Universität-Gesamthochschule Paderborn

Seminararbeit

Realisierung von Konzepten für Message-Objekte in Lotus Notes R5

Prof. Dr. Ludwig Nastansky
Betreuer: Dipl. Wirt.-Ing. Thomas Bruse

Wintersemester 1999/2000

vorgelegt von:

Harald Gabriel
Archenholdweg 2a
33100 Paderborn

Studiengang Wirtschaftsinformatik

Matrikelnummer: 3371533

Inhaltsverzeichnis

INHALTSVERZEICHNIS I

ABBILDUNGSVERZEICHNIS II

1 SZENARIO 1

1.1 EINGRENZUNG MESSAGE-OBJEKT 1

1.2 NAVIGATIONSKONZEPT ROADMAP 2

2 PROBLEMSTELLUNG 4

2.1 NUTZUNG VON ROADMAPS 4

2.2 VERARBEITUNGSLOGIKEN 4

2.3 ERWEITERTE NAVIGATIONSKONZEPTE 5

2.4 LÖSUNGSANSATZ UNTER VERWENDUNG VON OUTLINES 6

3 REALISIERUNG 8

3.1 VERWENDUNG EINER KONFIGURATIONSUMGEBUNG 8

3.1.1 Vordefinierung von Methoden 8

3.1.2 Benutzung der Umgebung 9

3.2 ÜBERTRAGUNG DER KONFIGURATION IN OPERATIVE STRUKTUREN 9

3.2.1 Gestaltung des Überführungsprozesses 10

3.2.2 Realisierungsformen der Agenten 10

3.2.2.1 Präsenz in allen Zieldatenbanken 10

3.2.2.2 Präsenz des Agenten in einer Datenbank 11

3.3 ÜBERTRAGUNG VON OPERATIVEN STRUKTUREN IN DIE KONFIGURATIONSUMGEBUNG 11

3.4 VORTEILE DER VORGESTELLTEN REALISIERUNG 12

3.4.1 Flexibles Navigationskonzept 12

3.4.2 Dynamische Anpassung beliebiger „Intelligenzen“ 13

3.4.2.1 Abstrahierung von der technischen Realisation 13

3.4.2.2 Vordefinierung gebräuchlicher Intelligenzen 13

3.4.2.3 Direkter Entwurf von Aktionenlogik weiterhin möglich 14

3.4.3 Mobilität der Konfigurationsumgebung 14

3.5 UMSETZUNG IN EINEN PROTOTYPEN 14

4 AUSBLICK 16

5 ZUSAMMENFASSUNG 18

LITERATURVERZEICHNIS ..... A

Abbildungsverzeichnis

Abb. 1: Beispiel einer Roadmap als Navigationsstruktur 3
Abb. 2: Bisherige Nutzung der Verarbeitungsintelligenzen durch Selbstdefinierung der Aktionen 5
Abb. 3: Reduzierung der Komplexität der Verarbeitungsmethoden durch Angabe von Parametern 6
Abb. 4: Ein Beispiel für eine Outline als instanziiertes Objekt 7
Abb. 5: Prozeß der Erstellung von Navigationsstrukturen über Konfigurationsdokumente 12
Abb. 6: Beispiel für ein Konfigurationsdokument (Ausschnitt) 15

1 Szenario

Ein wichtiger Bestandteil heutiger groupwarebasierter Datenbanksysteme sind message-objekt-orentierte Navigationsstrukturen auf Basis sogenannter Roadmaps. Sie erlauben dem Benutzer die Orientierung in den dort vorherrschenden nichtlinearen Hyperlinkstrukturen und stellen oftmals einen Einstiegspunkt zur Informationsgewinnung eines durch das Grundgerüst der Roadmap spezifizierten Themenkomplexes dar. Zudem bieten sie neben reinen Verzweigungsmechanismen im virtuellen Raum die Möglichkeit, dem Benutzer vom Kontext der Roadmap abhängige Methoden zur Verfügung zu stellen, die einerseits interaktiv genutzt werden, aber auch eine Erweiterung der traditionellen Verzweigungsstrukturen darstellen können. Die Definierung und Wartung solcher Methoden sind in der Praxis nicht unproblematisch; oftmals werden diese Tätigkeiten von Spezialisten übernommen.

In dieser Arbeit sollen die Begrifflichkeiten des Themenbereiches roadmapbasierter Navigationsstrukturen eingegrenzt und die dadurch entstehenden Probleme aufgezeigt werden. Desweiteren ist zu klären, inwieweit diese Navigationsstrukturen mittels message-objekt-orientierter Konzepte erweitert werden können. Anschließend wird ein mögliches Realisierungskonzept vorgestellt.

1.1 Eingrenzung Message-Objekt

Message-Objekte sind grundlegende Aufbauelemente computergestützter, verteilter Informations- und Kommunikationssysteme (IKS). Dabei sind Message-Objekte als intelligente elektronische Dokumente zu verstehen, die nicht nur strukturierte Daten und Informationen passiv aufnehmen, sondern auch aktiv verarbeiten können. Die Bandbreite der in Message-Objekten gehaltenen Daten und Informationen reicht von strukturierten, feldformatierten Daten bis hin zu reichhaltigen, multimedialen Informationen beliebiger Art. Die sogenannte „Intelligenz“ der Message-Objekte wird durch die den Objekten eingebetteten Methoden erreicht. Diese Methoden können grundsätzlich alle üblichen Verarbeitungsprozeduren für betriebswirtschaftliche Entitäten enthalten. Hierbei handelt es sich vor allem um spezifische Informationsverarbeitungsaufgaben für die dem jeweiligen Dokument enthaltenen Daten.

