der Daten ist eine weitere charakteristische Eigenschaft der Volumenmodelle zu sehen. So können im allgemeinen nur physikalisch realistische Körper definiert werden. Sich selbstdurchdringende Körper, deren Erzeugung bei Verwendung der Draht- oder Flächenmodelle noch möglich war, sind im Volumenmodell nicht mehr zulässig. Dabei ist zu beachten, daß „physikalisch realistisch“ nicht automatisch damit gleichzusetzen ist, daß ein solches Teil nach maschinenbaulichen Gesichtspunkten auch zu fertigen ist. Als Beispiel sei ein Körper genannt, der einen Hohlraum vollständig umschließt. In der Summe erfüllt das Volumenmodell damit alle wesentlichen Forderungen, die an ein Datenmodell gestellt werden, das einem 3D-CAD-System zugrunde gelegt wird.

Die wichtigsten Techniken zur Generierung von Volumenmodellen umfassen dabei (vgl. u. a. [Str00]) die Expansion zweidimensionaler Profile, zum Beispiel Extrudieren oder Rotieren, die Formelementekonstruktion, die 3D-Digitalisierung sowie die Erzeugung von Freiformgeometrien, zum Beispiel durch Sweeping, Lofting oder Skinning. Des weiteren sind *Boole’sche Operationen* von Bedeutung. Hierbei entstehen neue Körper durch mengentheoretische Verknüpfung bestehender Körper mittels Addition, Subtraktion bzw. Bildung des Durchschnitts. Die hierzu verwendeten Algorithmen müssen sicherstellen, daß das Ergebnis der Verknüpfung zweier eindeutig und vollständig beschriebener Modelle wiederum in eindeutig und vollständig beschriebenen Modellen darstellbar ist.

In **Abbildung 2.5** wurde bereits gezeigt, daß sich das Volumenmodell selbst weiter in verschiedene Modelltypen gliedern läßt, die eng mit dem Aufbau der jeweiligen Datenstruktur verbunden sind. Die klassischen 3D-Volumenmodelle unterteilen sich dabei hinsichtlich der Beschreibung des Objektmodells und der damit notwendig verbundenen Modellierungstechnik in *generative* Modelle und *akkumulative* Modelle. Zu den generativen Modellen zählen die CSG-Modelle (Constructive Solid Geometry: Halbraum- oder Grundkörpermodell), zu den akkumulativen die B-Rep-Modelle (Boundary-Representation: analytisch exaktes, Facetten- oder approximatives Modell). Neben diesen reinen Modelltypen existieren die sogenannten *hybriden* Modelle. Diese beschreiben ein Objektmodell auf der Basis einer Kombination unterschiedlicher Modelltypen.

Im CSG-Modell wird ein Objekt durch mengentheoretische Verknüpfung von Körpern beschrieben. Diese sind entweder vordefinierte Grundelemente oder Körper, die in einem Zwischenschritt selbst als Ergebnis aus einer Verknüpfung resultieren. Die Vorschrift zur Verknüpfung der Körper wird im sogenannten *Boole'schen Baum* abgelegt. Dabei werden die Knoten des Baums als Transformations- oder Konstruktionsoperationen aufgefaßt, während die Blätter des Baums die Grundelemente darstellen. An der Wurzel entsteht dann das zu beschreibende Modell. Als Grundelemente stehen in der Regel einfache Körper wie Quader, Zylinder, Kugeln oder Tori zur Verfügung.

Die Datenstruktur des CSG-Modells erlaubt eine kompakte Darstellung des resultierenden Körpers. Vorteilhaft ist auch die Tatsache, daß sich viele der Boole'schen Operationen technischen Operationen, wie etwa Bohren oder Fräsen, zuordnen lassen, was die Benutzung erleichtert. Allerdings ist zu beachten, daß einzelne Kanten oder Flächen des resultierenden Modells nicht explizit zur Verfügung stehen. Dadurch ist eine direkte Referenzierung solcher Objekte, etwa zum Anbringen von Attributen, die Fertigungsinformationen beinhalten, nicht möglich. Auch eine direkte graphische Darstellung ist damit nicht möglich. Die Visualisierung des im CSG-Modell beschriebenen Objekts erfolgt daher durch Evaluieren des Boole'schen Baums über speziell entwickelte Darstellungsalgorithmen, wie etwa Ray Tracing, oder – ähnlich wie bei den Boundary-Modellen – über temporär berechnete Begrenzungsflächen.

Aufgrund der fehlenden Möglichkeit, Kanten und Flächen direkt referenzieren zu können, spielen rein auf das CSG-Modell aufbauende Systeme kommerziell praktisch keine Rolle. Vielmehr haben sich Systeme durchgesetzt, die entweder ausschließlich auf dem B-Rep-Modell basieren oder zumindest auf einem hybriden Ansatz, der ein B-Rep-Modell einschließt. Wegen der Bedeutung, die der Boundary Representation dadurch zukommt, ist der Beschreibung dieses Modells ein eigener Abschnitt gewidmet. Darin wird neben dem Modell selbst auch dessen Datenstruktur sowie deren Abbildung auf das in Kapitel 2.1 erläuterte Entity-Konzept betrachtet.

2.2.2 Boundary Representation

Das Boundary-Representation-Modell[7] beschreibt den Ist-Zustand eines realen Bauteils. Die entsprechenden Elemente des Modells spiegeln die begrenzenden Flächen, Kanten und Eckpunkte des wirklichen Teils wider. Die Beziehungen der Elemente zueinander sind integraler Bestandteil des Modells, wohingegen eine zusätzliche Beschreibung, wie der Ist-Zustand erreicht wurde, anders als im CSG-Modell nicht vorhanden ist. Durch die Existenz von Flächen und Kanten ist es bei diesem Modell nun auch möglich, Referenzen direkt anzubringen. Fertigungsinformationen etc. können somit unmittelbar mit den entsprechenden Elementen des Modells assoziiert werden. Ebenso ist eine effiziente graphische Darstellung jeder Fläche, und damit des Bauteils insgesamt, möglich.

Die Komponenten des B-Rep-Modells lassen sich in zwei Klassen unterteilen. Die *Topologie* des Körpers wird durch die logischen Elemente, wie Flächen oder Kanten, und ihre Beziehungen zueinander definiert. Dies können nicht nur hierarchische, sondern auch Nachbarschaftsbeziehungen sein. Die *Geometrie* des Körpers wird durch die formgebenden Elemente, wie etwa Oberflächen, definiert. Topologie und Geometrie sind eng miteinander verbunden. Während die Topologie den prinzipiellen Aufbau des Körpers festgelegt, beschreibt die Geometrie dessen Form und Lage.

Darüber hinaus legen die einzelnen Topologie-Elemente den verwendeten Bereich der ihnen zugeordneten Geometrie-Elemente fest. Erst durch die vollständige Beschreibung beider Klassen kann ein Bauteil eindeutig dargestellt werden. Verschiedene Körper, die auf derselben Topologie basieren, können somit allein durch die Verwendungen verschiedener Geometrien eine unterschiedliche Gestalt aufweisen.

Die Datenstruktur zur Darstellung eines Bauteils im B-Rep-Modell beschreibt ein komplexes Netzwerk der beteiligten topologischen und geometrischen Elemente. Ihr prinzipieller Aufbau, der in **Abbildung 2.6** gezeigt wird, soll im folgenden näher betrachtet werden. Das Grundgerüst dieser Struktur besteht aus einer Hierarchie verschiedener Ebenen topologischer Elemente.

---

[7] Auch abkürzend B-Rep-Modell genannt

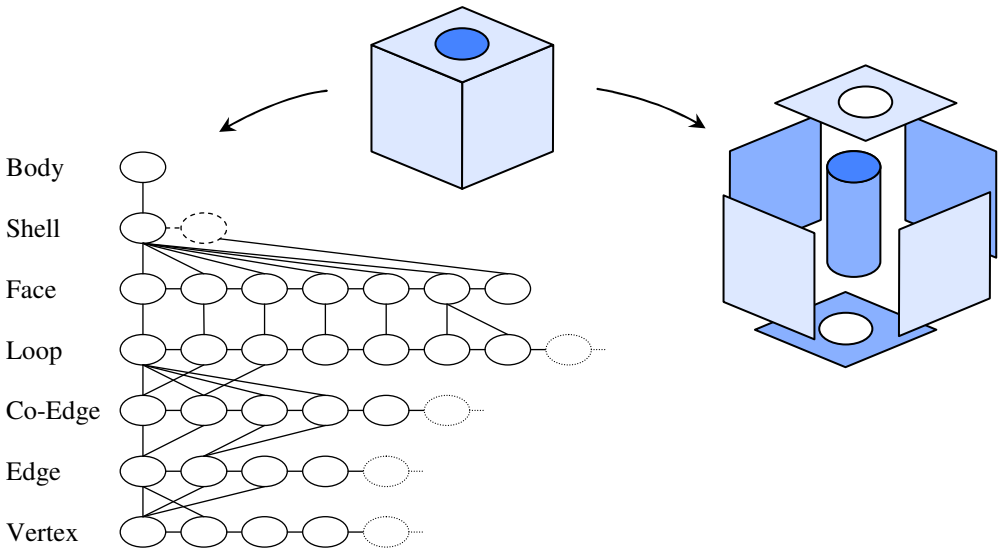


Abbildung 2.6: Prinzipieller Aufbau eines B-Rep-Modells

Der *Body* ist der Einstiegspunkt in diese Hierarchie. Er steht stellvertretend für den gesamten Körper, das heißt für das Modell eines Bauteils aus der realen Welt. Der Body beschreibt ein vollständiges, zusammenhängendes Volumen mit Hilfe von Schalen, sogenannten Shells.

Eine *Shell* ist ein zusammenhängender Flächenverband. Dieser kann die gesamte Oberfläche des Körpers umfassen – in diesem Fall muß der Verband vollständig und abgeschlossen sein. Er kann aber auch nur einen Teil der Oberfläche umfassen, etwa eine Tasche oder Rippe. In diesem Fall muß die Vereinigung aller Shells unterhalb des Bodys ein gültiges Volumen definieren. Einige Systeme lassen überdies eine weitere hierarchische Unterteilung von Shells in Subshells zu. Hierbei muß die Vereinigung der Subshells wieder einen zusammenhängenden Flächenverband ergeben.

Die *Face* beschreibt eine Fläche in topologischer Hinsicht. Die äußere Begrenzung der Fläche wird dabei mit Hilfe eines geschlossenen Konturzugs (Loop) festgelegt. Weitere, ebenfalls geschlossene Konturzüge zur Abgrenzung von Löchern im Inneren der Fläche können angegeben werden. Außerdem verweist die Face auf das ihr zugeordnete geometrische Element, die *Surface*, also die geometrische Beschreibung der Oberfläche, dessen Parameterbereich sie festlegt. Ein Beispiel einer solchen Flächendefinition wird in **Abbildung 2.7** gezeigt. Einige Systeme lassen zu, daß mehrere Faces, die auf derselben Geometrie basieren, das heißt Ausschnitte auf derselben Oberfläche definieren, ein und dieselbe Surface gemeinsam referenzieren.

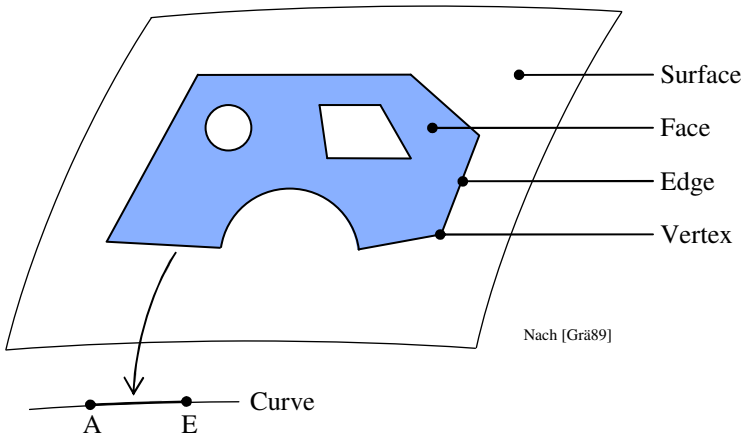


Abbildung 2.7: Begrenzung einer Surface durch Konturzüge der Face

Eine *Loop* legt einen geschlossenen Konturzug fest, der indirekt aus Kanten besteht. Da eine Loop mit einem Umlaufsinn belegt ist, ist ein direkter Verweis auf die Kanten nicht sinnvoll. Statt dessen verweist die Loop auf gerichtete Kanten, sogenannte Co-Edges. Die Orientierung der Loop, die die äußere Begrenzung einer Fläche beschreibt, ist im mathematisch positiven Drehsinn definiert. Somit ist das eingeschlossene Gebiet, das anschaulich betrachtet „links“ des Konturzugs liegt, der Materialbereich der Fläche. Daraus folgt, daß die Orientierung der Innenkonturen umgekehrt sein muß. Dies wird in **Abbildung 2.8** verdeutlicht. Der Materialbereich der Fläche ist dort dunkel hinterlegt.

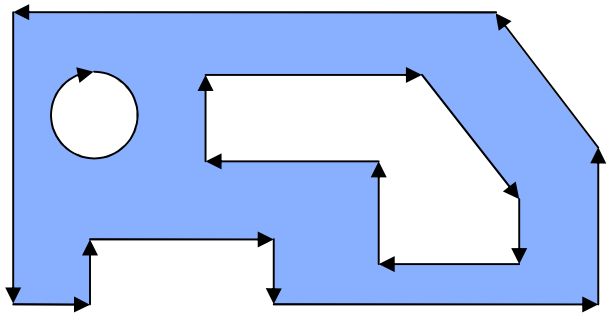


Abbildung 2.8: Orientierung der Konturzüge

Die *Co-Edge* ist ein mit einer Orientierung versehener Repräsentant der eigentlichen Modellkante. Zwei benachbarte Flächen zeichnen sich dadurch aus, daß sie mindestens eine gemeinsame Kante besitzen. Genaugenommen sind es ihre äußeren Konturzüge, die auf diese gemeinsame Kante verweisen. Da beide Konturzüge mathematisch positiven Umlaufsinn haben, sind sie zwangsläufig an der gemeinsamen Kante

entgegengesetzt orientiert. In **Abbildung 2.9** wird dies anschaulich dargestellt. Dem wird dadurch Rechnung getragen, daß jede Loop eine eigene Co-Edge besitzt, die beide gemeinsam auf dieselbe Modellkante verweisen. Die Co-Edges einer Loop werden in der Regel in einer bidirektionalen Ringliste verwaltet, so daß ein Traversieren des Konturzugs leicht möglich ist. Außerdem verweist jede Co-Edge auf ihren Partner, also die entsprechende Co-Edge aus der Begrenzung der anderen Fläche. Dadurch ist es möglich, alle Nachbarflächen einer Fläche effizient zu finden.

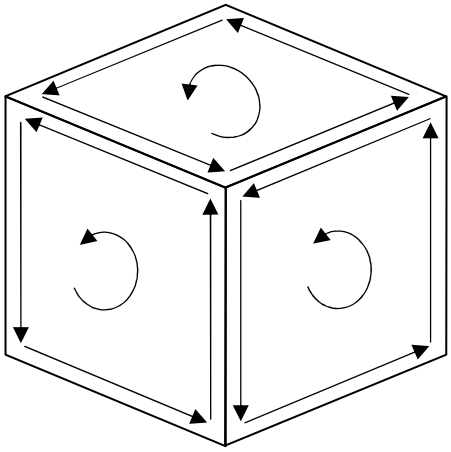


Abbildung 2.9: Entgegengesetzte Orientierung der Co-Edges an einer Modellkante

Topologische Aspekte der Modellkante werden durch die *Edge* beschrieben. Die Modellkante kann als Schnittkante der beiden ihr angrenzenden Flächen betrachtet werden. Zur Begrenzung der Kante verweist die Edge auf einen Anfangs- und Endpunkt, der jeweils gleichzeitig Eckpunkt des Modells ist. Bei geschlossenen Kanten fallen Anfangs- und Endpunkt zusammen. Es gibt einige Sonderfälle, wie zum Beispiel die Spitze eines Kegels, bei denen eine Modellkante nur von einer einzelnen Co-Edge referenziert wird. In solchen Fällen ist die Edge oft auch degeneriert, das heißt, sie zieht sich zu einem Punkt zusammen. Das geometrische Element, das von der Edge referenziert wird, ist die *Curve*. Auch hier legt die Edge den Parameterbereich der Curve fest.

Der *Vertex* schließlich beschreibt einen Eckpunkt im Modell. Ihm ist der *Point* als geometrisches Element zugeordnet, der die Lage des Punktes im Raum definiert. Vertices werden in einigen Systemen von den Edges, die sie als Anfangs- bzw. Endpunkt referenzieren, gemeinsam verwendet.

Wesentlich für die Konsistenz des beschriebenen Modells ist, daß in Relation stehende Geometrien an ihren Schnittstellen exakt oder zumindest im Rahmen einer vorgegebenen Toleranz übereinstimmen. Dies bedeutet, daß Endpunkte auf ihren Kurven liegen müssen. Genauso müssen auch die Kurven selbst auf den Flächen liegen, die an sie angrenzen. Die Güte der eingehaltenen Toleranz trägt in entscheidendem Maße zur Qualität des B-Rep-Modells bei. OneSpace Designer Modeling verwendet beispielsweise eine Toleranzgrenze von 10^{-6} mm.

Wie im CSG-Modell werden auch im B-Rep-Modell Änderungen des Modells durch Boole'sche Operationen erzeugt. Dabei werden wechselseitig alle Flächen der beteiligten Körper miteinander verschnitten. Die resultierenden Schnittkanten und Flächenstücke werden dann je nach Operation in das Modell integriert bzw. zu neuen Körpern zusammengefaßt. Darüber hinaus sind im B-Rep-Modell auch direkte Manipulationen möglich. Verrundungen oder lokale Veränderungen, wie das Hinzufügen von Taschen oder Rippen, lassen sich direkt in das Modell integrieren, ohne dafür vorher Hilfskörper erzeugen müssen.

In Kapitel 2.1 wurden Entities eingeführt und durch sie definierte Netzwerke beschrieben. Ausgehend von der Instanziierung jedes topologischen und geometrischen Elements als Entity des entsprechenden Elementtyps kann das B-Rep-Modell als Entity-Netzwerk betrachtet werden. Vom Body aus beginnend ist die Erreichbarkeit jedes Knotens aufgrund der beschriebenen Datenstruktur gewährleistet. Wird weiterhin vorausgesetzt, daß von jeder Hierarchie-Ebene aus auch die darüberliegende Ebene über Verweise zu erreichen ist – was der tatsächlichen Implementierung zumindest in OneSpace Designer Modeling entspricht –, erfüllt diese Datenstruktur auch die in Abschnitt 2.1.2 formulierten Anforderungen an einen Cluster. Alle Elemente des Modells sind auf Pfaden innerhalb des Modells erreichbar, genauso wie der Body, der in diesem Fall als Cluster Master fungiert, von allen Elementen aus erreichbar ist.

Als Ergebnis dieser Betrachtungen kann somit festgehalten werden, daß das geometrische Modell, realisiert durch die beschriebene Datenstruktur des B-Rep-Modells, in Form eines Clusters dargestellt werden kann. Im nächsten Abschnitt wird unter anderem aufgezeigt, wie diese Daten in die eines Teils integriert werden.

2.3 Strukturmodell

CAD-Modelle, insbesondere die im Rahmen dieser Arbeit betrachteten, bestehen aus einer Vielzahl unterschiedlicher Teile, die nach unterschiedlichen Aspekten zusammengefaßt werden können. Das Strukturmodell bildet dabei die strukturelle Gliederung von Bauteilen in ihre Unterelemente ab. Charakteristisch für das Strukturmodell ist die hierarchische Gliederung eines Produkts in Baugruppen, Unterbaugruppen und Einzelteile. Ergänzt werden kann dieses Modell durch die Hinzunahme sekundärer Elemente wie Arbeitsebenen oder Formelemente. Typischerweise wird der Aufbau durch einen gerichteten, azyklischen Graph abgebildet. Diese Gliederung ermöglicht eine Reduktion der Komplexität beim Entwurf umfangreicher Konstruktionen.

Die Betrachtungsweisen des Modells – und damit die Regeln für seinen Aufbau – können durchaus verschieden sein. Einerseits kann die Gliederung als Montagestruktur angesehen werden. Hierbei ist entscheidend, welches Teil in einem anderen verbaut wird. Andererseits kann die Struktur auch unter funktionalen Gesichtspunkten betrachtet werden, indem eine Funktion durch ihre Unterfunktionen aufgebaut wird. Eine Mischung der unterschiedlichen Formen innerhalb eines CAD-Modells kann dabei vorkommen. So sind Oberbaugruppen oftmals nach funktionalen Aspekten gegliedert. Je weiter hier ins Detail gegangen wird, um so mehr ist die Struktur von Montagekriterien geprägt. Zudem können Teile nach Funktion, Form oder Planungs- und Fertigungsanforderungen in Ähnlichkeitsklassen zusammengefaßt werden. Dies ermöglicht die Bildung und Fertigung von Teilefamilien. Produktvarianten können überdies dadurch beschrieben werden, daß unterschiedliche Kombinationen von Baugruppen verwendet werden.

Wesentliche Merkmale des Cluster-Konzepts finden sich in der Beschreibung des Strukturmodells wieder. Insbesondere die Möglichkeit, hierarchische Beziehungen zwischen Clustern aufbauen zu können, ist hier von Interesse. Diese Verbindung ist sicher nicht zufällig. In den folgenden Abschnitten soll daher erläutert werden, wie die eher abstrakten und äußerst technisch geprägten Eigenschaften der Entity-Welt genutzt werden können, um darauf Modelle aufzubauen, die der Begriffswelt eines CAD-Anwenders entsprechen.

2.3.1 Teile

Teile stellen den eigentlichen Kern des CAD-Modells dar. Sie repräsentieren quasi die Bausteine des Modells. Einerseits sind die Teile eine modellhafte Abbildung dessen, was später selbst das Ergebnis eines realen Fertigungsprozesses sein wird. Andererseits dienen sie in der Regel keinem Selbstzweck, sondern sind vielmehr Grundkörper, die zum Aufbau komplexer Baugruppen benötigt werden.

Häufig werden Teile innerhalb einer Baugruppe oder eines ganzen Produktes mehrfach verbaut. Als Beispiele sind hier Schrauben oder Muttern zu nennen. Um dieser Tatsache Rechnung zu tragen, werden die Daten zur Beschreibung von Teilen in zwei logische Einheiten unterteilt. Daten, die der modellhaften Abbildung des Teils als solches dienen, werden in der Einheit der *Modelldaten* zusammengefaßt. Alle Informationen dagegen, die für die Instanziierung – sprich das Verbauen eines Exemplars – des Modells benötigt werden, bilden die Einheit der *Instanzdaten*, die verschiedentlich auch *Exemplardaten* genannt werden. Die **Abbildung 2.10** zeigt eine auf [Schn99] zurückgehende schematische Darstellung der Teiledaten, aufgeteilt in Instanz- und Modelleinheit. Dieses Schema soll auch für weitere Darstellungen des Strukturmodells verwendet werden.

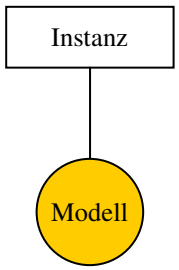


Abbildung 2.10: Schematische Darstellung eines Teils mit Instanz- und Modelldaten

Bestandteil der Modelldaten eines Teils sind alle Informationen, die für die eigentliche Darstellung des Teils als 3D-Modell notwendig ist. In erster Linie sind dies die Entities, die das geometrische Modell gemäß Abschnitt 2.2 definieren, sowie alle Entities, welche die daraus abgeleitete graphische Repräsentierung beschreiben. Außerdem gehören Grundeigenschaften, die allen verbauten Instanzen dieses Teils gemeinsam sind, wie zum Beispiel dessen Volumen oder umschließende Box, zu den Modelldaten. Anschaulich entspricht diese Einheit dem Modell eines Teils, das gera-

de zum Zwecke seines Einbaus aus dem Regal genommen wird. Form und Funktion sind vollständig definiert, die endgültige Bezeichnung und Lage in der Baugruppe sind noch offen. Weiterhin sind Daten, die der Verwaltung der Struktur dienen, Teil der Modelldaten. Diese spielen bei einem Teil – zumindest in der Betrachtungsweise des Anwenders – aber nur eine sekundäre Rolle. Im folgenden wird gezeigt, daß der Begriff der Modelldaten noch verallgemeinert werden kann.

Instanzdaten eines Teils enthalten die Informationen, die das Teil „greifbar“ machen. Hier sind vornehmlich Name und Position zu nennen, aber auch zum Beispiel Angaben zu dessen Sichtbarkeit. Durch das Erzeugen einer Instanz wird das Teil – das vorher abstrakt aus dem Regal genommen wurde – zu einem echten Bestandteil eines CAD-Modells.

Modell- und Instanzdaten bilden jeweils logische, in sich abgeschlossene Einheiten, deren Inhalt auf der Basis von Entities dargestellt wird. Weitere, zur Definition eines Clusters benötigte Eigenschaften, wie beispielsweise die Erreichbarkeit aller Entities von einem Master aus, können per Design sichergestellt werden. Folglich ist es naheliegend, das beschriebene Cluster-Konzept auf Modell- und Instanzdaten anzuwenden und dementsprechend beide Einheiten jeweils als Cluster zu behandeln.

Als Cluster Master dienen jeweils Entities, die eigens für diesen Zweck vom Basis-Entity abgeleitet und mit Grundfunktionalität zur Strukturverwaltung ausgestattet sind. Mit dem Verweis der Instanz auf ihre Modelldaten stellen die beiden Einheiten logisch eine hierarchische Cluster-Beziehung her. Zur Laufzeit wird diese Beziehung zur Steigerung der Performanz bidirektional erweitert.

Die Mehrfachverwendung von Teilen wird durch sogenanntes *Sharing* unterstützt. Dabei werden die Modelldaten für alle Exemplare dieses Teils nur einmal verwaltet und von allen Instanzen gemeinsam benutzt (vgl. **Abbildung 2.11**). Das geschieht aus zwei wesentlichen Gründen. Zum einen stellt dieses Vorgehen einen enormen Beitrag zum effizienten Umgang mit Ressourcen dar. Insbesondere Normteile wie Schrauben werden in großen Baugruppen oft mehrere tausendmal verbaut. Ohne Wiederverwendung der Modelldaten müßten sie ebenso oft parallel im Speicher gehalten werden.

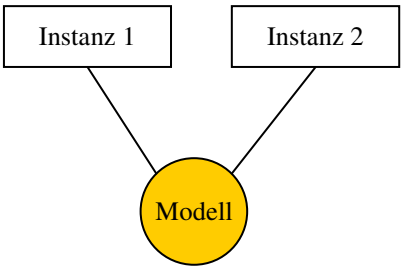


Abbildung 2.11: Zwei Exemplare desselben Teils

Zum anderen entspricht diese Methode genau der Assoziativität, die der Anwender von mehrfach verwendeten Teilen erwartet. Wenn ein solches Teil verändert wird, soll diese Änderung automatisch in allen Exemplaren reflektiert werden. Da die Modelldaten aber nur einmal vorhanden sind, ist dies trivialerweise der Fall.

2.3.2 Baugruppen

Die Notwendigkeit, die Konstruktion eines komplexen Modells nach verschiedenen Gesichtspunkten zu gliedern, liegt auf der Hand. Einige dieser Aspekte sind bereits zu Beginn dieses Kapitels kurz beschrieben worden. Diese Gliederung soll durch die Verwendung von Baugruppen realisiert werden, mit deren Hilfe eine Teilestruktur aufgebaut werden kann. Um dem Anwender einen möglichst natürlichen Umgang mit einer solchen Struktur zu erlauben, sind einige Anforderungen zu erfüllen:

Eine Baugruppe muß beliebig viele Teile aufnehmen können. Dabei spielt es keine Rolle, ob es sich dabei um Exemplare verschiedener Teile oder mehrere Exemplare desselben Teils – oder eine Kombination aus beiden – handelt. Ebenso müssen leere Baugruppen zulässig sein.

Eine Baugruppe muß ihrerseits andere Baugruppen, also Unterbaugruppen, aufnehmen können. Dadurch entsteht eine hierarchische Struktur. Es ist nicht zulässig, daß eine Baugruppe direkt oder indirekt Unterbaugruppe von sich selbst ist. Mit anderen Worten heißt das, zyklische Beziehungen sind nicht erlaubt.

Eine Mehrfachverwendung von Baugruppen soll möglich sein, ähnlich wie dies bereits für Teile beschrieben wurde.

Eine Baugruppe dient gleichsam als Synonym für ihren Inhalt. So bewirkt beispielsweise das Verschieben der Position einer Baugruppe eine entsprechende Verschiebung aller Teile in der Baugruppe.

Zur Umsetzung des Baugruppenkonzepts wird wieder auf die Teilung von Instanz- und Modelldaten zurückgegriffen. Die Vorgehensweise bei den Teilen wird dazu verallgemeinert und an die Gegebenheiten der Baugruppen angepaßt. Waren die Modelldaten eines Teils dadurch geprägt, daß sie Entities zur Definition und Repräsentierung des 3D-Modells enthielten, so sind die Modelldaten einer Baugruppe im wesentlichen durch die in der Baugruppe enthaltenen Teile bestimmt.

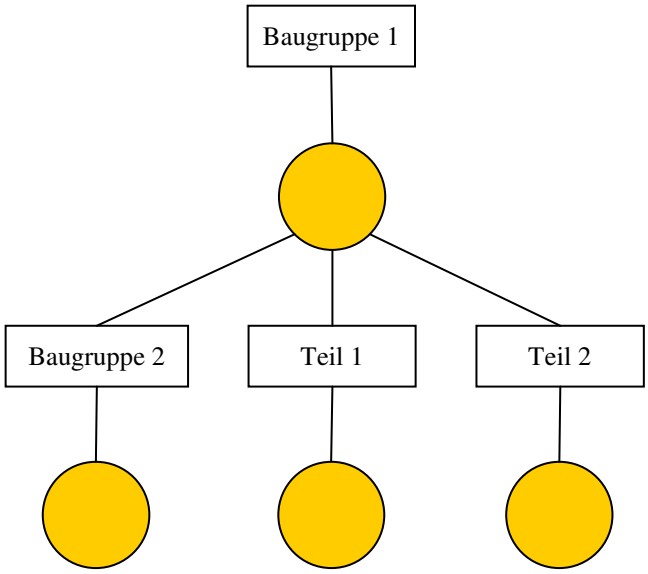


Abbildung 2.12: Baugruppe mit Teilen und Unterbaugruppe

Präziser ausgedrückt ist es das Wissen über die Zugehörigkeit der Instanzen von Teilen oder Unterbaugruppen, welches das abstrakte Modell einer Baugruppe ausmacht. Bei den Instanzdaten ist die Analogie noch offensichtlicher. Denn auch hier sind es Eigenschaften wie Bezeichnung oder Position, die die Baugruppe „greifbar“ machen. Diesem Konzept entsprechend ist in **Abbildung 2.12** eine Baugruppe schematisch dargestellt, die eine Unterbaugruppe und zwei verschiedene Teile enthält.

Die Anwendung des Cluster-Konzepts auf die Instanz- und Modelldaten von Baugruppen erfolgt in der gleichen Weise wie bei den Teilen. In der Implementierung gehen die entsprechenden Cluster Master faktisch auf gemeinsame Basisklassen

zurück. Auch bei den Baugruppen stellen die beiden Einheiten mit dem Verweis der Instanz auf ihre Modelldaten logisch eine hierarchische Cluster-Beziehung her. Dasselbe gilt für den Verweis des Baugruppen-Modells auf die zugehörigen Instanzen. Da eingangs für Baugruppen gefordert wurde, daß die Struktur durch einen gerichteten, azyklischen Graphen beschrieben werden kann, ist auch dieser Verweis mit den im Cluster-Konzept aufgestellten Regeln vereinbar.

2.3.3 Weitere Strukturelemente

Neben Teilen und Baugruppen gibt es in OneSpace Designer Modeling noch eine Reihe weiterer Objekte, die zur Strukturierung des 3D-Modells eingesetzt werden bzw. dessen Struktur ergänzen können. All diesen Objekten gemeinsam ist die Aufteilung in eine Instanz- und eine Modelleinheit. Da hier dasselbe Prinzip wie bei Teilen und Baugruppen weiterverwendet wird, fügen sich diese Objekte nahtlos in die Baugruppenstruktur ein und stellen vergleichbare Funktionalität zur Verfügung, wie etwa die gemeinsame Verwendung derselben Modelldaten durch mehrere Instanzen oder Methoden zur Datenpersistenz. Einige dieser Objekte werden im folgenden kurz beschrieben. Da sich die Instanzdaten auch bei diesen Objekten nur minimal unterscheiden, bezieht sich die Beschreibung in der Regel jeweils auf die Modelldaten.

Arbeitsebenen kommt im Modellierungsprozeß eine zentrale Rolle zu. Sie tragen die 2D-Profile, die den Umriß eines Werkzeugs definieren, mit dem das Modell bearbeitet wird. Durch Operationen wie zum Beispiel Extrudieren oder Rotieren werden aus den Profilen Hilfskörper generiert, die unter Verwendung Boole'scher Operationen mit dem Modell verknüpft werden.

Container stellen eine besondere Form von Baugruppen dar. Sie übernehmen dieselbe Funktion bei der Strukturierung, haben aber eine andere Bedeutung. Während Baugruppen häufig selbst mit Teilenummern versehen sind und als Bestandteil des Modells betrachtet werden, haben Container eine reine Verwaltungsfunktion. Sie bieten eine komfortable Möglichkeit, Hilfskörper, Werkzeuge, Rohteile etc. gemeinsam mit ihrem Konstruktionsteil oder ihrer Baugruppe zu verwalten. Da sie zusammen mit dem Teil positioniert werden, bleibt die relative Lage von Teil und Werkzeugen immer konsistent.

Zeichnungsdefinitionen legen im 3D-Modell den Inhalt der Zeichnungen fest, wohingegen die eigentliche 2D-Zeichnung außerhalb des Modells verwaltet wird. Zum Inhalt der Zeichnung gehören Ansichten, Detail- und Schnittdefinitionen ebenso wie Komponentenlisten aller dargestellten Teile etc. Die Zeichnungsdefinition besteht intern selbst aus einer mehrstufigen Hierarchie verschiedener Instanz- und Modelleinheiten. Die Zeichnungsdefinition und mit ihr alle Untereinheiten stehen zu ihrem Besitzer in integrativer Beziehung. Das heißt, daß die gesamte Cluster-Hierarchie, die eine Zeichnung beschreibt, zusammen mit den Modelldaten des Teils oder der Baugruppe verwaltet wird, die Besitzer der Zeichnung ist. Daraus folgt unmittelbar, daß auch bei mehrfacher Verwendung dieser Teile die Zeichnungsdefinition nur einmal existiert.

Koordinatensysteme repräsentieren den Ursprung von Teilen oder Baugruppen. Sie legen einen ausgezeichneten 3D-Punkt sowie die Richtungen der *x*-, *y*-, *z*-Achse für das jeweilige Objekt fest. Alle Referenzen innerhalb des Objekts und auf das Objekt können so einfach auf dieses System bezogen werden. Koordinatensysteme stehen ebenfalls mit ihrem Besitzer in integrativer Beziehung.

Zusammenfassend soll noch einmal der Bogen zum Cluster-Konzept gespannt werden. Die vorangegangenen Abschnitte haben gezeigt, daß die logischen Einheiten sowohl der Instanz- als auch der Modelldaten alle Anforderungen an einen Cluster erfüllen. Mit der Definition eines Wurzelknotens, intern durch eine spezielle Baugruppe dargestellt, lassen sich alle Strukturelemente zu einer einzigen Hierarchie zusammenfassen. Aufgrund ihrer Eigenschaft als gerichteter Graph ist sichergestellt, daß jedes Element vom Wurzelknoten aus erreichbar ist.

Durch eine vollständige Aufteilung des Entity-Netzes in Instanz- und Modelleinheiten wird auch der Forderung des Cluster-Konzeptes Rechnung getragen, daß jedes Entity eindeutig einem Cluster zuzuordnen sei. Insoweit kann durch die Einführung von Instanz- und Modelleinheiten eine totale Strukturierung des abstrakten Entity-Netzwerks erreicht werden, die die Begriffe aus der Welt der Anwender umfassend reflektiert.

2.3.4 Persistenz von Strukturelementen

Das Cluster-Konzept geht auf die frühen Anfänge von SolidDesigner, dem Vorläufer von OneSpace Designer Modeling, zu Beginn der 90er Jahre zurück. Zu dieser Zeit war es weitestgehend üblich, CAD-Daten in Form von Dateien auf einem Dateisystem zu speichern. Datenbankintegrationen gab es zwar schon, ihre Verbreitung war aber noch relativ gering.

Daher war es erforderlich, bezüglich des Inhalts der Dateien einen Kompromiß zu finden, der einerseits genügend Flexibilität zuließ, Daten wiederzuverwenden und Redundanz zu vermeiden, andererseits die Zahl der Dateien in einer Grenze hielt, die noch handhabbar ist. So ist sicher das Speichern eines jeden Clusters in eine eigene Datei höchst flexibel, würde aber die Anzahl der Dateien schier sprengen.

In Abschnitt 2.3.2 wurde beschrieben, daß die Modelldaten einer Baugruppe im wesentlichen durch die in der Baugruppe enthaltenen Teile bestimmt sind. Die dort verwalteten Instanzen legen unter anderem die Bezeichnung und Position innerhalb der – das heißt relativ zur – Baugruppe fest. Es ist daher naheliegend, diese Instanzen auch zusammen mit den Modelldaten der Baugruppe zu verwalten. Die Flexibilität bleibt erhalten, da die Modelldaten der referenzierten Teile eigenständig bleiben und somit in anderen Baugruppen wiederverwendet werden können – dort wiederum mit anderen Instanzen, die in die entsprechende Baugruppe eingehen.

Entsprechend auf das Schema der Instanz- und Modelldaten verallgemeinert heißt dies, daß der Cluster der Instanzdaten zum Cluster der übergeordneten Modelldaten in integrativer Beziehung steht. Da der gemeinsame Wurzelknoten der Modellstruktur eine rein interne Verwaltungsfunktion hat, kann dieser nicht abgespeichert werden. Eine integrative Beziehung zu den Modelldaten des Wurzelknotens wäre hier nicht zielführend. Somit bedarf es für die Instanzen der obersten Ebene einer Sonderregelung. Sie bleiben als Cluster eigenständig.

Da jeder Cluster mit der Fähigkeit ausgestattet ist, seine Daten persistent zu machen, ergeben sich Dateigrenzen, wie sie in **Abbildung 2.13** dargestellt sind. Die Baugruppen 1 und 2 beinhalten Instanzen desselben Teils, das heißt sie referenzieren dieselbe Teiledatei.

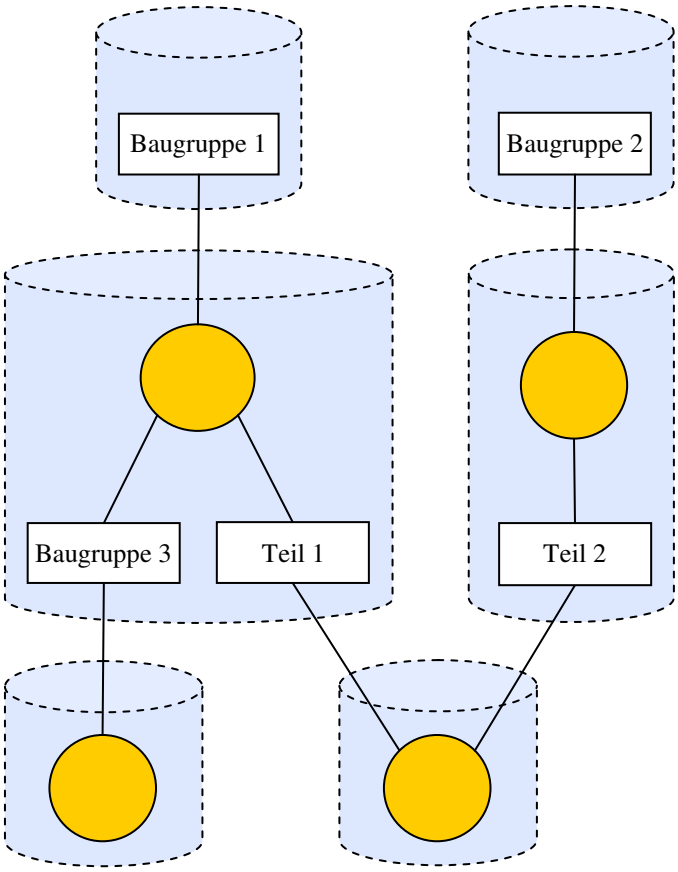


Abbildung 2.13: Dateigrenzen

2.4 Technologische Produktdaten

In den vorhergehenden Abschnitten wurden Datenstrukturen betrachtet, die der Beschreibung des geometrischen Modells eines einzelnen Bauteils dienen sowie der Strukturierung des gesamten CAD-Modells mit Hilfe von Baugruppen, die sich ihrerseits aus Bauteilen zusammensetzen. In der Praxis geht das Produktmodell aber weit über diese Abbildung eines realen Produkts in rechnerdarstellbare Strukturen hinaus.

Vielmehr wird dieses Modell um eine Vielzahl von Informationen ergänzt, die im Zusammenhang mit Applikationen oder Prozessen stehen, die zur Unterstützung des eigentlichen Konstruktionsprozesses oder nach dessen Abschluß zur Anwendung kommen. Nach [SpKr84] werden diese Informationen durch die *technologischen Produktdaten* dargestellt, die in bezug auf ihre Herkunft auch teilweise als *Applikationsdaten* bezeichnet werden. Aufgrund ihrer Komplexität erfordert diese technische

Beschreibung des Produkts mehr als das einfache Sammeln von Daten. Diese bedarf ebenso einer Strukturierung dieser Daten und deren sinnvoller Verknüpfung mit dem geometrischen Modell.

Die Informationen zur technischen Beschreibung des Modells lassen sich hauptsächlich nach ihren Anwendungsbereichen unterscheiden (vgl. [Hor97]). Zur Erzeugung von Geometrie werden oft *Konstruktionselemente* benutzt, die die Art und Lage von Hilfsgeometrien beschreiben sowie Vorschriften definieren, wie diese mit dem Modell zu verknüpfen sind. Zur Unterstützung von Prozessen, die der Fertigung bzw. Arbeitsvorbereitung dienen, werden *Fertigungselemente* verwendet. Elemente zur technischen Beschreibung werden allgemein unter dem Begriff *Features* zusammengefaßt.

2.4.1 Features

Eine prinzipielle Definition des Features wird von der Arbeitsgruppe FEMEX (Feature Modeling Experts) gegeben[8]. Demnach ist ein Feature eine Informationseinheit, die einen signifikanten Bereich innerhalb eines Produkts beschreibt. Das Feature wird als eine Zusammenfassung von Eigenschaften eines Produkts beschrieben. Diese Zusammenfassung beinhaltet alle maßgeblichen Eigenschaften einschließlich ihrer Werte und ihrer Beziehungen. Ein Feature ist jeweils für eine spezifische Sicht auf die Produktbeschreibung unter Berücksichtigung der möglichen Eigenschaften und der Phasen des Produktlebenszyklus definiert.

Features, die für Konstruktionszwecke verwendet werden, enthalten in der Regel Daten, die sehr speziell für das jeweilige CAD-System sind. Aus diesem Grund sollen sie hier nicht weiter verfolgt werden. Fertigungselemente zeichnen sich durch die Möglichkeit zur Identifizierung geometrischer Elemente einerseits und der Zuordnung von Fertigungsinformationen andererseits aus. Mit Hilfe dieser Features können Informationen über technologische Teilbereiche eines Produkts im jeweiligen Prozeßschritt strukturiert beschrieben werden. Verweise auf konkrete Geometrieelemente sind dabei je nach Art der Anwendung nicht notwendigerweise erforder-

[8] Vgl. [Hor97] / FEMEX WG1 „Feature Definition and Classification“: What is a feature and what is its use?

lich, statt dessen können die Features auch allgemeine technologische Informationen, wie etwa Verfahrensangaben, tragen. Damit kann eine Verbindung sowohl abstrakter als auch konkreter technologischer Informationen mit dem Modell erreicht werden. Da in der Regel innerhalb der Prozeßkette unterschiedliche Systeme zum Einsatz kommen, die Fertigungsinformationen verarbeiten, besteht die Forderung, diese Daten mit anderen Systemen im Rahmen des Produktmodells austauschen zu können.

Für den Einsatz von Fertigungsfeatures kommen verschiedene Anwendungsbereiche in Betracht, von denen einige nachfolgend exemplarisch umrissen werden sollen. Ziel der Arbeitsplanung ist es, eine möglichst automatische Erzeugung von kompletten Arbeitsplänen zu erreichen. Fertigungsfeatures dienen hier als Träger von Informationen zur Auswahl der Bearbeitungsverfahren und der benötigten Werkzeuge. Diese können ergänzt werden durch Bearbeitungsinformationen und Regeln zur Verifikation der Herstellbarkeit.

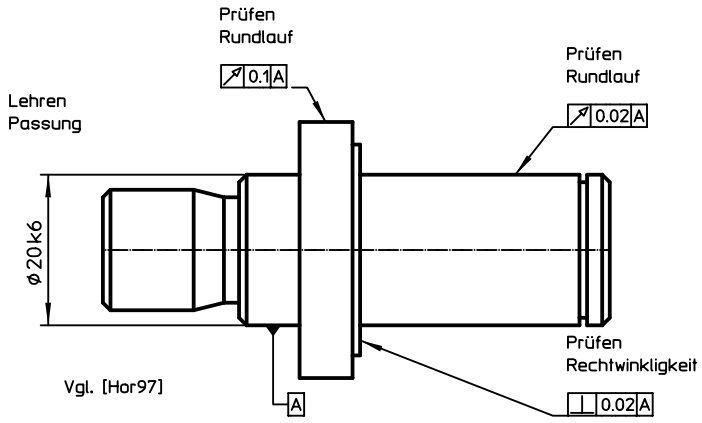


Abbildung 2.14: Toleranzinformationen

Im Rahmen der Qualitätssicherung werden Informationen über die zu prüfenden Flächen und deren Toleranzen durch Features dargestellt (vgl. **Abbildung 2.14**). Dazu werden weitere geometrische Bezugselemente benötigt, durch die die Toleranzinformationen einen eindeutigen Bezugspunkt erhalten. Bei der Montage- und Prozeßplanung beinhalten Features beispielsweise Informationen zur Auswahl von Vorrichtungen durch Kennzeichnung der Lage von Einspann- oder Aufnahmepunkten. Diese Information wird wiederum genutzt zur Simulation von Fertigungs- oder Montageverfahren oder bei Plausibilitätsprüfungen.

Aus der großen Bandbreite der Verwendungsmöglichkeiten von Features ergeben sich einige Anforderungen an deren Datenstruktur. So müssen die Eigenschaften der Features ebenso konkrete Werte annehmen wie abstrakte Informationen darstellen können. Weiterhin muß es möglich sein, Bezüge zu anderen Features aufzubauen. Darüber hinaus müssen auch Bezüge zu Geometrieelementen hergestellt werden können. Dies setzt die Existenz eines vollständigen Geometriemodells voraus, wie es mit dem B-Rep-Modell in Kapitel 2.2.2 beschrieben wurde. Ein solches Modell ermöglicht überdies die Darstellung und Erzeugung von Flächen, Kanten und Schnittkurven mit einer Genauigkeit, wie sie zum Beispiel zum Ableiten von Maßen und zur Angabe von Toleranzen erforderlich ist. Eine weitere wesentliche Anforderung ist die Assoziativität der Features mit dem Geometriemodell. Dies bedeutet, daß Änderungen an der Geometrie mit einer automatischen Aktualisierung der Applikationsdaten einhergehen.

In der Regel ist die Featuredatenstruktur eng mit den Datenstrukturen des geometrischen Modellierkerns verbunden. Bei featurebasierten 3D-CAD-Systemen werden Fertigungsfeatures meistens als Bestandteil des Feature-Baums verwaltet, der über Boole'sche Verknüpfungen von (Hilfs-) Körpern und Konstruktionsfeatures das geometrische Modell definiert. Entscheidend dabei ist, daß Bezüge zwischen der Feature-Datenstruktur und dem Geometriemodell bestehen, die eine nachträgliche Manipulation des Featuremodells ohne Konsistenzverlust ermöglichen.

Features, die sich explizit auf Geometrieelemente beziehen, lassen sich nicht immer eindeutig in einem reinen Featuremodell verankern. Dies gilt ebenso für Konstruktionsfeatures wie „*Fase*“ oder „*Verrunden*“ wie für viele Fertigungsfeatures. Die Geometrieelemente im Modell, auf die solche Features verweisen, entstehen teilweise erst durch die Verknüpfung anderer Features. Bei Veränderungen des Featuremodells müssen diese Abhängigkeiten mit hohem Aufwand überwacht werden. Nachfolgend soll daher eine Datenstruktur beschrieben werden, die technologische Produktdaten direkt mit dem B-Rep-Modell verknüpft.

2.4.2 Explizite Darstellung

CAD-Systeme, deren geometrischer Modellierkern auf dem B-Rep-Modell aufbaut, haben die Möglichkeit, auf jedes geometrische Element jederzeit direkt zugreifen zu können. Da eine wiederholte Evaluierung des Modells nicht erforderlich ist, ist eine eindeutige Identifizierung der Flächen, Kanten etc. einfach zu erreichen. Überdies bleibt diese Identifizierung auch über Arbeitssitzungen hinweg gültig. Diese Eigenschaft wird bei der expliziten Darstellung von technologischen Produktdaten bzw. Applikationsdaten mit Hilfe einer Datenstruktur ausgenutzt, die direkt mit den Elementen des B-Rep-Modells verknüpft ist. Exemplarisch soll hier die in OneSpace Designer Modeling verwendete Implementierung einer solchen Struktur beschrieben werden, die auch als Basis für die Realisierung von Fertigungsfeatures dient.

Den Betrachtungen in Kapitel 2.1 folgend werden alle beteiligten Elemente in Form von Entities in das Entity-Netzwerk des CAD-Modells integriert. Eine zentrale Rolle in bezug auf die Darstellung von Informationen nimmt die Entity-Klasse *FC_BASE* ein, die als abstrakte Basisklasse für alle weiteren Feature-Typen fungiert. Abhängig von der Art der darzustellenden Information und vom Typ des Fertigungsfeatures stehen abgeleitete Klassen zur Verfügung, die adäquate Möglichkeiten zur Verarbeitung der Applikationsdaten bereitstellen.

Ein solches Feature-Objekt ist bidirektional mit seinem Besitzer verknüpft. Dies kann neben einem topologischen Objekt – also einer Fläche, Kante, einem Eckpunkt oder dem Body selbst – auch ein Teil oder eine Baugruppe sein. Eine wesentliche Eigenschaft der Feature-Objekte ist die Fähigkeit, auf Modelländerungen definiert und eigenständig reagieren zu können. Dazu überwachen sie eine Vielzahl von Ereignissen, die Einfluß auf ihre Darstellung bzw. Konsistenz haben könnten. Wird beispielsweise die Fläche, die der Besitzer eines Features ist, durch eine Modelleroperation geteilt, so wird das Feature darüber informiert und entscheidet, wie es sich verhält. Je nach Feature-Typ kann es sich löschen, sich als ungültig markieren, den Besitzer wechseln, sich an beide Flächenteile hängen, das Ereignis ignorieren etc. Dieses Verhalten ist pro Feature einstellbar und ermöglicht gleichzeitig ein Höchstmaß an Assoziativität zum geometrischen Modell.

Um eine benutzergerechte Verwendung von Features zu erreichen, können diese mit Namen versehen oder auch mit Hilfe von Labels im graphischen Modell visualisiert werden. Diese Sekundärdaten werden wiederum auf der Basis von Entities implementiert, die innerhalb des Entity-Netzwerks mit dem Feature-Objekt in Relation stehen. Persistenz dieser Daten wird dadurch erreicht, daß alle Entities, die letztlich das Feature definieren, über ihre `local_master()`[9] Methode direkt oder indirekt auf das Feature-Objekt verweisen, das seinerseits auf seinen Besitzer zeigt. Dies hat zur Folge, daß alle beteiligten Entities mit dem Besitzer des Features zusammen in eine gemeinsame Datei geschrieben werden.

Der Verweis von einem Feature-Objekt auf andere Features bzw. die direkte Referenzierung von Flächen, Kanten und Eckpunkten wird durch bidirektionale Beziehungen zu den jeweiligen Objekten realisiert. Dieses Verfahren ist allerdings auf Objekte beschränkt, die alle in demselben Cluster leben. Sobald die Referenzen über Cluster-Grenzen hinausgehen – das heißt, innerhalb einer Baugruppe teileübergreifend verwendet werden – reicht dieses Vorgehen nicht mehr aus, da jetzt weitere Ereignisse hinzukommen können, die das Feature nicht mehr in der Lage ist, eigenständig zu überwachen. Dies können zum Beispiel Änderungen an der Baugruppenstruktur oder an der Transformation einzelner Teile sein.

Dementsprechend werden zur Abbildung der Referenzdaten zwei weitere Hilfsklassen definiert. Die Referenz selbst wird durch ein *REF_PATH*-Entity dargestellt, das eine bidirektionale Beziehung mit dem Zielelement eingeht. Stellvertretend für das Feature-Objekt überwacht dieses Entity alle Ereignisse, die das Zielobjekt betreffen und löst im Falle einer Benachrichtigung entsprechende Aktionen beim Feature aus. Der Pfad – also alle Teile und Baugruppen auf dem direkten Weg vom Besitzer des Features zu seiner Referenz – wird vom *CLUSTER_PATH*-Entity überwacht, das einerseits mit jedem Pfadelement in Verbindung steht, andererseits als Bindeglied zwischen Feature und Referenz-Objekt auftritt. Auch hier lösen alle Ereignisse, die eines der Pfad-Elemente betreffen, entsprechende Aktionen am CLUSTER_PATH-Entity selbst oder am Feature aus. Schließlich dokumentieren auch diese Entities mit Hilfe ihrer `local_master()` Methode ihre Zugehörigkeit zum Feature.

---

[9] Vgl. Kapitel 2.1.2 Cluster-Konzept

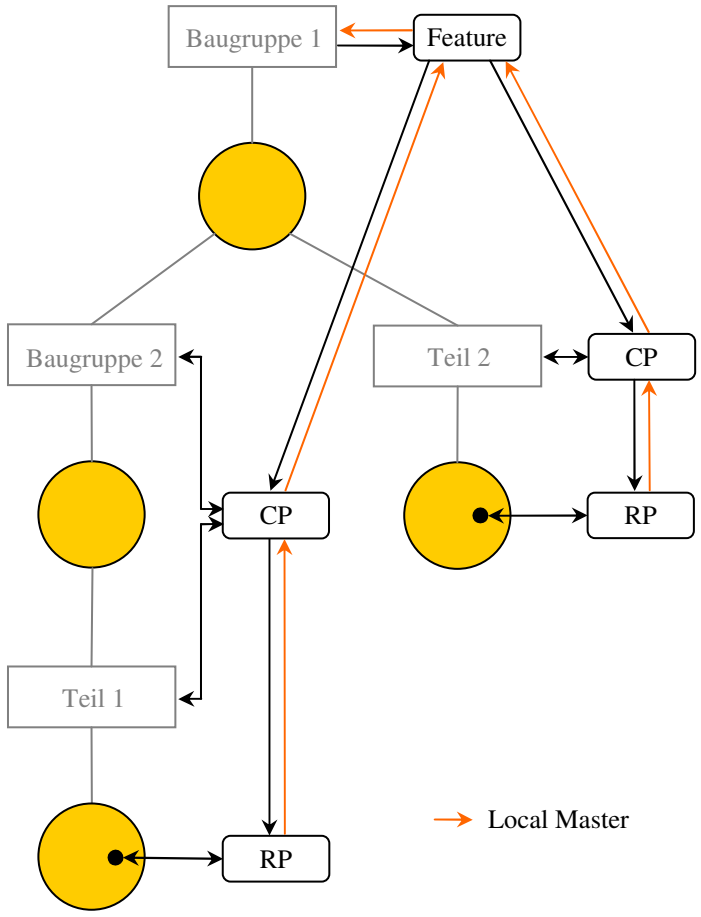


Abbildung 2.15: Datenstruktur eines Features mit teileübergreifenden Referenzen

Die Datenstruktur eines Features mit teileübergreifenden Referenzen ist in **Abbildung 2.15** dargestellt. Besitzer des Features ist Baugruppe 1. Das Feature verweist auf zwei Elemente innerhalb der geometrischen Modelle von Teil 1 und Teil 2, jeweils symbolisiert durch einen Punkt (●), über die Pfade „Baugruppe 2/Teil 1“ bzw. „Teil 2“. Die Abbildung verdeutlicht die Verbindungen zwischen den Entities des Features und allen Elementen des Modells, die durch Modellieroperationen verändert werden können. Entities der Klassen REF_PATH und CLUSTER_PATH werden abkürzend mit RP bzw. CP bezeichnet.

2.5 Technologien

In den bisherigen Abschnitten wurden Datenstrukturen betrachtet, die für den Aufbau eines CAD-Systems grundlegend sind. Ausgehend von der kleinsten Informationseinheit, dem Entity, wurde gezeigt, wie komplexe Netzwerke aufgebaut werden kön-

nen, die zentrale Modelleigenschaften wie das geometrische Modell, die Baugruppenstruktur und die technologische Produktbeschreibung abbilden. In diesem Abschnitt sollen einige Technologien und Methoden beschrieben werden, die bereits heute Verwendung finden, um die Handhabung großer Baugruppen zu erleichtern bzw. überhaupt zu ermöglichen. Der Schwerpunkt wird dabei auf solche Technologien gelegt, die auch für die weiteren Betrachtungen in dieser Arbeit von Bedeutung sind.

2.5.1 Partielles Laden

Spätestens wenn die Größe des CAD-Modells die Systemgrenzen der eingesetzten Rechner erreicht, stellt sich die Frage, wie die Datenmenge reduziert werden kann, so daß die zur Verfügung stehenden Ressourcen ausreichend sind, aber gleichzeitig ein sinnvolles Arbeiten weiterhin möglich ist. Der naheliegende Gedanke, die Konstruktion auf Einzelteile oder kleine Unterbaugruppen zu beschränken, greift dabei vielfach zu kurz, da eine adäquate Berücksichtigung der teile- bzw. baugruppenübergreifenden Informationen so nicht möglich ist. Demgemäß ist beispielsweise weder die Struktur des Gesamtmodells erkennbar, noch können eventuelle Kollisionen von Teilen ohne weiteres festgestellt werden.

