Architekturen von Differenzwerkzeugen für Modelle

Udo Kelter, Maik Schmidt, Sven Wenzel
Praktische Informatik
Universität Siegen, Hölderlinstr. 3, 57068 Siegen
{kelter,mschmidt,wenzel}@informatik.uni-siegen.de

Abstract: Die modellbasierte Softwareentwicklung erfordert in der Praxis die üblichen Versionsmanagement-Dienste, also insb. Werkzeuge, die Differenzen zwischen Modellen anzeigen oder Modelle mischen können. Solche Werkzeuge unterscheiden sich erheblich von Differenzwerkzeugen für Text-Dokumente: Für die Darstellung einer Differenz müssen neue Wege gefunden werden, und im Prinzip müssen für jeden Diagrammtyp dedizierte Werkzeuge entwickelt werden, insg. also eine ganze Systemfamilie. Dieses Papier stellt mehrere "Standardarchitekturen" für Differenzwerkzeuge für Modelle vor und bewertet diese Architekturen und die zugehörigen Darstellungsformen hinsichtlich Qualität der Darstellung, Implementierungsaufwand und weiteren Kriterien.

1 Motivation

Die modellbasierte Softwareentwicklung ist in einigen Applikationsbereichen inzwischen gängige Praxis, beispielsweise im Bereich eingebetteter Systeme für steuerungs- und regelungstechnische Aufgaben. Zahlreiche Anbieter wie Etas oder Mathworks adressieren mit ihren Modellierungsumgebungen wie ASCET [Et07] oder Matlab/Simulink [Ma07] gezielt einzelne Applikationsdomänen, wie den Automotive- oder Luft- und Raumfahrtsektor. In diesen spezialisierten Applikationsdomänen werden anstelle der UML wesentlich spezialisiertere Modellierungssprachen eingesetzt, die besser an die jeweiligen Bedürfnisse angepasst sind. In der Praxis zeigt sich nun sehr deutlich, dass man für Modelle die gleichen Versionsmanagement-Dienste benötigt, die man für textuelle Dokumente gewohnt ist, namentlich Werkzeuge bzw. Dienste zum Vergleichen und Mischen von Modellen. An entsprechenden qualitativ hochwertigen Werkzeugen herrscht zur Zeit noch großer Mangel, dies wird vielfach als ein wesentliches Hindernis eingeschätzt, modellbasierte Softwareentwicklung praktisch zu betreiben. Hauptursachen dieses Mangels sind:

- Es müssen neue Formen der Anzeige von Differenzen, der Interaktion mit Entwicklern und der Integration mit anderen Werkzeugen entwickelt werden.
- Für jeden Modelltyp braucht man im Prinzip eigene, dedizierte Werkzeuge. An den Modelltyp angepasst werden müssen die Anzeigeformen, die Differenzberechnung, die Konfliktbehandlung beim Mischen, die Integration mit anderen Werkzeugen für diesen Modelltyp usw.

In diesem Papier stellen wir daher mehrere Werkzeugarchitekturen vor, analysieren deren Vorteile und Einschränkungen, insb. für welche Arten von Modellen sie sinnvoll anwendbar sind. Wir werden zeigen, dass die Differenzberechnung als eigenständige Komponente verallgemeinert werden kann, die in verschiedenen Architekturen wiederverwendet wird. Ferner hat sich herausgestellt, dass die Darstellung der Differenz in engen Zusammenhang mit dem Modelltyp steht, weshalb nicht eine einzige Architektur gleichermaßen für alle Modelltypen geeignet ist. Hierin fließen Erfahrungen ein, die in mehreren Projekten gewonnen wurden, in denen Differenzwerkzeuge für unterschiedliche Modelltypen entwickelt und in die entsprechenden Entwurfsumgebungen integriert wurden.

2 Begriffsdefinitionen

Dieser Abschnitt führt einige Begriffe ein, die wir für die Beschreibung der Komponenten von Differenzwerkzeugen benötigen. Eine ausführliche Darstellung findet sich in [Ke07].

Grundbegriffe. Unter einer **Differenz** verstehen wir informell ausgedrückt *eine* Darstellung davon, wie sich zwei Modelle unterscheiden. Die Unterschiede zwischen zwei Modellen kann man i.a. auf verschiedene Art darstellen, daher ist "die" Differenz zwischen zwei Modellen nicht eindeutig definiert (im Gegensatz zum mathematischen Begriff der Differenz als Ergebnis einer Subtraktion). Im gesamten Themenkomplex müssen folgende Aspekte getrennt werden:

1. die Definition des **konzeptuellen Inhalts** einer Differenz
2. die **physische Darstellung** einer Differenz in einem Speichermedium
3. die **externe**, oft graphische **Darstellung** einer Differenz, die i.d.R. auf der Standarddarstellung des jeweiligen Modelltyps basiert
4. die **Berechnung** bzw. **Erzeugung** einer Differenz. Sehr oft werden Differenzen als Ergebnis des **Vergleichs** zweier Modelle erzeugt. Differenzen können aber auch durch Protokollieren von Editieroperationen, durch Verarbeiten vorhandener Differenzen und durch weitere Methoden erzeugt werden. Verfahren zur Berechnung von Differenzen behandeln wir hier nicht (eine detaillierte Diskussion findet sich in [Ke07]).
5. das **Mischen** von Modellen: dieses basiert auf einer Differenz, die i.d.R. durch Vergleich von 2 (oder 3) Ausgangsmodellen gewonnen wird. Ziel ist, ein drittes Dokument zu erzeugen, das die "Besonderheiten" der beiden Dokumente enthält.