Message-Objekte dienen als Kommunikationselemente in Unternehmungen und Organisationen. Innerhalb einer Kommunikation werden die Daten, Informationen und Methoden der Message-Objekte unter den Beteiligten leistungsfähig ausgetauscht. Das Spektrum der Kommunikation reicht hier von unstrukturierter ad-hoc-Kommunikation bis hin zu vordefinierten Kommunikationsabläufen. Die unstrukturierte Kommunikation erfolgt hierbei zumeist über den Austausch individueller elektronischer Nachrichten der Kommunikationspartner (e-mail). Erweitert wird dieses Modell um die Nutzung der Dokumente als Kommunikationselemente in verteilten Datenbankstrukturen, auf denen die Kommunikationspartner gemeinsam zugreifen und operieren. Die zunächst an einem Ort gehaltenen Informationen werden durch kontinuierliche Replikationsmechanismen über das gesamte Kommunikationsnetz verteilt. Als Besonderheit von Message-Objekte sind diese Intelligenzen, sofern vorhanden, idealerweise integraler Bestandteil des Objektes an sich und nicht an übergeordnete Aggregationsstrukturen gebunden. Damit wird eine uneingeschränkte Nutzung dieser Objekte im gesamten Informations- und Kommunikationssystem erreicht.

(vgl. Nastansky, Ludwig: Message-Objekte und Team-Kommunikation – Systembausteine für die Unternehmensführung in neuen Organisationsformen, 1998, S. 3-11)

1.2 Navigationskonzept Roadmap

Informationen werden in Datenbanken der Groupwareplattform Lotus Notes, im folgenden verkürzt Datenbanken genannt, in Dokumenten gespeichert. Den Dokumenten liegt keine lineare Aggregationsstruktur zugrunde, vielmehr werden sie in multidimensional kategorisierbaren Kollektionen, sog. Ansichten, gehalten. Die genaue Spezifikation der enthaltenen Menge von Dokumenten erfolgt über dynamisch angewandte Verarbeitungsmethoden. Somit ist ein Dokument einer Datenbank immer dann in einer Ansicht enthalten, wenn es den Verarbeitungsmethoden dieser Kollektion genügt.

Diese Ansichten bilden die Grundlage zur Navigation in den Datenbanken; zur individuellen, kontextbezogenen Aggregation von Dokumenten sind diese Strukturen nicht ausreichend. Ergänzt werden diese Funktionalitäten durch das Konzept der

Roadmaps. Sie führen mittels Verzweigungsmechanismen, Links genannt, durch Dokumente, bilden somit kontextbezogene Querverweise zu weiterführenden oder themenverwandten Informationen und stellen auch unter Benutzung von Rückverweisen zu Informationseinstiegspunkten eine mächtige Navigationsmöglichkeit für die Benutzer dar. Typisch für Roadmaps ist eine strukturierte Zusammenfassung dieser Querverweise zu einer Art Gliederung, welche auf den individuellen Kontext des Dokumentes bezogen, als Quelle für die weitere Informationsgewinnung anzusehen ist. Die oben angesprochenen Querverweise sind in den Datenbanken zumeist als mit textuellen oder graphischen Informationselementen angereicherte Links auf Dokumente, Ansichten, Datenbanken oder externe Quellen, wie beispielsweise das Internet, realisiert.

Weiterhin können die Roadmaps „Intelligenzen“ beinhalten, die in Analogie zu den unter 1.1 angeführten Message-Objekten als in das jeweilige Dokument eingebettete Verarbeitungsmethoden realisiert sind. Diese Funktionalitäten werden über Schaltflächen innerhalb der Roadmap aktiviert. Die Möglichkeiten verschiedener, dadurch realisierbarer Aktionen sind sehr vielfältig. Beispiele für solche Methoden sind das Starten externer Applikationen, Ausführung von Methoden zum Zwecke der Informationsgewinnung (Aktivierung von Volltextsuchmechanismen in einer oder mehreren Datenbanken) oder auch die Erweiterung der bereits vordefinierten Linkmechanismen. Abbildung 1 zeigt eine bespielhafte Roadmap, wie sie auch Umfeld des Groupware Competence Centers (GCC) der Universität Paderborn eingesetzt wird.

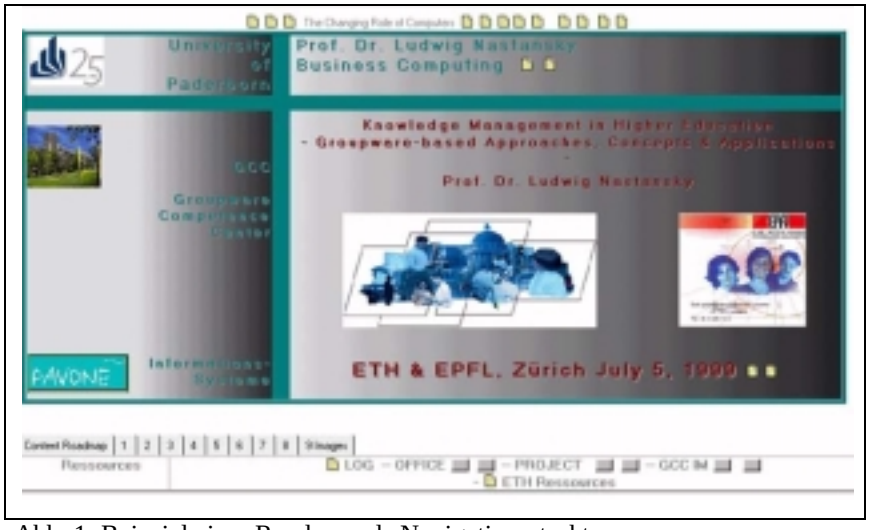

Abb. 1: Beispiel einer Roadmap als Navigationsstruktur

2 Problemstellung

Die in 1.2 dargestellte Roadmap stellt ein flexibles Navigationselement dar, das von den Endbenutzern einer Datenbankapplikation dokumentenbasiert erzeugt, gewartet und genutzt werden kann. Somit ist es nicht nur den mit Designerrechten ausgestatteten Datenbankbenutzern möglich, diese Strukturen dynamisch bei Veränderungen des jeweiligen Informationskontextes anzupassen. Dies ist besonders effizient, da für diese Modifikationen keine zeitintensiven Koordinationen mit Desingnerrechten ausgestatteten Benutzern erforderlich sind.