Einen anderen Ansatz verfolgt das partielle Laden. Unter der Annahme, daß das geometrische Modell zusammen mit seiner graphischen Darstellung den weitaus überwiegenden Teil der Daten ausmacht, wird nur die Baugruppenstruktur des Gesamtmodells komplett geladen, während geometrische Information lediglich dort geladen wird, wo sie wirklich notwendig ist. Dabei ist nicht von Bedeutung zu welchen Baugruppen die Teile gehören, die vollständig geladen werden sollen. Weil bei dieser Methode die graphische Repräsentation des Modells noch vom geometrischen Modell abgeleitet – und nicht abgespeichert – wird, ist allerdings zu beachten, daß auch Teile, die ausschließlich für Visualisierungszwecke benötigt werden, vollständig zu laden sind.

Messungen an verschiedenen Modellen in [Schn02] haben als groben Richtwert tatsächlich ergeben, daß ca. 10 % der gesamten Modelldaten auf die Baugruppenstruktur entfallen, während das geometrische Modell und die graphische Darstellung

desselben jeweils ca. 45 % der Daten ausmachen. Durch gezieltes partielles Laden, das heißt, durch gezieltes Weglassen nicht benötigter Geometrie- und Graphikdaten, können bei gleichem Speicherausbau eines Rechners wesentlich größere Modelle bearbeitet werden. Trotzdem bleibt die Struktur des Gesamtmodells und die damit verbundene teile- und baugruppenübergreifende Information erhalten. Da sich Lage und Position aller vollständig geladenen Teile aufgrund der kompletten Baugruppenstruktur automatisch auf das Gesamtmodell beziehen, sind eventuelle Kollisionen erkennbar, auch wenn die betreffenden Teile zu verschiedenen Unterbaugruppen gehören.

Die technische Realisierung dieses Konzeptes erfolgt mittels dynamischer Erzeugung von Leerteilen während des Ladevorgangs. Die Auswahl der vollständig zu ladenden Teile erfolgt bereits auf der PDM-Seite und kann jederzeit durch Nachladen ergänzt werden. Die Modelldaten[10] der Komponenten, die der Benutzer vom Laden ausschließt, werden durch Modelldaten eines standardisierten Leerteils, also eines Teils ohne geometrische Daten, ersetzt. Dadurch wird auch die graphische Darstellung dieser Komponenten unterdrückt, die sonst von den geometrischen Modellen abgeleitet worden wäre. Die für einen solchen Platzhalter benötigte Datenmenge ist äußerst gering und fügt sich homogen in bestehende Datenstrukturen ein.

In Abschnitt 2.1.4 wurde beschrieben, daß jeder Cluster zum Zwecke der externen Referenzierung mit einer eindeutigen Kennung (SysID) versehen sein muß. Daher können nicht verschiedene Teile durch dasselbe Leerteil ersetzt werden. Vielmehr muß jeder Platzhalter die SysID seines Ursprungsteils erben. Dadurch wird es möglich, Veränderungen an Baugruppen durchzuführen, die auf solche Leerteile verweisen, und diese auch wieder abzuspeichern. Durch das Vererben der SysIDs wird erreicht, daß die Referenzen auf Leerteile identisch sind mit den entsprechenden Referenzen auf die Originalteile. Das bedeutet, daß nach dem Abspeichern der Baugruppe diese später wieder zusammen mit ihren Originalteilen geladen werden kann, was offensichtlich eine notwendige Bedingung ist. Die Leerteile selbst sind jedoch gegen jegliche Veränderungen durch den Benutzer geschützt und können nicht abgespeichert werden, um ein Überschreiben der Originaldaten zu verhindern.

---

[10] Vgl. Kapitel 2.3.1 Teile

2.5.2 Open Reference Handling

Externe Referenzen, also Verweise von einem Cluster in einen anderen, finden Verwendung beim Aufbau der Baugruppenstruktur, ebenso bei der Darstellung teileübergreifender technologischer Produktdaten. Dabei wird unterschieden zwischen Referenzen *auf* einen Cluster und solche *in* einen Cluster. Erstere verweisen auf den Einstiegspunkt des Clusters – den Cluster Master – und sind kennzeichnend für die Beziehungen der Strukturelemente untereinander in der Baugruppenstruktur. Letztere sind dagegen Verweise auf Entities innerhalb des Clusters und typisch für die Beschreibung der technologischen Produktinformation. Die Distanz zwischen zwei Flächen benachbarter Teile veranschaulicht dies an einem Beispiel.

Im allgemeinen kann nicht davon ausgegangen werden, daß alle Entities, auf die von außerhalb des Clusters verwiesen wird, nach einem Ladevorgang auch tatsächlich verfügbar sind. Beim partiellen Laden beispielsweise werden Teile durch Leerteile ersetzt, das heißt, Referenzen auf Flächen oder Kanten des eigentlichen Modells können nicht aufgelöst werden, sie zeigen quasi ins Leere. Dasselbe gilt auch für den Austausch von Teileversionen beim Laden – eine Referenz auf eine Bohrung innerhalb eines Teils, das jetzt in der Version ohne Bohrung geladen wird, kann nicht mehr aufgelöst werden – oder für Veränderungen an der Baugruppenstruktur, die ohne direkte Beteiligung der Produktdaten vollzogen wurden.

Ist das Ziel einer externen Referenz nicht verfügbar, muß auf jeden Fall reagiert werden. Im einfachsten Fall wird der Verweis auf *NULL* gesetzt, um dem Anwendungsprogramm das Fehlen des Ziel-Entitys anzuzeigen und ihm die Möglichkeit zu geben, geeignet zu reagieren. Dieses Verfahren hat allerdings eine gravierende Schwäche. Muß der Cluster, aus dem der Verweis stammt, wegen einer anderweitigen Änderung abgespeichert werden, so wird der *NULL*-Verweis dadurch persistent. Demzufolge sind die ursprünglichen Daten verlorengegangen. Selbst wenn die Baugruppe danach wieder vollständig geladen wird – das Ziel-Entity also wieder verfügbar ist –, kann die Referenz darauf nicht wieder rekonstruiert werden. Um diesen Datenverlust zu verhindern, muß der Cluster, aus dem eine Referenz stammt, die während des Ladens nicht aufgelöst werden kann, gegen jede Veränderung durch den Benutzer geschützt werden. Für den Konstrukteur bedeutet dies jedoch eine starke Einschränkung.

Beim partiellen Laden werden die Referenzdaten innerhalb der Baugruppenstruktur dadurch erhalten, daß die Leerteile, die die ursprünglichen Teile ersetzen, deren SysID erben. Die Kennung der Zielobjekte bleibt auf diese Weise unverändert, der Verweis auf ein Leerteil ist somit von einem Verweis auf das entsprechende Originalteil nicht zu unterscheiden.

Open Reference Handling basiert auf einem ähnlichen Ansatz, der in [Schn01b] detailliert beschrieben ist und dessen Prinzip hier erläutert werden soll, geht aber weit über die Fähigkeit hinaus, nur Referenzdaten zwischenspeichern zu können. So können Referenzen vor dem Entfernen ihrer Ziel-Entities, etwa im Rahmen einer Modellvereinfachung, gesichert werden. Ebenso ist es möglich, diese mit ihrem Ziel automatisch wiederzuverbinden, sobald es wieder verfügbar wird, beispielsweise durch eine Nachladeoperation.

Jedes extern referenzierte Entity, das während einer Ladeoperation nicht zur Verfügung steht, wird durch einen Platzhalter, ein OPEN_REF_ENTITY, ersetzt. Dessen maßgebliche Aufgabe ist es, Referenzdaten zu puffern, die aus dem Verweis auf das fehlende Entity bekannt sind und zur Wiederherstellung der Referenz benötigt werden. Dazu zählen die SysID des Ziel-Entitys sowie die Kennung des Clusters, zu dem es gehört. Mehrfache Verweise auf ein und dasselbe nicht verfügbare Ziel werden dabei einem gemeinsamen Platzhalter zugeordnet. Jeder der Verweise, die jetzt auf das OPEN_REF_ENTITY zeigen, stammt aus einem Entity, das gleichsam der Besitzer der Referenz ist. Das OPEN_REF_ENTITY protokolliert alle Besitzer der Referenzen, die auf diesen Platzhalter zeigen, und ist dadurch über bidirektionale Verbindungen in das Entity-Netzwerk integriert.

Da das OPEN_REF_ENTITY nur ein formaler Platzhalter für ein momentan nicht verfügbares Entity ist, wird ein Verweis auf ein solches als *offene Referenz* bezeichnet. Das Vorgehen bei der Substitution durch OPEN_REF_ENTITYs ist absolut generisch. Es ist daher unerheblich, aus welchen Bereichen oder Applikationen die Entities mit den Verweisen stammen und von welchem Typ die Ziel-Entities sind. Das Erkennen einer offenen Referenz und die geeignete Reaktion darauf ist hingegen Aufgabe der jeweiligen Anwendungen.

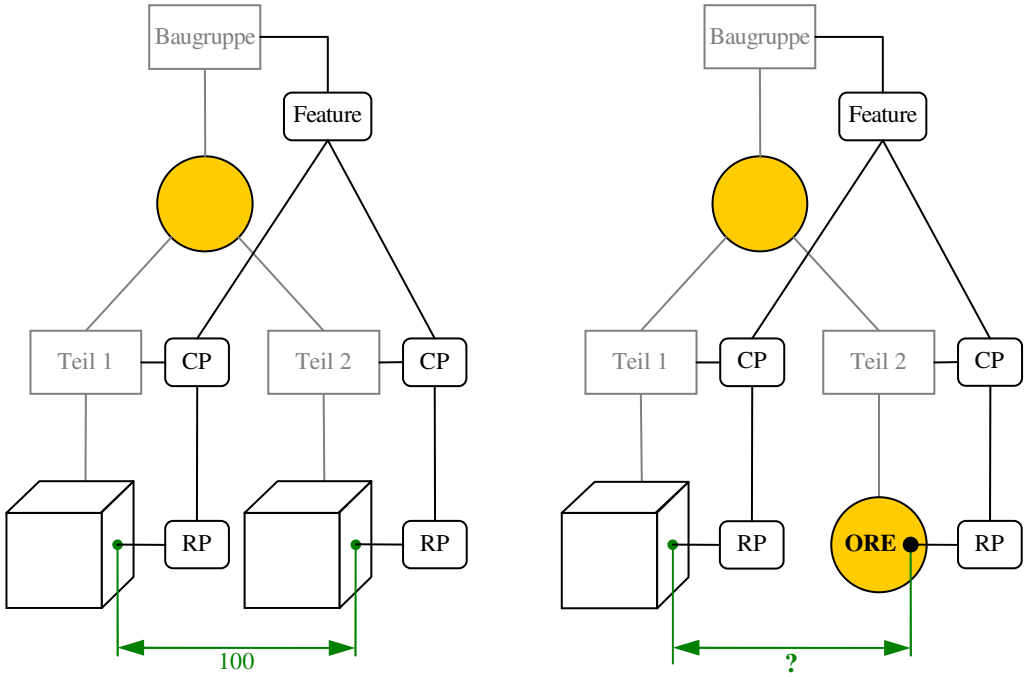


Abbildung 2.16: Bemaßung mit vollständigem Modell und mit offener Referenz

Die Integration eines OPEN_REF_ENTITYs wird in **Abbildung 2.16** illustriert. Auf der linken Seite ist ein vollständig geladenes Modell dargestellt. Beide referenzierten Flächen sind vorhanden, das Maß kann gezeichnet werden. Auf der rechten Seite dagegen wird das Teil 2 partiell geladen. Die referenzierte Fläche steht nicht zur Verfügung und wird durch ein OPEN_REF_ENTITY (ORE) ersetzt. Es ist nun Aufgabe der Applikation, hier des Bemaßungsmoduls, diese Situation zu erkennen und zu handhaben. Eine mögliche Reaktion in diesem Falle wäre, das Maß nicht anzuzeigen, da es nicht vollständig bestimmt ist.

In Abschnitt 2.1.4 wurde beschrieben, daß beim Abspeichern einer externen Referenz neben der erweiterten Kennung des Entities, auf das verwiesen wird, auch die SysID des Clusters, in dem sich dieses Entity befindet, berücksichtigt wird. Diese Daten werden durch Befragen des Ziel-Entities gewonnen. Beim Erzeugen eines Platzhalter-Entities werden genau diese Referenzdaten an das OPEN_REF_ENTITY weitergegeben. Daher kann der Algorithmus zum Sequentialisieren eines Verweises so verändert werden, daß er die benötigten Daten im Falle einer offenen Referenz durch entsprechendes Befragen des Platzhalters gewinnt, der nicht seine eigenen Daten liefert, sondern die des Entities, für das er steht. Somit ist sichergestellt, daß

eine gespeicherte Referenz auf ein existierendes Entity mit der auf das entsprechende OPEN_REF_ENTITY identisch ist. Dies ermöglicht das uneingeschränkte Arbeiten mit einem Modell, das offene Referenzen hat.

Einige Anwendungen erfordern auch, aktiv Referenzen auf Entities zu öffnen, die danach entfernt werden sollen. Dadurch ist es möglich, Verweise zumindest temporär offenzuhalten. Ein Beispiel einer solchen Anwendung ist die Modellsimplifikation. Bevor ein Modell vereinfacht wird, werden alle externen Referenzen auf das Modell geöffnet. Danach können Entities entfernt werden, die im vereinfachten Modell nicht mehr notwendig sind, zum Beispiel Flächen von Bohrungen oder Rippen, ohne daß Verweise darauf verlorengehen. Wird das vereinfachte Modell später wieder gegen eine detaillierte Variante ausgetauscht, die die entsprechenden Flächen enthält, werden die Referenzen automatisch reaktiviert und die zugehörigen Daten können vollständig dargestellt werden. Ähnliches gilt für das Nachladen von Teilen bzw. den Austausch von Versionen. In beiden Fällen wird – nach Öffnen aller externen Referenzen – der auszutauschende Teil des Modells entfernt und durch ein entsprechendes Äquivalent ersetzt. Nach Abschluß der Operation wird automatisch jede offene Referenz daraufhin überprüft, ob ein Wiederverbinden möglich ist.

Das Öffnen von Referenzen wird realisiert, indem der Teil des Modells – das heißt, der Teil des Entity-Netzwerks –, der (potentiell) entfernt werden soll, traversiert und für jeden Knoten die Methode `disconnect_ref()` aufgerufen wird. Diese prüft, ob das Entity Ziel externer Referenzen ist, und ruft für jeden Besitzer einer solchen Referenz die Methode `disconnect()` auf. Entscheidet sich der Besitzer, seinen Verweis auf das Entity zu öffnen, erzeugt dieser unter Verwendung der Referenzdaten ein entsprechendes OPEN_REF_ENTITY bzw. bestimmt ein bereits existierendes zur Wiederverwendung, löst seine Verbindung zum Ziel-Entity und verweist statt dessen auf den Platzhalter. Gleichzeitig hat der Besitzer die Möglichkeit, eine weiterreichende Reaktion auf die Änderung der Referenz zu veranlassen, etwa das Unterdrücken der Anzeige eines Maßes.

Im Anschluß an eine Ladeoperation wird für alle neuen Entities geprüft, ob Platzhalter mit identischen Schlüsseldaten, das heißt gleiche SysID im gleichen Cluster, existieren. Im Falle der Modellsimplifikation werden alle Entities des vereinfachten

Modells in diesem Sinne als neu betrachtet, da vor der Vereinfachung alle externen Referenzen präventiv geöffnet werden. Wird ein passendes OPEN_REF_ENTITY gefunden, so werden alle Besitzer von Verweisen auf diesen Platzhalter durchlaufen und für jeden von ihnen wird die Methode `reconnect()` aufgerufen. Diese hat die Aufgabe, die Verbindung zum OPEN_REF_ENTITY zu lösen und den Verweis wieder auf das reale Entity zu setzen. Derart über die Veränderung der Referenz informiert, kann auch hier eine adäquate Reaktion ausgelöst werden, etwa die Aktualisierung und Wiederanzeige eines Maßes. Mit Trennen der letzten Verbindung wird das OPEN_REF_ENTITY nicht mehr benötigt und daher automatisch gelöscht.

Zusammenfassend bietet Open Reference Handling die Möglichkeit, Referenzen auf temporär nicht erreichbare Entities zu puffern und diese nach Verfügbarwerden automatisch wiederzuverbinden. Währenddessen kann der Konstrukteur ohne Einschränkung weiterarbeiten. Überdies kann Funktionalität zum dynamischen Austausch verschiedener Teileversionen einschließlich Modellvereinfachungen angeboten werden, bei der sichergestellt ist, daß technologische Produktinformationen immer in genau dem Umfang zur Verfügung stehen, die dem aktuellen Status des Modells entsprichen. Die Aktualisierung der Daten erfolgt dabei objektorientiert in direkter Reaktion auf entsprechende Operationen, ohne daß es einer erneuten Evaluierung des Modells bedarf.

2.5.3 64-Bit-Adressierung

Der Adreßraum eines Prozessors ist der Wertebereich der Adressen von Speicherzellen, auf die der Prozessor zugreifen kann, um dorthin Daten zu schreiben bzw. sie von dort zu lesen. Die theoretisch maximale Größe dieses Wertebereichs ist durch die Breite der Register festgelegt, die zur Adressierung der einzelnen Speicherzellen eingesetzt werden. Bei der bislang weitverbreiteten Verwendung von 32-Bit-Registern ergibt sich daraus ein Bereich von 2^{32} Adressen, was 4 GigaByte (GB) Daten entspricht.

Der *physische Adreßraum* entspricht dem Bereich aller Adressen, die direkt auf einzelne Zellen in Speicherbausteinen abgebildet werden können. Der *virtuelle Adreßraum* ist der Wertebereich eines simulierten Speichers, der größer ist als der tatsäch-

lich vorhandene. Während der Programmlaufzeit übernimmt das Betriebssystem die Aufgabe, zum Lesen oder Schreiben benötigte Teilbereiche des Speichers in den physischen Speicher zu kopieren, um so den Zugriff zu ermöglichen. Dies geschieht im Austausch mit einem in diesem Moment unbenutzten Teilbereich des Speichers, der zu diesem Zweck ausgelagert wird.

Bei der Verwendung der 32-Bit-Windows-Betriebssysteme wird die theoretisch adressierbare Datenmenge von 4 GB nicht erreicht, da einer Applikation in der Regel nur die Hälfte dieses Bereichs für Daten zur Verfügung gestellt wird. Die andere Hälfte reserviert Windows für den eigenen Kernel. Durch Verwendung spezieller Betriebssystemversionen bzw. Konfigurationsparameter[11] kann diese Limitation zumindest teilweise aufgehoben werden. Für einige Anwendungen, wie etwa die Konstruktion großer Baugruppen in CAD-Systemen, ist aber auch dies noch nicht ausreichend. Hier können deutlich mehr als 4 GB Daten erforderlich sein.

Durch Verdopplung der Bus- und Registerbreite in den Prozessorchips von 32 auf 64 Bit konnte der Adreßbereich immens vergrößert werden. Damit kann jetzt theoretisch auf maximal 2^{64} Adressen zugegriffen werden, was einer Datenmenge von 16 Exa-Byte (EB) entspricht – das sind ca. 18 Milliarden Gigabyte. Die Weiterentwicklung der Hardware allein reicht aber nicht aus, um diese Möglichkeiten auch nutzen zu können. Vielmehr muß die 64-Bit-Technologie auch durchgängig im Betriebssystem und in den Anwendungsprogrammen selbst integriert sein. Mit Windows XP Professional x64 steht, neben LINUX, ein Betriebssystem zur Verfügung, das eine 64-Bit-Plattform auf PC-Basis bereitstellt. Der theoretisch mögliche Adreßraum ist hierin auf 1 TeraByte (TB) physischen Speicher und 8 TB virtuellen Speicher[12] je Prozeß beschränkt, was – zumindest für CAD-Anwendungen – momentan nicht als echte Einschränkung zu betrachten ist.

Die Portierung bestehender 32-Bit-Anwendungen auf 64-Bit gestaltet sich in der Praxis weit schwieriger, als dies auf den ersten Blick erscheinen mag. Formal ist nur eine Neukompilierung notwendig, um den x64-Befehlssatz des Prozessors auszu-

---

[11] Windows 2000 / XP, Server-Edition, Option /3GB, Physical Address Extension (Option /PAE) bei Verwendung entsprechender Hardware

[12] Gemäß Microsoft Platform SDK (VC++ Manuals)

nutzen und entsprechend statt vorher 32 Bit jetzt 64 Bit für die Darstellung einer Adresse bereitzustellen. Tatsächlich existieren in der Regel aber eine Vielzahl von Code-Stellen, an denen implizite Annahmen über die Größe von Datentypen gemacht wurden, die nach der Portierung nicht mehr gültig sind. Prominentes Beispiel ist in diesem Zusammenhang die Größengleichheit einer Adresse und einer Ganzzahl in der 32-Bit-Architektur, die unter 64-Bit nicht mehr gegeben ist. Zuweisungen von Werten des einen Typs in eine Variable des anderen können daher nach der Portierung zu Fehlern führen.

Bei Anwendungsprogrammen, deren Quellcode für mehrere Plattformen übersetzt wird, kommt als weitere Erschwernis hinzu, daß die Festlegung der Größe von Datentypen plattformübergreifend nicht einheitlich durchgeführt wurde. Auch wenn einige Werkzeuge und Compiler-Optionen zur Verfügung stehen, die die Portierung unterstützen, ist die geforderte Qualität beim Übergang auf die 64-Bit-Architektur nur durch hohen Testaufwand und ausgefeilte Testmethoden sicherzustellen.

Die Datenbreite sagt letztlich nur etwas darüber aus, wieviele Bits ein Prozessor in einem Arbeitsschritt verarbeiten kann und wie breit seine Arbeitsregister angelegt sind. Das bedeutet, daß der eigentliche Vorteil der 64-Bit Architektur die Möglichkeit ist, einen größeren Speicherbereich adressieren zu können. Damit verbunden ist aber auch, daß für die Darstellung der Adressen selbst mehr Speicher benötigt wird – was den Vorteil schmälert, ihn aber bei weitem nicht aufheben kann. Bei Modellen, zu deren Darstellung ein hochkomplexes Entity-Netzwerk mit einer großen Anzahl an Verweisen auf Nachbar-Entities erforderlich ist, kann dieser Mehrbedarf bis zu 50 % der Modellgröße[13] ausmachen.

[13] CoCreate interne Messungen mit OneSpace Designer Modeling

3 Anforderungen für die Handhabung großer Baugruppen

Mit fortwährendem Streben nach höherem Kundennutzen, kürzeren Entwicklungszeiten und kostengünstigerer Fertigung und Wartung sind in den vergangenen Jahren die Anforderungen an die zu entwickelnden Produkte, wie auch an die Entwicklung der Produkte selbst, gestiegen. Gleichzeitig stehen immer umfangreichere technische Möglichkeiten zur Konstruktion und Entwicklung zur Verfügung. Die Funktionalität der CAD- und Berechnungsprogramme wird von Version zu Version ausgeweitet, die verwendeten Methoden werden ausgefeilter und auch die Hardware-Systeme stellen eine wachsende Verarbeitungsgeschwindigkeit und Speichergröße bereit.

Als Folge davon ist zu beobachten, daß CAD-Modelle zunehmend größer und komplexer werden sowie der Grad der Detaillierung steigt. Dies führt jedoch zu einer gewissen Rückkopplung, denn dadurch entstehen wiederum neue Ansprüche an die zur Entwicklung verwendeten Hardware- und Softwaresysteme. Dies trifft im speziellen auf einen effizienten Umgang mit den voluminösen Datenmengen zu, die zur Modellbeschreibung heute üblicherweise verwendet werden.

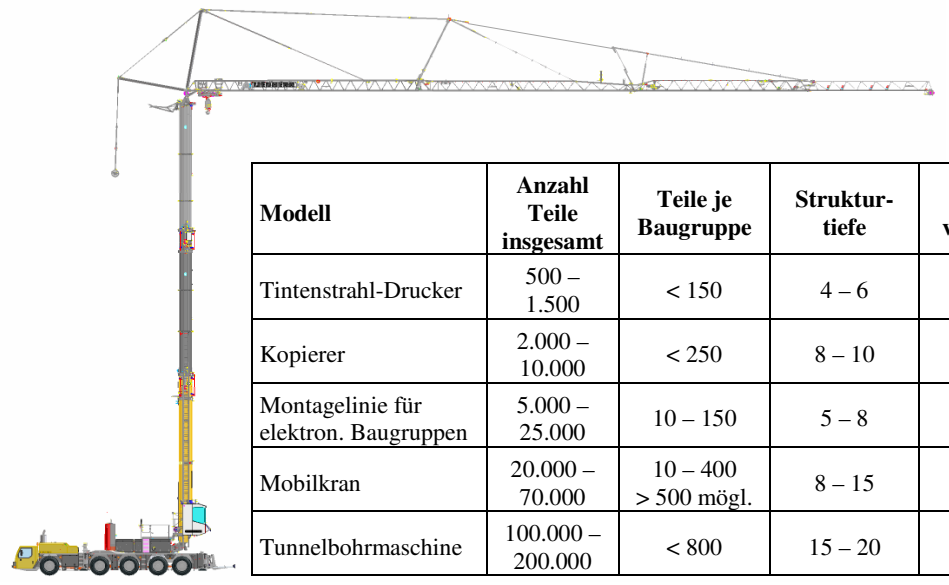
Nachfolgend sollen daher verschiedene Aspekte der CAD-gestützten Produktentwicklung im Hinblick auf Anforderungen betrachtet werden, die insbesondere aus der Handhabung großer Baugruppen resultieren. Nachdem eingangs definiert wird, unter welchen Bedingungen eine Baugruppe als groß angesehen werden soll, wird im weiteren untersucht, wie der Umgang mit entsprechenden Modellen heute praktiziert wird bzw. aus Konstrukteurssicht praktiziert werden sollte. Vor diesem Hintergrund wird sowohl der Konstruktionsprozeß, die Weiterverarbeitung der Daten als auch der Bedarf an Ressourcen näher beleuchtet. Abschnitt 3.4 widmet sich den Anforderun-

gen an das Datenmodell als solches. Abgeschlossen wird dieses Kapitel mit einer Zusammenfassung, die die Kernpunkte der verschiedenen Betrachtungen herausstellt und Gemeinsamkeiten bezüglich der Forderungen an die Datenverwaltung eines CAD-Systems aufzeigt.

Die durchgeführte Analyse basiert sowohl auf theoretischen Erkenntnissen nach [VDI93], [PaBe03] et al. als auch auf praktischen Erfahrungen verschiedener Konstruktionsabteilungen. Dabei wird auch auf die Ergebnisse umfangreicher Untersuchungen zurückgegriffen, die im Rahmen eines vorangegangenen Projektes bei der CoCreate Software GmbH gemacht wurden (vgl. [Schn02]).

3.1 Definition großer Baugruppen

Die Datenmenge, die zur Darstellung eines CAD-Modells notwendig ist, hängt von sehr unterschiedlichen Faktoren ab. Neben der Art der verwendeten Datenstrukturen spielen Eigenschaften des Modells selbst eine große Rolle. Diese lassen sich hauptsächlich nach der geometrischen Komplexität der verwendeten Teile sowie nach der strukturellen Komplexität der verwendeten Baugruppen unterscheiden.



Modell	Anzahl Teile insgesamt	Teile je Baugruppe	Struktur-tiefe	Wieder-verwendung
Tintenstrahl-Drucker	500 – 1.500	< 150	4 – 6	20 – 70 %
Kopierer	2.000 – 10.000	< 250	8 – 10	30 – 70 %
Montagelinie für elektron. Baugruppen	5.000 – 25.000	10 – 150	5 – 8	70 – 90 %
Mobilkran	20.000 – 70.000	10 – 400 > 500 mögl.	8 – 15	40 – 85 %
Tunnelbohrmaschine	100.000 – 200.000	< 800	15 – 20	70 – 90 %

Abbildung 3.1: Strukturelle Komplexität von CAD-Modellen (vgl. [Schn02])

Letztere ergibt sich nicht nur aus der Gesamtzahl aller Teile sondern ist ebenso entscheidend geprägt durch die Anzahl der Teile je Baugruppe, die Tiefe der Baugruppenstruktur, das heißt die Anzahl der Strukturebenen, und den Grad der Wiederverwendung. Dieser beschreibt das Verhältnis der Anzahl aller Instanzen mehrfach verwendeter Teile zur gesamten Teilezahl. **Abbildung 3.1** zeigt hierzu die Größenordnung der Kennzahlen anhand einiger Beispiele. Auffallend bei der Analyse von Anwenderdaten ist, daß der Grad der Wiederverwendung in sehr vielen Fällen jenseits von 50 % liegt, in einigen Fällen sogar bis 90 % reicht. Dies ist unter anderem dadurch bedingt, daß Standardteile wie Scheiben oder Muttern oft mehrere tausendmal verwendet werden.

Offensichtlich ist es schwierig, ein eindeutiges Kriterium zu definieren, nach dem eine Baugruppe als groß zu bezeichnen ist. Allgemein kann eine Größenordnung im Bereich von 5.000 Teilen vereinfachend als Richtwert für eine Untergrenze angenommen werden. Baugruppen, deren Teilezahl in diesem Bereich oder darüber liegt, sollen im folgenden als groß bezeichnet werden. Ein anschauliches Beispiel einer großen Baugruppe ist in **Abbildung 3.2** dargestellt.

Abbildung 3.2: CAD-Modell eines Mobilkrans, ca. 50.000 Teile

Dabei kann es durchaus zu Überschneidungen kommen. Eine Baugruppe mit 3.000 überwiegend hochkomplexen Einzelteilen kann sicher ein System weit mehr belasten als eine Baugruppe mit über 10.000 Teilen, die sich durch ihre Einfachheit und einen

sehr hohen Grad der Wiederverwendung auszeichnen. Diese Unschärfe soll aber bewußt in kauf genommen werden, da sie letztlich die reale Situation der verschiedenen CAD-Anwendungsbereiche widerspiegelt. Darüber hinaus sind viele Verfahrensweisen, die die Handhabung großer Baugruppen unterstützen, ebenso für kleinere Modelle sinnvoll einsetzbar, so daß die Existenz einer harten Trennlinie für die weiteren Untersuchungen ohnehin nicht zwingend notwendig ist.

Obwohl die Anforderungen an Systemressourcen auch mit zunehmender geometrischer Komplexität steigen, soll dieser Aspekt hier nicht weiter verfolgt werden. Die Optimierung der Datenstrukturen für Einzelteile, die eine komplexe Topologie, etwa eine Vielzahl von Bohrungen oder Rippen, besitzen oder aufwendige Freiformflächen verwenden, bedarf einer völlig anderen Betrachtungsweise und würde daher den Rahmen dieser Arbeit sprengen.

3.2 Methodik bei der Konstruktion

Der Konstruktionsprozeß ist mitentscheidend für den qualitativen und wirtschaftlichen Wert eines Produkts. Entsprechend groß ist die Notwendigkeit einer planbaren, zuverlässigen und effizienten Durchführung. In der Literatur findet sich daher eine Vielzahl an Richtlinien, Leitfäden und weiterreichenden Diskussionen zu diesem Thema – [VDI93], [Pah90], [PaBe03], [SpKr84], [Kro95] oder [Web05] markieren hier nur einen kleinen Ausschnitt.

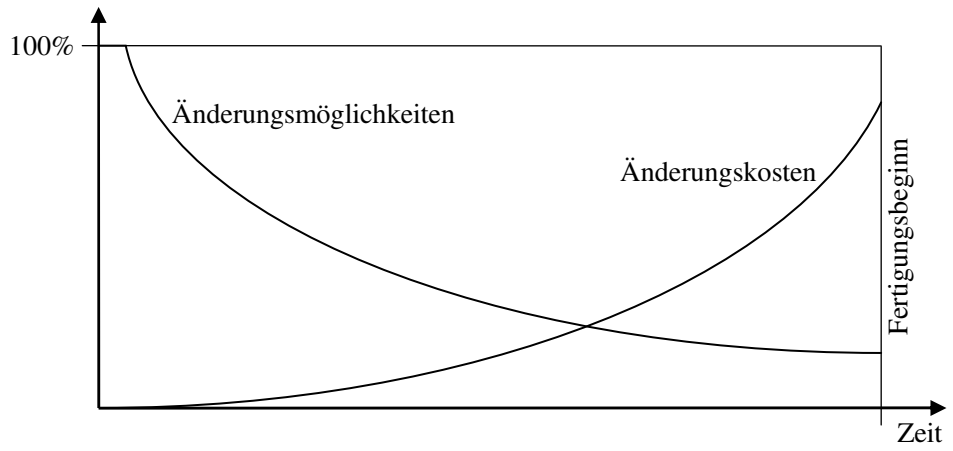


Abbildung 3.3: Änderungen während der Projektlaufzeit, vgl. [Kro95]

Aufgrund des zunehmenden Kostendrucks wird insbesondere die Effizienz des Konstruktionsprozesses immer wieder diskutiert. Untersuchungen[14] haben gezeigt, daß die Änderungskosten mit fortschreitender Projektlaufzeit progressiv steigen, während die Änderungsmöglichkeiten mit fortschreitender Projektlaufzeit degressiv fallen. Dieser Zusammenhang ist in **Abbildung 3.3** nach [Kro95] prinzipiell dargestellt (vgl. dazu auch [Ber02]). Jede konzeptionelle Festlegung schränkt demnach den Bewegungsspielraum der nachfolgenden Fachabteilungen weiter ein. Die Produktentwicklung legt bereits 80% der Gesamtkosten und Gesamtprobleme des Produktes fest, ohne daß nachfolgende Abteilungen entscheidende Einflußmöglichkeiten zur Korrektur haben.

Zur Erlangung größerer Flexibilität, und damit auch höherer Effizienz, haben Aspekte einer interdisziplinären Produktentwicklung in der Vergangenheit an Bedeutung gewonnen. Wurden früher verschiedene Schritte zur Konstruktion, Berechnung und Fertigungsvorbereitung strikt sequentiell ausgeführt – mit allen negativen Konsequenzen für den Fall, daß erst im späteren Verlauf Probleme aufgedeckt wurden, die einen Rückschritt erforderten – so kann heute in der Regel von einer engen Verzahnung aller an der Entwicklung beteiligten Bereiche ausgegangen werden. Dies gilt nicht nur für die systematische Lösungsentwicklung, sondern auch für die ihr zugrunde liegende Datenbasis.

3.2.1 VDI-Richtlinie 2221

Eine Standardisierung der Konstruktionsprozesse wurde mit der *VDI-Richtlinie 2221* (vgl. [VDI93]) angestrebt. Diese Richtlinie schlägt ein branchenunabhängiges, generelles Vorgehen zum Entwickeln und Konstruieren technischer Produkte vor. **Abbildung 3.4** gibt hierzu einen Überblick. Danach wird der Konstruktionsprozeß in vier ineinandergreifende Phasen unterteilt. Ausgehend von der Planung (I), über Konzeption (II) und Entwurf (III) bis hin zur Ausarbeitung (IV) nimmt die Konkretisierung mit jeder Phase zu. Das Gesamtvorgehen wird in sieben Arbeitsschritte gegliedert, aus denen jeweils ein Arbeitsergebnis hervorgeht. Diese Arbeitsschritte werden je nach Aufgabenstellung vollständig, nur teilweise oder mehrfach iterativ durchlaufen.

---

[14] Vgl. [Kro95]

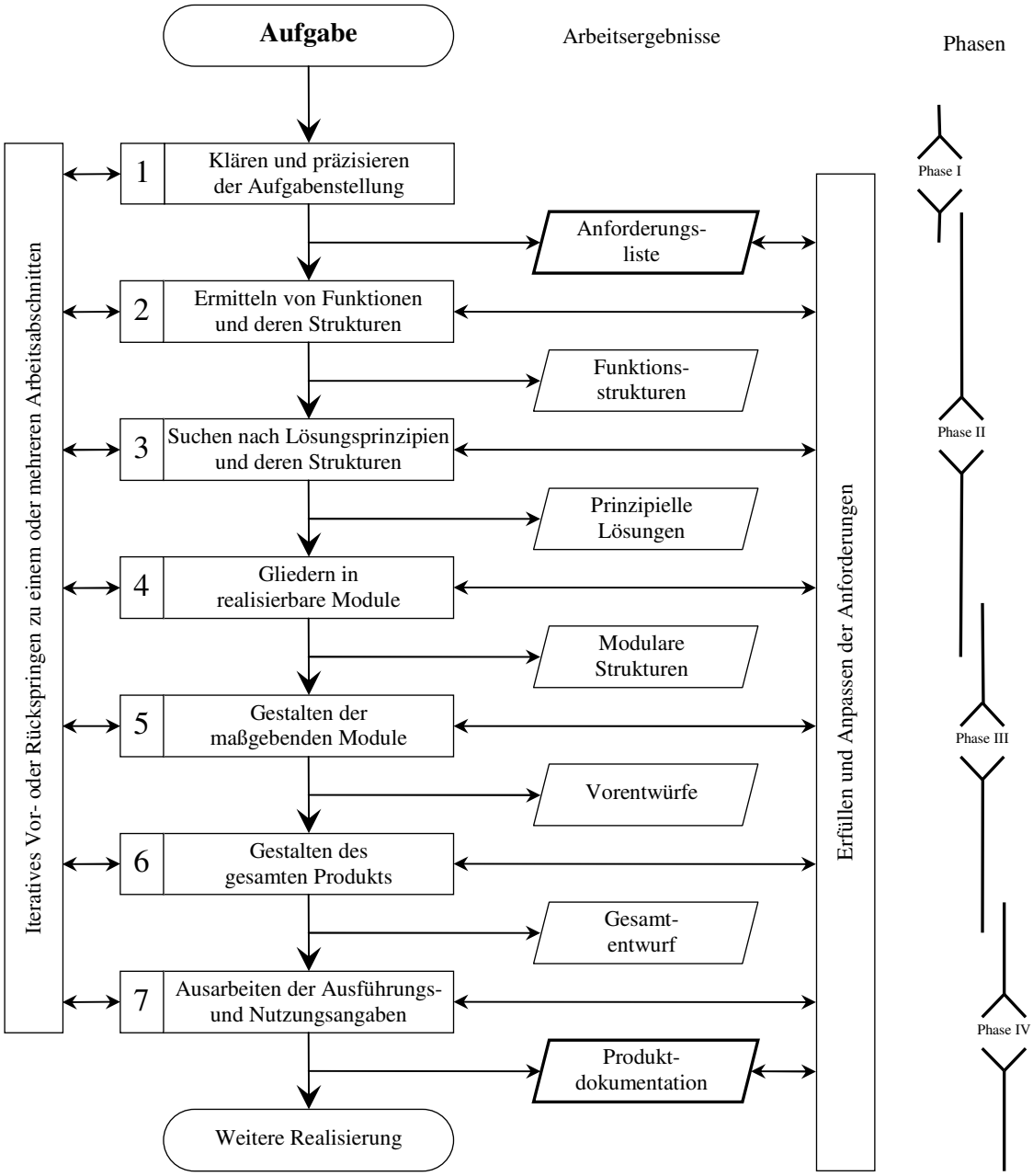


Abbildung 3.4: Arbeitsschritte beim Planen und Konstruieren nach VDI 2221

Bedingt durch die allgemeine Anwendbarkeit ist der Ablauf des Konstruktionsprozesses nur grob strukturiert und läßt deshalb eine Vielzahl von produkt- und unternehmensspezifischen Vorgehensvarianten zu. Die Richtlinie stellt damit quasi einen Leitfaden dar, an dem sich überdies terminliche und organisatorische Abläufe orientieren können. Besonders betont wird auch der iterative Charakter des Vorgehens.

3.2.2 Konstruktionslehre nach Pahl / Beitz

In [PaBe03] und [Pah90] wird der prinzipielle Ablauf der Konstruktion in ähnlicher Weise diskutiert. Auf die Beschreibung der Phasen des Entwerfens und Ausarbeitens soll an dieser Stelle aber noch etwas ausführlicher eingegangen werden, da hier das Haupteinsatzgebiet von CAD bzw. CAD-bezogenen Daten liegt.

Das Entwerfen umfaßt die konkrete Gestaltung des Produkts, die Wahl von Werkstoffen und Fertigungsverfahren, die Festlegung der Hauptabmessungen sowie eine Vielzahl von Untersuchungen, zum Beispiel die der räumlichen Verträglichkeit. Das zu erreichende Ergebnis ist die gestalterische Festlegung des Produkts. Die Tätigkeit auf dem Weg dorthin ist geprägt durch sich abwechselnde und ergänzende Vorgänge der Analyse und Synthese und unterteilt sich nach [Pah90] in die nachfolgend aufgeführten Arbeitsschritte, die wiederum in einem iterativen Prozeß durchlaufen werden.

1. *Analyse der Aufgabenstellung*
   Erarbeiten eines Konzepts, das heißt einer prinzipiellen Lösung, erkennen der gestaltbestimmenden Anforderungen, wie grundsätzliche Gestalt, Anordnung, Abmessungen, Werkstoffart etc.
2. *Strukturieren in Module*
   Gliedern nach wesentlichen Aufgabenbereichen
3. *Grobgestalten*
   Entwickeln einer Rohform, die die Hauptfunktionen erfüllt und bezüglich Abmessungen und Anordnung möglichst zutreffend ist
4. *Suchen von Lösungen für Nebenfunktionen*
   Entwickeln von Komponenten für Aufgaben wie dichten, halten, verbinden, befestigen. Diese Tätigkeit wird unterstützt durch die Verwendung von Norm- und Wiederholteilen und beeinflußt die weitere Gestaltung in hohem Maße.
5. *Generierung der Feingestalt*
   Detail- und Feingestaltung durch mannigfache Ergänzungen, Anpassungen und Vervollständigungen. Die Tätigkeiten erfolgen funktions- und fertigungsorientiert. Weiterhin wird die technische Absicht mit Hilfe von Rundungen, Sicherungsnuten, Fasen, Bohrungen, Rippen etc. definiert, wobei oftmals Formelemente und Features Verwendung finden.

6. *Optimieren*
   Durchführen von Berechnungen, Kontrolle auf Fehler und Störgrößeneinflüsse wie räumliche Verträglichkeit etc., vergleichen von Varianten

Die herstellungstechnische Festlegung erfolgt in der Phase des Ausarbeitens. Dazu werden endgültige Vorschriften für Form, Bemessung und Verwendung von Werkstoffen festgelegt, Herstellungs- und Gebrauchsmöglichkeiten überprüft sowie verbindliche zeichnerische und sonstige Unterlagen erstellt. Der Schwerpunkt dieser Phase liegt in der Erarbeitung von Fertigungsunterlagen. Die – wiederum iterativ auszuführenden – Arbeitsschritte werden folgendermaßen beschrieben:

1. *Detaillieren des endgültigen Entwurfs*
   Finale Ausarbeitung von Einzelteilen, Detailoptimierungen hinsichtlich des Werkstoffs, der Toleranzen etc., Erstellung von Einzelteilzeichnungen
2. *Zusammenfassen zu Baugruppen*
   Zusammenfassen von Einzelteilen zu Baugruppen, erstellen von Baugruppenzeichnungen
3. *Vervollständigen der Fertigungsunterlagen*
   Erstellen von Montage- und Transportvorschriften, Betriebsanleitungen etc.
4. *Prüfen der Fertigungsunterlagen*
   Überprüfung der Unterlagen, insbesondere der Zeichnungen, hinsichtlich der Einhaltung von Normen, einer fertigungsgerechten Bemaßung, Beschaffungsgesichtspunkten etc.

Eine Überschneidung der Arbeitsschritte aus der Entwurfsphase mit denen der Ausarbeitungsphase ist in der Praxis oft anzutreffen und im Sinne einer ganzheitlichen Betrachtung des Konstruktionsprozesses auch gewollt. Ob Tätigkeiten noch von der Konstruktionsabteilung oder bereits von einer nachgeschalteten Abteilung ausgeführt werden, hängt von der Organisationsstruktur des jeweiligen Unternehmens ab und ist für die Betrachtungen von untergeordneter Bedeutung. Wichtiger ist, daß der Konstrukteur, unabhängig davon, wer die Ausarbeitung übernimmt, diese bereits vorbereitend unterstützt.

3.2.3 Produktentwicklung auf der Basis von CPM / PDD

Einen formalisierten, weiterführenden Ansatz zur Modellierung und Entwicklung von Produkten beschreibt Weber in [Web05]. Dort wird zunächst das Konzept des *Characteristics-Properties-Modelling (CPM)* eingeführt, nach dem ein Produkt im wesentlichen durch seine Merkmale (Characteristics) und Eigenschaften (Properties) bestimmt ist. Darauf aufbauend wird mit *Property-Driven-Development (PDD)* ein Prozeß hergeleitet, in dessen Verlauf ausgehend von den Soll-Eigenschaften iterativ Merkmale des Produkts gesucht und verfeinert werden.

Characteristics-Properties-Modelling

CPM unterscheidet strikt zwischen den Merkmalen eines Produkts einerseits und seinen Eigenschaften andererseits. Erstere beschreiben dessen Struktur, Gestalt sowie Beschaffenheit und können vom Konstrukteur direkt beeinflußt werden. Letztere dagegen beschreiben das Verhalten des Produkts und werden in der Regel vorgegeben. Beide Kategorien stehen in enger Verbindung. Mittels Analyse können zu gegebenen Merkmalen des Produkts dessen Eigenschaften bestimmt, im Falle eines virtuellen Modells vorhergesagt werden. Umgekehrt kann von gegebenen oder geforderten Eigenschaften mittels Synthese auf notwendige Merkmale rückgeschlossen werden. **Abbildung 3.5** gibt dazu einen Überblick.

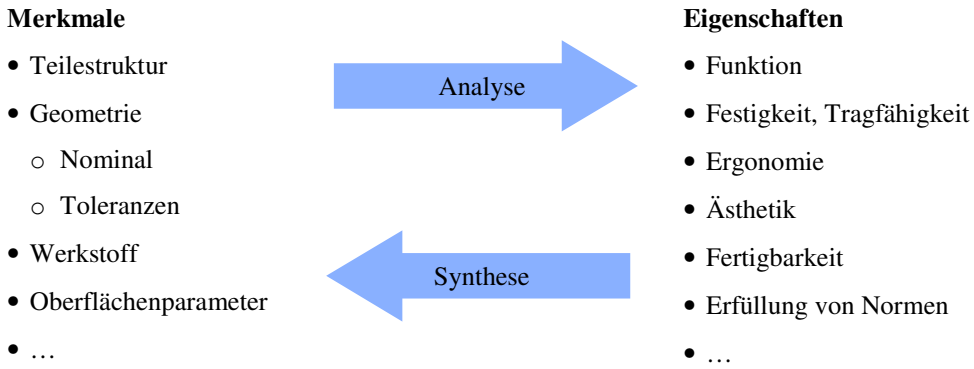


Abbildung 3.5: Merkmale und Eigenschaften eines Produkts

Die Prozesse der Analyse und Synthese setzen dabei n Eingangsgrößen zu m Ausgangsgrößen in Beziehung. **Abbildung 3.6** zeigt eine formalisierte Darstellung mit den Merkmalen C_i, (geforderten) Eigenschaften P_i (PR_i) und Beziehungen R_i. Diese können ferner externen Bedingungen EC_i unterliegen.

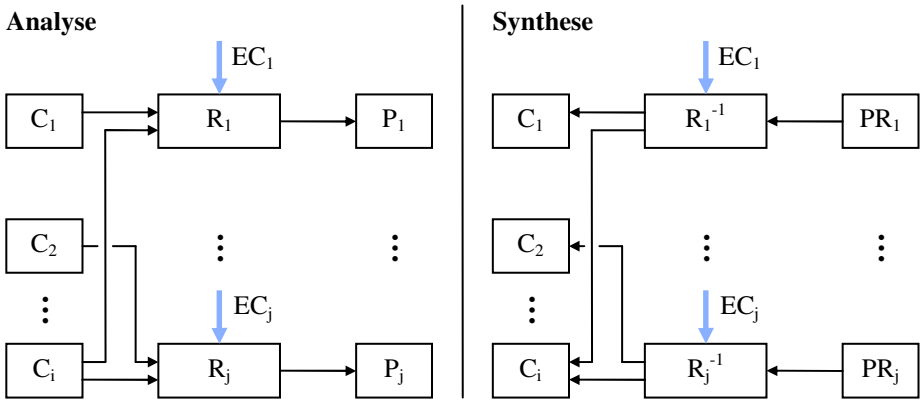


Abbildung 3.6: Formalisierte Darstellung von Analyse und Synthese

Property-Driven-Development

Produktentwicklung kann als ein Prozeß der Synthese aufgefaßt werden, in den – quasi zur Kontrolle und Steuerung – immer wieder Schritte der Analyse eingeschoben werden. Mit jedem Schritt der Synthese werden dabei weitere Merkmale des zu entwickelnden Produkts bestimmt und vergeben. Mit jedem Schritt der Analyse wird im Gegenzug das Wissen über die zu erwartenden Eigenschaften des Produkts verfeinert. Die Abweichung dieser Eigenschaften von den geforderten wird zur Steuerung des weiteren Vorgehens ausgewertet. Dies geschieht mit dem Ziel, die Abweichung schrittweise zu minimieren. Der Entwicklungsprozeß kann somit durch die folgenden Iterationsschritte verkürzt dargestellt werden.

1. Ausgehend von einigen Soll-Eigenschaften (PR_j) werden wesentliche Merkmale (C_j) der zukünftigen Lösung auf dem Weg der Synthese bestimmt.
2. Durch Analyse dieser Merkmale werden die resultierenden Ist-Eigenschaften (P_j) bestimmt. Dabei findet keine Beschränkung auf die ursprünglichen Eingangsgrößen (PR_j) statt, vielmehr werden alle resultierenden Eigenschaften betrachtet.
3. Soll- und Ist-Eigenschaften werden verglichen. Das Resultat (ΔP_j) beschreibt das Defizit des aktuellen Entwurfsschritts.
4. In Vorbereitung auf den folgenden Iterationszyklus werden anhand dieses Zwischenergebnisses Entscheidungen getroffen, welche Soll-Eigenschaften als nächstes betrachtet – verfeinert oder hinzugenommen – werden sollen.

3.2.4 Richtlinien und Empfehlungen zur praktischen Umsetzung

Neben der Methodik bei der Konstruktion selbst wird auch deren Umsetzung in die Praxis erörtert. Pahl / Beitz geben in [PaBe03] beispielsweise die Empfehlung, die Generierung der Grob- und Feingestalt generell nur so detailliert vorzunehmen, wie es der jeweilige Zweck erfordert. Damit wird das Ziel verfolgt, den Generierungsaufwand möglichst klein zu halten, Antwortzeiten zu reduzieren sowie das Erreichen etwaiger Modellgrenzen zu vermeiden bzw. hinauszuschieben. Darüber hinaus soll angestrebt werden, mit möglichst wenigen und einfachen Operationen das gewünschte Konstruktionsziel zu erreichen.

Ein einfaches Beispiel für die Grob- und Feingestaltung ist in **Abbildung 3.7** dargestellt. Die Grobgestaltung ist bereits ausreichend, um den Bauraum innerhalb einer Baugruppe festzulegen. Somit ist die Möglichkeit gegeben, mit der Konstruktion von Anschlußteilen zu beginnen. Eine näherungsweise Berechnung ist bereits jetzt möglich, der eventuell daraus resultierende Aufwand zur Änderung des Bauteils ist noch gering. Dieser steigt mit zunehmendem Detaillierungsgrad. Daher soll das Anbringen von Bohrungen, Verrundungen etc. so spät wie möglich erfolgen.

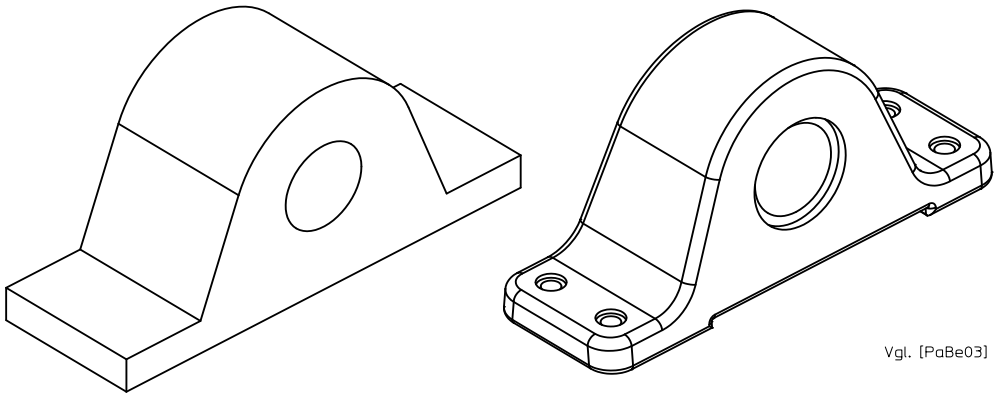


Abbildung 3.7: Grob- bzw. Feingestaltung am Beispiel eines Lagers

Fertigungstechnische Informationen können oftmals bereits früh in der Konstruktionsphase durch den Konstrukteur festgelegt werden. Andererseits ist eine sinnvolle Arbeitsteilung zwischen Konstruktion und produktionsvorbereitenden Abteilungen erforderlich, um den Konstrukteur nicht zu überlasten. Folgerichtig ist daher eine Festlegung der jeweiligen Bearbeitungstiefe am Objektmodell verbunden mit entsprechenden Schnittstellen anzustreben, damit nachgeordnete Tätigkeiten sich ohne

erneute Datenaufbereitung anschließen können. Dies geht einher mit der Definition eines Freigabesystems für das Objektmodell und die Regelung von Änderungsprozeduren mit dem Ziel, keine redundanten oder gar widersprüchlichen Informationen zum gleichen Objekt zuzulassen.

Zusammenfassend kann festgehalten werden, daß der Konstruktionsprozeß, so wie er in der Literatur diskutiert wird, zum einen durch die enge Zusammenarbeit von Konstrukteuren und Bearbeitern fertigungsvorbereitender Tätigkeiten geprägt ist und zum anderen einen stark iterativen Charakter hat. Auf die gemeinsame Datenbasis bezogen bedeutet dies, daß daraus hohe Anforderungen an die Integrität und Assoziativität der jeweils verwendeten Datenausschnitte resultieren.

3.3 Typische Arbeitsweisen und Prozesse in der Praxis

Moderne CAD-Systeme haben sich zunehmend zu integrierten Systemen entwickelt, die den Konstrukteur außer bei der Erstellung des virtuellen Produkts auch bei vielen nachgeschalteten, sekundären Tätigkeiten unterstützen. Diese Entwicklung spiegelt sich in der praktischen Umsetzung des Konstruktionsprozesses wider, wurde andererseits aber auch gerade dadurch vorangetrieben.

Typischerweise wird zunächst ein CAD-Modell in der Konstruktionsabteilung schrittweise aufgebaut. Danach – vielfach aber auch schon während der Konstruktion und auf Teilergebnissen basierend – erfolgt die Weiterverarbeitung und Ergänzung der Daten in diversen Anwendungsbereichen. Beispiele hierfür sind die technische Dokumentation, Kinematikanalyse und Kollisionsuntersuchungen ebenso wie der Formen- und Modellbau, aber auch kaufmännisch orientierte Bereiche wie die Angebotserstellung. Die Zuständigkeit für die einzelnen Bereiche fällt dabei oft in verschiedene Abteilungen. In der Anwendung ergibt sich jedoch ein eher fließender und von Unternehmen zu Unternehmen unterschiedlicher Übergang.

An dieser Stelle soll der praktische Ablauf verschiedener Schritte des Konstruktionsprozesses vom Projektstart bis zum Abschluß der Fertigungsvorbereitung näher betrachtet werden. Die nachfolgenden Abschnitte fassen das Ergebnis der Analyse

von Arbeitsweisen in verschiedenen Konstruktionsabteilungen zusammen, die sich in der Hauptsache mit der Entwicklung großer Baugruppen beschäftigen. Darin gehen auch Resultate umfassender Untersuchungen ein, die bei der CoCreate Software GmbH im Rahmen eines früheren Projekts durchgeführt wurden (vgl. [Schn02]). Die Zusammenfassung reflektiert nicht nur Praktiken, basierend auf dem heutigen Stand der Technik, sondern berücksichtigt auch Überlegungen seitens der Konstrukteure zu zukünftigen Arbeitsweisen, welche die Effizienz ihrer Tätigkeit weiter steigern sollen und über die heute von CAD-Systemen bereitgestellten Möglichkeiten hinausgehen.

3.3.1 Projektstart

Im Vorfeld der Konstruktion werden zunächst Anforderungen definiert, ein Lösungskonzept erarbeitet und Eckpunkte des zu entwickelnden Produkts wie Form, Funktion und Abmessungen festgelegt. Nachdem diese Rahmenbedingungen bekannt sind, erfolgt die Definition der anfangs noch groben Baugruppenstruktur. Mit dieser Tätigkeit werden in der Regel die ersten CAD-Daten angelegt.

Die Definition der Baugruppenstruktur wird meistens von Projektleitern in ihrem jeweiligen Verantwortungsbereich durchgeführt. Häufig anzutreffen ist eine Vorgabe der Struktur in Haupt-, Ober- und Unterbaugruppen. Damit wird ein erster Entwurf dokumentiert, der im Sinne einer Planung vorgedachte Elemente enthält, ohne daß diese bereits konstruiert worden sind. Die eigentliche Ausdetaillierung erfolgt anschließend durch die Konstrukteure.

In einigen Unternehmen ist die Vergabe von Zugriffsrechten auf die Daten ebenfalls an diesen Aufbau geknüpft. Dort kann ein Konstrukteur zwar in seinem Bereich Teile ändern, löschen und hinzufügen sowie die Baugruppenstruktur auf der unteren Ebene weiter verfeinern. Auf die grundlegende Baugruppenstruktur hat er jedoch nur lesenden Zugriff und muß eventuell erforderliche Änderungen beim Projektleiter beantragen bzw. von ihm durchführen lassen.

Eine Vielzahl der Konstruktionen sind Varianten- oder Änderungskonstruktionen, bei denen die Struktur und weitere Daten aus Vorgängerprojekten als initialer Schritt

übernommen werden können. Bei Neuentwicklungen wird die Baugruppenstruktur entsprechend von Grund auf erstmalig angelegt. Auch hier können gelegentlich Daten anderer Projekte wenigstens in Teilen als Vorlage dienen.

