PS Software Engineering

WS 2000/2001 18. Januar 2001

Aufgabe 7

Protoyp

Gruppe 7

Zehentner Hildegard
hzehent@cosy.sbg.ac.at

Röck Andrea
aroeck@cosy.sbg.ac.at

Osterrieder Christian
coster@cosy.sbg.ac.at

Mayr Manfred
mmayr@cosy.sbg.ac.at

Angulanza Florian
fangulan@cosy.sbg.ac.at

Projektleiter: Mag. Dipl.-Ing. Schwaiger Roland

Institut für Computerwissenschaften
Universität Salzburg

Korrespondenz an:
Universität Salzburg
Institut für Computerwissenschaften und Systemanalyse
Jakob–Haringer–Straße 2
A–5020 Salzburg
Austria

Kapitel 1

Prototype

Es wird ein Protoyp erstellt, um die größten Risiken abzufangen.
Wir haben uns entschlossen die Eingabe von Daten als Prototyp zu gestalten.
Das Grundprinzip dieser Funktion: ein Objekt erstellen, ein Objekt suchen, ein Objekt abspeichern und die entsprechende Zusammenarbeit mit der Datenbank findet man überall im Projekt wieder. Wenn hier beim Grundkonzept Probleme auftreten, haben diese sehr große Auswirkungen.
Im Prototyp wird realisiert:
Über eine Eingabemaske ist es möglich Daten über ein Artefakt einzugeben und auch Daten über ein Artefakt wieder auszulesen.
Ziel ist es nicht eine perfekte Implementierung zu erzielen, z.B. können beim Prototypen noch nicht beliebig viele Attribute eingeben werden, sondern die wichtigsten Funktionalitäten zu implementieren. Wir wollen Daten aus der Datenbank lesen und in die Datenbank schreiben, ebenso wollen wir mit den Daten im Programm effektiv arbeiten und gewährleisten, dass die Datenbank konsistent bleibt.
Wir haben uns dazu entschlossen das wir im Programm mit Objekten arbeiten, welche beim Ezeugen ihre Daten aus der Datenbank lesen.
Am Schluß der Bearbeitung werden die Daten des Objekts wieder in die Datenbank geschrieben.
Kleinere Änderungen werden direkt auf das Objekt ausgeführt, was schneller ist, als wenn jedesmal die Datenbank angesprochen wird.
Die Konsistenz der Daten kann man in den Griff bekommen indem man sich merkt, wer die Daten gelesen hat, und nur dieser die Daten auch neu abspeichern darf (noch nicht im Prototy realisiert, da wir erst 1 Benutzer haben).
Beim Prototypen wird über eine JDBC-Schnittstelle mit einer MySql-Datenbank kommuniziert, welche die entsprechenden Daten enthält.

1.1 Handlungsablauf

Das Programm wird von der html Seite index.htm gestartet. Als 1. erscheint die Startseite:

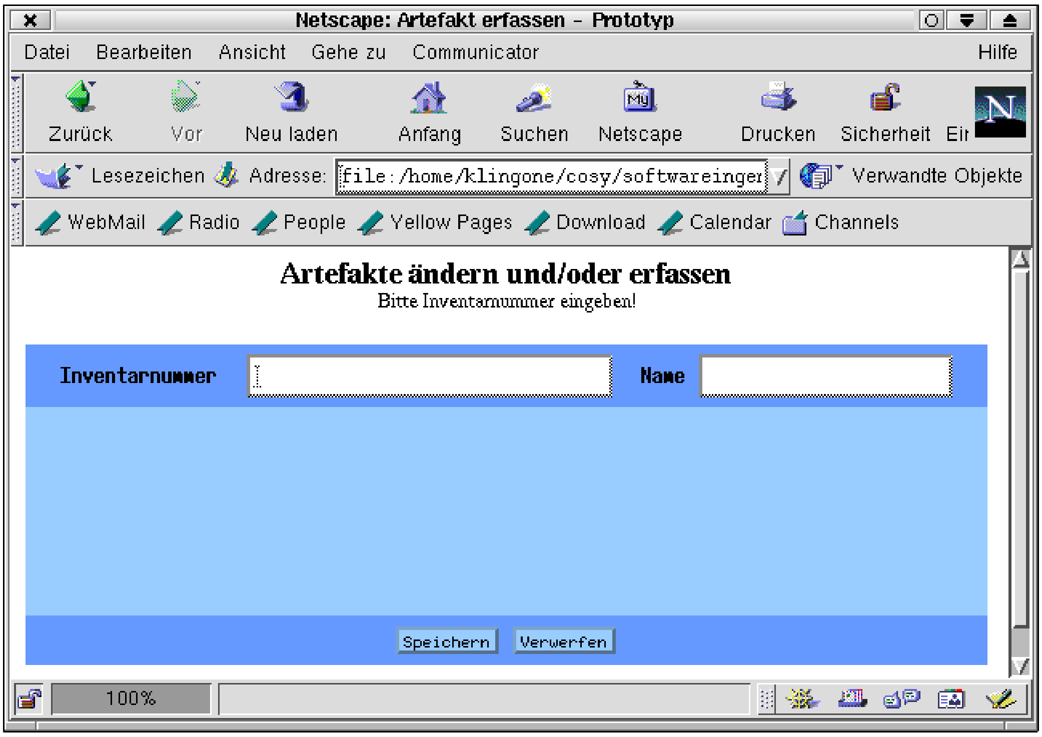

Abbildung 1.1: Startseite

Nur das Feld für die Inventarnummer ist aktiv. Der Name kann noch nicht eingegeben werden.
Als nächstes muss der Bentzer eine Inventarnummer eingeben. Es wird in der Datenbank gesucht, ob schon ein Artefakt mit dieser Inventarnummer existiert.