2.1 Nutzung von Roadmaps

Problematischer ist hingegen die Verwendung von Verarbeitungsmethoden in solchen Roadmaps. Die Anwendung solcher Methoden stellt keine besonderen Anforderungen an den jeweiligen Benutzer; das Erstellen und Modifizieren dieser Funktionalitäten erfordert jedoch spezifisches Fachwissen in Bezug auf Implementierung von Methoden in den vorherrschenden Datenbankapplikationen. So muß jede abstrakte Funktionalität in eine für die Datenbankapplikation verständlich interpretierbare Sprache, beispielsweise die Formelsprache in Lotus Notes, implementiert werden. Dies führt gerade bei komplexeren zu definierenden Aktionen zu erhöhten Koordinations- und Kooperationsaufwand mit spezialisierten Designern.

Darüber hinaus besitzen die Roadmaps kein Konzept zur dynamischen Anpassung beliebiger Intelligenzen; Modifikationen der Funktionalitäten können nur durch die Änderung der realisierten Implementierung erreicht werden. Dies kann in Analogie zur Erstellung der Aktionen nur manuell und sukzessiv durch den Benutzer erfolgen. Bei umfangreichen oder auch vielen einfachen Aktionen führt dies ebenfalls zu zeitlichen Verzögerungen des Aktualisierungsprozesses.

2.2 Verarbeitungslogiken

Verarbeitungslogiken bedienen sich statischen und dynamischen Komponenten. Das Grundgerüst (und somit der statische Teil) einer Verarbeitungslogik besteht aus der Sprache mit ihren semantischen und syntaktischen Konstrukten, die zur Basisdefinierung der Logik an sich erforderlich sind. Der dynamische Teil einer Verarbeitungslogik entspricht dem individuellen Aspekt derselben. Er beinhaltet die „Parameter“ dieser Funktionalität. Als Beispiel sei eine Methode angenommen, die einen Link zu einer genau spezifizierten Kategorie einer Ansicht in einer bestimmten

Datenbank darstellt. Die variablen Bestandteile sind hierbei die Zieldatenbank, die Ansicht und die Kategorie. Mit Änderung dieser Parameter ändert sich die Zielumgebung der Logik, die eigentliche Funktionalität des Linkmechanismus bleibt hiervon unberührt. Es können also die Parameter einer Verarbeitungslogik mit feldbasierten Daten eines Dokuments verglichen werden.

2.3 Erweiterte Navigationskonzepte

Bei der Erstellung von Logiken gleicher Art (d.h. nur der dynamische Teil einer Logik ist unterschiedlich) muß sukzessiv die gesamte Logik, also der statische und dynamische Teil, implementiert werden (siehe Abb.2). Viel effizienter wäre es jedoch, nur die Parameter der Logik angeben zu müssen und die Grundfunktionalität als vordefinierte Aktionen zu realisieren.

Abb. 2: Bisherige Nutzung der Verarbeitungsintelligenzen durch Selbstdefinierung der Aktionen

Hierzu ist es sinnvoll, gebräuchliche Aktionen vorzudefinieren, aus denen der Benutzer dann seine gewünschte Funktionalität auswählen kann. In einer Konfigurationsumgebung braucht der Benutzer nur noch die Parameter seiner ausgewählten Logik (z.B. feldbasiert) angeben und definiert somit eine neue Methode, die dann in die von ihm gewünschte Navigationsstruktur überführt werden kann. Dadurch haben Endbenutzer die Möglichkeit, komplexe Methoden innerhalb von Navigationsstrukturen zu erstellen, ohne sich vorher entsprechendes Fachwissen von der ursprünglich benötigten Logiksprache aneignen zu müssen. Durch die Abstrahierung von der technischen Realisation wird für die Benutzer die Komplexität solcher Methoden reduziert; dies bedeutet einen Effizienz- und Zeitgewinn gegenüber den „herkömmlichen“, unter 2.1 beschriebenen Ansätzen, da eine Kommunikation und Koordination mit entsprechend designtechnisch geschulten

Benutzern entfällt. Abbildung 3 zeigt für diese dynamisch anpassbaren „Intelligenzen“ eine mögliche Repräsentationform.

Abb. 3: Reduzierung der Komplexität der Verarbeitungsmethoden durch Angabe von Parametern

2.4 Lösungsansatz unter Verwendung von Outlines

Eine Möglichkeit des Ansatzes, dynamische Intelligenzen in Navigationsstrukturen innerhalb von Datenbankapplikationen auf Basis der Groupwareplattform Lotus Notes zu verwenden, besteht in der Verwendung von Outlines (Abb. 4). Dies sind flexible Baumstrukturen, die einerseits durch Verwendung von Linkmechanismen ebenfalls die Verzweigungsfunktionalität einer Roadmap beinhalten, sowie auch Verarbeitungsmethoden aufnehmen können. Dabei kann für jedes Element der Outline ein individueller Verzweigungsmechanismus oder eine Verarbeitungsmethode definiert werden. Die Besonderheit der Outlines gegenüber anderen Navigationskonzepten besteht in der Möglichkeit, dynamische Funktionaliäten, wie unter 2.3 beschrieben, zu realisieren. Dadurch wird es möglich, flexibel die „Intelligenzen“ solcher Strukturen zu erstellen und zu warten, ohne als Benutzer jedoch die gesamte Implementierung einer spezifischen Funktionalität durchführen zu müssen. Jedoch ist zu beachten, daß eine „Übertragung“ der Aktionen aus einer Konfigurationsumgebung mit vordefinierten Aktionen in eine Outline einen erst zu implementierenden Mechanismus darstellt; er gehört standartmäßig nicht zur Kernfunktionalität der Outline. Eine solche Realisierung wird im Kapitel 3 vorgestellt.