   Selbst dann, wenn die Ausgangsmodelle korrekt sind, ist das Mischergebnis nicht automatisch korrekt und muss i.a. manuell korrigiert werden. Um diesen Korrekturaufwand zu reduzieren, kann man darauf verzichten, einzelne Besonderheiten der Ausgangsmodelle in das Mischergebnis zu übernehmen; eine Entscheidung, ob eine Besonderheit übernommen wird, bezeichnen wir als **Mischentscheidung**.

Die meisten Mischentscheidungen sind positiv und können anhand bestimmter Indizien automatisiert getroffen werden. Wenn eine Mischentscheidung nicht automatisiert getroffen werden kann, spricht man von einem **Konflikt**.

Asymmetrische Differenzen. Es gibt zwei grundlegende Varianten des Begriffs Differenz: asymmetrische und symmetrische. Eine **asymmetrische Differenz** von einem Dokument D1 nach einem Dokument D2 ist eine Transformationsvorschrift, durch die D1 in D2 transformiert werden kann. Dies entspricht der Denkweise bei Patch-Werkzeugen und der internen Delta-Speicherung in Dokument-Repositorys. Die Transformation besteht aus einer Sequenz von Operationsaufrufen, in denen auszuführende Operationen gemäß einem Editierdatentyp (s.u.) und passende Parameter angegeben sind.

Man kann mit einer asymmetrischen Differenz von D1 nach D2 nicht wieder das Dokument D2 nach D1 zurücktransformieren, hierzu wird eine andere Transformation benötigt.

Editierdatentypen. Wir unterstellen, dass Dokumente mittels bestimmter dokumenttypspezifischer Operationen verändert ("editiert") werden können und dass diese Operationen durch einen abstrakten Datentyp definiert sind. Wir bezeichnen diesen Datentyp als den **Editierdatentyp**. Der Editierdatentyp muss Operationen zum Einfügen, Löschen, Ändern usw. von Dokumentelementen beinhalten, so dass letztlich alle Änderungen, die mit den Editoren dieses Dokumenttyps vorgenommen werden, nachvollziehbar sind.

Zu einem Dokumenttyp kann es unterschiedliche Editierdatentypen geben: beispielsweise kann man das Verschieben von Dokumentelementen als eigene Operation zulassen oder nicht. Die Wahl des Editierdatentyps ist nicht trivial.

Symmetrische Differenzen. Die Grundidee von symmetrischen Differenzen besteht darin, den aus der *Mengenlehre* gut bekannten Begriff der symmetrischen Differenz auf Dokumente zu übertragen. Hierzu muss man die Dokumente vereinfachend als Mengen von Dokumentkomponenten betrachten. Symmetrische Differenzen sind die Grundlage aller üblichen Differenzanzeigewerkzeuge und Basis für Mischungen. Ferner basieren fast alle Algorithmen, die Modelle vergleichen, begrifflich auf symmetrischen Differenzen.

Die mengenbasierte Definition eignet sich vor allem für eine sehr informelle Betrachtung von Differenzen. Man kann sie allerdings fast nie praktisch anwenden, weil fast alle Dokumente Multimengen sind, also an verschiedenen Stellen identische Komponenten (sog. Dubletten) enthalten können, und eine nichttriviale Struktur (Sequenz, Baum, allgemeiner Graph) haben. Für Multimengen kann der konzeptuelle Inhalt einer Differenz wie folgt definiert werden. Eine **Differenz** zwischen zwei Dokumenten D1 und D2 besteht aus

1. einer Menge von Korrespondenzen. Eine **Korrespondenz** ist ein Paar von je einer Komponente von D1 bzw. D2, die als einander entsprechend festgelegt werden.

   Sei i.f. `KK(D1,D2)` die Menge der **K**omponenten von D1, die einen **K**orrespondenzpartner in D2 haben, zusammen mit der Struktur gemäß D1.

2. je einer **Einfüge-Transformation** pro Dokument, die ausgehend von `KK(D1,D2)` bzw. `KK(D2,D1)` das komplette Dokument D1 bzw. D2 rekonstruieren. Jede Einfüge-Transformation enthält ausschließlich einfügende Operationen, keine Änderungen oder Löschungen. Die durch diese Transformationsvorschriften eingefügten Komponenten werden als **spezielle Komponenten** von D1 bzw. D2 bezeichnet.

Die in der Definition benutzte Menge von Korrespondenzen wirkt auf den ersten Blick ziemlich theoretisch. Tatsächlich beschreibt sie jedoch eine ganz zentrale Datenstruktur in den gängigsten Architekturen von Differenzwerkzeugen.

Fast alle externen Darstellungen von Differenzen beruhen begrifflich auf Korrespondenzen. Manche Darstellungen von Differenzen stellen zwei korrespondierende Komponenten nur einmal dar, beide Komponenten müssen hier *exakt identisch* ein. Andere Darstellungen stellen korrespondierende Komponenten doppelt dar und den Sachverhalt der Korrespondenz durch graphische Mittel; hier können beide Komponenten als irrelevant erachtete Unterschiede aufweisen, müssen aber *ähnlich* zueinander sein, sonst wäre die Korrespondenz unsinnig. Die Berechnung einer Differenz ist viel einfacher, wenn nur nach Paaren identischer Komponenten gesucht werden muss. Daher stellen die externen Darstellungsformen von Differenzen unterschiedliche Anforderungen an die Differenzberechnung.