- Wenn bereits eine Artefakt existiert wird als nächstes die Seite zur Änderung der Attribute aufgerufen, wobei in den Attributsfeldern die Daten des Artefakts mit der eingegebenen Inventarnummer stehen.

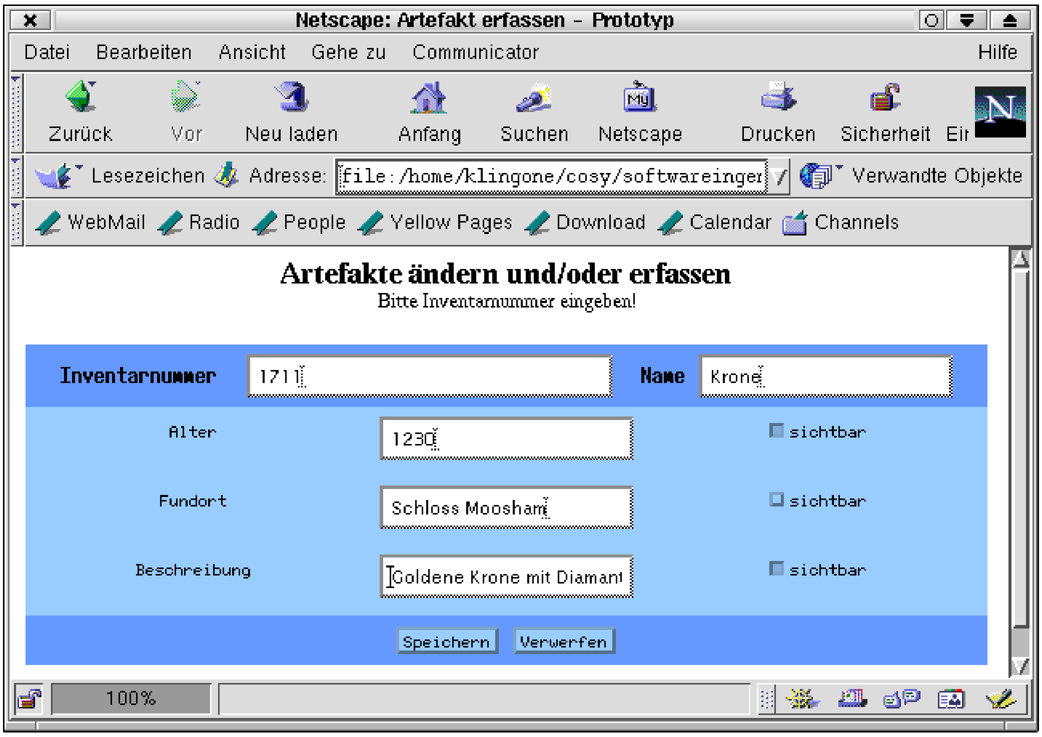

Abbildung 1.2: Ändern der Attribute eines vorhandenen Artefakts

Es können nun die Werte des Artefakts geändert werden.
Nach Abschluß der Arbeit kann man durch betätigen von *Speichern* die neuen Daten des Artefakts in die die Datenbank schreiben.
Wird *Verwerfen* gewählt so werden die Daten nicht abgespeichert und man gelangt wieder zur Startseite (siehe Abb. 1.1) und kann ein neues Artefakt bearbeiten.

- Ist in der Datenbank noch kein Artefakt mit der eingegebenen Inventarnummer vorhanden so erscheint ein leeres Fromular zum Ändern der Attribute.
  Es können nun in allen Feldern etwas eingegeben werden. Auch das Feld zur Namenseingabe ist jetzt freigegeben.
  Die Eingabe des Namens ist optional, die Eingabe der anderen Attribute ist optional.

