Fakultät für
Elektrotechnik, Informatik und Mathematik

Projektgruppe - Generierung von Web-Anwendungen aus visuellen Spezifikationen

PaderWAVE

Abschlussbericht

Teilnehmer:
Bastian Cramer
Oxana Paul
Jana Schäfer
Christian Schneider
Elvira Schumacher
Jens Siebert
Stephan Winter
Barbara Zimmermann

Betreuer:
Dipl. Inform. Carsten Schmidt und Dr. Michael Thies

Paderborn, den 9. Mai 2005

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Vorwort

Dieser Abschlussbericht ist Bestandteil der Entwicklungen im Rahmen der Projektgruppe „Generierung von Web-Anwendungen aus visuellen Spezifikationen“ [1] an der Universität Paderborn, in der Arbeitsgruppe „Programmiersprachen und Übersetzer“ von Prof. Dr. Uwe Kastens [2]. Ziel des Projektes ist die Entwicklung einer visuellen Entwicklungsumgebung, mit dessen Funktionalität der Benutzer die Möglichkeit hat, durch Eingabe einer auf visuellen Konzepten basierenden Spezifikationssprache, eine vollfunktionsfähige und konsistente Web-Anwendung zu generieren.

2 Einleitung

Eine der wichtigsten auf der Kommunikationsplattform *Internet* aufbauenden Anwendungen ist das World-Wide-Web (WWW). Das WWW basiert auf der der Verbindung von Ressourcen, die über das ganze Internet verstreut liegen können. Solche Ressourcen können zum Beispiel andere Dokumente, Bild- oder Audio-Dateien sein. Mit Einführung des WWW beschränkte man sich zunächst auf Dokumente mit rein statischem Inhalt. Die Dokumente lagen in Form einer Datei auf dem Web-Server vor und wurden bei Anfragen von Clients unverändert an diese übermittelt. Mit dem Aufkommen neuer Anwendungsfelder (Suchmaschinen, Portale) waren Dokumente mit rein statischem Inhalt unpraktikabel bzw. viele Anwendungen erst gar nicht realisierbar. Als Lösung für dieses Problem bot sich die dynamische Generierung von Web-Dokumenten an. Dabei wird das an einen Client übertragene Web-Dokument dynamisch von einem auf den Web-Server laufenden Programm/Skript erzeugt.

Mit Hilfe der dynamischen Generierung können Web-Dokumente um individuellen Inhalt angereichert werden. Mittlerweile gibt es unzählige Technologien mit dessen Hilfe man Web-Dokumente generieren kann. Die Skriptsprache PHP ist eine davon. Sie ist auf die speziellen Bedürfnisse der Generierung von Web-Dokumenten zugeschnitten. Eine weitere ist Java. Neben der Programmiersprache Java wird ein komplettes Framework für die Entwicklung von Web-Dokumente angeboten.

Durch die vielen unterschiedlichen Technologien zur Entwicklung von Web-Anwendungen und deren steigende Komplexität wird deren Entwicklung von der Technik dominiert. Die Entwicklung von Inhalt für die Web-Anwendungen gerät dabei immer mehr in den Hintergrund. Viele Web-Anwendungen (eBay) haben bereist die Komplexität von großen Desktop-Anwendungen bzw. Betriebssystemen.

Wie bei normalen Desktop-Anwendungen besteht auch bei Web-Anwendungen das Problem der Wartbarkeit. Änderungen im Programm-Code müssen mit entsprechender Vorsicht durchgeführt werden, um keine Fehler einzubauen. Insbesondere bei großen Anwendungen ist dies aufgrund der nicht überschaubaren Komplexität ein großes Problem. Eine Lösung dieses Problems ist die visuelle Modellierung von Anwendungen. Für die Entwicklung von Desktop-Anwendungen gibt es bereits zahlreiche visuelle Entwicklungsumgebungen (Together) mit denen man die statische Struktur von Desktop-Anwendungen *visuell* modellieren kann und daraus dann den

entsprechenden Programm-Code, zum Beispiel C++ oder Java, generieren lassen kann. Als Modellierungssprache wird häufig UML (Unified Modelling Language) eingesetzt. Die visuelle Modellierung einer Anwendung hebt die Entwicklung auf eine abstraktere Ebene und erleichtert damit die Überschaubarkeit/Verständlichkeit der Anwendung, wodurch die Wartbarkeit verbessert wird. Zusätzlich kann durch die Verifikation des Modells, zum Beispiel durch Konsistenzüberprüfungen und die Generierung von fehlerfreiem Programm-Code die Robustheit der Anwendung verbessert werden.

Ziel der Projektgruppe ist die Entwicklung einer visuellen Entwicklungsumgebung, mit der Web-Designer Anwendungen für das Web visuell modellieren können, um danach den erforderlichen Anwendungs-Code ohne eigenes zutun von der Umgebung generieren zu lassen. Der Anwendungs-Code der generierten Web-Anwendung basiert dabei auf HTML mit Anteilen an PHP-Skripten.

Durch die visuelle Modellierung und die Codeerzeugung durch die Umgebung ist kein teures Expertenwissen aus dem Bereich der Web-Programmierung erforderlich. Damit richtet sich die Umgebung vornehmlich an Web-Designer ohne Programmierkenntnisse. Der Entwickler kann sich ganz auf den Inhalt der Web-Anwendung konzentrieren, ohne sich mit technischen Fragen der Umsetzung beschäftigen zu müssen.

Ein weiterer Vorteil für den Benutzer der visuellen Umgebung ist die Vermeidung von Programmierfehlern, die bei handgeschriebenen Web-Anwendungen schnell entstehen, insbesondere bei steigender Komplexität. Die visuelle Umgebung erzeugt robusten und konsistenten Anwendungs-Code. Das Einschleichen von Fehlern in bestehenden Anwendungs-Code durch ständiges Modifizieren wird verhindert. Der Benutzer arbeitet nur mit der Spezifikation der Web-Anwendung. Aus diesem kann er einfach bestehende Sprachelemente entfernen oder die Spezifikation um zusätzliche Elemente erweitern. Aus der modifizierten Spezifikation wird dann der Anwendungs-Code, nach erfolgreichem Bestehen der Konsistenzprüfungen, vollständig neu erzeugt. Dies verhindert die Erzeugung von fehlerhaftem Anwendungs-Code oder die Notwendigkeit, schlecht strukturierten Code erweitern zu müssen.

Durch die Reduzierung der Entwicklung auf die reine Modellierung der Web-Anwendung lassen sich in kürzester Zeit einfache Prototypen (Rapid-Prototyping) erstellen oder komplexe Anwendungen mit einer hohen Lebenszeit generieren. Des Weiteren wird dem Benutzer die Möglichkeit gegeben, selbst geschriebene PHP-Skripte in die generierte Web-Anwendung einbinden zu können. Dies gibt ihm die Freiheit, seine Web-Anwendung um nicht vorgesehene Funktionalität zu erweitern oder bereits bestehenden Anwendungs-Code wieder zu verwenden.

Für die Entwicklung der visuellen Umgebung wird selbst wiederum ein Generator

eingesetzt. Mit dem an der Universität Paderborn entwickelten Generator DEViL [6] lassen sich visuelle Entwicklungsumgebungen generieren. Die visuelle Codeerzeugung, der damit zu entwickelnden visuellen Umgebung, lässt sich frei spezifizieren. Der Einsatz eines Generators für die Entwicklung der visuellen Umgebung weist die bereits oben genannten Vorteile, wie Wartbarkeit und schnelle, fehlerfreie Entwicklung, auf.

Die Entwicklung der visuellen Umgebung für Web-Anwendungen wurde zu Beginn in zwei Phasen geteilt. Das Ergebnis der ersten Phase war eine noch funktional eingeschränkte Umgebung (1. Prototyp) mit der man aber bereits einfache Web-Anwendungen modellieren und anschließend generieren konnte. So war zum Beispiel das Einbinden von benutzerdefinierten Skripten und die Modifikation von Datenbank-Tabellen nicht möglich. Der Zweck dieser Zwei-Phasen-Teilung war es, sich zunächst auf einen Teil der erforderlichen Aufgaben zu konzentrieren, um dabei Erfahrungen zu sammeln, die für eine zeitgemäße Umsetzung des Gesamt-Projektes von hohem Nutzen sein werden. Aus diesem Grund wurden alle Tätigkeiten mit Hilfe eines TimeTrackers zeitlich erfasst. Die dadurch gewonnenen Erkenntnisse wurden bei der Planung der zweiten Phase verwendet. In der zweiten Phase wurde dann das Sprachmodell und die Codeerzeugung der visuellen Umgebung um die fehlende Funktionalität vervollständigt. Das Ergebnis dieser Arbeit und damit das Endergebnis der Projektgruppe PaderWAVE wird in diesem Abschlussbericht vorgestellt.

Der Abschlussbericht ist wie folgt aufgebaut. In Kapitel 3 werden die Grundlagen erläutert. Dies umfasst Erläuterungen zu den für die Entwicklung verwendeten Werkzeugen, sowie verwendete Web-Technologien für die Codegenerierung. Des Weiteren werden verwandte Systeme vorgestellt. Danach folgt in Kapitel 4 die Beschreibung der Ziele des Systems und dessen Einsatzgebiet. Anschließend wird in Kapitel 5 der Sprachentwurf anhand einer mit der visuellen Umgebung erstellten Web-Anwendung erklärt. Dabei wird auf die einzelnen Sprachkonstrukte der visuellen Sprache eingegangen und die dahinter liegenden Konzepte vorgestellt. Das genaue Vorgehen bei der Modellierung einer komplexeren Web-Anwendung mit Datenbank-Anbindung wird in Kapitel 6 beschrieben. Aspekte der Realisierung werden in Kapitel 7 behandelt. Dabei werden einzelne Details der technischen Umsetzung, wie zum Beispiel das Anlegen der Datenbank-Schemata, besprochen. Im letzten Kapitel gibt es eine kurze Zusammenfassung der wichtigsten Punkte. An dieser Stelle werden auch die gesammelten Erfahrungen während der einjährigen Projektarbeit diskutiert.

3 Grundlagen

3.1 Visuelle Programmierung

Visuelle Programmierung kann man grob in zwei Teilgebiete aufteilen: Visuelle Umgebungen und visuelle Programmiersprachen. Visuelle Umgebungen stellen ein einfaches Hilfsmittel dar, mit dem man Programme einfach und intuitiv entwerfen kann, meist unter Verwendung einer visuellen Sprache. Zu den Haupteinsatzgebieten zählen Multimedia-Anwendungen, intelligente Zeichenprogramme und grafische Unterstützung für die Softwareentwicklung. Mittels visueller Umgebungen kann man Datenstrukturen visualisieren, Algorithmen animieren und Programmabläufe veranschaulichen. Visualisierungen bieten dem Benutzer eine intuitivere Bedienung der Umgebung, in der er sich schneller zurechtfindet und seine Daten besser verwaltet.

Visuelle Programmiersprachen werden erfolgreich im Bereich problemorientierter Spezifikationssprachen eingesetzt. Der Vorteil des Einsatzes liegt in den prägnanten visuellen Notationen, die leicht verständliche Modellbeschreibungen ermöglichen. Eine visuelle Sprache versetzt den Benutzer in die Lage Teile seines Programms intuitiv zu beschreiben. Visuell dargestellte Programme können schneller erfasst werden. Im Vergleich zu textuellen Sprachen möchte man bei visuellen Sprachen weniger textuelle Beschreibungen haben und versucht Programme schnell und einfach visuell zu implementieren. Die Realisierung unterschiedlicher Anwendungen soll durch grafische Interaktionen erreicht werden.

3.2 Verwandte Systeme

Im Bereich der Generatoren, die auf Basis visueller Spezifikationen Web-Anwendungen erstellen können, gibt es bereits einige existierende Systeme, die hier kurz vorgestellt werden.

3.2.1 ProGUM-Web

Das System ProGUM-Web [3], das an der Universität Paderborn entwickelt wurde, verwendet die Modellierungssprache UML um die Spezifikation der zu generierenden

Web-Anwendung zu erstellen. Die Zielsprache dieses Systems ist die Skriptsprache PHP.
ProGUM-Web verwendet einen Workflow-basierten Ansatz für die Entwicklung von Web-Anwendungen. Im Modelling-Workflow wird die Web-Anwendung mittels eines UML-Werkzeuges modelliert. Das so entstandene Modell der Anwendung wird anschließend in das UML-Austauschformat XMI exportiert. In dieser Form kann das Modell in den ProGUM-Web-Generator importiert werden. Dieser Generator arbeitet gleichzeitig als ein Repository, welches die Dateien des Projektes zentral verwaltet. So wird zusätzlich Teamarbeit durch den Generator unterstützt.
Innerhalb des Coding-Workflow wird die Anwendung entwickelt. Dazu können die Entwickler zusätzliche Templates, wie HTML-Dateien, Grafiken oder PHP-Skripte in das Repository einpflegen und mit den entsprechenden Teilen der Anwendung verknüpfen.
Im darauf folgenden Prototyping-Workflow wird die Anwendung durch den ProGUM-Web-Generator aus den im Repository vorhandenen Teilen erstellt. Dieser Anwendungs-Prototyp kann anschließend auf einem Web-Server installiert und getestet werden. Bei diesen Tests auftretende Fehler können dann sofort im Repository korrigiert und ein neuer Prototyp erstellt werden. So entsteht ein Kreislauf zwischen dem Coding-Workflow und dem Prototyping-Workflow.

3.2.2 T-Web

Das T-Web-System [4] des *Tokyo Institute of Technology* benutzt so genannte Web-Transition-Diagramme zur Darstellung der Web-Anwendung. Die erstellte Anwendung benutzt die JSP/Servlet-Technologie der Firma SUN Microsystems.
T-WEB verwendet so genannte Web-Transition-Diagramme um eine Web-Anwendung zu spezifizieren. Diese Web-Transition-Diagramme bestehen aus Knoten und Kanten, wobei es unterschiedliche Typen von Knoten und Kanten gibt. Die beiden wichtigsten Kantentypen sind der einfache Hyperlink, sowie der Dataflow-Link. Eine Hyperlink-Kante entspricht einem HTML-Hyperlink, welcher auf eine andere Web-Seite verweist. Der Dataflow-Link entspricht einem HTTP-Submit. Hierbei werden Daten, zum Beispiel aus einem Formular, vom Web-Browser zu einem Web-Server geschickt, welcher die Daten entsprechend auswertet. In einem Web-Transition-Diagramm werden die zu übermittelnden Daten durch die Beschriftung an der Dataflow-Link-Kante repräsentiert.
Ein Web-Transition-Diagramm kann eine Vielzahl unterschiedlicher Knotentypen enthalten. Die Wichtigsten sind hier der Fixed Web-Page-Knoten, der Output Web-Page-Knoten, der Server-Programm-Knoten, der Database-Knoten, sowie der Client-Browser-Knoten.
Der Fixed Web-Page-Knoten und der Output Web-Page-Knoten repräsentieren unterschiedliche Typen von Web-Seiten. Ein Fixed Web-Page-Knoten entspricht dabei einer statischen Web-Seite, die zur Laufzeit der Anwendung nicht mehr verändert wird. Der Output Web-Page-Knoten repräsentiert eine, zumindest

teilweise, durch ein serverseitiges Programm generierte Web-Seite.
Der Server-Programm-Knoten steht für ein serverseitiges Programm. Im Falle des T-WEB-Systems sind hiermit Java-Programme gemeint. Diese Programme verarbeiten die Daten, welche durch einen Dataflow-Link übermittelt werden und generieren, basierend auf den Ergebnissen der Verarbeitung, eine neue Web-Seite.
Der Database-Knoten repräsentiert eine Datenbanktabelle. Durch setzen entsprechender Dataflow-Links kann mit diesem Knotentyp eine Datenbankanfrage modelliert werden.
Der Web-Client des Benutzers der Web-Anwendung wird durch den Client-Browser-Knoten repräsentiert. Hiermit kann unter anderem die Speicherung von Daten im Web-Client mittels Cookies modelliert werden.
Das T-WEB-System setzt sich aus den Komponenten Generator und Editor zusammen. Im T-WEB-Editor wird die Web-Anwendung mit Hilfe der Transition-Web-Diagramme modelliert. Der Editor prüft bereits bei der Eingabe die Konsistenz der Anwendung. Wird zum Beispiel an einem Dataflow-Link ein Parameter definiert, der in der Quellseite nicht existiert, so erkennt der Editor dies als Fehler und der Benutzer wird darauf hingewiesen, dass die Anwendung inkonsistent ist.
Im Editor können bereits vorhandene Templates mit den einzelnen Knotentypen verknüpft werden. So kann zum Beispiel ein Fixed Web-Page-Knoten mit einer bereits existierenden HTML-Seite verknüpft werden oder ein Server-Program-Knoten mit einem existierenden Java-Servlet. Der T-WEB-Generator erstellt aus der Beschreibung des Web-Transition- Diagramms und den vorhandenen Templates anschließend den Prototypen der modellierten Web-Anwendung.

