Das Beobachterkonzept zum Erhalt der Konsistenz in Datenmodellen

Andreas Laabs, Peter Jan Pahl

1. PROBLEMSTELLUNG

Die Aufgaben des Bauingenieurwesens sind dadurch geprägt, daß sowohl die Planung als auch die Ausführung von Bauwerken häufigen Änderungen unterliegen. Beschreibt man das Verhalten der Bauwerke und den Bauprozeß im Computer mit Modellen, so ändern sich Umfang und Struktur der Modelle als Folge der Änderung in Planung und Ausführung. Diesen Vorgang nennt man Dynamisierung des Modells. Die Dynamisierung führt zu Veränderungen und Inkonsistenzen in den Modellen der Anwendungen. Die Inkonsistenzen müssen erkannt und wahlweise automatisch oder manuell aufgehoben werden.

Beispielsweise werden Änderungen in einem Finite-Elemente-System erst vollständig vorgenommen, da der Lösungsaufwand für große Gleichungssystemen auch bei modernen Rechnern noch beträchtlich ist.. Die Inkonsistenzen werden beim Vorgang des Änderns vermerkt und anschließend auf manuelle Anforderung beseitigt. Ebenso können Inkonsistenzen in einem System aus der Sicht eines Ingenieurs auch auf Dauer erhalten bleiben. Beispielsweise ändert sich die Belastung in einem Finite Elemente System. Das sich daraus ergebende Verhalten liegt im Vergleich zur vorangegangenen Belastung auf der sicheren Seite. Die Inkonsistenz kann im System bestehen bleiben.

Die Modellierung der Aktualisierung und der Konsistenz in Datenmodellen kann auf das folgende Kernproblem reduziert werden. Zwischen den Objekten in den verschiedenen Modellen eines Systems bestehen diverse Abhängigkeiten. Diese können graphisch als Netz dargestellt werden, wobei die Objekte die Knoten und die Abhängigkeiten die gerichteten Kanten des Netzes sind. Für die Modellierung besteht nun das Problem darin, daß beim Entwurf einer konkreten Modellfunktionalität die Auswirkungen auf andere Objekte und Modelle nicht vollständig vorab festgelegt werden können. Auf diesen Fall beschränken sich auch die von der objektorientierten Modellierung bereitgestellten Modelliermethoden.

Die reagierenden Objekte und Modelle sind beispielsweise zum Zeitpunkt der Modellierung noch gar nicht bekannt und lassen sich mit ihrer Reaktion daher auch nicht festlegen. Oder es werden Abhängigkeiten erst durch den Benutzer auf der Benutzungsoberfläche dynamisch festgelegt. Für das agierende Objekt sind die Reaktionen anderer Objekte des selben Modells oder eines anderen Modells unbekannt. Im Normalfall sind Reaktionen auf eine Aktion hoch flexibel und aus der Sicht des reagierenden Objekts zu modellieren. Die statische Festlegung von Reaktionen kann nur dann modelliert werden, wenn die reagierenden Objekte eindeutig vom agierenden Objekt identifiziert werden können und eine Flexibilität in der Reaktion von der Problemstellung nicht gefordert ist.

2. BEOBACHTERKONZEPT

Ein leistungsfähiges Konzept, mit dem systematisch die Aktualisierung und die Konsistenz in

---

Dipl.-Ing. Andreas Laabs, Prof. Dr. Peter Jan Pahl, Institut für
Technische Universität

Datenmodellen sichergestellt werden kann, ist der Beobachtermechanismus. Der Beobachtermechanismus teilt Objekte in Beobachtungsziele und Beobachter ein, wobei diese Teilung der Objekte nicht zu disjunkten Mengen führt, d.h. eine Beobachtungsziel kann selbst wieder Beobachter sein. Ein Beobachtungsziel versendet Meldungen über seine internen Zustandsänderungen. Objekte, die auf eine bestimmte Zustandsänderung des Beobachtungsziels reagieren wollen, müssen als Beobachter des Beobachtungszieles für diese Meldung registriert sein. Registration bedeutet, daß eine Objektmethode des reagierenden Objekts vereinbart worden ist, die aufzurufen ist, wenn die Änderung gemeldet wird. Von entscheidender Bedeutung ist, daß die Reaktion auf eine Aktion ausschließlich durch den Beobachter selbst und nicht das Beobachtungsziel festgelegt wird.