3 Bewertungskriterien für Differenzdarstellungen- und Werkzeuge

Im Folgenden listen wir mehrere Bewertungskriterien auf, die vor allem aus Benutzersicht für die Leistungsfähigkeit von Differenzwerkzeugen für Modelle wichtig sind. Die Kriterien betreffen vorwiegend die Anzeigeform und die Steuerbarkeit der Differenz:

- Können nur exakt gleiche Modellelemente als korrespondierend dargestellt werden oder auch ungleiche, aber ähnliche Modellelemente? Die Antwort hängt oft vom Typ des Modellelements ab. Sofern Korrespondenzen zwischen ungleichen Modellelementen dargestellt werden können, stellt sich die Frage, ob der Grad der Veränderung dargestellt wird. Speziell bei Modellelementen, die wenige oder keine lokale ähnlichkeitsrelevante Eigenschaften haben, ist an der graphischen Standarddarstellung nicht erkennbar, ob die Modellelemente identisch oder nur ähnlich sind.

- Können Verschiebungen von einzelnen Modellelementen dargestellt werden? Hierbei ist zu unterscheiden zwischen **lokalen Verschiebungen**, z.B. der Vertauschung von zwei Parametern in einer Parameterliste, und **nichtlokalen Verschiebungen**, z.B. der Verschiebung eines Attributs in eine andere Klasse.

- Können Verschiebungen von kompletten Teilnetzen dargestellt werden? Bei Modelltypen, die beliebige blockorientierte Schachtelungen erlauben, will man oft einen inneren Block erzeugen, der einen Teil des bisherigen Diagramms enthält. Die "Bedienschnittstelle" dieser Operation kann so gestaltet werden, dass man die Modellelemente, die das Innere des Blocks bilden sollen, einrahmt und dann den Umbau

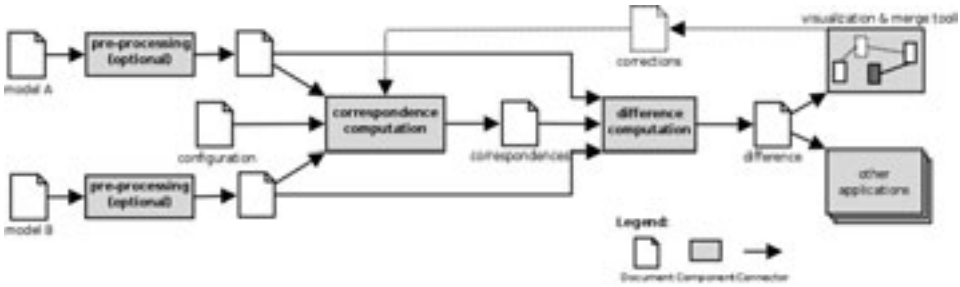

Abbildung 1: Prozess der Differenzberechnung

veranlasst. Man kann diesen Umbau auf eine Vielzahl von elementaren Verschiebungen und den Ab- bzw. Aufbau von Beziehungen zurückführen; diese aber alle einzeln darzustellen wäre sehr verwirrend.

- Ist die Anzeigeform auf einen Mehrwegevergleich zwischen 3 und mehr Modellen verallgemeinerbar?
- Müssen die graphischen Darstellungen ein neues Layout bekommen? Das Layout von graphischen Darstellungen von Modellen ist zwar sehr wichtig für die Lesbarkeit, wird aber i.d.R.[1] als irrelevant für die Korrespondenzbildung betrachtet.
- Kann man Differenzen punktuell korrigieren? Differenzen sind nicht eindeutig. Zwischen den Alternativen, den Unterschied zwischen zwei Modellen darzustellen, kann manchmal nur subjektiv entschieden werden, und die Entscheidung des Vergleichsalgorithmus kann unerwünscht sein. Korrekturen können zwei Formen haben: eine vorhandene Korrespondenz kann unerwünscht sein, oder zwei als nicht korrespondierend erkannte Modellelemente sollen trotzdem eine Korrespondenz bilden.

4 Differenzberechnung

Eine Untersuchung existierender Ansätze zur Differenzberechnung [Eb07, Ec07, Et07, KWN05, Se07, Tr07] zeigt, dass die Verfahren im Detail sehr verschieden sind. Der Grobablauf der Differenzberechnung folgt i.d.R. dem Schema in Abbildung 1. Die zu vergleichenden Modelle werden nach einer optionalen Vorverarbeitung eingelesen und es werden Korrespondenzen zwischen den Elementen der Modelle gebildet. Die Korrespondenzbildung unterscheidet sich je nach Dokumenttyp und ist bei den verschiedenen Ansätzen unterschiedlich implementiert. Adaptierbare Algorithmen wie SiDiff [Tr07] benötigen lediglich einige Steuerinformationen. Diese Eigenschaft wird in Abbildung 1 durch die Kon-

[1] Eine Ausnahme ist z.B. die Korrespondenzbildung in ASCET [Et07], bei der eine Verschiebung eines Modellelements in einem Diagramm über einen Fangbereich hinaus als relevante Änderung eingestuft wird.

figuration repräsentiert, die verallgemeinert auch den auszuführenden Algorithmus enthalten kann.

Das Ergebnis des ersten Verarbeitungsschritts ist eine Menge von Korrespondenzen. Hieraus wird im nächsten Schritt eine Differenz abgeleitet. Der Typ der Differenz wird von der darauf folgenden Weiterverarbeitung determiniert. Interaktive Misch- oder Anzeigewerkzeuge können das Differenzergebnis bzw. die Menge der Korrespondenzen korrigierten; dann werden die Korrespondenzen mit der Korrekturinformation erneut berechnet.