3.2.3 BioPro

BioPro [5] ist ein Generator, welcher an der *Universität Tokushima* entwickelt wurde. Dieser Generator benutzt einen Bild-basierten Ansatz mit Komponenten zur Darstellung der Anwendung. Die Zielsprache dieses Systems ist ebenfalls Java.
BioPro verwendet einen visuellen, bildorientierten Ansatz zur Spezifikation einer Web-Anwendung. Im BioPro-Editor wird zwischen drei grundlegenden Elementen für die Erstellung einer Web-Anwendung unterschieden. Zum Einen existieren die so genannten Programm-Tabellen. Diese dienen als Container für Daten, welche in der Web-Anwendung verwendet werden können. Diese Daten sind in Tabellenform, wie in einer Datenbank, abgelegt. Die Programm-Tabellen werden entweder vom Entwickler der Web-Anwendung entworfen und mit Daten gefüllt, oder können auch aus einer bereits bestehenden Datenbank erstellt werden. Im ersten Fall erstellt der BioPro-Generator ein entsprechendes Java-Objekt, welches die vom Entwickler festgelegten Daten, sowie Zugriffsmethoden für diese Daten enthält. Wird eine Programm-Tabelle aus einer existierenden Datenbank erstellt, so generiert der Generator die entsprechenden Datenbankabfragen und Zugriffsmethoden für die Daten.
Die eigentlichen Web-Seiten werden in der Web-Page-Ansicht entworfen. Diese

Ansicht ist eine Darstellung, die der fertigen Web-Seite bereits sehr ähnlich sieht. Die Gestaltung der Web-Seite wird mit Komponenten, wie Textfeldern oder Schaltflächen realisiert. Daten aus den Spalten einer Programm-Tabelle werden durch Platzhalter in die Web-Seite eingefügt. Diese Platzhalter sind im Editor durch eine Linie mit der entsprechenden Spalte der Programm-Tabelle verknüpft. Der entsprechende Code für den Zugriff der Web-Seite auf die Daten in der Programm-Tabelle wird durch den Generator erstellt.
Zu jeder Web-Seite im BioPro-System gehört eine so genannte Web-Source. Hier wird der Code eingefügt, welcher nicht vom Generator erzeugt werden kann. Dies kann zum Beispiel Code für eine Berechnung sein. Der Aufbau einer Anwendung, welche mit dem BioPro-System erstellt wurde, entspricht dem Model-View-Controler-Prinzip. Das Datenmodell besteht bei einer solchen Anwendung aus den Programm-Tabellen, bzw. Datenbank-Tabellen. Die View wird durch die im BioPro-Editor entworfenen Web-Pages dargestellt. Als Controller dient schließlich die zu jeder Web-Page vorhandene Web-Source, welche den generierten und den vom Entwickler eingefügten Anwendungscode enthält.
Eine Besonderheit des BioPro-Systems sind die so genannten „Actions“. Hierbei handelt es sich um eine Technik, mit deren Hilfe man Aktionen für den Zugang zu einer Web-Seite festlegen kann. Welche Aktion dabei ausgeführt wird, ist davon abhängig, von welcher Quell-Seite der Benutzer auf die Ziel-Seite zugreift. Ein Anwendungsbeispiel wäre, eine für nicht registrierte Benutzer gesperrte Web-Seite. Würde ein Benutzer, der sich noch nicht bei der Web-Anwendung registriert oder angemeldet hat, versuchen auf die gesperrte Web-Seite zuzugreifen, so würde er von der entsprechenden Aktion zunächst auf eine Anmelde-Seite weitergeleitet. Versucht jedoch ein angemeldeter Benutzer von einer anderen Web-Seite innerhalb der Web-Anwendung auf die zugriffsbeschränkte Web-Seite zuzugreifen, so wird die entsprechende Aktion dies registrieren und den Benutzer zur Ziel-Seite weiterleiten.

3.3 DEViL

Als Grundlage für die Entwicklung der visuellen Umgebung dient der Generator DEViL (Development Environment for Visual Languages). Der Generator erzeugt visuelle Entwicklungsumgebungen aus Spezifikationen der Sprachstruktur und der grafischen Darstellung einer visuellen Sprache. Dabei wird die Verarbeitung (Transformation) der generierten Umgebung (ausgehend vom abstrakten Strukturbaum) in eine Zielsprache separat spezifiziert. In Abbildung 3.1 wird dieser Sachverhalt grafisch verdeutlicht.

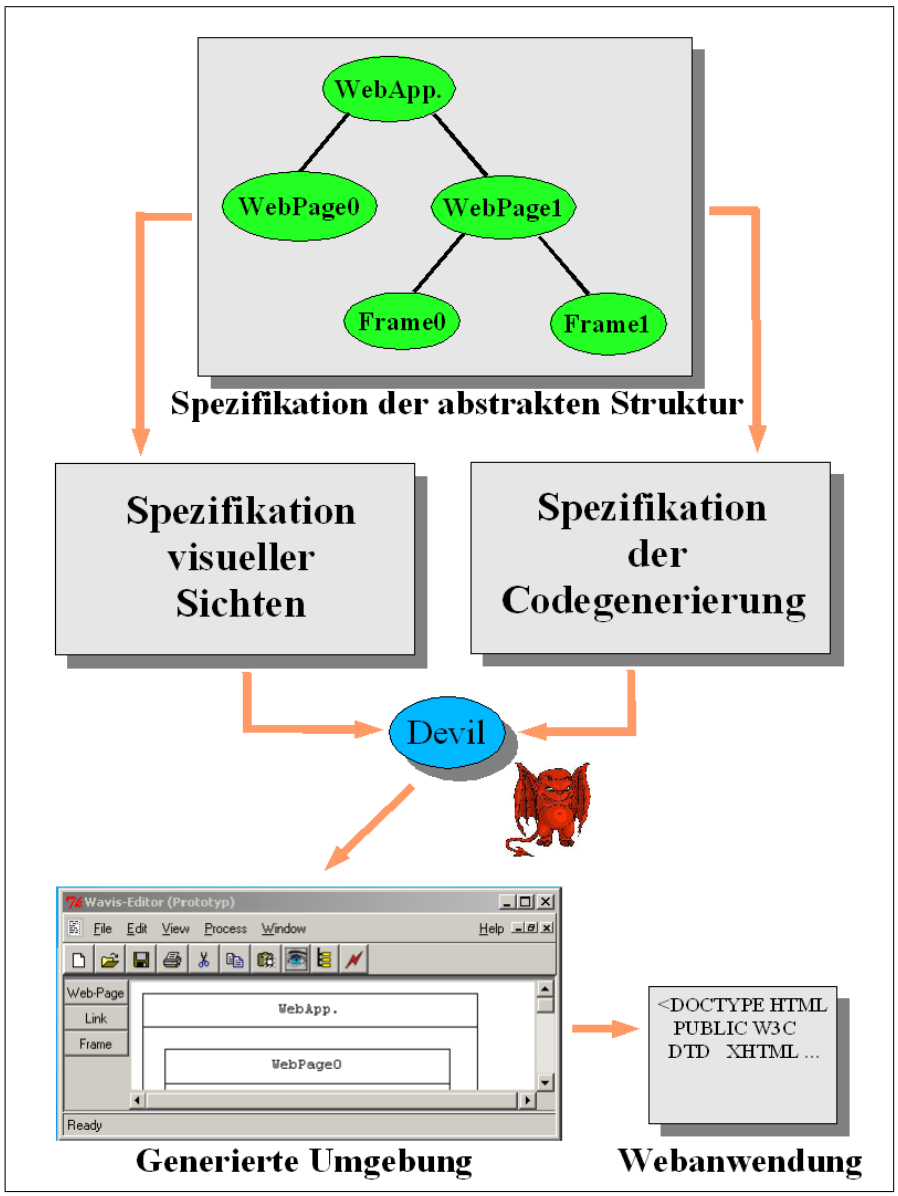

Abbildung 3.1: Generierung der visuellen Entwicklungsumgebung

Zu Beginn wird das Sprachmodell der visuellen Sprache spezifiziert. Die Spezifikation der Sprache des Modells basiert auf dem Konzept von Klassenhierarchien. Klassen des Sprachmodells enthalten Attributdefinitionen und können Eigenschaften von allgemeineren Klassen erben. Aufbauend auf dem Sprachmodell werden die visuellen Sichten und die Codegenerierung spezifiziert. Die Sichten legen visuelle Repräsentationen der abstrakten Struktur fest und können vom Benutzer dazu genutzt werden, eine Instanz des Sprachmodells, die Programmstruktur einer Web-Anwendung, zu erstellen. Mit Hilfe der definierten Code-Muster aus der Spezifikation der Codegenerierung wird die Programmstruktur einer Web-Anwendung dann in die anwendungsspezifische Zielsprache, zum Beispiel HTML mit eingebet-

teten Skriptsprachen-Anteilen (PHP), transformiert.

Die visuelle Sicht stellt einen bestimmten Teil der abstrakten Struktur grafisch dar und legt das Aussehen der Strukturteile fest. Mittels dieser Repräsentation können Änderungen an der abstrakten Struktur vorgenommen werden. Es ist zulässig mehrere Sichten zu definieren, die die gleiche Programmstruktur auf unterschiedliche Weise repräsentieren. Sichtdefinitionen können Toolbar-Knöpfe enthalten, welche am linken Rand des Sichtfensters angeordnet erscheinen. Sie ermöglichen neue Sprachkonstrukte per Drag-and-Drop einzufügen. Für die Sprachelemente werden automatisch Editier-Dialoge generiert. Die Spezifikation der grafischen Repräsentation einer Sicht besteht in der Angabe einer attributierten Grammatik, die das Zeichnen der entsprechenden Grafik veranlasst. Die grafische Darstellung einer Sicht wird somit berechnet, indem sie durch Attributberechnungen in ein grafisches Canvas gemalt wird.

Um die Spezifikation der grafischen Darstellung einfacher zu machen, wurden Implementierungen häufig vorkommender grafischer Repräsentationen in so genannte Visuelle Muster gekapselt. Visuelle Muster beschreiben, wie eine bestimmte Informationsstruktur visuell repräsentiert werden kann. Z.B. gibt es das Mengen-Muster(VPSet), in dem ein grafisches Element Unterelemente enthält, die frei in einem bestimmten Bereich positionierbar sind. Weitere wichtige visuelle Muster, von denen bei der Spezifikation Gebrauch gemacht wurde, sind Listen (List-Pattern), Formulare (Form-Pattern), Tabellen (Table-Pattern), Matrizen (Matrix-Pattern), Linienverbindungen (Connection-Pattern), Beschriftungen (Label-Pattern), Fliesstext (FlowLayout-Pattern) und Primitive (Primitive-Pattern). Visuelle Muster werden angewendet, indem bestimmte Grammatiksymbole von bestimmten Symbolrollen erben.

Zur grafischen Darstellung von Formularen lassen sich mit einem visuellen Editor die so genannten generischen Zeichnungen erstellen. Generische Zeichnungen sind grafische Spezifikationen, mit denen sich Ausprägungen bestimmter Formular-Darstellungen einfach und komfortabel beschreiben lassen, und die bei Anwendung der Muster VPForm und VPFormList referenziert werden.

3.4 HTML und PHP

Für die Verbreitung von Inhalten über das Internet hat sich das Dokumentenformat HTML [7] durchgesetzt. Neben dem eigentlichen Text enthalten HTML-Dokumente Formatierungsangaben, welche vom Web-Browser interpretiert werden und bei der Darstellung des Dokumentes verwendet werden. Web-Dokumente die auf HTML basieren, sind rein statisch und bieten keine Möglichkeit der Interaktion. Um eine Web-Anwendung bzw. deren einzelnen Web-Seiten mit Interaktionen und dynamischem Inhalt auszustatten, hat man die Wahl zwischen vielfältigen

Technologien. Dies reicht von einfachen Skriptsprachen wie zum Beispiel Perl, PHP bis zu kompletten Frameworks wie zum Beispiel das Java-basierende Struts-Framework [9]. Mit Hilfe dieser Technologien können auf der Seite des Web-Servers HTML-Dokumente dynamisch erzeugt werden. Damit hat man die Möglichkeit, jede Anfrage zu qualifizieren und individuell zu beantworten.

Für die Realisierung der dynamischen Teile einer Web-Anwendung, haben wir uns für PHP entschieden. Die entscheidenden Vorteile von PHP sind die einfache Syntax, die schnelle Text-Verarbeitung, eine gute SQL-Anbindung, die Verfügbarkeit von umfangreichen Bibliotheken und die hohe Verbreitung.

3.4.1 HTML

Bei HTML (Hyper Text Markup Language) handelt es sich um eine weit verbreitete Auszeichnungssprache zur Erstellung von Dokumenten, die über das Internet verteilt werden. Sie wurde vom Web-Gründer Tim Berners-Lee entwickelt und wurde im Laufe der Zeit zum erfolgreichsten und verbreitetsten Dateiformat der Welt.

HTML ist eine Sprache zur Strukturierung von Texten, wobei der Entwickler zusätzlich die Möglichkeit besitzt, Grafiken und multimediale Inhalte in Form einer Referenz einzubinden und in den Text zu integrieren.

Zusätzlich bietet HTML Schnittstellen zu Erweiterungssprachen wie Stylesheets oder Javascript, durch die der Entwickler in die Lage versetzt wird, HTML-Elemente persönlich gestalten zu können oder Interaktionen mit dem Anwender zu realisieren.

Bei der Sprache HTML handelt es sich um ein so genanntes Klartextformat. Das bedeutet in diesem Zusammenhang, dass zur Erstellung von nutzbaren HTML-Dateien keine bestimmte Softwareanwendung notwendig ist, sondern dass ein Web-Projekt durchaus mit Hilfe eines einfachen Editors erstellbar ist. Aus dieser Tatsache ergibt sich die Situation, dass sich HTML sinnvoll mit Hilfe von Programmen generieren lässt.

HTML wird als Auszeichnungssprache bezeichnet. Eine solche hat die Aufgabe, logische Bestandteile eines textorientierten Dokuments zu beschreiben. Dem Entwickler werden hierbei Möglichkeiten geboten, alle Elemente eines textuellen Dokuments zu beschreiben, wie beispielsweise Überschriften, Tabellen, Listen oder Referenzen. Innerhalb der Spracharchitektur herrscht eine hierarchische Gliederung vor, die sich auf die Dokumentenstruktur bezieht. Hierbei besitzen Dokumente globale Eigenschaften, die so genannten Kopfdaten. Innerhalb der einzelnen Dokumente besteht der Inhalt beispielsweise aus einer Überschrift, Textabsätzen oder Tabellen. Elemente dieser Art können wiederum untergeordnete Elemente besitzen, wie beispielsweise die Schriftart eines Textabsatzes.

Eine weitere Möglichkeit der Sprache HTML ist die Nutzung von Verweisen zwischen einzelnen Web-Seiten. Diese werden als Hyperlinks bezeichnet und können Elemente innerhalb eines Projektes referenzieren, jedoch auch Seiten außerhalb der eigenen Plattform.

3.4.2 PHP

HTML ist zwar eine mächtige Dokumentbeschreibungssprache, jedoch lassen sich keine dynamisch generierten Inhalte erzeugen, die beispielsweise durch Abfragen aus Datenbank gewonnen werden können. Zur Darstellung dieser erweiterten Inhalte ist die Verwendung einer so genannten Skriptsprache nötig.
Um Daten aus einer Datenbank darstellen zu können, muss eine server-seitige Verarbeitung stattfinden, da die Daten zentral gespeichert sein müssen, um sie jedem Nutzer zu Verfügung zu stellen.

Eine solche Möglichkeit bietet die Sprache PHP (Hypertext Preprocessor) [8], die 1994 von Rasmus Lerdorf entwickelt wurde.
PHP-Programme bzw. Skripte bestehen aus normalen HTML-Dokumenten mit eingebetteten PHP-Anweisungen. Der umliegende HTML-Code ist statischer Inhalt des Dokumentes und wird unverändert übernommen bzw. an den Web-Server weitergereicht. Die PHP-Anweisungen werden durch den PHP-Interpreter ausgeführt und die dabei erzeugten Ausgaben werden an Stelle der Anweisungen in das Dokument eingefügt. Die Kommunikation mit dem Web-Server verläuft über eine CGI-Schnittstelle.

Die Syntax von PHP ist ähnlich der von C und Java und daher für Entwickler, die aus diesem Bereich kommen, relativ schnell zu erlernen. Die aus diesen Programmiersprachen bekannten Kontroll-Strukturen, wie zum Beispiel die For- Schleife und bedingte Verzweigungen sind ebenso vorhanden wie die Standard-Datentypen int, boolean und string. Eine Besonderheit ist, dass Variablen nicht explizit deklariert werden und untypisiert sind. Der Typ einer Variable wird zur Laufzeit vom Interpreter bestimmt und kann sich im Verlauf der Lebensdauer einer Variable ändern.

Eine weitere nützliche Funktionalität von PHP ist die Unterstützung von regulären Ausdrücken, wie man sie von Perl kennt. Damit lassen sich flexibel Muster in Texten erkennen und sie sind damit bei der Verarbeitung von Zeichenketten ein nützliches Hilfsmittel.

Ein großer Vorteil der Sprache PHP liegt in der Tatsache, dass der PHP-Interpreter bereits viel Funktionalität besitzt und nicht durch zusätzliche Module aufgewertet werden muss. Zusätzlich ist der Interpreter für alle gängigen Plattformen verfügbar. Dies macht die Verwendung sehr einfach und unkompliziert.

Mit PHP ist die Entwicklung von Web-Anwendungen sehr schnell und einfach durchführbar. Die Sprache PHP stellt sich als sehr mächtig dar. Weiterhin sprechen die hohe Verbreitung und die hohe Unterstützung durch Internet-Service-Provider (ISP) für PHP. Ein großer Nachteil ist die notwendige Einbettung von PHP-Anweisungen in HTML-Dokumente. Dies macht eine saubere Trennung zwischen Anwendungscode und Darstellung unmöglich und reduziert deren Wiederverwendbarkeit. Somit sind Web-Anwendungen mit hohem algorithmischen Anteil weniger das Anwendungsfeld von PHP. Die schlechte Skalierung macht den Einsatz von PHP in Bereichen wie den eCommerce unattraktiv. Die geringe Objektorientiertheit und fehlende Datenkapselung verlangt dem Programmierer viel Disziplin ab, um sauberen und halbwegs wieder verwendbaren Programmcode zu entwickeln. Somit liegt das Einsatzgebiet von PHP insbesondere im Rapid-Prototyping und der schnellen Entwicklung von kleinen bis mittelgroßen Web-Anwendungen mit geringem Anwendungscode.

3.5 Stylesheets (CSS)

Neben der Strukturierung von Texten bietet HTML auch die Möglichkeit Texte mit zusätzlichen Formatierungsangaben zu versehen. Dies können zum Beispiel Angaben über die zu verwendenden Schriftart oder Zeichenfarbe bei der Darstellung des Dokumentes sein. Ein schwerwiegender Nachteil dieser Technik ist die fehlende Trennung von Text (Inhalt) und Darstellungsangaben (Formatierung). Die direkte Kodierung von Formatierungsangaben in HTML-Dokumenten erschwert deren Wartbarkeit und Wiederverwendbarkeit. Insbesondere wenn die Darstellungseigenschaften mehrer oder gar hunderter HTML-Dokumente konsistent gehalten werden müssen. Eine Lösung für dieses Problem ist die Verwendung von Cascading Stylesheets (CCS) [10]. CSS ist eine einfache Auszeichnungssprache, die man benutzen kann, um Formatierungsangaben in zusammenhängenden Anweisungs-Blöcken zu definieren. Diese Blöcke kann man einzelnen HTML-Elementen zu weisen. Ein großer Vorteil von CSS ist die Möglichkeit der Auslagerung von CSS-Blöcken in separate Dateien. Diese CSS-Dateien können von HTML-Dokumenten referenziert werden. Dadurch erreicht man zum einen eine strikte Trennung von Inhalt und Darstellung und zum anderen die einfache Möglichkeit der besseren Adaptierbarkeit der Darstellung.
Aufgrund der Vorteile von CSS wird in PaderWAVE bei der Codeerzeugung von Web-Anwendungen für die Formatierung von HTML-Elementen CSS verwendet.

