H. Wandke, S. Kain & D. Struve (Hrsg.): Mensch & Computer 2009: Grenzenlos frei!?
München: Oldenbourg Verlag, 2009, S. 183-192

Integrative Gruppen-, Rollen- und Aufgabenmodellierung

Birgit Bomsdorf[1], Andreas Reitschuster[2]

[1]FB Angewandte Informatik, Hochschule Fulda, [2]AUDI AG, Ingolstadt

Zusammenfassung

Für eine adäquate Beschreibung der Beziehung zwischen Benutzer und Aufgaben ist zwischen den organisatorischen Gruppen und den Rollen, in denen einzelne Aufgaben durchgeführt werden dürfen, zu unterscheiden. Diese Differenzierung hat bisher noch keinen Eingang in die Kernkonzepte der werkzeugtechnischen Aufgabenmodellierung gefunden. Dieser Beitrag schlägt einen entsprechenden Ansatz vor. Es entstehen Modelle, in denen Gruppen, Rollen und Aufgaben jeweils hierarchisch strukturiert sind und untereinander Zuordnungsbeziehungen aufweisen. Das hier vorgestellte Werkzeug bietet eine automatische Unterstützung bei der jeweiligen Hierarchiebildung. So geben die Modellierer die gewünschten Zuordnungen ein, auf deren Basis die resultierenden hierarchischen Strukturen unter Berücksichtigung verschiedener möglicher Konfliktfälle algorithmisch berechnen werden.

1 Einleitung

Die modellbasierte Entwicklung interaktiver Systeme stellt vielfach die von den Benutzern auszuführenden Aufgaben und darüber die dabei benötigten Aufgabenobjekte (Entitäten, Daten) in den Mittelpunkt der relevanten Modellkonzepte. Ein Benutzer darf jedoch nicht jede Aufgabe durchführen, er darf mit den Objekten nicht beliebig verfahren. So ist es nicht jedem Nutzer eines E-Shops erlaubt, die Daten registrierter Kunden zu sehen oder zu ändern. Die Art des Umgangs mit den Objekten (wie *Kunde*) wird im Aufgabenmodell beschrieben. Dieser Zusammenhang zwischen den Aufgaben und den Objekten wird über deren Zuordnung festgehalten, s. hierzu (Paternò et al. 1998), (van Welie et al. 1998), (Lu et al. 2002), (Stary 2005), (Baron & Scapin 2006) und (Bomsdorf 2007). Hinsichtlich der Beziehung zwischen Benutzern und Aufgaben zeigen die Modellierungsansätze eine größere Varianz. So wird innerhalb eines Ansatzes nicht zwischen organisatorischen Positionen und der Bündelung von Aufgaben unterschieden. In unserem Beispiel könnten nur Personen in der Rolle eines Administrators die Daten registrierter Kunden ändern dürfen, wobei diese Rolle vom Inhaber des E-Shops und von Mitarbeitern einer Gruppe IT-Service eingenommen werden könnte. Dieses kleine Beispiel zeigt bereits, dass die Differenzierung in organisatorische und

aufgabenbündelnde Rollen sinnvoll ist. So existiert sie in verschiedenen Arbeiten, wie etwa solchen zur Rechtevergabe oder Workflowmanagementsystemen - Eingang in die Kernkonzepte der Werkzeuge zur Aufgabenmodellierung hat sie jedoch kaum gefunden. Wird nur ein Rollenkonzept angeboten, können insbesondere die gegenseitigen Abhängigkeiten organisatorischer und aufgabenbündelnder Rollen nicht modelliert und damit auch nicht werkzeugunterstützt überprüft werden. Durch eine entsprechende Berücksichtigung nimmt aber auch die Komplexität der Modelle zu: Die Beschreibungen der organisatorischen Rollen und der Rollen zur Bündelung von Aufgaben sind gängiger Weise hierarchisch strukturiert. Mit den Aufgabenmodellen liegen somit drei Hierarchien vor, die verschiedene Querbezüge aufweisen.