Ergänzt wird die Grobstruktur vielfach durch weitere Daten, die in der Summe zu einer skelettartigen Beschreibung des Modells beitragen. Hierzu zählen hauptsächlich räumliche Vorgaben und Schnittstellen. Erstere können vordefinierte Arbeitsbereiche, vorgegebene geometrische Ausdehnungen oder auch äußere Formen sein, wie etwa der Ausschnitt einer Karosserie, in die eine Leuchteinheit zu konstruieren ist. Zu den Schnittstellen gehören Medienanschlüsse und Befestigungspunkte ebenso wie Referenzmaße zu weiteren Maschinen, benachbarten Aggregaten oder anderen wichtigen Elementen. Über reine Maße hinaus werden oft auch Anschlußgeometrien vorgegeben.

Die ergänzenden Vorgaben müssen nicht notwendigerweise bereits beim Projektstart vollständig festgelegt werden. Vielmehr kommt es hier im Verlauf der Konstruktion immer wieder zu Anpassungen und Komplettierungen.

3.3.2 Konstruktion

Unter Konstruktion im engeren Sinne wird das Entwerfen und Gestalten des virtuellen Produkts verstanden. In diesem Arbeitsschritt wird die vorgegebene Baugruppenstruktur mit Leben gefüllt. Die Konstrukteure entwerfen Schritt für Schritt die einzelnen Bauteile, kombinieren sie zu Baugruppen und schaffen eine weitere Feingliederung der Gesamtstruktur. Auf diese Weise entstehen die Basisdaten für alle nachgeschalteten Aufgabenbereiche. Nicht nur deshalb, sondern auch weil dieser Vorgang zu den aufwendigsten Arbeitsschritten zählt, wird dieser Phase im Entwicklungsprozeß eine hohe Bedeutung beigemessen.

Da nachgeschaltete Aufgaben wie Berechnung und Analyse häufig zu Änderungen am Modell führen, wird eine enge Verzahnung mit diesen Tätigkeiten angestrebt. Um den Änderungsaufwand möglichst gering zu halten, werden Berechnungen meistens möglichst frühzeitig durchgeführt und deren Ergebnisse unmittelbar in der weiteren Konstruktion berücksichtigt.

Die Art und Weise, wie diese Verzahnung in der Praxis umgesetzt wird, unterscheidet sich jedoch von Unternehmen zu Unternehmen stark. Die Bandbreite reicht von einer zentralen Konstruktionsabteilung, die für das gesamte Spektrum der Aufgaben zur Sicherstellung der Fertigung verantwortlich zeichnet, bis hin zur klaren Trennung der Zuständigkeitsbereiche für die einzelnen Aufgaben, verteilt auf verschiedene Abteilungen.

Im ersten Fall führt der Konstrukteur auch alle wesentlichen Berechnungen und Analysen durch und stellt Informationen bereit, die in nachfolgenden Prozeßschritten verwendet werden. Der Vorteil dieses Verfahrens besteht darin, daß alle Informationen in einer Hand liegen. Allerdings setzt es ein sehr komplexes Fertigungswissen des Konstrukteurs sowie die Kenntnis sämtlicher Bearbeitungsverfahren und Anforderungen anderer Bereiche wie Simulation und Arbeitsplanung zum Zeitpunkt der Konstruktion voraus. Bei klarer Trennung der Zuständigkeitsbereiche ist das Fachwissen dagegen mehr auf die einzelnen Bearbeiter verteilt, wobei hier ein wesentlich höherer Kommunikationsaufwand zur Gewährleistung eines reibungslosen und effizienten Entwicklungsablaufs erforderlich ist.

Concurrent Engineering

Aufgrund des zeitlichen Aufwands, der für die Konstruktion erforderlich ist, und des vorherrschenden Termin- und Kostendrucks besteht das Bestreben, möglichst viele Abläufe zu parallelisieren. Da es bei der Entwicklung größerer Maschinen tatsächlich ausreichend viele Konstruktionsaufgaben gibt, die unabhängig voneinander gelöst werden können, findet das Verfahren der zeitgleichen Konstruktion – *Concurrent Engineering* oder auch *Simultaneous Engineering* – in der Praxis eine häufige Anwendung.

Dabei orientieren sich die Zuständigkeitsbereiche der jeweiligen Konstrukteure stark an Einzelteilen oder kleineren Baugruppen. In der Folge hat jeder Bearbeiter nur in seine Bauteile und deren direkte geometrische Umgebung einen tieferen Einblick. Weiter entfernte Teile werden, wenn überhaupt, nur benötigt, um die eigene Arbeit in den Kontext der Gesamtbaugruppe einordnen zu können.

Auch wenn die verschiedenen Konstruktionsaufgaben an sich unabhängig sind, können einzelne von ihnen gemeinsame Schnittstellen haben, etwa angrenzende Bauräume oder auch direkte Verbindungspunkte. Insbesondere dann, wenn an solchen Aufgaben parallel gearbeitet wird, ist es notwendig, die Schnittstellen jederzeit überprüfen und eventuelle Konflikte zeitnah aufdecken zu können.

Konstruktionsumgebung, Konstruieren im Kontext

Beschränkt sich die Konstruktionstätigkeit auf einen Ausschnitt einer großen Baugruppe, ist es häufig nicht sinnvoll, das komplette Modell in voller Ausbaustufe zu bearbeiten. Details aus weiter entfernten Bereichen des Modells, die keinen Beitrag zur eigentlichen Aufgabe leisten, können ebenso weggelassen werden.

Überdies ist die Bearbeitung des vollständigen Modells je nach Datenmenge unter Umständen gar nicht möglich, weil damit die Grenzen des verfügbaren Speichers überschritten werden. Selbst wenn die verwendete Hardware und deren Speicherausbau das Laden erlaubten, wäre das System derartig belastet, daß akzeptable Antwortzeiten und damit effizientes Arbeiten nicht mehr in erforderlichem Maße erreichbar sind.

Konstrukteure wenden daher verschiedene Verfahren an, das zu bearbeitende Modell soweit auszudünnen, daß nur derjenige Ausschnitt tatsächlich geladen wird, der zur Erfüllung der konkreten Aufgabe erforderlich ist. Dies erfolgt beispielsweise durch die Verwendung von Konfigurationen, in denen Details bis hin zu ganzen Unterbaugruppen unterdrückt werden können, oder durch die Verwendung von partiellem Laden.

Als wichtige Bedingung wird dabei angesehen, daß das Arbeiten im Kontext des Gesamtmodells weiterhin möglich ist. Die logische Baugruppenstruktur soll vollständig erhalten bleiben, das heißt, es werden nur die datenintensiven Bauteile ausgelassen sowie gegebenenfalls Baugruppen der unteren Ebenen. Außerdem soll der geladene Ausschnitt relativ zur gesamten Baugruppe exakt positioniert sein. Auf diese Weise können jederzeit unabhängige Teile und Unterbaugruppen nachgeladen werden und erscheinen ebenfalls an exakter Stelle im Gesamtmodell. Positionierungsfehler können dadurch vermieden werden.

Die Konstruktionsumgebung wird im allgemeinen so zusammengestellt, daß alle Teile, die bearbeitet werden sollen oder für exakte Referenzierung benötigt werden, in vollem Umfang geladen werden. Dies sind normalerweise die Teile der eigenen Baugruppe sowie Randbereiche aus benachbarten Baugruppen. Damit sind alle geometrischen Daten vorhanden, die zur genauen Messung von Abständen, Ableitung von Bauformen oder Diskussion von Schnittstellen erforderlich sind. Ergänzt werden kann diese Auswahl durch informelle Geometrie, also Teile, die nur zur visuellen Orientierung dienen, nicht aber zur Referenzierung. Insgesamt soll die Umgebung so einfach wie möglich, aber so detailliert wie nötig sein. Dabei wird oft ein Kompromiß gesucht zwischen einer möglichst optimalen Umgebung und dem Aufwand, diese zusammenzustellen und zu pflegen.

Im Rahmen von Kollisionsprüfungen stellt sich regelmäßig die Aufgabe, die Testumgebung so aufzusetzen bzw. die Konstruktionsumgebung so zu erweitern, daß alle potentiellen Kollisionsteile tatsächlich darin enthalten sind. Die Struktur der Nachbarbaugruppen – insbesondere die räumliche Anordnung der Teile innerhalb der Baugruppen – ist dem Konstrukteur aber selten hinreichend bekannt, wenn er diese nicht selbst bearbeitet hat. Ein ähnliches Problem besteht, wenn naheliegende Teile aus anderen Baugruppen zur Bestimmung von Referenzmaßen benötigt werden.

Um die Umgebung entsprechend zu ergänzen, werden dazu heute in Frage kommende Nachbarbaugruppen oftmals wesentlich umfangreicher als notwendig nachgeladen, um sicherzustellen, daß alle erforderlichen Teile zur Verfügung stehen. Für eine effiziente Identifikation nachzuladender Teile wären jedoch Methoden zur visuellen oder räumlichen Auswahl wünschenswert.

Rüstzeiten

Typischerweise arbeiten Konstrukteure über einen längeren Zeitraum am selben Modellausschnitt, wobei die Dauer von einigen Tagen bis hin zu mehreren Monaten reichen kann. Änderungen dieses Ausschnitts sind eher selten und dienen hauptsächlich der Vervollständigung der Konstruktionsumgebung. Neben der Hinzunahme eigener Neuentwicklungen werden gelegentlich weitere Referenzteile aus Nachbarbaugruppen ergänzt. Ebenso findet eine Aktualisierung bereits verwendeter Refe-

renzmodelle statt, die unter der Kontrolle anderer Konstrukteure stehen und von diesen verändert wurden.

Dementsprechend verläuft die Prozedur, den Modellausschnitt für die geplante Arbeit eines Tages aufzubauen, jeden Morgen sehr ähnlich. Diese Rüstzeit bestehend aus Teileauswahl, Ladezeit sowie eventuellem Nachladen zur Aktualisierung und Ergänzung kann so durchaus einen signifikanten Zeitraum in Anspruch nehmen. Da es sich hierbei um unproduktive Zeit handelt, besteht die Notwendigkeit, diesen Prozeß weitestgehend zu automatisieren.

Iterationsschritte

Die Konstruktion kann als Prozeß stetiger Optimierung aufgefaßt werden. Vielfach werden bereits erste Berechnungen und Analysen an grob entwickelten Modellen bzw. Teilmodellen durchgeführt. Dabei spielt es keine Rolle, ob diese Untersuchungen von den Konstrukteuren selbst oder in anderen Abteilungen ausgeführt werden. Die Resultate fließen umgehend in den Konstruktionsprozeß wieder ein und können sofort berücksichtigt werden, bevor mit der weiteren Ausdetaillierung fortgefahren wird. Diese Schleife wird je nach Erfordernissen mehrfach durchlaufen.

Ein direkter Datenaustausch findet in vielen Fällen in beiden Richtungen statt. Der Konstrukteur ergänzt das Modell bereits um Informationen, die für die Verwendung in nachgeschalteten Arbeitsschritten bestimmt sind. Dies können beispielsweise Materialeigenschaften, kritische Maße oder eine vorgesehene Zusammenbaureihenfolge sein. In umgekehrter Richtung erhält er Ergebnisse aus Berechnung und Analyse zurück, die er direkt in seine weitere Konstruktionstätigkeit einflechten kann. Auf diese Weise entsteht ein fließender Übergang zwischen Konstruktion und Nachbereitung.

Eine Folge dieser engen Zusammenarbeit ist ein steigender Detaillierungsgrad, der zur weiteren Optimierung ausgenutzt wird. Exemplarisch kann diese Entwicklung an der Verwendung von Schweißnähten gezeigt werden. Wurden diese früher hinsichtlich ihrer Auswirkungen auf die Gewichts- und Schwerpunktberechnung nur überschlägig berücksichtigt, so sind sie heute bei einigen Anwendern Bestandteil der CAD-Daten und können somit wesentlich präziser in die Berechnungen eingehen.

Der Iterationsprozeß aus sich abwechselnden Schritten der Konstruktion und Berechnung ist dabei keineswegs auf einzelne Teile beschränkt. Vielmehr wird dieses Verfahren auch genutzt, um mehrere initiale Lösungsvorschläge auf Teile- oder auch Baugruppenebene bis zu einer gewissen Reife zu entwickeln, bevor die Entscheidung zur Weiterarbeit an einer ausgewählten Variante fällt.

3.3.3 Berechnung, Analyse, Simulation

Im folgenden werden Arbeitsschritte betrachtet, die auf dem jeweils aktuellen Stand des 3D-Modells ausgeführt werden und aufgrund ihrer Ergebnisse oftmals weitere Veränderungen der Konstruktion nach sich ziehen. Entsprechend hoch ist ihre Integration in den Entwicklungsprozeß. Auf den iterativen Charakter der Konstruktions- und Berechnungsschritte wurde bereits im vorangegangenen Abschnitt näher eingegangen. Bedingt durch die Rückkopplung zur Konstruktion ist es naheliegend, diese Tätigkeiten vor der endgültigen Freigabe der Modelldaten abzuschließen.

Die Zahl möglicher Anwendungen im Bereich der Berechnung und Analyse ist groß, daher sollen hier exemplarisch nur einige genannt werden. Da viele Verfahren stark produktspezifisch zum Einsatz kommen bzw. sich an den besonderen Anforderungen der nachgeschalteten Produktionsbereiche orientieren, läßt sich praktisch keine generelle Regel aufstellen, nach der einige Arbeitsschritte grundsätzlich auszuführen sind. Vielmehr ergibt sich die Art der erforderlichen Tätigkeiten aus den jeweiligen Produkteigenschaften und den Fertigungsgegebenheiten.

Berechnungen werden unter anderem zur statischen Auslegung und Optimierung der Modelle durchgeführt – häufig mit Hilfe von Finit-Element-Methoden. Dazu ist es auch notwendig, Gewicht, Schwerpunkt und Trägheitsmomente einzelner Teile oder ganzer Baugruppen zu ermitteln. Ebenso beruht die Massenermittlung auf Berechnungen am Modell.

Analysen werden beispielsweise durchgeführt, um Erkenntnisse über das Schwingungsverhalten zu gewinnen oder Aussagen über die voraussichtliche Haltbarkeit des Produkts treffen zu können. Das vorausberechnete Biegeverhalten eines Teils unter gegebenen Lastannahmen ist in **Abbildung 3.8** dargestellt. Weiterhin kann das

Modell bereits vorab auf mögliche Kollisionen einzelner Teile untersucht werden. Um eine solche Überprüfung effizient vornehmen zu können, ist die programmatische Kenntnis räumlicher, baugruppenübergreifender Zusammenhänge von großer Bedeutung.

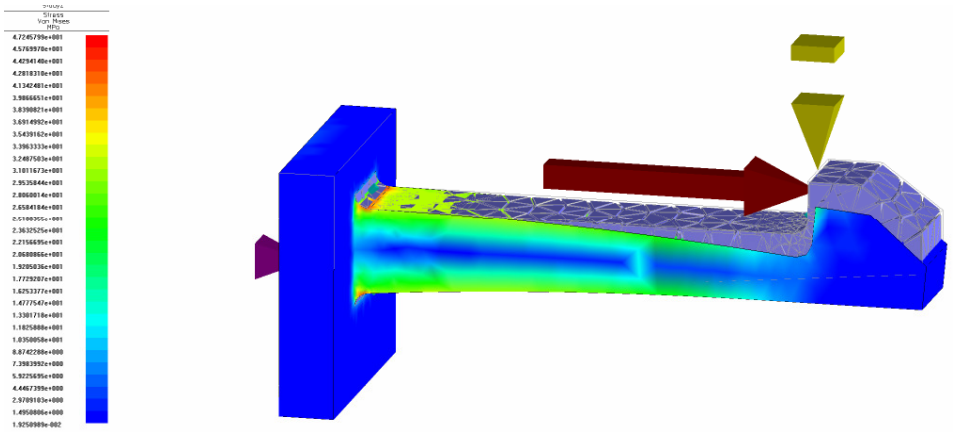


Abbildung 3.8: Biegeverhalten eines Teils unter Last

Dasselbe gilt auch für viele Simulationen, etwa das Erproben von Bewegungsabläufen oder Montage- und Demontageprozessen. Gerade der Bereich der Kinematik gewinnt hier zunehmend an Gewicht, da sich mittels rechnergestützter Simulation der zeit- und kostenintensive Bau von Prototypen reduzieren läßt.

Die Beispiele zeigen auch, daß die Übergänge zwischen Aufgaben der Berechnung, Analyse und Simulation fließend sind. So umfaßt eine Bewegungssimulation letztlich ebenso eine dynamische Kollisionsanalyse, wie zur Untersuchung des Schwingungsverhaltens wiederum umfangreiche Berechnungen notwendig sind. Ein weiterer Aspekt, der die Anwendung aller Bereiche erfordert, ist die ständige Überprüfung der Konstruktion auf gegebene Nebenbedingungen wie Fertigungsgerechtheit, Montagegerechtheit, Recyclinggerechtheit etc.

Einer Vielzahl dieser Anwendungen liegt eine gemeinsame Arbeitsweise zugrunde. Auch die prinzipiellen Anforderungen an die verwendeten Daten sowie der Umgang damit lassen Parallelen erkennen. In der Regel dient das geometrische Modell nur als Referenz. Es besteht also im Rahmen der ausgeführten Tätigkeiten keine Notwendigkeit, diese Kerndaten zu verändern – in vielen Fällen ist es sogar explizit untersagt, Änderungen daran außerhalb der Konstruktionsabteilung vorzunehmen. Ledig-

lich applikationsspezifische Daten müssen vom Bearbeiter hinzugefügt, geändert oder gelöscht werden. Diese Aufteilung der Daten wird noch offensichtlicher, wenn Konstruktion und Nachbearbeitung von verschiedenen Personen ausgeführt werden.

Viele Unternehmen verwalten die verschiedenen Dateneinheiten unter dem Dach eines gemeinsamen, rechnerinternen Objektmodells. Aus diesem werden geometrische und werkstoffliche Daten für Berechnungen und Analysen direkt bezogen, deren Ergebnisse werden ebenso direkt dort wieder zur Verfügung gestellt. Infolgedessen kann der Konstrukteur unmittelbare Konsequenzen aus den Resultaten ziehen und im Dialog seine Gestaltung anpassen, ohne dabei Daten neu eingeben zu müssen. So kann beispielsweise eine beabsichtigte Fertigungsoperation an komplexeren Teilen schon früh simuliert werden und basierend auf dem Resultat ohne Umwege eine fertigungstechnisch verbesserte Konstruktion erfolgen.

Sind approximative Ergebnisse ausreichend, weil etwa zur schnellen Überprüfung eines Iterationsschritts eine exakte Berechnung zu aufwendig wäre, so reicht es oftmals aus, vereinfachte Modelle oder Modellausschnitte zur Berechnung und Analyse zu verwenden. Zur näherungsweisen Bestimmung des Gewichts werden zum Beispiel häufig Kleinteile wie Schrauben, Scheiben und Muttern weggelassen – gegebenenfalls wird deren Anteil prozentual auf das Berechnungsergebnis aufgeschlagen. Ähnlich wird für geometrisch hochkomplexe Teile verfahren, die in stark vereinfachter Form – ohne Löcher, Rippen oder Rundungen – in den weiteren Prozeß eingehen.

Nachgeschaltete Arbeitsschritte werden im Laufe des Iterationsprozesses häufig mehrfach ausgeführt. Das Ergebnis einer Analyse führt zu einer Änderung an der Konstruktion, die im nächsten Schritt verifiziert werden muß. Sofern die vorausgehende Analyse auf einem Ausschnitt oder einer Vereinfachung des Modells durchgeführt wurde, ist es wichtig, diese Basis für den nächsten Durchgang effizient wiederherstellen zu können.

Es ist zu beobachten, daß die Zahl der Anwendungen zur Analyse, Simulation und Berechnung weiter steigt und die Anforderungen immer kunden- und produktspezifischer werden. Daher gewinnen Programmschnittstellen der CAD-Systeme zunehmend an Bedeutung. Unter der Voraussetzung, daß die zur Verfügung gestellte Funk-

tionalität ausreichend ist, erlauben sie eine Anpassung der Anwendungen an die jeweiligen Erfordernisse sowie eine tiefe Integration dieser Applikationen in den Konstruktionsprozeß.

3.3.4 Freigabe

Zum Abschluß der Gestaltungsphase, nach erfolgreicher Konstruktion und Optimierung des Produkts, erfolgt die Freigabe der Modelldaten. Zu diesem Zeitpunkt ist das Ergebnis des Iterationsprozesses von allen Verantwortlichen akzeptiert worden, weitere Iterationsschritte sind nicht geplant. Die Freigabe muß nicht notwendigerweise für das gesamte Modell in einem Zug erfolgen. Dies kann vielmehr auch komponentenweise geschehen.

Insgesamt ist der Freigabeprozeß aufwendig. Bevor das Modell freigegeben werden kann, ist sicherzustellen, daß alle an die Konstruktion gestellten Bedingungen und Anforderungen erfüllt sind. Darüber hinaus muß das Datenmodell konsistent sein, das heißt, einzelne Datenteilmengen dürfen keine sich widersprechenden Informationen beinhalten und alle Daten müssen auf dem aktuellen Stand sein. In der Praxis schließt dies normalerweise die Aktualität aller abgeleiteten Zeichnungen ein, worauf im folgenden Abschnitt noch eingegangen werden soll.

Nach der Freigabe werden Modelländerungen grundsätzlich nicht mehr zugelassen. Auf das Modell ist nur noch lesender Zugriff möglich, so daß darauf basierend zusätzliche Auswertungen, wie etwa das Erstellen der Dokumentation, weiterhin durchgeführt werden können.

In begründeten Ausnahmefällen kann von dieser Regel abgewichen und eine neuerliche Modelländerung zugelassen werden. In diesem Fall müssen mit dem Modell selbst jegliche Dokumente, die auf den ursprünglich freigegebenen Daten basieren, ihren Status ändern. Zur erneuten Freigabe sind alle Dokumente zu aktualisieren und der gesamte Prozeß muß nochmals durchlaufen werden. Der Aufwand für einen solchen Änderungsprozeß nach erfolgter Freigabe ist hoch und entsprechend fehleranfällig. Daher ist die Aufhebung des Freigabestatus in der Regel nur durch höhere Ebenen, zum Teil nur mit Billigung des Managements, möglich.

Infolgedessen wird die Notwendigkeit zur Aufhebung des Freigabestatus oft als inakzeptabel betrachtet, wenn sie bereits durch das Hinzufügen von Sekundärdaten, wie zum Beispiel Bemaßungen oder weiteren Analysen, gegeben ist. Dies kann der Fall sein, sofern 3D-Modell und Sekundärdaten sehr eng miteinander verknüpft sind. In der praktischen Anwendung treten solche Situationen auf, wenn beispielsweise durch Verlagerung der Produktion ins Ausland Zeichnungen ergänzt, das heißt, Texte übersetzt oder Maße umgerechnet werden müssen oder wenn bei der Montage festgestellt wird, daß zusätzliche Informationen wie Detailmaße benötigt werden.

3.3.5 Dokumentation

Um das neu entwickelte Produkt fertigen zu können, bedarf es einer ausführlichen technischen Beschreibung des Modells. Diese umfaßt neben der Festlegung von Materialien, die bereits während der Konstruktionsphase geschieht, vor allem eine fertigungsgerechte Bemaßung. Je nach Beschaffenheit des Produkts werden weitere Dokumente wie Explosionsdarstellungen, Montageanleitungen oder für den späteren Gebrauch entsprechend illustrierte Bedienungs- und Wartungshandbücher benötigt.

Das Erstellen dieser Unterlagen erfolgt üblicherweise nach Abschluß des Gestaltungsprozesses. Rückwirkungen auf die Konstruktion sind nur in seltenen Fällen möglich. Mit Ausnahme der Zeichnungsableitung werden die Arbeitsschritte oft auf Basis des freigegebenen Modells ausgeführt. Die Existenz der Zeichnungen dagegen ist vielfach gerade eine Voraussetzung zur Freigabe.

Ähnlich dem Vorgehen bei der Berechnung und Analyse dient das geometrische Modell auch hier nur als Referenz. Es besteht im Rahmen der ausgeführten Tätigkeiten keine Notwendigkeit, diese Kerndaten zu verändern – was bei freigegebenen Daten ohnehin nicht möglich ist. Der Bearbeiter muß lediglich dokumentationsspezifische Daten hinzufügen, ändern oder löschen.

Viele der erforderlichen Arbeitsschritte können auf vereinfachten Modellen oder Modellausschnitten erbracht werden. Bis auf wenige Ausnahmen ist es nicht erforderlich, das Gesamtmodell komplett zur Verfügung zu haben. Detaillierte Zeichnungen werden in der Regel nur für Einzelteile und Unterbaugruppen erstellt. Für Über-

sichtszeichnungen werden dagegen oft Kleinteile und Konstruktionsdetails ausgeblendet. Gleiches gilt für Illustrationen in Bedienungs- und Wartungshandbüchern. Aus Gründen der Reproduzierbarkeit ist es wiederum notwendig, auf Ausschnitte und Zusammenstellungen, die für die Anfertigung der Dokumentation gewählt wurden, effizient wieder zurückgreifen zu können.

Zeichnungsableitung, Bemaßung

Die Bemaßung des Modells einschließlich der Toleranzvorgaben kann sowohl am 3D-Modell erfolgen als auch auf dem klassischen Weg der 2D-Zeichnung. Dabei wird das Modell rechnerunterstützt in verschiedene Ansichtsebenen projiziert. Die eigentliche Bemaßung erfolgt danach im 2D. Im 3D bereits vorhandene Maße können in die 2D-Zeichnungen übernommen werden. In der praktischen Anwendung ist häufig eine Kombination aus beiden Verfahren anzutreffen: die Übernahme von Maßen aus dem 3D-Modell, die dann im 2D ergänzt werden.

Zur Definition der Zeichnung und deren Inhalten, wie Ansichten, Schnitten oder Detaildarstellungen, ist oftmals die graphische Repräsentierung des Modells, in einigen Fällen sogar die reine Baugruppenstruktur ausreichend. Unter Umständen kann es erforderlich sein, diese um die Box-Information von Einzelteilen zu ergänzen. Das bedeutet, daß zu jedem Teil die Position und Ausmaße eines minimal umschließenden, achsenparallelen Quaders bekannt sind. Auf diese Weise kann die Ausdehnung von Teilen oder Baugruppen in grober Näherung bestimmt werden.

Zur Berechnung der Zeichnungsableitung mit hinreichender Genauigkeit muß dagegen das vollständige 3D-Modell verfügbar sein. Dies gilt zumindest für Teile, die einen tatsächlichen Beitrag zur Zeichnung leisten. Im Regelfall ist das jedoch nur ein Ausschnitt des Gesamtmodells, zum Beispiel eine Unterbaugruppe oder eine Zusammenstellung von Teilen, die für eine Übersichtszeichnung benötigt werden. Bei geringeren Anforderungen an die Qualität der Zeichnung ist in der Regel auch die graphische Repräsentierung des Modells zur Ableitung ausreichend.

Montageanleitungen

Der Aufbau komplexer Baugruppen erfordert in der Regel eine detaillierte Beschreibung aller dazu notwendigen Arbeitsschritte. Dies kann durch statische, meistens in gedruckter Form vorliegende Montageanleitungen erfolgen, aus denen hervorgeht, in welcher Reihenfolge die einzelnen Teile zu kombinieren und welche Toleranzen, Anzugsdrehmomente etc. einzuhalten sind.

Insbesondere bei der Einzel- oder Kleinserienfertigung ist es jedoch unter Umständen sinnvoll, im Zuge der Montage direkt auf das 3D-Modell zuzugreifen, um Erkenntnisse für den Zusammenbau zu gewinnen. Bei dieser Form der dynamischen oder interaktiven Dokumentation muß das 3D-Modell mit der entsprechenden fertigungsrelevanten Information versehen sein. Auf eine weitere Bereitstellung statischer Anleitungen kann dann allerdings verzichtet werden.

Im Anwendungsfall sieht sich der Bearbeiter das 3D-Modell der zu montierenden Baugruppe am Rechner an, um Hinweise für deren Zusammenbau zu erhalten. Dazu benutzt er Funktionen zum dynamischen Betrachten einschließlich dem Ein- und Ausblenden einzelner Teile. Alle wesentlichen Informationen wie festgelegte Arbeitsschritte, Maße oder Toleranzen kann er dem Modell entnehmen. Damit dieses Vorgehen auf möglichst einfach ausgestatteten Rechnern möglich ist, wird vorher ein rein graphisches Modell zusammengestellt, das ausschließlich Informationen enthält, die für diesen Zweck notwendig sind.

Explosionsdarstellungen

Explosionsdarstellungen werden häufig als Bestandteil von Montageanleitungen verwendet. Mit ihrer Hilfe kann leicht illustriert werden, in welcher Reihenfolge einzelne Teile zu einer Baugruppe zusammengefügt werden sollen. In **Abbildung 3.9** wird eine Baugruppe in Explosionsdarstellung gezeigt.

Eine weitere Anwendung dieser Art der Illustration ist die übersichtliche Darstellung eines Produkts als Summe seiner Komponenten. Damit kann visuell verdeutlicht werden, in welche Hauptbestandteile sich eine Konstruktion untergliedert. Dieses Vorgehen findet in allen Hierarchieebenen der Baugruppenstruktur Verwendung. Der

Detaillierungsgrad steigt dabei mit zunehmender Tiefe an. Während bei der Unterteilung des Produkts in seine Hauptbaugruppen nur deren gestaltgebende Elemente von Interesse sind, gewinnen bei der Darstellung von Baugruppen der unteren Ebenen auch kleinere Einzelteile immer mehr an Bedeutung.

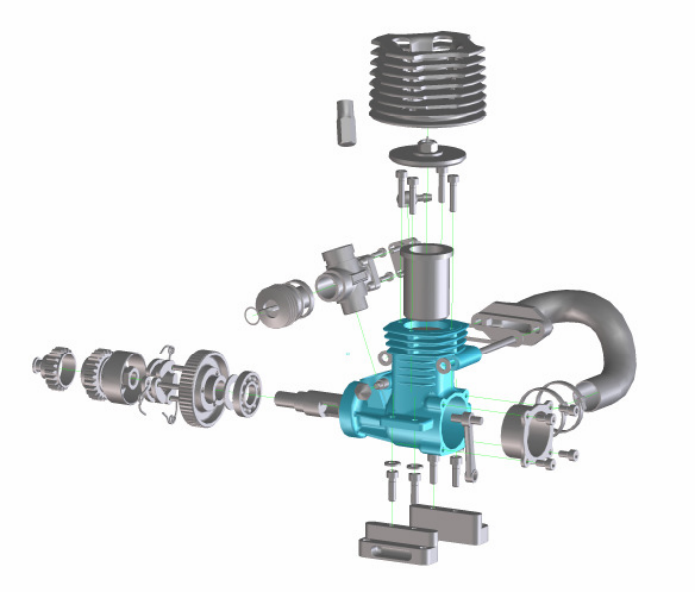


Abbildung 3.9: Explosionsdarstellung einer Baugruppe

Handbücher, Wartungsunterlagen

Zum besseren Verständnis werden auch Handbücher und Wartungsunterlagen mit Abbildungen des Produkts versehen, die direkt aus dem 3D-Modell ableitbar sind. Gezeigt werden häufig Modellausschnitte, die gerade soviel Information beinhalten wie für den jeweiligen Zweck notwendig ist. Dabei verzichtet die Darstellung auf irrelevante Details, zeigt aber gegebenenfalls bestimmte Einbausituationen und gibt Hinweise zur Montage oder Demontage, wie zum Beispiel beim Austausch der Toner-Kassette eines Druckers.

Zur Erstellung der Dokumentation greift der Autor direkt auf das 3D-Modell zurück, um verschiedene Ansichten, Einbauzustände oder Explosionsdarstellungen selbständig erzeugen zu können. Daraus leitet er die Abbildungen für die Handbücher ab. Für diese Tätigkeit ist das Grafikmodell in der Regel ausreichend. Der dazu erforderliche Funktionsumfang schließt neben Positionierung, dynamischer Betrachtung und dem Ausblenden nichtbenötigter Teile idealerweise Methoden ein, die das Anferti-

gen von Explosionsdarstellungen sowie die Simulation von Montageabläufen unterstützen. Letzteres ist insbesondere dann nützlich, wenn die Dokumentation durch kleine Filmsequenzen ergänzt werden soll.

Holzmodelle

Zur übersichtlichen Grobdarstellung der Konstruktion werden sogenannte *Holzmodelle* verwendet. Das sind graphische Modelle, die auf die wesentliche Gestalt und Funktion reduziert sind und auf Details weitestgehend verzichten. Solche Modelle werden beispielsweise bei der Layoutplanung von Fabriken oder Baustellen verwendet, um Aufstellplätze von Maschinen oder Kranen festzulegen. Darüber hinaus kann durch einfache Simulation geprüft werden, ob deren Aufbau unter den örtlichen Gegebenheiten möglich ist.

Eine weitere Anwendung ist der Einsatz als dynamisch anpaßbare Verkaufsunterlage. Sie kann genutzt werden, um das Produkts in einer virtuellen Demonstration seiner prinzipiellen Funktionen bereits so zu präsentieren, wie es später beim Interessenten eingesetzt werden soll, etwa in einer bestimmten Anordnung oder im Firmendesign des Kunden.

Die Anforderungen der Holzmodelle an Genauigkeit und Detaillierungsgrad sind normalerweise sehr gering, da nur die prinzipielle Gestalt und Funktion gezeigt werden soll. Vielmehr besteht die Aufgabe bei der Erstellung eines solchen Modells darin, die Ausgangskonstruktion möglichst effizient auszudünnen und zu vereinfachen. Dies geschieht vorwiegend durch Weglassen von Teilen, aber auch durch deren gezielte Vereinfachung. Die nachträgliche Ergänzung des Datenmodells durch konstruierte oder berechnete Simplifikationen muß daher möglich sein. Ebenso besteht die Notwendigkeit, die Zusammenstellung aller Komponenten eines Holzmodells einfach wiederherstellen zu können.

3.3.6 Fertigungsvorbereitung

Am Ende der Entwicklungsphase steht die Ergänzung weiterer fertigungsbezogener Informationen, bevor das virtuelle Produkt in die Realität umgesetzt werden kann. Dies sind zum einen Ergebnisse der Arbeitsplanung und anderer Bereiche, die keinen

unmittelbaren Bezug zum 3D-Modell haben und daher hier nicht näher betrachtet werden sollen. Zum anderen sind dies Daten, die hauptsächlich zur Automatisierung von Fertigung und Qualitätssicherung sowie für den Werkzeugbau benötigt werden. Diese können in der Regel direkt aus dem CAD-Modell abgeleitet werden oder stehen dazu in unmittelbarem Zusammenhang.

Als herausragendes Beispiel für die Automatisierung kann die NC-Programmierung für die eingesetzten Werkzeugmaschinen genannt werden. Hierbei besteht die Aufgabe, sowohl direkt aus der 3D-Geometrie der Teile als auch den angehängten Fertigungsfeatures Programme abzuleiten, die die Steuerung der Maschinen übernehmen. Zur Lösung dieses Problems existieren unterschiedliche Softwarepakete, die auf die benötigten Daten in geeigneter Weise zugreifen können müssen. Für die Erstellung von Prüfplänen und die Programmierung von Prüfmaschinen gelten vergleichbare Randbedingungen.

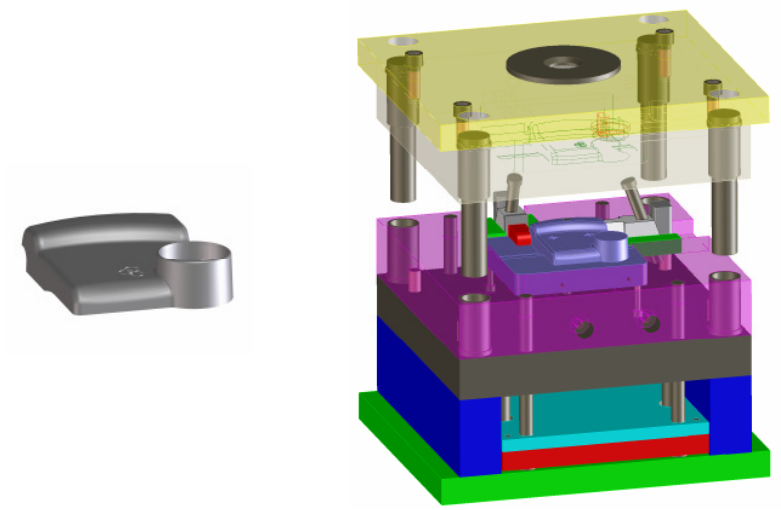


Abbildung 3.10: Plattenwerk für ein Spritzgußteil mit Ursprungsmodell

Der Entwurf von Werkzeugen kann mit Hilfe mengentheoretischer Verknüpfungen erfolgen, indem zunächst ein Überkörper zum entsprechenden Teil konstruiert wird, von dem das Teil dann abgezogen wird. Danach findet die Separierung des Werkzeugs durch Einziehen geeigneter Trennflächen statt, so daß das Werkzeug selbst wieder herstellbar ist. Bei der Konstruktion von Gußwerkzeugen ist überdies noch eine Skalierung zum Ausgleich des Schwindmaßes vorzunehmen. **Abbildung 3.10** zeigt ein Plattenwerk zur Herstellung eines Spritzgußteils. Zusätzlich ist das

Ursprungsteil dargestellt, von dem das Plattenwerk abgeleitet wird. Insgesamt kann der Bau eines Werkzeuges als eigenständiges Entwicklungsprojekt betrachtet werden, in welches das Modell des zu fertigenden Teils als Vorgabe analog zu Abschnitt 3.3.1 übernommen wird.

Die Tätigkeiten zur Fertigungsvorbereitung benötigen überwiegend Daten einzelner Teile des 3D-Modells in voller Genauigkeit. Diese dienen aber – wie bei anderen nachgeschalteten Aufgaben – nur der Referenz und werden selbst nicht verändert. Auch hier erfolgt lediglich eine Ergänzung des Datenmodells durch applikationstypische Daten.

3.3.7 Viewing, Collaboration

Mit zunehmender Verzahnung von Konstruktionstätigkeiten und sekundären Arbeitsschritten haben sich neue Anforderungen an Projektorganisation, Kommunikation und Datenaustausch entwickelt. Infolgedessen ist in den vergangenen Jahren ein neuer Markt für Werkzeuge entstanden, die die angepaßten Arbeitstechniken und Arbeitsweisen während des Entwicklungsprozesses integrativ unterstützen. Auf diesem Gebiet ist zur Zeit eine rege Entwicklungstätigkeit zu beobachten.

Der Einsatz dieser Werkzeuge findet hauptsächlich während der Konstruktionsphase und der damit verbundenen Berechnung und Analyse statt. Einen weiteren Schwerpunkt bildet die Kommunikation mit externen Entwicklungspartnern, die ebenfalls immer enger in den Prozeß eingebunden werden. Der übergeordnete Zweck dieser Werkzeuge ist die Unterstützung der abteilungs- oder standortübergreifenden Diskussion des aktuellen Entwicklungsstandes und sich daraus ergebender Folgeschritte. Im Gegensatz zu herkömmlichen Telefonkonferenzen steht bei kollaborativen Besprechungen ein 3D-Modell im Mittelpunkt, das von allen Beteiligten interaktiv bearbeitet und zeitgleich betrachtet werden kann. Dadurch wird erreicht, daß sich alle Konferenzteilnehmer während der Diskussion auf demselben Wissensstand befinden.

Aus Gründen eines geringeren Datentransfervolumens werden Modelle vorbereitet, die so ausgedünnt sind wie möglich, aber gleichzeitig soviel Information beinhalten wie notwendig. In der Regel ist dafür eine Zusammenstellung weniger Teile in rein

graphischer Repräsentierung ausreichend. Diese soll gegebenenfalls in mehreren Sitzungen wiederverwendet werden und muß daher auf einfache Art und Weise reproduzierbar sein.

Ein weiterer Grund für das Fortlassen der 3D-Geometriedaten kann der Schutz des geistigen Eigentums – *intellectual property* – sein. Insbesondere in der Diskussion mit externen Partnern ist oftmals lediglich ein Austausch über die Gestalt bzw. definierte Schnittstellen gewünscht, ohne die eigentlichen Konstruktionsdaten offenlegen zu müssen.

Design Reviews

Der häufigste Anwendungsfall für kollaborative Werkzeuge sind *Design Reviews*. Das sind Besprechungen des aktuellen Entwicklungsstandes sowie möglicher Änderungen, die oft auch interdisziplinär, das heißt team- oder abteilungsübergreifend, stattfinden. Gegebenenfalls sind die Teilnehmer über mehrere Standorte verteilt.

Während der Sitzung wird ein Modell dynamisch betrachtet, Maße werden bestimmt, eventuell werden Analysedaten diskutiert. Zur Verifikation von Schnittstellen kann es erforderlich sein, Teile oder Unterbaugruppen, die von verschiedenen Teams entwickelt wurden, in ein gemeinsames Modell zu laden. Außerdem kann der aktuelle Stand des Modells hinsichtlich Kollisionen, Standard-Konformität, Erfüllung des Pflichtenheftes etc. überprüft werden.

Getroffene Vereinbarungen werden, sofern sie unmittelbaren Bezug zum 3D-Modell haben, diesem direkt hinzugefügt. Dies erfolgt in Form von Features oder ähnlichen Strukturen, die zusammen mit dem Datenmodell verwaltet werden können. Dadurch hat jeder Teilnehmer die Möglichkeit, die Ergebnisse nach der Besprechung ohne erneute Eingabe in seine weitere Entwicklungstätigkeit einfließen zu lassen.

Externe Entwicklungspartner und Zulieferer

Ein weiteres Anwendungsgebiet ist die Einbindung externer Partner. Ein Beispiel dazu ist die Verhandlung mit einem möglichen Zulieferer über die Konstruktion und Fertigung einer Baugruppe. Bevor der Vertrag nicht unterschrieben ist, soll der

Zulieferer noch keine genauen Modelldaten erhalten. Zur Angebotserstellung und Herstellbarkeitsanalyse benötigt er aber exakte Geometrien einiger Kontaktflächen oder Schnittstellenteile, wie beispielsweise Befestigungselemente. In einer kollaborativen Sitzung, in deren Mittelpunkt ein graphisches Modell steht, welches die Einbausituation der Baugruppe darstellt und nur um ein Minimum an realer Geometrie ergänzt wurde, können Eckpunkte einer möglichen Zusammenarbeit besprochen werden. Der potentielle Zulieferer kann die Daten für die weitere Angebotsbearbeitung speichern.

Hat der Zulieferer mehrere Designvarianten erstellt, ist es nun Sache des Auftraggebers, einen Vorschlag auszuwählen. Dazu lädt der Entscheidungsträger in einer gemeinsamen Sitzung den relevanten Teil des eigentlichen Modells und fügt nacheinander die verschiedenen Designvarianten der zu entwickelnden Baugruppe ein. Diese können im Rahmen des Gesamtmodells auf Paßgenauigkeit, Form und Funktion überprüft und mit dem Zulieferer direkt diskutiert werden.

Erforderlich dazu ist ein schlanker Ausschnitt des Gesamtmodells, ergänzt durch einzelne exakte Geometrien, analog zu dem Modell, das in der Angebotssitzung verwendet wurde. Ähnliches gilt für die Designvorschläge. Auch hier kann der Anbieter seine Entwicklung bis zur endgültigen Vertragsunterzeichnung schützen wollen und zunächst nur graphische Modelle in die Sitzung einbringen. Typische Tätigkeiten in solchen Konferenzen sind dynamisches Betrachten, Messen oder das Durchführen von Kollisionsprüfungen.

3.4 Das 3D-Modell als Mastermodell für verschiedene Anwendungen

Ein *Produktmodell* enthält alle Informationen zur eindeutigen Beschreibung eines Produkts sowie Informationen darüber, wie aus diesen Daten weitere Informationen ermittelt werden können. Das können zum Beispiel Stücklisten oder NC-Programme sein. Als Obermenge aller expliziten und impliziten Informationen, die das Produkt und dessen Herstellung betreffen, stellt das Produktmodell eine wichtige Grundlage für die nachgeschalteten Produktionsbereiche dar, auf die bereits während des Ent-

wurfsprozesses zurückgegriffen werden kann. Seine zentrale Bedeutung im Rahmen des **C**omputer **I**ntegrated **M**anufacturing (*CIM*) wird in **Abbildung 3.11** verdeutlicht. Weiterreichende Betrachtungen zur Definition eines Produktmodells und zu Aspekten seiner Anwendung können unter anderem in [Ber02], [Lob02], [Pah90], [Schi02] und [Str00] nachgelesen werden.

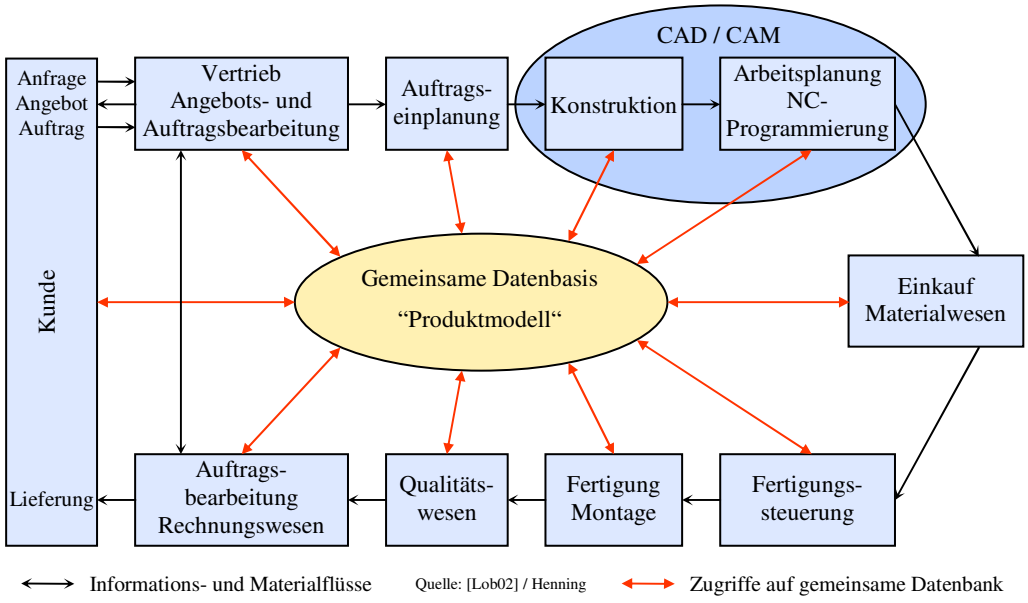


Abbildung 3.11: Integrierte Datenverarbeitung mit gemeinsamer Datenbasis

Eingebettet in ein Produktmodell stellt das *3D-Modell* den Kern aller expliziten, direkt vorhandenen Informationen dar. Von diesem Modell – häufig auch als CAD-Modell bezeichnet, da es die geometrische Festlegung des Produkts beschreibt – werden Daten für zahlreiche Anwendungen abgeleitet. Daten anderer Anwendungen stehen dazu in enger Beziehung. Maßgebend für die Gültigkeit dieser sekundären Daten ist immer der jeweilige Stand des CAD-Modells. Änderungen desselben erfordern somit eine Aktualisierung der damit in Verbindung stehenden Informationen. Insoweit sind die sekundären Daten vom 3D-Modell abhängig – das 3D-Modell dient als *Mastermodell*.

Die Speicherung der ergänzenden Informationen erfolgt je nach Applikation mittels Integration in das 3D-Modell oder in Form von separaten Dokumenten. Dementsprechend werden Schnittstellen benötigt, die einen wechselseitigen Datenzugriff ermöglichen. Zur Verwaltung des CAD-Modells sowie der sekundären Daten werden in der Regel PDM-Systeme verwendet.

In [CGM04] wurde die prozentuale Verteilung der PDM-Arbeitsplätze in Deutschland untersucht. Dabei ist ein abnehmender Trend beim Anteil der Konstruktionsarbeitsplätze bei entsprechendem Wachstum sonstiger Arbeitsplätze erkennbar. Diese Entwicklung läßt auch auf eine steigende Integration nachgeschalteter Prozesse schließen und betont dadurch die Relevanz einer engen Verbindung sekundärer Daten mit dem 3D-Modell.

3.4.1 Integration geometrischer und nichtgeometrischer Daten

Daten des CAD-Modells und nachgeschalteter Prozesse sind in hohem Maße voneinander abhängig. Dies gilt nicht nur für Anpassungen der Sekundärdaten, die als Folge einer Modellierung der 3D-Daten erzwungen werden, sondern ebenso für die Gegenrichtung. Beispielsweise kann eine Veränderung der werkstofflichen Daten eine Konstruktionsänderung notwendig machen. Daher ist eine Integration nichtgeometrischer Daten in das CAD-Modell erforderlich.

Prozeßbasierte Integration

Bereits bei der Definition nachgeschalteter Prozesse wird hierbei sichergestellt, daß geometrische und werkstoffliche Daten, die zur Ausführung der beschriebenen Tätigkeit notwendig sind, direkt aus dem rechnerinternen Objektmodell stammen. Darüber hinaus wird vereinbart, Ergebnisse auch wieder direkt dort zur Verfügung zu stellen. Dadurch wird der Konstrukteur in die Lage versetzt, direkte Konsequenzen daraus zu ziehen und im Dialog seine Gestaltung anzupassen, ohne dabei Daten erneut eingeben zu müssen.

Zur Unterstützung dieses Vorgehens sind automatisch generierte Änderungsmitteilungen, sogenannte *Change Notifications*, geeignet. Das sind Nachrichten, die vom PDM-System an definierte Adressaten verschickt werden, wenn sich ein Datensatz ändert. Dadurch werden Abhängigkeiten selbsttätig überwacht und die jeweiligen Empfänger der Mitteilung veranlaßt, in geeigneter Weise zu reagieren. Das kann programmatisch erfolgen, beispielsweise durch den zwangsläufigen Start einer Neuberechung von Analysen nach Modelländerungen oder im einfachsten Fall durch Versenden einer E-Mail an den Konstrukteur.

Datenbasierte Integration

In der Vergangenheit stand die reine Verwaltung der Ergebnisse von nachgeschalteten Tätigkeiten im Vordergrund. Aus der Sicht anderer Anwendungen lagen diese Ergebnisse als alphanumerische Daten ohne jede Semantik vor, so daß für diese Anwendungen eine weitergehende Auswertung oder Nutzung der Daten nicht möglich war. Ebenso war die permanente Aktualität aller Daten schwer sicherzustellen, da Abhängigkeiten nicht in ausreichendem Maße abgebildet wurden.

Inzwischen sind allerdings immer höhere Anforderungen an die Assoziativität von 3D-Modell und Daten nachgeschalteter Anwendungen festzustellen. Insbesondere von Applikationsdaten wird zunehmend erwartet, daß sie automatisch auf Modelländerungen reagieren. Das setzt eine enge Verknüpfung dieser Daten voraus, bietet aber auf der anderen Seite die Möglichkeit, das CAD-Modell flächendeckender als Mastermodell einzusetzen. Darüber hinaus wird eine höhere Datenintegrität gewährleistet.

Bei konsequenter Umsetzung einer engen Verbindung sekundärer Daten mit dem 3D-Modell können CAD-Systeme bereits während der Konzeptphase unterstützend verwendet werden. Während der Phasen des Entwerfens und Ausarbeitens wird der Konstrukteur durch Integration entsprechender Daten in die Lage versetzt, bereits fertigungstechnische Aspekte zu berücksichtigen, die umgekehrt wiederum durch die Konstruktion beeinflußt werden können. Dadurch wird das CAD-Modell zunehmend zum Rückgrat des gesamten Entwicklungsprozesses.

Am Beispiel einer statischen Berechnung, zu der basierend auf verschiedenen Lastannahmen mehrere Ergebnisse verwaltet werden müssen, wird deutlich, daß das Anlegen von Versionen und Varianten nicht nur für das 3D-Modell, sondern auch für sekundäre Daten erforderlich sein kann. Überdies werden viele Arbeitsschritte auf vereinfachten oder ausgedünnten Modellen erbracht. Daraus ergibt sich die Notwendigkeit, bei einer Reaktion auf eine Modelländerung auch die der ursprünglichen Berechnung zugrundeliegende Modellzusammenstellung wieder zu berücksichtigen.

3.4.2 Selektiver, applikationsspezifischer Datenzugriff

Zur Ausführung eines nachgeschalteten Prozeßschritts wird eine definierte, genau auf diesen Zweck abgestimmte Sicht auf das Produktmodell benötigt. Diese besteht in der Regel aus dem 3D-Modell bzw. einem Ausschnitt daraus und den anwendungsspezifischen Daten. Weitere Daten, insbesondere von anderen Applikationen, sind normalerweise nicht erforderlich.

Das 3D-Modell wird dabei im Regelfall nur zur Referenz benötigt. Dessen Veränderung ist nicht notwendig, außerhalb der Konstruktionsabteilung üblicherweise sogar unzulässig. Andererseits muß es dem Bearbeiter möglich sein, Daten seiner eigenen Applikation ohne Einschränkungen anzulegen, zu verändern und zu löschen. Dementsprechend soll ein Zeichner beispielsweise bereits freigegebene Teile mit weiteren Maßen oder Fertigungshinweisen ergänzen können. Ebenso soll er in der Lage sein, baugruppenübergreifende 3D-Maße anzubringen, auch wenn er kein Schreibrecht auf die Oberbaugruppe hat, die das Maß verwaltet.

Zusammenfassend läßt sich die Anforderung formulieren, daß jedem Bearbeiter genau die Teilmenge der Daten zur Verfügung stehen muß, die er für die Erfüllung seiner Aufgabe benötigt. Dabei ist sicherzustellen, daß ein unberechtigter Schreibzugriff auf Daten, die nur zu Referenzzwecken gebraucht werden, unterbunden wird. Aus Gründen der Effizienz soll darauf verzichtet werden, Daten zu laden, die für die Ausführung einer Tätigkeit nicht unmittelbar notwendig sind. Daraus ergibt sich die Forderung an die Datenverwaltung, eine entsprechend granulare Unterteilung zu unterstützen.

3.4.3 Offenheit

Für die Auswertung sowohl der 3D-Daten als auch der sekundären Daten ist es oft unzweckmäßig, das datengenerierende System starten zu müssen, um auf die Informationen zugreifen zu können. In einigen Fällen ist dies sogar unmöglich, da das verarbeitende System keine Integration in das bereitstellende aufweist. Vielmehr ist es bei der Verwendung heterogener Systeme erforderlich, die Informationen mit Hilfe von Schnittstellen oder standardisierten Dateiformaten offenzulegen und damit zugänglich zu machen.

Solche, speziell für den Datenaustausch verwendete Dateien werden im Regelfall zusätzlich zu den proprietären Formaten der jeweiligen Applikationen geschrieben. Je nach Anwendungsfall geschieht dies direkt aus der Applikation heraus oder als Folgeschritt unter Verwendung von Konvertierungsprogrammen. Beides kann automatisch in den Prozeß eingebunden oder nach Bedarf manuell erfolgen.

Für eine standardisierte Beschreibung von 3D-Geometrie werden verschiedene Formate verwendet, von denen IGES, SAT und STEP exemplarisch genannt werden sollen. Insbesondere STEP unterstützt überdies weit mehr als den Austausch reiner 3D-Daten. Vielmehr wurde bei der Definition dieses Standards zum Ziel gesetzt, die Informationen des gesamten Produktmodells abzubilden. Für die offene Beschreibung sekundärer Information wird in jüngerer Vergangenheit auch immer öfter XML eingesetzt. Dieser Standard ermöglicht zunächst nur die strukturierte Darstellung der einzelnen Informationseinheiten in generischer Weise. Daher muß eine Anwendung, die Daten in diesem Format bereitstellt, ein Schema definieren, wie die geschriebene Struktur zu interpretieren ist.

Die Möglichkeit des unmittelbaren Datenzugriffs ist besonders für die in Abschnitt 3.4.1 beschriebene prozeßbasierte Integration wichtig. Nicht zuletzt wird dadurch eine automatische Aktualisierung von beispielsweise Stücklisten oder Analysedaten in einer heterogenen Umgebung erst durchführbar. Weiterhin ergibt sich daraus die Flexibilität, in der Zukunft zusätzliche Applikationen einzusetzen, die auf dem existierenden Datenbestand arbeiten können, ohne in bestehende Anwendungen integriert werden zu müssen.

Neben dem Austausch der Modelldaten zwischen CAD- und anderen CAx-Systemen spielen auch Aspekte der Informationsbeschaffung eine Rolle. Dazu gehört ein informeller Zugriff auf Daten über Werkstoffe, Norm- und Zukaufteile ebenso wie auf bereits vorhandene Erzeugnisse. Liegen die entsprechenden Daten in standardisierter Weise offen vor, so kann deren Auswertung zum Beispiel mit dem Ziel erfolgen, höhere Wiederholraten von Teilen zu erreichen.

3.5 Speicherbedarf und Performance

Die verschiedenen Arbeitsschritte während des Entwicklungsprozesses unterscheiden sich mitunter stark hinsichtlich ihrer Anforderungen an Speicherausbau und Verarbeitungsgeschwindigkeit. Neben der Rechenintensität, etwa bei Kollisionsprüfungen oder Finit-Element-Berechnungen, ist hauptsächlich der Umfang der für die Bearbeitung notwendigen Baugruppen dafür ausschlaggebend.

In der Regel schlägt sich dies auch in der Hardware-Struktur der Entwicklungsabteilungen nieder. Während für Dokumentationsaufgaben oftmals Standard-Büro-PCs eingesetzt werden, finden in der Konstruktion und Berechnung eher High-End-Rechner Verwendung. Nach heutigem Stand der Technik sind das PCs, die mit 3 bis 4 GB Hauptspeicher bestückt sind und Prozessoren mit Taktraten von mehr als 3 GHz verwenden, oft als Doppelprozessor-Maschinen. Typischerweise kommt Windows XP als Betriebssystem zum Einsatz. Gerade bei der Entwicklung großer Baugruppen werden allerdings von einigen Unternehmen nach wie vor UNIX-Workstations eingesetzt. Jedoch auch hier läßt sich ein Trend zur Ablösung durch PC-Systeme erkennen.