4 Entwurf

4.1 Ziele des Systems

Zu Beginn der ersten Projektphase wurden die Ziele des zu entwickelnden Systems definiert. Dabei wurde unter anderem festgelegt, welchen Nutzen der spätere Benutzer des Systems haben soll. Dies schließt die Festlegung der wichtigsten Funktionalitäten ein, die das spätere System dem Benutzer zur Verfügung stellt. Dazu war es notwendig, als erstes ein Profil des späteren Benutzers zu bestimmen. Der typische Benutzer hat Grundkenntnisse im Bereich der Erstellung von Web-Anwendungen. Dabei muss er über kein Expertenwissen verfügen, insbesondere was deren technische Realisierung betrifft. Der vertraute Umgang mit Modellierungswerkzeugen ist hilfreich, aber keine Grundvoraussetzung. Aufbauend auf diesem Benutzer-Profil wurden die Ziele des Systems formuliert.

Eines der wichtigsten Ziele des Systems ist die Kapselung von Aspekten der technischen Realisierung. Die technische Umsetzung der Web-Anwendung mit Hilfe von HTML und PHP wird komplett von der visuellen Umgebung übernommen. Es sind also seitens des Benutzer keine Fähigkeiten in diesem Bereich erforderlich. Der Benutzer soll sich ganz auf den Inhalt der Web-Anwendung und nicht auf die Technik konzentrieren können.

Um jedoch das System möglichst flexibel und offen zu halten, muss es dem technisch versierten Benutzer möglich sein, eigenen Anwendungs-Code, zum Beispiel in Form eines PHP-Skriptes, in die Web-Anwendung zu integrieren. Aus diesem Grund ist es für das geplante System wichtig, klare Schnittstellen zwischen generierten Anwendung-Code und benutzereigenen Anwendungs-Code zu schaffen. Es muss zum Beispiel möglich sein, zur Laufzeit der Web-Anwendung innerhalb von benutzerdefinierten Skripten auf generierte Variablen zugreifen zu können. Diese Funktionalität muss im Sprachentwurf berücksichtigt werden.

Ein weiteres wichtiges Ziel ist die Entwicklung einer intuitiv bedienbaren Benutzungsoberfläche. Die Modellierung muss schnell und einfach sein und auch bei komplexen Anwendungen darf der Benutzer nicht den Überblick verlieren.

Der Lernaufwand für die Benutzung der visuellen Umgebung muss möglichst gering

gehalten werden. Die Einarbeitungszeit des Benutzers soll durch eine intuitiv bedienbare Benutzungsoberfläche und eine schlanke und gut strukturierte visuelle Sprache minimiert werden.

Robustheit und Korrektheit haben einen ebenso hohen Stellenwert. Die umfasst zum einen die Robustheit der visuellen Umgebung selbst, als auch die Robustheit und Korrektheit des generierten Codes. Es muss sichergestellt werden, dass der Benutzer aus unvollständigen oder fehlerhaften Spezifikationen keinen Anwendungs-Code generieren kann. Dazu müssen Konsistenz-Checks zur Überprüfung der Spezifikation der Web-Anwendung eingebaut werden, sowie die Korrektheit der verwendeten Code-Muster bei der Code-Generierung gewährleistet sein.

4.2 Einsatzgebiete

PaderWAVE soll dazu genutzt werden, kleine bis mittel-große Web-Anwendungen zu erstellen. Dies können Web-Anwendungen sein, die eine Datenbank zu Haltung persistenter Daten nutzen, wie zum Beispiel ein Job-System. Für ein Job-System sind auch verschiedene Benutzer-Rollen, wie zum Beispiel Gast oder Kunde erforderlich. Der dafür notwendige Zugriffsmechanismus kann auch in PaderWAVE spezifiziert werden.
Durch die schnelle Entwicklungsgeschwindigkeit mit PaderWAVE lassen sich schnell lauffähige Prototypen (RAPID-Prototyping), zum Beispiel zu Präsentationszwecken, erstellen. Die Machbarkeit von geplanten Web-Anwendungen lässt sich auch schnell prüfen.

5 Sprachentwurf

5.1 Beispiel einer Web-Anwendung

Als erstes betrachten wir die Web-Anwendung „Aufgabenplaner“. Bei dieser Anwendung können Anwender sich als Benutzer registrieren und ihre „Zu erledigen“- Liste erstellen. Die Web-Anwendung besitzt drei Zugriffsbereiche. Um auf den Inhalt eines Bereiches zugreifen zu dürfen, muss der Anwender die erforderlichen Rechte besitzen.

Der erste Bereich ist für alle Anwender zugänglich. Dazu gehören die Startseite und die Registrierungsseite des Aufgabenplaners. Die Startseite ist in Abbildung 5.1 dargestellt. Auf dieser befindet sich im oberen Teil die Navigationsleiste. Diese enthält Links zu allen Web-Seiten der Anwendung, deren Nutzen für den Anwender im Verlauf dieses Abschnittes erläutert wird.

Abbildung 5.1: Startseite des Aufgabenplaners

Auf der Registrierungsseite (siehe Abbildung 5.2) kann jeder Anwender der Web-

Anwendung sich selbst als Benutzer registrieren. Bei der Registrierung muss der Anwender seinen *Benutzernamen*, *Passwort*, *Vor- und Nachnamen* eintragen. Zusätzlich muss noch angegeben werden, ob man als Administrator, Gast oder Benutzer registriert werden will. Anwender die sich registriert haben, können neue Einträge erstellen.

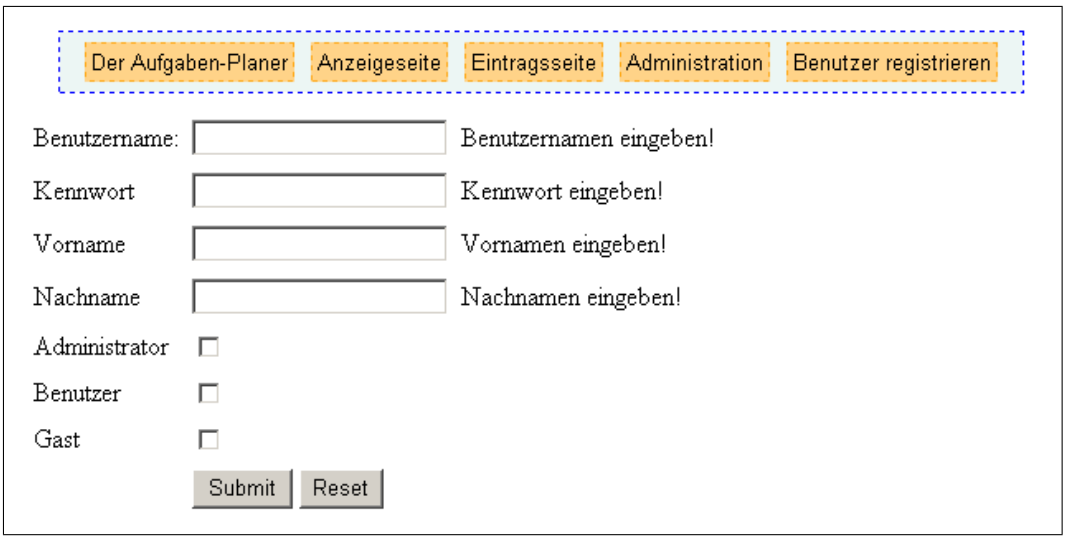

Abbildung 5.2: Registrierungsseite des Aufgabenplaners

Darüber hinaus existiert in diesem Bereich die Anzeigeseite (Abbildung 5.3). Auf dieser Web-Seite kann man sich alle Einträge der Aufgabenliste anzeigen lassen, sowie neue Einträge hinzufügen.

Abbildung 5.3: Anzeigeseite des Aufgabenplaners

Der zweite Bereich ist für registrierte Benutzer gedacht. Die Eintragseite befindet sich in diesem Bereich und ist aus diesem Grund nur für registrierte Benutzer zugänglich. In Abbildung 5.4 ist das Formular dargestellt, mit dem man neue Einträge erstellen kann. In dem ersten Feld kann man die Priorität der Aufgabe festlegen. Als nächstes gibt man seinen Namen ein. In dem letzten Feld des Formulars wird eine kurze Beschreibung der Tätigkeit eingetragen.

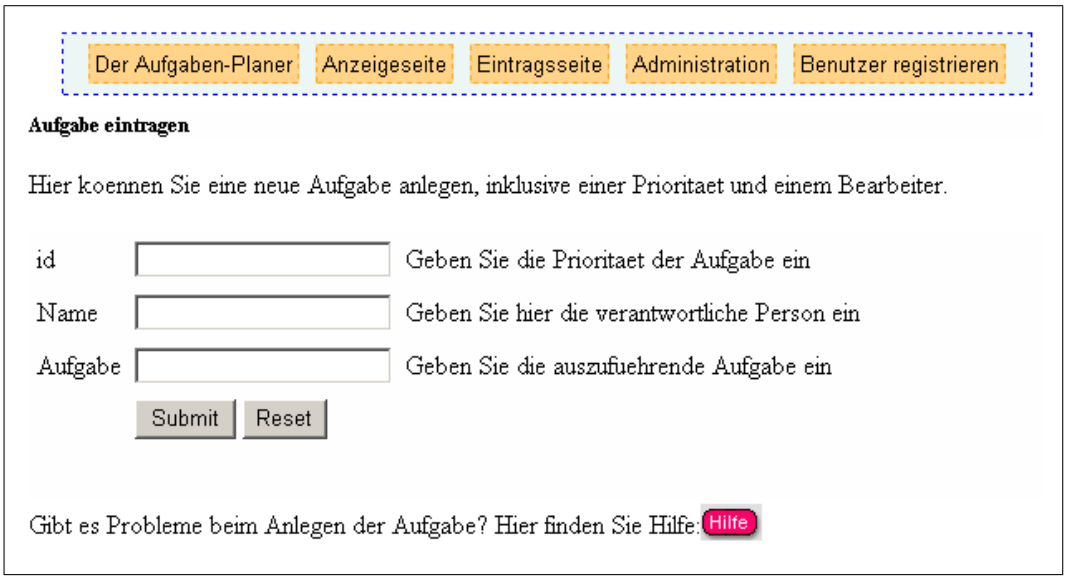

Abbildung 5.4: Eintragseite des Aufgabenplaners

Versucht ein nicht angemeldeter Anwender diesen Bereich zu betreten, wird er automatisch auf die Anmeldeseite weitergeleitet. Dort kann er sich mit seinem registrierten Benutzernamen und Passwort anmelden, um anschließend Einträge vornehmen zu können.

Der dritte Bereich ist nur für Anwender mit Administrator-Rechten zugänglich. In diesen Bereich gelangt man über den Link *Administration* aus der Navigationsleiste. In Abbildung 5.5 ist die Werkzeugseite des Administrators dargestellt. Als Administrator kann man sich alle registrierten Benutzer der Web-Anwendung anzeigen lassen, sowie neue Benutzer registrieren.

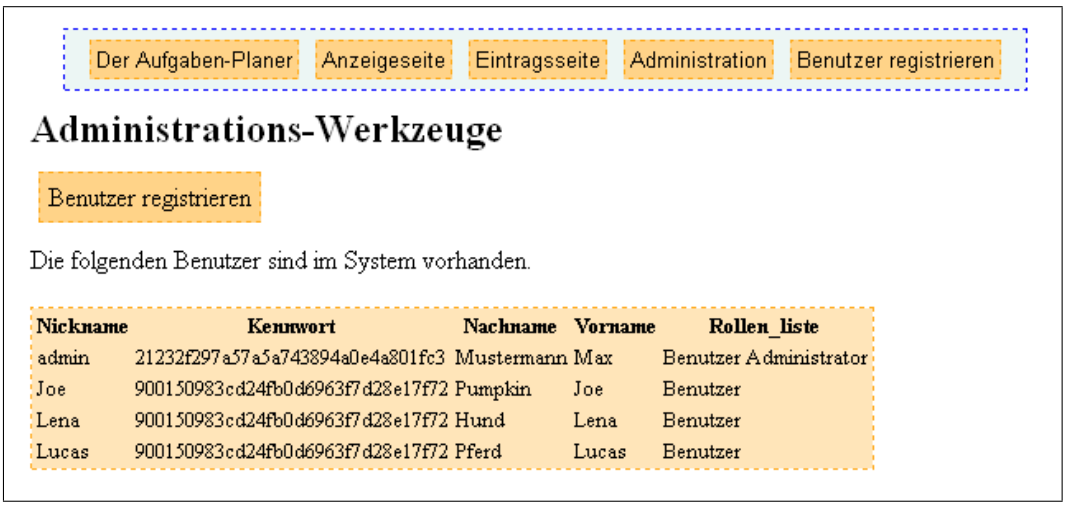

Abbildung 5.5: Administratorseite des Aufgabenplaners

5.2 Visuelle Entwicklungsumgebung (PaderWAVE)

Das System PaderWAVE ist ein Strukturediteor mit integriertem Generator. Beim Starten des Strukturediteors und erzeugen einer neuen Spezifikation bzw. öffnen einer zuvor gesicherten wird die Hauptsicht „RootView“ geöffnet. In Abbildung 5.6 ist die Hauptsicht einer neu angelegten Spezifikation dargestellt.

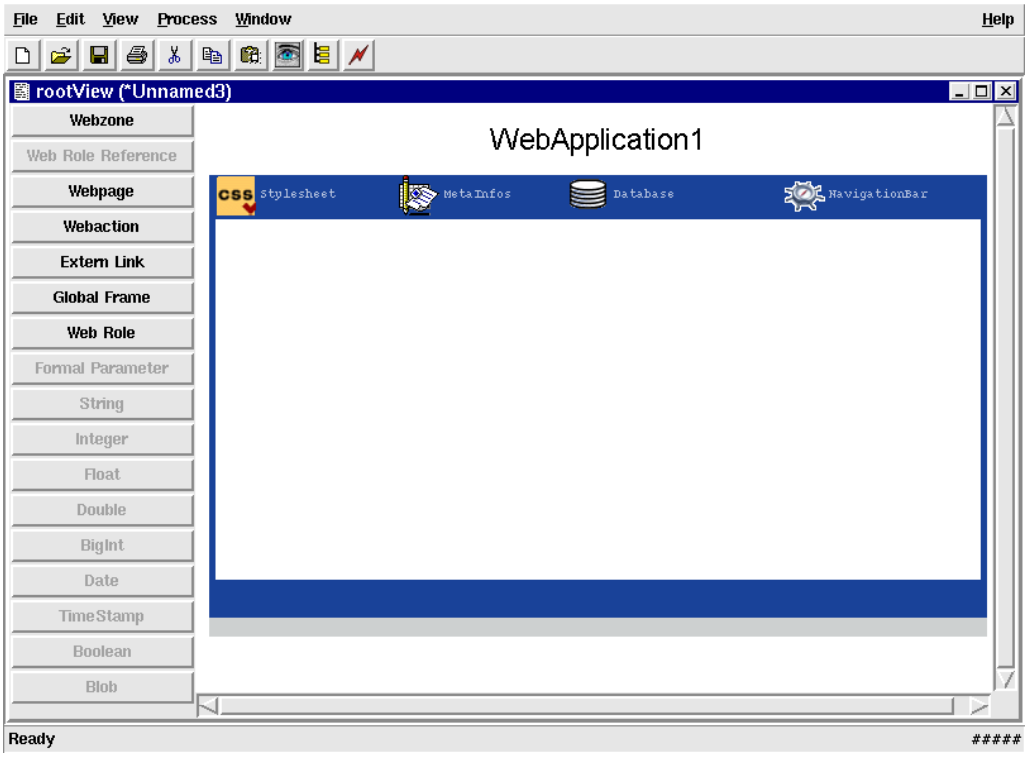

Abbildung 5.6: Hauptsicht des Aufgabenplaners

Wie zu sehen ist, enthält diese noch keine Sprachelemente. In der Hauptsicht, wie auch in den anderen Sichten, existieren Schalter. Sie befinden sich auf der linken Seite des Sichten-Fensters. Mit Hilfe dieser können einer Web-Anwendung Sprachelemente hinzugefügt werden. Dazu muss der Benutzer den Schalter des gewünschten Sprachelementes betätigen und das darauf hin im Feld erscheinende Element entsprechend platzieren. Jedes Sprachelement besitzt Eigenschaften, wie zum Beispiel ein Name-Attribut. Diese Eigenschaften sind in einem Editierdialog bearbeitbar. Dieser Dialog wird durch das Drücken der rechten Maustaste und das Auswählen des Menüpunktes „Edit Attribute“ geöffnet. In Abbildung 5.7 ist der Editierdialog für das Sprachelement „WebPage“ dargestellt.

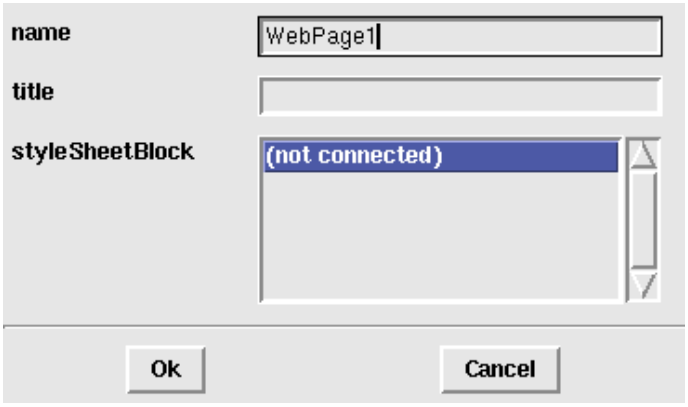

Abbildung 5.7: Editierdialog für Web-Seiten

Um das visuelle Spezifizieren von Web-Anwendungen für den Entwickler übersichtlicher zu gestalten, gibt es weitere Sichten in PaderWAVE. Dies sind:

- *Stylesheetsicht(Stylesheet)*: In dieser Sicht können Stylesheetblöcke für die gesamte Web-Anwendung definiert werden.
- *Datenbanksicht(Database)*: In dieser Sicht können Datenbank-Parameter, Datenbank-Tabellen, Datensätze, Daten-Filter und Datenbank-Operationen definiert werden.
- *Navigationssicht(NavigationBar)*: In dieser Sicht kann festgelegt werden, welche Seiten in der Navigationsleiste erscheinen sollen. Wenn man eine Web-Seite in die Navigationsleiste eingefügt hat, ist es sofort in der Hauptsicht anhand des Navigationszeichens links neben der Web-Seite zu sehen.
- *Metainformationssicht(MetaInformation)*: In dieser Sicht können die globalen Meta-Informationen der Web-Anwendung definiert werden.