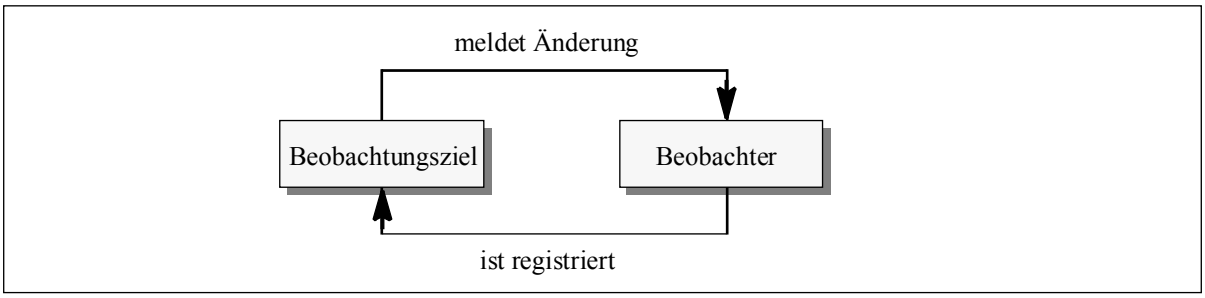

Abb.1 : Beobachterkonzept

Um die abstrakte Beschreibung von Objekten unabhängig vom Beobachtungsmechanismus zu lassen, ist zwischen Beobachtungsziel und Beobachtern eine externe Verwaltung aufzubauen. Die Verwaltung stellt die Methodenschnittstelle zur Registration von Beobachtungszielen und Beobachtern sowie zum Senden von Meldungen zur Verfügung. Tritt eine Änderung bei einem Beobachtungsziel ein, so meldet das Beobachtungsziel die Änderung der Verwaltung. Die Verwaltung meldet das Ereignis den Beobachtern durch Aufruf der hinterlegten Funktion.

Der hier vorgestellte Beobachtermechanismus ist durch die folgenden drei fundamentalen Eigenschaften charakterisiert :

- Für jedes Objekt braucht erst zur Laufzeit entschieden werden, ob es ein Beobachtungsziel ist. Die Registration als potentielles Beobachtungsziel ist zu jedem Zeitpunkt der Existenz eines Objektes änderbar.
- Zur Beobachtung wird eine externe Verwaltung installiert. Die externe Verwaltung ist für alle Modelle identifizierbar. Damit bekommen Objekte eines Modells A die Möglichkeit, Änderungen von Objekten in einem Modell B zu beobachten.
- Die Transformation von Aktion in Reaktion liegt in der Verwaltung. Dazu muß die Verwaltung aus der Menge aller Beobachter die entsprechende Teilmenge der Objekte selektieren.