In der Vergangenheit galt quasi die Regel, daß steigende Anforderungen an Speichermenge und Verarbeitungsgeschwindigkeit durch immer bessere Hardware kompensiert wurden. Mittlerweile ist mit dem beschriebenen Speicherausbau jedoch die Grenze des Adreßraums erreicht, der mit 32-Bit-Systemen zur Verfügung gestellt werden kann. Trotzdem reicht diese Speichermenge vielfach nicht aus, um große Baugruppen vollständig zu laden und zu bearbeiten.

Zur Lösung dieses Problems werden von den Entwicklungsabteilungen hauptsächlich zwei verschiedene Wege verfolgt. Zum einen wird ein Umstieg auf Hardware forciert, die auf 64-Bit-Prozessoren basiert. Dies geschieht normalerweise im Rahmen des turnusmäßigen Austauschs der Rechner. Die Möglichkeit dieser Systeme, einen weit größeren Adreßraum zur Verfügung stellen zu können, setzt jedoch voraus, daß die verwendete Software die 64-Bit-Architektur in entsprechender Weise unterstützt. Darüber hinaus wird gefordert, daß ein schrittweiser Umstieg vollzogen werden kann, da im Regelfall nicht alle Rechner gleichzeitig ausgetauscht

werden. Daher muß die Koexistenz von 32- und 64-Bit-Versionen der eingesetzten Programme unterstützt werden. Dies schließt insbesondere auch den Austausch von Dateien ein.

Der zweite Weg betrifft Arbeitsweisen, die im allgemeinen die Systemanforderungen verringern und im speziellen ein Erreichen der Speichergrenze verhindern sollen. Zur Bearbeitung wird dabei das Modell auf solche Bestandteile reduziert, die für die Ausführung einer Tätigkeit wesentlich sind. Dazu wird eine Auswahl der Teile so getroffen, daß irrelevante Teile gar nicht geladen werden. Weniger wichtige Teile oder Baugruppen werden durch vorher berechnete vereinfachte Modelle ersetzt.

Die Schwierigkeit dieses Vorgehens liegt darin, a priori entscheiden zu können, welche Teile für die spätere Bearbeitung unerläßlich sind – am Beispiel einer Zeichnung, welche Teile schließlich in der Darstellung sichtbar sind. In der Praxis wird deshalb die Auswahl oftmals viel zu umfangreich getroffen. Des weiteren wird die Reproduzierbarkeit einer solchen Konfiguration gefordert, da sie für wiederkehrende Tätigkeiten mehrfach benötigt wird.

3.6 Schutz von Informationen und Konstruktionsdetails

Konstruktionsdaten sind äußerst sensible Informationen. Das Bekanntwerden dieser Daten, sei es in Form einzelner Details oder ganzer Datensätze, verursacht für ein Unternehmen regelmäßig hohen Schaden. Dies gilt um so mehr, wenn das betreffende Produkt noch nicht am Markt verfügbar ist. Die Auswirkungen reichen vom Verlust des Informations- und Entwicklungsvorsprungs bis hin zur Verhinderung des patentrechtlichen Schutzes, wenn die Entwicklung vorher veröffentlicht wird.

Demgegenüber steht die Notwendigkeit, die Entwicklungstätigkeit auf verschiedene interne Abteilungen sowie externe Partner und Subunternehmer zu verteilen. Diese sind soweit in die Projekte einzubinden und mit Daten zu versorgen, daß ihre Ergebnisse nahtlos in den übergeordneten Entwicklungsprozeß integriert werden können. Damit steigt jedoch die Zahl derer, die berechtigten Zugriff auf entsprechende Informationen haben – und damit auch die Zahl potentieller Schwachstellen.

Zur passiven Absicherung des geistigen Eigentums – *intellectual property* – sind daher vertragliche Vereinbarungen selbstverständlich, die den exakten Umfang der Nutzung und Weitergabe aller Informationen regeln. Darüber hinaus werden von einigen Unternehmen aktive Maßnahmen gefordert, die sicherstellen sollen, daß jeder am Entwicklungsprozeß beteiligte ausschließlich die Daten zur Verfügung hat, die er für seine Zwecke benötigt. Alle anderen Informationen werden ihm vorenthalten und dadurch vor unberechtigter Verwendung geschützt.

In vielen Fällen wird auf diese Weise zumindest die Weitergabe des 3D-Modells vermieden. Dieses wird – sofern notwendig – durch ein graphisches Modell ersetzt. Oftmals werden dafür standardisierte Formate wie VRML verwendet, die allerdings einige Nachteile haben. So können beispielsweise viele Metadaten nicht adäquat übermittelt werden und eine genaue Angabe von Kontaktflächen oder Schnittstellenteilen ist nicht möglich. Ebenso wird die erreichbare Aufteilung der Daten in kleinere Einheiten häufig als zu grob betrachtet. Das heißt, daß aus datentechnischen Gründen immer noch mehr Informationen weitergegeben werden müssen als dies für die Bearbeitung der eigentlichen Aufgabe notwendig ist.

3.7 Randbedingungen

Die vorhandene Datenbasis ist ein Teil des Kapitals eines Unternehmens. Als Grundlage der bisher entwickelten Produkte ist sie auch über deren Markteinführung hinaus unter anderem relevant für Gewährleistungsfragen und Ersatzteilversorgung. Daher muß jede mögliche Änderung in der Datenverwaltung absolute Kompatibilität zu Bestandsdaten gewährleisten.

Dies gilt insbesondere beim Übergang von einer bestehenden Installation auf eine neue Programmversion, welche die im Rahmen dieser Arbeit zu entwickelnden Konzepte – und sei es teilweise – umsetzt. Grundsätzlich wird in größeren Unternehmen ein Versionswechsel eines CAD-Systems nicht quasi über Nacht vollzogen, sondern ist vielmehr ein Prozeß, der sich über mehrere Tage oder Wochen hinzieht, in denen ein Parallelbetrieb beider Versionen erforderlich ist. Die reine Installation der Software fällt dagegen in der Regel weniger ins Gewicht.

Nach formalen, oft automatisierten oder manuell nach detaillierten Plänen ablaufenden Tests mit der neuen Version erfolgt zunächst ein Probebetrieb. Weiterhin müssen Anpassungen des Systems aktualisiert werden, die kundenspezifisch angefertigt wurden. Erst nach erfolgreichem Probebetrieb wird die neue Version für den allgemeinen Einsatz freigegeben.

Der Probebetrieb wird nur von einer kleinen Gruppe von Konstrukteuren, allerdings unter realen Bedingungen, durchgeführt. Das bedeutet, daß die dabei erzeugten Daten in den laufenden Entwicklungsprozeß eingehen. Die Möglichkeit zur Weiterverarbeitung bereits vorliegender Daten wird dabei als selbstverständlich betrachtet. Neue Daten hingegen müssen in einem Format geschrieben werden, das von den anderen Anwendern, die noch die bisherige Programmversion im Einsatz haben, weiterverarbeitet werden kann. Im Einzelfall kann das heißen, daß eine neue Funktionalität, deren Daten nicht auf das bisherige Format abgebildet werden können, solange nicht benutzt werden kann, wie diese Kompatibilität erforderlich ist.

Schulungen finden, insbesondere bei großen Konstruktionsabteilungen, meistens gruppenweise statt. Nach deren Abschluß steigen die Konstrukteure dann direkt auf die neue Programmversion um. Für den sich daraus ergebenden temporären Mischbetrieb gelten die gleichen Kompatibilitätsbedingungen wie zuvor beim Probebetrieb. Erst wenn ausschließlich die neue Version eingesetzt wird, kann das aktuelle Dateiformat flächendeckend eingesetzt werden. Sind Zulieferer oder Konstruktionspartner in den Entwicklungsprozeß eingebunden, mit denen CAD-Daten direkt ausgetauscht werden und die ihrerseits auf die neue Version wechseln müssen, verläuft das Vorgehen analog.

3.8 Zusammenfassung

Die Entwicklung neuer Produkte ist heute ein Prozeß, bei dem Konstruktion und nachgeschaltete Aufgaben eng miteinander verbunden sind. Die Ausführung der erforderlichen Tätigkeiten erfolgt in iterativer Folge von Teilschritten des Entwurfs und der Ausarbeitung, deren Ergebnisse in den jeweiligen Folgeschritt eingehen.

Die abwechselnde Bearbeitung wird bis zum Erreichen des angestrebten Resultats fortgeführt. Zur Lösung der Aufgaben findet ein ständiger Informationsaustausch auch über Zuständigkeitsbereiche hinweg statt.

Zur Unterstützung dieses Prozesses werden CAD-Systeme eingesetzt, die sich zunehmend zu integrierten Systemen entwickeln. Ihre Aufgabe geht inzwischen weit über den ursprünglichen Anspruch hinaus, dem Konstrukteur allein den Entwurf eines virtuellen Produkts zu ermöglichen. Vielmehr wird von modernen CAD-Systemen ebenso die Bereitstellung bzw. Integration von Funktionalität erwartet, die die Ausführung nachgeschalteter, sekundärer Tätigkeiten erlaubt.

Integrität und Assoziativität

Die engere Verbindung von Konstruktion und nachfolgenden Tätigkeiten macht eine stärkere Verzahnung von CAD-Daten und technologischen Produktdaten erforderlich. Dadurch entstehen gegenseitige Abhängigkeiten, die permanent überwacht und gepflegt werden müssen. Im Zentrum dieser Daten steht das 3D-Modell, das dadurch zum Rückgrat des Entwicklungsprozesses wird.

In dessen Verlauf soll es möglich sein, für jeden Iterationsschritt die benötigten geometrischen und sekundären Daten direkt aus dem rechnerinternen Objektmodell zu beziehen. Ergebnisse der Tätigkeit sollen dort ebenso direkt wieder zur Verfügung gestellt werden können. Auf diese Weise kann der Konstrukteur unmittelbare Konsequenzen aus den Resultaten einer Analyse ziehen und seinen Entwurf anpassen, ohne dabei Daten erneut eingeben zu müssen. Gleiches gilt für Arbeitsschritte, die aufgrund einer Konstruktionsänderung notwendig werden.

Die Wahrung der Assoziativität soll vorzugsweise durch die beteiligten Daten selbst – mit Hilfe geeigneter Algorithmen – automatisch erfolgen, anderenfalls manuell durch die Bearbeiter, die über Änderungen entsprechend informiert werden. Dies kann beispielsweise durch automatisch vom PDM-System generierte *Change Notifications* geschehen. Die Integrität der Daten ist fortwährend sicherzustellen. Sich gegenseitig widersprechende Daten dürfen daher zu keiner Zeit vorhanden sein.

Aufgabenspezifische Segmentierung des CAD-Modells

Bei der Bearbeitung großer Baugruppen ist es in der Regel unzweckmäßig – in vielen Fällen sogar unmöglich – die Gesamtheit aller Daten im Rechner zur Verfügung zu haben. Vielmehr soll aus Gründen der Ressourcen-Optimierung nur derjenige Ausschnitt geladen werden, der zur Ausführung eines konkreten Arbeitsschritts tatsächlich erforderlich ist. Dies setzt eine Unterteilung der Daten in entsprechend kleine Einheiten voraus, die eine aufgabenspezifische Kombination zulassen.

Für Konstruktionsaufgaben übernimmt die Baugruppenstruktur die Funktion eines Grundgerüsts, das je nach Erfordernissen um exakte und informelle Geometrie sowie sekundäre Daten ergänzt wird. Teile, die keinen Beitrag zur Konstruktion leisten, werden nicht geladen. Zusätzlich benötigte Teile müssen positionsgenau nachladbar sein. Dasselbe gilt für den Austausch von informeller Geometrie durch exakte Geometrie. Dadurch wird das Bearbeiten eines beliebigen, baugruppenübergreifenden Modellausschnitts im Kontext des Gesamtmodells möglich.

Zur Durchführung vieler nachgeschalteter Arbeitsschritte ist ein noch weiter ausgedünntes Modell ausreichend oder, beispielsweise zum Schutz von Konstruktionsdetails, ausdrücklich erwünscht. Neben der Baugruppenstruktur und wenigen Teilen – oft genügt deren graphische Darstellung oder sogar deren kleinster umschließender Quader – sind nur die applikationsspezifischen Daten notwendig. Insbesondere Daten anderer Anwendungen, die ebenfalls eng zusammen mit dem 3D-Modell verwaltet werden, sind dagegen nicht erforderlich. Gerade an dieser Stelle wird die Aufteilung der Daten in kleinere Einheiten allerdings häufig als zu grob empfunden.

Das Zusammenstellen und die Pflege einer solchen aufgabenspezifischen Konfiguration kann mitunter aufwendig sein. Dies gilt um so mehr, je größer die Anzahl der Dateneinheiten ist, die kombiniert werden sollen. Möglichkeiten, diese einfach aufbauen, ergänzen und reproduzieren zu können, sind daher unabdingbar.

Zugriff auf Geometrie- und Applikationsdaten

Mit der Unterteilung der Daten in kleine Einheiten geht auch die Forderung nach entsprechend feiner Vergabe von Zugriffsrechten darauf einher. Die Bearbeiter

sollen nicht nur bei der Zusammenstellung einer für ihre Tätigkeiten möglichst optimalen Konfiguration unterstützt, sondern ebenso daran gehindert werden, auf Daten zuzugreifen, die sie nicht benötigen. Dies betrifft lesenden Zugriff auf Daten anderer Applikationen genauso wie schreibenden Zugriff auf Daten, die ausschließlich zu Referenzzwecken erforderlich sind, wie zum Beispiel das geometrische Modell.

Das Bearbeiten und Löschen aufgabenspezifischer Daten muß dagegen uneingeschränkt möglich sein – sofern diese noch nicht freigegeben sind. Besondere Anforderungen werden von einigen Unternehmen an das Hinzufügen neuer Daten gestellt. Dies soll auch dann noch möglich sein, wenn alle referenzierten Daten bereits freigegeben wurden. Eine Veränderung der bestehenden Datenbasis bleibt jedoch auch hier ausgeschlossen. Hintergrund dieser Forderung ist die nachträgliche Ergänzung fertigungsbezogener Daten. Diese kann unter anderem erforderlich werden, wenn durch Verlagerung der Produktion ins Ausland entsprechend angepaßte Fertigungsunterlagen zusätzlich erstellt werden müssen.

Durch die stärkere Integration der Applikationsdaten in das CAD-Modell ergibt sich außerdem die Notwendigkeit, auf anwendungsspezifische Informationen quasi „von außen“ zugreifen zu können, also ohne dazu das CAD-System starten zu müssen. Neben der klassischen Stücklistenverwaltung gilt dies insbesondere für Materialeigenschaften, Fertigungsdaten und Ergebnisse von Analysen, zum Beispiel Kollisionsuntersuchungen.

Räumliche Beziehungen von Teilen und Baugruppen

Eine wesentliche Voraussetzung für die effiziente Definition baugruppenübergreifender Modellausschnitte im Kontext des Gesamtmodells ist die Existenz von Selektionsmethoden, die auf räumlichen Beziehungen von Teilen und Baugruppen basieren. Zum Aufbau einer Konstruktionsumgebung ist es beispielsweise von Bedeutung, welche Teile sich in der Nachbarschaft des zu bearbeitenden Modells befinden. Das können Teile sein, die einen bestimmten Abstand dazu unterschreiten oder auch solche, die anschließende Bauräume begrenzen.

Auswahlmethoden, die lediglich auf der Teilehierarchie oder graphischen Darstellung beruhen, können dazu keine ausreichende Unterstützung bieten. Die räumliche

Anordnung der Teile innerhalb einer benachbarten, nicht von ihm erstellten Baugruppe ist dem Konstrukteur in der Regel nicht hinreichend bekannt, so daß die Möglichkeit einer unvollständigen Selektion besteht. Um dies zu verhindern, werden häufig wesentlich mehr Teile ausgewählt als eigentlich notwendig sind.

Die Auswahl über graphische Elemente verdeutlicht eine andere Problematik bei der Selektion im Zusammenhang mit der Verwendung von Modellausschnitten. Teile, die nur in der Baugruppenstruktur vorkommen, für die aber keine weiteren Daten geladen werden, sind naturgemäß auch nicht über graphische Elemente selektierbar. Wird jedoch zur Aufdeckung lokaler Interferenzen eine Kollisionsanalyse durchgeführt, können sie trotzdem von Interesse sein. Daraus ergibt sich implizit die Forderung, daß eine Selektion, die auf räumlichen Beziehungen basiert, auch solche, nur partiell geladene Teile berücksichtigen soll. Die virtuelle Gestalt dieser Teile kann für diesen Zweck jedoch grob angenähert sein.

Optimierung der Ressourcen

Große Baugruppen können auf 32-Bit-Systemen oftmals nicht mehr vollständig bearbeitet werden, da ihr Speicherbedarf den zur Verfügung stehenden Adreßraum übersteigt. Zur Vermeidung dieser Limitation werden in den Konstruktionsabteilungen zwei verschiedene Ansätze verfolgt. Zum einen werden die zu bearbeitenden Modelle den Speichergegebenheiten angepaßt. Dies geschieht, wie bereits diskutiert, durch Ausdünnung der Modelle oder durch Verfahren zur Modellvereinfachung, bei denen einzelne Teile oder ganze Baugruppen durch geometrisch vereinfachte Darstellungen ersetzt werden. Zum anderen wird gefordert, durch Unterstützung der 64-Bit-Architektur die Systeme so zu erweitern, daß sie den Anforderungen der Modelle Rechnung tragen.

Ein weiteres Ziel der Optimierung sind Bearbeitungszeiten. Insbesondere unproduktive Zeiten, die zur täglichen Vorbereitung der eigentlichen Tätigkeit erforderlich sind, stehen hier im Mittelpunkt des Interesses. Darunter fällt sowohl der zeitliche Aufwand, einen geeigneten Modellausschnitt für die jeweilige Aufgabe zusammenzustellen, als auch die nachfolgende Ladezeit. Damit wird die Zeitspanne beschrieben, die das System vom Beginn der Ladeoperation bis zu dem Moment benötigt, an dem der Benutzer weitere Kommandos ausführen kann.

4 Analyse bestehender 3D-CAD-Systeme

Der Umsatz von CAD/CAM- und CAE-Software in Deutschland belief sich laut einer Studie in [CGM04] auf etwas mehr als 900 Mio. Euro im Jahr 2003. Der mit Abstand größte Anteil dieses Umsatzes (über 60 %) entfällt auf die Mechanik-Branche. Dieser Bereich wird sowohl deutschland- als auch weltweit von den Anbietern Dassault Systemes / IBM, UGS (Unigraphics Solutions) und PTC (Parametric Technology) dominiert, gefolgt von SolidWorks, das zu Dassault Systemes gehört, aber nach wie vor unter eigenem Markennamen firmiert.

Ihrer Marktstellung entsprechend sollen die wichtigsten der von diesen Systemherstellern angebotenen 3D-CAD-Systeme in eine Bestandsanalyse einbezogen werden. Ergänzend wird das 3D-CAD-System von CoCreate hinzugenommen, das sich im Hinblick auf Marktanteile auf einem der Folgeplätze befindet, allerdings die Möglichkeit bietet, im Rahmen dieser Arbeit „hinter die Kulissen“ zu schauen und neue Konzepte exemplarisch zu implementieren.

4.1 Betrachtete Systeme

Bevor auf die zur Verfügung stehende Funktionalität zur Unterstützung der Handhabung großer Baugruppen im einzelnen eingegangen wird, sollen die für den Vergleich ausgewählten CAD-Systeme im folgenden kurz beschrieben werden. Die Reihenfolge, in der die Systeme genannt werden, entspricht dabei derjenigen der Marktanteile. Die Analyse der Systeme erfolgte während der Entstehungszeit dieser Arbeit und wurde aufgrund fortwährender Beobachtung ergänzt. Eine letzte Aktualisierung fand im März 2006 statt.

Catia V5

Catia wird von Dassault Systemes [3DS] entwickelt. Das System ist ein Hybridmodellierer, der Parametrik und Features ebenso unterstützt wie direkte Flächenmanipulation. Die Version 5 (V5) basiert auf einem von Dassault entwickelten Kernel, der völlig überarbeitet und im Jahr 1999 veröffentlicht wurde. Die aktuelle Version ist unter der Bezeichnung Catia V5R16 verfügbar.

Dassault wurde im Jahr 1981 gegründet und bezeichnet sich selbst als „Weltmarktführer von Product Lifecycle Management (PLM) Softwarelösungen“ (vgl. [3DS]). Catia wird dementsprechend als Teil der PLM-Angebotspalette positioniert und nicht als CAD-System. Als solches findet es weite Verbreitung unter anderem in der Luft- und Raumfahrt- sowie Automobilindustrie. Der Vertrieb des Systems erfolgt hauptsächlich über die Vertriebskanäle von IBM [IBM].

Unigraphics NX

Unigraphics NX ist ein featurebasiertes, parametrisches 3D-CAD-System aus dem Hause Unigraphics Solutions [UGS], das auf dem Parasolid Kernel aufbaut. Seit einiger Zeit bietet dieses System ebenfalls Methoden zur direkten Veränderung des CAD-Modells an. Die NX Serie ist das Ergebnis der Zusammenführung der CAD-Systeme Unigraphics und I-deas. Die Version NX 4 ist seit Anfang 2006 erhältlich.

UGS geht im wesentlichen aus der Fusion von EDS Unigraphics und SDRC im Jahr 2001 hervor. Das Unternehmen wurde 2004 aus EDS ausgegründet und ist heute eigenständig. Erste CAD-Installationen der ursprünglichen Unternehmen waren Mitte der 80er Jahre verfügbar. Das von SDRC eingebrachte CAD-System I-deas wird ebenfalls weiterentwickelt und unter der Bezeichnung I-deas NX vertrieben. Als weiteres System von UGS wird SolidEdge angeboten. Die beiden Letztgenannten sollen hier aber nicht weiter untersucht werden.

Pro/ENGINEER Wildfire

Pro/ENGINEER – kurz Pro/E – ist ebenfalls ein featurebasierter, parametrischer Volumenmodellierer. Dieser wird von Parametric Technology [PTC] entwickelt und verwendet den Granite Kernel, der eine Eigenentwicklung von PTC ist.

Das Unternehmen Parametric Technology stammt aus dem Jahr 1985, die Markteinführung von Pro/E erfolgte im Jahr 1988. Die Freigabe für die kommende Version, die unter der Bezeichnung Wildfire 3.0 auf den Markt kommen soll, ist für Ende März 2006 angekündigt.

SolidWorks

SolidWorks ist das 3D-CAD-System des gleichnamigen Herstellers [SWk]. Aufbauend auf dem Parasolid Kernel ist auch dies ein parametrisches System auf Featurebasis, das in der aktuellen Version 2006 erhältlich ist.

Das Unternehmen SolidWorks wurde im Jahr 1993 gegründet und 1997 von Dassault Systemes übernommen. SolidWorks tritt jedoch weiterhin als eigenständige Marke unter dem Dach von Dassault auf und adressiert in Ergänzung zum High-End-System Catia den Massenmarkt. Dementsprechend wird das System als „The world's #1 mainstream 3D design software" (vgl. [SWk]) vermarktet.

OneSpace Designer Modeling

Das 3D-CAD-System von CoCreate [CoC] wird unter der Bezeichnung OneSpace Designer Modeling vertrieben. Im Gegensatz zu den anderen Systemen ist dies ein nichtparametrischer Volumenmodellierer. Vielmehr zeichnet sich das System durch seine dynamische, historienfreie CAD-Architektur aus, die es einem Konstrukteur erlaubt, sein Modell quasi wie ein Stück Ton zu bearbeiten. Der Vorteil dieser Art des Modellierens liegt darin, daß Änderungen einfach am 3D-Modell durchgeführt werden können, ohne dessen Entstehungsgeschichte kennen zu müssen. Daraus ergeben sich für dieses System jedoch auch teilweise andere Anforderungen an die verwendeten Datenstrukturen. Die Software ist derzeitig als Version 2006 (14.0) verfügbar, die in Teilen auf Ergebnissen dieser Arbeit beruht.

CoCreate ist aus der Hewlett-Packard Mechanical Design Division hervorgegangen und seit dem Jahr 2000 eigenständig. OneSpace Designer Modeling basiert auf der Technologie des CAD-Systems SolidDesigner, das Anfang der 90er Jahre eingeführt wurde.

4.2 Funktionalität

Im folgenden sollen Funktionalitäten betrachtet werden, die die zu analysierenden CAD-Systeme bereits heute zur Unterstützung der Handhabung von Baugruppen bereitstellen. Dabei liegt der Fokus auf grundlegenden Konzepten, die insbesondere für große Baugruppen von Interesse sind. Gegenstand der Untersuchung ist ausdrücklich nicht ein 1:1-Vergleich einzelner Funktionen. Dieser würde den gegebenen Rahmen übersteigen und wäre zudem nicht zielführend.

4.2.1 Definition und Veränderung von Baugruppen

Strukturverwaltung

Methoden zum Aufbau von Baugruppen – bei Catia als „Produkte“ bezeichnet – und zur Verwaltung der Baugruppenstruktur werden von allen Systemen angeboten. Einschränkungen bezüglich der Teileanzahl und der Schachtelungstiefe einer Struktur bestehen dabei nicht. Einige Systeme erlauben ein einfaches Verschieben von Teilen und Unterbaugruppen innerhalb der Baugruppenstruktur mittels drag & drop.

Die Mehrfachverwendung von Teilen wird durchgängig unterstützt. Entsprechende Teile werden häufig als Exemplare oder Referenzkopien bezeichnet. Das gemeinsam verwendete geometrische Modell wird in einer Teiledatei abgelegt. Informationen zur Instanziierung der jeweiligen Exemplare in den Baugruppen werden in den zugehörigen Baugruppendateien gespeichert. Dies betrifft beispielsweise Namen sowie Lageregeln (Catia) resp. Transformationen (OneSpace Designer Modeling).

Unterstützung des Designprozesses

Einen Beitrag zur Konstruktion innerhalb eines Ausschnitts der Baugruppenstruktur – oft als *Design in Context* bezeichnet – leisten die Systeme durch die Möglichkeit, Konfigurationen zu spezifizieren oder Modelle lediglich partiell zu laden. Darüber hinaus können Baugruppen bei den meisten Systemen in eine Master-Baugruppe sowie ein ergänzendes Skelett untergliedert werden. Sowohl Konfigurationen als auch Skelett-Baugruppen werden im folgenden noch betrachtet.

Beim Aufbau von Baugruppen lassen sich zwei Vorgehensweisen unterscheiden. Zum einen findet der Aufbau von oben nach unten – *Top-Down* – Anwendung. Dabei wird zunächst eine Baugruppenstruktur vorgegeben, die durch weitere Hilfs- oder Referenzelemente ergänzt werden kann. Die Konstruktion der Teile erfolgt im Baugruppenkontext, gegebenenfalls mit Bezug zu anderen Teilen oder dedizierten Referenzelementen. Im anderen Fall werden zunächst die Teile konstruiert und später zu Baugruppen zusammengefaßt. Dieser Aufbau von unten nach oben wird mit *Bottom-Up* bezeichnet. Die CAD-Systeme erlauben beide Vorgehensweisen.

Eine neuere Entwicklung ist die programmatische Unterstützung bei der Verwaltung und Aufbereitung von Konstruktionswissen. Diese unter dem Begriff *Knowledge Based Engineering* zusammengefaßte Technologie wird momentan in der Hauptsache von Catia und Unigraphics NX angeboten.

Skelettmodelle

Elemente der Baugruppenstruktur, deren Aufgabe die Verwaltung von Schnittstellen sowie Hilfsgeometrien wie Ebenen, Achsen und 2D-Entwürfen ist, die dem Zweck der zentralen Referenzsteuerung innerhalb einer Baugruppe dienen, werden als *Skelett*-Baugruppe bezeichnet. Eine solche, anderen Ortes auch Entwurfsmodell genannte Struktur steht in direkter Beziehung zu einer Baugruppe, geht jedoch in der Regel nicht in die Stücklistenberechnung ein und ist oftmals mit eingeschränkten Privilegien zur Veränderung versehen, sofern sie genutzt wird, um eine bestimmte vorgegebene Konstruktionsabsicht zu dokumentieren.

Beim Top-Down-Entwurf von Baugruppen finden Skelettmodelle als zentrale Steuereinheit der Baugruppe Verwendung. Alle Referenzen der zugehörigen Komponenten werden auf Elemente des Skelettmodells bezogen. Dadurch können beispielsweise Bewegungen zentral definiert werden. Da alle essentiellen Referenzelemente, einschließlich etwaiger Bauraumdefinitionen, in gebündelter Form vorliegen, kann eine Baugruppe mit Hilfe ihres Skeletts zur Konzeptbetrachtung auf das Wesentliche reduziert werden. Die Einhaltung aller Schnittstellen und Bauraumvorgaben vorausgesetzt – beide können lokal überprüft werden –, ist die Sicherstellung modellweiter Konsistenz in einigen Fällen bereits auf der Basis der Skelettmodelle möglich.

Im Gegensatz zu den anderen CAD-Systemen bietet OneSpace Designer Modeling kein explizites Skelettmodell an. Jedoch existiert ein Container („Behälter“) genanntes Strukturelement in der Baugruppenstruktur, welches vergleichbare Aufgaben wahrnehmen kann.

Konfigurationen

Ein Aspekt der Konfigurierbarkeit von Baugruppen beschäftigt sich mit der Darstellung verschiedener Design-Varianten eines Teils oder einer Baugruppe. In der Regel werden in Zusammenhang mit dem jeweiligen PDM-System Methoden angeboten, die es ermöglichen, eine ausgewählte Zusammenstellung persistent und damit reproduzierbar zu machen. Diese Funktionalität wird mit Begriffen wie *Configuration Management*, *Assembly Arrangements* oder *Snapshot* umschrieben.

Einige der Konfigurationen sind an das Referenzschema innerhalb einer Baugruppe gebunden. Die Granularität der Auswahl beschränkt sich vielfach auf Teile und Baugruppen. Teilweise werden verschiedene Design-Varianten eines Teils bzw. einer Baugruppe innerhalb eines einzigen Dokuments verwaltet. Nicht benötigte Komponenten werden dann ausgeblendet und sind bei Bedarf automatisch oder manuell nachladbar.

4.2.2 Modellvereinfachungen, graphische Modelle

Die Vereinfachung von Modellen wird hauptsächlich unter zwei Gesichtspunkten verfolgt. Einerseits besteht darin eine Möglichkeit, den Ressourcenbedarf zu verrin-

gern und im Gegenzug die Verarbeitungsgeschwindigkeit zu steigern. Andererseits können dadurch zum Schutz des geistigen Eigentums Konstruktionsdetails unterdrückt werden, wenn Daten mit Dritten ausgetauscht werden müssen.

Auslassen von Teilen während des Ladeprozesses

Die CAD-Systeme bieten verschiedene Methoden an, einzelne Teile vom Laden auszuschließen. Catia stellt beispielsweise Selektives Laden zur Verfügung, womit eine Baugruppe lediglich bis zu einer vorgegebenen Strukturtiefe geladen wird. In Verbindung mit dem jeweiligen PDM-System ist es sowohl in Unigraphics NX als auch OneSpace Designer Modeling möglich, die Struktur einer Baugruppe vollständig zu laden, einzelne Teiledaten jedoch auszulassen und damit ausschließlich die Teile komplett zu laden, die tatsächlich benötigt werden. Dies erlaubt das Arbeiten im Kontext der Gesamtbaugruppe, jedoch beschränkt auf einen Ausschnitt davon. Dadurch bleiben sowohl Änderungen an der Struktur als auch die Berechnung von Stücklisten weiterhin möglich.

Eine detailliertere Auswahlmöglichkeit als auf Teileebene ist jedoch nicht gegeben. Trotzdem kann der Auswahlprozeß für eine solche Zusammenstellung umfangreich sein. Folgerichtig werden – den Konfigurationen vergleichbar – Methoden angeboten, die gewählten Ladezustände abzuspeichern. In OneSpace Designer Modeling kann dies so eingestellt werden, daß für den Ladeprozeß Regeln angewendet werden, die das Laden des jeweils aktuellen Zustands einer Baugruppe sicherstellen. Ergänzt wird diese Funktionalität durch die Möglichkeit zur Benachrichtigung des Anwenders für den Fall, daß Komponenten seiner Konfiguration von anderen Konstrukteuren verändert wurden.

Graphische Modelle

Neben der vollständigen Repräsentation der Teile, bestehend aus geometrischem und graphischem Modell, bieten die CAD-Systeme eine rein graphische Darstellung der Teile an. Für diese wird oft der Begriff des *Lightweight Modells* verwendet. Das geometrische Modell kann in diesem Fall – sofern die Daten verfügbar sind – nachgeladen werden. In der Regel geschieht dies, durch entsprechende Kommandos implizit

verursacht, automatisch. Zur Ausführung vieler Kommandos ist jedoch bereits die graphische Darstellung eines Modells ausreichend. Dies schließt die Möglichkeit einiger CAD-Systeme ein, 2D-Zeichnungen von dieser Darstellung abzuleiten.

Bei der Speicherung der Tesselierungsdaten werden unterschiedliche Strategien verfolgt. Während Catia diese Daten in separaten Dateien verwaltet, die auch von weiteren Baugruppen referenziert werden können, sind sie bei anderen Systemen Bestandteil der Modelldatei. Teilweise werden die graphischen Daten außerdem mit weiteren Informationen wie Layern oder Modellkanten angereichert oder durch verschieden genaue Detaillierungsstufen ergänzt. SolidWorks bietet die Möglichkeit, die Darstellung einer kompletten Baugruppe in der zugehörigen Modelldatei zu verwalten.

Vereinfachungen

Zur Verringerung des Ressourcenbedarfs sowie zur Vereinfachung der Übersicht in großen Baugruppen werden von den CAD-Systemen verschiedene Methoden zur Modellvereinfachung angeboten. Die Bandbreite reicht vom Auslassen der Modellhistorie über das Unterdrücken einzelner Details bis zum Ersatz von Unterbaugruppen durch stark reduzierte Bauraummodelle.

Die Architektur der featurebasierten Systeme bietet dabei die Gelegenheit, einzelne Features von der Regenerierung eines Modells auszuschließen. Details wie Bohrungen, Verrundungen oder Rippen können auf diese Weise leicht unterdrückt werden, das Modell wird entsprechend vereinfacht. OneSpace Designer Modeling kommt zu vergleichbaren Ergebnissen, aufgrund der direkten Modellrepräsentation jedoch basierend auf einer anderen Methodik. Bei diesem System werden zu unterdrückende Features manuell bestimmt oder anhand vorgegebener Bedingungen unter Verwendung von Heuristiken automatisch erkannt und explizit aus dem Modell entfernt.

Weitere Vorgehensweisen zur Modellvereinfachung beruhen auf der Verwendung von Ersatzmodellen. Einige Systeme bieten dazu die Erzeugung von Hüllkörpern an, die ein Teil umschließen und alle Details dadurch quasi verstecken. Ebenso kommen auf minimale Kriterien reduzierte geometrische Modelle zum Einsatz. Im äußersten Fall werden ganze Baugruppen durch sogenannte Konzeptblöcke ersetzt. Das sind Grundkörper wie Quader oder Zylinder bzw. einfache Verknüpfungen davon. Bei

Bedarf können jedoch kritische Bereiche exakt erhalten werden. Auf diese Weise wird ermöglicht, eine Baugruppe auf den Bauraum zu reduzieren, den sie maximal ausfüllen darf – unter Beibehaltung aller Schnittstellen in exakter Darstellung für eventuelle Anschlußkonstruktionen.

Wenngleich die CAD-Systeme verschiedene Methoden zur Wahrung der Assoziativität zwischen tatsächlichem und vereinfachtem Modell anbieten, bleibt die Verantwortung für die Übereinstimmung der Modelle letztlich doch beim Anwender. Die Unterstützung hierbei reicht von der Übernahme einiger Körpereigenschaften wie Masse oder Schwerpunkt bis hin zur (halb-) automatischen Regenerierung der vereinfachten Komponenten bei Veränderungen am Original.

Der Austausch vereinfachter Modelle gegen die korrespondierenden Ursprungsmodelle – und umgekehrt – ist in der Regel jederzeit möglich. Konfigurationen können dazu genutzt werden, Zusammenstellungen, die vereinfachte Komponenten enthalten, persistent zu machen. Ebenso kann dort verwaltet werden, welche Features von der Regenerierung des Modells ausgenommen und welche Teile bei der Darstellung einer Baugruppe ausgeblendet werden sollen.

4.2.3 Weitere Ressource-Optimierungen

Neben Methoden zur Vereinfachung und der Verwendung rein graphischer Modelle stellen die CAD-Systeme eine Reihe von Möglichkeiten zur Einstellung von Variablen zur Verfügung, die das Ziel verfolgen, den Ressourcenbedarf eines Modells zu verringern. Parameter zur Steuerung der Facettierungsgenauigkeit oder der Kantendarstellung beeinflussen die Geschwindigkeit des Graphikaufbaus ebenso wie die Entfernung kleinerer Teile aus der Darstellung bzw. deren ersatzweise Repräsentation mit Hilfe von Quadern. SolidWorks läßt darüber hinaus beispielsweise diverse Vorgaben zur Referenzverfolgung und -aktualisierung zu.

Mit der Unterstützung der 64-Bit-Architektur wird ein anderer Weg gegangen. Statt den Umfang des Modells hinsichtlich des Speicherbedarfs zu verringern, werden Fähigkeiten von Hardware und Betriebssystemen genutzt, um verfügbare Ressourcen optimaler ausschöpfen und damit die Bearbeitung größerer Modelle erreichen zu

können. Als Ergebnis der Entwicklung in der jüngeren Vergangenheit werden alle betrachteten Systeme inzwischen ebenso in einer 64-Bit-Version angeboten.

Überdies verfügen einige Systeme über Funktionalität, die automatische Bearbeitung von Aufgaben auf andere Rechner oder in die Zukunft zu verlagern. Die Ableitung von Zeichnungen sowie die Berechnung von Analysen auf dedizierten Servern bei der Verwendung von OneSpace Designer Modeling können dazu exemplarisch genannt werden. Ein weiteres Beispiel dafür ist der Scheduler von SolidWorks, der es erlaubt, Aufgaben wie die Regenerierung einer großen Baugruppe oder das Erstellen einer eDrawings-Datei zu einem festgelegten Zeitpunkt auszuführen.

4.2.4 Selektionsmethoden

Eine Vielzahl grundlegender Methoden zur Auswahl – etwa über den Strukturbaum, das graphische Modell oder Boxen – wird von allen Systemen angeboten. Einige Systeme stellen darüber hinaus weitere Arten der Selektion zur Verfügung, die auf Eigenschaften von Elementen oder logischen Zusammenhängen beruhen. Damit kann eine Auswahl zum Beispiel anhand der Größe eines Teils oder einer Farbkodierung erfolgen. Ebenso können Filter verwendet werden, um eine Selektion auf Komponenten zu beschränken, die zu einer Bibliothek gehören, oder die, wie etwa entsprechend gekennzeichnete Befestigungsteile, eine bestimmte Funktion übernehmen.

Logische Zusammenhänge werden ausgewertet, wenn Elemente gemeinsam selektiert werden, die explizit oder implizit in Verbindung stehen. Dies können sowohl Elemente einer vorher vom Anwender definierten Gruppe als auch beispielsweise alle Kanten einer Fläche sein. In die gleiche Kategorie fällt die rekursive Selektion aller Teile einer Baugruppe und deren Unterbaugruppen. Unigraphics NX bietet überdies die Möglichkeit, Komponenten in der Nachbarschaft eines Teils zu selektieren.

4.2.5 Unterstützung nachgeschalteter Prozesse

In Ergänzung zu den CAD-Systemen wird eine Vielzahl zusätzlicher Werkzeuge angeboten, die in der Ausführung nachgeschalteter Prozesse – wie der Finit-Element-Analyse, der Berechnung von Schweißverbindungen oder dem Formenbau – ange-

wendet werden. Einige dieser Werkzeuge sind abhängig vom CAD-System bereits in dieses integriert und werden damit zusammen ausgeliefert. Andere sind stark auf die Anforderungen des jeweiligen Systems zugeschnitten und stehen als zusätzliche Anwendung zur Verfügung.

Zur Wahrung der Assoziativität der von diesen Werkzeugen generierten Daten zum geometrischen Modell existieren verschiedene Ansätze. Weit verbreitet ist das Prinzip der Benachrichtigung der Benutzer über Modellveränderungen mit Hilfe von Change Notifications. In der Regel wird diese Funktionalität in Zusammenhang mit PDM-Systemen angeboten und stützt sich auf das Nachziehen von geometrischen Änderungen an den Applikationsdaten.

Catia bietet darüber hinaus weiterreichende Funktionalität zum Link-Management und zur Analyse von Beziehungen zwischen Dateien an. Darauf basierend kann die mögliche Reichweite der Auswirkungen einer Änderung in bezug auf Baugruppen, Zeichnungen sowie Analyse- und Fertigungsprozesse bestimmt werden, bevor diese ausgeführt wird. Außerdem stehen Methoden zur Verfügung, verschiedene Kopien derselben Baugruppe, die gegebenenfalls unabhängig voneinander verändert wurden, wieder zusammenzuführen. Unigraphics NX stellt ebenfalls Methoden bereit, die beispielsweise eine automatische Anpassung von Werkzeugdaten nach Modelländerungen ermöglicht. SolidWorks verwendet sogenannte Multi-Body Parts, um Schweißverbindungen zusammen mit dem Modell in einer Teiledatei zu verwalten, damit die Assoziativität direkt erhalten werden kann.

Für viele nachgeschaltete Prozesse ist die Bearbeitung von vereinfachten oder ausgedünnten Modellen ausreichend. Um diesem Umstand Rechnung zu tragen, ermöglichen die Systeme ebenso den Export von vereinfachten oder partiell geladenen Modellen in Formate wie STEP, IGES oder eDrawings zum Zwecke des Datenaustauschs zwischen verschiedenen Applikationen. Außerdem werden weitere Formate wie 3D-XML definiert, um die Möglichkeiten dieses Austauschs weiter zu verbessern. Die CAD-Systeme selbst sind beispielsweise in der Lage, bereits auf der Basis von ausschließlich graphisch geladenen Modellen schnelle, ungenauere Zeichnungsableitungen durchzuführen.

4.3 Gegenüberstellung von bestehender Funktionalität und Anforderungen

Nach eingehender Untersuchung der Anforderungen für die Handhabung großer Baugruppen und einem Überblick zu bereits existierenden Konzepten, die in verschiedenen CAD-Systemen realisiert sind, um diesen Bereich zu adressieren, sollen beide Seiten im folgenden gegenübergestellt werden. Ziel dieses kritischen Vergleichs von Anspruch und Realität ist die Identifikation von weiterhin vorhandenen Defiziten, die einer benutzergerechten Handhabung von großen Baugruppen im Wege stehen. Die Betrachtung orientiert sich dabei an den in Kapitel 3.8 beschriebenen Hauptanforderungen. Ein Fazit daraus wird im nachfolgenden Abschnitt gezogen.

Integrität und Assoziativität

Die zunehmende Verzahnung von CAD-Daten und technologischen Produktdaten stellt hohe Anforderungen an deren Integrität und Assoziativität. Beides soll vorzugsweise permanent automatisch sichergestellt werden. Sofern dies nicht vollständig gewährleistet werden kann, soll der Benutzer zumindest durch Change Notifications über Veränderungen informiert und auf einen möglichen Handlungsbedarf hingewiesen werden.

Die CAD-Systeme verwenden hauptsächlich zwei Arten der permanenten Speicherung von technologischen Produktdaten. Zum einen werden diese außerhalb des CAD-Modells in applikationseigenen Dateien gehalten, die häufig zusammen mit den CAD-Dateien unter der Kontrolle eines PDM-Systems stehen. Bei dieser Art der Verwaltung sind die Möglichkeiten zur Erhaltung der Assoziativität eingeschränkt. Änderungen des CAD-Modells müssen später für die Applikationsdaten nachgezogen werden. Dies gilt gegebenenfalls analog für die umgekehrte Richtung. Change Notifications zur Unterstützung dieses Prozesses sind jedoch in der Regel verfügbar. Catia bietet darüber hinaus Möglichkeiten, Abhängigkeiten frühzeitig zu erkennen.

Alternativ zur externen Verwaltung der Applikationsdaten werden diese in das CAD-Modell integriert. Dies geschieht entweder durch Programmschnittstellen (*APIs*) – dann meistens in Form von Features, die mit dem Modell verbunden werden – oder durch proprietäre Einbindung der Applikationen mit direktem Zugriff auf die CAD-Daten. Die Speicherung der Applikationsdaten erfolgt zusammen mit den zugehörigen Daten des CAD-Modells in einer gemeinsamen Datei. Ein hohes Maß an Assoziativität ist erreichbar, weil die Daten bei Modelländerungen verfügbar gemacht werden können.

Aufgabenspezifische Segmentierung des CAD-Modells

Aus Gründen der Ressource-Optimierung erfolgt häufig eine Ausdünnung der Baugruppen, bevor mit der eigentlichen Bearbeitung begonnen wird. Idealerweise arbeitet der Konstrukteur an einem baugruppenübergreifenden Modellausschnitt im Kontext des Gesamtmodells. Der Ausschnitt ist dabei so definiert, daß alle benötigten Daten zur Verfügung stehen, während alle anderen nicht geladen werden. Die Baugruppenstruktur dient als Grundgerüst, das bei Bedarf ein positionsgenaues Nachladen ermöglicht. Zum Schutz von Konstruktionsdetails müssen einzelne Dateneinheiten gezielt fortgelassen werden können. Da die Definition eines solchen Modellausschnitts komplex sein kann, ist dessen Aufbau und Pflege adäquat zu unterstützen.

Diesen Forderungen steht eine Reihe von Konzepten gegenüber, die sich mit der Reduzierung des Umfangs von Modellen befassen. Einige Systeme verfügen über die Möglichkeit, Baugruppen nur partiell oder bis zu einer gegebenen Tiefe zu laden. Mit Hilfe der Skelett-Technik kann ein Modell auf eine Konzeptbetrachtung reduziert werden und eine schlanke, schnelle Entwurfsumgebung darstellen. Die angebotenen Methoden zum Konfigurations-Management verfolgen unterschiedliche Ansätze vom Speichern einer Baugruppenzusammenstellung bis zur Auswahl verschiedener Designvarianten.

Dabei steht in der Regel eine Obermenge der Daten zur Verfügung, die kompakt in einer Datei verwaltet, jedoch nur zum Teil evaluiert wird. Dasselbe gilt für die Verwendung von Lightweight-Modellen, die bei einigen Systemen zusammen mit dem

geometrischen Modell in gemeinsamen Dateien gespeichert sind. Die Segmentierung der Daten beschränkt sich dadurch auf das laufende System und hat keine Auswirkung auf die Dateistruktur.

Der Aufbau von Modellausschnitten mit hinreichend detailliertem Umfang ist häufig nur mit größerem Aufwand zu erreichen, die Granularität der Auswahl ist an Teile und Baugruppen gebunden. Lediglich die Differenzierung zwischen graphischen Daten und Modelldaten stellt eine Verfeinerung dar. In der Folge werden die Ausschnitte daher oft größer gewählt, als dies zur Bearbeitung der jeweiligen Aufgabe notwendig wäre.

Zugriff auf Geometrie- und Applikationsdaten

Mit einer stärkeren Verzahnung von Geometrie- und Applikationsdaten gewinnt auch die Frage nach Zugriffsrechten und -möglichkeiten an Bedeutung. Nicht jeder Bearbeiter soll alle Privilegien für die gesamten Daten besitzen, vielmehr wird eine auf die jeweiligen Aufgaben zugeschnittene Zugriffskontrolle erwartet. Unabhängig von der Art der Integration der Applikationsdaten soll außerdem der direkte Zugriff auf diese Daten möglich sein, ohne ein CAD-System starten zu müssen.

Die heutigen Möglichkeiten hängen wiederum von der Art der permanenten Speicherung der Applikationsdaten ab. Werden diese außerhalb des CAD-Modells in applikationseigenen Dateien verwaltet, ist das Anlegen und Verändern zumindest prinzipiell unabhängig vom Status der CAD-Datei möglich. Dasselbe gilt für die granulare Vergabe von Zugriffsrechten. Ob ein Zugriff auf die Daten erfolgen kann, ohne ein CAD-System oder die jeweilige Applikation starten zu müssen, hängt jedoch von den verwendeten Dateiformaten ab.

Anders stellt sich die Situation bei Applikationsdaten dar, die in das CAD-Modell integriert sind. Der Zugriff auf die integrierten Daten hängt hierbei vom Status der CAD-Datei ab, eine feinere Vergabe von Zugriffsrechten ist nicht vorgesehen. Das heißt insbesondere, daß der Zugriff auf einzelne applikationsspezifische Datenausschnitte nicht unterbunden werden kann. Ebenso wenig ist ein direkter Zugriff von außen möglich.

Räumliche Beziehungen von Teilen und Baugruppen

Gerade bei großen Baugruppen ist die Forderung relevant, eine Konstruktionsumgebung unabhängig von der Baugruppenstruktur aufbauen zu können. Mangels exakter Kenntnis der Struktur angrenzender Baugruppen ist die Gefahr einer unzureichenden Auswahl, beispielsweise für eine Kollisionsanalyse, gegeben. Statt dessen sollen benachbarte Teile automatisch bestimmt werden – auch in einem lediglich partiell geladenen Modell, um das Nachladen dieser Teile zu ermöglichen.

Unigraphics bietet die Definition räumlicher Arbeitsbereiche innerhalb großer Baugruppen an. Ergänzend dazu ist die Selektion von Teilen in der Nachbarschaft verfügbar. Andere Systeme können lediglich Teile anhand ihrer Größe auswählen oder sind bei der Auswertung räumlicher Beziehungen auf den Kontext einer Baugruppe beschränkt.

Optimierung der Ressourcen

Die Anforderungen im Bereich des Ressourcenbedarfs konzentrieren sich darauf, die Ladezeiten zu verringern sowie zu vermeiden, daß die Obergrenze des verfügbaren Speichers erreicht wird. Letzteres resultiert sowohl in Forderungen nach dem Einsatz der 64-Bit-Architektur, die diese Grenze weiter nach oben verschiebt, als auch nach weiteren Vereinfachungsmöglichkeiten für Modelle, mit deren Hilfe der Speicherbedarf gesenkt wird.

Von der Verwendung stark vereinfachter Austauschbaugruppen über die Erzeugung von Hüllgeometrien bis hin zum Ausschluß einzelner Features von der Evaluierung gibt es eine Vielzahl von Konzepten, die sich der Vereinfachung von Modellen widmen. Weitere Funktionalität, wie das partielle oder selektive Laden bis zu einer gegebenen Strukturtiefe, dienen hauptsächlich der Verringerung des Ressourcenbedarfs eines Modells im allgemeinen, haben aber zumindest positive Nebenwirkungen auf die zum Laden benötigte Zeit. Gleiches gilt für den Einsatz von Lightweight-Modellen, deren Verwendung durch Methoden zum automatischen Nachladen benötigter Geometriedaten gefördert wird. Die Forderung nach 64-Bit-Versionen ist durch die seit kurzer Zeit herstellerübergreifende Verfügbarkeit dieser Systeme inzwischen obsolet geworden.

4.4 Fazit

Aufgrund ihrer Reife – die Software ist weit über 10, teilweise über 20 Jahre am Markt – bieten alle CAD-Systeme bereits weitreichende Unterstützung für die Handhabung von Baugruppen im allgemeinen. Viele Philosophien und Konzepte sind ähnlich oder haben sich im Laufe der Zeit angeglichen, Unterschiede finden sich hauptsächlich in Details. Grundlegende Funktionalität zur Bearbeitung großer Baugruppen im speziellen steht in allen Systemen zur Verfügung. Der Aufbau beliebig großer und komplexer Strukturen, Top-Down Design sowie die Anbindung an PDM-Systeme stellen eine gemeinsame Basis dar. Darüber hinaus sind jedoch verschiedene Ausprägungen und Schwerpunkte erkennbar.

Diverse Verfahren zur Modellvereinfachung und zur Erzeugung von Ersatzmodellen werden angeboten, um eine Reduzierung des Speicherbedarfs und eine bessere Übersicht in großen Modellen zu erreichen. Ebenso stehen Verfahren zur rein graphischen Darstellung zur Verfügung sowie Methoden, Masterbaugruppen zu laden und um benötigte Teile zu ergänzen. Konfigurationen werden genutzt, um Darstellungsarten und Zusammenstellungen geladener Teile reproduzierbar zu machen.

Eine optimierte Abstimmung solcher Modellsichten auf konkrete Aufgaben ist jedoch in der Regel nur mit größerem Aufwand zu erreichen. Die tatsächlich verwendeten Ausschnitte sind daher oft zu umfangreich. Die Granularität der Auswahl ist zudem auf Teile und Baugruppen beschränkt, lediglich bei der Art der Repräsentation kann weiter differenziert werden.

Daten des 3D-Modells und solche mit engem Bezug dazu – wie Features, das graphische Modell sowie einige Applikationsdaten – werden in der Regel gebündelt in CAD-Dateien gehalten. Dies bedeutet eine starke Integration der Daten und ermöglicht einen hohen Grad an Assoziativität, schließt aber gleichzeitig einen selektiven Zugriff aus. Ebenso können Zugriffsrechte nur auf Basis der CAD-Dateien vergeben werden, eine höhere Granularität der Vergabe, etwa auf der Ebene der Applikationsdaten, ist nicht vorgesehen.

Andere Applikationsdaten werden ergänzend in separaten Dateien mit Beziehung zur CAD-Datei verwaltet. Dadurch werden bessere Zugriffsmöglichkeiten mit feinerer Rechtevergabe erreicht, dies geschieht jedoch auf Kosten der Integration und Assoziativität. Die Möglichkeit auf Applikationsdaten von außen zuzugreifen, ohne das zugehörige Programmsystem starten zu müssen, ist dabei prinzipiell gegeben, hängt aber weiterhin von den verwendeten Dateiformaten ab.

In der Summe entsprechen beide bisherigen Wege der dauerhaften Speicherung von Applikationsdaten – die volle Integration in das CAD-Modell einerseits, die externe Verwaltung parallel dazu andererseits – nicht ausreichend den gestellten Anforderungen. Die notwendige Kombination aus gezieltem Zugriff sowie gleichzeitig hoher Integration und Assoziativität ist derzeit nicht realisierbar. Vielmehr muß der Anwender heute eine Entscheidung entweder für das eine oder das andere treffen.

Während die begrenzte Verfügbarkeit des Hauptspeichers durch 64-Bit-Versionen adressiert wird, die für die verschiedenen CAD-Systeme mittlerweile angeboten werden, besteht bei der Ladezeit noch Optimierungspotential und -bedarf. Das gilt auch für Selektionsmethoden, die bislang in der Hauptsache struktur- und graphikbasiert verwendet werden und mit einer Ausnahme räumliche Aspekte nicht ausreichend berücksichtigen.

5 Konzepte zur Handhabung großer Baugruppen

Im Verlauf dieser Arbeit wurden zunächst Anforderungen an CAD-Systeme identifiziert, die sich aus der Handhabung großer Baugruppen ergeben. Vor diesem Hintergrund wurde anschließend der derzeitige Stand der Technik anhand einiger CAD-Systeme analysiert. Die kritische Gegenüberstellung von Anforderungen einerseits und vorhandenen Konzepten andererseits läßt dabei insbesondere Defizite in der Art und Weise erkennen, wie geometrische und nichtgeometrische Daten heute verwaltet werden. Darüber hinaus werden Verbesserungspotentiale für Selektionsmethoden im Hinblick auf räumliche Zusammenhänge sowie für die Verringerung der Ladezeiten deutlich.

Die Diskrepanz zwischen dem Anspruch der Konstrukteure und dem Status quo in bezug auf die Verwaltung der Daten liegt darin, daß die Anwender eine hohe Integration und Assoziativität der Applikationsdaten *und* flexiblen Zugriff darauf benötigen, die CAD-Systeme ihnen aber nur entweder das eine oder das andere zur Verfügung stellen können. So wird in dem häufigen Fall, daß Applikationsdaten zusammen mit dem 3D-Modell in einer gemeinsamen Datei gespeichert werden, ein hoher Grad an Integration nur auf Kosten von mangelnder Flexibilität beim Datenzugriff erreicht. Ziel der nachfolgend vorgestellten Konzepte ist daher, durch eine feinere Segmentierung der Daten – und damit verbunden der Dateistruktur – beiden Forderungen gleichermaßen gerecht zu werden.

Bei der Entwicklung der Konzepte sind zwei grundsätzliche Randbedingungen zu berücksichtigen. Zum einen müssen die Ergebnisse Akzeptanz beim Benutzer finden können. Vorgeschlagene Technologien und Verfahren bringen demnach dem Anwender zusätzlichen Nutzen und unterstützen seine natürliche, intuitive Arbeitsweise. Ein eventueller Mehraufwand zugunsten der Möglichkeit, eine große Bau-

gruppe überhaupt bearbeiten zu können, kann dabei in Kauf genommen werden, ist aber weitestgehend zu minimieren. Zum anderen ist sicherzustellen, daß eine Umsetzung der Ergebnisse erreichbar ist, die die in Abschnitt 3.7 beschriebenen Forderungen nach Kompatibilität mit vorhandenen Datenbeständen gewährleistet.

Um den in praxi vielfältigen Vorgehensweisen bei der Konstruktion großer Baugruppen und den damit verbundenen Aufgaben und Prozessen Rechnung tragen zu können, soll im folgenden ein Kernkonzept entwickelt werden, das um weitere Detailkonzepte ergänzt wird, die verschiedene Facetten der Anwendungsbereiche adressieren. Im Fokus dieses Kernkonzepts liegt die Integration von technologischen Produktinformationen in ein CAD-Modell bei gleichzeitiger Erhöhung der Datengranularität. Dazu werden Algorithmen und Strukturen zur Datenseparierung als Schlüsseltechnologie eingeführt.