Außer diesen vier Untersichten existieren noch Detailsichten für die wichtigsten Sprachelemente der Spezifikationssprache. Unter anderem für die Sprachelemente „Webpage“ und „Webaction“.

5.3 Navigationsstruktur

Die Hauptsicht (RootView) ist die wichtigste Sicht in dem PaderWAVE-System. Sie enthält unter anderem die Visualisierung der Web-Seiten und Web-Zonen der Web-Anwendung. Web-Seiten sind die Grundbausteine einer Web-Anwendung. Die Web-Seiten können individuell mit Inhalt wie Texte und Bilder gefüllt werden. Durch die Verknüpfung der Web-Seiten untereinander, kann eine Web-Dokumenten-Struktur aufgebaut werden, die dem späteren Benutzer der Web-Anwendung das Navigieren durch die Web-Anwendung ermöglicht. Für die Erstellung einer Navigationsstruktur gibt es in PaderWAVE ein eigenes Sprachelement, die „NavigationBar“, welches die Erstellung erleichtert. Mehr zu der NavigationBar findet man im Abschnitt 5.6.

Für die Modellierung von Web-Seiten gibt es in PaderWAVE das Sprachelement „WebPage“. In Abbildung 5.8 ist die Spezifikation der Benutzer-Web-Zone der *Aufgabenplaner-Anwendung* zu sehen.

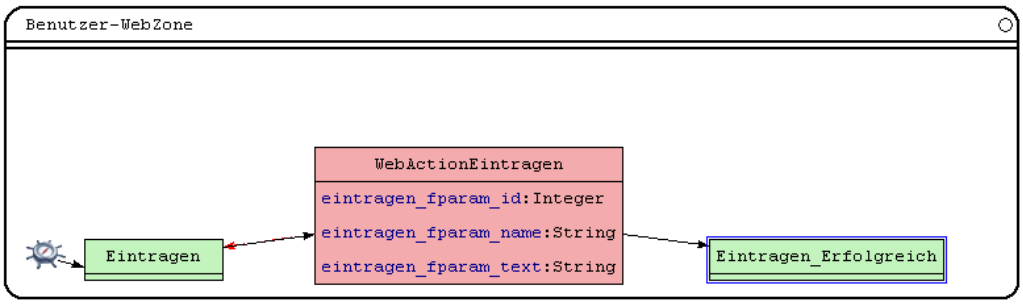

Abbildung 5.8: Web-Seiten in der Benutzer-Web-Zone

Die „WebZone“ ist ein spezielles Sprachkonstrukt in PaderWAVE, um verschiedene Zugriffsbereiche modellieren zu können. Nur Benutzer der Web-Anwendung, welcher die entsprechenden Zugriffrechte (Web-Rolle) einer Web-Zone besitzen, dürfen auf die darin enthaltenen Web-Seiten der Web-Anwendung zugreifen. Das Prinzip der Web-Zones wird in Abschnitt 5.4 erläutert. Wie der Abbildung zu entnehmen ist, befinden sich in der Benutzer-Web-Zone zwei Web-Seiten-Elemente. Dies sind die Web-Seiten „Eintragen“ und „Eintrage_Erfolgreich“. Web-Seiten werden in der visuellen Umgebung durch ein grünes Rechteck dargestellt. Web-Seiten können im Modell von anderen Elementen mit Hilfe eines *Links* verknüpft werden.

Weiterhin ist es möglich in der Hauptsicht Web-Actions einzufügen. Web-Actions dienen dazu, benutzerdefinierte PHP-Skripte in die Web-Anwendung konform einzubinden. In Abschnitt 5.9 werden die Web-Actions genauer beleuchtet. Web-Seiten und Web-Actions können formale Parameter besitzen. Diese werden über den Schalter „Formal Parameter“ der gewünschten Web-Seite oder Web-Action hinzugefügt. Formale Parameter werden unter anderem dazu genutzt, um zur Laufzeit

der Web-Anwendung aus einem Formular gewonnene Daten an ein Web-Action-Element zu übergeben, welches die Daten verarbeitet. Diese Parameterübergabe wird in der Hauptsicht mit Hilfe von schwarzen Navigationspfeilen zwischen zwei Web-Seiten oder einer Web-Seite und einer Web-Action dargestellt. Diese werden automatisch eingefügt, sobald ein formaler Parameter übergeben wird. Der Typ der formalen Parameter wird anhand der Schalter „String“, „Integer“ usw. festgelegt.

Die Sicht für die Web-Seitensicht enthält weitere Schalter, welche die verfügbaren Sprachelemente einer Web-Seite darstellen. Web-Seiten können damit durch visuelle Elemente wie Formular, Header, Paragraph, Tabelle usw. erweitert werden.

Eine weitere Detailsicht ist die Web-Action-Sicht. Hier wird eine Web-Action aus der Hauptsicht vollständig beschrieben und zwar auf welche weitere Web-Seite (im Erfolgsfall) oder eine Errorseite (im Fehlerfall) sie verweisen soll und welches PHP-Skript aufgerufen wird. Das Skript kann der Benutzer selbst in einem Editorfenster eingeben.
Wenn der Benutzer in seiner Web-Anwendung auf andere externe Web-Seiten verweisen möchte, hat er die Möglichkeit dies mit Hilfe des Schalters „Extern Link“ zu tun. Hierfür muss er den Namen und die URL eingeben.

5.4 Benutzerauthentifikation (geschützte Bereiche)

Ein zentraler Punkt dynamischer Web-Anwendungen ist die Integration einer Benutzerauthentifikation. Hierbei handelt es sich um die Möglichkeit, sichere Bereiche in die eigene Web-Anwendung aufzunehmen, so dass nicht jeder potenzielle Benutzer dieselben Zugriffsrechte besitzt. Die unterschiedlichen Rechte beziehen sich auf die spezifizierten Web-Seiten.

Für die Modellierung von unterschiedlichen Zugriffsbereichen gibt es in PaderWAVE die „WebZones“. Um Zugriffsbereiche mit Hilfe von „WebZones“ festzulegen, können diesen zuvor definierte Rollen zugewiesen werden. Rollen können in der Hauptsicht mit dem Sprachkonstrukt „WebRole“ definiert werden. Web-Zonen können mehrere Rollen zugewiesen werden. Damit haben alle Benutzer der Web-Anwendung, welche mindestens eine dieser Rollen besitzen, Zugriff auf die in der Web-Zone enthaltenen Web-Seiten. Wird einer Web-Zone keine Rolle zugewiesen, so kann jeder Benutzer der Web-Anwendung auf dessen Inhalt zugreifen. Damit können ungeschützte Bereiche modelliert werden. Dies ist zum Beispiel sinnvoll bei Web-Seiten die für jeden frei zugänglich sein sollen oder sogar müssen, wie zum Beispiel die Anmeldeseite einer Web-Anwendung. Oft kommt es in Anwendungen vor, dass sich Zugriffsbereiche überschneiden bzw. ein Benutzer mehr können soll als andere. Zu diesem Zweck gibt es ein Vererbungs-Mechanismus. Es ist möglich Web-Zonen ineinander zu schachteln. Dabei haben Benutzer, die Zugriff auf einer innenliegenden Web-Zone besitzen, Zugriff auf die außenliegenden Web-Seiten. Es

besteht eine *Is-A* Beziehung zwischen innen- und außenliegender Web-Zone.

Das Konzept der Benutzerauthentifikation führt sich auch auf die Verwendung der Navigationsleiste fort. Hierbei ist zu beachten, dass auf jeder Seite nur jeweils die Navigationsverweise angezeigt werden, für die der Benutzer ausreichende Rechte besitzt. Im Beispiel wird also vor dem Login-Vorgang für aktuelle Benutzer der Verweis zum Eintragen in den Aufgabenplaner, nicht jedoch der Zugang zum Administrationsbereich angezeigt.

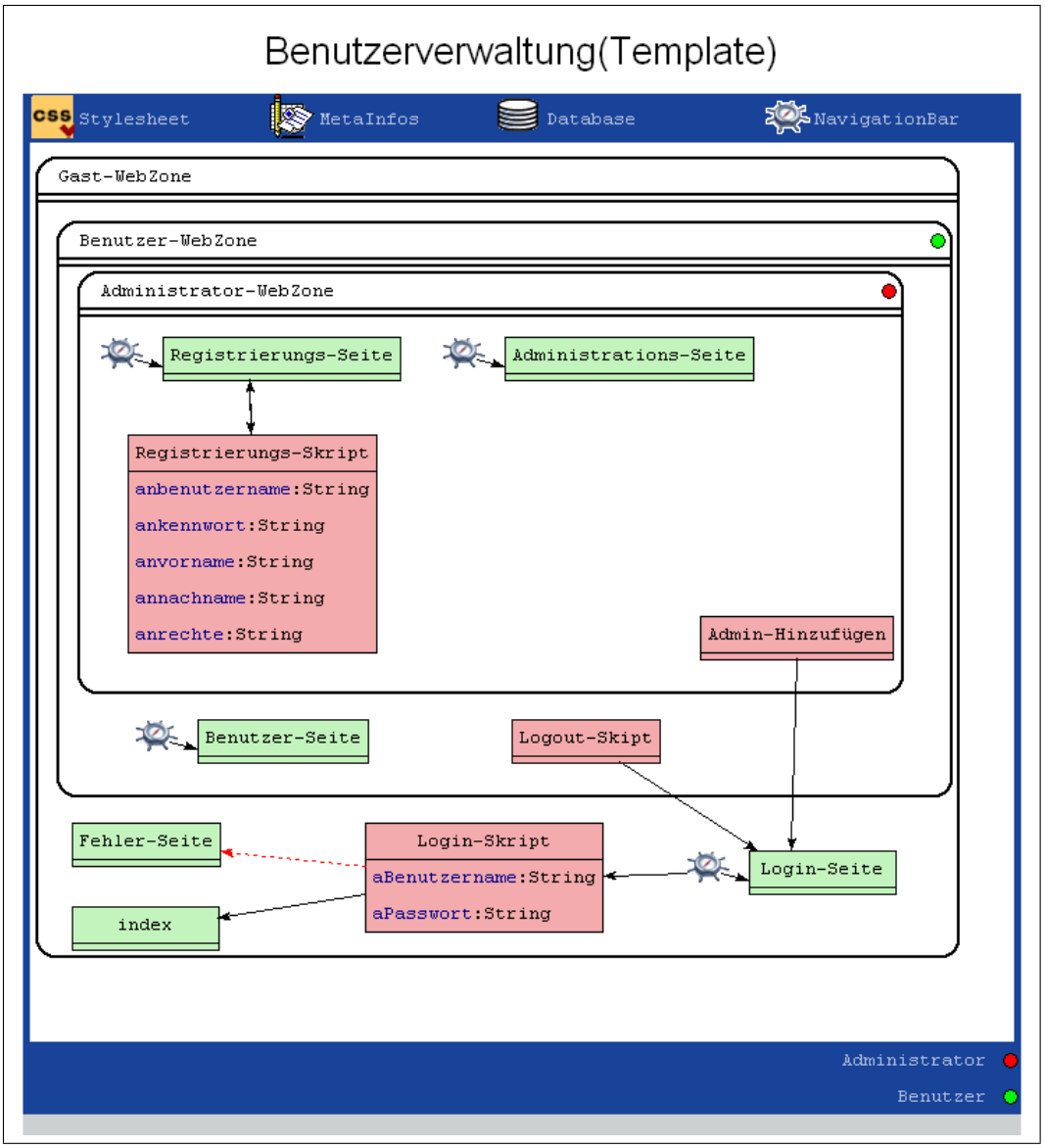

Abbildung 5.9: Benutzer-Verwaltungs-Template

Um den Benutzer die Integration einer Benutzerverwaltung in eigenen Web-

Anwendungen zu erleichtern, wird ein Template (userAdministrator) bereitgestellt, das die erforderliche Funktionalität besitzt. Das Template enthält alle notwendigen Mechanismen, wie Datenbanken zur Verwaltung der Benutzer und Web-Seiten/Skripte zur Implementierung der Schutzmechanismen. In Abbildung 5.9 ist die Hauptsicht dieses Templates abgebildet. Der Abbildung ist zu entnehmen, dass in dem Template bereits zwei Rollen existieren. Dies sind die Rolle des *Administrators* und die des *Benutzers*. Zur besseren visuellen Unterscheidung, können Rollen unterschiedliche Farben zugeordnet werden. Der Gast-Web-Zone in dem Template ist keine Rolle zugewiesen. Deren Inhalt ist damit für jeden Anwender der Web-Anwendung frei zugreifbar. Des Weiteren dürfen Anwender, welche die Administrator-Rolle besitzen, auf Web-Seiten der Benutzer-Web-Zone zugreifen.

Zur Speicherung der Rechte der Benutzer ist eine Datenbanktabelle (Abbildung 5.10) angelegt. Diese enthält alle notwendigen Benutzerdaten, wie den Namen und Vornamen, den gewählten Benutzernamen mit verschlüsseltem Passwort (md5-Verschlüsselung) und die Rechte. Die Rechte liegen in Form eines Strings vor, der alle Rollen-Namen enthält, die dem Benutzer entsprechend zugewiesen wurden.

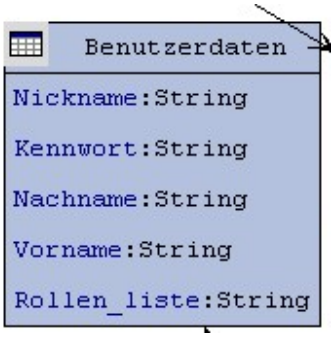

Abbildung 5.10: Datenbanktabelle zur Verwaltung der Rechte

Jede Web-Seite, die einen Schutz benötigt, besitzt vor Ausführung des Codes einen Abschnitt, der die gültigen Rechte des Benutzers überprüft. Diese ergeben sich für jeden Benutzer aus einer Session-Variable, die während der gesamten Nutzung der Anwendung besteht. Das Setzen der Variable geschieht während des Login-Mechanismus (Login-Seite/Login-Skript). Hierbei handelt es sich um eine Web-Seite mit zugehörigem Skript, das zur Verfügung gestellt wird. Ist der Benutzer nicht authentifiziert, ist die angesprochene Variable nicht gesetzt und der Benutzer wird auf eine Fehlerseite verwiesen. Nach dem Login-Vorgang wird während des Authentifizierungsmechanismus das in der Datenbank gespeicherte Recht mit dem in der aktuellen Session-Variable verglichen. Besitzt der Benutzer mindestens eine der geforderten Rollen, wird die aufgerufene Seite angezeigt. Andernfalls, wird der Aufrufer zu einer vorher festgelegten Fehlerseite, bestimmt durch das Attribut *errorLink*, weitergeleitet.
Jeder Benutzer hat die Möglichkeit, sich nach dem Login-Vorgang von der

Web-Anwendung wieder abzumelden, indem er den Logout-Mechanismus aufruft. Hierbei werden die Session-Variablen gelöscht, die vorher den Zugang zu Seiten gesteuert haben. Dieser Mechanismus ist im Template (Logout-Skript) ebenfalls implementiert.

Um neue Benutzer zur Laufzeit der Web-Anwendung hinzufügen zu können, befinden sich in der Administrator-Zone die *Registrierungs-Seite* und das *Registrierungs-Skript*. Auf diese hat nur ein Benutzer mit der Rolle Administrator Zugriff. Die Rechte sind über Auswahllisten zuzuweisen. Alle verfügbaren Rollen in der Web-Anwendung werden dynamisch über eine globale Variable generiert. Zusätzlich müssen die Felder der Datenbanktabelle über Formulareinträge gefüllt werden. Der initiale Administrator muss einmalig über ein Skript (Admin-Hinzufügen) angelegt werden, das zur Verfügung steht.

Der Designer der Web-Anwendung hat die Möglichkeit, beliebig viele Web-Zonen einzufügen. In diesem Fall muss das bestehende Template nicht verändert werden. Die Umbenennung oder Erweiterung der zur Verfügung gestellten Web-Seiten, Skripte oder Formularfelder ist möglich, muss jedoch durchgängig in der gesamten Web-Anwendung fortgeführt werden. Zur Überprüfung der Benutzerrechte, wird in dem generierten PHP-Code davon ausgegangen, dass in der Session-Variable *$_SESSION['Rollen_liste']* die Rechte des Benutzer definiert sind. Diese Randbedingung muss bei eigenen Anpassungen des Login-Skripts berücksichtigt werden.

5.5 Web-Seiten und Komponenten

Die Web-Seiten-Elemente selbst enthalten nur wenige Attribute, die inhaltlichen Wert für die Web-Anwendung haben. So kann neben dem Titel der Web-Seite nur ein Verweis auf ein Stylesheet-Block festgelegt werden. Damit lassen sich globale Formatierungseigenschaften für Web-Seite festlegen.

Um den Inhalt einer Web-Seite zu beschreiben, bietet PaderWAVE die Möglichkeit Meta-Informationen zu definieren. Dazu kann er in der Sicht „Meta-Information“ (siehe Abbildung 5.11) zusätzliche Informationen definieren, um welche die Web-Seiten angereichert werden. Die so definierten Meta-Informationen werden global für alle Web-Seiten festgelegt. Die Meta-Informationen können zum Beispiel von Suchmaschinen verwendet werden, um ein besseres Profil über den Inhalt der Web-Seiten erstellen zu können. Mit dem Schalter „Attribute/Value Pair“ können die Meta-Informationen in Form von Attribut-Werte-Paaren, zum Beispiel **„Author = Joe“** angelegt werden.

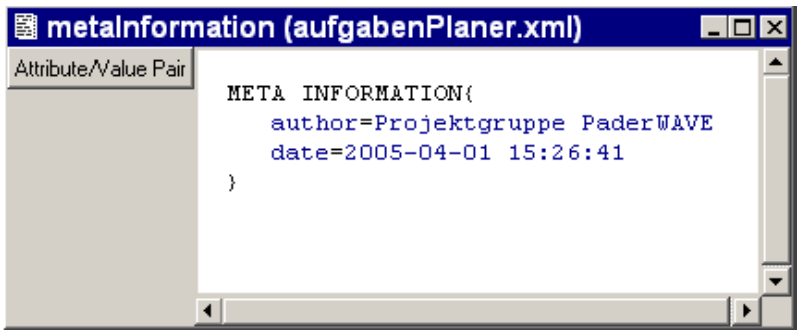

Abbildung 5.11: Meta-Informationen

Nachdem die statische Struktur der Web-Anwendung mit Hilfe von Web-Seiten modelliert wurde, kann man diese mit dem gewünschten Inhalt füllen. Zu diesem Zweck kann man jeder Web-Seite weitere Elemente zuweisen. Diese Elemente, in PaderWAVE als Komponenten bezeichnet, sind zum Beispiel Textblöcke, Überschriften, Bilder und Formulare. In Abbildung 5.12 sind alle in PaderWAVE verfügbaren Komponenten aufgeführt.