Dieser Beitrag widmet sich diesem Problem und stellt eine Erweiterung der Aufgabenmodellierung und deren werkzeugtechnischer Umsetzung vor. Es wird damit die explizite Spezifikation sowohl von organisatorischen Rollen (hier bezeichnet als Gruppen) als auch von aufgabenbündelnden Rollen zur Vergabe von Privilegien ermöglicht. Benutzer sind dabei Mitglieder einer Gruppe und erlangen mittels assoziierter Rollen den Zugang zu Aufgaben und Aufgabenobjekten. Das entwickelte Werkzeug bietet automatische Unterstützung bei der Erstellung der Gruppen-, Rollen- und Aufgabenhierarchien unter Berücksichtigung von deren Interdependenzen. Es bietet zudem erste Möglichkeiten, verschiedene Sichten auf die Verknüpfungen zu generieren. Die Zielsetzung ist, durch entsprechende unterliegende Algorithmen die Modellierungskomplexität, insbesondere zur Vermeidung von Modellierungsfehlern, zu reduzieren. Im folgenden Kapitel 2 wird zunächst die Spezifikation der Benutzer-Aufgaben-Beziehung in existierenden Ansätzen betrachtet. Anschließend wird die in diesem Beitrag vorgeschlagene integrative Gruppen-, Rollen- und Aufgabenmodellierung vorgestellt. Hierzu wird die grundlegende Umsetzung mit Fokus auf den Rollen und deren Zuordnung zu Aufgaben präsentiert (Kapitel 4), während die Konzepte Aufgabe, Benutzer, Gruppe und Objekt deutlich kürzer betrachtet werden.

2 Existierende Arbeiten

Im Zentrum dieses Beitrags steht die werkzeugtechnische Handhabung der Benutzer-Aufgaben-Zuordnung. Daher werden hier solche aktuellen Aufgabenmodellierungsansätze betrachtet, zu denen entsprechende Editoren existieren.

In GTA (Groupware Task Analysis), unterstützt durch das Werkzeug Euterpe, existiert eine explizite Trennung zwischen Rollen und den Zugriffsrechten auf Objekte (van Welie et al. 1998). Im Gegensatz zu vielen anderen Aufgabenmodellen war GTA von Beginn an auf die Beschreibung von Gruppenarbeit ausgelegt. Die Rollen können hierarchisch strukturiert und den Aufgaben zugeordnet werden. Das Konzept Agent, das sowohl der Erfassung von Benutzern als auch von einzelnen Gruppen dient, wird einerseits mit Rollen und andererseits mit Objekten zur Beschreibung der Zugriffsrechte verknüpft. Euterpe bietet jedoch keine Prüfungen auf Modellierungsfehler, auch nicht hinsichtlich der Rollen- und Aufgabenhierarchien. Ein vielfach zitierter Ansatz ist CTT bzw. das Werkzeug CTTE (Paternò et al. 1998), in dem für jede Rolle ein separates Modell der von ihr allein auszuführenden Aufgaben er-

stellt wird. Die Zusammenarbeit der Rollenträger (Benutzer) wird in einem zusätzlichen, sog. kooperativen Aufgabenmodell spezifiziert. CTT/CTTE wird auch in anderen Arbeiten verwendetet, die damit das CTT-Rollenkonzept entweder übernehmen oder modifizieren. In (García et al. 2008) werden beispielsweise zunächst in Workflows die Rollen-Aufgaben-Zuordnungen getroffen und hiervon ausgehend die Aufgabenmodelle pro Rolle weiter verfeinert (mittels CTT, ohne kooperative Aufgaben). In MAD/K-MADe (Baron & Scapin 2006) hingegen werden den Aufgaben ein oder mehrere Benutzer (sog. Akteure) zugeordnet. Die Rollen werden als Attribut der Benutzer erfasst, d.h. eine separate Modellierung der Rollen existiert hier nicht. In Diane+/Tamot (Lu et al. 2002) werden Benutzer ebenfalls als Akteure einzelner Aufgaben beschrieben, während in TaskArchitect (Stuart & Penn 2004) Rollen als Eigenschaften von Aufgaben diesen zugeordnet werden, wobei eine weitergehende Rollenmodellierung nicht unterstützt wird. Im Werkzeug Amboss (Giese et al. 2008) dient das Rollenkonzept zunächst der Unterscheidung, ob ein Mensch oder ein System eine Aufgabe ausführt. Mittels hierarchischer Verfeinerung wird die sog. Rolle „human“, ähnlich zu GTA, in Gruppen und individuelle Benutzer untergliedert. Das Werkzeug TADEUS (Stary 2005) erlaubt die Differenzierung in organisatorische und aufgabenbündelnde Rollen. Es berücksichtigt darüberhinaus mittels Benutzerprofile individuelle Sichten auf die Durchführung von Aufgaben. Der in diesem Beitrag präsentierte Ansatz umfasst noch keine Benutzermodelle, unterstützt jedoch, wie im Weiteren vorgestellt, die Erstellung der Modellstrukturen unter Berücksichtigung gegenseitiger Abhängigkeiten.