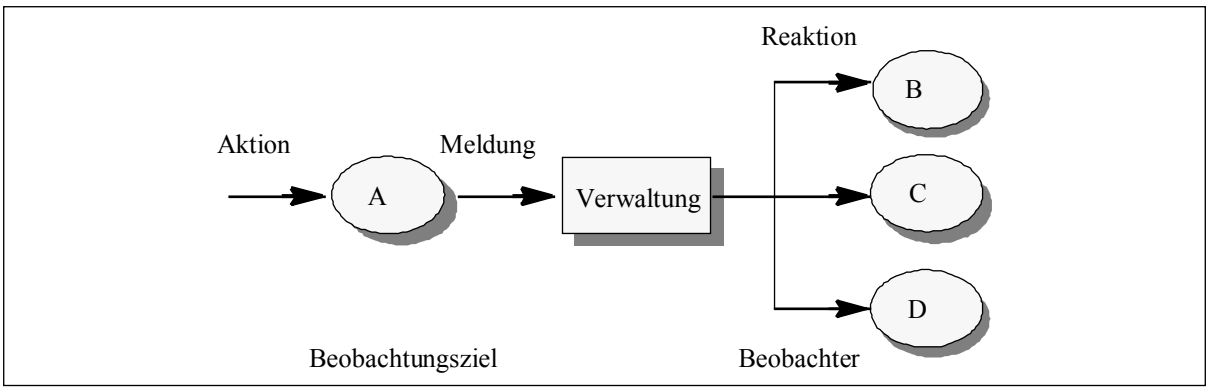

Abb.2 : Beobachtungsmechanismus mit externer Verwaltung

In Abb.2 ist der Beobachtungsmechanismus mit externer Verwaltung schematisch gezeigt. Das Objekt A ist potentielles Beobachtungsziel. Die Objekte B, C und D sind die Beobachter von A. Zwischen Beobachter und Beobachtungsziel steht die externe Verwaltung. Das Beobachtungsziel A registriert sich bei der Verwaltung selbst. Die Objekte B, C und D registrieren sich anschließend als Beobachter des Objektes A. Tritt eine Zustandsänderung des Beobachtungszieles A ein, so meldet A dies der Verwaltung. Die Verwaltung informiert alle Beobachter mit ihrer hinterlegten Methode.

Zur Darstellung der Meldungen und Beobachterbeziehungen kann eine eindeutige graphische Notation entwickelt werden. Die Beobachtungsbeziehung wird durch einen gerichteten Pfeil vom Beobachter, Klasse A in Abb.3, zum Beobachtungsziel, Klasse B in Abb.3, notiert. Welche Meldung des Beobachtungszieles beobachtet wird, wird an den Pfeil annotiert, hier CVkSelectObject. Die Menge der Meldungen, die ein Beobachtungsziel senden kann, wird mit einem Fähnchen an das Beobachtungsziel angehängt.

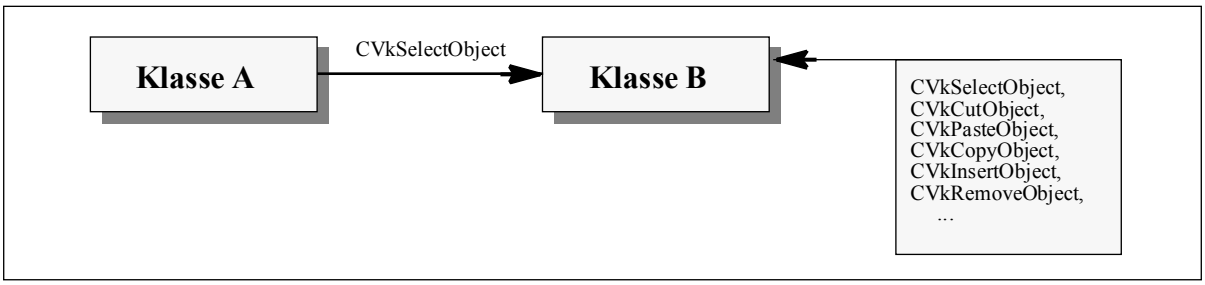

Abb.3 : Graphische Notation der Beobachterbeziehung

3. REALISIERUNG

Der Beobachtungsmechanismus ist durch einen Dienst realisiert worden, der Objekte und Methoden zur Verfügung stellt. Der Beobachtungs-Dienst wird am Anfang seines Einsatzes geöffnet und am Ende seines Einsatzes geschlossen. Eine externe Verwaltung wird durch einen Beobachtungsmanager repräsentiert. Ein Beobachtungsmanager wird durch eine Menge von Beobachtungszielen und eine Menge von geordneten Paaren definiert. Die geordneten Paare beschreiben die geprägten Beobachtungsbeziehungen. Das erste Element im geordneten Paar ist die Identität des Beobachtungszieles. Das zweite Element des geordneten Paares ist ein 4-Tupel, das aus der Identität des Beobachters, dem Namen der beobachteten Meldung, der Identität einer Methode des Beobachters und einer Adresse aus dem Adressbereich des Beobachters gebildet wird. Beobachtungsziel und Beobachter müssen nicht Objekte des selben Modells sein. Eine Beobachtungsbeziehung wird in zwei Schritten aufgebaut :