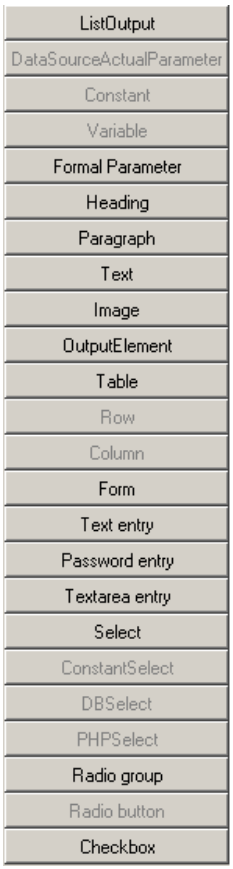

Abbildung 5.12: Verfügbare Komponenten in PaderWAVE

Der Abbildung ist zu entnehmen, dass einige Komponenten, wie zum Beispiel die

„Radio button“-Komponente, ausgegraut sind. Das liegt daran, dass einige Komponenten nur in einem bestimmten Kontext verwendet werden können. So muss in unserem Beispiel zunächst eine Komponente von Typ „Radio group“ hinzugefügt werden. Erst dann kann man dieser Gruppe die einzelnen *Buttons* mit Hilfe der „Radio button“-Komponente hinzufügen. Neben den visuellen Komponenten, wie Text- und Bild-Komponente, gibt es noch ein Element zur Strukturierung von Komponenten. Dies ist die „Table“-Komponente. Mit Hilfe dieser ist es dem Benutzer möglich, Komponenten zu gruppieren und in einer bestimmten Reihenfolge oder Position auf der Web-Seite anzuordnen. Insbesondere in Kombination mit Stylesheets lässt sich die Strukturierung von Komponenten auf einer Web-Seite einfach umsetzen. Auf die genaue Verwendung von Tabellen wird im späteren Verlauf noch eingegangen.
Des Weiteren gibt es Komponenten, die keine visuelle Repräsentation in der Web-Anwendung besitzen. So zum Beispiel die „FormalParameter“-Komponente. Diese kann verwendet werden, um dynamische Daten von einer Web-Seite zu einer anderen Web-Seite weiterzureichen. Dies können zum Beispiel Daten sein, die in einem Formular eingegeben worden sind und auf einer weiteren Web-Seite dargestellt oder verarbeitet werden sollen. Mehr zu diesem Thema findet man im Abschnitt 5.10.
Nach einem ersten groben Abriss werden die Komponenten nun auf den nachfolgenden Seiten im Detail erläutert.

5.5.1 Texte und Bilder

Zwei der häufigsten Bestandteile einer Web-Anwendung sind Texte und Bilder. In PaderWAVE gibt es zwei Arten von Texten. Zum einen gibt es Überschriften (Heading) und zum anderen Fließ-Text, welcher in einem Paragraphen eingebettet wird. Überschriften sind spezialisierte Texte, für die vordefinierte Formatierungsvorlagen existieren. Diese Formatierungsvorlagen sind bereits in HTML verankert und werden automatisch verwendet, wenn nichts anderes vom Benutzer definiert worden ist. In Abbildung 5.13 ist die Spezifikation der Hauptseite der Aufgabenplaner-Anwendung zu sehen.

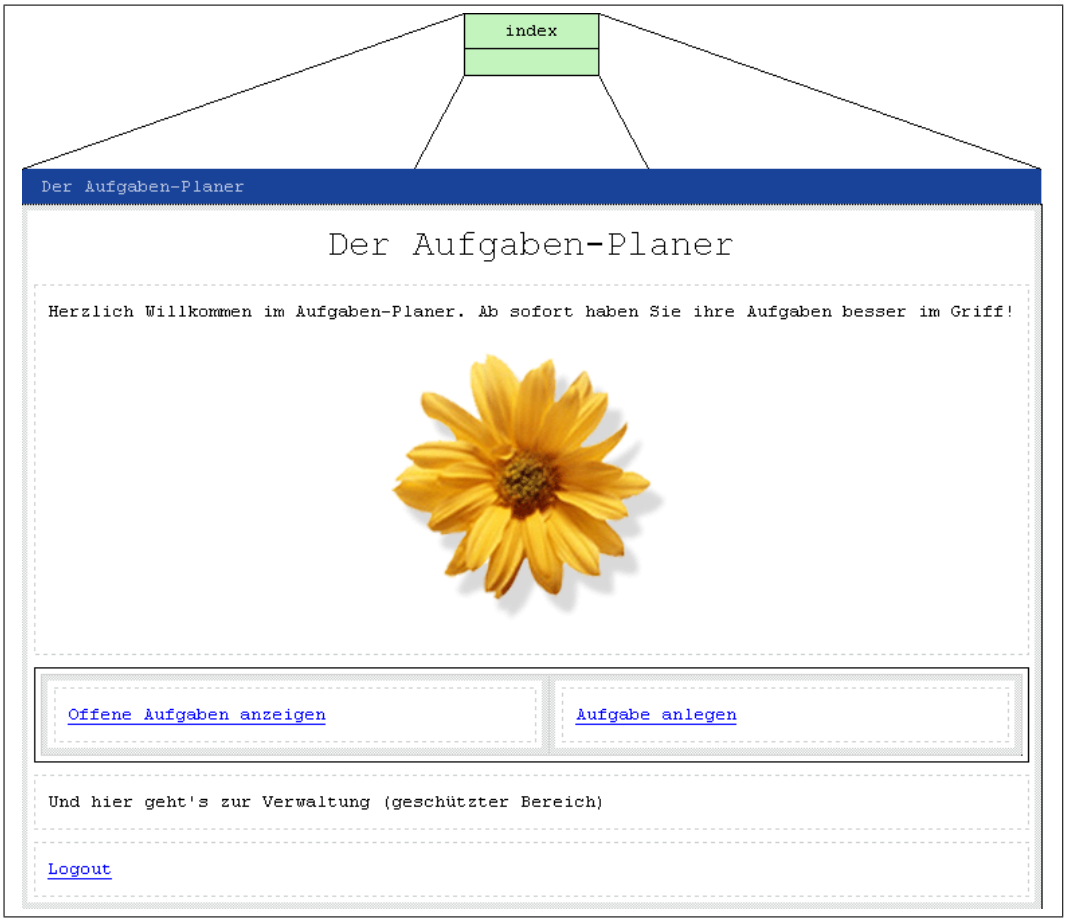

Abbildung 5.13: Die Hauptseite der Aufgabenplaner-Anwendung.

Dort ist der Text „Der Aufgaben-Planer“ zu sehen. Dieser Text wurde mit Hilfe der „Heading“-Komponente modelliert. In Abbildung 5.14 sind die editierbaren Attribute der Komponente aufgeführt.

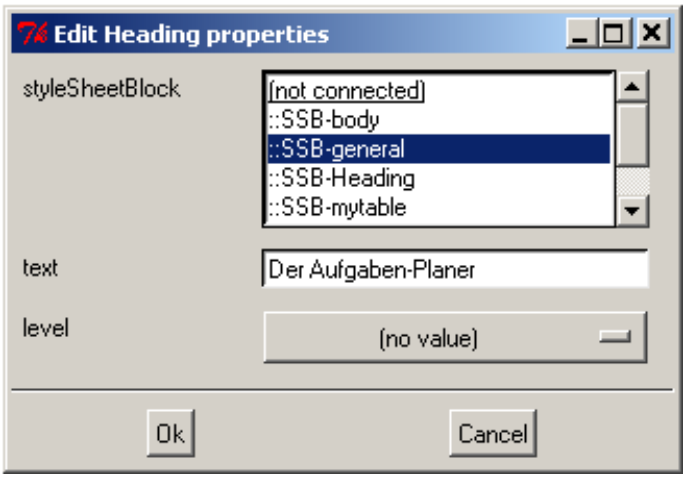

Abbildung 5.14: Attribut-Dialog für das Element Heading.

Das erste Attribut „stylesheetBlock“ kann auf einen zuvor selbst definierten StylesheetBlock verweisen. Für diese Überschrift ist das der Fall. Damit wird die Formatierungsvorlage, wie sie von HTML normalerweise für Überschriften verwendet wird, außer Kraft gesetzt und durch eine eigene ersetzt. Das nächste Attribut-Feld enthält den eigentlichen Text der Überschrift. Als letztes kann man noch den *level* der Überschrift festlegen. Damit legt man die Klasse fest, zu der diese Überschrift gehört. Das Attribut ist nur von Bedeutung, wenn man die vordefinierten Formatvorlagen für Überschriften von HTML verwenden will. HTML bietet sechs Klassen für Überschriften an, die sich jeweils in ihren Formatierungen, zum Beispiel der Textgröße, unterscheiden.

Für die Darstellung von Fließ-Text auf einer Web-Seite, gibt es die „Text“-Komponente. Die „Text“-Komponente muss aus Gründen der besseren Strukturierung immer in einem Paragraphen eingebettet werden. Zu diesem Zweck gibt es die „Paragraph“-Komponente, deren einziges, optionales, Attribut der Verweis auf einen benutzerdefinierten StylesheetBlock ist. Damit lassen sich Formatierungsangaben für diesen Paragraphen festlegen. Ein Paragraph wird in der visuellen Umgebung durch ein Rechteck mit einer grau gestrichelten Linie visualisiert. In Abbildung 5.13 enthält der erste Paragraph eine „Text“- und eine „Image“-Komponente. Für „Text“-Komponenten lassen sich folgende Attribute festlegen. Siehe dazu Abbildung 5.15.

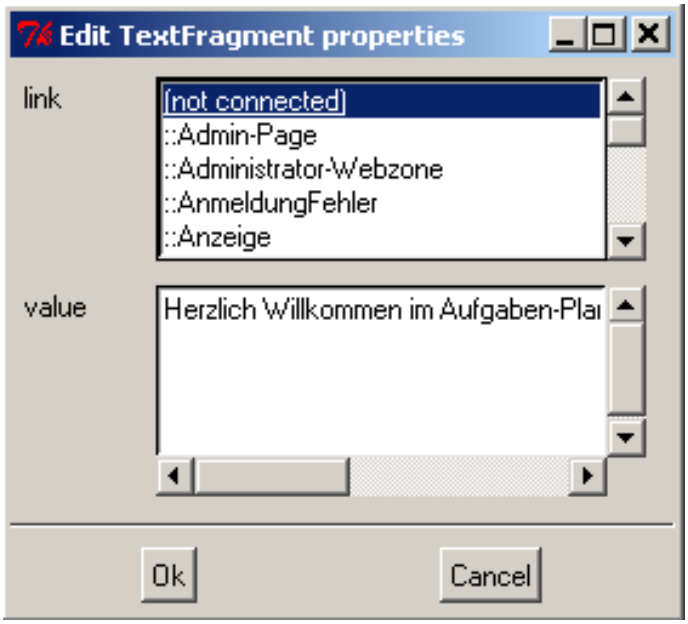

Abbildung 5.15: Attribut-Dialog für das Element Text.

Als erstes kann ein Verweis auf eine andere Web-Seite oder Skript angegeben werden. Dadurch wird ein einfacher Text zu einem *Link*. In der visuellen Umgebung wird ein Link dadurch visualisiert, in dem der Text in blau und unterstrichen dargestellt wird. In dem letzten Attribut-Feld kann schließlich der eigentliche Text angegeben werden.

Neben Texten könnten auch Bilder in Paragraphen eingebunden werden. Auf der Hauptseite der Aufgabenplaner-Anwendung wurde das Bild einer Sonnenblume eingebunden (siehe Abbildung 5.13). Um auf einer Web-Seite Bilder verwenden zu können, muss zunächst die Spezifikation der Web-Anwendung gespeichert werden. Danach kann über Attribute der Web-Anwendung der Pfad des Wurzelverzeichnisses der Bilder (Bilder-Verzeichnis) gesetzt werden. Erst dann kann über den Dialog das Bild-Konstrukt ausgewählt werden. Für Bilder können folgende Attribute gesetzt werden. Der Dialog zum Setzen dieser ist in Abbildung 5.16 dargestellt.

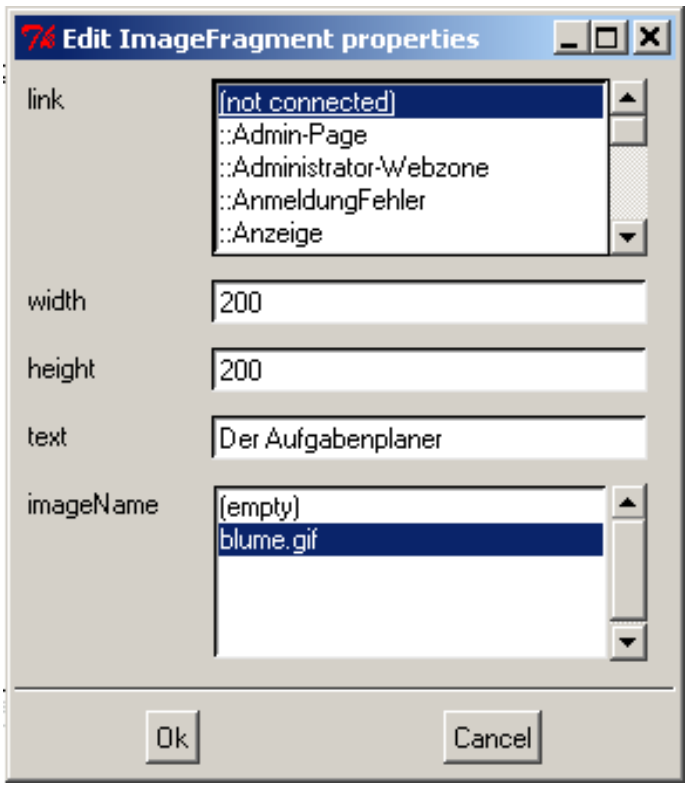

Abbildung 5.16: Attribut-Dialog für das Element Image.

Auch Bilder können als Link verwendet werden. Dazu kann man beim *link*-Attribut auf ein entsprechendes Ziel verweisen. Die Breite und Höhe des Bildes können in Pixel festgelegt werden. Entsprechen die Angaben nicht den tatsächlichen Maßen des Bildes, dann wird das Bild bei der Darstellung im Web-Browser auf die angegebene Größe skaliert. Das *text*-Attribut enthält einen Text, der alternativ zum Bild dargestellt wird, zum Beispiel wenn das Bild nicht geladen werden konnte. Mit dem letzten Attribut wählt man das Bild selbst aus. Dazu werden in der Auswahl-Liste die Namen alle Bilder angezeigt, die sich in dem zuvor angegebenen Bilder-Verzeichnis befinden.
Bei der Codegenerierung werden die in der Web-Anwendung verwendeten Bilder automatisch in das Zielverzeichnis kopiert und innerhalb der Web-Seiten korrekt referenziert. Dadurch wird ein manuelles kopieren der Dateien durch den Entwickler überflüssig.

5.5.2 Formulare

Ein wichtiges Element dynamischer Web-Anwendungen sind Formulare. Durch sie erhält der Benutzer der Anwendung die Möglichkeit Daten an die Anwendung zur weiteren Verarbeitung zu übergeben. Ein Beispiel hierfür ist die Eingabe von Adressen, die in einer Datenbank gespeichert werden sollen.

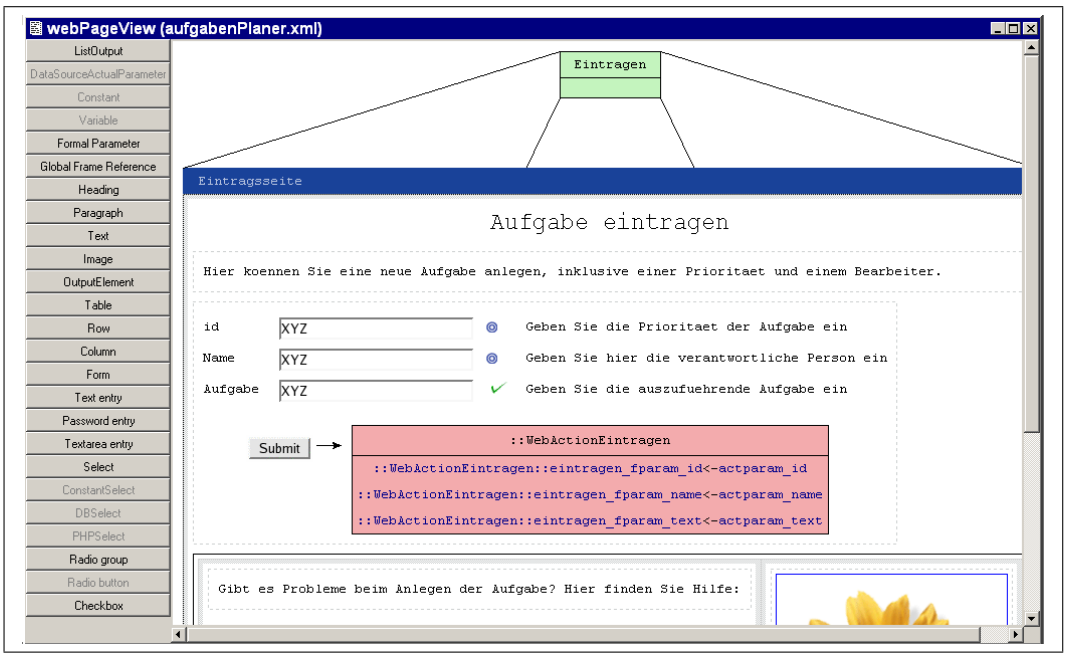

Abbildung 5.17: Spezifikation eines Formulars

In PaderWAVE werden Formulare als Komponenten in eine Web-Seite eingefügt. In ein Formular kann der Benutzer anschließend Formular-Elemente einfügen, die den Standard-HTML Formular-Elementen entsprechen. Diese sind verschiedene Texteingabe-Elemente, Check- und Radio-Buttons und Auswahllisten. Jedes dieser Elemente kann mit einer Beschriftung (Label) und einer kurzen Beschreibung (Description) versehen werden. Zusätzlich kann jedes Feld mit einem Wert vorbelegt werden. Weiterhin lassen sich jedem Formular-Element so genannte Konsistenzüberprüfungsfunktionen zuweisen. Diese dienen dazu, den im Formular-Element eingegebenen Wert auf bestimmte vorgegebene Eigenschaften hin zu überprüfen. So kann zum Beispiel überprüft werden, ob ein Benutzer in einem Textfeld keine Eingabe gemacht hat oder ob sich eine Zahl innerhalb eines bestimmten Bereichs befindet. Schlägt eine dieser Überprüfungen fehl, so wird der Benutzer dazu aufgefordert seine Eingaben zu korrigieren und die Daten erneut abzuschicken.