Architektur der Differenzberechnungskomponente. Für die nachfolgende Betrachtung von Werkzeugarchitekturen kann man diesen grundlegenden Differenzbildungsprozess als eine Komponente zusammenfassen, die in allen Architekturen Anwendung findet. Der Grobablauf der Differenzberechnung wird von der umgebenden Applikation gesteuert, hier sind mehrere unterschiedlich optimierte Varianten wählbar.

Die innere Struktur dieser Differenzberechnungskomponente ist in Abbildung 2 zu sehen. Die zu vergleichenden Dokumente werden zentral in einem Dokumentrepository gespeichert, das von der Korrespondenzberechnung und der Differenzberechnung zugreifbar ist. Die u.U. notwendige Vorverarbeitung der Dokumente ist nicht als Teil der Differenzberechnungskomponente anzusehen und kann im Bedarfsfall außen vorgeschaltet werden.

Dem Dokumentrepository liegt eine Typverwaltung zugrunde, die überwacht, welche Elementtypen in den Dokumenten vorkommen. Je nach Implementierung kann die Typverwaltung statisch vorab oder dynamisch während des Einlesens der Dokumente aufgebaut werden. Die Typverwaltung wird zudem von der Korrespondenzberechnung benötigt, um sicherzustellen, dass nur Elemente gleichen Typs als korrespondierend erkannt werden. Ähnliche Konsistenzkriterien werden während der Differenzbildung benötigt, sie beschreiben i.d.R. den Editierdatentyp.

Die Korrespondenztabelle ist eine zentrale Datenstruktur, die jedem Element eines Dokuments das korrespondierende Element in einem anderen Dokument zuordnet, sofern ein derartiges Element existiert. Sie wird i.w. während der Korrespondenzberechnung gefüllt und während der Differenzbildung ausgelesen. Je nach Verfahren wird jedoch auch schon während der Korrespondenzberechnung lesend auf die Tabelle zugegriffen, wenn z.B. die Korrespondenzen benachbarter Knoten eine Rolle spielen.

Es fällt auf, dass die Komponenten der Differenzberechnung eng miteinander verzahnt sind. Konkrete Implementierungen der Korrespondenz- und Differenzberechnungskomponenten können aber beliebig ausgetauscht werden[2]. Andere Applikationen, z.B. Werkzeuge zur Historienanalyse [WHK07], benötigen nur Korrespondenzinformationen und können auch direkt auf der Korrespondenztabelle arbeiten; die differenzbildende Komponente wird hier nicht benötigt.

[2] Im Falle von SiDiff wird die Korrespondenzberechnung durch Konfigurationsdaten gesteuert [Tr07], man kann diesen Systemteil als Interpreter der Konfigurationsdaten auffassen.

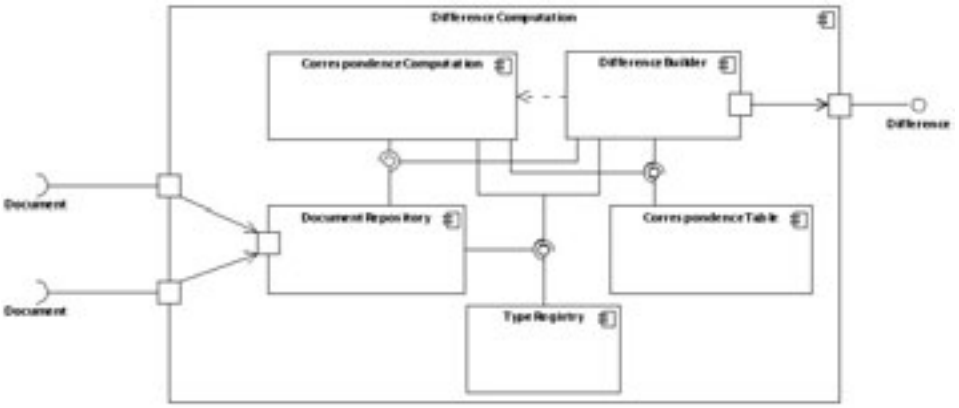

Abbildung 2: Grundlegende Architektur der Differenzberechnungskomponente

5 Darstellungsformen von Differenzen

5.1 Aggregierte Darstellungen

Man kann die Darstellungsformen von Differenzen einteilen in exakte Darstellungen, die keine Details weglassen, und aggregierte bzw. vergröbernde Darstellungen. Beispiele für aggregierte Darstellungen sind Statistiken zu einer Differenz, die analog zu Programm-Metriken definiert werden können und die textuell, graphisch, durch Farbcodierung oder mit anderen Methoden dargestellt werden können.

Eine Mischform besteht darin, die Grobstruktur der Modelle und Unterschiede daran exakt darzustellen, Unterschiede in der Feinstruktur aber nur in aggregierter Form. I.f. werden wir nur noch exakte Darstellungsformen behandeln.

5.2 Textuelle Darstellung

Textuelle Darstellungen sind sowohl bei symmetrischen wie asymmetrischen Differenzen leicht realisierbar. Korrespondenzen können durch Paare von Bezeichnern von Modellelementen dargestellt werden, Transformationen durch textuell notierte Sequenzen von Operationen. Offensichtlicher Nachteil ist die schlechte Anschaulichkeit.

Sofern die Editierdatentypen komplexe Operationen enthalten, was insb. bei asymmetrischen Differenzen auftreten kann, versagen jedoch alle anderen Darstellungsformen, hier sind textuelle Darstellungen oft die einzige Alternative. Ähnliches gilt für sehr große Modelle mit umfangreichen Unterschieden.