Insgesamt kann festgehalten werden, dass Rollen der Beschreibung organisatorischer Aspekte (Limbourg & Vanderdonckt 2003) dienen und Verantwortlichkeiten bzw. Zuständigkeiten für Aufgaben bündeln. Eine explizite Differenzierung dieser beiden Aspekte und damit eine separate Beschreibung werden nicht betrachtet. Es werden vielmehr die in den jeweiligen Ansätzen existierenden Konzepte (wie Agent, Akteur, Rolle, Gruppe) mehrdeutig verwendet. Zudem zeigt sich, dass die Werkzeuge verschiedenartige Unterstützung zur Modellanalyse bieten (z.B. rollenspezifische Simulation oder Erstellung von Sichten auf das Modell zur Darstellung der Rolle-Aufgaben-Abhängigkeiten). Modellzusammenhänge können so gut erkannt werden, jedoch wird eine Unterstützung bei der Spezifikation der einzelnen Modelle und deren durch die Zuordnungen geschaffenen Interdependenzen nicht thematisiert. Wie eine solche werkzeugbasierte Unterstützung aussehen kann, wird im Folgenden erläutert.

3 Aufgaben

In diesem Kapitel werden die Konzepte der Aufgabenmodellierung soweit vorgestellt, wie sie für die folgenden Ausführungen relevant sind; im Wesentlichen betrifft dies das Konzept der Aufgabenhierarchie, mit der Aufgaben in Teil- bzw. Unteraufgaben untergliedert werden. Abbildung 1 zeigt, unter Verwendung der von MAD (Baron & Scapin 2006) eingeführten Symbole, ein Aufgabenmodell, das aus drei separaten Teilmodellen besteht: Die Aufgabe *online einkaufen* besitzt hier die Unteraufgaben *Verkäufer bewerten* und *Produkt kaufen*, wobei letztere ihrerseits weiter untergliedert ist. *online verkaufen* ist in genau zwei Unteraufgaben unterteilt, während *Website administrieren* lediglich aus einer einzelnen Aufgabe besteht.

Zum Verständnis des Aufgabenmodells seien noch folgende Erklärungen angemerkt: Die Ausführungsreihenfolge von *online einkaufen* und *online verkaufen* ist mittels der Angabe *No order* nicht festgelegt. Hingegen definiert *enabling* die sequentielle Ausführung, wobei die graphische Anordnung von links nach rechts die genaue Reihenfolge festlegt. *Elementary* wird für Aufgaben auf der untersten Ebene vergeben. *OPT* kennzeichnet optionale Aufgaben.

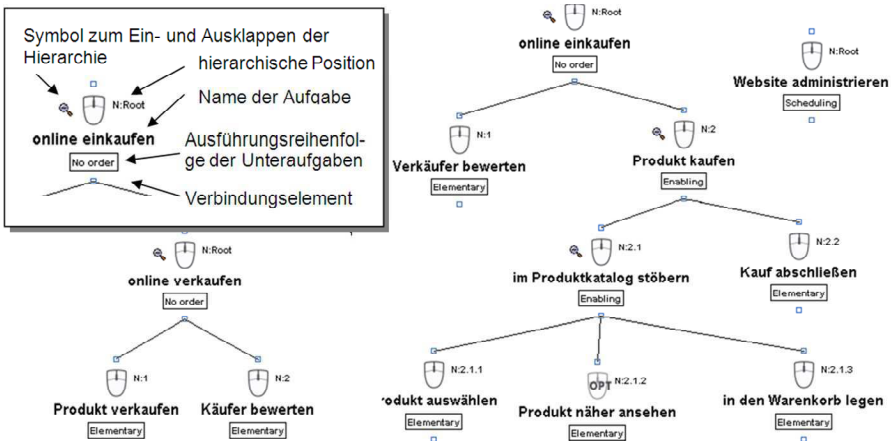


Abbildung 1: Aufgabenmodell

4 Rollen

Eine Rolle legt die Aufgaben fest, die ein Benutzer in dieser Rolle ausführen kann (bzw. muss). Die Rollen werden ebenfalls hierarchisch strukturiert. Während die Aufgabenhierarchie Kompositionsbeziehungen ausdrückt, beschreibt die Rollenhierarchie eine Vererbung von Privilegien.