Wie die Daten, die ein Benutzer in einem Formular eingegeben hat, weiter verarbeitet werden, legt der Entwickler der Anwendung fest. Es besteht die Möglichkeit, die eingegebenen Daten an eine weitere Web-Seite weiterzuleiten, wo sie zum Beispiel einfach ausgegeben werden können. Weiterhin existiert jedoch die Möglichkeit die Daten an eine serverseitige Anwendung (Web-Action) zu übergeben. Die Übergabe von Parametern wird in Abschnitt 5.10 näher erläutert. Web-Actions werden in Abschnitt 5.9 erklärt.

5.5.3 Wiederverwendung von Komponenten

Wie man bisher gesehen hat, werden die einzelnen Web-Seiten einer Web-Anwendung unabhängig voneinander mit Inhalt gefüllt. Jede Web-Seite beschreibt eine eigene, von anderen Seiten gekapselte Sicht der Web-Anwendung. Diese einfache und klare Strukturierung besitzt den Nachteil, dass bereits definierte Komponenten nicht mehrfach auf verschiedenen Seiten eingebettet werden können. Man stelle sich die Anwendung vor, in der auf jeder Web-Seite im unteren Bereich der gleiche Text, zum Beispiel die rechtlichen Bestimmungen des Web-Inhalts, dargestellt werden sollen. Mit den oben beschriebenen Restriktionen müsste auf *jeder* Seite mindestens eine Text-Komponente definiert werden. Dies würde die Modellierungsarbeit unnötig vergrößern und die Wartbarkeit der Spezifikation verschlechtern. Um die Restriktionen aufzubrechen, wurde das Sprachelement „Global Frame“ eingeführt. Dieses Sprachelement kann wie Web-Seiten Komponenten aufnehmen. GlobalFrame's werden in Web-Seiten eingebettet. Die Einbettung erfolgt über das Sprachelement „Global Frame Reference“. Durch die Referenzierung kann ein „Global Frame“ von beliebigen Web-Seiten verwendet werden. Dadurch brauchen Komponenten, die mehrfach in der Web-Anwendung benötigt werden, nur noch einmal in einem GlobalFrame definiert werden. Erzeugt werden GlobalFrame's in der Hauptsicht. Dabei werden sie unten links angezeigt, wie der Abbildung 5.18 zu entnehmen ist.

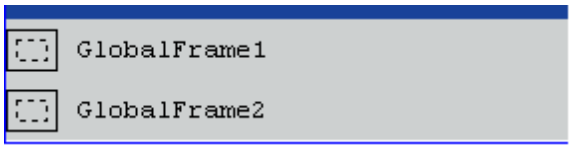

Abbildung 5.18: Darstellung von „Global Frame“-Elementen in der Hauptsicht

Die Sicht eines „Global Frame's“ entspricht der von Web-Seiten. Es gibt eine Auflistung der verfügbaren Komponenten die, unter Einbehaltung der Randbedingungen, beliebig dem GlobalFrame hinzugefügt werden können.

5.6 Navigationsleiste

Bei der Navigationsleiste handelt es sich um eine Komponente, die dem Entwickler die Modellierung der Verweise zwischen Seiten erleichtern soll. Die Verweise werden auf jeder Seite in Form eines Menüs eingefügt. Dieses fungiert so als eine Art HTML-Frame, der ständig verfügbar ist.
Die Navigationsleiste muss nicht manuell eingebunden werden, sondern ist automatisch auf jeder Web-Seite vorhanden. Über die entsprechende Auswahl in der Hauptsicht gelangt der Entwickler in den Bereich, der die Eigenschaften der Navigationsleiste steuert (siehe Abbildung 5.19).

Abbildung 5.19: Auswahl der Sicht der Navigationsleiste

Das Design der Navigationsleiste kann über verschiedene Attribute manipuliert werden. So kann zum einen ein benutzerdefinierter Stylesheet-Block verwendet werden, der über ein Auswahlmenü referenziert werden kann. Durch diesen kann der Entwickler wie gewohnt das Design der Schrift anpassen, um die Navigationsleiste grafisch seinem Projekt anzupassen.
Zum zweiten bietet PaderWAVE die Möglichkeit, das Layout anhand der Richtungsattribute (Alignment), „top, bottom, left, right“, zu platzieren. Die Auswahl wirkt sich direkt auf die aktuelle Sicht aus. Werden die Richtungen „top“ oder „bottom“ gewählt, wird eine vertikale Liste untereinander liegender Einträge generiert. Bei der Platzierung an den Seitenrändern (links oder rechts) hingegen sind die Verweise nebeneinander zu finden.

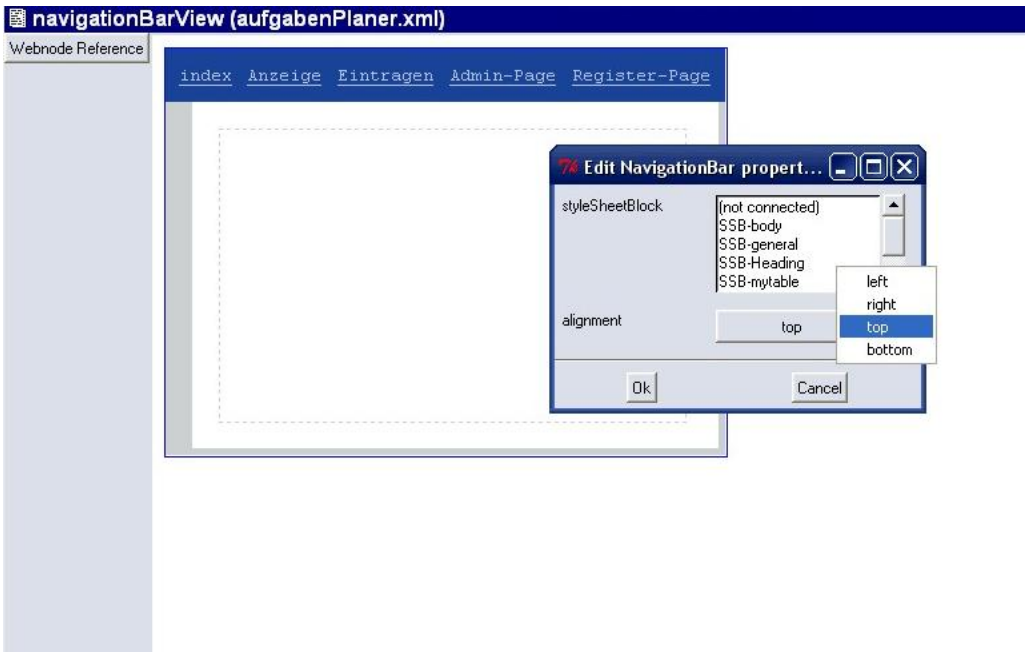

Abbildung 5.20: Festlegung der Design-Attribute

Um Web-Seiten in die Navigationsleiste einzufügen, kann eine Liste von Einträgen erzeugt werden, wobei für jeden Eintrag die spezielle Referenz ausgewählt werden muss. Die Wahl der Web-Seiten, die in der Navigation erscheinen sollen, liegt bei dem Benutzer und ist in keiner Weise von dem System vorgegeben. Um die Arbeit komfortabel zu gestalten, wird in der Hauptsicht die Wahl des Benutzers grafisch kenntlich gemacht (siehe Abbildung 5.21)

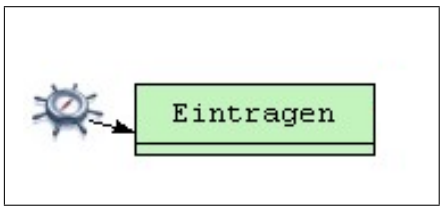

Abbildung 5.21: Symbol für die Erreichbarkeit einer Web-Seite aus der Navigationsleiste

5.7 Darstellungseigenschaften von Komponenten

In Abschnitt 3.5 wurde bereits Cascading-StyleSheets und dessen Vorteile erläutert. Das Konzept wurde in PaderWAVE übernommen. Zum einen die saubere Trennung von Inhalt und Darstellung, sowie die Definition von Darstellungseigenschaften, in Form von Stylesheet-Blöcken, an einem zentralen Punkt.

Die Komponenten in PaderWAVE besitzen keine eigenen Attribute, um Darstellungseigenschaften wie zum Beispiel die Schriftfarbe bei Text, festzulegen. Zu diesem Zweck besitzt jede Komponente ein Attribut mit dem Namen „stylesheetBlock“. Dieses kann eine Referenz auf einen Stylesheet-Block aufnehmen. Wenn eine Referenz gesetzt wird, dann werden die Formatierungsangaben aus dem Stylesheet-Block für die Darstellung der Komponente im Web-Browser verwendet. Vorausgesetzt, die Formatierungsangaben für diese Komponenten haben Gültigkeit. Ansonsten werden sie ignoriert. Durch die Verwendung von Stylesheet-Blöcken gibt es eine klare Trennung von Inhalt und Darstellung. Zusätzlich können Stylesheet-Blöcke von beliebig vielen Komponenten referenziert werden. Dadurch erzielt man ein konsistente Darstellung von gleichen Komponenten auf verschiedenen Web-Seiten und eine hohe Wiederverwendbarkeit.

Definiert werden alle Stylesheet-Blöcke an einer zentralen Stelle in einer eigenen Sicht. Über die entsprechende Auswahl in der Hauptsicht kann die „Stylesheet-View“ geöffnet werden. Siehe Abbildung 5.22.

Abbildung 5.22: Auswahl der Stylesheet-View

In der Stylesheet-View werden alle in der Web-Anwendung verfügbaren Stylesheet-Blöcke definiert. Dies erleichtert die Definition neuer Stylesheet-Blöcke und die Wartbarkeit bereits vorhandener. In Abbildung 5.23 ist die Stylesheet-View der Aufgabenplaner-Anwendung dargestellt.

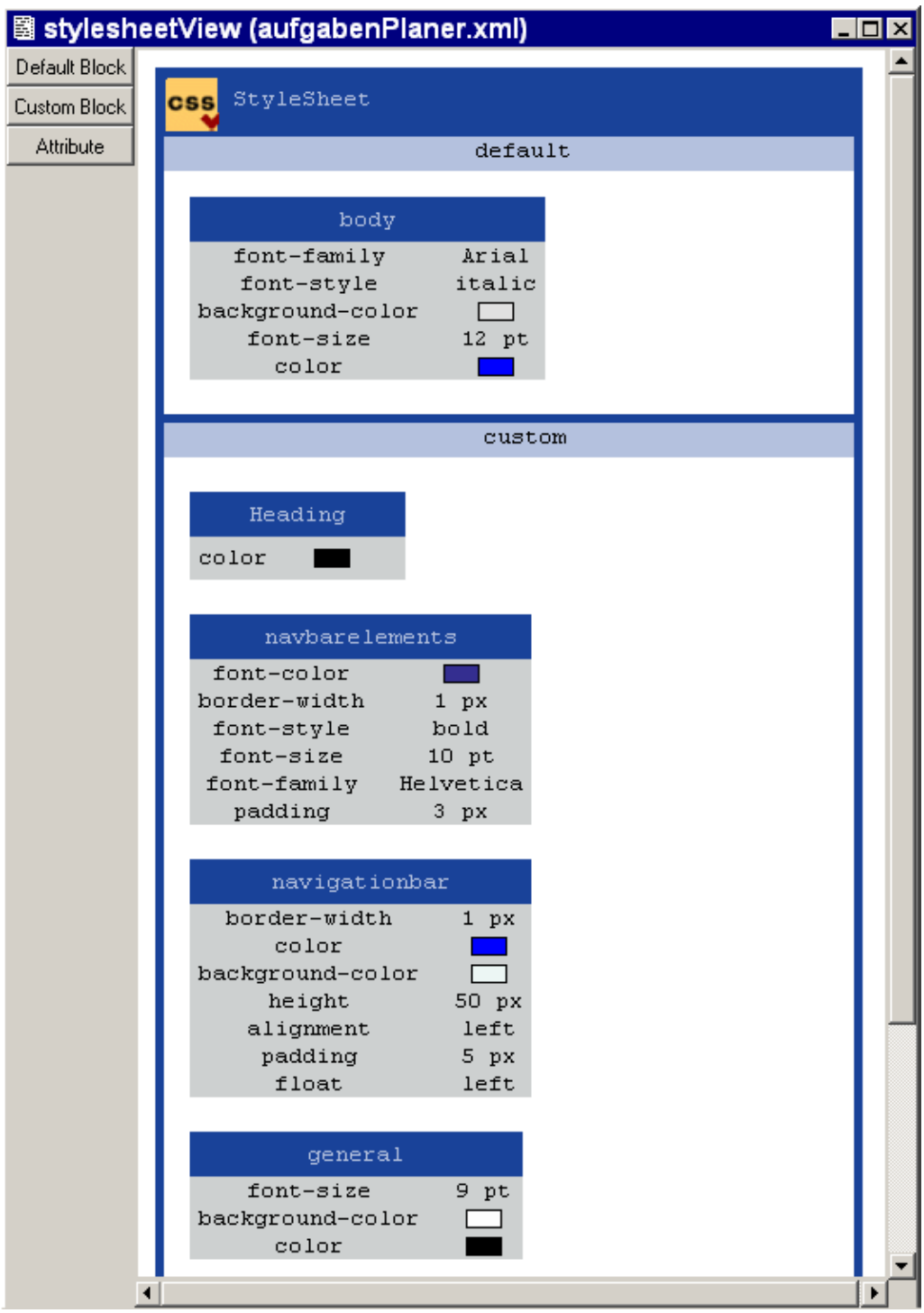

Abbildung 5.23: Stylesheet-View der Aufgabenplaner-Anwendung

Es gibt zwei Ausprägungen von Stylesheet-Blöcken. Es gibt Default-Blöcke und Custom-Blöcke. Default-Blöcke werden bei der Erzeugung mit entsprechenden Attribut-Werte-Paaren vorbelegt. Dazu kann das Attribut „name“ editiert werden. Beim Editieren des Attributs wird eine Liste von HTML-Elemente aufgelistet (Siehe

Abbildung 5.24).

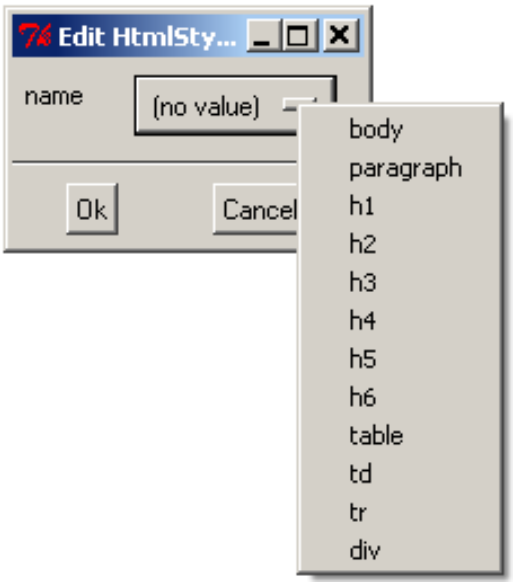

Abbildung 5.24: Dialog für Default-Stylesheet-Blöcke

Es kann ein Element aus der Liste gewählt werden. Daraufhin wird automatisch der StyleSheet-Block mit, für dieses Element gültigen, Attribut-Werte-Paaren gefüllt. Danach können die Werte der Attribute noch vom Entwickler manuell angepasst werden. Dieser Mechanismus soll die Modellierung von Stylesheet-Blöcken für Standard-Komponenten wie Absätze und Überschriften beschleunigen. Mit Hilfe der Custom-Blöcken kann der Entwickler Stylesheet-Blöcke von Grund auf selbst entwickeln.

5.8 Datenquellen

Dynamische Web-Seiten lassen sich nur unter Benutzung eines Datenspeichers realisieren. Der Datenspeicher liefert dazu die benötigten Inhalte und erlaubt auch die Manipulation dieser. In PaderWAVE wurde das Konzept der Datenquellen realisiert. Eine Datenquelle kann dabei eine Datenbank oder auch eine Textdatei sein. PaderWAVE unterscheidet dazwischen nicht, unterstützt zurzeit aber nur relationale Datenbanken wie beispielsweise MySQL.
Eine Datenquelle kann als Tupel von Spalten angesehen werden. Die Anzahl der zurückgegebenen Tupel hängt dabei vom gewählten Filter ab. Prinzipiell unterscheiden wir zwei Arten von Datenquellen und Änderungsoperationen darauf. Zum einen gibt es die Filter, die Tupel nach bestimmten Kriterien zurückliefern. Hierzu zählen DataSourceJoins, DataSourceProjectionFilter, LimitDataSources und die abstrakten DataSources. Zum anderen gibt es DataSourceModifyOperation, die Tupel der Datenbank verändern. Hierzu zählen die DataSourceInserts, DataSourceUpdates und DataSourceDeletes. Im Folgenden soll nun auf die Definition von Datenquellen und die Benutzung von Filtern und Modify Operationen eingegangen werden.

Um konkret Datenquellen zu definieren, muss erst in die bereits oben erwähnte DataBase-View gewechselt werden (siehe Abbildung 5.25). Mit einem rechten Mausklick können die globalen Einstellungen für die Datenbank angegeben werden. Diese Angaben werden später für den PHP Interpreter benötigt, um auf die Datenbank zuzugreifen.