5.3 Integrierte Paralleldarstellung

Bei allen Paralleldarstellungen werden beide Dokumente komplett in Fenstern oder Unterfenstern dargestellt und horizontal oder vertikal ausgerichtet einander gegenübergestellt. Für Textdokumente wird diese Darstellung häufig verwendet. Für Modelle eignet sie sich nur dann, wenn die Modelle sehr klein sind oder zumindest horizontal oder vertikal wenig Ausdehnung haben, z.B. Timing-Diagramme oder Sequenzdiagramme. Andernfalls muss das Modell in eine geeignete Darstellungsform mit geringer Ausdehnung (z.B. eine Baumdarstellung) transformiert werden, was jedoch die Anschaulichkeit mindert. Für die Darstellung von Korrespondenzen gibt es zwei Hauptvarianten:

Bei der **Paralleldarstellung mit "Dehnstellen"** werden korrespondierende Komponenten neben- oder übereinanderstehend angezeigt, Korrespondenzen werden also durch Layout sichtbar gemacht. Diese Darstellungsform kann daher prinzipiell keine Verschiebungen darstellen. Korrespondierende Komponenten können hier Unterschiede aufweisen.
Eine nur in einem Dokument vorhandene Komponente wird an ihrer Position geeignet markiert angezeigt, im Dokument gegenüber entsteht eine "Dehnstelle", d.h. dort wird ein eigentlich nicht vorhandener Leerraum angezeigt. Dieser verzerrt i.a. das Layout des Modells.

Bei der **Paralleldarstellung mit Verbindungslinien** werden Korrespondenzen zwischen Komponenten oder Blöcken von Komponenten durch **Verbindungslinien** dargestellt. Für Texte ist das Verfahren günstig, weil es Verschiebungen in begrenztem Umfang darstellen kann und man die Texte unabhängig scrollen kann. Bei graphartigen Modellen besteht die Gefahr, dass die Verbindungslinien und Graphkanten ein optisches Durcheinander bilden und die Unterschiede letztlich schlecht erkennbar sind.

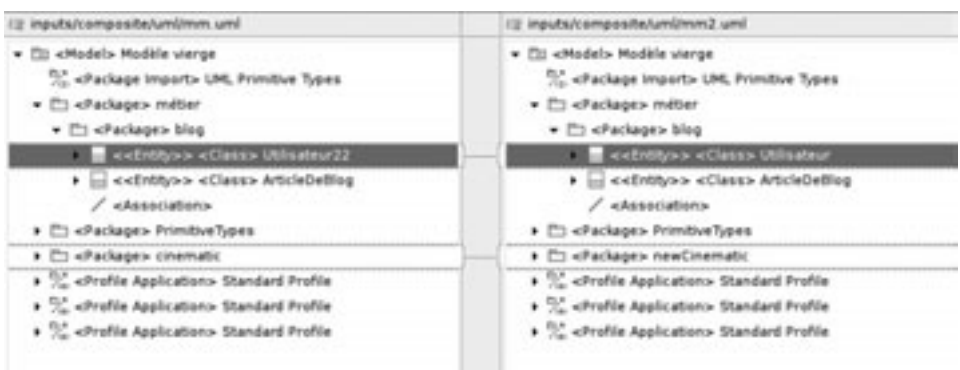

Abbildung 3: Paralleldarstellung in EMF Compare [Ec07]

Alle integrierten Paralleldarstellungen verursachen einen sehr hohen, oft prohibitiven Implementierungsaufwand, da vorhandene Bedienelemente i.d.R. nicht wiederverwendet werden können. Zum Vergleich von 3 oder mehr Modellen ist diese Anzeigeform kaum geeignet. Zudem ist diese Darstellung für größere Dokumente ungeeignet. Entweder reicht der Bildschirmplatz nicht aus, um zwei Modelle parallel darzustellen, oder die räumliche

Distanz wird so groß, dass Korrespondenzen nur schwer zu überblicken sind.

In der Praxis findet man daher anstelle der Paralleldarstellung in der ursprünglichen Modellierungssprache lediglich der textuellen Darstellung angelehnte Baumdarstellungen, wie am Beispiel EMF Compare [Ec07] (Abbildung 3) zu sehen ist.

5.4 Mehrfenstertechnik und Liste klickbarer lokaler Unterschiede

Sofern ein von außen ansteuerbares Anzeigewerkzeug für einzelne Modelle verfügbar ist, bietet sich folgende Lösung an, die man als Zwitter zwischen textueller Anzeige und Paralleldarstellung ansehen kann. In einem Steuerfenster wird eine **klickbare Liste lokaler Unterschiede**[3] angezeigt. Die Liste sollte jeden lokalen Unterschied mit einem domänenspezifischen Kurztext beschreiben.

Abbildung 4: Liste lokaler Unterschiede

Wenn man einen Listeneintrag anklickt, werden die Dokumente in jeweils einem eigenen Editorfenster des Anzeigewerkzeugs angezeigt und möglichst auf die jeweils involvierten Dokumentteile positioniert. Das ursprüngliche Layout der Dokumente bleibt dabei erhalten. Im Gegensatz zu allen anderen Darstellungsformen kann eine Differenz hier nicht vollständig angezeigt werden, sondern nur stückweise[4]. Die Differenzanzeige ist hier prinzipbedingt interaktiv.

Ein prinzipielles Problem dieser Anzeigeform liegt darin, dass ein Betrachter selbst durch Vergleich der beiden Fensterinhalte den Kontext der Änderung herausfinden muss. Hierbei kann er durch Einfärben der betroffenen Modellelemente unterstützt werden; hierzu muss

[3] Informell kann ein lokaler Unterschied als “kleine Differenz”, die graphisch überschaubar angezeigt werden kann, definiert werden. Eine präzisere Definition ist aufwendig, s. hierzu [Ke07].