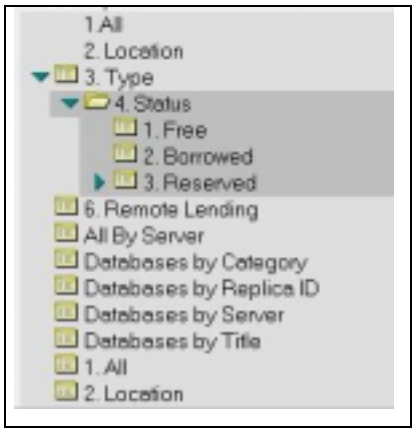

Abb. 4: Ein Beispiel für eine Outline als instanziiertes Objekt

Die Outline ist innerhalb einer Datenbank ein zentrales Datenbankelement. In einer konkreten Datenbank können beliebig viele solcher Designelemente vorhanden sein. Die eigentliche Benutzung dieser Outlines erfolgt durch von diesem Element instanziierten Objekte. Jede Instanz kann gestalterisch, jedoch nicht funktional, individuell angepaßt werden. Somit können mehrere Ausprägungen derselben Outline unterschiedliche Benutzerrepräsentationen besitzen. Ein weiterer Vorteil liegt in der automatischen Aktualisierung der Instanzen bei der Änderung des zentralen Outline-Objektes. Dies führt gerade bei Mehrfachverwendung der Instanzen innerhalb einer Datenbank zu einer Arbeitsersparnis und Konsistenzerhaltung dieser Navigationsstrukturen.

Der entscheidene Nachteil bei der Verwendung von Outlines liegt jedoch in der Natur der Outline, ein zentrales Designelement zu sein. Dadurch ist die Erstellung und Wartung outlinebasierter Navigationsstrukturen Datenbankdesignern vorbehalten.

3 Realisierung

In diesem Kapitel wird aufgezeigt, wie mit Hilfe von Message-Objekten Outlines über eine Konfigurationsumgebung mobil und von Endbenutzern erstellt und gewartet werden können. Der vorgestellte Lösungsvorschlag bietet die Möglichkeit zur Nutzung der Navigationsstruktur im gesamten Informations- und Kommunikationsystem einschließlich deren Verteilung via e-mail und Replikationsmechanismen. Weiterhin können die den Navigationselementen eingebetteten Methoden flexibel und dynamisch erstellt und angepaßt werden; somit werden die Vorteile der Roadmap und Outline verbunden.

3.1 Verwendung einer Konfigurationsumgebung

Die grundsätzliche Idee in der vorgestellten Lösungsmöglichkeit besteht darin, dem Benutzer zunächst eine Konfigurationsumgebung zur Verfügung zu stellen, mit deren Hilfe eine Navigationsstruktur erstellt und gewartet werden kann. Konkret bedeutet dies unter Verwendung von Lotus Notes, daß der Benutzer feldbasiert die Eigenschaften seiner gewünschten Navigationsmöglichkeiten- und methoden auf in einer für ihn verständlichen Art und Weise angeben und diese dann in eine operative Datenbankumgebung einbinden kann. Die jeweiligen Eigenschaften der Navigationsstruktur sollen innerhalb der Konfigurationsumgebung dokumentenbasiert gespeichert werden. Dabei entspricht immer eine Gruppe von Feldern einem Navigationselement; mehrere Elemente werden zu einem Navigationskonstrukt zusammengeführt.

3.1.1 Vordefinierung von Methoden

Ein wesentlicher Bestandteil der Konfigurationsumgebung ist die Vordefinierung oft genutzter Funktionalitäten, die als Verarbeitungsmethoden in die Navigationsstruktur eingebunden werden können. Die Vordefinierung der Aktionen muß designtechnsich einmalig in die Konfigurationsumgebung eingepflegt werden, d.h. es müssen erst Mechanismen geschaffen werden, die aufgrund von Feldinhalten Aktionen in Makrosprache erzeugen und diese Makros als Feldinhalte speichern. Als vorbereitende Tätigkeit hierauf sollte der Designer zusammen mit den Endbenutzern analysieren, welche Aktionen benötigt werden. Anschließend muß dieser festlegen, welche Aktionen er tatsächlich implementieren wird (womöglich sind einige

Aktionen technisch nicht zu realisieren). Durch diese Vordefinierung wurde der statische Teil der Verarbeitungsmethoden bereits geschaffen (vgl. 2.2).

3.1.2 Benutzung der Umgebung

Nach Beendigung des Vordefinierungsprozesses können die Benutzer die so geschaffenen Methoden anwenden. Mittels der Konfigurationsumgebung erstellt der Benutzer ein Dokument, im weiteren Konfigurationsdokument genannt, welches den Aufbau der zu schaffenden Navigationsstruktur enthält. Dabei handelt es sich zum einen um die Gliederung der einzelnen Elemente als auch um die genaue Beschreibung der einzelnen Verzweigungsmechanismen oder Verarbeitungsmethoden. Jedes Verzweigungselement besteht im wesentlichen aus der Definition der Verzweigungsparameter. Dies sind insbesondere die Art der Verzweigung (beispielsweise ein Link auf ein Dokument) und die eindeutige Beschreibung des Zielobjektes. Die in den Elementen enthaltenden Verarbeitungsmethoden werden über Auswahl der gewünschten Aktion sowie der Angabe der dynamischen Komponenten (vgl. 2.2) der Funktionalität erstellt.

Jeder anzugebende Parameter erfolgt über feldbasierte Eingaben und wird in einer für Benutzer verständlichen Repräsentationsform angezeigt. Dadurch reduziert sich gerade bei der Definierung der dynamischen Methoden die Komplexität für den Benutzer, er benötigt keine Kenntnisse über technische Realisierungsformen der gewüschten Aktionen.