Abbildung 5.25: Auswahl der Datenbanksicht

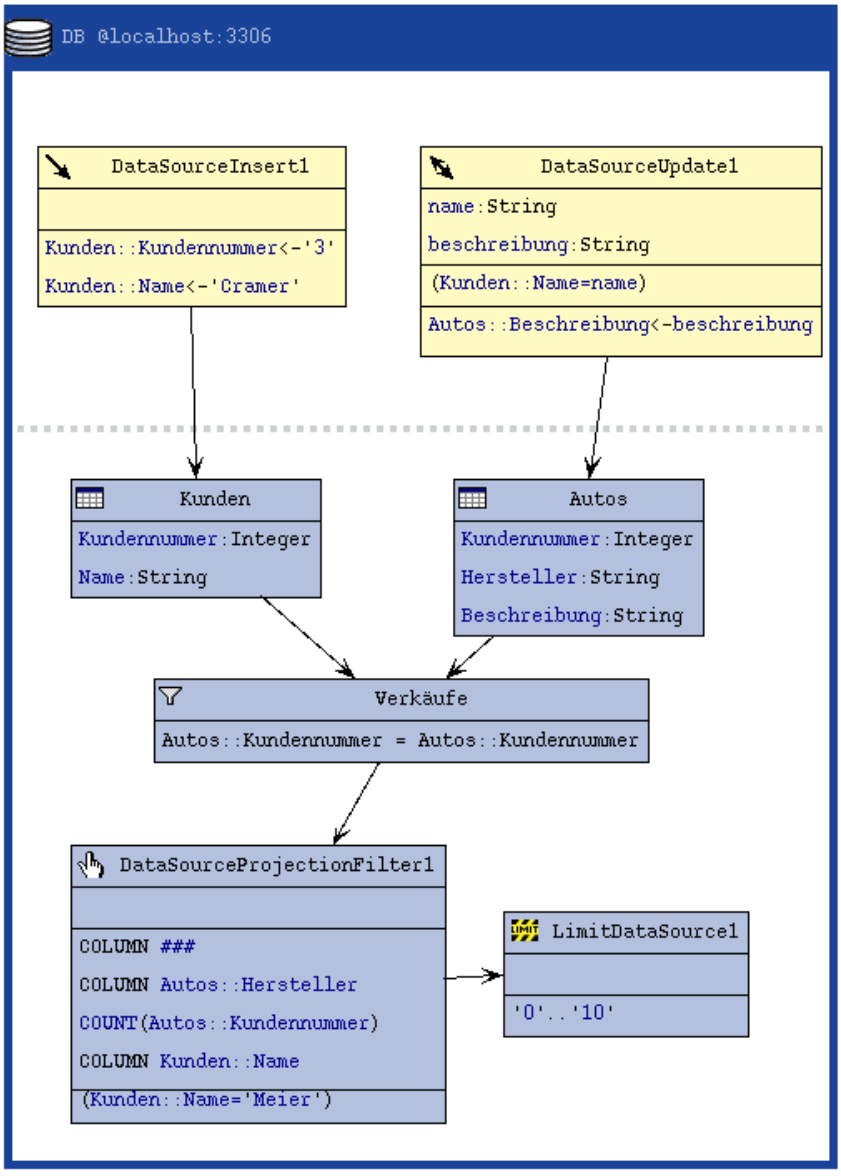

Abbildung 5.26: Datenbanksicht

5.8.1 Definition von Datenquellen

In der gerade definierten Datenbank können nun einzelne Datenquellen definiert werden. Diese Datenquellen können prinzipiell auch als Tabellen, wie sie in einer relationalen Datenbank existieren, angesehen werden. Sie bestehen aus beliebig vielen Spalten, denen jeweils ein in der Tabelle eindeutiger Name und ein Typ zugeordnet werden. Nach dem eine Spalte eingefügt wurde, können ein Name und ein Typ zugeordnet werden. Als Typen stehen die gängigsten Typen der relationalen Datenbank MySQL zur Verfügung: String, Integer, Float, Double.... Wenn die Datenquellen vollständig definiert sind, wird ein PHP Skript dbinit.php erzeugt, dass, wenn es auf einen Webserver geladen und ausgewählt wird, das visuell erstellte Datenbankschema automatisch erstellt.
Eine spezielle Art von Datenquellen stellen die Joins dar. Joins erlauben es, mehrere Tabellen miteinander zu verknüpfen. Das Ergebnis ist eine neue Datenquelle, also wieder eine Reihe von Tupeln über Spalten. Diese neue Datenquelle kann ebenfalls als Quelle für neue Filter dienen. Ein Join referenziert dabei zwei Datenquellen, so können auch zwei Joins wieder miteinander verbunden werden. Zusätzlich muss angegeben werden, über welche Spalten der Join gemacht werden muss. Dazu muss aus den beiden verwendeten Datenquellen jeweils eine Spalte gewählt werden. Diese beiden Spalten werden dann beim Join gleichgesetzt. Der in PaderWAVE verwendete Join entspricht dem SQL Natural Join. Wenn die Tabellen ausgewählt sind, wird in der Datenbanksicht dies durch Pfeile grafisch verdeutlicht. Sie entsprechen dem Datenfluss.
Um bestimmte Tupel aus der Datenbank auszuwählen gibt es in PaderWAVE die Möglichkeit DatasourceFilter zu spezifizieren. Hier gibt es zwei mögliche Filterarten: die ProjectionFilter und die LimitFilter. Ein ProjectionFilter wählt bestimmte Spalten aus und ein LimitFilter wählt spezielle Zeilen aus. Beide Filter benötigen eine DataSource als Quelle auf der sie operieren. Ein Limit Filter kann als Quelle aber auch einen ProjectionFilter haben. Dies ist die typische Anwendung, wenn zum Beispiel bestimmte Teile von Datensätzen angezeigt werden sollen wie beispielsweise beim Blättern durch einen Online Katalog. Beim ProjectionFilter können noch zusätzlich die Spalten hinzugefügt werden, die später in der Anwendung angezeigt werden sollen.
Um die Tupel, die ausgewählt werden sollen zu selektieren, wurde in PaderWAVE ein Expression basiertes Prinzip gewählt. Eine Expression setzt sich dabei rekursiv wieder aus Expressions oder bestimmten Endstücken zusammen. Um zum Beispiel eine Spalte nach einem bestimmten Wert zu filtern, wählt man zunächst eine Binary Expression aus und setzt in diese dann eine ColumnValueExpression und eine ConstantExpression ein. Durch dieses System ist es möglich, beliebige Ausdrücke zu spezifizieren. Das Prinzip der Expressions wird auch bei den DataSourceModify Operations eingesetzt. Beim Limit Filter ist es durch die Expressions auch möglich bestimmte Tupel zu selektieren, also einen Offset anzugeben. Damit Expressions auch parametrisierbar sind, ist es möglich auch formale Parameter zu definieren und diese dann in den Expressions zu benutzen.

5.8.2 DataSourceModify Operations

Die zweite Variante der Datenbankoperationen sind die DataSourceModify Operations. Sie manipulieren den Inhalt der Datenbank. Hier sind die DataSourceDeletes zu nennen, die bestimmte Tupel löschen, die DataSourceInserts, die Tupel einfügen und die DataSourceUpdates, die den Wert von Tupeln ändern. In Abbildung 5.26 sieht man in der oberen Bildhälfte einen DataSourceInsert und einen DataSourceUpdate. Beide beziehen sich auf zwei unterschiedliche Datenquellen, was durch die Pfeile verdeutlicht wird. Die DataSourceModifyOperations haben alle gemein, dass sie ebenfalls ein Ziel haben auf der sie operieren. Dies kann diesmal allerdings nur eine Tabelle sein, da ein ändern von Joins aus konzeptionellen Gründen nicht möglich ist. Das Auswählen von Tupeln ist, wie bei den DataSourceFiltern ebenfalls, durch das Expression System möglich. Bei den DataSourceInserts und DataSourceUpdates kommt ein neues Sprachelement hinzu: die DataSourceValues. Dies sind Werte, die an die Datenbank übergeben und dann eingefügt werden.

5.8.3 Ausgabe von Datenbankinhalten (ListOutput-Element)

Damit Datenbankinhalte in der Web-Seiten-Sicht dargestellt werden können, gibt es das Sprachelement ListOutput Element.

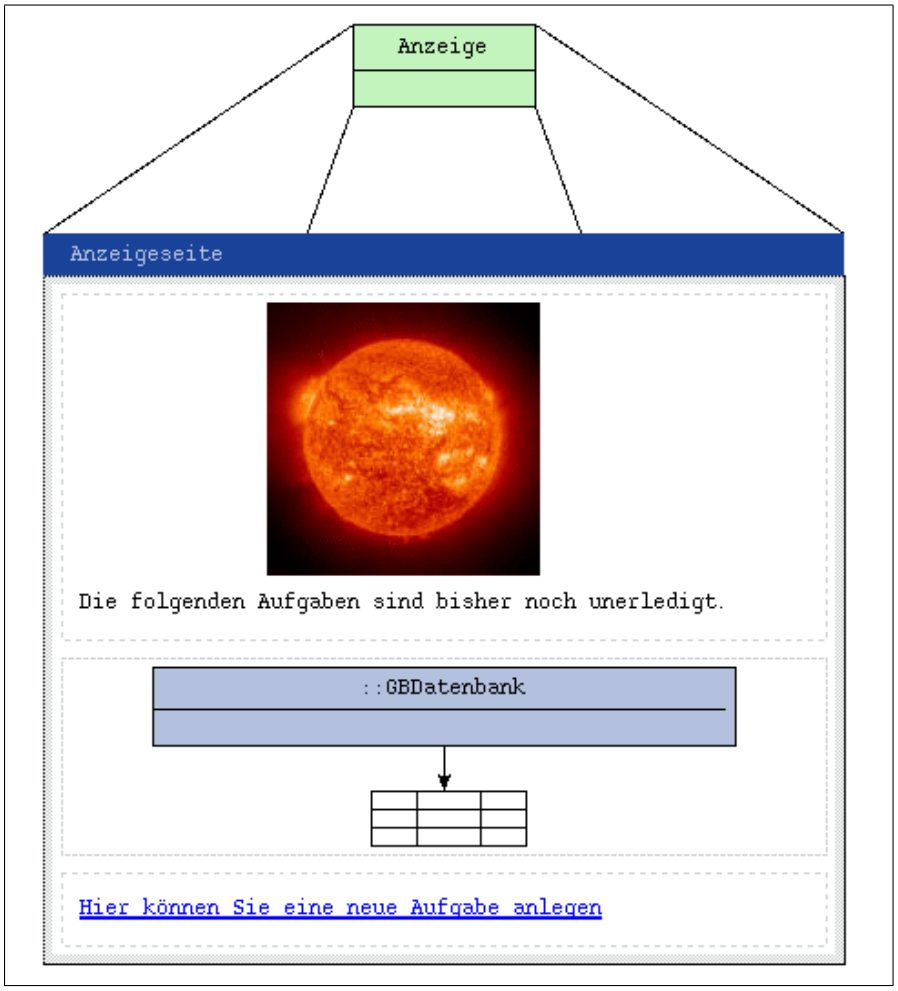

Abbildung 5.27: Das Ausgabe Element

Es wird mit einer Datenquelle verbunden und stellt die zurück gelieferten Datenbanktupel als Liste dar. Wenn ein ListOutput Element mit einer LimitDataSource verbunden wird, dann werden zusätzlich noch Navigationselemente hinzugeneriert, die es erlauben, nur eine bestimmte Menge an Tupeln darzustellen. Hier gibt es zwei Möglichkeiten. Zum einen können Textlinks mit den Beschriftungen Vor und Zurück generiert werden oder es werden Pfeile angezeigt. Beide Varianten sind fest in der mitgelieferten Bibliothek verankert und können über den Properties Dialog, wie in Abbildung 5.28 zu sehen, festgelegt werden. Außerdem kann dem Listoutput Element eine Zahl übergeben werden, die die maximal anzuzeigende Elementzahl angibt.

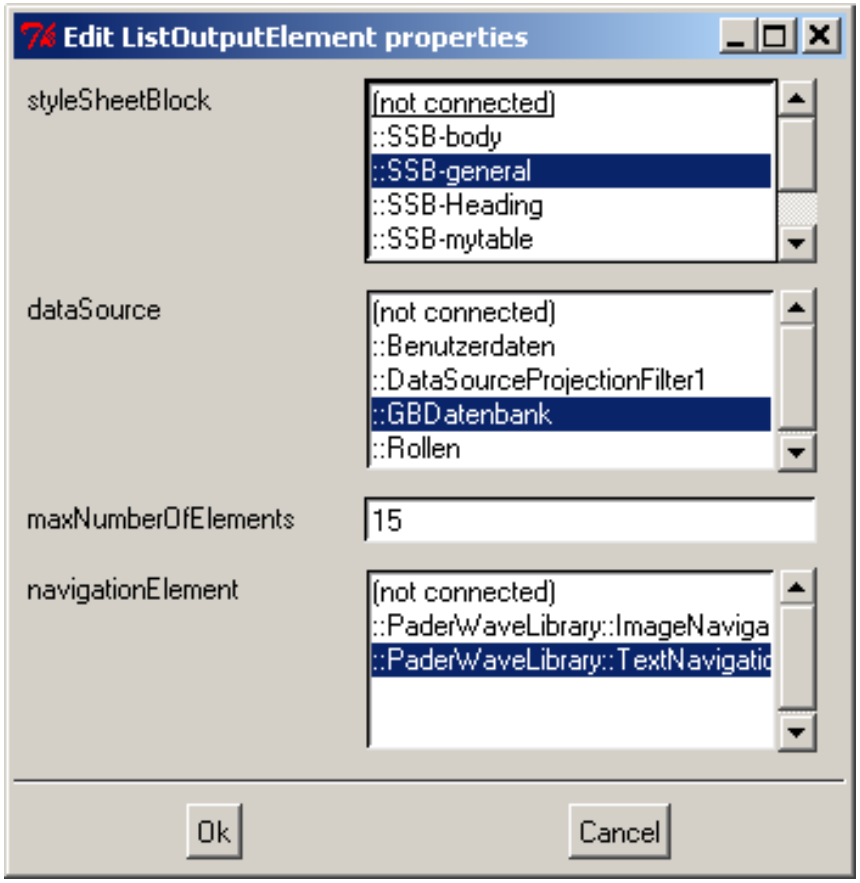

Abbildung 5.28: Der Eigenschaften Dialog des Ausgabe Elements

5.9 Benutzerdefinierte Skripte

Bei benutzerdefinierten Skripten handelt es sich um Programme, die zur Laufzeit der Web-Anwendung auf dem Server ausgeführt werden. Diese benutzerdefinierten Skripte werden in PaderWAVE durch Web-Actions repräsentiert.

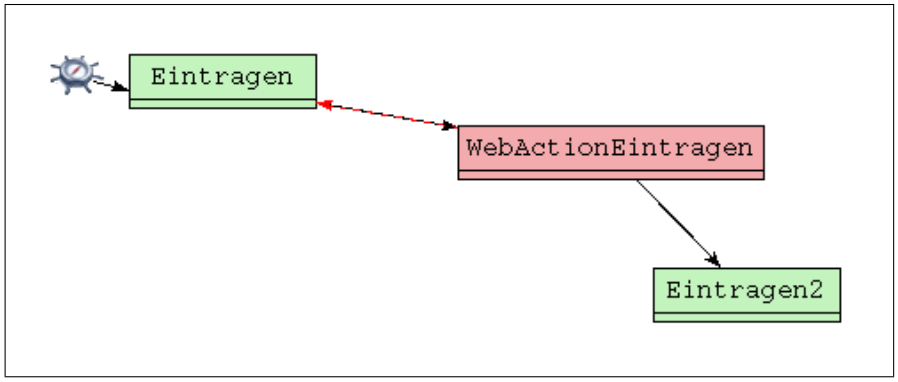

Abbildung 5.29: Web-Action mit Quell- und Ziel-Web-Seite

Web-Actions ermöglichen es dem Entwickler einer Web-Anwendung, eine Folge von Operationen zu spezifizieren, die sequentiell abgearbeitet werden. Zusätzlich kann der Entwickler, ähnlich wie bei textuellen Programmiersprachen, formale Parameter festlegen, mit deren Hilfe Daten an die Operationen der Web-Action übergeben werden können (siehe Abschnitt 5.10).
Es gibt unterschiedliche Typen von Operationen. Die im Rahmen einer Web-Anwendung am häufigsten verwendeten Operationen sind Datenbank-Operationen. Hier hat der Entwickler die Möglichkeit, Operationen wie Insert-, Update- und Delete-Operationen zu spezifizieren. Für jede dieser Operationen muss festgelegt werden, auf welcher Datenbank-Tabelle sie ausgeführt werden soll und welche Einschränkungen bei der Ausführung zu beachten sind. Die eigentliche Datenbank-Operation wird in der Datenbank-Sicht modelliert. In der Operation einer Web-Action wird lediglich auf die dort modellierte Datenbank-Operation verwiesen. Dies soll die Wiederverwendbarkeit der spezifizierten Datenbank-Operationen ermöglichen.
Ein weiterer Typ von Operationen ist die PHP-Operation. Hier hat der Entwickler die Möglichkeit, selbst geschriebene PHP-Skripte in seine Anwendung zu integrieren. Dies soll dazu dienen, die Entwicklung einer Web-Anwendung so flexibel wie möglich zu gestalten. Die formalen Parameter, die der Entwickler der Web-Action zuweisen kann, können auch in diesen Operationen verwendet werden.

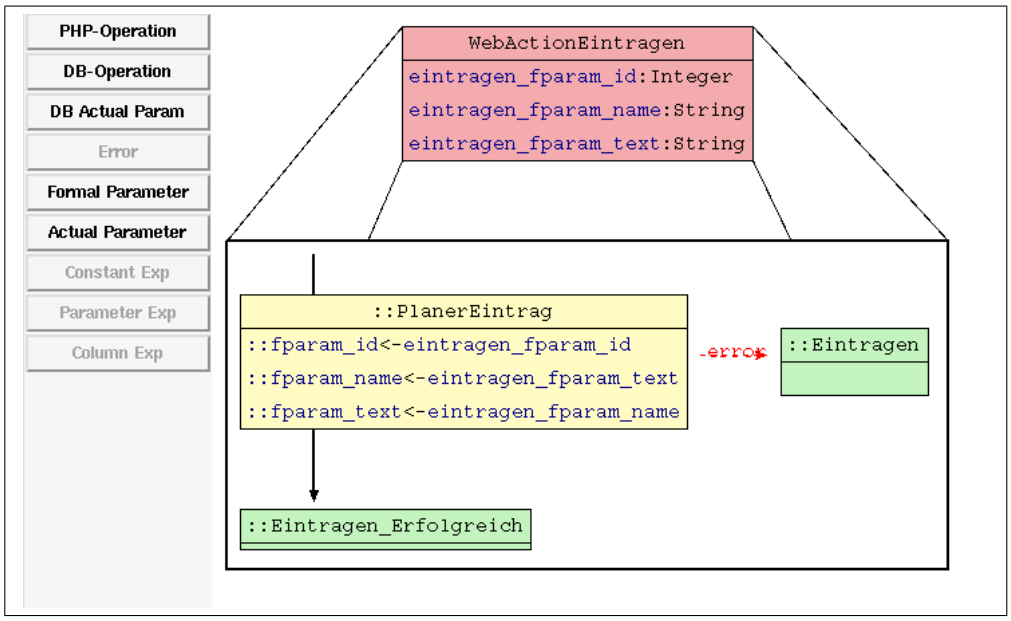

Abbildung 5.30: Spezifikation einer Web-Action

Wird die Folge von Operationen in einer Web-Action ohne Fehler ausgeführt, so wird der Benutzer der Anwendung auf eine Zielseite weitergeleitet, die vom Entwickler angegeben werden muss. Tritt während der Ausführung einer Operation ein Fehler auf, so kann der Entwickler jeder Operation eine spezielle Fehler-Seite

zuweisen, auf die der Benutzer im Fehlerfall umgeleitet wird. Hier können bestimmte Fehlermeldungen ausgegeben werden.