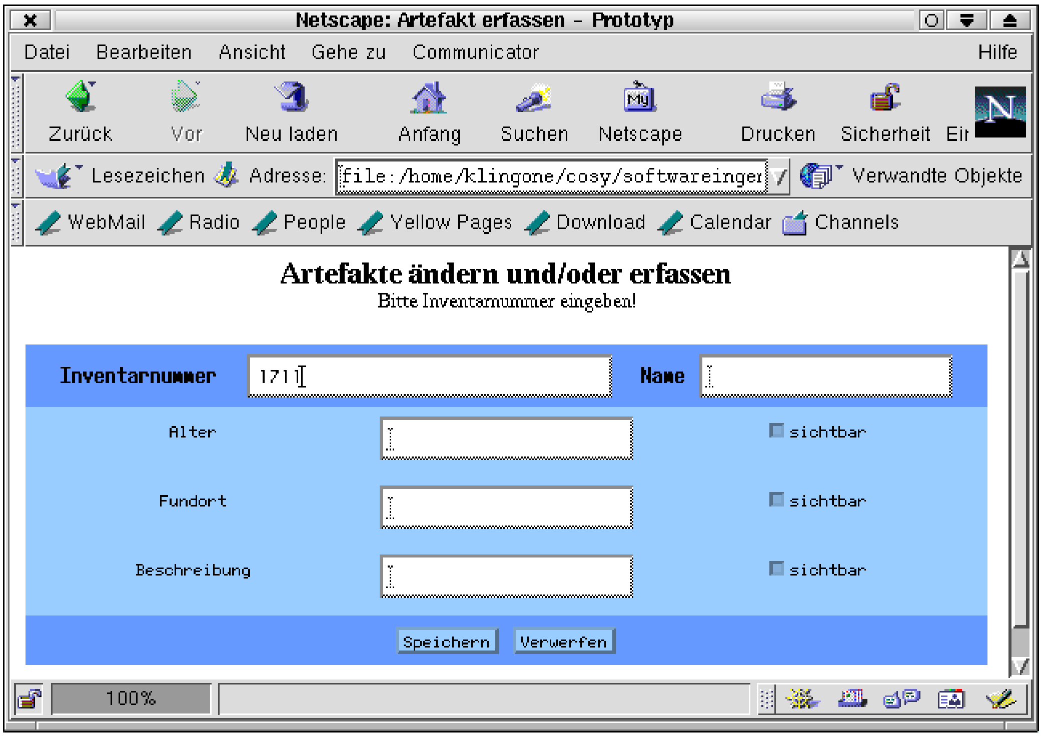

Abbildung 1.3: Neues Artefakt wurde angelegt

Wird nach der Eingabe der Attribute *Speichern* ausgewählt so wird in der Datenbank ein neues Artefakt mit den entsprechenden Daten angelegt.
Bei *Verwerfen* werden die Daten nicht weiterverarbeitet und der Benutzer gelangt zur Startseite (siehe Abb. 1.1) und kann ein neues Artefakt bearbeiten.

1.2 Funktionsweise

Beim Öffnen der Startseite versucht das Applet über JDBC eine Verbindung zur Datenbank herzustellen.
Nachdem die Inventarnummer eingegeben wurde wird in der Datenbank gesucht, ob schon ein Objekt mit dieser Inventarnummer existiert. Wenn dies der Fall ist, wird ein Objekt mit den Daten des Artefakts aus der Datenbank erzeugt. Mit diesem Objekt wird nun in dem Applet weitergearbeitet.
Ist noch kein Artefakt mit dieser Inventarnummer vorhanden, so wird ein Objekt ohne Attribute (bis auf die Inventarnummer) erzeugt und mit diesem weitergearbeitet.
Bei *Speichern* wird in beiden Fällen die Daten des Objekts mit welchem in dem Applet gearbeitet wurde, in der Datenbank gespeichert.
Bei *Verwerfen* wird in beiden Fällen das Objekt verworfen.

1.2.1 Datenstruktur

In der Datenbank existiert eine Entität ARTIFACTS, welche die Inventarnummer (ID) und den Namen des Artefakt (NAME) enthält.
Ebenso existiert eine Entität ARTATTRIBUTES welche einem optionalem Attribut entspricht. Es wird durch seinen Namen (NAME), seinem Wert (VALUE), seiner Sichtbarkeit (VISIBLE) und natürlich dem Verweis zu welchem Attribut es gehört (ART_ID) bestimmt.
Durch die Methode alle Attribute eines Artefakts, außer der Inventarnummer und dem Namen auszulagern, hat man die Möglichkeit jedem Artefakt genau jene Anzahl und Art der Attribute zu geben, welche es am besten beschreiben.

1.2.2 Performance

Um eine möglichst schnelle Bearbeitung und Verwendung der Daten zu erreichen haben wir uns entschlossen nicht bei jeder Datenänderung auf die Datenbank zu zugreifen, sondern die Daten in Objekten zwischen zu halten. Dies hat den Preis dass zusätzlich geregelt werden muss, das nicht 2 Personen gleichzeitig an einem Artefakt arbeiten und somit die Konsistenz der Datenbank gefärdet wird.
Wir haben uns für die Verwendung von Applets entschieden, da dies die schnellste Variante ist. Nur beim holen der Daten und beim Abspeichern der Daten muss auf den Server und somit die Datenbank zugegriffen werden. Alle anderen arbeiten werden vom Applet direkt beim Benutzer erledigt. Es muss jedoch auch darauf geachtet werden, dass der Benutzer ohne größere

Probleme und ohne langes herunterladen von benötigten libaries arbeiten kann.