3.2 Übertragung der Konfiguration in operative Strukturen

Mittels Konfigurationsdokument wird die zu erstellende Navigationsstruktur definiert. Nun sind noch Konzepte zur Überführung dieser Daten in eine reale Umgebung notwendig. Die Zielstruktur ist als Outline realisiert; sie bietet in Lotus Notes R5 die alleinige Möglichkeit zur dynamischen externen, von der technischen Realisation abstrahierten Erstellung von Verarbeitungsmethoden. Da die Erstellung von Outlines nur Datenbankdesignern vorbehalten ist (vgl. 2.4), müssen zunächst Konstrukte zur automatischen Überführung der definierten Navigationsstruktur in eine Outline vorhanden sein.

3.2.1 Gestaltung des Überführungsprozesses

Zur Durchführung dieses Vorhabens wird sich Agenten bedient. Dies sind durch Benutzerinteraktion aktivierte oder periodisch gesteuerte Prozesse, die in einer Datenbankumgebung wiederkehrende Aufgaben automatisch durchführen.

Im dem konkreten Fall der Überführung einer mit einem Konfigurationsdokument spezifizierten Outline in die gewünschte Zieldatenbankumgebung liest der Agent bei Aktivierung die in Feldern gehaltenen Parameter für die Navigationsstruktur aus und erstellt eine neue bzw. modifiziert eine bereits vorhandene als zentrales Datenbankobjekt realisierte Outline. Ist die Outline dann in der Datenbank enthalten, kann sie als instanziiertes Objekt in beliebige Dokumente eingefügt und gestalterisch individuell angepaßt werden. Die Endbenutzer der Datenbank können nun die so geschaffene Navigationsstruktur nutzen.

Der Agent dient ebenfalls der Modifikation einer bereits vorhandenen, durch ein Konfigurationsdokument spezifizierten Outline. Hierbei brauchen nur die entsprechenden Einträge des Konfigurationsdokumentes geändert und der Agent neu gestartet werden. Die durchgeführten Änderungen werden zu der Outline übertragen. Die instanziierten Objekte der Zielumgebung erben automatisch die so neu gestaltete Struktur von dem zentralen Designelement.

3.2.2 Realisierungsformen der Agenten

Eine Outline stellt ein zentrales Datenbankelement dar. Somit ist eine Änderung dieser Objekte nur Datenbankdesignern vorbehalten. Damit die Agenten die Navigationsstrukturen erstellen und modifizieren können, benötigen sie generell Designerrechte in der Zieldatenbank, in der sie die Navigationsstruktur erstellen sollen. Darüber hinaus bieten sich mehrere Möglichkeiten zur Ausgestaltung ihres Überführungs- bzw. Aktualisierungsprozesses an.

3.2.2.1 Präsenz in allen Zieldatenbanken

Eine Realisierungsform der beschriebenen Agenten besteht seiner Existenzforderung in allen Zieldatenbanken. Konkret bedeutet dies, daß in jeder Datenbank, in der eine Navigationsstruktur gefordert wird, ein Agent existiert, der die Outlines aus dem Konfigurationsdokument ausliest und erstellt. In diesem Fall benötigt dieser Agent nur die Designerrechte auf der Zieldatenbank, dessen Bestandteil er selber ist.

Praktischerweise könnte der Agent einen Teil einer Standarddatenbankapplikation darstellen. Da oftmals in realen groupwarebasierten Datenbankumgebungen häufig solche Standardapplikationen die Grundlage für spezifische, dem Zwecke der gewöhnlichen Nutzung dienenden Weiterentwicklungen einer Datenbank darstellen, kann somit schon die Existenz in den Zielumgebungen gewährleistet werden, ohne das ein zusätzlicher Mehraufwand für die Bereitstellung der Agenten erforderlich wird.

Der Agent wird jeweils bei der Überführung oder Modifikation einer Outline aus der Zieldatenbank heraus aktiviert. Es wird somit möglich, den Benutzern dieser Funktionalitäten die Übertragung der Navigationsstruktur auf andere Datenbanken zu verwehren; in Umgebungen, in denen der Agent nicht die geforderten Rechte besitzt, hat der Benutzer keine weitere Möglichkeit, eine Outline zu erstellen.

3.2.2.2 Präsenz des Agenten in einer Datenbank

Eine andere Möglichkeit zur Realisierung der Agenten ist die Haltung des Agenten in nur einer Datenbank, typischerweise in der Konfigurationsdatenbank. Hierbei benötigt der Agent Designerrechte in allen Zieldatenbanken. Dabei muß über die Zugriffskontollliste jeder Zieldatenbank dem Agenten die entsprechenden Rechte gewährt werden. Dies erfordert das Signieren des Agenten mit einem Zertifikat, welches „universelle Designrechte“ besitzt. Dies kann insbesondere bei einer Verwendung heterogener Datenbankapplikationen sinnvoll sein, da diese nun nicht mehr designtechnisch verändert werden müssen.

Das Verhindern des Aktualisierungs- und Modifikationsprozesses des Agenten ist nicht mehr für einzelne Benutzer möglich; lediglich kann nur die Ausführung des Agenten in der Zieldatenbank für alle Benutzer verhindert werden.

3.3 Übertragung von operativen Strukturen in die Konfigurationsumgebung

Eine Erweiterung der angesprochenen Funktionalitäten wird durch das Übernehmen bereits vorhander Outline-Strukturen in die Konfigurationsumgebung erreicht. Es ist nunmehr möglich, operative Navigationselemente auszulesen und für andere Zielumgebungen zu benutzen. Dadurch ist der Benutzer in der Lage, effizient eine gewünschte Struktur aus der Vorlage bereits existierender Elemente zu erzeugen.