4.1 Ermittlung der Modellstruktur

Das obige Aufgabenmodell soll nun so erweitert werden, dass jeder Besucher der Website den Produktkatalog durchstöbern kann, aber nur Personen in einer Rolle *Käufer* einen Kauf abschließen und Verkäufer bewerten können. Abbildung 2 zeigt auf der linken Seite das Modell nach dem Einfügen einer Rolle *jeder*, welcher wir beim Anlegen die Aufgabe *Produktkatalog durchstöbern* zugeordnet hatten. Jeder Rollengraph besitzt eine sogenannte minimale Rolle (*minRole*), deren Menge der Privilegien *priviledges*(*minRole*) leer ist, und eine maximale Rolle (*maxRole*), die stets im Besitz aller im gesamten Graphen vergebenen Privilegien ist. Beide Rollen werden im Algorithmus zur Konstruktion der Rollenhierarchie benötigt (Reitschuster 2008). Das Einfügen der ersten Rolle ist ein einfacher Schritt. Wie in dem Beispiel wird sie zwischen *minRole* und *maxRole* positioniert. Eine gerichtete Kante zwischen zwei Rollen drückt eine echte Teilmengenbeziehung aus. So gilt im Beispiel *priviled-*

ges(*minRole*) ⊂ *priviledges*(*jeder*) ⊂ *priviledges*(*maxRole*). Gemäß der allgemeinen Sprechweise (Guo 1999) wird in einer Beziehung *priviledges*(*r1*) ⊂ *priviledges*(*r2*) die Rolle r1 als Juniorrolle und r2 als Seniorrolle bezeichnet. Somit ist hier *jeder* senior zu *minRole* und junior zu *maxRole* und es gilt, dass *minRole* sowohl junior zu *jeder* als auch zu *maxRole* ist.

Wird nun einer Rolle eine Aufgabe zugeordnet, beinhalten die Privilegien auch alle Unteraufgaben. Wie bereits in (Bomsdorf & Szwillus 2004) ausgeführt, darf die Zuordnung zwischen Rollen und Aufgaben nicht beliebig erfolgen, sondern es müssen die jeweils existierenden hierarchischen Strukturen Berücksichtigung finden. Es dürfen keiner Juniorrolle mehr Aufgaben zugeordnet werden als einer ihrer Seniorrollen (dies wäre ein Zuordnungskonflikt). In einem solchen Fall würde *priviledges*(*Seniorrolle*) ⊂ *priviledges*(*Juniorrolle*) gelten, was jedoch im Widerspruch zur Definition der Rollenhierarchie steht.

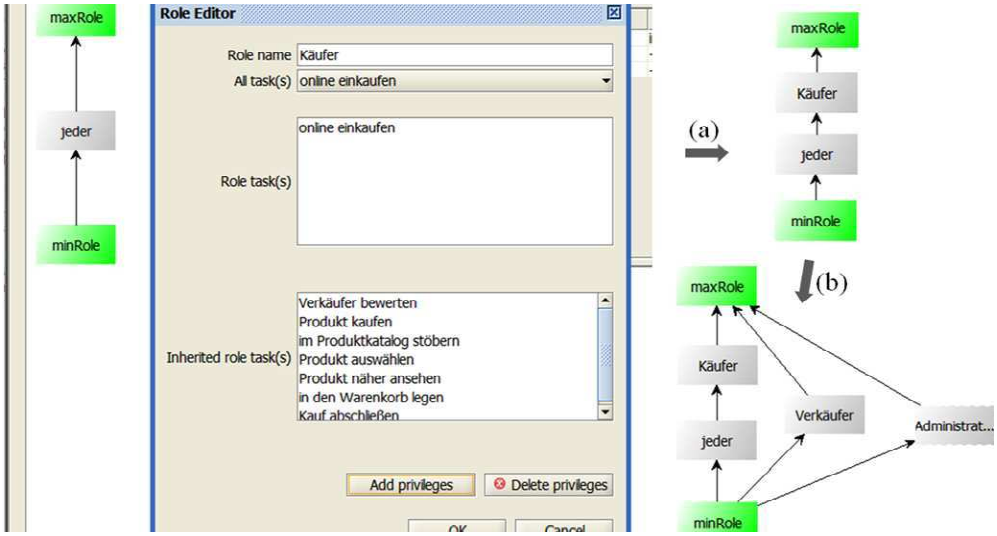


Abbildung 2: Entwicklung eines Rollenmodells

In unserem Beispiel fügen wir über den Rolleneditor (s. Abbildung 2) nun die Rolle *Käufer* ein und ordnen ihr die Aufgabe *online einkaufen* und damit alle entsprechenden Unteraufgaben zu (siehe *Inherited role task(s)* im Fenster des Rolleneditors). Als Ergebnis erhalten wir ein Rollenmodell (Pfeil (a)), in dem die *Käufer*-Rolle entsprechend der für sie definierten Privilegien und der bereits existierenden Hierarchie eingeordnet ist. *online einkaufen* ist eine Oberaufgabe von *Produktkatalog durchstöbern*, oder anders ausgedrückt, es gilt *priviledges*(*jeder*) ⊂ *priviledges*(*Käufer*), wodurch sich die Positionierung ergibt. Erläuterungen zu komplexeren Fällen finden sich in (Reitschuster 2008).