- Das Beobachtungsziel wird bei einem Beobachtungsmanager angemeldet. Die Identität wird in die Menge der potentiellen Beobachtungsziele eingetragen. Das Beobachtungsziel ist anschließend für andere Objekte beobachtbar.
- Der Beobachter meldet sich für ein bestimmtes Beobachtungsziel und eine bestimmte Meldung beim Beobachtungsmanager an, indem der Beobachter eine Methode im Beobachtungsmanager für diese Meldung hinterlegt. Das geordnete Paar, das die Beobachtungsbeziehung beschreibt wird erzeugt und in die Menge der Beobachtungsbeziehungen eingetragen.

Der Abbau einer Beobachtungsbeziehung kann durch das Beobachtungsziel oder den Beobachter erfolgen. Trägt sich ein Beobachtungsziel aus dem Manager selbst aus, so werden die Beobachter nicht informiert. Tritt ein Ereignis für ein Beobachtungsziel ein, so meldet das Beobachtungsziel dies dem Beobachtungsmanager. Der Beobachtungsmanager bestimmt alle Beobachter, die sich für diese Meldung interessieren und ruft jeden Beobachter mit seiner hinterlegten Methode auf. Indem die Beobachter selbst wieder Beobachtungsziele sind, werden Aktionsketten aufgebaut. Jeder Beobachter wird im Rahmen einer Aktionskette mit seiner Methode nur einmal aufgerufen. Damit sind Zyklen unterbunden. Die Persistenz von Funktionszeigern kann durch ein Objektbanksystem oder ein eigenes Konzept sichergestellt werden.

Die Realisierung des Beobachtungsmechanismus im Rechner benötigt nur einen minimalen Funktionssatz. Diese Methoden sind in der folgenden Tabelle zusammengestellt.

Dienstfunktionen	
RmOpenObserver	Dienst öffnen
RmCloseObserver	Dienst schließen
Beobachterfunktionen	
RmCreateObserver	Observer erzeugen
RmDestroyObserver	Observer entfernen
RmManageObject	Beobachtungsziel erzeugen
RmUnmanageObject	Beobachtungsziel entfernen
RmObserveObject	Beobachtung beginnen
RmNoObserveObject	Beobachtung einstellen
RmCallObserver	Beobachter rufen

4. ANWENDUNG

Nachfolgend wird ein einfaches Beispiel für die Anwendung des entwickelten Beobachtungsmechanismus gezeigt. Das Beispiel ist einem Modell zur editierbaren Visualisierung dreidimensionaler Körper, das im Rahmen des von der Deutschen Forschungsgemeinschaft geförderten Forschungsvorhabens ”Objektorientierte Analyse und Visualisierung physikalischer Zustände dreidimensionaler Körper” entwickelt worden ist, entnommen. Das Beispiel wird inhaltlich nur soweit dargestellt, daß das Beobachtungskonzept in seiner Anwendung und Bedeutung für die Konsistenzsicherung nachvollziehbar wird. Dazu wird zunächst die Aufgabe formuliert, die mit dem Beobachtungsmanager realisiert werden soll.