Dadurch wird die Entwicklungszeit von neuen Outlines verkürzt. Ferner werden nach Erzeugen der Navigationselemente die dazugehörigen Konfigurationsdokumente nicht mehr benötigt. Im Fall einer späteren Wiederverwendung der Navigationsstruktur können jederzeit die Navigationselemente wieder aus der operativen Umgebung ausgelesen werden.

Abbildung 5 zeigt eine schematische Darstellung des Prozesses zur Erstellung von outline-basierten Navigationselementen unter Verwendung der bisher unter Kapitel 3 erwähnten Realisierungsmöglichkeiten.

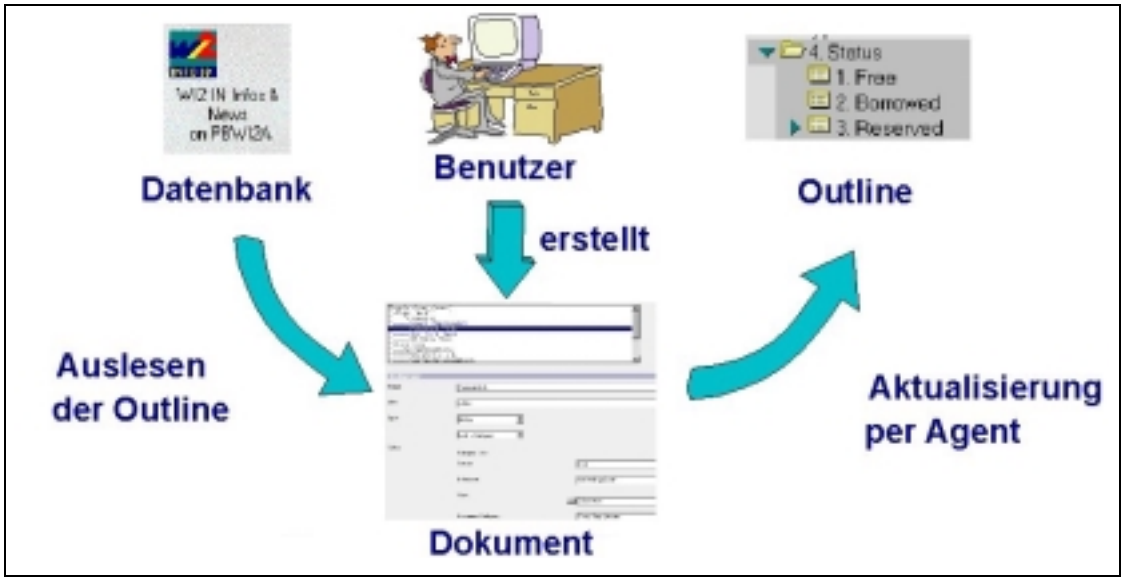

Abb. 5: Prozeß der Erstellung von Navigationsstrukturen über Konfigurationsdokumente

3.4 Vorteile der vorgestellten Realisierung

Die vorgestellte Realisierungsmöglichkeit bieten vielfältige Vorteile gegenüber klassischen Navigationsstrukturen auf Basis von Roadmaps oder „reinen“ (d.h. im ursprünglichen Sinne benutzten) Outlines. Die wichtigsten Punkte werden im folgenden kurz erläutert.

3.4.1 Flexibles Navigationskonzept

Durch die Verwendung von Outlines über eine Konfigurationsumgebung ist eine einfache Erstellung solcher Navigationsstrukturen möglich. Endbenutzer erhalten die Möglichkeit, effizient Outlines zu erstellen und zu modifizieren. Durch die automatische Vererbung der funktionalen Eigenschaften der zentral gehaltenen Outline-Objekte an die jeweiligen Instanzen vereinfacht sich die Konsistenzwahrung mehrerer genutzter Outlines eines Typs in der Zielumgebung. Darüber hinaus bieten

somit die Outlines alle Vorteile von roadmapbasierten Navigationsstrukturen (vgl. 2.1).

3.4.2 Dynamische Anpassung beliebiger „Intelligenzen“

Ein entscheidener Nachteil einer Roadmap ist das Fehlen der Möglichkeit, eingebundene Verarbeitungsmechanismen für den designtechnisch unerfahrenen Benutzer zu erstellen, zu warten und zu ersetzen. Weiterhin mußte für jede Verarbeitsmethode die ihr zugrunde liegende Logik vollständig neu erstellt werden. Dies führt gerade bei mehreren auf diese Weise zu erstellenden „Intelligenzen“ zu erheblichen Mehraufwand. Diese Nachteile werden unter Verwendung der vorgestellten Lösungsmöglichkeit vermieden.

3.4.2.1 Abstrahierung von der technischen Realisation

Die Erstellung einer Navigationsstruktur sollte unkompliziert und mit dem Wissen eines erfahrenen, aber nicht designtechnsich geschulten Benutzers möglich sein. Das Abstrahieren von der technischen Realisierung bietet diesen Anwendern erst die Möglichkeit, Verarbeitungsmethoden sinnvoll einsetzen zu können, da sie nunmehr zum Einsatz der Verarbeitungsmethoden nur die Angaben über die Parameter der jeweiligen Methode (vgl. 2.2) tätigen müssen. Dies bedeutet eine Reduzierung der Komplexität in der Benutzung solcher Funktionalitäten.

3.4.2.2 Vordefinierung gebräuchlicher Intelligenzen

Mehrere Verarbeitungsmethoden gleicher Grundfunktionalität brauchen nicht mehr für jedes Verarbeitungsintelligenzen beinhaltende Element der Navigationsstruktur neu erstellt zu werden. Hierbei kann auf bereits vordefinierte Intelligenzen zurückgegriffen werden. Es sollten die häufigsten Verarbeitungsfunktionalitäten von einem Designer in die Konfigurationsumgebung eingebunden werden. Diese können dann beliebig wiederverwendet werden. Hierdurch wird dem Benutzer ein weitaus höheres Potential an Nutzungsmöglichkeiten solcher Methoden geschaffen, als es durch klassische Navigationskonzepte wie beispielsweise die Roadmap möglich war, da dieser sie nicht mehr implementieren muß oder sich mit einem designtechnisch erfahrenden Benutzer koordinieren braucht.