[4] Alle anderen Darstellungsformen erlauben es, Differenzen nur teilweise anzuzeigen bzw. selektiv zu markieren; dies kann bei der Anzeige großer Differenzen nützlich sein.

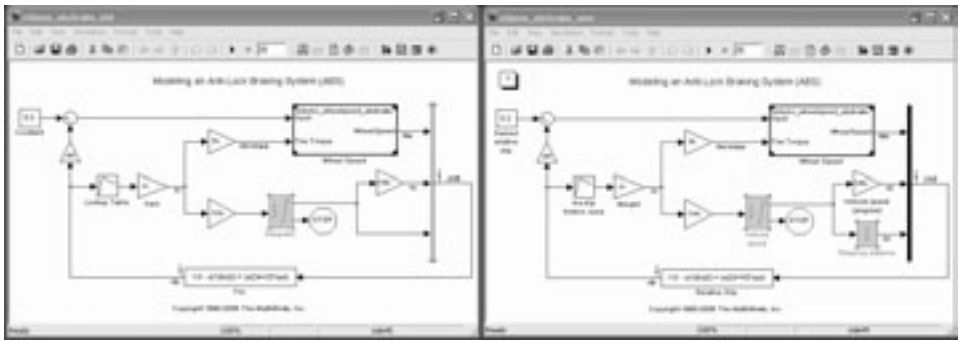

Abbildung 5: Zwei Editorfenster nach einer Selektion

das Anzeigewerkzeug entsprechend von außen steuerbar sein. Die hier ggf. notwendigen Erweiterungen am Anzeigewerkzeug sind aber immer noch weitaus weniger aufwendig als die Implementierung einer integrierten Paralleldarstellung. Dieser Ansatz kann leicht auf eine beliebige Anzahl von Dokumenten verallgemeinert werden. Wird ein Editor als Anzeigewerkzeug verwendet, können die Dokumente sofort weiterbearbeitet werden.

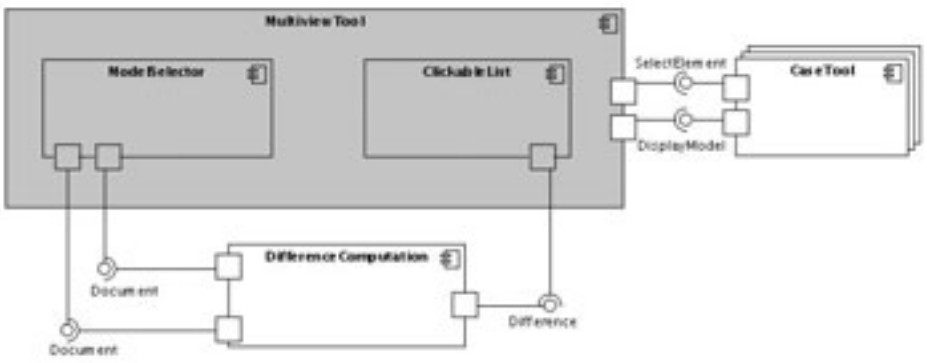

Abbildung 6: Architektur für Mehrfensterdarstellung

Zur die Realisierung dieser Darstellungsform ist es notwendig, ein vorhandenes Werkzeug gemäß Abbildung 6 zu erweitern. Im Rahmen des MATE-Projektes [St07] wurde dies für die Matlab/Simulink-Plattform durchgeführt. Eine im Kontext der Matlab-Umgebung zu Startende GUI Komponente (Abbildung 4) ermöglicht die Auswahl der zu vergleichenden Modelle. Gleichzeitig dient sie als interaktiver Interpreter, der die Ergebnisse der Differenzbildung in Form einer Liste darstellt und nach Interaktion die an der Differenz beteiligten Elemente der verglichenen Modelle in zwei Fenstern fokussiert (Abbildung 5). Die Differenzbildung wird von der Differenzberechnungskomponente (vgl. Abschnitt 4) übernommen. Eine detaillierte Beschreibung des Prototypen kann [SW07] entnommen werden.

5.5 Darstellung in einem Vereinigungsdokument

Bei der Anzeige in einem Vereinigungsdokument wird eine “ad-hoc-Mischung” beider Dokumente erzeugt, in der korrespondierende Komponenten nur einmal enthalten sind. Hauptvorteil der Vereinigungsdarstellung ist der geringere Platzbedarf und das bessere Erkennen des gemeinsamen Kontextes eines lokalen Unterschieds, optisch ist diese Darstellungsform in vieler Hinsicht am attraktivsten. Auf ihrer Basis kann man auch bessere Mischwerkzeuge realisieren als auf Basis der Paralleldarstellung.