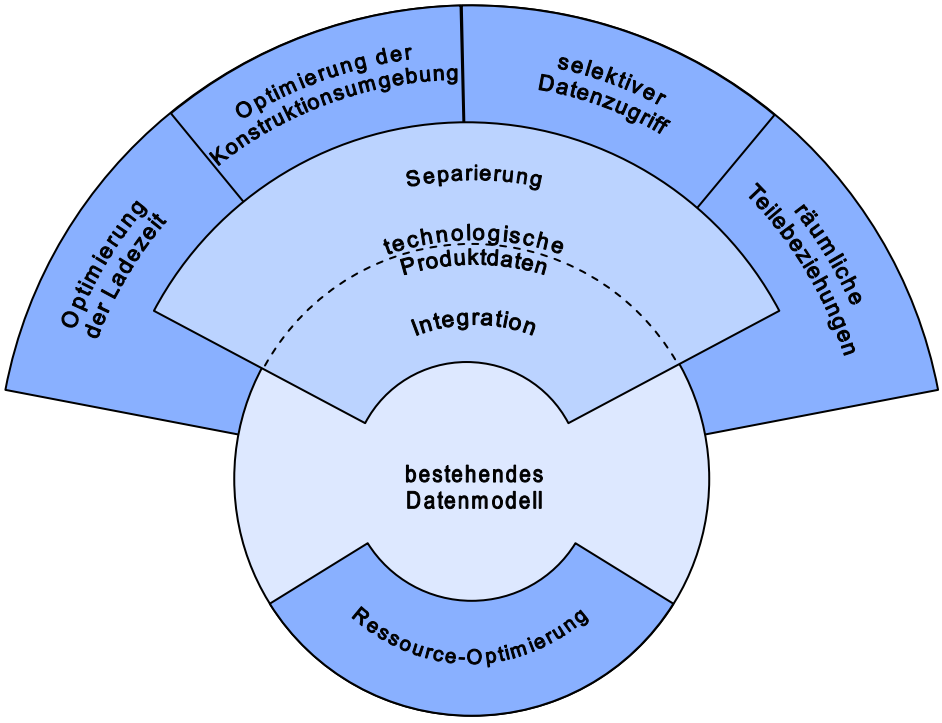


Abbildung 5.1: Konzepte zur Handhabung großer Baugruppen

Darauf basierend werden Detailkonzepte beschrieben, die sich der Optimierung von Ladezeit und Konstruktionsumgebung, dem selektiven Datenzugriff sowie räumlichen Teilebeziehungen widmen. Ergänzend dazu werden allgemeine Verfahren zur optimierten Ausnutzung von Ressourcen kurz skizziert. Der schematische Aufbau des Gesamtkonzepts wird in **Abbildung 5.1** veranschaulicht.

Die Konzepte sind so angelegt, daß sie je nach konkreten Erfordernissen baukastenartig kombinierbar sind. Der Schwerpunkt der Darstellung liegt dabei insbesondere bei den zur Separierung von Daten eingeführten Technologien auf den Datenstrukturen eines CAD-Systems und deren Anwendung. Sofern eine Verallgemeinerung auf verschiedene Systeme nicht zweckmäßig ist, werden die Vorgehensweisen und Datenstrukturen unter Berücksichtigung der Gegebenheiten im CAD-System OneSpace Designer Modeling beschrieben.

Auch wenn die vorgestellten Konzepte durch große Baugruppen motiviert sind, ist ihre Anwendung ebenso sinnvoll auf kleinere und mittlere Baugruppen übertragbar. Dies resultiert daraus, daß nicht nur Probleme betrachtet werden, die mit hohen Anforderungen an Ressourcen und Verarbeitungsgeschwindigkeit im Zusammenhang stehen – wie sie im besonderen bei sehr großen Baugruppen typisch sind –, sondern auch konzeptionelle Schwächen im Umgang mit Baugruppen adressiert werden. Letztere treten ebenso bei kleineren Baugruppen auf, fallen dort aber häufig nicht übermäßig ins Gewicht, da sie leichter zu umgehen sind.

5.1 Kernkonzept

CAD-Daten und technologische Produktdaten sind zwei wesentliche Bestandteile eines Produktmodells. Die enge Bindung beider Kategorien kommt durch eine Vielzahl assoziativer Beziehungen zwischen einzelnen Datenelementen zum Ausdruck. Aus Gründen der Pflege dieser Beziehungen sind daher viele Daten mit direktem Bezug zu einem 3D-Modell in dasselbe integriert oder stehen zumindest im CAD-System zur Verfügung. Beispielsweise können dies Daten der 3D-Bemaßung, der Struktur- oder Kollisionsanalyse sowie auch Materialeigenschaften etc. sein.

Technologische Produktdaten, die zusammen mit 3D-Daten in ein CAD-Modell integriert sind, werden heute mit diesen in gemeinsamen CAD-Dateien verwaltet. Dies hat zur Folge, daß sie nur en bloc gleichzeitig mit den 3D-Daten geladen werden können. Eine feinere Granularität als die der Dateistruktur, deren Aufbau in der Regel aus Teile- und Baugruppendateien besteht, kann nicht erreicht werden.

Insbesondere ist es nicht möglich, auf einzelne Dateneinheiten selektiv zuzugreifen, diese unabhängig voneinander zu bearbeiten oder freizugeben. Das Kernkonzept, als Lösungsansatz zu diesem Problem, soll daher durch den nachstehenden Leitgedanke ausgedrückt werden.

> *Integration von technologischen Produktdaten in ein CAD-Modell basierend auf einer hinreichend feinen, applikationsgerechten Segmentierung der Daten, die es dem Anwender erlaubt, exakt die für seine Aufgabe benötigte Kombination von Dateneinheiten zu laden und zu bearbeiten*

Integration markiert einen Eckpunkt dieses Leitgedankens. An vorderster Stelle ist die Integration der technologischen Produktdaten in ein CAD-Modell eine grundlegende Voraussetzung für ein hohes Maß an Assoziativität. Je umfangreicher und qualifizierter die Datenelemente sind, die bei Modellieroperationen zur Verfügung stehen und auf diese direkt reagieren können, um so gezielter lassen sich abhängige Daten aktuell halten. Gleiches gilt analog für die Umkehrrichtung, den Rückfluß von Berechnungsergebnissen in das CAD-Modell. Dadurch ist jene kontinuierliche Informationsbereitstellung erreichbar, die in integrierten CAD-Prozessen eine große Bedeutung hat (vgl. [PaBe03]).

Den anderen Eckpunkt des Leitgedankens markiert die *Segmentierung*. Damit wird eine weitere Unterteilung der Daten in kleinere, logisch zusammenhängende Einheiten beschrieben, deren Grenzen sich aus der Zuordnung zu verschiedenen Applikationen ergeben. Ein wesentliches Merkmal der Segmentierung ist die Übertragbarkeit dieser Unterteilung auf die Dateistruktur. Dadurch wird es möglich, daß eine Applikation die für ihre Erfordernisse benötigte Kombination von Dateneinheiten paßgenau laden kann. Dies läßt die Definition von feineren Partialmodellen zu, die eine Sicht des Produktmodells aus dem engen Blickwinkel eines bestimmten Anwendungsbereichs widerspiegeln.

Als Schlüsseltechnologie zur Subdivision des Modells in applikationsgerechte Einheiten wird die Datenseparierung verwendet, die im folgenden Abschnitt detailliert beschrieben wird. Die Ergebnisse einer feineren Segmentierung erlauben einen

besseren Zuschnitt der bereitgestellten Daten auf die zu lösenden Aufgaben. Darauf aufbauende Konzepte zur Optimierung der Konstruktionsumgebung sowie der Ladezeit werden in den Abschnitten 5.3ff. entwickelt.

Durch eine granularere Dateistruktur und damit einhergehende Vergrößerung der Zahl ladbarerer Objekte ergeben sich höhere Anforderungen an das Konfigurations-Management, denen entsprechend Rechnung zu tragen ist. Andererseits wird der Benutzer dadurch in die Lage versetzt, auf Dateneinheiten selektiv zugreifen und damit verbunden Zugriffsrechte gezielt vergeben zu können. Beide Möglichkeiten sind für Concurrent Engineering essentiell und werden in Abschnitt 5.6 näher betrachtet.

5.2 Separierung von Daten

Ein Hauptziel des übergeordneten Konzepts ist die Integration und Verknüpfung von geometrischen und nichtgeometrischen Informationen, um die Einbettung produktspezifischen Wissens in CAD-Modelle zu ermöglichen. Gleichzeitig soll die Forderung nach Austauschbarkeit von Informationen zwischen verschiedenen Anwendungen erfüllt werden. Zur semantischen Beschreibung des Modells ist es notwendig, daß neben der geometriebasierten Repräsentation weitere Darstellungen zur Verfügung stehen, die in der Gesamtheit verschiedene Partialmodelle und darauf aufbauende Sichten in ein CAD-Modell integrieren. Die Definition anwendungsspezifischer Partialmodelle – beispielsweise alle Informationen, die zur Beschreibung einer Kollisionsanalyse und ihrer Ergebnisse benötigt werden – erfordert ihrerseits eine entsprechend feingranulare Struktur dieser Darstellungen, um Aspekte der jeweiligen Applikation möglichst exakt abbilden zu können.

Viele technologische Produktdaten werden auf die in Abschnitt 2.4.2 erläuterte explizite Darstellung von Features abgebildet und zusammen mit ihrem Besitzer – in der Regel einem Teil oder einer Baugruppe – verwaltet. Die dort beschriebenen Dateigrenzen sind jedoch zu grob, um eine hinreichend granulare Struktur der Datenrepräsentierung im Sinne anwendungsspezifischer Partialmodelle zu erreichen. Dasselbe gilt in vergleichbarer Weise ebenso für weitere Datenstrukturen, die zur Abbil-

dung technologischen Produktwissens verwendet und zusammen mit den 3D-Daten verwaltet werden. Diese Bündelung von Struktur-, Geometrie- und Applikationsdaten in gemeinsamen Dateien, wie sie dem heutigen Stand entspricht, wird in **Abbildung 5.2** illustriert. Das graphische Modell wird nicht explizit gespeichert, sondern zur Laufzeit dynamisch vom geometrischen Modell abgeleitet. Der gezeigte prinzipielle Aufbau wiederholt sich für jede Komponente einer Baugruppe.

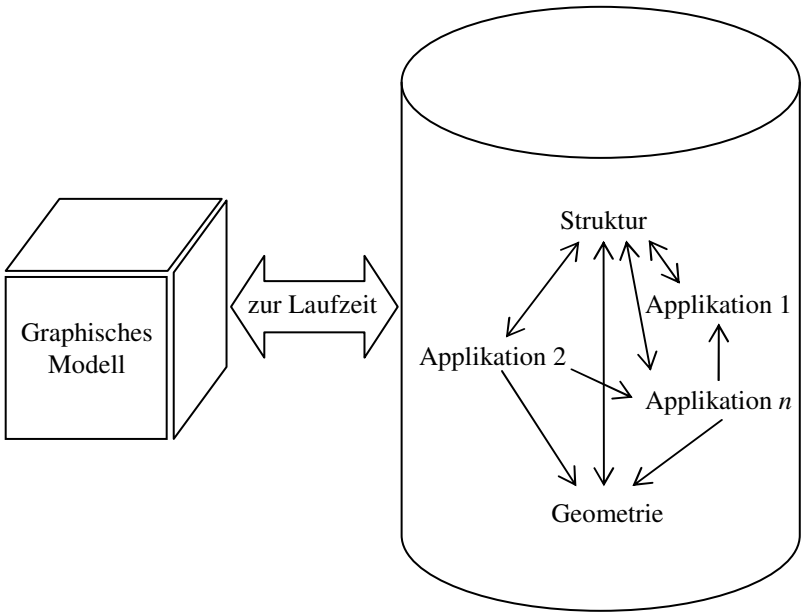


Abbildung 5.2: Bündelung aller persistenten Daten in einer gemeinsamen Datei

Die Abbildung verdeutlicht ebenfalls, daß der geforderte selektive Zugriff auf einzelne Dateneinheiten nicht möglich ist. Um dies zu erreichen, soll die Granularität sowohl auf der Ebene der Datenrepräsentation als auch für die zur Speicherung notwendige Dateistruktur erhöht werden. Dazu wird das Prinzip der Trennung von Modell- und Graphikdaten, wie es in einigen Systemen zur Darstellung sogenannter *Lightweight-Modelle* Anwendung findet, aufgegriffen und weiterentwickelt.

Der Kern des Vorgehens ist dabei, die Daten jeder Komponente der Baugruppenstruktur in handhabbare logische Einheiten aufzuteilen, die weitestgehend unabhängig voneinander und in verschiedenen Kombinationen geladen werden können. Die Unterteilung und Strukturierung der Daten soll eine Granularität der Informationseinheiten erreichen, die es für jeden Anwendungsbereich ermöglicht, alle erforderlichen Daten – ohne irrelevante Zusätze – auf elementare Weise bereitzustellen.

Gleichzeitig soll jedoch die Zahl der Informationseinheiten überschaubar und beherrschbar bleiben. Die technologische Bandbreite für einen Lösungsansatz bewegt sich zwischen den Extremen einer objektorientierten Datenbank auf Entity-Ebene und der heute verwendeten Bündelung der Daten.

Mit der Granularität der Dateistruktur – einhergehend mit der Anzahl der Dateien – wird ebenfalls der Aufwand für deren Verwaltung vergrößert. Dies gilt insbesondere auf Seiten der PDM-Systeme, deren Einsatz zur Erfüllung dieser Aufgabe vorausgesetzt werden soll. In [Kal97] wurde gezeigt, daß die Verwendung einer objektorientierten Datenbank auf Entity-Basis zur Verwaltung eines Netzwerks, das aus mehreren Millionen Entities für ein 3D-Modell bestehen kann, nicht zum Erfolg führt. Ebenso wurde die Unzweckmäßigkeit der Bündelung bereits diskutiert.

Im folgenden wird daher – quasi als Mittelweg – eine applikationsbasierte Separierung der Daten eines Clusters erörtert. Der bisher direkt zusammenhängende Datenverbund wird in einzelne Segmente unterteilt, die Repräsentationen für Struktur-, Geometrie- und verschiedene applikationsspezifische Daten beinhalten. Ergänzt wird eine Einheit, die das graphische Modell beschreibt. Die Einheiten unterhalten typischerweise Beziehungen untereinander. Ihre Daten werden in separaten Dateien persistent gemacht. **Abbildung 5.3** zeigt die Aufteilung der eingangs betrachteten Komponente einer Baugruppe in separierte Daten sowie die Abbildung der einzelnen Einheiten auf jeweils eigene Dateien.

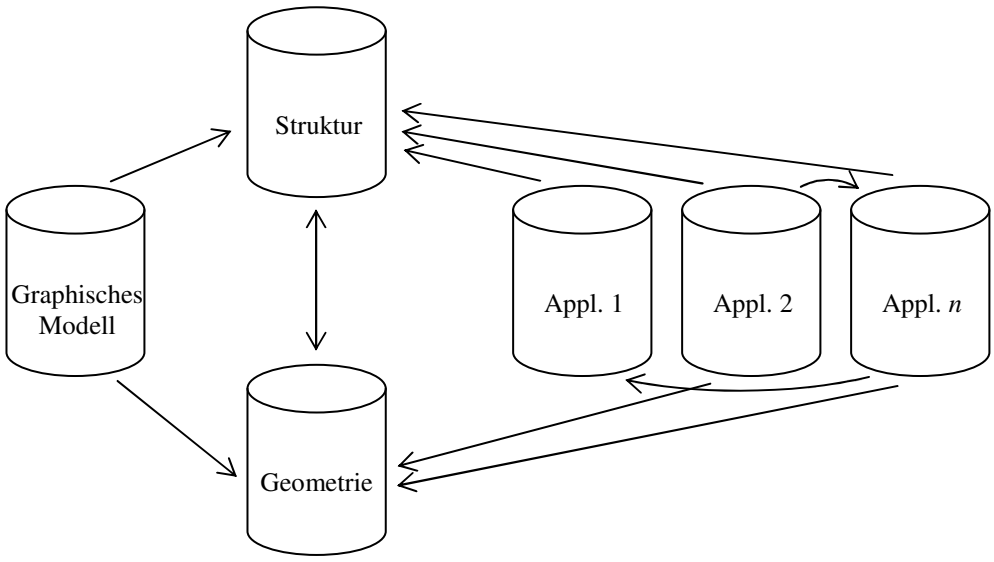


Abbildung 5.3: Separierte Daten

Beziehungen zwischen verschiedenen Einheiten der Strukturdaten legen die Baugruppenstruktur fest. Anschaulich wird dadurch das Rückgrat der Baugruppe definiert, alle anderen Einheiten sind von diesen quasi abhängig. Aufgrund dieser Eigenschaft kommt der Struktur-Einheit eine besondere Bedeutung zu. Auf die sich daraus ergebende Sonderstellung wird im folgenden noch näher eingegangen.

Daten des geometrischen Modells werden innerhalb eines Teils in ein eigenständiges Segment ausgegliedert. Struktur- und Geometrie-Einheiten haben eine enge Bindung zueinander. Die Eigenständigkeit der Geometrie-Einheit erlaubt die Darstellung verschiedener Varianten des geometrischen Modells – beispielsweise Vereinfachungen oder Einbauzustände – unterhalb eines Strukturknotens. Im laufenden System wird jedoch eine eindeutige Zuordnung vorausgesetzt, das heißt, eine parallele Darstellung verschiedener Varianten ist nicht vorgesehen. Allerdings wird ausdrücklich gestattet, daß einem Strukturknoten zur Laufzeit keinerlei geometrische Daten zugeordnet werden. Als Ziel von Referenzen aus Applikationsdaten ist das geometrische Modell eine treibende Einheit, deren Veränderung in der Regel zu Reaktionen in anderen Einheiten führt.

Das graphische Modell wird nicht mehr nur zur Laufzeit erzeugt, das heißt, vom geometrischen Modell abgeleitet, sondern zur Unterstützung rein graphischer Darstellungen in eine Datei abgespeichert. Die Kombination von Struktur- und Graphik-Daten entspricht dem Prinzip nach der Basis von Lightweight-Modellen. Ähnlich wie bei der Geometrie-Einheit sind auch hier verschiedene Varianten – Levels of Detail – in jeweiligen Dateien möglich. Sofern dem Strukturknoten eine Darstellung zugeordnet ist, soll ebenso von deren Eindeutigkeit im laufenden System ausgegangen werden.

In sich abgeschlossene Datenverbünde von Applikationen werden ebenfalls in Form eigenständiger Segmente ausgegliedert. Anders als bei geometrischen oder graphischen Einheiten ist es dabei jedoch möglich, mehrere Segmente von Applikationsdaten parallel im laufenden System zu halten. Dies können sowohl Einheiten aus verschiedenen Applikationen sein – Struktur-Analyse neben Bemaßung – als auch mehrere Einheiten desselben Typs – verschiedene Zeichnungsdefinitionen – nebeneinander.

Mit dieser Unterteilung wird bewirkt, daß exakt die Dateneinheiten geladen werden können, die für die Bearbeitung einer speziellen Aufgabe erforderlich sind. Dies umfaßt einzelne Einheiten der Struktur, Graphik und Geometrie sowie Daten aller involvierten Applikationen. Zur Aufrechterhaltung der Assoziativität können zusätzlich Einheiten abhängiger Applikationen ergänzt werden. Im Ergebnis wird dadurch das im Leitgedanke ausgedrückte Ziel vollständig erreicht. Bislang verwendete Funktionalität zum partiellen Laden[15] wird dadurch obsolet.

Die nachfolgenden Abschnitte beschreiben datenstrukturbezogene Aspekte der Separierung von Daten. Zunächst wird mit dem Subcluster-Konzept die Basis aller weiteren Strukturen hergeleitet. Darauf aufbauend wird im einzelnen die Separierung von Struktur-, Geometrie-, Graphik- und Applikationsdaten entwickelt.

5.2.1 Subcluster-Konzept

Die Separierung von Daten beruht auf der Unterteilung einer Dateneinheit in verschiedene, in sich abgeschlossene Untereinheiten, deren Inhalt zur permanenten Sicherung in jeweils eigenständige Dateien gespeichert wird. Das Subcluster-Konzept beschreibt Methoden und Datenstrukturen, die zur Erfüllung der dazu erforderlichen Aufgaben notwendig sind, und stellt damit das Kernstück der Datenseparierung dar. Ausgehend vom vorhandenen Cluster-Konzept, das in Kapitel 2.1.2 beschrieben wird, folgt im weiteren Verlauf die Entwicklung der Basis für eine Umsetzung der Separierung auf Entity-Ebene.

Zuvor soll anhand von **Abbildung 5.4** noch einmal der heutige Zustand ohne Separierung näher betrachtet werden. Die zu unterteilende Dateneinheit einer Komponente der Baugruppenstruktur entspricht einem Cluster[16], der alle Entities der verschiedenen Anwendungsbereiche umfaßt. In der schematischen Darstellung sind Entities, die inhaltlich zusammengehören und damit bereits eine gewisse implizite Segmentierung bilden, farblich unterlegt. Die Zahl der Applikationen, die jeweils ein eigenes Segment bilden, ist dabei nicht beschränkt. Über Local Master Beziehungen werden alle Entities in den Cluster integriert, wobei semantisch verbundene Ein-

[15] Vgl. Kapitel 2.5.1 Partielles Laden

[16] Vgl. Kapitel 2.3.4 Persistenz von Strukturelementen, Kapitel 2.4 Technologische Produktdaten

heiten häufig bereits über ein spezielles Entity verfügen, das die Einheit als solche integriert. Als Master des gesamten Clusters fungiert ein ausgezeichnetes Element der Strukturdaten.

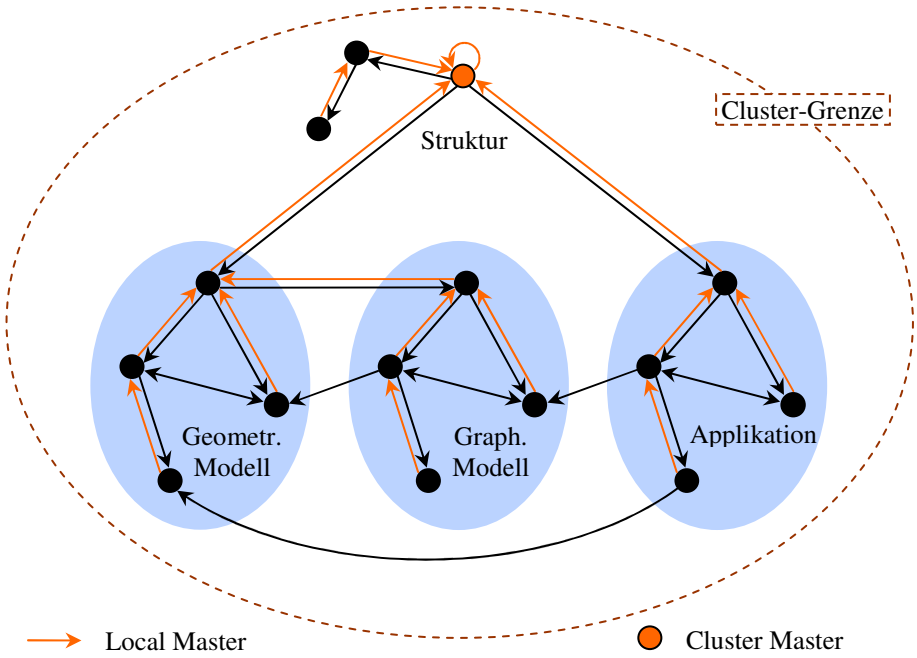


Abbildung 5.4: Entity-Beziehungen und Clusterzuordnung ohne Separierung

Eine wesentliche Randbedingung an die Entwicklung eines Konzepts zur Datenseparierung besteht in der Forderung nach Kompatibilität zu Altdaten, die in Kapitel 3.7 erhoben wird. Aus diesem Grund wird bewußt ein Ansatz verfolgt, der bereits bewährte Technologie aufgreift und durch möglichst geringe und gezielte Änderungen minimalinvasiv die gestellten Anforderungen erreicht.

Das existierende Cluster-Konzept definiert einen Cluster als logische, in sich abgeschlossene Einheit eines Entity-Netzwerks, deren Inhalt einer Datei zugeordnet werden kann. Die Zugehörigkeit einzelner Entities zu einem Cluster wird über Local Master Beziehungen bestimmt, die rekursiv fortgesetzt zum Repräsentanten eines Clusters, dem Cluster Master, führen. Durch rekursive Anwendung desselben Prinzips und mit Hilfe von integrativen Cluster-Beziehungen kann eine weitere Unterteilung des Clusters in logische Teileinheiten erreicht werden, die jedoch weder vollständig sein muß, noch für die übergeordnete Cluster-Struktur von Bedeutung ist.

Ein Vergleich der charakteristischen Eigenschaften des Cluster-Konzepts mit den Anforderungen, die sich aus der Separierung von Daten ergeben, läßt weitreichende Parallelen erkennen. Vor dem Hintergrund dieses Konzepts soll daher der Begriff des *Subclusters* eingeführt werden, der eine semantisch verbundene Untereinheit eines Clusters definiert, die selbst den Anforderungen an einen Cluster genügt. Darüber hinaus soll gefordert werden, daß die Unterteilung eines Clusters in Subcluster vollständig erfolgen kann, um für jedes Entity eine eindeutige Zuordnung zu einem Subcluster und damit zu einer Datei zu erreichen.

In bezug auf die in **Abbildung 5.4** dargestellte Entity-Struktur bedeutet dies eine Organisation der einzelnen Segmente, die Geometrie-, Graphik- und Applikationsdaten beschreiben, in Form von Subclustern. Folglich ist für jedes dieser Segmente das Erfüllen der Anforderungen an einen Cluster zu gewährleisten. Aufgrund der Forderung nach Vollständigkeit müssen die Strukturdaten ebenfalls zu einem Subcluster zusammengefaßt werden. Dadurch wird der prinzipielle Aufbau eines Clusters auf die in **Abbildung 5.5** gezeigte Struktur verändert.

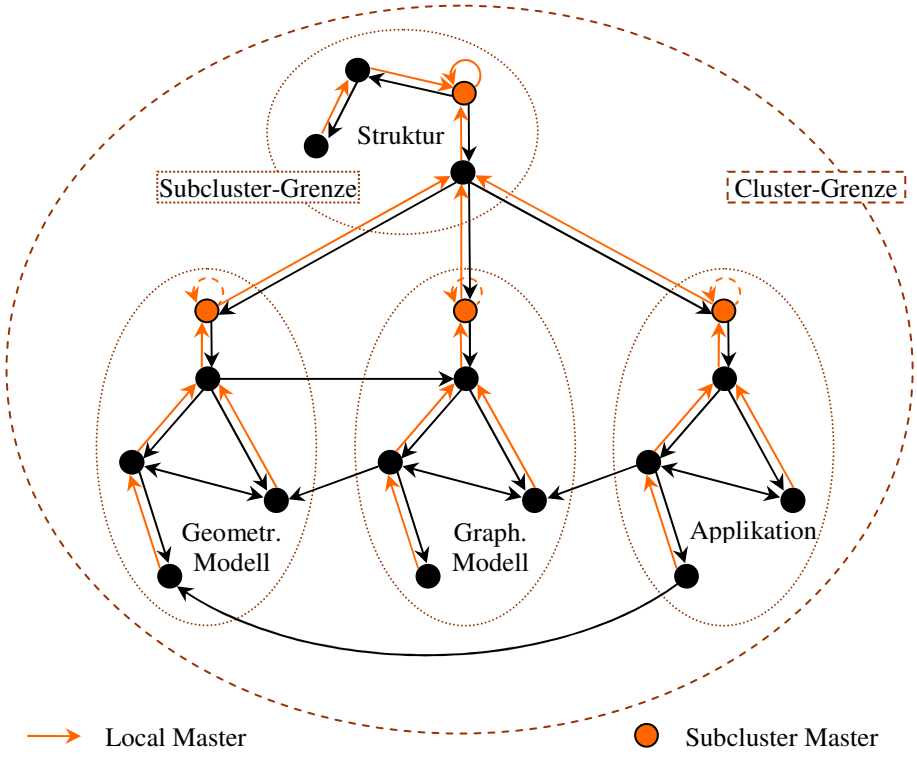


Abbildung 5.5: Entity-Beziehungen unter Verwendung von Subclustern

Bedingt durch die geforderten Cluster-Eigenschaften verfügt jede semantisch verbundene Einheit jetzt über einen eigenen Repräsentanten, den *Subcluster Master*. Dieses Entity muß zu der vorhandenen Datenstruktur gegebenenfalls addiert werden. Alle Subcluster Master stehen mit dem Struktur-Subcluster in direkter Beziehung. Ein ausgezeichnetes Entity dieses Subclusters dient allen Repräsentanten weiterer Subcluster als Local Master. Dadurch wird die logische Struktur des Clusters – das heißt, die Zusammenfassung aller Informationen dieses Clusters als Einheit – erhalten. Der Master des Struktur-Subclusters übernimmt dadurch automatisch die Funktion des Repräsentanten für den gesamten Cluster.

Das soweit skizzierte Subcluster-Konzept dient als generische Grundlage für alle Typen von Subclustern. Weitere Einzelheiten des Konzepts, charakteristische Eigenschaften und Datenstrukturen werden im folgenden detailliert beschrieben.

Charakteristika von Subclustern

Eine wichtige Eigenschaft eines Clusters ist dessen Abgeschlossenheit. Auf die separierten Dateneinheiten übertragen bedeutet dies für jeden Subcluster die Möglichkeit, ausgehend vom Subcluster Master, alle Entities des Subclusters auf Pfaden innerhalb desselben erreichen zu können. In umgekehrter Richtung muß der Subcluster Master auf einer Kette von Local Mastern, die alle innerhalb des Subclusters liegen, erreichbar sein. Der Algorithmus zur Traversierung aller zu einem Subcluster gehörenden Entities ergibt sich demnach analog zum Cluster-Algorithmus in 2.1.2 wie folgt.

1. Starte mit dem Subcluster Master
2. Bearbeite das Entity gemäß Aufgabenstellung
3. Besuche jedes Nachbar-Entity und stelle fest, zu welchem Subcluster es gehört
    a. Wenn es zum selben Subcluster gehört, das heißt, wenn sein Subcluster Master mit dem Startpunkt der Suche übereinstimmt, führe den Algorithmus mit der Bearbeitung des Entities fort (2.)
    b. Wenn es zu einem anderen Subcluster gehört, hat die Suche die Subcluster-Grenze überschritten. Dieses Entity wird nicht weiter prozessiert und die Suche wird von diesem Knoten aus nicht weitergeführt.

Grundsätzlich ist jedem Subcluster eine eigene Datei zugeordnet. Ist diese Zuordnung – gegebenenfalls aus Gründen der Kompatibilität – im Einzelfall nicht zweckmäßig, kann der Repräsentant des Subclusters auf seine Master-Eigenschaft verzichten und statt dessen eine integrative Beziehung mit dem Struktur-Subcluster eingehen. Dadurch werden automatisch alle Entities des Subclusters dem Struktur-Subcluster zugeordnet und mit diesem zusammen abgespeichert.

Beibehaltung der Cluster-Eigenschaft

Die Aufteilung eines Clusters in Subcluster ist vollständig. Infolgedessen gibt es kein Entity innerhalb des Clusters, das nicht einem Subcluster zugeordnet ist. Im Gegensatz zu einem Cluster Master – der gleichzeitig als dessen eigener Local Master auftritt – verweist die `local_master()`-Methode eines Subcluster Masters auf ein ausgezeichnetes Entity innerhalb des Struktur-Subclusters. Dadurch wird die übergeordnete Struktur auf Clusterebene unabhängig von der Verwaltung der Daten innerhalb des Clusters erhalten. Dies ermöglicht weiterhin die clusterbasierte Orientierung im Netzwerk der Modelldaten.

Beim Schreiben von Dateien wird die Local Master Beziehung von einem Subcluster Master zum Struktur-Subcluster dagegen nicht ausgewertet. Sie wird vielmehr so behandelt, als verwiese die `local_master()`-Methode, wie für den Repräsentanten eines Clusters definiert, auf den Subcluster Master selbst und gewährleistet dadurch die Eigenständigkeit der separierten Einheit. Dies wird in **Abbildung 5.5** durch eine unterbrochene Linie dargestellt. Der Informationsgehalt eines Clusters wird somit nicht mehr auf eine einzelne Datei abgebildet, sondern durch die Summe der Subcluster-Dateien dargestellt.

Der Struktur-Subcluster verwaltet alle anderen Subcluster in einem Cluster. Da der Master dieser Untereinheit gleichzeitig die Rolle des Repräsentanten für den gesamten Cluster übernimmt, kommt dem Struktur-Subcluster im Rahmen der Baugruppenverwaltung eine besondere Rolle zu, auf die in Abschnitt 5.2.2 noch näher eingegangen werden soll. In diesem Sinne werden die Begriffe Struktur-Subcluster und Cluster im weiteren gleichbedeutend verwendet.

Referenzen zwischen Subclustern

Die enge Verbindung zwischen den Subclustern einer Komponente der Baugruppenstruktur erfordert die Referenzierbarkeit einzelner Entities auch über Subclustergrenzen hinweg. Neben den in Abschnitt 2.1.4 beschriebenen internen und externen Referenzen werden daher zusätzlich solche zwischen Subclustern im selben Cluster zugelassen. Da Subcluster aus verschiedenen Arbeitssitzungen stammen können, sind diese Referenzen beim Speichern wie externe Verweise zu behandeln.

Besonderer Betrachtung bedürfen Referenzen zwischen dem Struktur-Subcluster und weiteren Subclustern, da diese den grundlegenden Aufbau des Subcluster-Verbunds beschreiben. Im Hinblick auf die Referenz-Beziehungen an dieser Nahtstelle kann das Subcluster-Konzept als eine Synthese aus hierarchischen Beziehungen – zwischen dem Cluster und seinen Subclustern – und integrativen Beziehungen – jedes Subclusters mit dem Cluster als solchem – verstanden werden.

Die integrative Referenz eines Subcluster Masters auf den Struktur-Subcluster *muß* abgespeichert werden. Dadurch ist es möglich, schreibgeschützten Clustern nachträglich Subcluster hinzuzufügen. In diesem Fall kann eine Referenz auf den Subcluster wegen des Schreibschutzes nicht persistent gemacht werden. Daß der Cluster jedoch über die Referenzdaten eindeutig bestimmbar ist, erlaubt, den Subcluster nach dem Laden dort wieder anzumelden und die Beziehung wiederherzustellen. Insofern wird eine lose Kopplung von Subclustern an den Struktur-Subcluster erreicht.

Eine Referenz von einem Cluster auf den Master eines Subclusters hingegen *kann* abgespeichert werden – sofern der Cluster nicht schreibgeschützt ist. In diesem Fall besteht beim Laden des Clusters die Möglichkeit, die abhängige Subcluster-Datei automatisch mitzuladen, da die Referenz alle erforderlichen Informationen wie Pfad und Name der Datei enthält. Anderenfalls ist es die Angelegenheit des Benutzers, abhängige Dateien mit in den Ladeprozeß einzubringen. In der Regel wird er diese Aufgabe an das verwendete PDM-System delegieren.

Beim Speichern von Referenzen innerhalb eines Subclusters kann die Speicheradresse des jeweiligen Zielobjekts als eindeutige Kennung zur späteren Zuordnung beim Laden verwendet werden, da zum Zeitpunkt des Abspeicherns alle Entities des Subc-

lusters im Hauptspeicher sein müssen. Dies entspricht dem Verfahren zum Speichern interner Referenzen, welches im Rahmen des Cluster-Konzepts Anwendung findet. Die Konvertierung interner Referenzen muß jedoch pro Subcluster erfolgen. Da Subcluster eines Clusters in verschiedenen Arbeitssitzungen erstellt und bearbeitet werden können, kann von clusterweit eindeutigen Speicheradressen nicht mehr ausgegangen werden.

Referenzen, die aus einem Subcluster in andere Cluster oder Subcluster verweisen, werden wie externe Referenzen behandelt. Die Referenzdaten beinhalten die eindeutige SysID des Zielobjekts sowie den Dateiname und weitere Informationen über den Zielcluster. Verweise auf einen Subcluster können durch zusätzliche Subcluster-Informationen ergänzt werden.

Parallelität und Eindeutigkeit von Subclustern

Subcluster repräsentieren die Daten verschiedener Typen von Anwendungen. Eine Folge davon sind unterschiedliche Anforderungen hinsichtlich der Eindeutigkeit der Untereinheiten innerhalb eines Clusters. Diesbezüglich können Subcluster zwei Kategorien zugeordnet werden.

Zum einem gibt es Anwendungen, zu denen auch das geometrische Modell zählt, die die Eindeutigkeit des zugehörigen Subclusters verlangen. Das bedeutet, daß im laufenden System innerhalb desselben Clusters keine weitere Einheit dieses Typs existieren darf. Die Existenz weiterer Varianten dieses Typs außerhalb des laufenden Systems – insbesondere unter Verwaltung eines PDM-Systems – wird allerdings nicht ausgeschlossen.

Zum anderen gibt es Anwendungen, die im Gegensatz zur Eindeutigkeit ausdrücklich die Existenz paralleler Subcluster fordern. Mit Analysen und Zeichnungsdefinitionen fallen hierunter hauptsächlich Applikationen, bei denen mehrere typverwandte Aufgaben unter verschiedenen Gesichtspunkten ausgeführt werden können. Jede Problemstellung kann jedoch als eigenständige Einheit betrachtet werden und ist daher einem separaten Subcluster zuzuordnen, so daß daraus mehrere parallele Subcluster desselben Typs resultieren.

Da das Wissen über die Zuordnung einer Applikation zur einen oder anderen Kategorie nicht zentral im System verankert werden kann, ist eine generische Methode zur Verfügung zu stellen, mit deren Hilfe jeder Subcluster während des Ladeprozesses selbst entscheidet, ob seine Existenz im gegebenen Kontext den Regeln entspricht. Diese Revision muß nicht auf die beschriebenen Aspekte der Eindeutigkeit beschränkt sein, sondern kann ebenso durch weitere Konsistenzprüfungen ergänzt werden. Zum Beispiel ermöglicht diese Methode sicherzustellen, daß graphisches und geometrisches Modell harmonieren, sofern beide gleichzeitig im laufenden System verfügbar sein sollen.

Kompatibilität

Zur Wahrung der Kompatibilität und um einen gleitenden Übergang auf das Konzept zu erreichen, kann die Segmentierung eines Clusters in Subcluster schrittweise, das heißt, Applikation für Applikation, erfolgen. Aufgrund der Forderung nach einer vollständigen Unterteilung des Clusters ist dabei zu gewährleisten, daß die verbleibenden, noch nicht in Subcluster separierten Entities dem Struktur-Subcluster auch weiterhin zugeordnet bleiben. Insbesondere dürfen deren Pfade vom und zum Cluster Master nicht über die Gebiete der neuen Subcluster verlaufen.

Daten im laufenden System sollen ausschließlich nach den Regeln des Subcluster-Konzepts verwaltet werden. Ohne jegliche Separierung kann aufgrund deren Äquivalenz zumindest jeder Cluster als Struktur-Subcluster aufgefaßt werden. Applikationen, die bereits auf das Subcluster-Konzept aufbauen, müssen folglich Altdaten nach dem Laden in die neuen Datenstrukturen konvertieren. Dazu sind insbesondere die Subcluster Master zu erstellen und einzubinden. Beim rückwärtskompatiblen Schreiben der Daten muß die Subcluster-Struktur entsprechend aufgelöst und der bisherige Zustand mit direkter Zuordnung der Entities zum Cluster wiederhergestellt werden. Die eigentliche Struktur der jeweiligen Daten bleibt hingegen weitestgehend unverändert.

Besitzt eine Einheit bereits einen im Cluster inaktiven Cluster Master, weil dieser bislang in integrativer Beziehung zum Cluster stand – dies ist beispielsweise beim geometrischen Modell der Fall –, so kann dieser Repräsentant die Funktion des Subcluster Masters unmittelbar übernehmen. Auf ein zusätzliches dediziertes Entity für

den Subcluster Master kann dann verzichtet werden. Dadurch wird ebenfalls eine Konvertierung von und in Altdaten erleichtert oder entfällt günstigstenfalls komplett.

Voraussetzungen

Eines der Hauptziele, die mit der Entwicklung des Subcluster-Konzepts verfolgt werden, ist die Separierung der Daten in einer Weise, die das Laden anwendungsspezifischer Kombinationen von Dateneinheiten erlaubt. Dies bedeutet implizit, daß eine Vielzahl von Dateneinheiten, die zum Bearbeiten einer bestimmten Aufgabe nicht notwendig sind, auch nicht geladen werden muß und infolgedessen ebensowenig als Ziel von Referenzen zur Verfügung steht.

Andererseits soll das dynamische Ergänzen des Modells durch Nachladen zusätzlich benötigter Subcluster ausdrücklich ermöglicht werden. Dies erfordert jedoch die Fähigkeit, Referenzen mit einem Ziel, das temporär nicht erreichbar ist, automatisch wiederzuverbinden, sobald dieses infolge einer Ladeoperation wieder zur Verfügung steht. Eine wesentliche Grundvoraussetzung für die Separierung der Daten ist daher die Existenz einer Technologie, die es dem System erlaubt, Referenzen auch dann zu erhalten, wenn deren Ziele temporär nicht erreichbar sind, mit solchen offenen Referenzen während der Laufzeit umzugehen und diese automatisch wiederzuverbinden, sobald dies möglich ist. In OneSpace Designer Modeling wird eine derartige Technologie durch das in Kapitel 2.5.2 beschriebene Open Reference Handling bereitgestellt.

Ein weiteres Kriterium für die Separierung von Daten einer Applikation ist deren Abgeschlossenheit und Vollständigkeit. Die Einheit der Entities, die in einen Subcluster ausgegliedert werden soll, liegt daher idealerweise bereits in einer clusterähnlichen Struktur vor. Das bedeutet, ein Einstiegspunkt ist vorhanden, von dem aus alle Entities der Applikation erreicht werden können und der in umgekehrter Richtung von den Ketten der Local Master all dieser Entities durchlaufen wird. Diese Vorbedingung muß jedoch nicht zwingend erfüllt sein. Die erforderliche Struktur kann auch noch während der Umsetzung des Konzepts durch entsprechende Integration des neuen Subcluster Masters in das vorhandene Entity-Netzwerk erreicht werden. Diese Erweiterung erfolgt dann allerdings auf Kosten eines höheren Aufwands für die Kompatibilität.

Datenstrukturen

Im folgenden soll eine Übersicht über die Architektur gegeben werden, die der Separierung der Daten unter Verwendung des Subcluster-Konzepts zugrunde liegt. Dazu ist in **Abbildung 5.6** der prinzipielle Aufbau einer Klassenstruktur dargestellt, die zur Unterteilung einer Komponente der Baugruppenstruktur Anwendung findet. Darin verwendete Klassen bilden die Basis für diejenigen Entities, denen beim Aufbau der in **Abbildung 5.5** beschriebenen Beziehungen zwischen den Subclustern eine Schlüsselrolle zukommt.

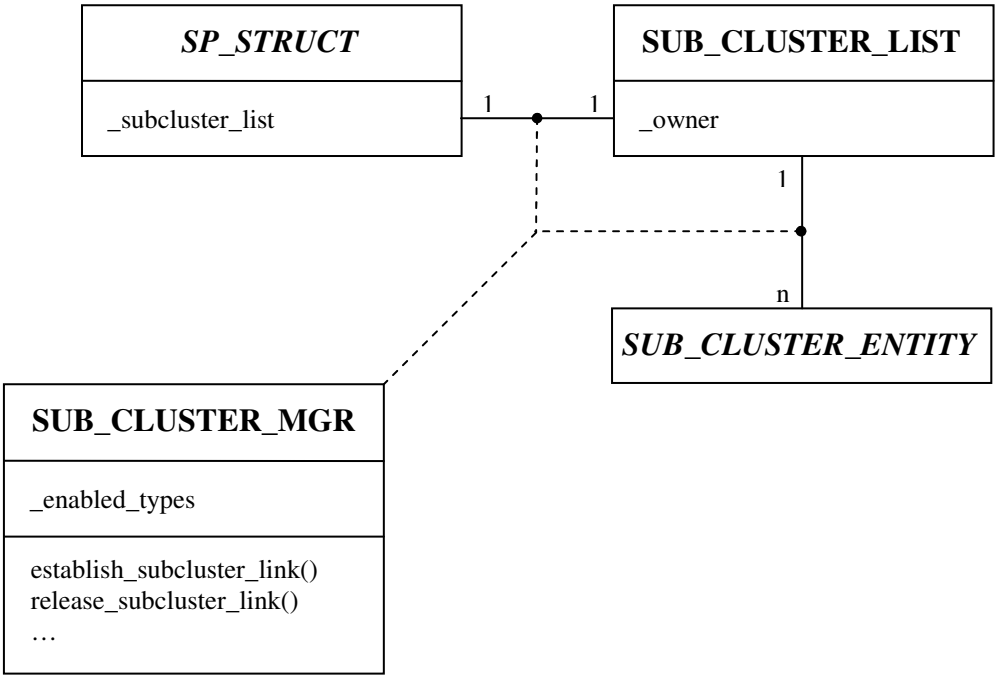


Abbildung 5.6: Prinzipieller Aufbau der Klassenstruktur zur Separierung

Die Klasse *SP_STRUCT* stellt den Master des Struktur-Subclusters – und damit gleichzeitig den Repräsentant des gesamten Clusters. Als generische Basis für alle Einheiten der Modelldaten[17] ist diese Klasse abstrakt.

Die Aufgabe des Struktur-Subclusters, alle weiteren Einheiten zu verwalten, wird an ein Element der Klasse *SUB_CLUSTER_LIST* delegiert. Dieses Entity geht mit dem Cluster Master, der gleichzeitig dessen Lebenszeit kontrolliert, eine 1:1-Beziehung ein und dient jedem Repräsentant der ihm zugeordneten Subcluster als Local Master.

[17] Vgl. Kapitel 2.3 Strukturmodell

De facto sind dies alle Subcluster innerhalb des Clusters. Dadurch entsteht eine 1:n-Beziehung zwischen dem Element der Klasse SUB_CLUSTER_LIST und den Subclustern, repräsentiert durch Entities der Klasse SUB_CLUSTER_ENTITY.

Ein Element der Klasse *SUB_CLUSTER_ENTITY* fungiert als Master des jeweiligen Subclusters und hält über die SUB_CLUSTER_LIST die Verbindung zum Cluster Master. Aufgrund fehlenden Applikationswissens ist die Klasse abstrakt. Für jede zu separierende Anwendung wird eine Ableitung erforderlich, die die Anbindung und Erreichbarkeit aller Entities dieser Applikation unter den lokalen Randbedingungen sicherstellt.

Die Klasse *SUB_CLUSTER_MGR* ist eine Manager-Klasse, die den Aufbau der Subcluster-Struktur unterstützt sowie ihre Konsistenz sicherstellt und überwacht. Weitere Details dieser Struktur werden – insbesondere unter Berücksichtigung anwendungsbezogener Gegebenheiten – in den folgenden Abschnitten sowie im Rahmen der Darstellung der softwaretechnischen Umsetzung des Subcluster-Konzepts in Kapitel 6.2 beschrieben.

5.2.2 Strukturdaten

Der strukturelle Aufbau von Baugruppen, wie in Kapitel 2.3 näher beschrieben, wird durch hierarchische Clusterbeziehungen definiert. Demnach wird die gesamte Baugruppe vollständig in clusterbasierte Instanz- und Modelleinheiten unterteilt. Jeder dieser Cluster enthält eine Anzahl von Entities, deren Beziehungen untereinander und clusterübergreifend die strukturelle Erscheinung der Baugruppe festlegen. Neben der reinen Hierarchiedefinition, welche die Eltern-Kind-Beziehung zwischen der Baugruppe und darin verwendeten Teilen und Unterbaugruppen bzw. zwischen Instanzen und zugehörigen Modelldaten ausdrückt, enthält dieser Kern eines Clusters weitere Informationen, die zur Beschreibung des Modellgerüsts notwendig sind. Bei Instanzen sind dies beispielsweise deren Name sowie deren Position relativ zum jeweiligen Besitzer, das heißt, zur direkt übergeordneten Baugruppe.

Dieses Rückgrat der Baugruppenstruktur ist heute sowohl Träger als auch Besitzer jeglicher Zusatzinformation. Neben den Kerndaten zur Beschreibung der Struktur

werden das geometrische Modell und Applikationsdaten für jedes Element der Baugruppe in einem gemeinsamen Cluster verwaltet und daher ebenso in eine gemeinsame Datei geschrieben. Die **Abbildung 5.7** zeigt diesen Zustand. Außer der Integration der Instanzdaten in den Cluster der Modelldaten[18] wird darin ebenfalls deutlich, daß die Beziehungen der Dateien untereinander denen der Clusterstruktur entsprechen.

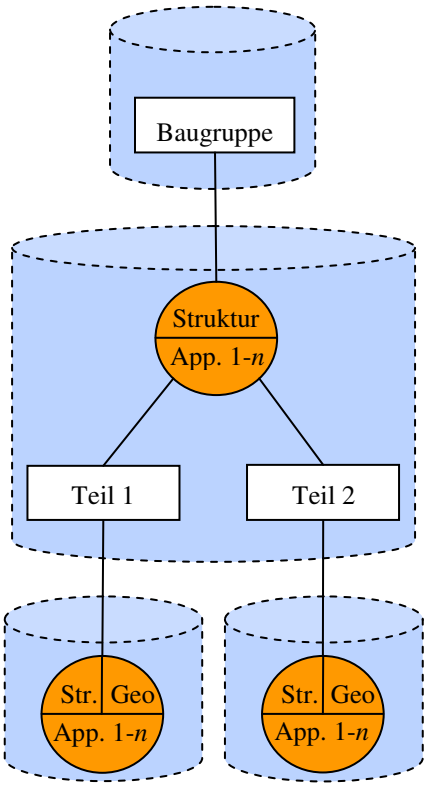


Abbildung 5.7: Heutige Cluster- und Dateistruktur einer Baugruppe

Im Rahmen der Datenseparierung erfolgt die Aufteilung jedes eigenständigen Clusters in einen Subcluster zur Strukturdefinition und weitere Subcluster zur Verwaltung zusätzlicher Daten. Der Struktur-Subcluster beinhaltet dabei alle Kerndaten, die zur Beschreibung eines Elementes des Baugruppengerüsts notwendig sind, und übernimmt damit die organisatorische Funktion des Clusters in bezug auf die Baugruppenstruktur. Diese ist folglich allein durch Beziehungen zwischen Struktur-Subclustern definiert. Dadurch wird die Rückgrat-Funktion der Baugruppenstruktur betont und gleichzeitig deren eigenständige Verwaltung – ohne direkte Integration von Zusatzdaten – ermöglicht.

[18] Vgl. Kapitel 2.3.4 Persistenz von Strukturelementen

Auf die Separierung nicht strukturrelevanter Daten, wie dem geometrischen bzw. graphischen Modell sowie Applikationsdaten, wird in den folgenden Abschnitten genauer eingegangen. Subcluster, die aus dieser Separierung hervorgehen, werden zum Struktur-Subcluster in direkte Relation gesetzt. Die sich daraus ergebende Unterteilung der eingangs betrachteten Baugruppe in Subcluster sowie die Beziehungen der Subcluster und Dateien untereinander sind in **Abbildung 5.8** dargestellt. Die Struktur-Subcluster sind darin farblich hinterlegt. Dadurch wird deren gerüstdefinierende Eigenschaft verdeutlicht.

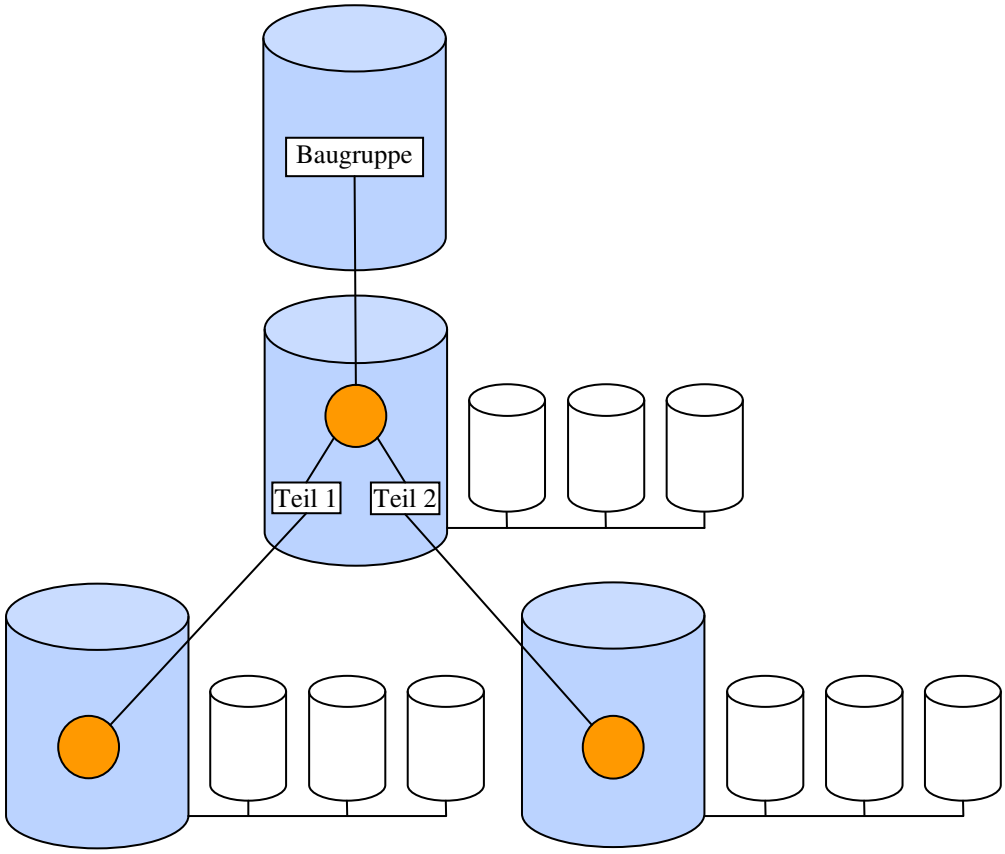


Abbildung 5.8: Baugruppe mit separierten Strukturdaten

Aufgrund seiner Eigenschaft, die Baugruppenstruktur zu definieren, nimmt der Struktur-Subcluster eine Sonderstellung unter allen Subclustern ein, die sich auch in den Datenstrukturen widerspiegelt. Als Pars pro toto dem Cluster gleichgesetzt, ist nicht nur die Forderung nach Eindeutigkeit dieses Subclusters innerhalb des Clusters offensichtlich. Vielmehr übernimmt der Master des Struktur-Subclusters auch gleichzeitig die Funktion des Masters für den gesamten Cluster. Dieser Äquivalenz

entsprechend ist dieses Entity dem prinzipiellen Aufbau der Klassenstruktur in Abbildung 5.6 gemäß ein Repräsentant der Klasse SP_STRUCT, die bereits im vorhandenen Design Grundlage für den Cluster-Master ist und im Zuge der Datenseparierung um Funktionalität zur Subclusterverwaltung erweitert wird.

Durch entsprechende Eingliederung der weiteren Subcluster mittels integrativer Beziehungen an der Schnittstelle zum Struktur-Subcluster wird darüber hinaus die übergeordnete Clusterstruktur der Baugruppe erhalten. Dadurch bleibt unter anderem die clusterbasierte Orientierung in der Organisation der Modelldaten unverändert. Dies ist eine grundlegende Voraussetzung zur Kompatibilität mit vorhandenen Algorithmen und Datenstrukturen.

In der praktischen Anwendung läßt die Existenz von strukturrelevanten Informationen wie Teilenamen, Positionen, Mehrfachverwendung etc. bereits im Gerüst der Baugruppe umfangreiche Bearbeitungsmöglichkeiten zu. Zum einen sind strukturbasierte Operationen wie das Verschieben, Löschen oder Erzeugen von Teilen und Baugruppen ebenso möglich wie die Ableitung von Stücklisten. Zum anderen steht mit dem Gerüst eine sehr schlanke Struktur der Baugruppe zur Verfügung, auf deren Basis die Umgebung für Konstruktionsaufgaben positionserhaltend in graphischer oder geometrischer Repräsentation sowie darüber hinaus benötigte Applikationsdaten nachgeladen werden können.

5.2.3 Geometrisches Modell

Die Beschreibung des geometrischen Modells wird in OneSpace Designer Modeling mit Hilfe einer Datenstruktur realisiert, die auf der Boundary Representation basiert. Zu den daran beteiligten Klassen gibt **Abbildung 5.9** einen architektonischen Überblick. Das B-Rep-Modell und dessen Implementierung als Entity-Netzwerk werden bereits in Kapitel 2.2.2 ausführlich erläutert. An gleicher Stelle wird auch gezeigt, daß diese Datenstruktur durch eine in sich abgeschlossene Darstellung der Geometrie und Topologie charakterisiert ist, die alle Anforderungen an einen Cluster erfüllt. Der Body fungiert dabei als Master eines Clusters, der seinerseits in den Cluster der Modelldaten integriert ist.

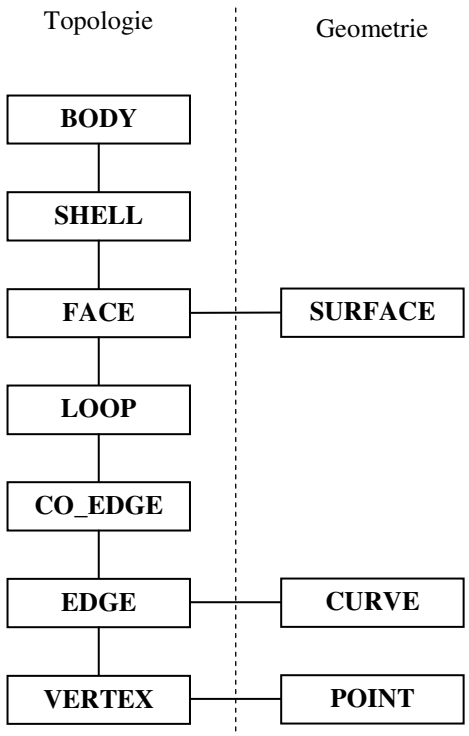


Abbildung 5.9: Klassenstruktur eines B-Rep-Modells

Zur Realisierung des Subcluster-Konzepts ist daher die Verwendung der vorhandenen Begrenzung als Trennlinie für die Separierung naheliegend. Als Repräsentant des geometrischen Modells erfüllt der Body ebenso alle Grundvoraussetzungen für einen Subcluster Master. Um dem Konzept vollständig gerecht zu werden, wird – wie in **Abbildung 5.10** dargestellt – die Klasse BODY, die das Body-Entity stellt, von der Klasse SUB_CLUSTER_ENTITY abgeleitet. Alternativ kann ein dedizierter Repräsentant des geometrischen Modells parallel zum Body installiert werden.