In ähnlicher Weise fügen wir die Rollen *Verkäufer* und *Administrator* hinzu, mit denen die Privilegien für *online verkaufen* und *Website administrieren* festgelegt werden. Da diese Aufgaben ohne „Verbindung“ zu den bisher betrachteten stehen, werden die Rollen jeweils separat in die Rollenhierarchie eingefügt (siehe Pfeil (b)).

4.2 Konflikte

Durch die algorithmische Berechnung der Position einer Rolle können keine Zuordnungskonflikte entstehen. Trifft der Modellierer im Rolleneditor eine Zuordnung zu Aufgaben, die zu einem solchen Konflikt führen würde, wird dies durch eine entsprechende Fehlermeldung angezeigt und vom Modellierer eine Korrektur gefordert. Der Modellierer hat zusätzlich die Möglichkeit, explizit Konfliktfälle (sog. Privilegienkonflikte und sog. Rollenkonflikte) zu definieren, die sowohl während der Rollen- als auch während der Aufgabenmodellierung geprüft werden. Wird z.B. ein Aufgabenmodell modifiziert, so wird in den hierdurch angestoßenen Prüfroutinen auch die Rollenhierarchie einbezogen. Wir könnten beispielsweise, wie in Abbildung 3 gezeigt, *Website administrieren* als Unteraufgabe von *online einkaufen* definieren. Der Modellierer wird darauf hingewiesen, dass dies Auswirkungen auf das Rollenmodell haben wird. In dem in der Abbildung gezeigten Szenario sind wir damit einverstanden (Aktivierung des *Ja*-Buttons) und erhalten als Ergebnis einen geänderten Rollengraphen. *Administrator* ist nun eine Juniorrolle von *Käufer*, d.h. ein Käufer darf alle Aufgaben eines Administrators durchführen und hat darüberhinaus weitere Privilegien. Um einen solchen Fall auszuschließen, können die Entwickler Rollenkonflikte festgelegen.

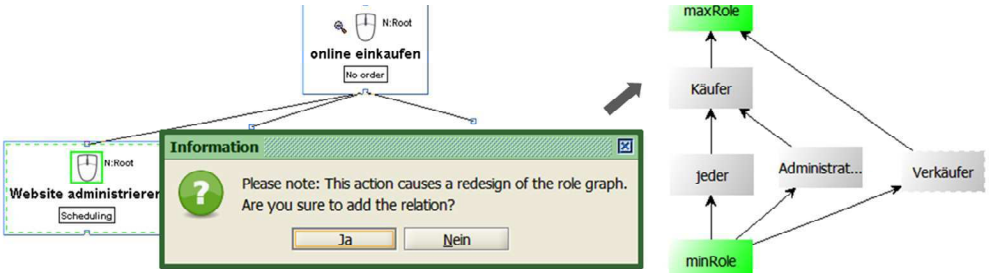


Abbildung 3: Modifikationen des Aufgabenmodells führen zur Änderung des Rollenmodells

Ein Rollenkonflikt wird für zwei Rollen definiert, zwischen denen damit keine Senior-Junior-Beziehung existieren darf. In unserem Beispiel fügen wir zwei Rollenkonfliktpaare ein (Abbildung 4, links oben) und schließen so gemeinsame Privilegien der Rollen *Verkäufer* und *Administrator* sowie der Rollen *Käufer* und *Administrator* aus. Nun führt der Versuch, *Website administrieren* als Unteraufgabe von *online einkaufen* festzulegen, zu einem Fehler.