3.4.2.3 Direkter Entwurf von Aktionenlogik weiterhin möglich

Selbst bei intensivster Benutzerbefragung wird der Designer der Konfigurationsumgebung nicht alle benötigten Verarbeitungsmethoden implementieren. Zum einen können Situationen auftreten, in der neue Verarbeitungsintelligenzen benötigt werden, welche nicht im dynamischen Arbeitsumfeld vorgesehen wurden. Weiterhin mögen Situationen auftreten, in denen nur wenige Benutzer eine bestimmte Methode benötigen oder diese in nur seltenen Fällen nutzen. Dann ist eine Vordefinierung einer solchen Aktion nicht unbedingt zweckmäßig. Damit dennoch solche Methoden genutzt werden können, gibt es in der Konfigurationsumgebung die Möglichkeit, Aktionenlogiken selbständig zu erstellen. Die Grundlage für die Definierung solcher Methoden bildet die Makrosprache in Lotus Notes. Allerdings benötigt der Ersteller solcher Aktionen Kenntnisse über deren Implementierung. Wurden solche Kenntnisse nicht erworben, ist die Koordination mit designtechnisch geschulten Benutzern erforderlich.

3.4.3 Mobilität der Konfigurationsumgebung

Eine weiterer Vorteil der Konfigurationsumbegung ist die hochgradige Mobilität. Dies wird über eine Realisierung dieser Umgebung als Message-Objekt erreicht. Somit besitzt jedes Dokument der Konfigurationsumgebung die Möglichkeit, einerseits über replikative Mechanismen im gesamten Informations- und Kommunikationsnetz verteilt zu werden. Weiterhin ist auch das Versenden dieser Dokumente als elektronische Nachricht möglich. Der Empfänger besitzt die Möglichkeit, die Konfigurationsumbegung in jeder beliebigen Datenbank zu benutzen und weiter zu verteilen; alle benötigten Informationen und Verarbeitungsmethoden zur Erstellung der gewünschten Navigationsstruktur sind in dem Dokument als Message-Objekt enthalten. Somit erfolgt keine strikte Bindung an übergeordnete Aggregationsstrukturen, nur die in 3.2 beschriebenen Voraussetzungen müssen bereits gegeben sein.

3.5 Umsetzung in einen Prototypen

Dieses Kapitel ist eine kurze, generelle Funktionsbeschreibung und keine technische Dokumentation. Der Prototyp ist eine eigenständige Datenbank, in der Konfigurationsdokumente für outlinebasierte Navigationsstrukturen erstellt werden können. Dabei repräsentiert jeweils ein Dokument eine Outline (vgl. Abb. 6). Zudem enthält die Datenbank Agenten zum Auslesen von Outlines (vgl. 3.3) bzw. zum

Erstellen von Outlines aus der Konfigurationsumgebung heraus (vgl. 3.2). Im letzteren Fall kann der Agent auch in der jeweiligen Zieldatenbank vorhanden sein und von dort aus die in dem Konfigurationsdokument spezifizierte Navigationsstruktur erstellen (vgl. 3.2.2). Weiterhin sind alle unter Kapitel 3 beschriebenen Eigenschaften implementiert worden. Die Aktivierung der Agenten sowie das Versenden der definierten Navigationsstruktur aus der Konfigurationsumgebung heraus ist durch die Aktivierung ausgezeichneter Schaltflächen in dem jeweiligen Dokument möglich. Die baumartige Struktur der Outline ist in dem Konfigurationsdokument nachgebildet. Die jeweiligen Daten der einzelnen Navigationselemente werden per Auswahl des Eintrages aus dem „Baum“ heraus sichtbar. Desweiteren werden Möglichkeiten geboten, diese Einträge ein- und auszurücken. Damit können die jeweiligem Elemente der Outline kaskadiert werden. Nach erfolgter Erstellung bzw. Aktualisierung einer Outline wird mittels der implementierten Agenten für den Benutzer ein Protokoll erstellt, welches eventuelle Fehler dieses Prozesses aufzeigt. Diese können insbesondere bei ungültigen Einträgen im Konfigurationsdokument (beispielsweise ist bei einem Dokumentenlink das Dokument unrichtig spezifiziert worden) auftreten. Um diese Fehler vermeiden zu können, ist der Prototyp mit einer Simulationskomponente versehen. Hiermit wird die Erstellung bzw. Modifizierung einer Outline nicht operativ ausgeführt, wohl aber wird ein Protokoll erzeugt, welches potientielle Fehler des Prozesses aufzeigt. Der Prototyp ist unter Lotus Notes/Domino ab der Version 5.0 sowohl server- als auch clientbasiert lauffähig.