Das geometrische Modell ist ausschließlich Bestandteil der Modelldaten eines Teils, die durch ein Entity der Klasse PART_STRUCT repräsentiert werden. Die Verbindung zwischen diesem Repräsentant und dem Body bleibt weiterhin, nicht zuletzt aus Gründen der Kompatibilität, als redundante Beziehung erhalten. Zusätzlich dazu wird die eigentliche Anbindung des geometrischen Modells an den Struktur-Subcluster nach den Regeln des Subcluster-Konzepts aufgebaut. Aufgrund der geforderten Eindeutigkeit ist die 1:1-Beziehung zwischen den Klassen BODY und SUB_CLUSTER_LIST zu beachten.

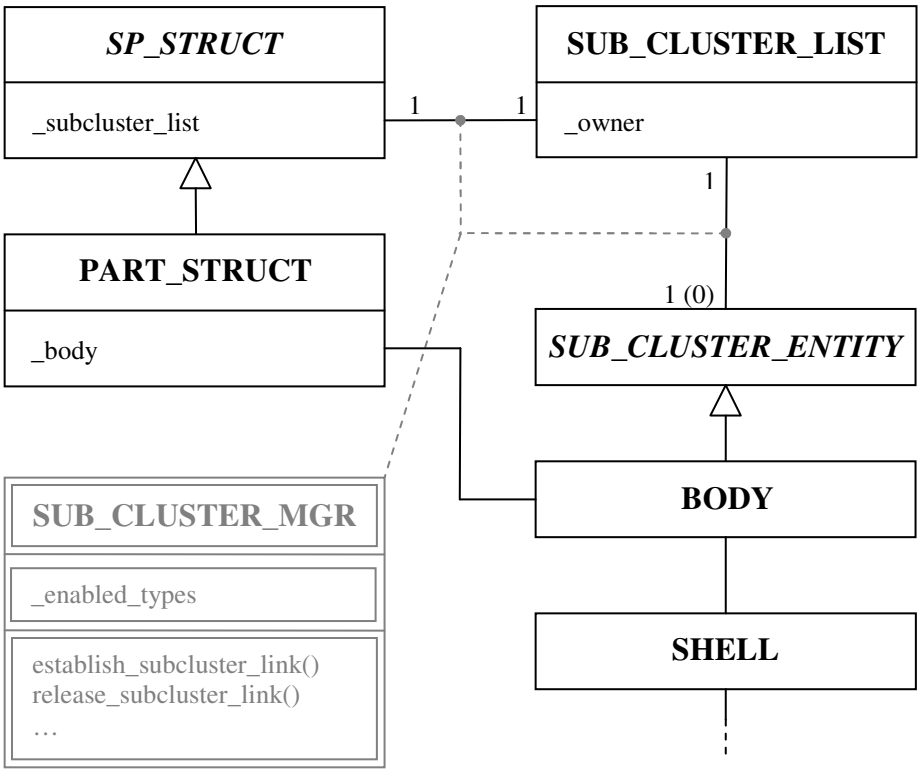


Abbildung 5.10: Klassenstruktur zur Separierung des geometrischen Modells

Durch die Separierung des geometrischen Modells wird es möglich, verschiedene Varianten desselben Teils unter einem Strukturknoten wechselweise darzustellen. Im Gegensatz zur Variation der formdefinierenden Parameter des Teils, die für jeden Parametersatz eine Variante auf Teileebene liefert, können durch den Austausch von Subcluster-Varianten verschiedene Darstellungen eines bestimmten Teils erreicht werden. Dies kann beispielsweise ausgenutzt werden, um Vereinfachungen oder unterschiedliche Einbauzustände dieses Teils – wie etwa entspannte oder belastete Federn – zu zeigen.

Dies veranschaulicht nochmals die notwendige Eindeutigkeit des Geometrie-Subclusters. Damit ist einem Teil zu einem Zeitpunkt entweder kein Subcluster zugeordnet – wenn keiner geladen wurde – oder genau ein Subcluster, der die gewählte modellhafte Abbildung des Teils beschreibt. Im zweiten Fall ist durch die Präsenz aller geometrischen und topologischen Informationen die volle Modellierfähigkeit gegeben. Ebenso sind genaues Messen, die Auswahl basierend auf geometrischen Elementen sowie die volle Referenzierbarkeit des Modells möglich.

5.2.4 Graphisches Modell

Die Daten eines graphischen Modells können bei einigen CAD-Systemen hinsichtlich ihrer Persistenz unterschieden werden. Während die darstellungsdefinierende Komponente zusammen mit dem 3D-Modell abgespeichert wird, erfolgt die Generierung des Facettenmodells dynamisch nach jedem Laden sowie nach Modellveränderungen durch Ableitung vom geometrischen Modell – wozu dessen Verfügbarkeit vorausgesetzt werden muß. In OneSpace Designer Modeling bildet das graphische Modell eine Einheit miteinander verbundener Entities. Die Struktur der dem Modell zugrunde liegenden Klassen wird in **Abbildung 5.11** schematisch gezeigt.

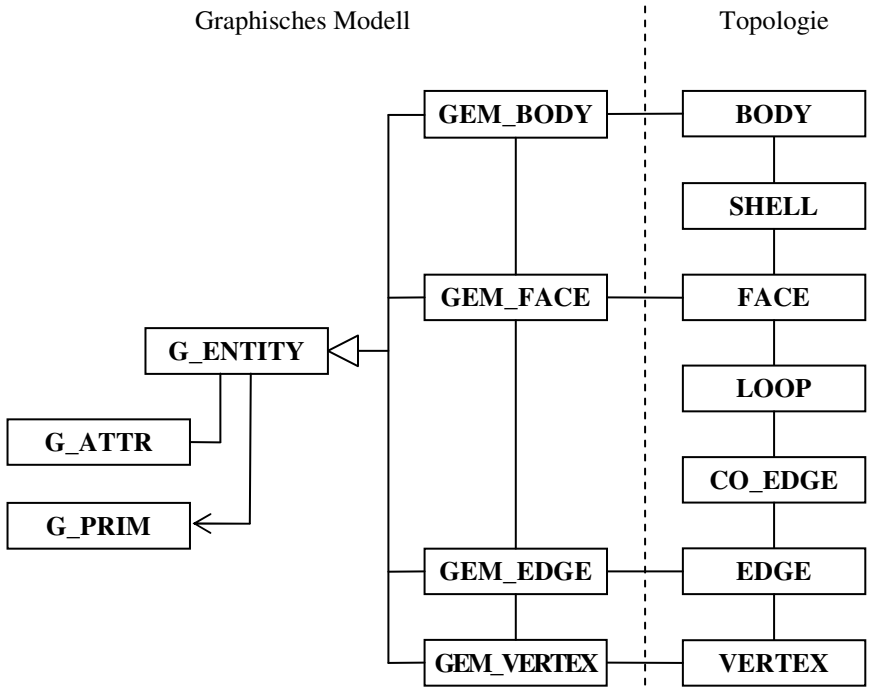


Abbildung 5.11: Klassenstruktur des graphischen Modells

Das Entity-Netzwerk der darstellungsdefinierenden Struktur beruht auf einer quasi dualen Topologie, die unter anderem Attribute verwaltet, welche das Aussehen der einzelnen graphischen Elemente beschreiben. Diese schließen Eigenschaften wie Farbe, Transparenz und Reflexionsparameter ein, legen aber auch die Genauigkeit der Elemente, das heißt, deren maximalen Abstand zur Bezugsgeometrie, fest. Die Verwaltung der Attribute wird von der gemeinsamen Basisklasse der graphischen Topologie-Entities unterstützt. Die beteiligten Entities definieren ihre Local Master derart, daß sie direkt oder indirekt über die referenzierten Topologieelemente in den Cluster des 3D-Modells integriert werden und dadurch ihre Persistenz erreichen.

Das Facettenmodell wird mit Hilfe von graphischen Primitiven beschrieben. Diese dynamisch erzeugten Elemente sind jeweils einem Entity der graphischen Topologie eindeutig zugeordnet und stehen unter dessen Kontrolle. Die Referenzen zwischen den topologischen Entities und den Elementen des Facettenmodells sind lediglich einseitig, die Speicherung der graphischen Primitive ist bislang nicht vorgesehen.

Die Trennung von Darstellungsdefinition und Facettenmodell soll auch durch die Separierung der Daten reflektiert werden. Demgemäß wird das graphische Modell in zwei Subcluster unterteilt. Insbesondere die graphischen Attribute werden sowohl für die Ableitung der graphischen Repräsentation vom geometrischen Modell benötigt als auch zum Anzeigen des Facettenmodells, wenn die Geometriedaten nicht zur Verfügung stehen. Im Fall der Ableitung muß es möglich sein, eine äquivalente graphische Darstellung dynamisch zu erzeugen, falls keine solche zusammen mit den 3D-Daten geladen wurde. Überdies wird dadurch eine einheitliche Definition der Attribute erreicht, die auch im rein graphischen Modell geändert werden kann.

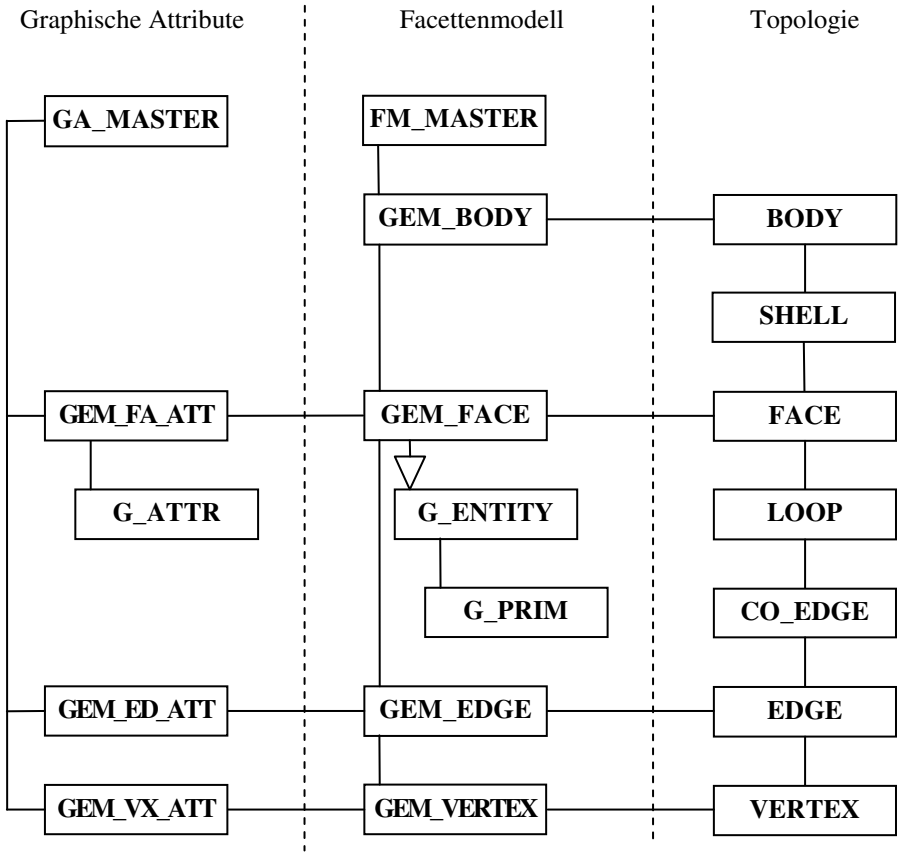


Abbildung 5.12: Klassenstruktur des separierten graphischen Modells

Um das Ziel der Separierung der Daten in zwei Subcluster zu erreichen, muß in beide Einheiten jeweils ein Subcluster Master eingebunden werden. **Abbildung 5.12** zeigt die resultierende Klassenstruktur. Neben den neuen Mastern wird darin auch eine veränderte Verwaltung der Attribute deutlich. Diese werden jetzt nicht mehr durch die Basisklasse der graphischen Topologie direkt referenziert, sondern über eigens für diesen Zweck eingeführte Klassen, deren Elemente in 1:1-Beziehung zum topologischen Element stehen, in einen separaten Subcluster eingebunden. Die exemplarisch dargestellte Beziehung zwischen den Klassen GEM_FA_ATT und G_ATTR gilt analog für Edge- und Vertex-Attribute.

Das Facettenmodell wird hauptsächlich durch die graphische Topologie und ihr zugeordnete Primitive beschrieben. Die grundsätzliche Verbindung zwischen einem Topologieelement und seinen Primitiven – wie zwischen GEM_FACE und G_PRIM dargestellt und analog für die weiteren Elemente gültig – bleibt erhalten. Jedoch dokumentiert jetzt jedes Primitiv seine Zugehörigkeit zum Subcluster durch einen Local Master, der auf das übergeordnete graphische Topologieelement verweist. Dieses wiederum verweist mit seinem Local Master – gegebenenfalls indirekt – auf den Subcluster Master statt wie bisher auf das assoziierte Element der 3D-Topologie.

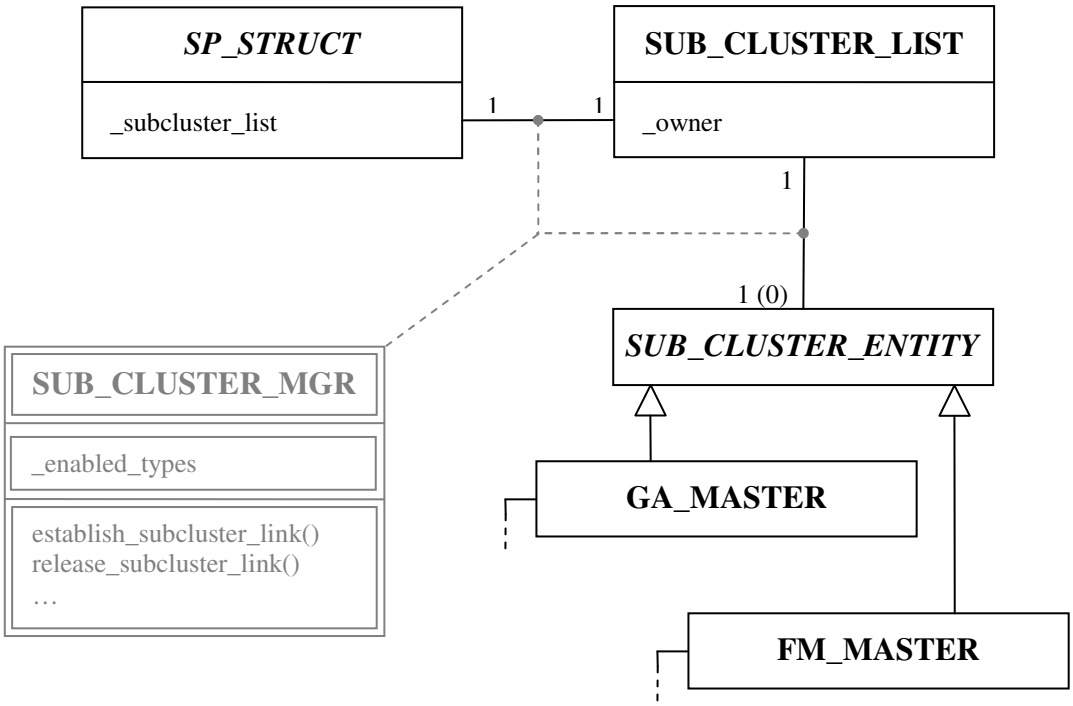


Abbildung 5.13: Klassenstruktur zur Separierung des graphischen Modells

Die Integration der beiden graphischen Subcluster nach den Regeln des Subcluster-Konzepts wird in **Abbildung 5.13** gezeigt. Durch die beschriebene Datenstruktur wird schließlich neben der Persistenz der graphischen Attribute auch die des Facettenmodells erreicht. Zusammen mit Strukturmodell ist damit die Darstellung sogenannter Lightweight-Modelle realisierbar. Diese entsprechen einer rein graphischen Abbildung einer Baugruppe, ohne Verwendung geometrischer Daten, jedoch unter Erhaltung aller Strukturinformationen.

Wie beim geometrischen Modell können mehrere Varianten des graphischen Modells eines Teils – gegebenenfalls mehrere ***L**evels **o**f **D**etail* (*LOD*) – von einem PDM-System parallel verwaltet werden. Zur Laufzeit wird jedoch lediglich eine eindeutige Repräsentation dargestellt, die darüber hinaus mit dem geometrischen Modell korrespondieren muß, sofern dieses ebenfalls geladen ist.

Bereits die Verwendung von Lightweight-Modellen eröffnet eine Reihe Bearbeitungsmöglichkeiten in einem 3D-CAD-System. Von der Präsentation des optischen Eindrucks eines Modells in unterschiedlichen Darstellungen über Methoden zur dynamischen Betrachtung bis hin zur graphikbasierten Selektion, Messung, Zeichnungsableitung oder Kollisionsanalyse können viele Schritte ausgeführt werden. All diesen Funktionalitäten gemein ist eine geringere Genauigkeit, die aber für die meisten Anwendungen völlig ausreichend ist. Durch die vorhandene Strukturinformation können Teile des Modells außerdem jederzeit durch erforderliche geometrische Daten ergänzt werden.

5.2.5 Applikationsdaten

Applikationsdaten lassen sich in OneSpace Designer Modeling nach den zur Verwaltung verwendeten Datenstrukturen vornehmlich in zwei Klassen einteilen. Einerseits solche, die auf einer expliziten Feature-Darstellung beruhen, andererseits solche, die eine clusterintern weitergeführte Struktur verwenden, die der Baugruppenstruktur ähnlich ist und auf deren Basisklassen aufbaut.

Clusterintern weitergeführte Strukturen

Das Prinzip der weitergeführten Struktur findet zum Beispiel bei Koordinatensystemen, Dokumentationsebenen und Zeichnungsdefinitionen Anwendung. Durch die gemeinsame Grundlage mit der Baugruppenstruktur erben diese Elemente Funktionalität zur weiteren Unterteilung und Strukturierung unter Verwendung von Eltern-Kind-Beziehungen. Außerdem besitzen sie die Eigenschaft, als Pfad- oder Zielelemente direkt von Features referenziert werden zu können. Der Einstiegspunkt einer solchen Struktur ist eine Instanz, die ihre Stellung als Cluster Master zugunsten einer integrativen Beziehung zum besitzenden Cluster aufgegeben hat. **Abbildung 5.14** zeigt die Struktur zur Definition einer Zeichnung innerhalb der Modelldaten eines Teils mit der Instanz „Zeichnung“ als Einstiegspunkt.

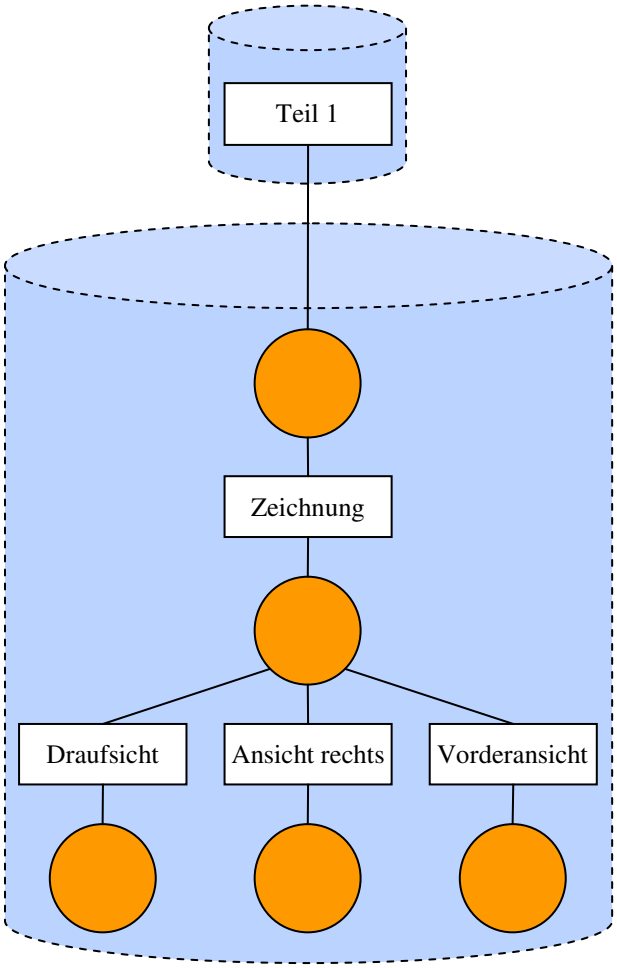


Abbildung 5.14: Clusterintern weitergeführte Struktur zur Zeichnungsdefinition

Aufgrund der bereits vorhandenen Cluster-Struktur kann die Anwendung des Subcluster-Konzepts durch minimale Anpassung der Datenstruktur erreicht werden. Der Einstiegspunkt in die zu separierende Einheit, das heißt, der Master des höchsten Instanz-Clusters, übernimmt die Aufgabe des Subcluster Masters. Alle Cluster unterhalb dieser Instanz bleiben darin integriert.

Exemplarisch für die Ausgliederung einer clusterintern weitergeführten Struktur wird in **Abbildung 5.15** die Klassenstruktur zur Separierung einer Zeichnungsdefinition dargestellt. Für weitere Applikationen, deren Daten auf der gleichen Basis verwaltet werden, kann analog verfahren werden. Ähnlich dem geometrischen Modell, dessen Repräsentant ebenfalls clusterbasiert ist, wird die Klasse LAYOUT von der Klasse SUB_CLUSTER_ENTITY abgeleitet. Diese Klasse stellt den Einstiegspunkt in das Entity-Netzwerk zur Definition einer Zeichnung.

Auch hier wird die Verbindung zwischen der – jetzt separierten – Instanz und den übergeordneten Modelldaten, die die Fortsetzung der Struktur bewirkt, als redundante Beziehung erhalten. Die eigentliche Anbindung der Zeichnungsdefinition an den Struktur-Subcluster wird wiederum nach den Regeln des Subcluster-Konzepts aufgebaut, wobei der geforderten Parallelität von Zeichnungsdefinitionen in diesem Fall durch eine 1:n-Beziehung zwischen SUB_CLUSTER_LIST und LAYOUT Rechnung getragen wird.

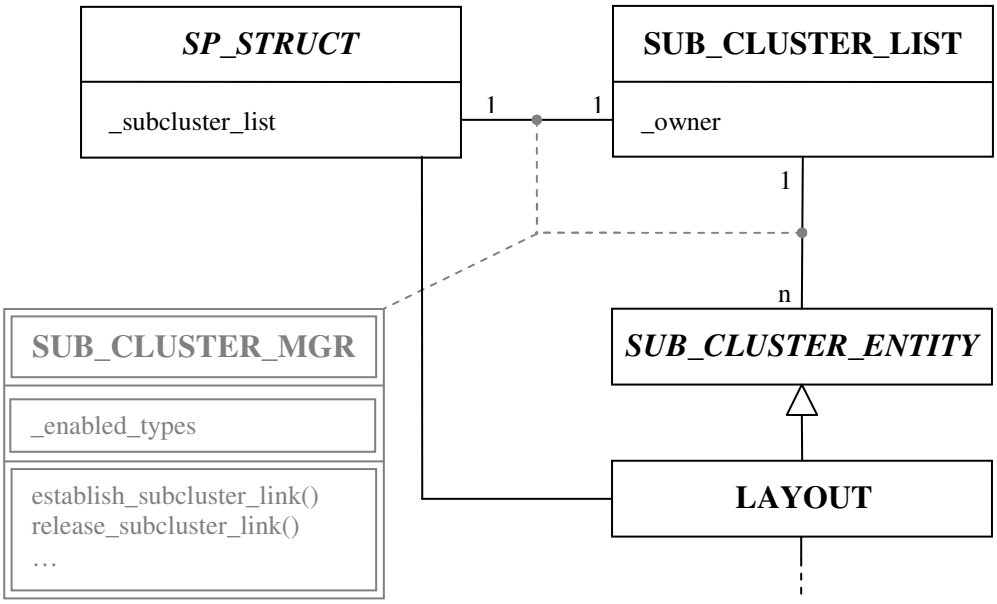


Abbildung 5.15: Klassenstruktur zur Separierung einer Zeichnungsdefinition

Explizite Feature-Darstellung

Auf der Grundlage der expliziten Feature-Darstellung ist die Verwaltung einer Vielzahl von Applikationen realisiert. Unter anderem verwenden die integrierten Module zur 3D-Bemaßung, Finit-Element-Analyse und Parametrik diese Basis, um eigene Funktionalität darauf aufzubauen. Alle Entities, die zur Definition eines derartigen Features beitragen, bilden eine in sich abgeschlossene Einheit, die über die Local Master Beziehung des Feature-Entities zu seinem Besitzer an diesen gebunden wird. Kapitel 2.4.2 beschreibt diese Datenstruktur im Detail, vergleiche dazu auch Abbildung 2.15.

Zur Realisierung des Subcluster-Konzepts für Features muß ein neuer Subcluster Master eingeführt werden. Dieser wird parallel zum Feature-Entity eingebunden. Die Beziehung zwischen dem Feature und seinem Besitzer bleibt erhalten, die Funktion des Local Masters übernimmt jetzt jedoch das neue Entity. Die sich daraus ergebende Klassenstruktur zur Separierung eines Features wird in **Abbildung 5.16** dargstellt.

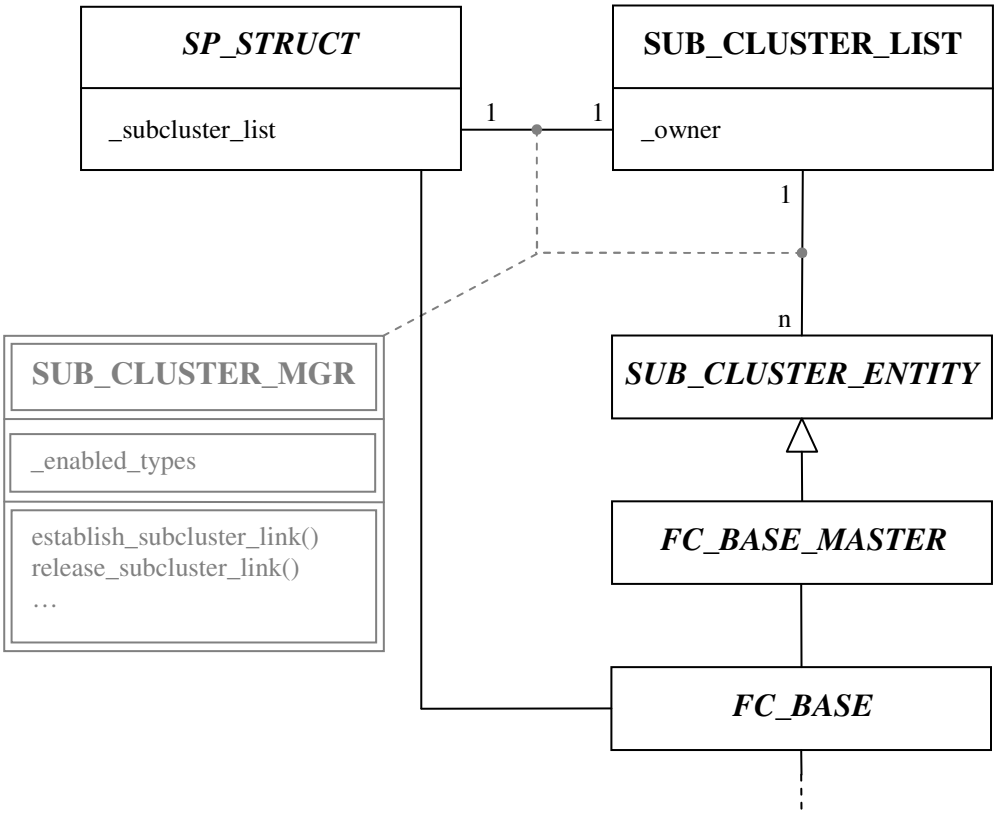


Abbildung 5.16: Klassenstruktur zur Separierung eines Features

Sowohl das Feature selbst als auch dessen Subcluster Master werden in der Abbildung durch ihre abstrakten Basisklassen dargestellt. Zur besseren Differenzierung der Repräsentanten verschiedener Anwendungsbereiche wird auch deren applikationsspezifische Ableitung verlangt. Durch Integration des Designs auf der Ebene der Basisklassen steht das Subcluster-Konzept mittels Vererbung automatisch allen featurebasierten Applikationen zur Verfügung.

Die Verbindung zwischen dem Feature und seinem Besitzer wird durch eine Beziehung zum Master der Strukturdaten symbolisiert, kann jedoch ebenso als Referenz zu einer Instanz innerhalb der Strukturdaten realisiert sein. Während jedes Feature eindeutig einem Master zugeordnet ist, kann dieser durchaus verschiedene Features referenzieren. Dadurch lassen sich mehrere Einheiten zu einem Subcluster gruppieren. Zumindest eine semantische Beziehung der Features zueinander soll allerdings gegeben sein, in der Regel werden diese zur selben Anwendung gehören.

Die 1:n-Beziehung zwischen der Klasse SUB_CLUSTER_LIST und den Klassen der einzelnen Subcluster Master ermöglicht die erforderliche Parallelität der Dateneinheiten. Dadurch können sowohl Daten aus verschiedenen Anwendungen als auch mehrere Einheiten derselben Applikation nebeneinander im laufenden System verwaltet werden.

Durch Einführung des Subcluster-Konzepts auf der Ebene der Applikationsdaten können Daten einzelner Anwendungen aufgrund der erlangten Granularität und Eigenständigkeit unabhängig voneinander geladen bzw. nachgeladen werden. Neben dem Schutz geistigen Eigentums durch gezieltes Ausklammern einzelner Subcluster und der darin enthaltenen Informationen ermöglicht dies Concurrent Engineering auf der Basis der Applikationsdaten.

5.2.6 Assoziativität nicht geladener Applikationsdaten

Die enge Bindung von 3D-Modell und technologischen Produktdaten ist motiviert durch die damit erreichbare Assoziativität, das heißt, die Möglichkeit einer automatischen Anpassung infolge von Modelländerungen. Dementsprechend stellt sich die

Aufgabe, auf Veränderungen adäquat zu reagieren, wenn Modellkonfigurationen bearbeitet werden, die nicht alle abhängigen Daten umfassen.

Diese Konstellation kann bereits heute entstehen. Exemplarisch soll das in **Abbildung 5.17** gezeigte Modell betrachtet werden, in dem Applikationsdaten an einer Oberbaugruppe verwaltet werden und Referenzen auf ein Teil haben, das zu einer Unterbaugruppe gehört. Grundsätzlich ist es möglich, lediglich die Unterbaugruppe mit dem Teil zu laden – folglich ohne Oberbaugruppe und somit ohne die referenzierenden Daten – und das Teil zu verändern oder zu entfernen. Wird später die gesamte Baugruppe wieder geladen, begegnen die Referenzen einer neuen Situation, an deren Entstehen sie nicht beteiligt waren.

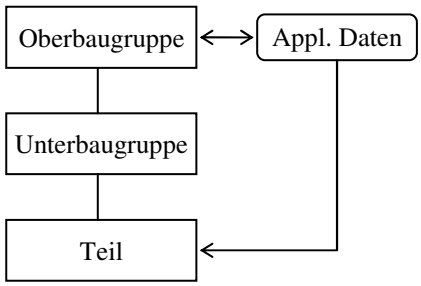


Abbildung 5.17: Vereinfachte Darstellung einer Baugruppe mit Applikationsdaten

Intelligente Validierungsalgorithmen versuchen daher posthum, auf das Ergebnis geeignet zu reagieren. Deren Funktionalität ist jedoch zwangsläufig eingeschränkt. Das Aufsplitten einer Face kann in einem B-Rep-Modell beispielsweise nachträglich nicht erkannt werden. Dementsprechend ist es einer Applikation nicht mehr möglich, zu einer Referenz auf die Ursprungsfläche automatisch eine weitere auf die neu entstandene Fläche zu erzeugen – was sie unter Umständen veranlaßt hätte, wäre sie an der Modelländerung direkt beteiligt gewesen.

Unterstützt durch das Subcluster-Konzept können noch leichter Modellkonfigurationen geladen werden, die nicht alle Applikationsdaten umfassen. Tatsächlich ist es ausdrückliches Ziel des Konzepts, die Datenmenge zu reduzieren. Dadurch kann häufiger die Situation entstehen, daß Daten im System nicht zur Verfügung stehen, die Referenzen auf sich verändernde Objekte haben. Andererseits kann das Subcluster-Konzept aber ebenso dazu ausgenutzt werden, die benötigten Daten gezielt nachzuladen, um alle Referenzen an der Operation zu beteiligen und ihnen damit die

Gelegenheit zu geben, entsprechend direkt zu reagieren. Dazu sollen hier zwei Lösungsansätze skizziert werden, deren Entwicklung im Detail den Rahmen dieser Arbeit jedoch übersteigen würde.

Der erste Ansatz baut auf intelligentes Link-Management des PDM-Systems auf. Dazu muß das PDM-System über Informationen verfügen, welche Subcluster auf welche anderen verweisen. Einen Beitrag dazu, dieses Wissen aufzubauen, kann unter anderem eine Öffnung der Dateiformate leisten (vgl. Kapitel 5.6.1), die die Möglichkeit zur Auswertung der Dateiinhalte eröffnet. Soll ein Subcluster verändert werden, so können zunächst mit Hilfe des PDM-Systems alle weiteren Subcluster mit davon potentiell betroffenen Referenzen bestimmt und nachgeladen werden.

Ein gänzlich anderer Ansatz wird mit der Verwendung von *Konnektor-Entities* verfolgt. Das sind spezielle Entities, die mit Referenzen assoziiert werden und deren Reaktionsverhalten auf Veränderungen ihrer Ziele emulieren. Die Referenz verweist gleichsam statt direkt auf das Ziel nun auf einen Konnektor, der wiederum mit dem Ziel verbunden ist (vgl. **Abbildung 5.18**). Von diesem Konnektor wird verlangt, daß er bei jeder Veränderung des eigentlichen Ziels zur Verfügung steht und somit adäquat reagieren kann – in der gleichen Weise wie das eigentliche Referenzobjekt reagiert hätte.

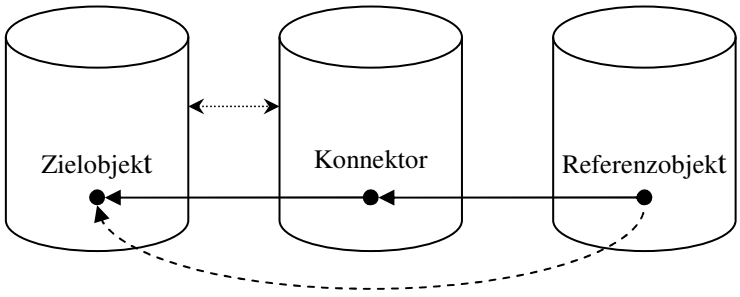


Abbildung 5.18: Referenz unter Verwendung eines Konnektors

Dies impliziert einige Eigenschaften, die beim Design solcher Konnektor-Entities zu berücksichtigen sind. Da die Applikationen teilweise unterschiedlich auf Ereignisse reagieren, müssen verschiedene Klassen von Konnektoren bereitgestellt werden, die die jeweiligen Verhaltensmuster abbilden, so daß Referenzen mit einem Konnektor assoziiert werden, der exakt ihrem Verhalten entspricht.

Über die jeweilige Reaktion hinaus muß ein Konnektor eine Folge von Änderungen protokollieren können. Ebenso muß jede Referenz in der Lage sein, aufgezeichnete Reaktionen des assoziierten Konnektors nachzuvollziehen – das heißt, auf eigene Daten abzubilden – sobald diese wieder im laufenden System verfügbar wird. Damit wird quasi eine Trennung von Referenzen und Daten erreicht. Schließlich müssen abgearbeitet Protokolle gelöscht werden können, um Mehrfachreaktionen auf dasselbe Ereignis zu verhindern.

Mit dem Ziel, den Datenaufwand zu begrenzen, ist die Mehrfachverwendung von Konnektoren derselben Klasse durch verschiedene Referenzen dieses Reaktionstyps anzustreben. Daran können ohne Einschränkung Referenzen verschiedener Applikationen beteiligt sein. Dies bedingt jedoch ebenfalls eine dedizierte Verwaltung der Ereignisprotokolle, da die referenzierenden Daten die jeweiligen Reaktionen zu unterschiedlichen Zeitpunkten nachvollziehen können und daher gegebenenfalls abweichende Aktualität aufweisen.

Alle Konnektoren mit Ziel in einem gemeinsamen Subcluster – das wird hauptsächlich derjenige des geometrischen Modells sein – werden in einem separaten Subcluster verwaltet, der dem Ziel-Subcluster zugeordnet ist. Mit Ausnahme des Schutzes vor paralleler Bearbeitung bestehen für diesen Konnektoren-Subcluster keinerlei Einschränkungen hinsichtlich der Zugriffsberechtigung. Aufgrund dessen wird die ständige Verfügbarkeit der Konnektoren gewährleistet. Überdies ist es dadurch sowohl möglich, Konnektoren auf schreibgeschützte Objekte anzulegen und zu bearbeiten, als auch Applikationsdaten aktuell zu halten, ohne diese laden zu müssen.

5.3 Optimierung der Konstruktionsumgebung

Die Konstruktionsumgebung beschreibt einen Ausschnitt des Gesamtmodells, in dem eine Konstruktionsaufgabe innerhalb eines CAD-Systems bearbeitet wird. Dies umfaßt Struktur- und Geometriedaten ebenso wie die graphische Darstellung von Teilen des Modells sowie technologische Produktinformationen. Die Größe dieses

Ausschnitts möglichst passend auf die jeweilige Aufgabe abstimmen zu können ist eine zentrale Voraussetzung für die effiziente Handhabung großer Baugruppen und damit verbundener Prozesse.

Bereits die Anwendung der Methoden zur Datenseparierung leistet dabei einen wesentlichen Beitrag. Dazu kann auf die Einführung weiterer Datenstrukturen weitestgehend verzichtet werden. Notwendig ist hingegen die durchgängige Integration des Subcluster-Konzepts sowohl auf der CAD- als auch auf der PDM-Seite. Dies erfordert hauptsächlich Anpassungen in vorhandenen Kommandos und Nutzungskonzepten, die eine feiner unterteilte Struktur der Daten reflektieren.

Die CAD-Systeme müssen in der Lage sein, die Subcluster-Struktur in den einzelnen Kommandos adäquat zu berücksichtigen. Neben der Fähigkeit, Subcluster laden und speichern zu können, bedeutet dies in erster Linie, den Wirkungsbereich der einzelnen Kommandos, der sich heute in der Regel auf komplette Cluster erstreckt, auf die jeweils involvierten Einheiten zu fokussieren. Das schließt auch das Markieren modifizierter Subcluster – statt wie bisher modifizierter Cluster – für spätere Speicheroperationen ein. Darüber hinaus sollen Veränderungen an Applikationsdaten zugelassen werden, wenn das sie besitzende Teil selbst schreibgeschützt ist. Die Realisierung dieser Integration ist weniger eine Frage des Konzepts als vielmehr eine Fleißaufgabe.

Von den PDM-Systemen wird verlangt, neben der reinen Verwaltung auch eine selektive Auswahl von Subclustern zu unterstützen und diese für gezielte Nachladeoperationen verfügbar zu machen. Daraus ergeben sich Anforderungen, die höhere Komplexität, insbesondere die Abhängigkeit der Subcluster untereinander, abzubilden und Methoden zur Versionierung bereitzustellen, auf deren Umsetzung hier jedoch nicht eingegangen werden soll.

Eine hinreichend tiefe Integration vorausgesetzt, ermöglicht die Datenseparierung Arbeitsweisen, die eine effiziente Bearbeitung großer Baugruppen erlauben. Zur Optimierung der Konstruktionsumgebung kann zunächst das Gerüst einer Baugruppe geladen werden, welches ausschließlich aus Struktur-Subclustern besteht. Mit diesem Gerüst steht eine sehr schlanke Struktur der Baugruppe zur Verfügung, auf deren

Basis in weiteren Schritten die Umgebung für eine konkrete Konstruktionsaufgabe positionserhaltend in graphischer oder geometrischer Repräsentation nachgeladen werden kann. Unterstützende Konzepte zum Aufbau des Kernbereichs der Konstruktionsumgebung unter räumlichen Gesichtspunkten werden in Kapitel 5.4.1 näher betrachtet.

Außerdem können darüber hinaus direkt involvierte oder zur Pflege der Assoziativität benötigte Applikationsdaten ergänzt werden. Die Bearbeitung der eigentlichen Aufgabe erfolgt schließlich in exakt der Umgebung, die dafür notwendig ist. Dadurch wird ein Höchstmaß an Integrität bei einem gleichzeitigen Minimum an Überschneidung der Daten erreicht und damit eine Grundbedingung für Concurrent Engineering erfüllt.

In der praktischen Anwendung läßt die Existenz von strukturrelevanten Informationen wie Teilenamen, Positionen, Mehrfachverwendung etc. bereits im Gerüst der Baugruppe umfangreiche Bearbeitungsmöglichkeiten zu. Strukturbasierte Operationen wie das Verschieben, Löschen oder Erzeugen von Teilen und Baugruppen sind ebenso möglich wie die Ableitung von Stücklisten.

Mit den graphischen Subclustern kommt Funktionalität zum dynamischen Betrachten und zur graphikbasierten Analyse hinzu. Durch die Selektion graphischer Komponenten wird ein einfaches und komfortables Ergänzen des Modells mit weiteren Daten ermöglicht. Dabei ist die visuelle Selektion der Teile wesentlich benutzerfreundlicher als die Auswahl über Teilelisten. Für Teile, die in voller Repräsentation, das heißt inklusive des geometrischen Modells resp. der Applikationsdaten, geladen werden, erhält der Benutzer schließlich die volle Modellierfähigkeit.

5.4 Räumliche Beziehungen von Teilen und Baugruppen

Die Organisation von Teilen und Baugruppen in einem CAD-Modell erfolgt in der Regel nach strukturellen und funktionalen Kriterien. Infolgedessen werden Baugruppen auf eine hierarchische Struktur aus Teilen und Unterbaugruppen abgebildet.

Insbesondere dann, wenn mehrere Konstrukteure gleichzeitig einen Entwurf bearbeiten, was bei großen Baugruppen regelmäßig der Fall ist, erweist sich diese Art der Strukturierung jedoch nicht immer als vorteilhaft.

Häufig steht der Konstrukteur vor der Aufgabe, Teile benachbarter Baugruppen zu identifizieren, die aufgrund ihrer räumlichen Anordnung für seinen eigenen Entwicklungsbereich von Interesse sind. Dies kann beispielsweise zum Bezug auf Schnittstellen, zur Bestimmung von Referenzmaßen oder zur Vermeidung möglicher Kollisionen erforderlich sein. Im folgenden sollen daher Möglichkeiten zur Verwendung räumlicher Beziehungen von Teilen und Baugruppen betrachtet werden, die die vorhandene Strukturinformation ergänzen.

Für die Beschreibung der räumlichen Ausdehnung eines Teils ist es ausreichend, dieses durch einen minimal umschließenden Quader – eine sogenannte *Bounding Box* – zu approximieren. Werden die Boxen aller Teile einer Baugruppe mit denselben Transformationen beaufschlagt wie die zugehörigen Teile, das heißt, genauso positioniert wie diese, resultiert daraus ein stark vereinfachtes Bild der Baugruppe, das zur näherungsweisen Navigation im Raum verwendet werden kann.

Stellen die Struktur-Subcluster sowohl die Boxdaten als auch die Transformationen bereit, kann diese Funktionalität bereits innerhalb eines Baugruppengerüsts genutzt werden. Für die Transformationen, die Bestandteil der Instanzdaten und damit des Struktur-Subclusters sind, ist diese Bedingung schon heute erfüllt. Eine Datenstruktur zur Verwaltung der Boxdaten wird daher entsprechend ergänzt.

Da die Basis der Struktur-Subcluster für Teile und Baugruppen identisch ist, stehen die Datenstrukturen automatisch jedem Knoten der Baugruppenstruktur zur Verfügung. Aus Gründen einer besseren Verarbeitungsgeschwindigkeit soll dies ausgenutzt werden, um jeden Knoten mit einer Box auszustatten, die der Summe der Quader aller darunterliegenden Elemente unter Berücksichtigung von deren Position entspricht. Die Berechnung dieser Boxen kann dabei zeitverzögert bei Bedarf erfolgen. Diese Zusammenfassung erfordert jedoch, daß bei der Veränderung einer Box infolge einer Modellieroperation alle darüberliegenden Boxen invalidiert werden müssen, da diese von der modifizierten abhängig sind.

5.4.1 Definition eines Kernbereichs

Die räumliche Navigation in der Baugruppenstruktur kann unter anderem dazu genutzt werden, den Aufbau einer Konstruktionsumgebung, der in Kapitel 5.3 beschrieben wird, zu erleichtern. Damit wird es möglich, die Zugehörigkeit eines Teils zu einer Konstruktionsumgebung nicht nur über logische oder technische Zusammenhänge, sondern auch über räumliche Eigenschaften baugruppenübergreifend zu definieren. Durch die zusätzliche Anwendung der Datenseparierung kann die Konstruktionsumgebung in verschiedene Bereiche mit zunehmendem Detaillierungsgrad unterteilt werden. Vorteilhaft dabei ist, daß diese Bereiche nicht mehr an die Baugruppenstruktur gekoppelt sein müssen.

Das *Gerüst* der Baugruppe stellt die Grundlage der Umgebung dar. Wegen seiner Schlankheit kann die Struktur des vollständigen Modells geladen werden. Dadurch wird die Verfügbarkeit aller Strukturinformationen, beispielsweise der Teilehierarchie, erreicht sowie die Basis für die exakte Positionierung aller Teile im Modellkontext geschaffen. Damit wird das Arbeiten im Kontext des Gesamtmodells ermöglicht.

In das Gerüst eingebettet wird die *Arbeitsumgebung*, die der Orientierung dient und durch Teile in rein graphischer Darstellung sichtbar gemacht wird. Dies liefert gleichzeitig ein vertrautes Arbeitsumfeld und bietet darüber hinaus visuelle Unterstützung zur Identifikation der Teile, an denen gearbeitet werden soll. Damit wird eine effiziente Auswahl der Teile möglich, da diese sichtbar sind und nicht mehr mühsam über die Teileliste ausgewählt werden müssen.

Der *Kernbereich* markiert schließlich das Zentrum der Arbeitsumgebung. Das ist der Bereich, an dem der Konstrukteur aktiv arbeitet. In der Regel sind die Daten hier in einem hohen, zur Bearbeitung notwendigen Detaillierungsgrad geladen. Darin eingeschlossen sind normalerweise geometrische Daten und gegebenenfalls Applikationsdaten.

Abbildung 5.19 zeigt eine schematische Darstellung von Arbeitsumgebung und Kernbereich als Ausschnitt des Gesamtmodells in Verbindung mit einem vereinfachten Beispiel. Nach einer Erhebung in [Schn02] kann von circa 20 aktiven Teilen im Kernbereich und 250 Teilen in der Umgebung als Richtwert ausgegangen werden.

Dies verdeutlicht das Einsparpotential bei Baugruppen mit einer Größenordnung von mehreren zehntausend Teilen – ohne dabei Einbußen an Funktionalität oder Integrität in Kauf nehmen zu müssen.

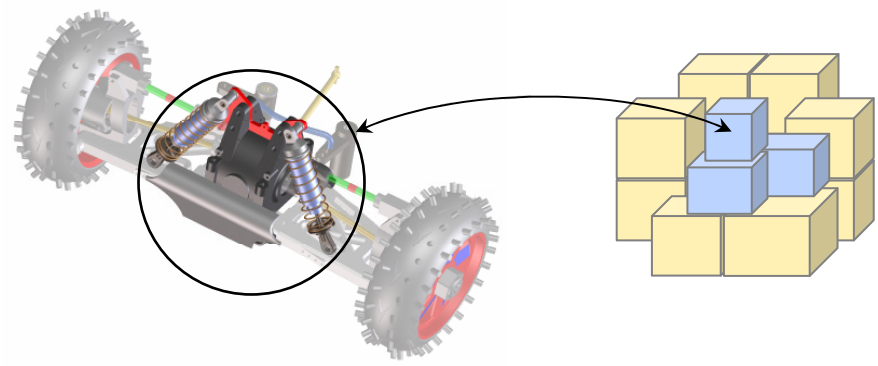


Abbildung 5.19: Arbeitsumgebung und Kernbereich

5.4.2 Selektion basierend auf räumlichen Teilebeziehungen

Auswahlmethoden, die lediglich auf der Teilehierarchie oder der graphischen Darstellung beruhen, können oftmals keine ausreichende Unterstützung bieten, wenn der Konstrukteur Teile benachbarter Baugruppen identifizieren muß. Vielfach sind die Teile nicht Bestandteil der Konstruktionsumgebung, so daß sie für eine graphische Auswahl nicht zur Verfügung stehen, oder dem Konstrukteur ist die räumliche Anordnung der Teile innerhalb der Nachbarbaugruppen nicht hinreichend bekannt.

Ausgehend von dem Quadermodell, das mit Hilfe von Boxdaten und Transformationen aufgebaut wird, die bereits für das Gerüst der Baugruppe vorliegen, ist es jedoch möglich, Nachbarschaftsbeziehungen für die gesamte Baugruppe auf Basis der Boxen effizient zu bestimmen. Vereinfachend kann angenommen werden, daß zwei Teile benachbart sind, wenn sich ihre Boxen schneiden. Darauf aufbauend können in Ergänzung zu den vorhandenen Auswahlmethoden, denen hierarchische Beziehungen zugrunde liegen, solche entwickelt werden, die sich an räumlichen Beziehungen der Teile untereinander orientieren.

Damit wird es unmittelbar möglich, alle Teile zu identifizieren, die sich in der Nachbarschaft eines Quaders befinden, das heißt, deren Boxen sich mit dem Ausgangsquader schneiden. Dieser Ausgangsquader kann ein abstrakt definierter Arbeitsraum

sein oder von der Box eines Referenzteils abgeleitet werden. Durch entsprechendes Erweitern dieses Quaders kann auf die Größe der Nachbarschaft – das heißt, auf den Abstand zum Referenzteil – Einfluß genommen werden.

Mit Hilfe dieser Funktionalität kann der Kernbereich einer Konstruktionsumgebung gezielt und baugruppenübergreifend um Daten ergänzt werden, die für die nächsten Arbeitsschritte zusätzlich benötigt werden. Das Laden ganzer Unterbaugruppen „auf Verdacht“ – wie es heute oft geschieht – wird dadurch entbehrlich. Weiterhin ist es möglich, auf einfache Weise eine Zeichnung eines Teils im Kontext einer bestimmten Einbausituation – mit allen benachbarten Teilen – abzuleiten. Darüber hinaus können Selektionsmethoden entwickelt werden, die beispielsweise auf der Grundlage des Quadermodells alle Teile bestimmen, die die äußere Hülle eines Modells beschreiben.

Überdies können die Boxen zur visuellen Unterstützung der Selektion zumindest temporär angezeigt werden. Da alle dazu erforderlichen Daten bereits im Baugruppengerüst vorhanden sind, ist dies auch für Teile möglich, von denen lediglich der Strukturanteil geladen ist. Die Daten und Algorithmen zum Aufbau und zur Auswertung des Quadermodells sind elementar, insbesondere Kenntnisse über den geometrischen Aufbau der Teile sind nicht erforderlich. Daher ist es sogar denkbar, auf räumlichen Beziehungen basierende Selektionsmethoden in ein PDM-System zu integrieren.

5.4.3 Selektion von Teilen in Abhängigkeit von ihrer Größe

Bei einigen Anwendungen, wie etwa der Ableitung von Übersichtszeichnungen, kann auf viele Details verzichtet werden. Die Größe der Teile ist dabei häufig ein wichtiges Kriterium. Teile, die eine vorgegebene Ausdehnung unterschreiten und daher keinen wesentlichen Beitrag zum Ergebnis leisten, sollen entsprechend ausgefiltert werden.

Da der vorgegebene Schwellwert ohnehin nur als Richtwert zu verstehen ist und nicht die exakte Größe der Teile von Interesse ist, kann dies hinreichend genau unter Ausnutzung der Boxinformation erfolgen. Damit ist es möglich, bereits geladene

Teile bei der Selektion für ein Kommando auszufiltern. Noch effizienter ist es jedoch, irrelevante Teile bereits vom Nachladen auszuschließen. Dies kann erreicht werden, weil die Boxinformation im Struktur-Subcluster und damit im Baugruppengerüst verfügbar ist. Damit wird nicht nur die konkrete Operation beschleunigt, sondern das System insgesamt entlastet.

Speziell bei Anwendungen, deren Schwerpunkt darauf liegt, einen übersichtlichen Gesamteindruck der Konstruktion zu vermitteln, kann darüber hinaus auf das zuvor beschriebene Quadermodell zurückgegriffen werden, um eine näherungsweise Abschätzung der Sichtbarkeit von Teilen aufgrund ihrer Größe und Lage zu erhalten. Diese wird als Grundlage für einen initialen Ladeschritt verwendet, nach dessen Abschluß weitere Teile – gegebenenfalls unter Ausnutzung räumlicher Beziehungen – ergänzt werden können.

5.5 Optimierung der Ladezeit

Bei der Analyse der Anforderungen für die Handhabung großer Baugruppen konnte unter anderem die Notwendigkeit zur Verbesserung des Prozesses zur täglichen Vorbereitung der eigentlichen Konstruktionstätigkeit festgestellt werden. Speziell die aus Konstruktionssicht unproduktive Zeit, die für die Zusammenstellung eines für die jeweilige Aufgabe geeigneten Modellausschnitts sowie das nachfolgende Laden erforderlich ist, wird oft als zu lang empfunden.

Allein durch eine stärker zielgerichtete Kombination der zur Bearbeitung benötigten Daten kann deren Menge und damit die zum Laden erforderliche Zeit verringert werden. Die Anwendung des Subcluster-Konzepts leistet dazu einen grundlegenden Beitrag. Darüber hinaus bietet die Möglichkeit, eine einmal zusammengestellte Konfiguration abspeichern und damit reproduzierbar machen zu können, weiteres Einsparpotential beim Aufbau der Konstruktionsumgebung. Verfahren zum automatischen, kontextsensitiven Nachladen während einer Operation benötigter Subcluster vergrößern dieses Potential weiter, indem sie zunächst minimale Konfigurationen zulassen, deren Ergänzung erst nach Bedarf erfolgt.

Überdies sollen die Ergebnisse der Datenseparierung genutzt werden, um den Ladevorgang in mehrere kleine und damit flexible Operationen aufzuteilen. Dadurch wird ermöglicht, Teile des Ladeprozesses in den Hintergrund oder auf Zeiten zu verschieben, in denen sie den Benutzer an der Ausübung seiner eigentlichen Tätigkeit nicht hindern. Infolgedessen kann das CAD-System eine frühere Interaktion erlauben und so zumindest die subjektiv empfundene – „gefühlte“ – Ladezeit verringern. Kann eine Sequenz der Ladeschritte erreicht werden, die frühzeitig alle für die weitere Bearbeitung wesentlichen Komponenten bereitstellt, ist dies mit einer objektiven Ladezeitverkürzung gleichzusetzen.

5.5.1 Konfigurierbare Ladezustände

Ein Teil der täglichen Rüstzeit entfällt auf die Konfiguration der Konstruktionsumgebung. Mit zunehmender Zahl von Subclustern steigt dabei auch die Vielfalt der Kombinationsmöglichkeiten und damit die Komplexität des Auswahlprozesses. Da Konstrukteure in der Regel über einen Zeitraum von mehreren Tagen oder Wochen am selben Modellausschnitt arbeiten, der nur gelegentlich ergänzt wird, wiederholt sich dieser Auswahlprozeß häufig.

Mit Hilfe einer reproduzierbaren Selektion kann daher die Rüstzeit signifikant verkürzt werden. Dazu ist eine Definition der konfigurierten Konstruktionsumgebung in geeigneter Weise abzuspeichern und zu verwalten. Diese Aufgabe fällt in den Bereich der PDM-Systeme, soll hier aber zumindest kurz beschrieben werden. Sie ist ein weiteres Indiz dafür, daß, um eine spürbare Verbesserung der Datenhandhabung zu erreichen, sowohl die PDM- als auch die CAD-Systeme eng zusammenarbeiten müssen.

Die hier betrachtete Funktionalität ist trotz einiger Parallelen nicht zu verwechseln mit dem Konfigurationsmanagement einiger CAD-Systeme, das auf einer breiten Datenbasis beruht, aber lediglich einen konfigurierten Ausschnitt daraus aktiv verwendet. Weitergehende Information ist vielfach vorhanden – teilweise sogar im laufenden System –, wird aus Gründen der Vereinfachung jedoch nicht genutzt, beispielsweise dann, wenn ein Modell nicht vollständig regeneriert wird, weil bestimmte Features unterdrückt werden sollen.

Im Gegensatz dazu spiegelt die Definition einer reproduzierbaren Auswahl einen konfigurierten Ladezustand wider. Dieser repräsentiert alle zu ladenden Cluster und Subcluster in der ausgewählten Version resp. Variante, wie etwa eine vereinfachte Darstellung eines Teils. Nicht benötigte Informationen – das heißt, nicht erforderliche Subcluster – werden hingegen nicht geladen. Die Datenbasis bleibt dadurch so schlank wie möglich.

Hinsichtlich der Berücksichtigung von Laderegeln ergeben sich zwei Klassen der Anwendung von konfigurierten Ladezuständen. Einerseits können sie der Archivierung von Konstruktionsumgebungen, die für Berechnungen oder zur Ableitung von Zeichnungen benutzt werden, als Grundlage dienen. In diesem Fall ist die Reproduktion exakt des Zustands erforderlich, der zum Zeitpunkt der Durchführung aktuell war. Damit kann eine Analyse auf einem eigens dafür zusammengestellten Modellausschnitt mit vereinfachten Teilen und in einer bestimmten Einbausituation jederzeit – gegebenenfalls unter verschiedenen Randbedingungen – neu berechnet werden.

Andererseits können konfigurierte Ladezustände so verwendet werden, daß der prinzipielle Aufbau der ausgewählten Umgebung erhalten bleibt, alle Teile jedoch immer in der neuesten Version geladen werden. Dies ist besonders während der laufenden Entwicklung sinnvoll, damit jedem Konstrukteur nach einer Ladeoperation der aktuelle Zustand benachbarter Baugruppen – soweit diese von seiner Auswahl erfaßt sind – zur Verfügung steht. Neben bereits referenzierten Teilen schließt dies solche ein, die neu hinzugekommen sind und aufgrund ihrer Lage einer zuvor definierten Nachbarschaft zugerechnet werden müssen.