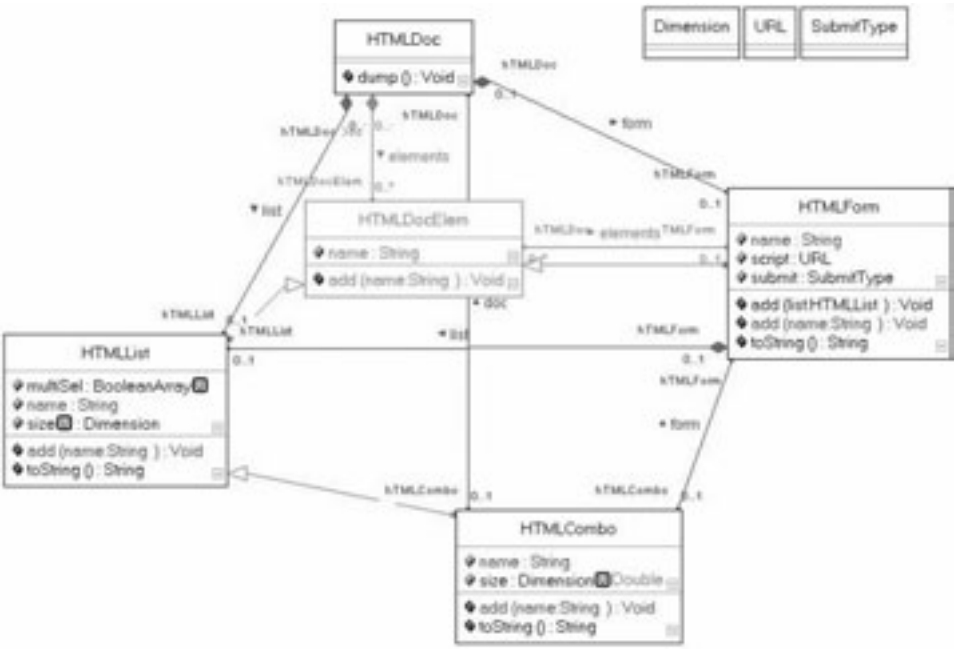

Abbildung 7: Vereinigungsdarstellung bei Fujaba [Fu07]

Nachteilig ist der (überraschend) hohe Implementierungsaufwand. Durch die Vereinigung werden i.d.R. die Konsistenzkriterien des Basismodelltyps verletzt (z.B. Namensräume), d.h. man kann Editoren des Basismodelltyps nicht ohne weiteres verwenden. Selbst dann, wenn die Editoren des Basismodelltyps z.B. durch eine Meta-CASE-Architektur leicht modifizierbar sind, können i.d.R. nicht alle Anpassungen durch Änderungen der Metamodelle abgefangen werden (z.B. geänderte Namen nebeneinander darstellen, vgl. [OWK03]), d.h. es werden aufwendige Änderungen erforderlich.

Ein weiteres gravierendes Problem ist die Layoutanpassung. Entweder muss ein völlig neues Layout für die ad-hoc-Mischung generiert werden, was vielfach auf Ablehnung stößt, oder man nimmt eines der Modelle als Basis und fügt darin an passender Stelle die speziellen Komponenten des anderen Modells ein; dann muss ggf. Platz geschaffen werden, wobei das ursprüngliche Layout möglichst wenig verändert werden sollte.

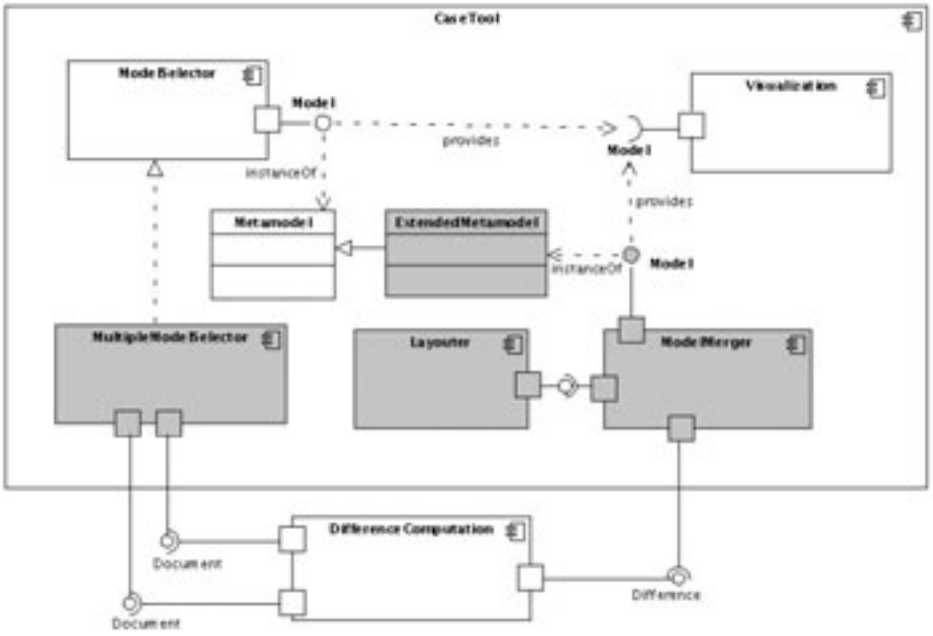

Abbildung 8: Architektur zur Darstellung eines Vereinigungsdokuments

Abbildung 8 zeigt die Architektur dieser Darstellungsform. Es wird ein Editor mit Meta-CASE-Architektur vorausgesetzt, bei dem das Metamodel angepasst werden kann. Anpassungen am Metamodell sind nötig, um z.B. alternative Elemente doppelt anzuzeigen, Elemente einzufärben, oder Attributänderungen anzeigen zu können. Ferner muss das Werkzeug um eine Modellauswahl erweitert werden, z.B. in Form eines Dialoges, um die zu vergleichenden Modelle selektieren zu können. Zur Differenzberechnung kann eine eigenständige Komponente genutzt werden. Die Differenz muss jedoch weiterverarbeitet, und es muß ein Vereinigungsdokument erzeugt werden.

Diese Darstellungsform wurde beim Differenz-Plugin für das CASE-Tool Fujaba verwendet [Fu07] (Abbildung 7).

6 Zusammenfassung

Differenz- und Mischwerkzeuge können unterschiedliche Darstellungsformen nutzen. Jede dieser Darstellungsformen impliziert eine eigene Werkzeugarchitektur sowie einen definierten (konzeptuellen) Inhalt der darstellbaren Differenz. Die Autoren haben verschiedene Formen in unterschiedlichen Projekten eingesetzt und evaluiert.