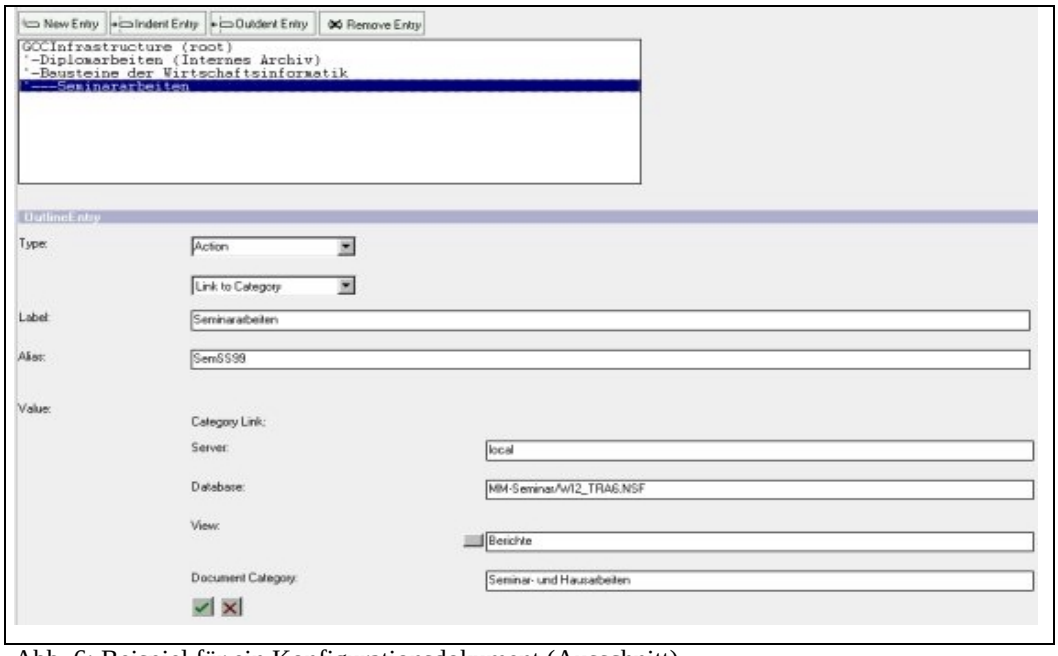

Abb. 6: Beispiel für ein Konfigurationsdokument (Ausschnitt)

4 Ausblick

Das Werkzeug auf Basis des Prototyps kann in verschiedenen Punkten ausgebaut und verbessert werden.

So ist es bisher nur möglich, allen Benutzern einer Datenbank das Erstellen bzw. Modifizieren der Navigationsstrukturen zu erlauben oder zu verbieten. Das in dieser Arbeit vorgestellte Werkzeug erfordert vom Benutzer eine gewissenhafte Anwendung, da sonst in einer Datenbank viele nur kurzfristig benötigte Outlines verweisen können. Dies könnte durch ein verfeinertes Sicherheitsmanagement verbessert werden. So könnte beispielsweise das Erstellen und/oder Modifizieren der Outlines aufgrund von Gruppenzugehörigkeiten oder Rollenvergabe eingeschränkt werden.

Der in dem vorgestellten Prototyp enthaltene Menge an vordefinierten Aktionen sollte um alle von den potentiellen Benutzern benötigten Funktionalitäten erweitert werden. Dies ist gerade bei der Weiterentwicklung dieses Prototypen zu einem Standardprodukt sinnvoll.

Die verwendeten Updatemechanismen bereits vorhandener Outlines könnten dahingehend verbessert werden, daß nicht nur die Möglichkeit besteht, die operative Outline um die im Konfigurationsdokument angegebenen Eigenschaften zu erweitern oder sie gegen diese zu ersetzten, sondern vielmehr das gezielte Einfügen einer gesamten Outline in eine bereits vorhandene an beliebiger Stelle zu ermöglichen. Somit entfällt der Prozeß des vorherigen Auslesens der operativen Navigationsstruktur und anschließender manuellen Änderung mit den gewünschten Einträgen.

Eine weitere Verbesserungs kann eine Erweiterung der Agenten derart darstellen, daß neben der manuellen Benutzung der bisherigen Überführungs- und Modifikationsfunktionalitäten die Konfigurationsumgebung um periodisch gesteuerte Aktualisierungsfunktionalitäten erweitert werden. Dies ist gerade für wiederkehrende, gleichartige Funktionen sinnvoll. Eine denkbare konkrete Realisierung dieses Ansatzes stellt eine Volltextsuche dar, die in einer oder mehrerer Datenbanken Dokumente aufgrund bestimmter Kriterien sucht und das Ergebnis als

Outlinestruktur erzeugt. Hierbei kann jedes gefundenes Dokument als ein Element in der Navigationsstruktur aufgefaßt werden. Damit die Menge der gefundenen Dokumente nicht zu groß wird, könnte eine Einschränkung des Suchergebnisses aufgrund der Aktualität der gefundenen Dokumente erfolgen.

5 Zusammenfassung

Die Benutzung von Navigationsstrukturen in nichtlinearen Datenbankstrukturen ist ein wichtiges Konzept zur Orientierung im virtuellen Raum. Gerade das in den letzten Jahren explosionsartig gewachsende Internet stellt im Hinblick auf die Informationsvielfalt die Benutzer vor die Herausforderung, effizient und zielgerichtet zu navigieren. Dadurch wächst die Bedeutung von Navigationskonzepten, die neben traditionellen Verzweigungsmechanismen die Verarbeitung von Methoden zum Zwecke der zielgerichteten Navigation erlauben. Diese Arbeit bietet die Möglichkeit, Navigationssrukturen mit solchen intelligenten Mechanismen benutzerfreundlich zu erzeugen, ohne daß tiefgreifende Kenntnisse in Programmier- bzw. Makrosprachen erforderlich sind. Dies schont wiederum die ohnehin schon begrenzten Kapazitäten von mit Programmierkenntnissen ausgestatteten Spezialkräften.

Durch die Erweiterung um die in Kapitel 4 beschriebenen Entwicklungspotentiale läßt sich der vorgestellte Prototypen in ein leistungsfähiges Standardprodukt überführen.

Literaturverzeichnis

Lotus Development Corporation (2000): Domino 5.0.3 Designer Help; aus: Software Lotus Notes/Domino 5.0.3

Lotus Development Corporation (1999): Domino 5.0.1 Designer Help; aus: Software Lotus Notes/Domino 5.0.1

Nastansky, Ludwig (1998): Message-Objekte und Team-Kommunikation – Systembausteine für die Unternehmensführung in neuen Organisationsformen; aus http://gcc.uni-paderborn.de am 01.07.1999