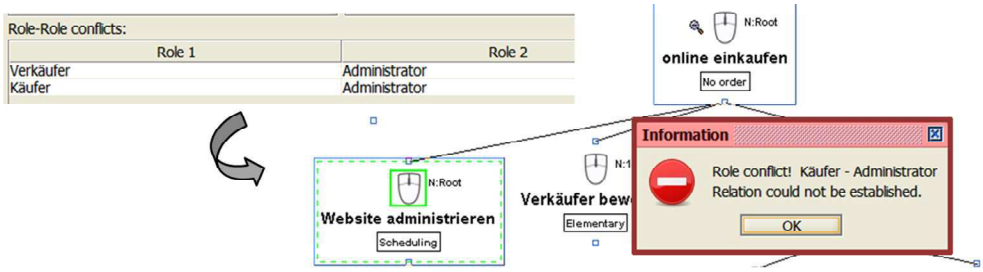


Abbildung 4: Rollenkonflikte: Definition und Effekt

Bestehen Konflikt nur zwischen einigen Privilegien von Rollen, dann ist es oft besser, Konflikte zwischen den Privilegien zu definieren. Ein Privilegienkonflikt spezifiziert zwei Aufgaben, die niemals gleichzeitig einer Rolle zugewiesen werden können.[1] In Abbildung 5 (oben) spezifizieren wir einen Privilegienkonflikt für die Aufgaben *Verkäufer bewerten* und *online verkaufen*. Der Rolleneditor in der Abbildung zeigt den Versuch, *Verkäufer* das Privileg zu geben, einen Verkäufer zu bewerten. Da der Rolle bereits die Aufgabe *online verkaufen* zugeordnet ist, kann sie das zusätzliche Privileg nicht erhalten. Der Editor reagiert mit einer Fehlermeldung und verhindert diese Modifikation. In ähnlicher Weise hat der definierte Privilegienkonflikt Auswirkungen auf die Aufgabenmodellierung. Die Aufgabe *online verkaufen* kann nicht als Unteraufgabe von *Verkäufer bewerten* spezifiziert werden (Abbildung 5, rechts). Ohne Konfliktdefinition würde *Verkäufer* aufgrund der automatischen Berechnung der Rollenhierarchie Juniorrolle von *Käufer*.

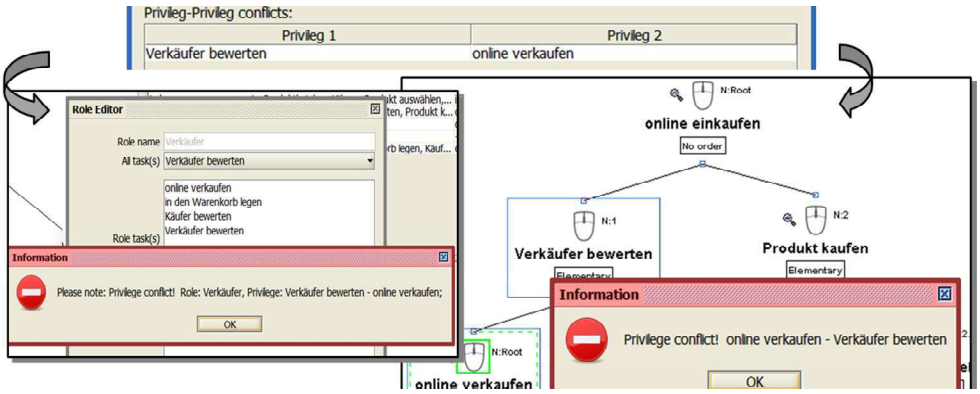


Abbildung 5: Privilegienkonflikt: Definition und Effekt

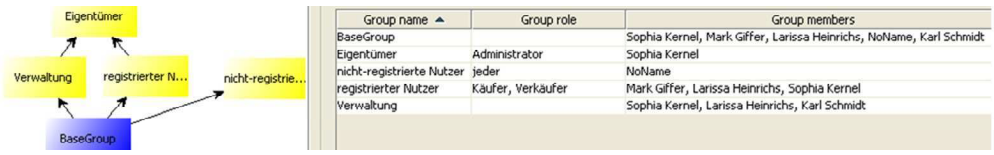
[1] Die einzige Ausnahme ist die maximale Rolle, welche alle im Modell vorkommenden Privilegien umfasst, aber ohnehin keinem Benutzer bzw. keiner Gruppe zugeordnet werden kann.

5 Benutzer, Gruppen und Objekte

In diesem Abschnitt werden drei weitere Konzepte (Benutzer, Gruppe und Objekt) kurz vorgestellt. Im Vordergrund stehen dabei die Zusammenhänge zum Aufgaben- und Rollenmodell. Mit dem Konzept Benutzer werden individuelle Personen modelliert, die anschließend verschiedenen Gruppen zugeordnet werden können. In der derzeitigen Version des Editors stehen zur Kennzeichnung beider Konzepte zunächst nur wenige Attribute zur Verfügung, u.a. Attribute für Benutzer- bzw. Gruppennamen, die jeweils eindeutig sein müssen.