Ein dreidimensionaler Körper wird analog zur Methode der Finiten Elemente durch geometrische Elemente beschrieben. Ein Bild zeigt eine Teilmenge der geometrischen Elemente des dreidimensionalen Körpers. Zur Beurteilung des physikalischer Verhaltens von dreidimensionalen Körpern ist die gleichzeitige Visualisierung des Körpers in verschiedenen Bildern notwendig. In diesen Bildern können verschiedene Teile des Modells, verschiedene Projektionen,

unterschiedliche physikalische Zustände oder verschiedenen Präsentationsformen eines physikalischen Zustandes dargestellt werden. Ein Bild wird im Visualisierungsmodell unter anderem durch eine Menge von geometrischen Elementen definiert. Ein Element kann Bestandteil verschiedener Bilder sein. Die Modellierung des Objekttyps *Bild* ist in Abb.4 gezeigt.

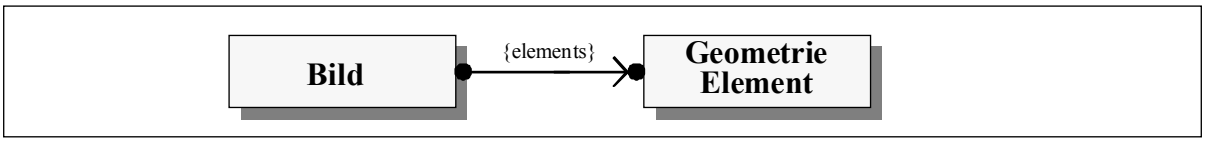

Abb.4 : Modellierung der Objektklassen Bild und Geometrie-Element

Wird ein Element aus dem Visualisierungsmodell entfernt, so werden die Bilder, die das Element enthalten, inkonsistent. Alle Bilder, die das geometrische Element enthalten, sollen automatisch aktualisiert werden. Die Aktualisierung nach dem Beobachtermechanismus erfordert die folgenden Schritte. Nach dem Beobachtungsmechanismus sind die Bilder die Beobachter und die geometrischen Elemente die Beobachtungsziele. Für die Beobachtungsziele wird ein Beobachtungsmanager erzeugt. Die Beobachtungsziele definieren als eine ihrer potentiellen Meldungen unter anderem die Meldung *CVkDestroyElement*. Die Meldung wird gesendet, bevor das Element gelöscht wird. Die Bilder beobachten die enthaltenen Elemente hinsichtlich dieser Meldung. Wird eine Meldung empfangen, so kann das Bild zur Konsistenzsicherung dieses Element aus seiner Elementmenge entfernen und eine erneute Darstellung veranlassen. Die Modellierung der dynamischen Schnittstelle ist in der Notation in Abb.5 gezeigt.

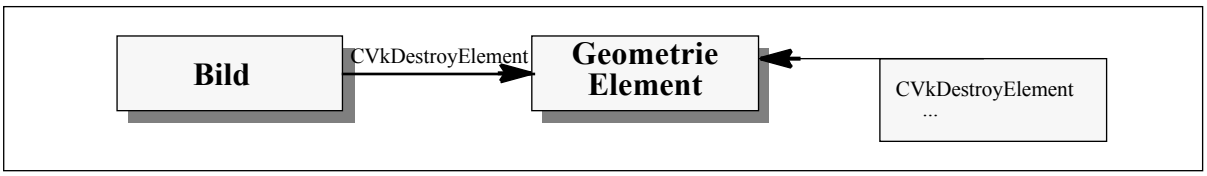

Abb.5 : Beobachtung Bild geometrisches Element

Mit den definierten Methoden des Beobachtungsmanagers wird die Beobachtung in den folgenden Schritten aufgebaut und wieder abgebaut :

Beobachtungsziel : VkGElement* element ;
Beobachter : VkPicture* picture ;

- Anmelden des Beobachtungszieles

  ```
  RmManageObject ( vkElementObserver,element) ;
  ```

- Setzen der Beobachtung für die Meldung CVkDestroyElement

  ```
  RmObserveObject( vkElementObserver,
                   picture,element,CVkDestroyElement,
                   _VkDeOMGElementToPicture,NULL) ;
  ```