In diesem Rahmen ist es gelungen, die Differenzbildung modelltypübergreifend zu modu-

larisieren und somit unter geringem Aufwand wiederzuverwenden. Demgegenüber mussten die darstellungsformbezogenen Teile individuell und mit erheblichem Aufwand für jedes Werkzeug neu entwickelt werden.

		Textdarstellung	Paralleldarstellung Dehnstellen	Paralleldarstellung Verbindungslinien	Interaktive klickbare Liste	Vereinigungs-Dokument
Darstellbarkeit von Editier-operationen	Hinzufügen u. Löschen	✓	✓	✓	✓	✓
	Attributwert änderung	✓	(✓)	(✓)	✓	✓
	Lokale Verschiebung	✓	✗	✓	✓	✓
	Globale Verschiebung	✓	✗	(✓)	✓	✗
	Komplexe Operationen	(✓)	✗	✗	✓	✗
Zeitgleiche Darstellung		ja (global)	ja (global)	ja (global)	nein (lokal)	ja (global)
Bewertung	Anschaulichkeit	gering	bedingt	bedingt	gut	gut
	Kontext erkennbar?	nein	ja (verzerrt)	ja	ja	ja
Implementierungsaufwand		sehr gering	hoch	hoch	gering	hoch

Tabelle 1: Inhalt und Bewertung unterschiedlicher Differenzdarstellungen

Tabelle 1 fasst die gewonnenen Erfahrungen zusammen. Die textuelle Darstellung kann mit wenig Aufwand implementiert werden und hat den nicht zu unterschätzenden Vorteil, im Prinzip beliebige Editieroperationen darstellen zu können. Die Darstellung ist andererseits wenig anschaulich, weil die gängigen modelltypspezifischen graphischen Darstellungen fehlen, ferner wird den Kontext einzelner Modifikationen unzureichend wiedergegeben.

Alle anderen Darstellungsformen stellen Modelle in der gewohnten Notation dar. Die Paralleldarstellung eignet sich jedoch nur für ”lange und schmale” Modelle. Allgemein sind Verschiebungen und komplexe Operationen nicht oder nur schlecht darstellbar. Eine Ausnahme bildet die Mehrfensterdarstellung mit klickbarer Liste. Diese ist aus Sicht der Autoren ist in den meisten Fällen der günstigste Kompromiss aus relativ geringem Implementierungsaufwand und präziser Darstellung der Differenz.

Literatur

[Eb07] Ebert, J.; Kelter, U.; Schürr, A.; Westfechtel, B.: Bericht über den Workshop Vergleich und Versionierung von UML-Modellen (VVUM07) im Rahmen der GI-Fachtagung Software Engineering 2007, Hamburg; Softwaretechnik-Trends 27:2; 2007

[Ec07] Eclipse Community: EMF Compare Projekt `http://www.eclipse.org/modeling/emft/?project=compare`; 2007

[Et07] ETAS: ASCET Software-Produkte `http://www.etas.com/de/products/ascet_software_products.php`, 2007

[Fu07] Fujaba Difference Plugin. `http://www.fujaba.de/projects/differences`; 2004

[Ke07] Kelter, U.: Lehrmodul Dokumentdifferenzen; Fachgruppe Praktische Informatik, Universität Siegen; 2007; http://pi.informatik.uni-siegen.de/kelter/lehre/lm/dif

[KWN05] Kelter, U.; Wehren, J.; Niere, J.: A Generic Difference Algorithm for UML Models; Proc. GI-Fachtagung Software Engineering 2005, Essen, LNI; 2005

[Ma07] Matlab/Simulink Software-Produkte; http://www.mathworks.de/products/simulink; 2007

[OWK03] Ohst, D.; Welle, M.; Kelter, U.: Differences between Versions of UML Diagrams; p.227-236 in: Proc. Joint European Software Engineering Conference (ESEC) and ACM SIGSOFT Symposium on Foundations of Software Engineering (FSE), Helsinki, 2003; ACM Press; ISBN 1-58113-743-5

[Se07] Selonen, P.: A Review of UML Model Comparison Techniques; p.37-51 in: Proc. 5th Nordic Workshop on Model Driven Engineering, 27-29 August 2007, Ronneby, Sweden; Research report, U. Göteborg; 2007; ISBN: 978-91-7295-985-9

[St07] Stürmer, I.; Dörr, H.; Giese, H.; Kelter, U.; Schürr, A.; Zündorf, A.: Das MATE Projekt - Visuelle Spezifikation von MATLAB/Simulink/Stateflow Analysen und Transformationen, Dagstuhl Seminar Modellbasierte Entwicklung eingebetteter Systeme; 2006

[SW07] Schmidt, M.; Wenzel, S.: Ähnlichkeitsbasierte Berechnung von Simulink-Modelldifferenzen mit SiDiff; p.61-71 in: Proc. Modellbasierte Software-Entwicklung für eingebettete Systeme, 2. Workshop der Special Interest Group „Model-Driven Software Engineering“, Stuttgart, 2007; ISBN 978-3-8325-1595-9

[Tr07] Treude, C.; Berlik, S.; Wenzel, S.; Kelter, U.; Difference computation of large models; p.295-304 in: Proc. Joint European Software Engineering Conference (ESEC) and ACM SIGSOFT Symposium on Foundations of Software Engineering (FSE), Dubrovnik, 2007; http://doi.acm.org/10.1145/1287624.1287665

[WHK07] Wenzel, S.; Hutter, H.; Kelter, U.: Tracing model elements; in: Proc. of the 23rd International Conference on Software Maintenance (ISCM'07), Paris, France; 2007