Die Vielzahl denkbarer Anwendungsfälle erfordert die Möglichkeit zur Verwaltung mehrerer Konfigurationen pro Benutzer und Baugruppe ebenso wie das schnelle Auffinden einer benötigten Konfiguration. Da sich die zugrundeliegende Auswahl ändern kann, müssen entsprechende Methoden zur Pflege eines konfigurierten Ladezustands bereitgestellt werden. Optional können weitere die Konstruktionsumgebung betreffende Parameter, wie Einstellungen zur Sichtbarkeit von Teilen etc., in die Konfiguration aufgenommen werden.

5.5.2 Automatisches Nachladen

Durch das Konzept der Datenseparierung wird der Aufbau von Konstruktionsumgebungen ermöglicht, die auf die Bearbeitung einer Aufgabe optimal zugeschnitten sind. In der Praxis erweist sich jedoch oft mit Fortschreiten der Tätigkeit eine zu Beginn getroffene Auswahl als nicht ausreichend. In der Folge müssen zusätzlich benötigte Daten den Anforderungen entsprechend nachgeladen werden. Sofern das Fehlen der Daten während der Ausführung eines Kommandos vom CAD-System selbst erkannt wird, soll automatisch und transparent nachgeladen werden, ohne daß der Benutzer in seinem Arbeitsablauf gestört wird oder sogar das begonnene Kommando abbrechen muß.

Die Notwendigkeit des Nachladens wird in der Regel bei der Auswertung einer Selektion erkannt, wenn ersichtlich wird, daß Teile des Modells referenziert oder bearbeitet werden sollen, die nicht im erforderlichen Datenumfang vorliegen. Dabei ist unerheblich, ob die Selektion im Verlauf des Kommandos erfolgt oder bereits vorab und erst später mit Beginn des Kommandos ausgewertet wird.

Als Voraussetzung sowohl für die Erkennung der Situation als auch für die anschließende Ladeoperation muß für jedes Kommando exakt definiert sein, welche Typen von Dateneinheiten bearbeitet werden sollen. Ferner ist erforderlich, daß für jeden Knoten der Struktur bekannt – oder unmittelbar bestimmbar – ist, welche Subcluster auf der PDM-Seite zur Verfügung stehen. Da das nachträgliche Hinzufügen von Subclustern zu schreibgeschützten Strukturelementen möglich ist, muß diese Information im Struktur-Subcluster nicht notwendigerweise enthalten sein. Daraus kann schließlich ermittelt werden, welche Subcluster im einzelnen nachzuladen sind, um die Konstruktionsumgebung für die weitere Fortführung des Kommandos zu ergänzen. Stehen Methoden zur Verfügung, die die Identifikation abhängiger Subcluster zur Pflege der Assoziativität erlauben[19], sind auch diese nachzuladen.

Eine wesentliche Grundlage für die Transparenz des Nachladevorgangs ist die nahtlose Einbettung des Ergebnisses in den Datenfluß des Kommandos. Dabei ist insbesondere die Integration in das Transaktionsschema des UNDO-Systems sicherzustel-

[19] Vgl. Kapitel 5.2.6 Assoziativität nicht geladener Applikationsdaten

len. Darüber hinaus kann die Abbildung der Selektion von einem ausgewählten Element auf ein nachgeladenes erforderlich werden. Dies ist beispielsweise bei der Auswahl der graphischen Repräsentation einer Fläche der Fall, wenn die Fläche selbst das eigentliche Ziel der Selektion ist und erst durch Nachladen verfügbar wird.

Die Frage zu klären, ob die Subcluster direkt vom PDM-System oder aus einem eigens für diesen Zweck angelegten lokalen Cache nachzuladen sind, ist Gegenstand der Implementierung. Beide Verfahren haben Vor- und Nachteile. Während der Cachezugriff gegenüber dem Datenbankzugriff deutlich weniger Zeit in Anspruch nimmt, muß der Zwischenspeicher zunächst aufgebaut und später gepflegt werden. Einige CAD-Systeme, die bereits Lightweight-Modelle unterstützen, nutzen lokale Dateien, um das geometrische Modell auf Teileebene zu ergänzen. Da jedoch nicht a priori bestimmt werden kann, welche Subcluster potentiell nachzuladen sind, müßte für eine Cache-Lösung beim Arbeiten im Kontext des Gesamtmodells das vollständige Modell mit allen Subclustern lokal zur Verfügung gestellt werden. Die zu übertragende und zu pflegende Datenmenge kann demzufolge sehr groß werden und damit Teile des Zeitvorteils wieder zunichte machen.

5.5.3 Schrittweises Laden

Ein Ziel der Optimierung der Ladezeit ist die möglichst sofortige Systemantwort. Beim Laden einer großen Baugruppe ist es nicht im Sinne des Benutzers, nach Abschluß der Auswahl und des Ladekommandos quasi ad infinitum zu warten, bis die selektierte Konfiguration vollständig verfügbar ist und bearbeitet werden kann. Vielmehr ist er daran interessiert, mit der Vorbereitung seiner Tätigkeit – eventuell mit der Konstruktion selbst – zu beginnen, sobald alle dafür relevanten Teile geladen sind. Zu diesem Zweck ist zumindest Funktionalität zur Orientierung und Navigation im Modell sowie zum dynamischen Betrachten erforderlich. Parallel dazu können weniger wichtige Teile des Modells ergänzt werden.

Nach einem initialen Schritt zum Aufbau des Gerüsts der Baugruppe wird der Ladeprozeß in weitere kleine Schritte unterteilt und strukturiert. Die kleinste Einheit des Ladeprozesses entspricht dem vollständigen Laden eines Subclusters. Diese Einheiten werden sequentiell solange geladen, bis das Modell im ausgewählten Umfang

zur Verfügung steht. Nach jedem Schritt kann der aktuelle Zustand visualisiert werden – aus Gründen der Verarbeitungsgeschwindigkeit kann das Zusammenfassen mehrerer Schritte sinnvoll sein – und der Ladevorgang zugunsten einer anderen Operation unterbrochen werden.

Dies ermöglicht die gewohnte, uneingeschränkte Benutzung des Systems zu einem sehr frühen Zeitpunkt. In Leerlaufzeiten, das heißt, wenn der Benutzer keine weiteren Kommandos ausführt, wird der Ladeprozeß automatisch wieder aufgenommen. Der Benutzer hat jedoch jederzeit die Möglichkeit, die Ladeoperation gänzlich abzubrechen, etwa wenn der geladene Zustand für seine Belange ausreichend ist und er auf weitere Details verzichten kann.

Dieses Vorgehen kann durch verschiedene Verfahren unterstützt und ergänzt werden. Zum einen kann die Strukturierung des Ladeprozesses vom CAD-System selbst durchgeführt werden. Dazu ist die Bereitstellung von Informationen über alle Subcluster, die im Verlauf der gesamten Operation geladen werden sollen, spätestens nach Abschluß des initialen Ladeschritts erforderlich, damit diese nach vorgegebenen Kriterien ausgewertet werden können und die Festlegung einer Reihenfolge möglich ist. Diese Aufgabe kann jedoch ebenso von einem PDM-System übernommen werden, das schließlich dem CAD-System eine geordnete Liste von Ladeaufträgen übergibt.

In Multi-Threading-fähigen Systemen ist darüber hinaus bis zu einem gewissen Grad eine Parallelisierung der Prozesse erreichbar. Dadurch kann das eigentliche Laden der nächsten Einheit(en) bereits erfolgen, wenn der aktuelle Schritt visualisiert wird oder der Benutzer andere Kommandos ausführt. Lediglich die Integration der neu geladenen Subcluster in das Gesamtmodell wird auf Leerlaufzeiten verschoben, um Interferenzen mit aktiven Operationen zu vermeiden.

Die Methoden zur Strukturierung der Ladeoperation – und damit unmittelbar zur Festlegung der Sequenz zum Laden der Subcluster – sind von besonderer Bedeutung. Mit der Bearbeitung der Baugruppe bereits während des Ladevorgangs kann erst dann begonnen werden, wenn alle dafür relevanten Daten zur Verfügung stehen.

Je früher dies der Fall ist, um so geringer ist die unproduktive Wartezeit. Nach dem

Laden der Baugruppenstruktur wird daher eine Fokussierung auf den Kernbereich[20] vorgeschlagen, das heißt auf Teile, an denen der Konstrukteur regelmäßig arbeitet.

Um die subjektiv empfundene Antwortzeit des Systems zu verkürzen, erfolgt das Laden von graphischen Daten mit höherer Priorität, die nach Größe und Sichtbarkeit weiter abgestuft sein kann. Die dazu notwendigen Informationen zu Boxen und Transformationen sind bereits in der Baugruppenstruktur vorhanden. Graphische Modelle, die in mehreren Levels of Detail vorliegen, können zunächst in einer groben Detaillierungsstufe geladen und im weiteren Verlauf gegen feinere ausgetauscht werden. Dieses Verfahren ist bereits unter anderem von der Darstellung von Bildern in Web-Browsern bekannt.

Weitere Subcluster, die keinen oder nur einen geringen Beitrag zur visuellen Darstellung des Modells leisten – typischerweise sind dies Applikationsdaten –, werden, sofern sie nicht Teil des Kernbereichs sind, zum Schluß geladen.

Einige Kriterien zur Festlegung der Ladereihenfolge hängen von Informationen ab, über die in der Regel nur ein PDM-System verfügt, und sind daher von diesem, gegebenenfalls unter Auswertung der Konfigurationsdaten, in geeigneter Weise bereitzustellen. Der Vollständigkeit halber sei erwähnt, daß Operationen, die während des Ladeprozesses durchgeführt werden, Auswirkungen auf die festgelegte Ladesequenz haben können, zum Beispiel wenn ein Teil entfernt wird, von dem noch weitere Subcluster zu laden sind. Die zugrundeliegenden Datenstrukturen müssen daher auf entsprechende Ereignisse adäquat reagieren.

5.6 Einfacher Zugriff auf Applikationsdaten

Technologische Produktdaten werden zunehmend in CAD-Modelle integriert, um einen möglichst hohen Grad der Assoziativität zu erreichen. Damit verbunden ist jedoch vielfach die Schwierigkeit, einzelne Daten gezielt zu extrahieren oder sie extern zu referenzieren, ohne das 3D-Modell laden zu müssen, in das sie eingebettet

[20] Vgl. Kapitel 5.4.1 Definition eines Kernbereichs

sind. Darüber hinaus sind Daten aus der Konstruktion und verschiedenen Applikationen an unterschiedliche Prozesse gekoppelt. Sie sollen gemeinsam von einem oder parallel von mehreren Konstrukteuren bearbeitet werden können, wenngleich ihre Freigabe unabhängig voneinander erfolgt.

Der selektive Zugriff auf Dateneinheiten sowie deren Inhalt ist daher eine wesentliche Voraussetzung für einen effizienten Konstruktionsprozeß. Dies gilt um so mehr, je größer eine Baugruppe ist – und damit sowohl die Anzahl parallel tätiger Bearbeiter als auch die Datenmenge. Datenseparierung leistet in diesem Zusammenhang einen Beitrag, die Granularität der Dateistruktur zu erhöhen und dadurch einen gezielteren Zugriff auf die Daten zu erlauben.

5.6.1 Verwendung offener Dateiformate

In einem heterogenen Umfeld aus verschiedenen Applikationen, das für Konstruktionsabteilungen typisch ist, muß die Austauschbarkeit von Informationen zwischen den Anwendungsprogrammen gewährleistet sein. Da die meisten Programme proprietäre Datenformate verwenden, erfolgt die anwendungsübergreifende Interpretation von Informationen in der Regel auf der Basis einer neutralen Repräsentation. Dazu werden oft standardisierte Datenformate wie STEP oder XML eingesetzt. Mittlerweile finden auch Schnittstellen wie Web-Services oder das IFilter-Interface Verbreitung, die auf den proprietären Daten direkt arbeiten und definierte Methoden zur Verfügung stellen, deren Inhalt zu erschließen. In beiden Fällen wird eine Entkopplung der Daten vom generierenden System erreicht.

Bei der Separierung der Daten auf der Grundlage von Subclustern stellt sich das Problem des Zugriffs auf Informationen in gleicher Weise, da diese zunächst ihren Inhalt unter Verwendung von einem proprietären Datenformat persistent machen. Daher sollen generische Methoden bereitgestellt und in den Speicherprozeß eingebunden werden, mit deren Hilfe jede Applikation wesentliche Informationen in einer separaten Datei veröffentlichen kann – zusätzlich zur binären Datei oder in Ergänzung zu dieser. Die Kriterien, nach denen Daten in die binäre, die offen lesbare oder in beide Dateien geschrieben werden, hängen dabei stark von deren Art und Komplexität ab. Bei teilweise redundanter Datenhaltung ist die Überprüfung der Konsistenz zwin-

gend erforderlich, insbesondere wenn die Veränderung der offenen Daten zugelassen werden soll.

Aufgrund seiner großen Flexibilität und Offenheit bietet sich das XML-Format zum Speichern der zusätzlichen Daten an. Dabei ist jedoch zu berücksichtigen, daß dieses Format lediglich ein Werkzeug zur strukturierten Darstellung ist. Die Definition und Veröffentlichung von Schemata zur Interpretation der Daten bleibt Aufgabe der jeweiligen Applikationen.

Mit Informationen zu Referenzen zwischen Subclustern für den Aufbau eines Link-Managements sowie zur Beschreibung der Baugruppenstruktur einschließlich Transformationen und Boxen wurden im Verlauf dieser Arbeit bereits Beispiele für Daten genannt, die applikationsübergreifend von Interesse sein können. Materialeigenschaften kommen dafür ebenso in Betracht wie Fertigungsinformationen oder Zeichnungsdefinitionen. Diese Liste kann quasi beliebig fortgeführt werden.

5.6.2 Zugriffsrechte mit erhöhter Granularität

Mit der Realisierung des Konzepts zur Datenseparierung erhöht sich die Anzahl der Dateien, die Dateistruktur ist feiner untergliedert. Die dadurch gewonnene Flexibilität soll direkt auf die Verwaltung der Daten in PDM-Systemen abgebildet werden. Bislang sind Freigabeprozesse, Zugriffsrechte und Sperrvermerke für Objekte in Bearbeitung in der Regel an Teile und Baugruppen gebunden und beziehen sich auf das Objekt selbst sowie auf alle damit zusammen gespeicherten Daten. Diese Funktionalität kann jetzt wesentlich granularer und gezielter bereits auf Subclusterebene zur Verfügung gestellt werden.

Dadurch wird ein essentieller Beitrag zur Unterstützung des Concurrent Engineerings geleistet. Die gleichzeitige Bearbeitung verschiedener Applikationsdaten desselben Teils ist damit genauso möglich wie die Modifikation oder Ergänzung von Applikationsdaten an Teilen, für die der Bearbeiter selbst keine Schreibrechte hat.

Die Einführung von Varianten auf der Basis separierter Daten eröffnet neue Möglichkeiten, verschiedene Repräsentationen eines Teils abzubilden, ohne den Struktur-

knoten verändern zu müssen. Varianten des graphischen Modells erlauben beispielsweise unterschiedlich genaue Darstellungen eines Teils, Varianten des geometrischen Modells ermöglichen die Abbildung verschiedener Einbauzustände, wie etwa eine Feder im gespannten oder entlasteten Zustand. Die Versionierung von separierten Daten erlaubt dagegen die Aufzeichnung von Entwicklungsständen, die unabhängig von der Entwicklung des übergeordneten Teils sind.

Aufgrund der höheren Granularität kann der Freigabeprozeß flexibilisiert werden, da die Freigabe der einzelnen Subcluster schrittweise und unabhängig voneinander erfolgen kann. Beispielsweise besteht die Möglichkeit, nach Abschluß der eigentlichen Konstruktion zunächst nur das geometrische Modell freizugeben. Die Daten der nachgeschalteten Applikationen werden Zug um Zug ergänzt, verifiziert und schließlich ebenfalls freigegeben.

Überdies ist ein besserer Schutz geistigen Eigentums erreichbar. Die dedizierte Vergabe von Zugriffsrechten für jeden Subcluster erlaubt die Zusammenstellung von Konfigurationen, die exakt die Informationen enthalten, die für den Benutzer bestimmt sind. Darüber hinausgehende Daten, die bislang mitgeladen werden müssen, da sie mit den benötigten Daten in derselben Datei verwaltet werden, bleiben vom Ladevorgang ausgeschlossen und können mangels Berechtigung auch nicht nachgeladen werden.

In diesem Abschnitt wird erneut deutlich, daß die vorgeschlagenen Veränderungen in der Datenrepräsentation nicht allein auf Seiten der CAD-Systeme umgesetzt werden können. Dort sind unter anderem die Schreibberechtigungen auf Subcluster-Ebene statt wie bisher auf Clusterebene zu berücksichtigen. Auch innerhalb der PDM-Systeme müssen Möglichkeiten geschaffen werden, die neuen Strukturen zu verwalten und benutzergerecht zur Verfügung zu stellen.

5.7 Verringerung des Ressourcenbedarfs

Um das Bild abzurunden sollen in diesem Abschnitt einige Technologien kurz betrachtet werden, die rein unter dem Aspekt des Ressourcenbedarfs das Ziel verfol-

gen, die Handhabung großer Baugruppen in CAD-Systemen zu ermöglichen. Funktionaler Zusatznutzen oder eine darüber hinausgehende Unterstützung des Konstruktionsprozesses sind nicht im Fokus dieser Ansätze. Teilweise sind diese Technologien – zumindest dem Prinzip nach – in einigen Systemen bereits implementiert.

Modellvereinfachungen werden eingesetzt, um durch Reduktion der geometrischen Komplexität eines Teiles Ressourcen zu sparen. Einige Systeme bieten ebenfalls an, komplette Baugruppen zu einem reduzierten Teil zusammenzufassen. Die vereinfachten Elemente werden in der Regel als Austauschteile oder -baugruppen parallel zum eigentlichen Modell verwaltet. Selbst wenn die Assoziativität von vollständigen und reduzierten Modellen bis zu einem gewissen Grad gewährleistet werden kann, ist letztlich der Anwender für die Übereinstimmung beider Modelle verantwortlich. Dieses Problem wird von Verfahren umgangen, die zur Unterdrückung von Features das Modell nur bis zu einem vorgegebenen Stand regenerieren und so einen Kompromiß zwischen erhaltener Assoziativität und Ressourcenbedarf suchen.

Das *Auslagern* ganzer Teile, die längere Zeit nicht aktiv waren oder seit längerem nicht sichtbar sind, ist ein rein technischer Ansatz, der die Modelle als solche unverändert läßt. Er entspricht im wesentlichen den Paging-Verfahren, die von der Speicherverwaltung der Betriebssysteme her bekannt sind. Diese können zwar als Grundlage für die Auslagerung von Teilen dienen, sind per se aber nicht ausreichend, da sie ausschließlich in der Lage sind, zusammenhängende Speicherseiten auszutauschen. Im allgemeinen ist ein Teil jedoch durch ein komplexes Netzwerk kleiner Dateneinheiten repräsentiert, die über verschiedene Speicherseiten verteilt liegen. Eine Voraussetzung für die Auslagerung eines Teils ist daher, daß über eine geeignete Anpassung der Datenstrukturen und Algorithmen eine seitenweise Bündelung der Dateneinheiten des zugehörigen Netzwerks erreicht werden kann.

Mit wachsender Verbreitung entsprechender Hardware gewinnt die Verwendung der *64-Bit-Architektur* zunehmend an Bedeutung. Grundlagen dieser Technologie werden in Kapitel 2.5.3 beschrieben. Durch Erweiterung des Adreßraumes wird dabei die Möglichkeit geschaffen, umfangreichere Modelle zu laden. Dies bedeutet jedoch auch, daß größere Datenmengen verarbeitet werden müssen. Andererseits können

dadurch komplexe Berechnungen und Simulationen durchgeführt werden, die unter Verwendung der 32-Bit-Architektur aufgrund eines zu hohen Speicherbedarfs noch nicht möglich waren. Für eine reibungslose schrittweise Migration auf die neue Architektur ist ein temporärer Parallelbetrieb von 32- und 64-Bit-Versionen der CAD-Software erforderlich. Daraus leitet sich insbesondere die Forderung nach Kompatibilität der Dateien ab, die von den jeweiligen Systemen geschriebenen werden.

6 Realisierung grundlegender Konzepte

Die vollständige und durchgängige Integration aller beschriebenen Konzepte in ein CAD-System bedeutet einen Aufwand, dessen Umfang nicht zu unterschätzen ist. Mit Entwicklungsaufgaben, die sowohl durch Änderungen an zentralen Basisdatenstrukturen und Algorithmen in die Tiefe gehen als auch durch Anpassungen in sehr vielen Kommandos eine beachtliche Breite erreichen, summieren sich Aufwände, die einige Mannjahre umfassen. Dazu muß noch der Zeitbedarf für die Integration in ein PDM-System addiert werden.

Daraus wird unmittelbar ersichtlich, daß die vollständige Umsetzung der Konzepte im Rahmen dieser Arbeit nicht geleistet werden kann. Vielmehr soll die Integration soweit erfolgen, daß eine Beurteilung der Realisierbarkeit und Praxistauglichkeit der Konzepte dem Grunde nach möglich ist. Dazu wird das Subcluster-Konzept als Basis der Datenseparierung in ein CAD-System integriert und ergänzend auf Daten des graphischen sowie geometrischen Modells exemplarisch angewendet.

Die Integration erfolgt im Rahmen der laufenden Entwicklung des 3D-CAD-Systems OneSpace Designer Modeling und wird als Bestandteil von dessen Version 14.0 veröffentlicht. Ein Überblick zu den allgemeinen Produkteigenschaften dieses CAD-Systems ist im Anhang beschrieben. In einem weiteren Entwicklungsprojekt werden parallel zur CAD-seitigen Umsetzung des Subcluster-Konzepts grundlegende Bestandteile der Verwaltung von Subcluster-Dateien in das ebenfalls von CoCreate entwickelte PDM-System ModelManager rudimentär integriert. Die Entwicklungstätigkeit auf der PDM-Seite ist nicht Gegenstand dieser Arbeit, wohl aber eng damit koordiniert, und leistet einen wesentlichen Beitrag, um die Beurteilung insbesondere der Praxistauglichkeit zu ermöglichen.

Aus der Entscheidung, die Umsetzung des grundlegenden Konzepts in die laufende Entwicklungstätigkeit einzubetten, resultiert eine Reihe von Vorteilen. Neben der Verfügbarkeit eines bewährten und „lebenden“ Systems, das eine Vielzahl ausgereifter Algorithmen und Werkzeuge bereitstellt sowie einer ständigen Qualitätskontrolle unterliegt, ist dies vor allem die Möglichkeit, die Entwicklung unter realen Bedingungen testen und unmittelbar für den Anwender nutzbringende Funktionalität beitragen zu können.

Daraus folgen jedoch auch klar definierte Rahmenbedingungen für die Entwicklung. Zu den wichtigsten Kriterien zählen dabei Sicherheit und Kompatibilität. Große Teile der Realisierung betreffen den hochsensiblen Bereich des Ladens und Speicherns, für den die Einhaltung der Null-Fehler-Toleranz gefordert wird. Daher soll überwiegend auf bewährte und äußerst zuverlässige Konzepte zurückgegriffen werden. Die Ableitung des Subcluster-Konzepts vom Cluster-Konzept spiegelt sich darin bereits wider.

Darüber hinaus sollen Änderungen an existierendem Code in Anzahl und Umfang minimiert werden, um dadurch ein mögliches Risiko zu begrenzen. Eine Diskussion der Kompatibilitätsaspekte erfolgt bereits in Kapitel 3.7. Weiterhin ist sicherzustellen, daß die Entwicklungsversion trotz laufender Änderungen permanent funktionsfähig ist, um Ausfallkosten für die Entwicklungsabteilung zu vermeiden, die durch Kompilationsfehler oder die Analyse fehlgeschlagener Regressionstests verursacht werden.

In den folgenden Abschnitten wird zunächst die prinzipielle Vorgehensweise bei der Implementierung kurz erläutert. Anschließend werden die grundlegenden und applikationsspezifischen Datenstrukturen detailliert beschrieben. Den Entwicklungen im Bereich Laden und Speichern ist aufgrund ihrer Relevanz ein eigener Abschnitt gewidmet, im Anschluß daran erfolgt eine zusammengefaßte Betrachtung weiterer Anpassungen des Gesamtsystems.

6.1 Implementierungsschritte

Mittels Unterteilung in mehrere aufeinander aufbauende Implementierungsschritte wird die Realisierung des Subcluster-Konzepts strukturiert. Dies erfolgt zur Verrin-

gerung des Risikos, aber auch zur besseren Planung und Kontrolle des Projekts. In Koordination damit wird die Integration des Konzepts auf der Seite des PDM-Systems durch ein parallel laufendes Projekt verfolgt.

In einem ersten Schritt muß die Grundlage für die Umsetzung des Subcluster-Konzepts geschaffen werden. Dies umfaßt hauptsächlich die Implementierung der Basisdatenstrukturen sowie deren Integration in den Ablauf der Lade- und Speicherprozesse. Ebenso darin eingeschlossen sind Methoden zur Konvertierung von und zurück in Altdaten. Kapitel 6.4 befaßt sich detailliert mit dieser Thematik.

Im nächsten Schritt wird das graphische Modell separiert. Dazu sind zwei Teilaufgaben zu bearbeiten. Zum einen muß die Persistenz des Facettenmodells erreicht werden, das bislang nicht abgespeichert wird. Zum anderen ist es notwendig, die graphischen Attribute vom 3D-Modell zu trennen und so aufzubereiten, daß ihre Daten ebenfalls in eine separate Datei geschrieben werden können.

Die Äquivalenz von Cluster und Struktur-Subcluster wird für eine erste einfache Integration in das PDM-System ausgenutzt. Die vom PDM-System für das Partielle Laden dynamisch erzeugten Leerteile (vgl. Kapitel 2.5.1) können damit als Struktur-Subcluster aufgefaßt werden. Aufgrund ihrer temporären Existenz eignen sie sich nicht für eine vollständige Integration, zur Darstellung von Lightweight-Modellen im Zusammenhang mit den graphischen Dateien sind sie jedoch ausreichend.

Durch eine elementare Erweiterung der Methoden zum partiellen Laden kann somit bereits Funktionalität entwickelt werden, die sowohl die Auswahl verschiedener Ladezustände eines Teils – partiell, lightweight, vollständig – zuläßt als auch mit Hilfe des Nachladens einen späteren Wechsel in einen dieser Zustände erlaubt. Dadurch ist nicht nur die Möglichkeit gegeben, das Konzept hinsichtlich seiner praktischen Tauglichkeit zu überprüfen. Vielmehr erhält der Anwender neue Funktionalität, die er für seine Zwecke nutzbringend einsetzen kann.

Zur Vereinfachung und zur Verringerung des Aufwands werden Methoden zum automatischen Nachladen zunächst nicht implementiert. Statt dessen werden alle Kommandos angepaßt, in denen Teile geometrisch referenziert resp. modifiziert

werden können. Da dies für Lightweight-Modelle mangels geometrischer Daten nicht möglich ist, wird der Anwender gewarnt und zum Nachladen der Teile aufgefordert. Diese Warnung wird an einer zentralen Stelle in der Selektions-Methode ausgelöst, an der entsprechender Code zukünftig ohnehin eine Nachladeoperation ausführen würde.

Für die Verwendung des auf Partiellem Laden basierenden Ansatzes ist die Existenz separater Geometriedateien noch nicht erforderlich. Die Vervollständigung eines Teils – im eigentlichen Sinne des Subcluster-Konzepts das Nachladen allein des geometrischen Modells – erfolgt bei diesem Ansatz durch den Austausch des Leerteils, eventuell inklusive des graphischen Modells, gegen die vollständige Repräsentation des Teils. Daher erfolgt die Separierung des geometrischen Modells erst in einem weiteren Schritt. Wegen der weitreichenden Folgen für die Integration in das PDM-System – der Ansatz basierend auf Partiellem Laden muß dann durch eine echte, subclusterbasierte Integration abgelöst werden – wird die Anwendung des Subcluster-Konzepts auf das geometrische Modell nur prototypisch umgesetzt. Die Verwaltung der Dateien wird dementsprechend auf das Dateisystem beschränkt. Operationen zum Laden und Nachladen können jedoch durch geeignete – manuelle oder werkzeuggestützte – Kombination der einzelnen Dateien zu Testzwecken ausgeführt werden. Ebenso wird bei der prototypischen Separierung technologischer Produktdaten auf der Basis von Features verfahren.

Aufgrund der Architektur kann die vollständige Umsetzung der Konzepte in weiteren Folgeschritten erreicht werden. Dies ist für die Separierung bislang nicht betrachteter Applikationsdaten ebenso zutreffend wie für die Realisierung der anwendungsbezogenen Konzepte oder die umfassende Integration auf der Seite des PDM-Systems. Die Implementierung dieser Schritte ist jedoch nicht mehr Gegenstand dieser Arbeit.

6.2 Basisdatenstrukturen

Das in Kapitel 5.2.1 eingeführte Subcluster-Konzept, das heißt, die vollständige Unterteilung von Clustern in Subcluster, beruht auf einem Verbund weniger Klassen. Deren Aufgaben umfassen hauptsächlich die Organisation der Entities logischer Ein-

heiten als Subcluster im Sinne der Definition sowie den Aufbau und die Verwaltung der grundsätzlichen Beziehungen zwischen den Subclustern eines Clusters. Die dazu notwendigen Vorgehensweisen sind für alle Applikationen prinzipiell gleich. Um den gestellten Aufgaben gerecht werden zu können, müssen die Klassen generische Basisfunktionalität bereitstellen, die von den Applikationen spezialisiert werden kann. Dazu entwickelte Methoden und Datenstrukturen sollen im folgenden beschrieben werden.

Die Darstellung der Klassen und ihrer Beziehungen erfolgt in UML-Notation. Der besseren Übersicht halber werden nur relevante Attribute und Methoden aufgeführt. Auf obligatorisch zu überladende virtuelle Basismethoden wie `copy()`, `store()` etc. wird daher verzichtet, sofern die Methoden keinen mittelbaren oder unmittelbaren Beitrag zum Konzept leisten. Attribute und Methoden, die lediglich durch Implementierungsdetails bedingt sind, die sich aus Sonderfällen in der vorhandenen OneSpace Designer Modeling-Architektur ergeben, bleiben ebenfalls größtenteils unberücksichtigt.

Prinzipieller Aufbau der Klassen-Struktur

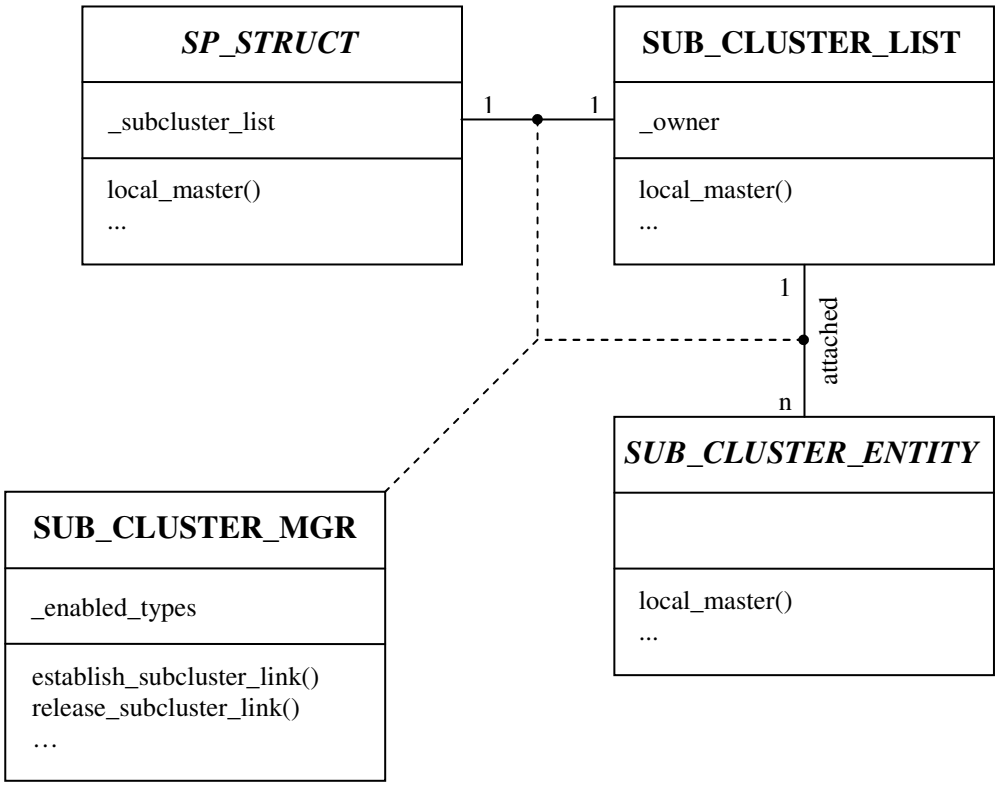


Abbildung 6.1: Prinzipieller Aufbau der Klassen-Struktur zur Separierung

Die an der Separierung beteiligten Klassen werden – mit Ausnahme der Manager-Klasse – von Entities abgeleitet, damit eine nahtlose Integration in das Entity-Netzwerk zur Beschreibung des CAD-Modells gewährleistet werden kann. Die **Abbildung 6.1** veranschaulicht die Beziehungen zwischen den Klassen im Überblick. Darin wird die 1:1-Beziehung zwischen dem Cluster Master (SP_STRUCT) und der Klasse SUB_CLUSTER_LIST deutlich, die die Verwaltung der Subcluster übernimmt – ausgedrückt durch die in Form einer Attribut-Verbindung implementierte 1:n-Beziehung zur Klasse SUB_CLUSTER_ENTITY. Der Aufbau und die Überwachung der Beziehungen werden durch die Klasse SUB_CLUSTER_MGR unterstützt, deren Einfluß durch die unterbrochene Linie dargestellt wird.

Darüber hinaus wird analog zur Methode `global_master()`, die die Kette aller Local Master traversiert, um schließlich den Cluster Master zu liefern, an zentraler Stelle die Methode `subcluster_master()` bereitgestellt. Diese ist in der Basisklasse ENTITY implementiert und folgt der Kette aller Local Master solange, bis sie den Subcluster Master erreicht.

SUB_CLUSTER_ENTITY

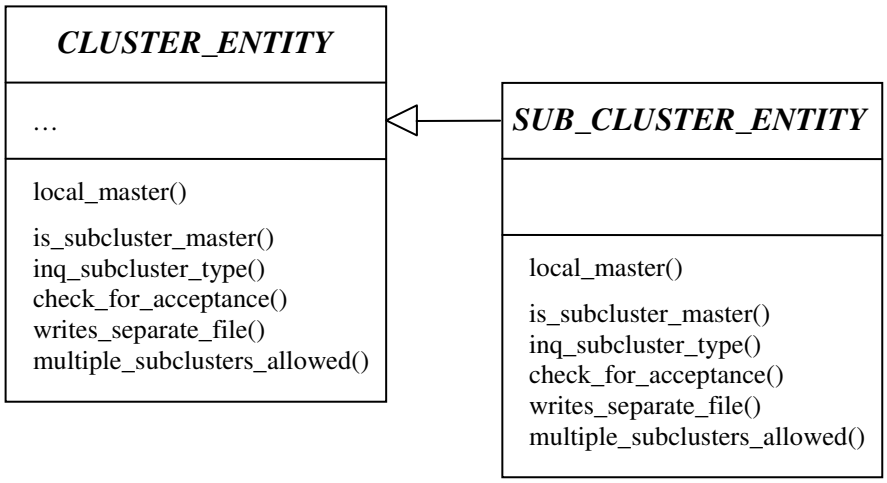


Abbildung 6.2: Die Klasse SUB_CLUSTER_ENTITY

Die Klasse SUB_CLUSTER_ENTITY stellt den Subcluster Master und damit den Repräsentant eines Subclusters. **Abbildung 6.2** zeigt die Ableitung von der Klasse CLUSTER_ENTITY. Dadurch wird sowohl die prinzipielle Eignung als Cluster Master vererbt als auch die Eigenschaft, Attribut-Verbindungen eingehen zu können,

auf denen die Beziehung zur Klasse SUB_CLUSTER_LIST beruht. Da jedem Subcluster Master applikationsspezifische Merkmale und Aufgaben zukommen, wird diese Klasse als abstrakte Basisklasse deklariert.

In der Herleitung des Subcluster-Konzepts wird von der idealen Situation ausgegangen, daß jeder Repräsentant eines Subclusters ein SUB_CLUSTER_ENTITY sei. Aus Gründen der Kompatibilität erweist sich dieser Ansatz in einzelnen Teilbereichen bei der praktischen Umsetzung jedoch als unvorteilhaft, weil dadurch ein größerer Aufwand für die Konvertierung von Altdaten erforderlich würde, der mit einer geringen Abweichung vom beschriebenen Konzept vermieden werden kann.

Betroffen davon sind die Repräsentanten des Struktur-Subclusters (SP_STRUCT) und des geometrischen Modells (BODY), die jeweils bereits Cluster-Eigenschaften besitzen. In diesen Fällen wird auf die Einführung einer Zwischenschicht in der Klassenhierarchie verzichtet. Statt von SUB_CLUSTER_ENTITY leiten diese Klassen weiterhin von CLUSTER_ENTITY ab. Virtuelle Methoden des Subcluster Entities, die von diesen Klassen ebenfalls implementiert werden müssen, werden daher in der gemeinsamen Basisklasse CLUSTER_ENTITY verankert. Dies wird auch bei der Darstellung der Klasse CLUSTER_ENTITY in **Abbildung 6.2** sichtbar. Aus rein objektorientierter Sicht ist dieses Vorgehen zwar unsauber, in der praktischen Realisierung bedingt durch die gegebenen Randbedingungen aber dennoch vertretbar.

Die Klasse SUB_CLUSTER_ENTITY implementiert folgende Methoden, die für die Realisierung des Subcluster-Konzepts von Bedeutung sind:

`local_master()` (überladen)

Liefert normalerweise einen Verweis auf das zugeordnete Entity der Klasse SUB_CLUSTER_LIST um die Zugehörigkeit zum Cluster zu dokumentieren. Während einer Speicheroperation liefert diese Methode jedoch einen Verweis auf das Entity selbst, um seine Eigenschaft als Cluster Master herauszustellen.

`inq_subcluster_type()`

Liefert den objektspezifischen Subclustertyp zur Klassifizierung des Subclusters, vgl. SUB_CLUSTER_MGR.

`is_subcluster_master()`

Gibt an, ob das Entity im aktuellen Kontext als Subcluster Master aktiv ist. Dies ist dann nicht der Fall, wenn der Subcluster selbst integrativ in den Cluster eingebunden ist. Diese Methode wird zur Bestimmung der Zugehörigkeit von Entities zu einem Subcluster benötigt und stellt insoweit die Basis für die Methode `ENTITY::subcluster_master()` dar.

`check_for_acceptance()`

Wird bereits sehr früh im Ladeprozeß gerufen – nachdem Informationen zum Subcluster und Teile des Subcluster Entities gelesen wurden – um zu prüfen, ob dieser Subcluster im gegebenen Kontext akzeptiert werden kann oder zum Beispiel aus Konsistenzgründen abgelehnt werden muß.

`writes_separate_file()`

Gibt an, ob die Daten dieses Subclusters in eine separate Datei geschrieben werden oder in den Struktur-Subcluster integriert werden sollen.

`multiple_subclusters_allowed()`

Erlaubt verschiedene Dateien für parallele Subcluster eines Typs.

SP_STRUCT

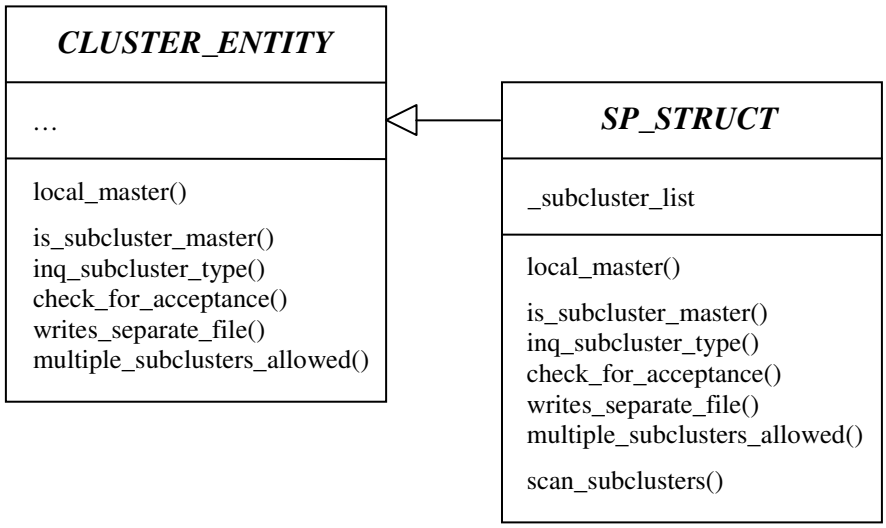


Abbildung 6.3: Die Klasse SP_STRUCT

Die Klasse SP_STRUCT existierte bereits vor Einführung des Subcluster-Konzepts. Abgeleitet von CLUSTER_ENTITY fungiert sie als abstrakte Basisklasse aller Strukturknoten und stellt somit den Cluster Master. Qua Design übernimmt dieser ebenfalls die Rolle des Repräsentanten des Struktur-Subclusters.

Auf die Einführung einer weiteren Stufe der Klassenhierarchie, die durch die Ableitung der Klasse von SUB_CLUSTER_ENTITY bedingt wäre, wird jedoch – wie bereits diskutiert – aus Kompatibilitätsgründen verzichtet. Statt dessen erbt die Klasse SP_STRUCT alle für die Umsetzung des Subcluster-Konzepts relevanten Methoden direkt von CLUSTER_ENTITY und überlädt diese entsprechend. Dies wird in **Abbildung 6.3** verdeutlicht.

Über die Methoden hinaus, die von jedem Subcluster zur Verfügung zu stellen sind, bzw. abweichend davon, implementiert die Klasse SP_STRUCT folgende Methoden:

`local_master()` (überladen)

Liefert in seiner Eigenschaft als Cluster Master immer einen Verweis auf sich selbst.

`scan_subclusters()`

Ruft eine mitgegebene Task-Funktion für jeden Subcluster dieses Clusters auf.

Zusätzlich wird die Klasse SP_STRUCT um ein Datenfeld ergänzt. Die Variable `_subcluster_list` hält einen Verweis auf ein Objekt der nachfolgend beschriebenen Klasse SUB_CLUSTER_LIST. Durch diesen Verweis – sowie das entsprechende Pendant in der Rückrichtung – wird die 1:1-Beziehung zwischen den beiden Entities implementiert. Der Cluster Master delegiert alle Aufgaben im Zusammenhang mit der Verwaltung der Subcluster an dieses Objekt und kontrolliert dessen Lebenszeit. Das Datenfeld selbst ist innerhalb der Klasse SP_STRUCT gekapselt, die Methoden zum Zugriff darauf werden jedoch der besseren Übersicht halber nicht weiter betrachtet.

SUB_CLUSTER_LIST

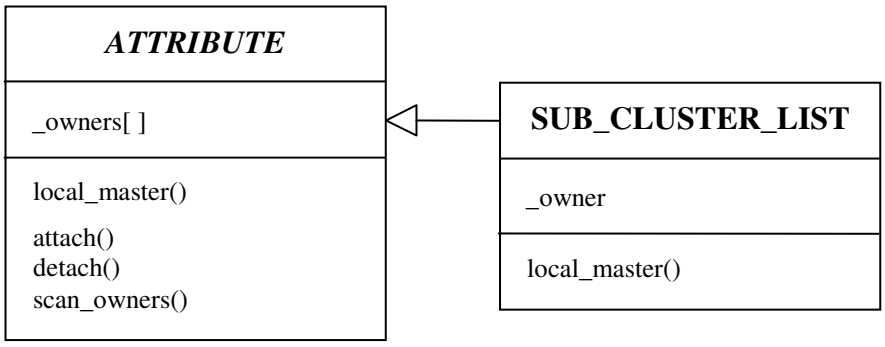


Abbildung 6.4: Die Klasse SUB_CLUSTER_LIST

Die Verwaltung aller Subcluster innerhalb des übergeordneten Clusters obliegt der Klasse SUB_CLUSTER_LIST. Sie stellt das Bindeglied zwischen dem Struktur-Subcluster und allen weiteren Subclustern dar. Gemäß **Abbildung 6.4** wird diese Klasse von ATTRIBUTE abgeleitet und erbt somit Funktionalität, die den Aufbau generischer Attribut-Beziehungen ermöglicht. Dies sind 1:n-Beziehungen zwischen einem Attribut und mehreren geeigneten Zielobjekten. Während die assoziierte Eigenschaft für das Zielobjekt eindeutig sein muß, können viele Zielobjekte dasselbe Attribut besitzen.

Auf das Subcluster-Konzept bezogen stellen die Subcluster Entities die Zielobjekte dar, die assoziierte Eigenschaft entspricht der Zugehörigkeit zu einem Cluster, die durch ein Entity der Klasse SUB_CLUSTER_LIST abgebildet wird. Damit wird die Konsistenz dieser Verbindung bereits durch die Verwendung von Basisfunktionalität sichergestellt. Der Aufbau der Beziehung – das heißt, die Zuordnung eines Subclusters zum Cluster – erfolgt durch die Methode `attach()`, die Auflösung erfolgt entsprechend durch `detach()`. Mit Hilfe der Methode `scan_owners()` kann eine Task-Funktion für alle Zielobjekte aufgerufen werden. Dadurch können alle Subcluster eines Cluster auf einfache Weise erreicht werden.

Darüber hinaus unterstützen Attribute bereits die Verwendung externer Referenzen auf ihre Zielobjekte und sind prinzipiell für Open Reference Handling[21] vorbereitet. Die Möglichkeit, Dateien abhängiger Subcluster während des Ladens eines Struktur-

[21] Vgl. Kapitel 2.5.2 Open Reference Handling

Subclusters automatisch nachzuziehen, ist somit quasi ein Nebeneffekt der Ableitung von der Klasse ATTRIBUTE. Im Rahmen des Subcluster-Konzepts ist lediglich eine weitere Funktion zu implementieren:

`local_master()` (überladen)

Liefert einen Verweis auf den Cluster Master – SP_STRUCT – und integriert dadurch alle Subcluster in den übergeordneten Cluster.

Die Klasse SUB_CLUSTER_LIST wird ebenfalls um ein Datenfeld ergänzt. Die Variable `_owner` ist das Gegenstück zu `SP_STRUCT::_subcluster_list` und vervollständigt die 1:1-Beziehung zum Cluster Master. Dieses Datenfeld ist ebenfalls gekapselt.

SUB_CLUSTER_MGR

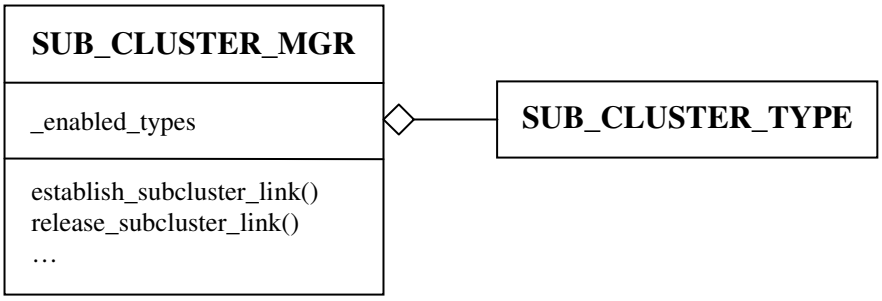


Abbildung 6.5: Die Klasse SUB_CLUSTER_MGR

Zu den Aufgaben der Klasse SUB_CLUSTER_MGR (vgl. **Abbildung 6.5**) gehört die Verwaltung klassenübergreifender Eigenschaften ebenso wie der Aufbau und die Überwachung der Verbindungen zwischen den einzelnen Subclustern. Die Klasse implementiert ein Singleton Pattern – es existiert genau ein Manager im laufenden System – und hat kein Wissen über die Subcluster als solche. Dieses Wissen liegt vollständig in der Datenstruktur selbst und wird durch die Beziehungen der Subcluster untereinander repräsentiert.

Neben einer Reihe kleinerer Hilfsfunktionen, die hier nicht näher betrachtet werden sollen, implementiert die Klasse zwei Methoden, die für die Verwaltung der Subcluster-Beziehungen essentiell sind:

`establish_subcluster_link()`

Baut die Verbindung zwischen Subcluster und dem Verwaltungs-Entity vom Typ SUB_CLUSTER_LIST auf. Letzteres wird bei Bedarf automatisch erzeugt. Dies ist insbesondere beim Laden von Altdaten erforderlich.

`release_subcluster_link()`

Beendet die Eigenständigkeit eines Subclusters durch Auflösen der Verbindung zum Verwaltungs-Entity. Dies wird nicht nur beim Löschen eines Subclusters notwendig, sondern auch, wenn dieser wieder in den Cluster integriert werden muß, was für das rückwärtskompatible Speichern in alte Dateiformate von großer Bedeutung ist.

Zu den Daten der Manager-Klasse gehört das Bit-Feld `_enabled_types`, mit dessen Hilfe die Datenseparierung einzelner Subcluster-Typen aktiviert oder deaktiviert werden kann. Dieses Feld wird für Kompatibilitäts- und Testzwecke verwendet und unterstützt die schrittweise Einführung des Subcluster-Konzepts. Normalerweise sind alle registrierten Subcluster-Typen aktiv.

Darüber hinaus aggregiert die Klasse SUB_CLUSTER_MGR einen Enumerator-Typ zur Klassifizierung von Subclustern. SUB_CLUSTER_TYPE definiert Konstanten, die den jeweiligen Subcluster-Typen eindeutig zugeordnet werden können. Neu hinzukommende Subcluster-Typen müssen folglich entsprechend registriert werden.

6.3 Applikationsspezifische Datenstrukturen

Die im vorhergehenden Kapitel beschriebenen Datenstrukturen stellen das Grundgerüst für die Datenseparierung dar und liefern die dazu benötigte generische Funktionalität. Um auf dieser Basis die Daten einer konkreten Applikation separieren zu können, müssen die Klassen und Methoden unter Berücksichtigung der lokalen Bedingungen spezialisiert werden. Dazu sind die nachfolgend genannten Regeln einzuhalten.

1. Für jede Applikation ist eine dedizierte Entity-Klasse erforderlich, die den Subcluster Master stellt. In der Regel wird diese durch direkte oder indirekte Ableitung von der Klasse SUB_CLUSTER_ENTITY bereitgestellt. In Ausnahmefällen kann eine von CLUSTER_ENTITY abgeleitete Klasse verwendet werden, sofern diese der Applikation eindeutig zugeordnet werden kann.
2. Die in der Basisklasse verankerten virtuellen Methoden zum Aufbau und zur Verwaltung der Subcluster-Struktur müssen überladen und den lokalen Gegebenheiten entsprechend angepaßt werden.
3. Jede Applikation ist bei der Manager-Klasse SUB_CLUSTER_MGR mit einem eindeutigen Subcluster-Typ zu registrieren.
4. Ansonsten ist die Einhaltung des in Kapitel 5.2.1 beschriebenen allgemeinen Regelwerks für Subcluster zu gewährleisten.

Im folgenden soll die Realisierung des Subcluster-Konzepts exemplarisch an Klassen skizziert werden, mit deren Hilfe die Separierung der graphischen Daten bzw. des geometrischen Modells erreicht wird. Ergänzend wird die Klassenstruktur zur Separierung von Features in expliziter Darstellung beschrieben.

GA_MASTER

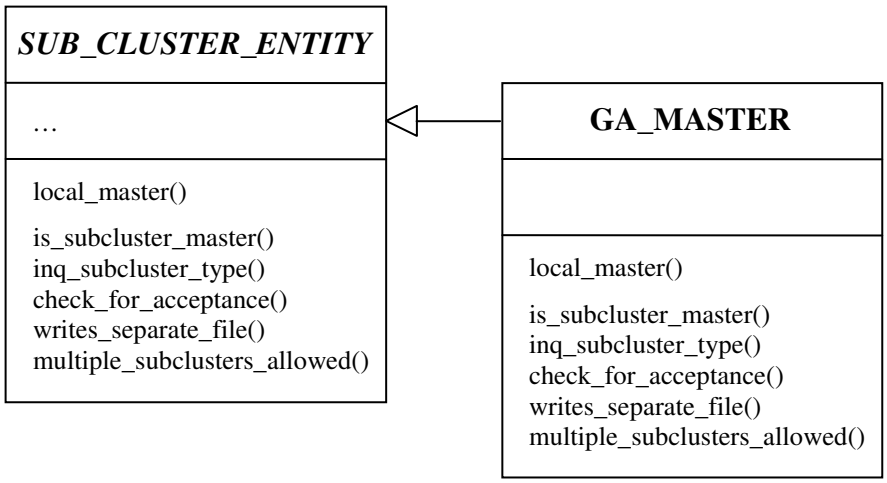


Abbildung 6.6: Die Klasse GA_MASTER

Die Klasse GA_MASTER wird von SUB_CLUSTER_ENTITY abgeleitet (vgl. **Abbildung 6.6**) und übernimmt die Verwaltung der graphischen Attribute. Bislang liefert die `local_master()` Methode eines solchen Attributs einen Verweis auf das

besitzende graphische Topologieelement und bewirkt dadurch, daß das Attribut dem geometrischen Modell zugerechnet wird. Da die graphischen Topologieelemente nach der Separierung zusammen mit dem Facettenmodell einen eigenständigen Subcluster bilden, der im folgenden Abschnitt beschrieben wird, kann diese Verbindung nicht aufrecht erhalten werden. Der Pfad vom graphischen Attribut zum Subcluster Master würde sonst über das Gebiet eines fremden Subclusters verlaufen. Daher wird die Klasse GEM_FA_ATT[22] eingeführt, die zum einen die Attribute eines graphischen Topologieelements gruppiert und diesem zuordnet, sowie zum anderen die Verbindung zum GA_MASTER herstellt. **Abbildung 6.7** stellt die Situation am Beispiel der graphischen Attribute einer GEM_FACE dar.

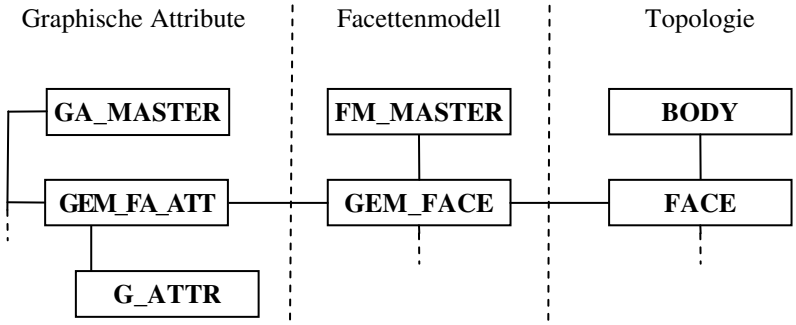


Abbildung 6.7: Verwaltung graphischer Attribute

FM_MASTER

Das Facettenmodell wird durch die Klasse FM_MASTER repräsentiert, die gemäß **Abbildung 6.8** von SUB_CLUSTER_ENTITY abgeleitet ist. Durch die Trennung der Attribute von der graphischen Topologie und die Anpassung der Local Master von GEM_FACE etc. wird bereits das Grundgerüst dieses Subclusters definiert.

Die eigentlichen Facettendaten werden bisher von isolierten Entities dargestellt, die in direkter einseitiger Abhängigkeit zur graphischen Topologie stehen und nicht abgespeichert werden. Mit entsprechender Implementierung der `local_master()` Methode wird sichergestellt, daß jetzt alle Elemente ihre direkte Zugehörigkeit zum Subcluster Master ausdrücken und dadurch ebenso ihre Persistenz erreicht wird.

---

[22] Für Faces; analog dazu GEM_ED_ATT und GEM_VX_ATT für Edges bzw. Vertices (vgl. Abbildung 5.12)

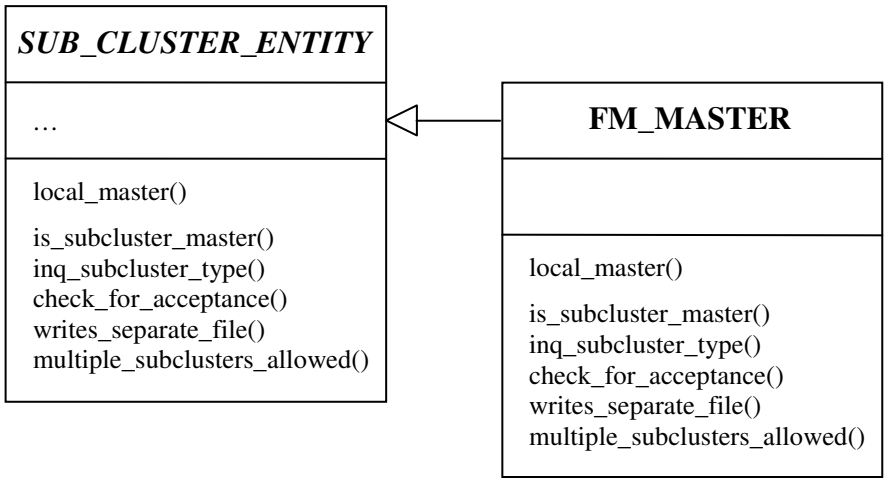


Abbildung 6.8: Die Klasse FM_MASTER

BODY

Mit der Erweiterung der Klasse BODY um Subcluster-Funktionalität wird das geometrische Modell separiert. Als Repräsentant dieser Dateneinheit erfüllt das Body-Entity bereits alle Grundvoraussetzungen für einen Subcluster Master. Die in **Abbildung 6.9** dargestellte Ableitung von der Klasse CLUSTER_ENTITY bleibt daher aus Kompatibilitätsgründen – abweichend vom Konzept – unverändert.