Außerdem dürfen keine zwei Gruppen, auch wenn sie unterschiedliche Namen tragen sollten, identische Benutzermengen haben. Ähnlich zum Rollenmodell erfolgt auch die Definition der Gruppenstruktur über Teilmengenbeziehungen (als Polyhierarchie). Falls die Benutzer einer Gruppe g1 eine echte Teilmenge einer weiteren Gruppe g2 darstellen, d.h. falls *users*(g1) $\subset$ *users*(g2) gilt, werden g1 und g2 über eine Unter-Obergruppen-Beziehung miteinander verknüpft und befinden sich dadurch innerhalb der Struktur auf verschiedenen Ebenen. Das entstehende Gruppenmodell ist ein azyklischer Graph mit gerichteten Kanten. Eine Kante von einer Gruppe g1 zu einer Gruppe g2 verläuft immer von einer niedrigeren zu einer höheren Gruppengraphebene. Als eine weitere Eigenschaft ist festgelegt, dass die Verbindung zweier Gruppen entweder auf einer Kante oder auf einer Kantenfolge basiert (Verfahren der transitiven Reduktion (Guo 1999, 31)).

Diese Eigenschaften werden zur Konstruktion des Gruppengraphen herangezogen. Der Modellierer modifiziert das Modell, indem er für jede Gruppe die Benutzermenge (*users*) festlegt und Rollen zuordnet. Auf Basis dieser Informationen wird, in ähnlicher Weise wie bei der Konstruktion des Rollenmodells, die Struktur des Gruppenmodells automatisch gebildet (s. Beispiel in Abbildung 6). Insbesondere wird auch geprüft, ob ein aktueller Editiervorgang einen Konfliktfall impliziert. Wurde beispielsweise ein Rollenkonflikt festgelegt, dürfen die Rollen nicht gemeinsam einer Gruppe zugeordnet werden. Außerdem darf kein Benutzer, dem eine der beiden Rollen zugewiesen wurde, das Recht auf irgendein Privileg der anderen Rolle erhalten. Sollte dies zu Einschränkungen führen, müssen die Rollen im Graphen mit einer feineren Granularität gestaltet werden. Tritt durch eine Gruppen-Rolle-Zuordnung ein möglicher Konflikt auf, dann wird, wie auch schon bei der Rollenmodellierung, die Zuordnung nicht zugelassen und eine Fehlermeldung ausgegeben.



Group name	Group role	Group members
BaseGroup		Sophia Kernel, Mark Giffer, Larissa Heinrichs, NoName, Karl Schmidt
Eigentümer	Administrator	Sophia Kernel
nicht-registrierte Nutzer	jeder	NoName
registrierter Nutzer	Käufer, Verkäufer	Mark Giffer, Larissa Heinrichs, Sophia Kernel
Verwaltung		Sophia Kernel, Larissa Heinrichs, Karl Schmidt

Abbildung 6: Gruppengraph(links) und Gruppenüberblick (rechts)

Die Aufgabenobjekte werden in dem hier vorgestellten Editor derzeit in einfacher Weise berücksichtigt, indem ein eindeutiger Name, eine Beschreibung und der Typ festgelegt werden können. Anschließend können die Objekte verschiedenen Aufgaben zugeordnet werden.

Hierbei finden keine Prüfungen statt. Ein Benutzer erhält über seine Gruppenzugehörigkeit sowie die verknüpften Rollen und Aufgaben Rechte an den Objekten, wobei die Aufgaben beschreiben, welcher Art die Rechte sind. In unserem Beispiel können wir so ein Objekt *DVD* einfügen und der Aufgabe *Produkt kaufen* zuordnen.

6 Diskussion

Der grundlegende Modellierungsansatz der hier vorgestellten Arbeit besteht in der Definition der Privilegien einer Person, wobei unter der Berücksichtigung von deren Rolle (definiert über die Gruppenzugehörigkeit) festlegt wird, welche Aufgaben ausgeführt und, damit verbunden, auf welche Aufgabenobjekte zugegriffen werden kann. Der theoretische Unterbau hierzu kombiniert die eigenen Arbeiten zur Aufgabenmodellierung (Bomsdorf 2007) mit denen zur Konfliktbehandlung von (Guo 1999). Unsere Arbeit berücksichtigt damit, wie bereits in (Bomsdorf & Szwillus 2004) vorgeschlagen, Arbeiten aus dem Gebiet der Zugriffskontrolle. Wir gehen über unsere frühere Arbeit in zweifacher Weise hinaus – zum einem durch die Differenzierung organisatorischer Gruppen und Privilegienvergabe und zum anderen durch die werkzeugtechnische Unterstützung einer integrativen Modellierung.

Der Editor unterstützt die Erstellung der Modellstrukturen unter Berücksichtigung gegenseitiger Abhängigkeiten. Die Modellierer geben die gewünschten Zuordnungen ein, wovon ausgehend dann die sich ergebenden Hierarchien berechnet werden. Beispielsweise können die Modellierer Rollen und Aufgaben einander zuordnen, ohne zuerst die damit erforderlichen Umstrukturierungen der Hierarchien vornehmen zu müssen. Stattdessen ordnet der Editor unter Berücksichtigung resultierender und explizit definierter Konfliktfälle Rollen und Aufgaben hierarchisch an. Diese Vorgehensweise entspricht der Tatsache, dass die Modellhierarchien sich aus den verschiedenen Zuordnungen und deren Bündelungen ergeben.