- Aufheben der Beobachtung für die Meldung CVkDestroyElement

  ```
  RmNoObserveObject ( vkElementObserver,
                      picture,element,CVkDestroyElement) ;
  ```

- Abmelden des Beobachtungszieles

  ```
  RmUnmanageObject ( vkElementObserver,element) ;
  ```

Wird das Element aus dem Visualisierungsmodell entfernt, so sendet das Element vorab die Meldung CVkDestroyElement an den Beobachtungsmanager. Dieser informiert die Bilder

über dieses Ereignis. Den Beobachtern sendet das Beobachtungsziel seine Identität als zusätzliche Information mit. Dies sieht mit den Methoden des Beobachtungsmanagers wie folgt aus :

- Melden des Ereignisses an den Beobachtungsmanager

  RmCallObserver (vkElementObserver,element,CVkDestroyElement,element);

Der Beobachtungsmanager identifiziert die zu informierenden Bilder und ruft jedes Bild mit der hinterlegten Methode auf.

```
void _VkDeOMGElementToPicture
(    VkGElement*     this       ,
     QuData          storeData  ,
     VkPicture*      sendPicture )
{    ...        }
```

5. BEWERTUNG

Der Beobachtungsmechanismus wurde bisher im Forschungsprojekt DAVOS (Dynamische Analyse und Visualisierung von Staumauern), das von der Deutschen Forschungsgemeinschaft DFG) gefördert wird, zur Aktualisierung und Konsistenzsicherung der verschiedenen Modelle eines komplexen Finite-Elemente-Systems erfolgreich eingesetzt.

Die systematische Modellierung der Aktualisierung und der Konsistenzsicherung mit dem gezeigten Beobachtermechanismus führte zu einer sauberen Trennung der strukturellen Eigenschaften der Objekte von den dynamischen Eigenschaften. Die Reaktionen anderer Objekte brauchten zum Zeitpunkt der Modellierung nicht bekannt sein und konnten ohne Änderung der bestehenden Modellierung und Codierung zu einem späteren Zeitpunkt ergänzt werden. Diese Flexibilität in der Erweiterung blieb auch erhalten, wenn das System um neue Komponenten erweitert oder eine bestehende dynamische Schnittstelle ergänzt wurde.

Die Modellierung komplexer Algorithmen konnte ebenfalls als ein Konsistenzproblem aufgefaßt werden. Dazu werden komplexe Algorithmen in Kernalgorithmen und assoziierte Algorithmen zerlegt. Die Objekte eines assoziierten Algorithmus reagieren auf Änderungen der Objekte des Kernalgorithmus. Ein komplexer Algorithmus wird flexibel erweitert, indem neue assoziierte Algorithmen über Beobachtungsbeziehungen definiert werden.

Der Beobachtungsmechanismus führte zu einem merkbaren Effizienzverlust, wenn die Anzahl Objekte eines Systems hoch ist, wie beispielsweise bei einem Finite-Elemente-System zur Beschreibung einer Bogenstaumauer. Die Effizienzverluste konnten weitestgehend vermieden werden, indem Objekte zu Mengen zusammengefaßt wurden. Die Beobachtung konnte dann statt zwischen Objekten zwischen Mengen hergestellt werden. Die Anzahl der Mengen war erheblich geringer als die Anzahl Objekte.

Literatur

[1] Enseleit, J., Laabs, A., Damrath, R. Pahl, J. P :
Analyse und Visualisierung zeitabhängiger physikalischer Zustände dreidimensionaler Körper
Arbeitsberichte an die DFG 1996

[2] Hitz, M.
C++ Grundlagen und Programmierung
Springer Verlag Wien, New York 1992

[3] Kappel, G., Schrefl, M.
Objektorientierte Informationssysteme, Konzepte, Darstellungsmittel, Methoden
Springer Verlag Wien 1996

[4] Laabs, A. , Huhnt, W., Damrath, R. :
Objektorientierte Visualisierung Physikalischer Zustände
Berichte an die DFG 1992, 1993, 1994