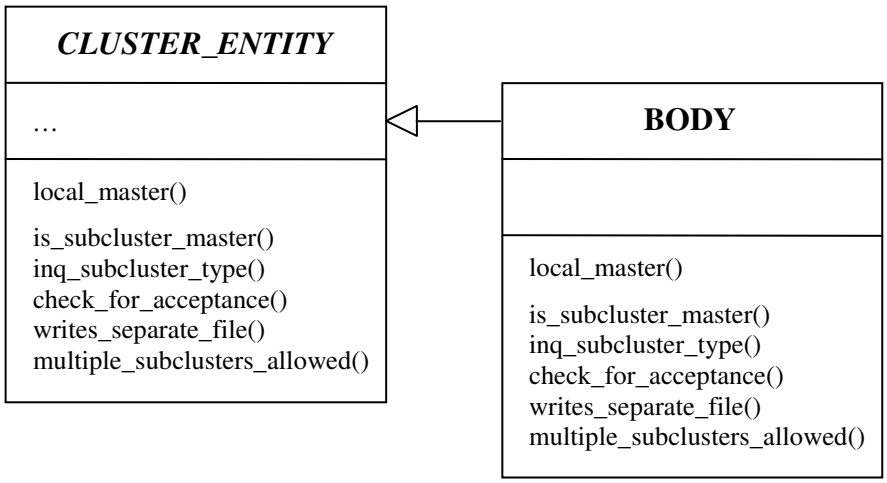


Abbildung 6.9: Die Klasse BODY

Der Aufwand für die eigentliche Separierung der Daten ist somit begrenzt. Um so mehr fällt jedoch die durchgängige Integration der Geometriedateien – insbesondere in das PDM-System – ins Gewicht, die die Möglichkeiten des gewählten, Leerteil-basierten Ansatzes übersteigt. Die Umsetzung erfolgt deshalb nur als Prototyp.

FEATURE_SUB_CLUSTER

Die an der Beschreibung eines Features beteiligten Entities bilden eine in sich abgeschlossene Einheit im Sinne eines Clusters, jedoch ohne einen dedizierten Repräsentant zu besitzen. Zur Anpassung dieser Einheit an das Subcluster-Konzept ist es daher ausreichend, ein neues – von SUB_CLUSTER_ENTITY abgeleitetes – Entity zu integrieren, das die Aufgabe des Subcluster Masters übernimmt und in bidirektionaler Verbindung zum Feature steht.

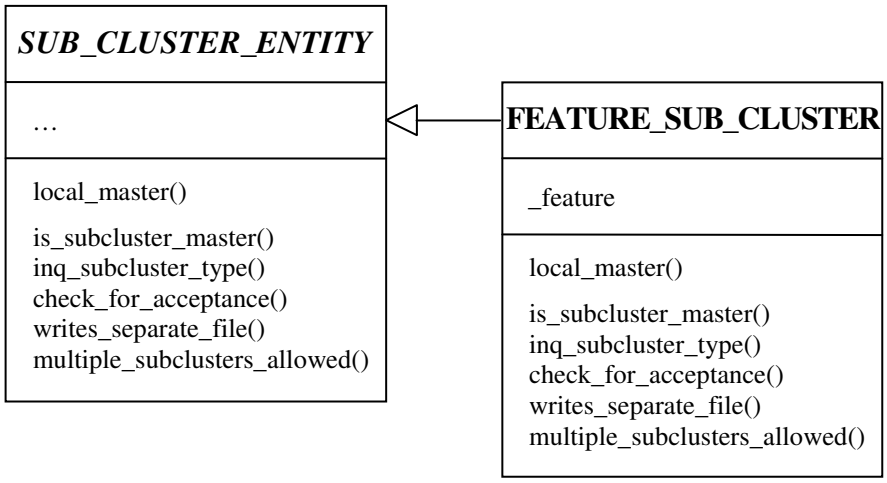


Abbildung 6.10: Die Klasse FEATURE_SUB_CLUSTER

Die in **Abbildung 6.10** gezeigte verallgemeinerte Darstellung eines Feature Masters kann ausschließlich für eine prototypische Realisierung verwendet werden, in der alle Features – gleich zu welcher Applikation sie gehören – undifferenziert nebeneinander existieren. Um eine produkttaugliche Implementierung zu erreichen, muß dagegen jeder Applikation ein dediziertes Entity zugeordnet werden, welches von dem hier gezeigten abgeleitet ist.

6.4 Laden und Speichern

Die Anpassungen der vorhandenen Algorithmen und Datenstrukturen zum Laden und Speichern nehmen einen breiten Raum in der Realisierung des Subcluster-Konzepts ein. Die wichtigsten Entwicklungen sollen im folgenden kurz umrissen werden.

Zunächst werden die Kommandos zum Speichern von Daten und die darin verwendeten internen Datenstrukturen so erweitert, daß eine gezielte Auswahl der zu speichernden Subcluster möglich ist. Weiterhin kann spezifiziert werden, ob Subcluster eigenständige Dateien schreiben oder eine kompatible Dateistruktur erzeugt wird, die jedoch intern auf Subclusterbasis untergliedert ist.

Bei der Ausführung einer Speicheroperation wird davon ausgegangen, daß die Daten im System bereits in separierter Form vorliegen. Beginnend mit dem Struktur-Subcluster werden alle Subcluster eines Clusters sequentiell bearbeitet. Sofern ein Subcluster aktiv – somit nicht temporär in den Struktur-Subcluster integriert – und Bestandteil der Auswahl ist, wird der vorhandene Code zum Speichern eines Clusters durchlaufen. Lediglich die Scout-Funktion[23], mit deren Hilfe die Zugehörigkeit eines Entities zum Cluster bestimmt wird, muß so geändert werden, daß sie prüft, ob ein Entity dem gerade bearbeiteten Subcluster zugeordnet ist. Darüber hinaus werden die Methoden zur automatischen Dateinamengenerierung erweitert.

Der größere Informationsumfang der Dateien erfordert ebenfalls neue Schlüsselwörter für die Binärkodierung der Daten. Diese werden zur Kennzeichnung von Referenzen zwischen Subclustern benötigt, aber auch um Informationen zu sequentialisieren, die der Identifikation des Subclusters selbst sowie des übergeordneten Clusters dienen. Diese Daten erscheinen im Prolog der Subclusterdaten und erlauben die spätere Anmeldung eines Subclusters beim übergeordneten Cluster.

Für Ladeoperationen wird die Regel eingeführt, daß Subcluster ausschließlich dann akzeptiert werden können, wenn deren übergeordneter Cluster entweder bereits als Resultat einer früheren Ladeoperation im Hauptspeicher verfügbar ist oder zumindest Bestandteil derselben Ladeoperation ist, aber bereits prozessiert wurde. Dadurch können alle existierenden Laderegeln bezüglich Exemplarisierung, SysID-Vergabe etc. vom Cluster auf den Subcluster vererbt werden. Dies vereinfacht den Ladevorgang wesentlich, ohne für den Anwender eine wahrnehmbare Restriktion darzustellen.

---

[23] Vgl. Kapitel 2.1.2 Cluster-Konzept und Kapitel 5.2.1 Subcluster-Konzept

Lade-Deskriptoren – Hilfsklassen, die alle clusterspezifischen Informationen während des Ladevorgangs verwalten und für die Zuordnung von Clustern und Referenzen benötigt werden – müssen für die Zuordnung von Subclustern erweitert werden. Zusätzlich sind zur Verwaltung der subclusterspezifischen Informationen separate Subcluster-Deskriptoren notwendig.

Außerdem erfolgt eine Integration von Konsistenz- und Akzeptanzprüfungen für Subcluster in den Ablauf des Ladevorgangs. Bereits nach dem Lesen der subclusterbeschreibenden Informationen aus dem Prolog der Subclusterdaten und dem Anlesen des Subcluster Masters – das ist das erste Entity in der Folge der Subclusterdaten – kann entschieden werden, ob der Subcluster im gegebenen Kontext akzeptiert werden soll. Diese Entscheidung trifft jede Applikation mittels Überladen der Methode `CLUSTER_ENTITY::check_for_acceptance()` selbst. Als Kriterien können beispielsweise die Cluster-Daten sowie Version und Zeitstempel des Subclusters herangezogen werden. Falls der Subcluster Akzeptanz findet, werden die weiteren Daten bearbeitet, ansonsten wird die Datei bis zum Ende des Subcluster-Abschnitts überlesen.

Da die Subcluster eines Clusters in verschiedenen Arbeitssitzungen abgespeichert worden sein können, wodurch die clusterweite Eindeutigkeit der Kennung von internen Referenzen nicht mehr gegeben ist, muß eine subclusterbasierte Vorkonvertierung der internen Referenzen erfolgen. Dies geschieht direkt nach dem Laden eines Subclusters durch Abbildung der Entity-Kennungen auf aktuelle Speicheradressen. Die Metadaten zur Verwaltung der Referenzinformation gehen nach entsprechender Aktualisierung in die spätere Gesamtkonvertierung ein. Externe Referenzen auf Entities in verschiedene Subcluster eines Clusters werden in einer Tabelle innerhalb des Clusters gesammelt. Somit wird die Konvertierung externer Referenzen nach wie vor auf der Basis von Clustern durchgeführt.

Weitere Anpassungen und Hilfsfunktionen sind sowohl im Kernbereich als auch in den Applikationen notwendig, um kompatibel mit Altdaten zu bleiben. Dies schließt die Konvertierung beim Lesen von Dateien älterer Versionen ebenso ein wie das Schreiben dieser Formate.

6.5 Weitere Anpassungen des Gesamtsystems

Über die Implementierung des Subcluster-Konzepts hinaus – das heißt, zusätzlich zur Separierung von Daten im laufenden System und der Funktionalität zur Verwaltung der Untereinheiten in eigenständigen Dateien – sind weitere Anpassungen notwendig, um die veränderten Datenstrukturen durchgängig im Gesamtsystem zu berücksichtigen. Bislang konnte in allen Kommandos davon ausgegangen werden, daß ein Teil entweder in vollständiger Repräsentation vorliegt – mit graphischem *und* geometrischem Modell – oder aber als Leerteil geladen ist. Letzteres kommt ohne geometrisches Modell aus, verfügt jedoch auch nicht über eine graphische Darstellung, kann somit insbesondere nicht graphisch selektiert werden.

Mit der Einführung der Lightweight-Modelle hat sich diese Bedingung grundlegend geändert. Diese Teile sind sichtbar und können ebenso über ihre graphischen Elemente ausgewählt werden. Dementsprechend werden alle Kommandos auf ihre Modellanforderungen hin überprüft und gegebenenfalls so geändert, daß die Auswahl von Lightweight-Modellen unterbunden wird. De facto wird diese Änderung an zentraler Stelle in der Selektionsmethode durchgeführt, so daß der Benutzer in einem solchen Fall aufgefordert wird, zunächst eine vollwertige Repräsentation des gewählten Lightweight-Teils nachzuladen. Die einzelnen Kommandos übermitteln lediglich das erforderliche Auswahlverhalten an den Selektor.

Darüber hinaus erfolgt die Anpassung einer Vielzahl von Kommandos, die nicht bereits bei der Auswahl explizit mit Lightweight-Teilen konfrontiert werden, sondern erst im weiteren Verlauf der Bearbeitung des Modells. Die Bandbreite reicht dabei von Fly-by Highlight über das Kopieren von Baugruppen bis zur Zeichnungsableitung. Seitens des PDM-Systems werden hauptsächlich die Verwaltung von Subcluster-Dateien implementiert sowie die Methoden zum partiellen Laden bzw. Nachladen erweitert. Überdies wird Funktionalität zur Migration der Bestandsdaten bereitgestellt, die die Ergänzung von graphischen Daten zu bereits vorhandenen Teilen erlaubt.

6.6 Diskussion der erzielten Ergebnisse

Im Mittelpunkt dieser Arbeit steht die Entwicklung einer Datenstruktur, die eine benutzergerechte Handhabung großer Baugruppen in 3D-CAD-Systemen ermöglicht. Insbesondere soll damit die bisher existierende Lücke zwischen einer hohen Integration technologischer Produktdaten in ein CAD-Modell einerseits und selektivem Zugriff auf diese Daten andererseits geschlossen werden.

In den vorangegangenen Abschnitten wird dazu das Konzept der Datenseparierung eingeführt. Darüber hinaus werden Vorschläge erarbeitet, wie dieses Konzept unter anderem zur Optimierung von Ladezeit und Konstruktionsumgebung einzusetzen ist und wie eine granulare Vergabe von Zugriffsrechten sowie die Auswertung räumlicher Beziehungen auf dessen Basis realisiert werden können.

Nachfolgend soll untersucht werden, inwieweit die beschriebenen Konzepte und Datenstrukturen den Anspruch erfüllen, einen Fortschritt bei der Handhabung großer Baugruppen zu erzielen. Das Basiskonzept, zugehörige Datenstrukturen als Fundament desselben und einige Anwendungen wurden zu diesem Zweck teils produktreif, teils als Prototyp im Rahmen der Entwicklung von OneSpace Designer Modeling 14.0 softwaretechnisch umgesetzt. Dies hat gleichzeitig deren prinzipielle Realisierbarkeit unter Beweis gestellt. Eine Reihe einzelner Teilaspekte konnte bereits während der Entwicklungsphase getestet werden. An dieser Stelle erfolgt die Zusammenfassung dieser Ergebnisse durch die ganzheitliche Betrachtung der Realisierung anhand eines Beispiels. Dazu werden die Daten einer großen Baugruppe separiert und anschließend zum Aufbau einer Umgebung für eine fiktive Konstruktionsänderung verwendet.

6.6.1 Testumgebung

Testsystem

Die Untersuchungen erfolgen auf der Basis einer Entwicklungsversion des CAD-Systems OneSpace Designer Modeling 14.0, die prototypische Programmergänzungen zur Separierung von Geometrie- und Applikationsdaten enthält. Letztere ermög-

lichen die Bildung von Subclustern sowohl für featurebasierte Applikationsdaten – in generischer, undifferenzierter Weise – als auch für Zeichnungsdefinitionen, die auf einer fortgesetzten Struktur beruhen.

Mit dieser Programmversion ist das Schreiben und Laden von Subcluster-Dateien mit Hilfe des Dateisystems möglich. Auf die Integration in ein PDM-System kann daher verzichtet werden. Unter der Voraussetzung, daß das Laden der Struktur-Subcluster immer vollständig erfolgt, können beliebige Kombinationen weiterer Subcluster ergänzt werden. Die Daten der teileumschließenden Boxen stehen in der Struktur zur Verfügung. Auf eine weiterreichende Integration der Datenseparierung in die Kommandos wurde jedoch verzichtet.

Testmodell

Das in **Abbildung 6.11** gezeigte CAD-Modell dient als Grundlage für die folgenden Betrachtungen. Das Unternehmen Liebherr-Werk Ehingen GmbH hat das Mastermodell dieses Mobilkrans freundlicherweise für Test- und Verifikationszwecke im Rahmen dieser Arbeit zur Verfügung gestellt. Die Baugruppe umfaßt einen kompletten Mobilkran mit Anbauteilen wie Winden, Ballastierung etc. und ist das Ergebnis eines Konstruktionsprozesses, der auf der Verwendung von OneSpace Designer Modeling beruht. Die Ursprungsdaten spiegeln daher den Zustand der Datenverwaltung ohne Datenseparierung wider und eignen sich somit sehr gut für einen Vergleich.

Die Konstruktion gliedert sich in mehrere Hauptbaugruppen – wie Oberwagen und Unterwagen – und verwendet eine streng hierarchische Struktur auf der Basis von Teilenummern. Diese Organisation des Modells ist quer durch alle Konstruktionsabteilungen bekannt und akzeptiert. Die Bearbeitung findet in der Regel im Kontext einer Haupt- oder Oberbaugruppe statt. Diese wird partiell geladen und um erforderlich Teile bzw. Unterbaugruppen ergänzt. Dazu wird die Kenntnis der Struktur ausgenutzt, jedoch erfolgt die Auswahl, um die dafür benötigte Zeit in Grenzen zu halten, eher grob und umfangreich. In einzelnen Fällen wird dieses Verfahren auch auf die Gesamtstruktur angewendet.

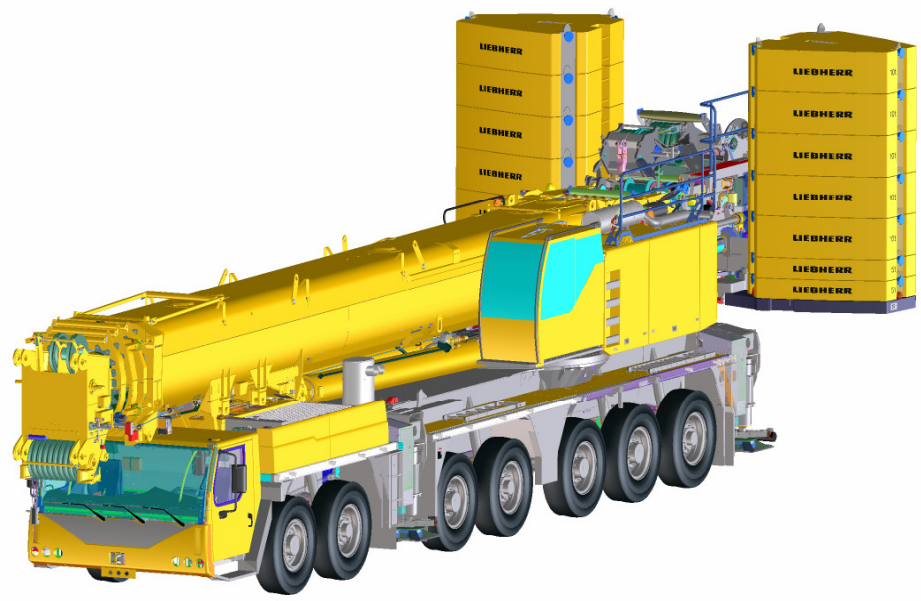


Abbildung 6.11: Im Test verwendetes Modell

Die PDM-Systeme von CoCreate verfügen über Methoden, Daten einer Baugruppe zunächst in eine Datei zu exportieren, die zu einem späteren Zeitpunkt mit OneSpace Designer Modeling eingelesen werden kann. Dadurch ist es möglich, die Daten entkoppelt von den Systemen eines Anwenders in ein anderes Zielsystem zu laden. Auf diese Weise wurden für diesen Test sowohl Dateien bereitgestellt, die die Gesamtbaugruppe enthalten als auch solche, die jeweils eine Hauptbaugruppe umfassen.

Das Modell repräsentiert eine im wesentlichen vollständige Fertigungsgrundlage. Details wie elektrische und hydraulische Komponenten fehlen jedoch teilweise. Bei den technologischen Produktdaten ergibt sich ein gemischtes Bild. Diese sind zum einen in das Modell integriert – Zeichnungsdefinitionen, Notizen oder Schweißverbindungen – und werden zum anderen noch extern gehalten – Analysedaten.

Insgesamt kann die Größenordnung dieser Baugruppe durch die in **Abbildung 6.12** dargestellten Kennzahlen beschrieben werden. Von der Gesamtzahl der Teile entfallen ca. 20.000 auf den Unterwagen, ca. 28.000 auf den Oberwagen, der Rest auf weitere Baugruppen. Die Verteilung der verschiedenen Datentypen wurde im Vorgriff auf den folgenden Abschnitt anhand der separierten Daten bestimmt.

Strukturelle Komplexität	
Teile insgesamt	51.000
Teile je Baugruppe	< 400
Strukturtiefe	11
Wiederverwendung	80 %

Datenanteile	
Struktur	4%
Geometrisches Modell	54 %
Graphisches Modell	37 %
Applikationsdaten	5 %

Abbildung 6.12: Kennzahlen der Testbaugruppe

6.6.2 Testresultate

Um Aussagen hinsichtlich des Bedarfs an Hauptspeicher und Ladezeiten treffen zu können, werden vier Varianten des Modells geladen. Zunächst wird die vollständige Baugruppe betrachtet. Die Resultate kommen dem Verhalten ohne Datenseparierung gleich und dienen als Referenzwerte für die weiteren Messungen. In der zweiten Variante werden ausschließlich die Struktur-Subcluster geladen. Dies entspricht dem Gerüst der Baugruppe und stellt gleichzeitig die Minimalanforderung dar. Die folgende Messung betrifft das komplette Modell in graphischer Darstellung. Schließlich wird am Beispiel einer fiktiven Modelländerung eine Konstruktionsumgebung aufgebaut und analysiert. Die Zusammenfassung der einzelnen Resultate erfolgt zum Abschluß dieses Abschnitts.

Laden des kompletten Modells

Das Laden des Gesamtmodells im bisherigen Datenformat – alle Daten einer Komponente der Baugruppe sind in einer gemeinsamen Datei gebündelt – soll gleichzeitig genutzt werden, um die Daten in die separierte Form zu überführen. Ein erster Versuch zeigt jedoch, daß dies mit gegebener Hardware – ein 64-Bit-System mit 8 GB Hauptspeicher – nicht möglich ist. Der Speicherbedarf übersteigt 10 GB, so daß der Vorgang abgebrochen werden muß. Dies kann jedoch als ein weiteres Beispiel gewertet werden, das zeigt, wie notwendig eine Verbesserung in diesem Bereich ist.

Alternativ zum Laden des Gesamtmodells werden die einzelnen Hauptbaugruppen nacheinander geladen, in bezug auf ihren Ressourcenbedarf analysiert und in separierter Form abgespeichert. Informationen, die ausschließlich am Einstiegspunkt der

Gesamtbaugruppe verfügbar sind, werden auf Dateiebene extrahiert. Als Ergebnis dieses Schritts liegt schließlich das vollständige Modell als Hierarchie von Subclusterdateien auf dem Dateisystem vor.

Laden des Baugruppengerüsts

Mangels Integration in ein PDM-System werden alle Dateien, die zum Laden des Gerüsts der Baugruppe benötigt werden – dies sind alle Dateien von Struktur-Subclustern – auf dem Dateisystem manuell zusammengestellt. Da sich die Cluster-Hierarchie in den Referenzen zwischen den Dateien der Struktur-Subcluster widerspiegelt und die Auflösung von Referenzen auf weitere Subcluster während des Ladeprozesses optional ist, kann diese Struktur geladen und ausgewertet werden.

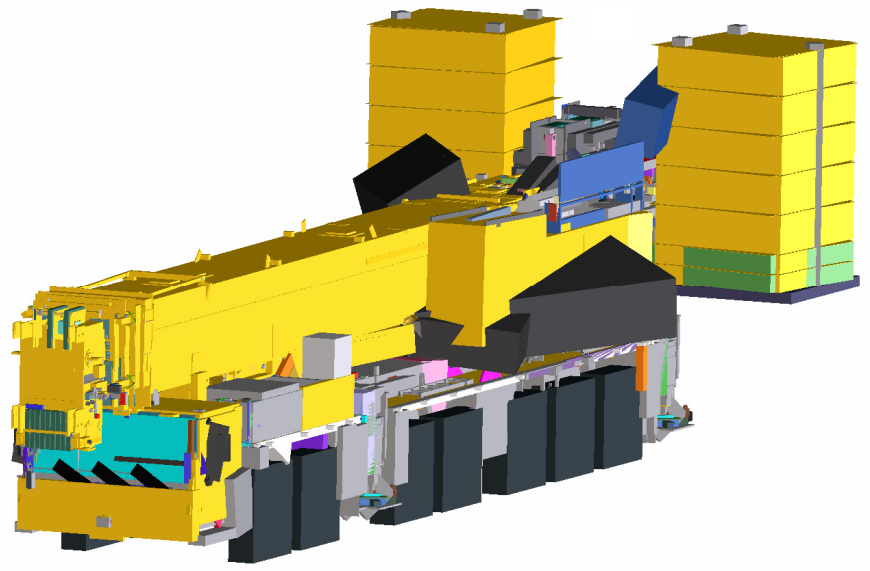


Abbildung 6.13: Quadermodell

Aufgrund der in der Struktur vorhandenen Boxinformation steht unmittelbar nach dem Laden ein Quadermodell der Baugruppe zur Verfügung. Als abstraktes, maschinell auswertbares Modell kann es verwendet werden, um bereits in dieser Minimalkonfiguration der Baugruppe zumindest näherungsweise Aussagen über räumliche Beziehungen von Teilen zu treffen. In **Abbildung 6.13** wird das – für das menschliche Auge nicht besonders hilfreiche und daher normalerweise nicht sichtbare – Modell durch Ergänzung graphischer Elemente visualisiert. In dieser Darstellung werden die minimalumschließenden Quader genauso positioniert wie die zugehöri-

gen Teile. Die Quader sind hierbei im allgemeinen kleiner als bei der Verwendung achsenparalleler Boxen. Die Auswahl kann dadurch genauer erfolgen, die Berechnung ist jedoch aufwendiger.

Laden des graphischen Modells

Das graphische Modell der Gesamtbaugruppe besteht aus der Summe aller strukturdefinierenden sowie graphischen Subcluster des Modells. Zum Aufbau von dessen Datenbasis werden lediglich die Dateien der graphischen Subcluster zur Auswahl der Struktur-Dateien aus dem vorhergehenden Schritt hinzugefügt – was im realen Anwendungsfall Aufgabe eines PDM-Systems ist. Das Ergebnis des Ladevorgangs entspricht im wesentlichen den von anderen Systemen angebotenen Lightweight-Modellen und stellt damit die Praxistauglichkeit des Konzepts prinzipiell unter Beweis. Abbildung 6.11 ist auf diese Weise entstanden.

Aufbau und Laden einer Konstruktionsumgebung

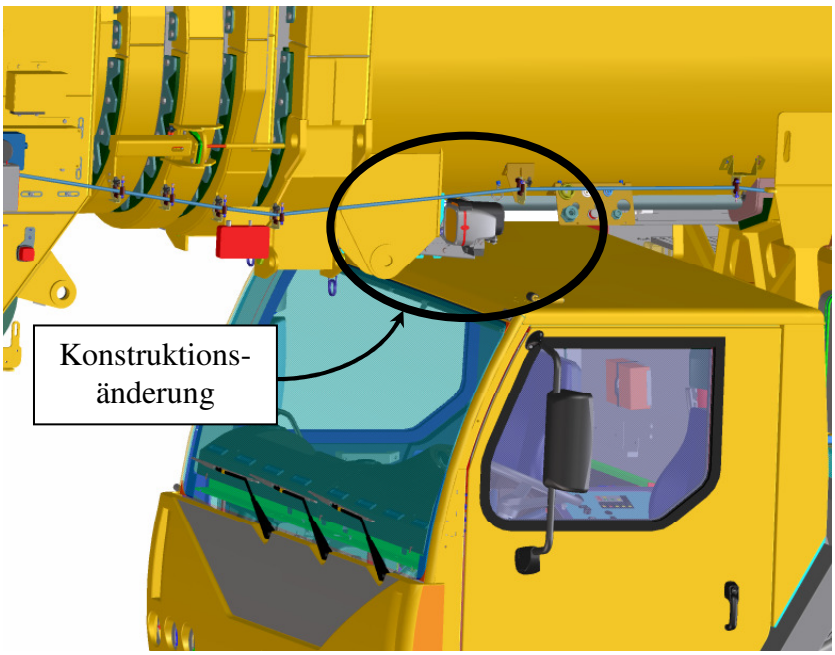


Abbildung 6.14: Ausschnitt des Gesamtmodells für eine Konstruktionsänderung

Als Arbeitshypothese für diesen Schritt wird angenommen, daß die in **Abbildung 6.14** markierte Beleuchtungseinheit, die zu den Anbauteilen des Auslegers zählt, konstruktiv verändert werden soll. Beim Aufbau einer geeigneten Konstruktionsumgebung sind dabei verschiedene Aspekte zu berücksichtigen. Zum einen soll der

Umfang der zu ladenden Daten auf ein Minimum beschränkt, zum anderen müssen alle für die Veränderung erforderlichen Daten identifiziert werden. Neben der direkt von der Veränderung betroffenen Baugruppe sind dies hauptsächlich Teile in der Umgebung, die die Einordnung der Arbeit in den Kontext des Gesamtmodells ermöglichen.

Wichtig für den Aufbau der Konstruktionsumgebung ist auch die Bedingung, daß die Veränderung nicht zu Kollisionen mit anderen Teilen führen darf – dafür in Frage kommende Teile sollen mithin in der Umgebung zur Verfügung stehen. Im betrachteten Beispiel betrifft dies insbesondere das Fahrerhaus, das jedoch zu einer anderen Hauptbaugruppe gehört.

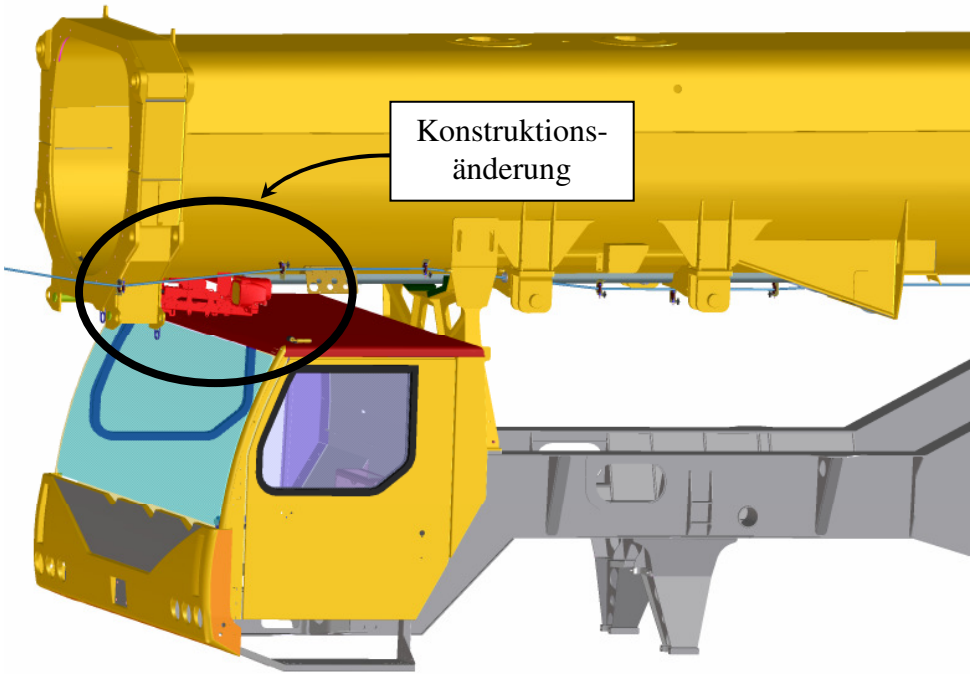


Abbildung 6.15: Konstruktionsumgebung zur Durchführung der Änderung

Ausgehend vom Gerüst des Gesamtmodells und der zu verändernden Baugruppe, deren Position im Strukturbaum als bekannt angenommen werden kann, werden sukzessive weitere Teile in der Umgebung aufgrund räumlicher Beziehungen identifiziert. Da das Nachladen von Subcluster-Dateien in der verwendeten Testversion noch nicht verfügbar ist, werden die meisten der benötigten Teile von Hand auf Basis des Quadermodells bestimmt und auf Dateiebene ergänzt. Auf diese Weise entsteht die in **Abbildung 6.15** gezeigte Konstruktionsumgebung. Die rot eingefärbten Teile –

die Baugruppe selbst sowie das Dach des Fahrerhauses – markieren den Kern der Konstruktionsumgebung und werden daher in vollständiger Repräsentation geladen, das heißt, mit geometrischem Modell. Alle anderen Teile liegen lediglich in graphischer Repräsentation vor.

Zur besseren Veranschaulichung des Ergebnisses und zur Verringerung des manuellen Aufwands zum Aufbau der Umgebung wird auf eine weitere Optimierung der Zusammenstellung der Teile verzichtet, die mit Rechnerunterstützung noch erreicht werden kann. Ebenso unterbleibt die mögliche Ergänzung der ausgewählten Teile durch Applikationsdaten. Schließlich umfaßt die dargestellte Konstruktionsumgebung 140 vollständig geladene Teile im Kernbereich sowie weitere 425 graphische Teile in der Umgebung.

Zusammenfassung der Meßergebnisse

Die Graphik in **Abbildung 6.16** gibt einen Überblick zu den Meßwerten der einzelnen Varianten. Bestimmt wurde jeweils – soweit möglich – der Speicherbedarf für die Modelldaten beim Laden des Gesamtmodells sowie die für den Ladevorgang benötigte CPU-Zeit. Zum Vergleich wurden diese Größen, außer beim Laden der Umgebung, ebenfalls für die einzeln verfügbaren Hauptbaugruppen ermittelt.

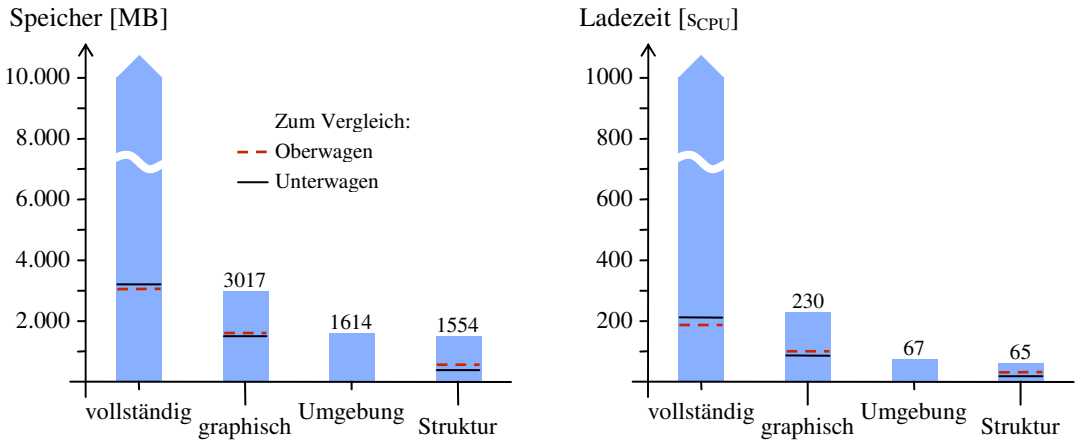


Abbildung 6.16: Speicherbedarf und Ladezeit des Gesamtmodells

Die Ergebnisse bestätigen eine deutliche, mit zunehmender Ausdünnung des Modells einhergehende Verringerung des Ressourcenbedarfs, die in dieser Form entsprechend der Verteilung der Datenanteile gemäß Abbildung 6.12 in etwa erwartet werden

konnte. Wichtiger ist jedoch die Beobachtung, daß die Kennzahlen beim Laden der Konstruktionsumgebung nur marginal über denjenigen des Baugruppengerüsts liegen, die als Minimalanforderung angenommen werden können.

Während bereits das Laden der Gesamtbaugruppe selbst auf einem für heutige Verhältnisse sehr gut ausgestatteten Rechner nicht zum Erfolg führt, ist die Bearbeitung einer Konstruktionsaufgabe in einer situationsgerecht zusammengestellten Umgebung noch auf einem Mittelklasse-PC möglich. Dabei muß weder auf den Kontext der gesamten Baugruppe noch auf Assoziativität von Applikationsdaten verzichtet werden, die jederzeit bei Bedarf zielgerichtet ergänzt werden können.

Das Potential dieses Konzepts ist damit jedoch noch nicht ausgeschöpft. Einerseits macht die Grundlast der jedesmal zu ladenden Strukturdaten prozentual nur einen sehr geringen Anteil am gesamten Datenaufkommen aus, der überdies mit zunehmender Größe der Baugruppen näherungsweise lediglich linear steigt. Da die Konstruktionsumgebung jedoch lokal und somit in ihrem Umfang mehr oder weniger konstant ist, profitiert dieser Ansatz überproportional von einer wachsenden Baugruppengröße.

Andererseits besteht die Möglichkeit, die Strukturdaten noch konsequenter auszudünnen, das heißt, weitere Dateneinheiten zu separieren, als das mit dem getesteten System verfolgt wurde. Ebenso bietet die Unterstützung beim Aufbau der Konstruktionsumgebung weitere Verbesserungsmöglichkeiten mit dem Ziel, die Anzahl der benötigten Teile weiter zu reduzieren.

6.6.3 Fazit

Die Entwicklung des Konzepts zur Datenseparierung sowie der dazu erforderlichen Datenstrukturen schlägt eine Brücke zwischen der Integration technologischer Produktdaten in ein CAD-Modell auf der einen und einem selektiven Zugriff auf diese Daten auf der anderen Seite. Damit wird eine Anforderung an die Handhabung großer Baugruppen erfüllt, die bislang keine ausreichende Berücksichtigung in CAD-Systemen findet, wie aus dem Fazit der Gegenüberstellung von bestehender Funktionalität und Anforderungen in Kapitel 4.4 hervorgeht.

Bereits existierende Konzepte gehen entweder von einer zu groben Unterteilung der Daten aus – in der Regel werden Applikationsdaten zusammen mit den Komponenten einer Baugruppe verwaltet –, was einen gezielten Zugriff auf einzelne Dateneinheiten erschwert, oder beruhen auf der Verwaltung der Applikationsdaten außerhalb des CAD-Modells. Dieser Ansatz wird jedoch mit Einschränkungen bei der Assoziativität der Daten erkauft.

Durch die feinere Untergliederung des Datenmodells und die damit verbundene Erhöhung der Granularität der Dateistruktur ermöglicht das Konzept der Datenseparierung hingegen *sowohl* eine hohe Integration der Applikationsdaten *als auch* einen gezielten Zugriff darauf. Dadurch wird eine stärkere Verzahnung von CAD-Daten und technologischen Produktdaten erreicht, die eine wesentliche Grundlage für Concurrent Engineering darstellt und dem iterativen Charakter der Konstruktionsprozesse Rechnung trägt.

Durch die Verwendung des Konzepts und der zugrundeliegenden Datenstrukturen als Basis für den Einsatz von Lightweight-Teilen in OneSpace Designer Modeling konnte die Realisierbarkeit sowie die Produktfähigkeit der Datenseparierung und insbesondere des Subcluster-Konzepts nachgewiesen werden. Die generische Anlage der Datenstrukturen erleichtert darüber hinaus die Umsetzung für nachfolgende Applikationen. Weitere im Rahmen dieser Arbeit beschriebene und auf der Separierung von Daten basierende Konzepte leisten einen zusätzlichen Beitrag zum Erreichen des angestrebten Ziels der benutzergerechten Handhabung großer Baugruppen in 3D-CAD-Systemen.

Die Möglichkeit, Dateneinheiten aufgrund der höheren Granularität aufgabenspezifisch zu kombinieren, wird beispielsweise zur Optimierung der Konstruktionsumgebung ausgenutzt. Im Kontext des Gesamtmodells, das lediglich als Baugruppengerüst in das CAD-System geladen wird, erfolgt die Definition einer graphischen Umgebung zur besseren Einordnung der Tätigkeit. Darin eingebettet liegt ein kleiner Kernbereich, der alle für die Bearbeitung notwendigen Daten enthält. Dies können ebenso Teile des geometrischen Modells wie auch Dateneinheiten einzelner Applikationen sein, die für die Bearbeitung selbst oder zur Wahrung der Assoziativität

erforderlich sind. Das große Potential, das diese auf Datenseparierung beruhende Vorgehensweise zur Verringerung des Ressourcenbedarfs bietet, wird an dem Testbeispiel deutlich, das im vorhergehenden Abschnitt analysiert wurde.

Der Aufbau einer Konstruktionsumgebung kann grundsätzlich baugruppenübergreifend erfolgen. Zusätzlich wird die Auswahl der benötigten Daten durch die Berücksichtigung von räumlichen Beziehungen unterstützt, die bereits im Gerüst der Baugruppe verfügbar sind. Der Anwender kann somit seine Umgebung jederzeit um benachbarte Teile ergänzen, ohne die Struktur angrenzender Baugruppen kennen zu müssen.

Bereits durch die Fokussierung auf den für die Bearbeitung wesentlichen Ausschnitt des Modells kann die Ladezeit signifikant gesenkt werden. Die Messungen anhand des Testbeispiels im vorhergehenden Abschnitt belegen dies. Darüber hinaus besteht die Möglichkeit, die aufgrund der Datenseparierung erhöhte Granularität der Dateistruktur zur feineren Untergliederung des Ladeprozesses einzusetzen. In der Folge können Teilergebnisse des Ladens schneller verfügbar gemacht werden – die Wartezeit für den Anwender verkürzt sich entsprechend.

Schließlich ermöglicht die feinere Dateistruktur eine gezieltere Vergabe von Zugriffsrechten. Dadurch wird in dieser Hinsicht derselbe Zustand erreicht wie bei einer externen Verwaltung von Applikationsdaten. Der Schutz vor unberechtigtem Zugriff kann bereits auf dieser Ebene realisiert werden. Durch die gleichzeitige Integration der Daten in das CAD-Modell entfallen jedoch die Beschränkungen bezüglich der Assoziativität.

7 Zusammenfassung und Ausblick

Nicht zuletzt bedingt durch äußere Einflüsse wie eine zunehmende Globalisierung hat sich der Verlauf der Konstruktion zu einem komplexen, interdisziplinären Prozeß entwickelt. Der ohnehin stark iterative Charakter dieses Prozesses und die enge Verzahnung mit nachgeschalteten Aufgaben wie Berechnung und Analyse werden durch diese Entwicklung noch betont. Aufgrund dessen ist auch eine zunehmende Integration von Simulations- und Berechnungswerkzeugen in die CAD-Systeme zu beobachten.

Für das gemeinsame Datenmodell, das sowohl Konstruktionsdaten als auch technologische Produktdaten umfaßt, bedeutet dies einerseits höhere Anforderungen an Integrität und Assoziativität. Andererseits ist ein gezielter und kontrollierter Zugriff auf einzelne Einheiten erforderlich, da unter diesem Dach die Daten verschiedener Anwender und Applikationen zusammenkommen.

Vor diesem Hintergrund beschäftigt sich die vorliegende Arbeit mit der Entwicklung von Konzepten und Datenstrukturen, die bei der internen Datenverwaltung eines 3D-CAD-Systems zum Tragen kommen. Dabei wird das Ziel verfolgt, eine benutzergerechte Handhabung großer Baugruppen zu ermöglichen, die heute aufgrund des hohen Datenvolumens und beschränkter Ressourcen nicht immer gewährleistet ist.

Zunächst werden Anforderungen für die Handhabung großer Baugruppen anhand praktischer Arbeitsweisen und theoretischer Betrachtungen zur Konstruktionsmethodik analysiert. Zur Ermittlung des aktuellen Standes der Technik werden anschließend marktbestimmende 3D-CAD-Systeme auf verfügbare Funktionalität und Konzepte hin untersucht, die zur Handhabung großer Baugruppen einen Beitrag leisten können.

Neben Optimierungspotentialen bei Auswahlmethoden und Ladezeiten deckt die Gegenüberstellung dieser Ergebnisse Defizite in der Art der Integration von technologischen Produktdaten auf. Diese ist heute derart realisiert, daß die Daten *entweder* über eine hohe Assoziativität verfügen *oder* über Möglichkeiten zum selektiven Zugriff. Bei großen Baugruppen ist jedoch der Aufbau von Modellausschnitten, die optimal auf die jeweiligen Aufgaben zugeschnitten sind, zum effizienten Umgang mit Ressourcen sinnvoll, teilweise sogar unabdingbar. Eine Voraussetzung hierfür ist unterdessen gerade die Verfügbarkeit von Assoziativität *und* selektivem Zugriff.

Im folgenden wird daher das Konzept der Datenseparierung auf der Basis des im CAD-System OneSpace Designer Modeling verwendeten Datenmodells eingeführt. Bislang war eine Komponente der Baugruppenstruktur die kleinste separat speicherbare Einheit. Diese wird nun weiter unterteilt in eigenständige Untereinheiten – sogenannte Subcluster – für den strukturdefinierenden Anteil der Komponente, das geometrische Modell, das graphische Modell sowie jeweils eigene Untereinheiten für applikationsspezifische Daten. Dies sind technologische Produktdaten, die nach ihrer Zugehörigkeit zu einer Applikation gruppiert werden können.

Das bislang verwendete Konzept zur Gruppierung aller Informationen (Entities), die in einer Komponente zusammengefaßt sind, wird zum Subcluster-Konzept weiterentwickelt, die vorhandenen Datenstrukturen werden entsprechend erweitert und durch zusätzliche Klassen ergänzt. Dadurch erben die Subcluster insbesondere die Fähigkeit, ihre Daten in eigenständige Dateien zu speichern, womit die Unterteilung der Komponenten zu einer entsprechend höheren Granularität der Dateistruktur führt.

Den Struktur-Subclustern kommt eine Sonderstellung zu. Mit ihrer Hilfe kann ein sehr schlankes Gerüst – quasi das Rückgrat – der Baugruppe aufgebaut werden, das bei Bedarf um beliebige Kombinationen weiterer Subcluster ergänzt wird. Zusammen mit der höheren Granularität der Dateistruktur ist damit der Brückenschlag zwischen einer hohen Integration und selektivem Zugriff gelungen. Applikationsdaten können in das CAD-Modell integriert werden und verfügen bei Bedarf über die volle Assoziativität. Als eigenständige Dateien, die optional geladen werden können, profitieren sie ebenso von der Möglichkeit des selektiven Zugriffs.

Basierend auf dieser Technologie werden weitere Konzepte entwickelt, die einen direkten Beitrag zur benutzergerechten Handhabung großer Baugruppen leisten. Das Konzept der Datenseparierung wird eingesetzt, um Konstruktionsumgebungen aufzubauen, die optimal auf die jeweiligen Aufgaben zugeschnitten sind. Ausgehend vom Gerüst der Baugruppe, welches das Arbeiten im Kontext des Gesamtmodells erlaubt, wird eine graphische Umgebung im Nahbereich der Änderung ergänzt. In diese eingebettet liegt der Kern, der alle Teile vollständig geladen enthält, die für die Bearbeitung notwendig sind. Ebenso ermöglicht das Konzept verbesserte Ladealgorithmen, mit denen die Wartezeit während des Ladevorgangs verkürzt werden kann. Mit der Ergänzung der Struktur-Subcluster durch Box-Informationen – das heißt, minimal umschließende Quader für Teile – kann bereits auf der Basis des Baugruppengerüsts Funktionalität zur Nachbarschaftsselektion implementiert werden.

Die Realisierbarkeit und Produkttauglichkeit der Datenseparierung und des Subcluster-Konzepts wurde mit der Implementierung der Lightweight-Funktionalität in OneSpace Designer Modeling 14.0 unter Beweis gestellt. Diese basiert auf der beschriebenen Technologie und verwendet Struktur- und Graphik-Subcluster, während das geometrische Modell nicht geladen wird. Darüber hinaus wurde die Separierung von Applikationsdaten prototypisch implementiert. Umfangreiche Tests mit realen Modellen haben gezeigt, daß der Ressourcenbedarf einer Konstruktionsumgebung, die für eine fiktive Änderung an einem Modell aufgebaut wurde, nur einen Bruchteil des Bedarfs des Gesamtmodells ausmacht. Dadurch konnte die Änderung bereits an einem Mittelklasse-PC durchgeführt werden, obwohl die Gesamtbaugruppe wegen eines zu hohen Speicherbedarfs selbst auf einer 64-Bit-Maschine mit 8 GB Hauptspeicher nicht geladen werden konnte.

Insgesamt wurde mit der Datenseparierung und dem darunterliegenden Subcluster-Konzept eine Basistechnologie entwickelt, die in verschiedenen Bereichen eines CAD-Systems Beiträge zur benutzergerechten Handhabung großer Baugruppen leisten kann. Als Basis von Lightweight-Modellen bzw. mit der Verringerung von Speicherbedarf und Ladezeiten, dem Aufbau aufgabenspezifischer Konstruktionsumgebungen, der Auswertung räumlicher Beziehungen sowie dem Schutz von Konstruktionsdetails durch die Möglichkeit der Vergabe von Zugriffsrechten auf

Subcluster-Ebene wurden im Laufe dieser Arbeit bereits einige diskutiert. Weitere Anwendungskonzepte können auf dieser Basis entwickelt werden. Concurrent Engineering auf der Ebene von Applikationsdaten sei hier nur exemplarisch genannt.

Die Erhaltung der Assoziativität für Applikationsdaten wird durch das Konzept der Datenseparierung unterstützt, sofern diese während einer Operation geladen und damit im Speicher verfügbar sind. In Kapitel 5.2.6 wird eine Möglichkeit angedacht, die es erlaubt, Assoziativität auch dann zu erhalten, wenn die Applikationsdaten selbst nicht verfügbar sind. Die abschließende Klärung dieser Frage bedarf jedoch einer weiterreichenden Betrachtung, die im Rahmen dieser Arbeit nicht erfolgen konnte, und bietet damit Gelegenheit zur weiteren Forschung.

8 Literaturverzeichnis

[3DS] *www.3ds.com*. Dassault Systemes, Suresnes, Frankreich

[Abe97] Abeln, O.: *The "CAD-Reference-Model" An Architectural Approach to Future CAD-Systems*. In Roller, D.; Brunet, P. (ed.): *CAD Systems Development – Tools and Methods*. Springer-Verlag, Berlin, 1997

[Ber02] Bergers, D.: *Produktentwicklung*. Vorlesungsskript, Institut für Produkt Engineering, Gerhard-Mercator-Universität Duisburg, 2002

[BrKu95] Brod, C.; Kublin, M.: *Providing CAD Object Management Services through a Base Class Library*. Hewlett-Packard Journal, Vol. 46, Num. 5, Palo Alto, 1995

[CGM04] *Computer-Grafik-Markt 2004/2005*. Dressler Verlag, Heidelberg, 2004

[CoC] *www.cocreate.com*. CoCreate Software GmbH & Co. KG, Sindelfingen

[CoC05] *OneSpace Designer Modeling: Best practices for large assembly handling*. CoCreate Software GmbH & Co. KG, Sindelfingen, 2005

[Con05] Conrad, K.-J.: *Grundlagen der Konstruktionslehre*. Hanser Verlag, München, 2005

[EAR03] *SolidWorks 2004: Making the Difficult Stuff Easier for All Users*. Engineering Automation Report, Vol. 12, Issue 8, Bellingham, WA, USA, 2003

[Enc88] Encarnacao, J.: *Computer Graphics*. Oldenbourg Verlag, München, 1988

[Grä89] Grätz, J.-F.: *Handbuch der 3D-CAD-Technik*. Siemens Aktiengesellschaft, Berlin, 1989

[Hen01] Hennig, H.: *Konzept für den Aufbau von Bauteil- und Baugruppeninformationen für komplexe Produkte auf Basis eines funktionsorientierten Strukturmodells*. Dissertation, Lehrstuhl für Konstruktionstechnik, Universität Erlangen-Nürnberg, 2001

[Hor97] Horn, Th,: *Entwicklung eines Verfahrens zur interaktiven Definition und Erkennung benutzerdefinierter Features*. Dissertation, Fachbereich Maschinenwesen, Universität Gesamthochschule Essen, 1997

[HrRu02] Hruschka, P.; Rupp C.: *Agile Softwareentwicklung für Embedded Real-Time Systems mit der UML*. Hanser Verlag, München, 2002

[IBM] *www.ibm.com*. International Business Machines Corporation, Armonk, NY, USA

[IDC04a] IDC White Paper: *Intel´s Enterprise Processor Plans: Positioning the Xeon Processor and the Itanium Processor*. IDC, Framingham, MA, USA, 2004

[IDC04b] IDC White Paper: *The AMD Opteron Processor and the Transistion to Industry-Standard 64-Bit Workstation Computing*. IDC, Framingham, MA, USA, 2004

[Jar97] Jared, G.: *Product Modelling for Design for Manufacture and Design for Assembly*. In Roller, D.; Brunet, P. (ed.): *CAD Systems Development – Tools and Methods*. Springer-Verlag, Berlin, 1997

[Kal97] Kaltenecker, O.: *Evaluierung des Einsatzes von ODBMS als Basis von CAD-Systemen am Beispiel von CoCreate / SolidDesigner*. Diplomarbeit Nr. 1465, Institut für Informatik, Universität Stuttgart, 1997

[Kni04] Knieps, M.: *Entwicklung von Konstruktionsrichtlinien für die Handhabung von 3D-CAD-Systemen zur Generierung komplexer Maschinenbaugruppen*. Dissertation, Fachbereich Maschinenwesen, Universität Duisburg-Essen, 2004

[Kro95] Krottmaier, J.: *Leitfaden Simultaneous Engineering*. Springer-Verlag, Berlin, 1995

[Lie05] Liese, H.: *Wissensbasiertes CAD – Erweiterte Möglichkeiten und Standardisierungspotentiale*. ProduktDaten Journal, Nr. 2/2005, ProSTEP iViP e.V., Darmstadt

[Lob02] Lobeck, F.: *Betriebsdatenverarbeitung I – PDM/EDM*. Vorlesungsskript, Institut für Ingenieurinformatik, Universität Essen, 2002

[Lob04] Lobeck, F.: *Konstruktionselemente III (CAD) / Einführung in Catia V5*. Vorlesungsskript, Institut für Ingenieurinformatik, Universität Essen, 2004

[PaBe03] Pahl, G.; Beitz, W.: *Konstruktionslehre*. 5. / 6. Auflage, Springer-Verlag, Berlin, 2003 / 2005

[Pah90] Pahl, G.: *Konstruieren mit 3D-CAD-Systemen*. Springer-Verlag, Berlin 1990

[PTC] *www.ptc.com*. Parametric Technology Corporation, Needham, MA, USA

[Rem05] Rembold, W.: *Einstieg in Catia V5*. Hanser Verlag, München, 2005

[RuUh00] Rues, K.; Uhlich, D.: *Pro/ENGINEER – Strategien zum Umgang mit großen Baugruppen*. Hanser Verlag, München, 2000

[SiDi01] Siering, P.; Diedrich, O.: *18 Milliarden Gigabyte*. c't 13/2001, Heise Verlag, Hannover, 2001

[Schi02] Schichtel, M.: *Produktdatenmodellierung in der Praxis*. Hanser Verlag, München, 2002

[Schn99] Schneider, Th.: *SolidDesigner's Parcel and Filing Concepts*. Präsentation (CoCreate intern), CoCreate Software GmbH & Co. KG, Sindelfingen, 1999

[Schn01a] Schneider, Th.: *OneSpace Advise Large Model Assembly*. Präsentation (CoCreate intern), CoCreate Software GmbH & Co. KG, Sindelfingen, 2001

[Schn01b] Schneider, Th.: *Open Reference Handling*. Interne Projektdokumentation, CoCreate Software GmbH & Co. KG, Sindelfingen, 2001

[Schn02] Schneider, Th.: *Mixed Fidelity*. Interne Projektdokumentation, CoCreate Software GmbH & Co. KG, Sindelfingen, 2002

[Schn05] Schneider, Th.: *Large Model – Directions*. Interne Projektdokumentation, CoCreate Software GmbH & Co. KG, Sindelfingen, 2005

[SpKr84] Spur, G.; Krause, F.-L.: *CAD-Technik*. Hanser Verlag, München, 1984

[Str00] Stracke, H.-J.: *Ingenieurinformatik II / CAD I*. Vorlesungsskript, Institut für Ingenieurinformatik, Universität Essen, 2000

[SWk] *www.solidworks.com*. SolidWorks Corporation, Concord, Ma, USA

[SWk02a] *How to increase assembly performance*. SolidWorks Express Tech Tip Sep. 02, SolidWorks Corporation, Concord, Ma, USA, 2002

[SWk02b] *How to plan assemblies for top-down designs*. SolidWorks Express Tech Tip Sep. 02, SolidWorks Corporation, Concord, Ma, USA, 2002

[UGS] *www.ugs.com*. Unigraphics Solutions Corporation, Plano, TX, USA

[UGS05a] *Assembly Set – Capabilities for assembly, design and evaluation*. Unigraphics Solutions Corporation, Plano, TX, USA, 2005

[UGS05b] *NX – Digital Engineering neu definiert*. Unigraphics Solutions Corporation, Plano, TX, USA, 2005

[VDI93] VDI-Richtlinie 2221: *Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte*. VDI-Verlag, Düsseldorf, 1993

[Web05] Weber, C.: *CPM/PDD – An Extended Theoretical Approach to Modelling Products and Product Development Processes*. In Bley, H.; Jansen, H. et al. (ed.): *Proceedings of the 2nd German-Israeli Symposium on Advances in Methods and Systems for Development of Products and Processes*. Fraunhofer-IRB-Verlag, Stuttgart, 2005

9 Anhang

Abbildungsverzeichnis

Datenblatt des CAD-Systems OneSpace Designer Modeling

Die Beschreibung der Produkteigenschaften des zur Realisierung der Konzepte verwendeten 3D-CAD-Systems OneSpace Designer Modeling zitiert im wesentlichen aus dem von CoCreate veröffentlichten Datenblatt dieses Systems, vgl. [CoC].

Produkteigenschaften

Dynamic Modeling

- Vollständige 3D Modellierungslösung für Teile und Baugruppen, Volumen-, Flächen- und Drahtmodelle
- 3D Konstruktionsänderungen direkt am Modell
- Hochentwickelte Baugruppenbearbeitung

Unterstützung für große Baugruppen

- Schnelle 3D Graphikmodelle
- Bis zu 128 GB verfügbar für große Baugruppen mit XP Professional x64 Edition
- Gemeinsame Teile und Baugruppen für Speicheroptimierung

Blechkonstruktion

- Erstellen und Ändern von Biegungen und Laschen inklusive Biege- und Knickentlastungen vollständig in 3D
- Animierte Biegeprozeßsimulation
- Automatisiertes Abwickeln in 3D und 2D

Machining

- Integriert und verfügbar während der Konstruktion, direkte Übergabe an Standard CAM-Systeme
- Echtzeitprüfung bei laufendem Konstruktionsprozeß
- Integriertes Design-for-Manufacturing

Surfacing

- Erstellen komplexer Oberflächen
- Helixerzeugung; Lofting und Sweeping
- Definition von 3D Kurven im Raum

Assoziative Zeichnungserstellung

- Automatisches, standardisiertes Ableiten von originalgetreuen 2D Zeichnungen aus dem 3D Modell
- 3D Konstruktionsänderungen werden assoziativ in den 2D Zeichnungen mitgeführt
- Datenübergabe in den Fertigungsprozeß über DXF, DWG, IGES und MI, dem nativen Format von OneSpace Designer Drafting

Drafting

- Modeling beinhaltet Drafting, die 2D Konstruktionslösung von CoCreate
- Schnelle und intuitive Erstellung von professionellen technischen 2D Zeichnungen
- Leistungsstarke Funktionen zur optimalen Anpassung auf den unternehmens-spezifischen Produktentwicklungsprozeß

Prozeßinformationen

- Automatische Kollisionsanalyse mit Bericht und Interaktion
- Zusätzliche, nicht-geometrische Informationen wie Notizen, Analyse-ergebnisse und 3D Bemaßungen
- Verwendbar in 2D Zeichnungen oder zur Anzeige in 3D Access

Datenaustausch

- Importierte Modelle sind wie native OneSpace Modelle weiter verwendbar
- Import und Export über STEP, IGES 3D, SAT und IDF
- Export in VRML und STL
- Funktionen zur Erzeugung und Anzeige reduzierter Graphikdaten mit eDrawings
- Automatische Integritätsprüfung und Reparatur