Die Erstellung der Hierarchien und die Überprüfungen auf mögliche Konflikte basieren auf den mit dem Meta-Modell festgelegten Regeln und zusätzlich auf den durch die Entwickler definierten Konfliktfällen. Die angezeigten Fehlermeldungen resultieren systematisch aus diesen, so werden die ursächlichen Konfliktpaare im Text angezeigt.

In einem umfangreichen Modell ergeben sich verschiedenartige Fragestellungen, etwa, wer was machen darf oder welche Aufgaben ein bestimmtes Objekt benötigen. Zur entsprechenden Überprüfung erlaubt unser Werkzeug verschiedenartige Abfragen bzw. Sichten auf die Modellverknüpfungen. Generell kann der Entwickler verschiedene Abfragen formulieren, in dem er interaktiv entsprechende Filter festlegt. Die Ergebnisse werden als Diagramme analog zu den während der Modellierung verwendeten Darstellungen angezeigt.

Die Werkzeugimplementierung hat derzeit den Status eines Prototyps, der auch Import- und Exportfunktionen zum Austausch der Modelle mit anderen Werkzeugen (auf Basis von XML) besitzt.

Literaturverzeichnis

Baron, M. & Scapin, D. (2006) K-MADe User Manual, http://kmade.sourceforge.net, (21.3.2009).

Bomsdorf, B. (2007) The WebTaskModel Approach to Web Process Modelling. In *Proceedings of the 6th TAMODIA Workshop*, Springer, Lecture Notes in Computer Science 4849, 240-253.

Bomsdorf, B. & Szwillus, G. (2004) Rollen- und aufgabenbasierte Webmodellierung. In *Mensch & Computer 2004: Allgegenwärtige Interaktion*, 115–126.

Guo, Y. (1999) User/Group Administration for RBAC. The University of Western Ontario, May 1999.

García, J., Lemaigre, C., Calleros, J. & Vanderdonckt, J. (2008) Model-Driven Approach to Design User Interfaces for Workflow Information Systems. In *Journal of Universal Computer Science*, 14 (19), 3160-3173.

Giese, M., Mistrzyk, T., Pfau, A., Szwillus, G. & von Detten, M. (2008) AMBOSS: A Task Modeling Approach for Safety-Critical Systems. In *Proceedings of the 7th TAMODIA Workshop*, 25-26.

Limbourg, Q., Vanderdonckt, J. (2003) Comparing Task Models for User Interface Design. In Diaper, D., Stanton, N., (Hrsg.): *The Handbook of Task Analysis for Human-Computer Interaction*. Lawrence Erlbaum Associates.

Lu, S., Paris, C., Vander Linden, K. (2002) Tamot: Towards a Flexible Task Modeling Tool. In *Proceedings of Human Factors*, 878-886.

Paternò, F., Santoro, C. & Tahmassebi, S. (1998) Formal Models for Cooperative Tasks: Concepts and an Application for En-Route Air Traffic Control. In *5th Int. Workshop on Design, Specification, and Verification of Interactive Systems, DSV-IS '98*, Springer-Verlag, Abingdon, 71-86.

Reitschuster, A. (2008) *Realisierung eines Editors zur integrativen Gruppen-, Rollen- und Aufgabenmodellierung*. Masterarbeit an der FernUniversität in Hagen.

Stary, C. (2005) Role-Adapted Access to Medical Data: Experiences with Model-Based Development, In *Universal Access in Health Telematics*, 224-239.

Stuart, J. & Penn, R. (2004) TaskArchitect: taking the work out of task analysis. In *Proceedings of the 3rd Annual Conference on Task Models and Diagrams, TAMODIA* 2004, 145–154.

van Welie, M., van der Veer, G.C. & Eliëns, A. (1998) Euterpe - Tool support for analyzing cooperative environments. In *Proceedings of the Ninth European Conference on Cognitive Ergonomics.*

Kontaktinformationen

Birgit Bomsdorf

Angewandte Informatik
Hochschule Fulda
D-36039 Fulda
Marquardstraße 35
Tel.: +49 (0) 661-9640-327
Fax: +49 (0) 661-9640-349
E-Mail: birgit.bomsdorf@hs-fulda.de

Andreas Reitschuster

AUDI AG
I/FP-52
D-85045 Ingolstadt

E-Mail: andreas.reitschuster@audi.de