5.10 Parameterübergabe

Um Daten innerhalb einer Web-Anwendung zwischen verschiedenen Komponenten austauschen zu können, muss es einen Mechanismus geben, mit dessen Hilfe es möglich ist, Datenübergaben zwischen Komponenten der Web-Anwendung zu modellieren. Zu diesem Zweck wurde in PaderWAVE die Übergabe von Parametern eingeführt. Diese Art der Modellierung ist der Parameterübergabe bei Funktionsaufrufen in textuellen Programmiersprachen nachempfunden.
Bei der Parameterübergabe wird zwischen Datenquelle und Datenziel unterschieden. Bei der Datenquelle kann es sich zum Beispiel um ein Formular handeln. Datenziele können andere Web-Seiten, Web-Actions oder Projektions-Operationen auf einer Datenbank sein.
Das Datenziel entspricht einer Funktionsdefinition in einer textuellen Programmiersprache. Einem Datenziel können formale Parameter zugewiesen werden, die aus einem Namen und einem Datentyp bestehen. Innerhalb des Datenziels können diese formalen Parameter dann weiter verwendet werden. So kann zum Beispiel ein formaler Parameter, der einer Web-Action zugewiesen wurde, in eine benutzerdefinierten PHP-Skript innerhalb dieser Web-Action verwendet werden.

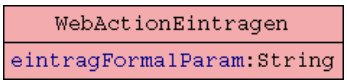

Abbildung 5.31: Spezifikation eines formalen Parameters in einer Web-Action

Die Datenquelle würde in diesem Modell dem Funktionsaufruf in einer textuellen Programmiersprache entsprechen. Hier werden aktuelle Parameter definiert, die einerseits mit einem formalen Parameter des Datenziels verknüpft sind und andererseits einen Ausdruck enthalten. Dieser Ausdruck könnte, sofern die Datenquelle ein Formular enthält, mit einem Eingabefeld dieses Formulars verknüpft sein. In diesem Anwendungsfall würde der vom Benutzer der Web-Anwendung in das Eingabefeld eingegebene Wert über die Verknüpfung innerhalb des aktuellen Parameters an den entsprechenden formalen Parameter weitergegeben werden. Der Ausdruck, welcher in einem aktuellen Parameter spezifiziert wird, kann auch andere Formen annehmen. So ist es unter anderem möglich, konstante Werte anzugeben.

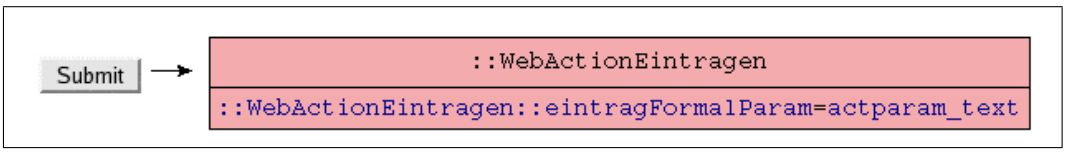

Abbildung 5.32: Spezifikation einer Verknüpfung zwischen einem Eingabefeld und einem formalen Parameter mittels aktuellen Parameters

Mit Hilfe des Konzeptes der Parameterübergabe ist es also möglich, den Datenfluss zwischen den Komponenten einer Web-Anwedung zu modellieren.

6 Weiterführendes Beispiel

6.1 Aufgabenstellung

Als weiterführendes Beispiel wird ein Autohaus dargestellt, welches seine Autos zum Verkauf anbietet. Die Web-Anwendung enthält eine Datenbank, die Automodelle, Automarken, PS-Werte, Hubraum-Angaben und Preise beinhaltet. Weiterhin gibt es zwei Zonen: Eine Benutzerzone, auf die alle Besucher der Web-Seite Zugriff haben, und eine Administratorzone, die nur für Autohausmitarbeiter zugänglich ist.

Benutzerzone:
Die Benutzerzone enthält die Web-Seiten Home, Suche, Autoübersicht, Mitarbeiter, Kontakt, Finanzierung und Hilfe.

- Home: Die Startseite sollt die Besucher begrüßen.
- Suche: Auf der Suchseite soll eine Auswahlbox mit allen in Datenbank vorhandenen Automarken und der Button „Suchen“ dargestellt werden. Bei der Auswahl einer Automarke und nach dem Klicken auf Button „Suchen“ sollen alle vorhandenen Autos der ausgewählten Marke in einer Tabelle aufgelistet werden.
- Autoübersicht: Die Übersichtseite soll alle Autos aus der Datenbank in einer Tabelle anzeigen.
- Finanzierung: Die Finanzierungseite bietet den Benutzer die Möglichkeit, die Finanzierungsmöglichkeiten zu überprüfen.
- Mitarbeiter: Die Mitarbeiterseite soll ein Formular enthalten, wo die Mitarbeiter sich einloggen und Änderungen in der Datenbank vornehmen können.
- Kontakt: Auf der Kontaktseite sollen die Namen, sowie E-Mail-Adressen, vorhanden sein.
- Hilfe: Die Hilfeseite soll die Anweisungen zum Bedienen dieser Web-Anwendung enthalten.

Administratorzone:
Die Administratorzone enthält die Web-Seiten AutoübersichtAdminEdit, EditModus, NeuModus und LogoutScript.

- AutoübersichtAdminEdit: Die Übersichtseite soll alle Autos aus der Datenbank in einer Tabelle anzeigen. Hier können neue Autos eingefügt und vorhandene Autos editiert werden.

- EditModus: Hier können die Mitarbeiter vorhandene Autosätze editieren.

- NeuModus: Diese Seite ist für Eingabe von Daten für neuen Autos, d.h. Autos, die nicht in der Datenbank gespeichert sind.

- LogoutScript: Hier können sich die Mitarbeiter ausloggen.

Die untere Abbildung zeigt die fertige Spezifikation der Web-Anwendung „Autohaus“.

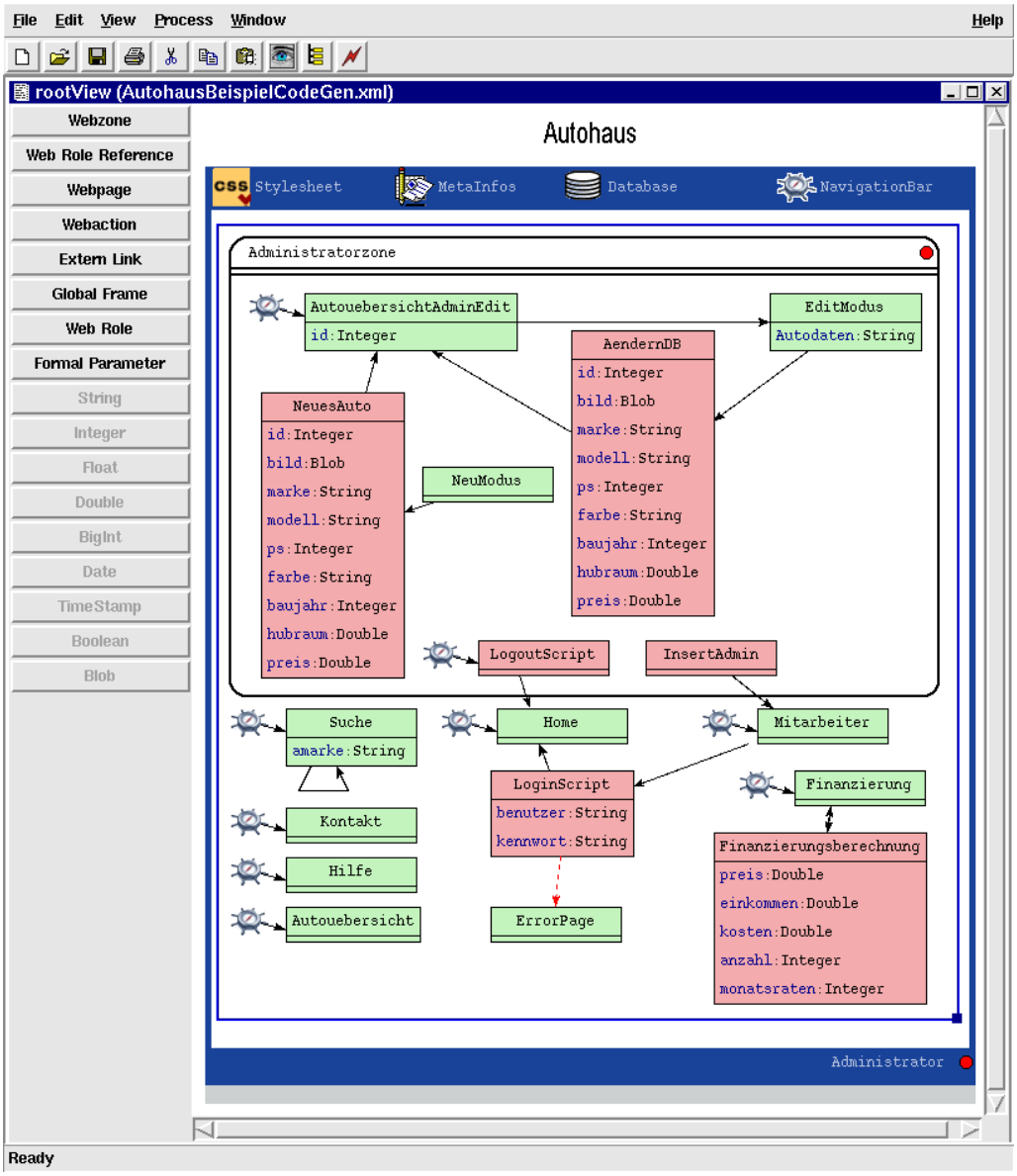

Abbildung 6.1: Spezifikation der Web-Anwendung „Autohaus“

6.2 Realisierung

Databaseview

Die Datenbanksicht besteht aus zwei Tabellen: „Autos“, wo die Daten für Autos gespeichert werden und „Benutzerdaten“, wo der Benutzername und das Kennwort abgelegt werden. Weiterhin beinhaltet die Datenbanksicht eine Insert-Operation „CarInsert1“. Diese Operation wird benötigt, um in das Formular zum Editieren vom Autos, auf der EditModus-Seite, eingegebenen Werte in Tabelle „Autos“ speichern zu können. Die Daten von editierten Autos werden mit Hilfe von Update-Operation „DataSourceUpdate1“ in der Datenbank-Tabelle „Autos“ überschrieben. Es werden

vier Filter angelegt: „AutomarkeFilter“, „SelectElementProjectionFilter“, „AutoSelectById“ und „SelectIDProjectionFilter1“. Die ersten beiden Filter werden bei der Suche gebraucht. Der „AutomarkeFilter“ sorgt dafür, dass aus der Datenbank-Tabelle „Autos“ die Daten aus den Feldern „bild“, „marke“ und „preis“ in einer Tabelle ausgegeben werden. Der „SelectElementProjectionFilterffiltert nach Automarken in der Datenbank-Tabelle „Autos“. Der „SelectIDProjectionFilter1“ selektiert alle Ids aus der Datenbank-Tabelle „Autos“ und zeigt diese in einer Auswahlliste an. Und mit „AutoSelectById“-Filter werden die Datensätze der ausgewählten Id ausgegeben.

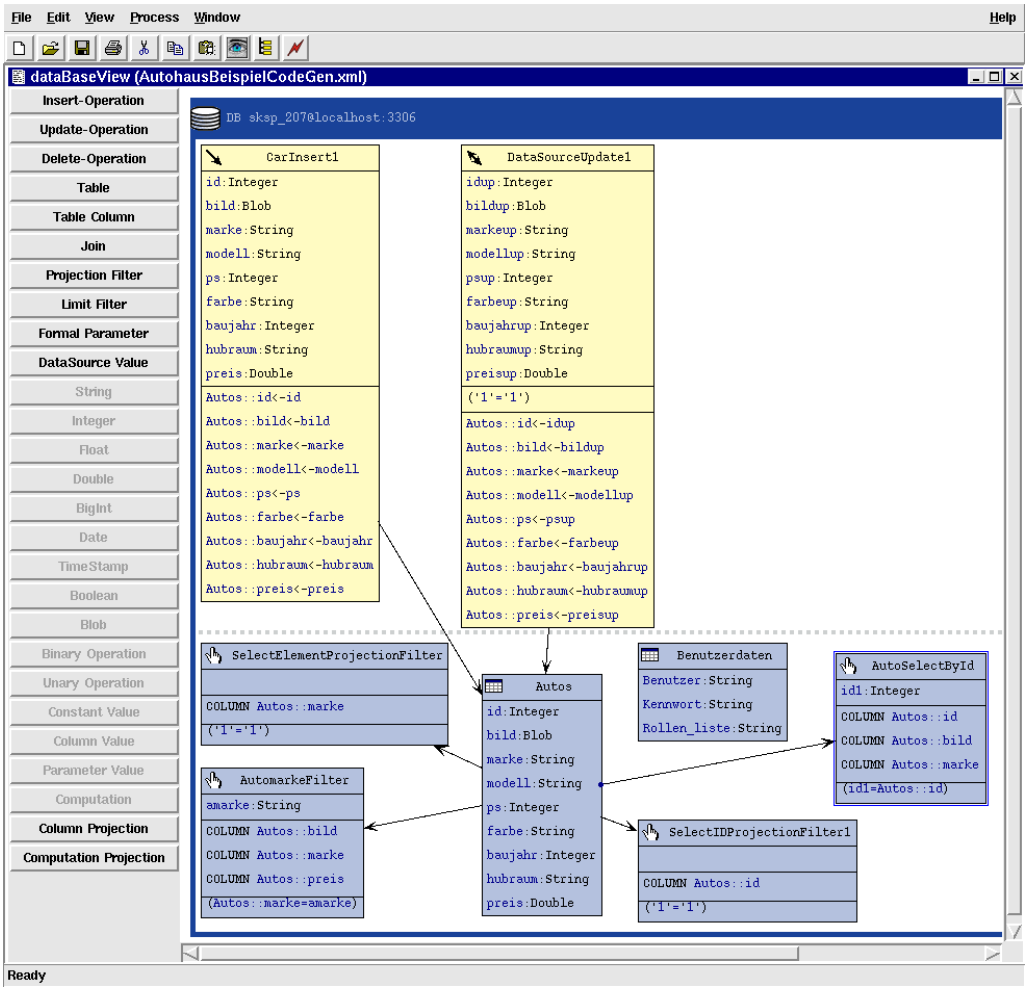

Abbildung 6.2: Datenbanksicht

Rootview

In der Hauptsicht werden alle Zonen und Web-Seiten angelegt, die für die Web-Anwendung benötigt werden. Es werden zwei Zonen angelegt: eine Benutzerzone, die für alle zugänglich ist, und eine Administratorzone für die Mitarbeiter. Da die Web-Anwendung schon eine Rootwebzone erhält wird nur eine Administratorzone innerhalb der Benutzerzone (Rootwebzone) eingefügt. Um auf die Administrator-

zone zugreifen zu können, muss der Benutzer die notwendigen Rechte besitzen. Die Darstellung von Rollen wird mit dem Sprachkonstrukt „Web Role“ realisiert. Dazu wird für den Administrator eine Farbe vergeben. Die Administratorzone selbst erhält eine Rollenreferenz mit dem Sprachkonstrukt „Web Role Reference“. Die Zonen enthalten unterschiedliche Web-Seiten. Es ist zu beachten, dass die Administratorzone automatisch Zugriff auf alle Seiten der Benutzerzone erhält, d.h. dass in die Administratorzone die Seiten eingefügt werden, die nur für Administratoren zugänglich sind.

Webpageview

Als nächstes werden die wichtigen Sprachkonstrukte der einzelnen Seiten beschrieben.

Home-Seite

Laut Aufgabenstellung enthält die Home-Seite nur ein Begrüßungstext. Der Firmenlogo „PS Autohaus“, das auf jeder Web-Seite unter Navigationsleiste angezeigt werden soll, wird mit dem Sprachkonstrukt „Heading“ dargestellt. Der Begrüßungstext muss mit Hilfe vom Sprachkonstrukt „Paragraph“ und entsprechendem Sprachkonstrukt „Text“ eingefügt.

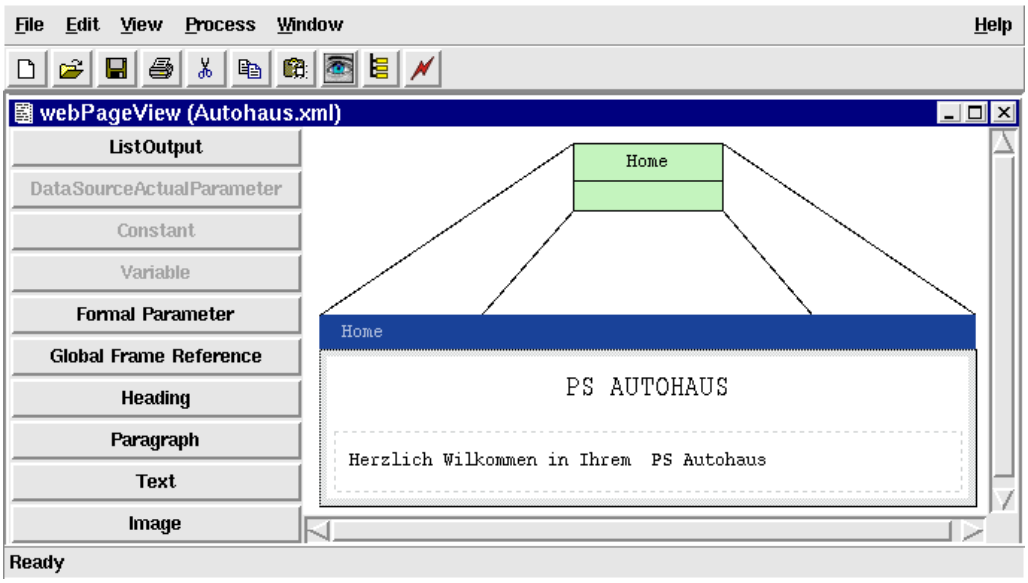

Abbildung 6.3: Home-Seite

Suche-Seite

Laut Aufgabenstellung enthält diese Seite eine Auswahlbox mit allen in der Datenbank vorhandenen Automarken und den Button „Suchen“. Bei der Auswahl einer Automarke und nach dem Klicken auf den Button „Suchen“ sollen alle vorhandenen Autos der ausgewählten Marke in einer Tabelle aufgelistet werden.

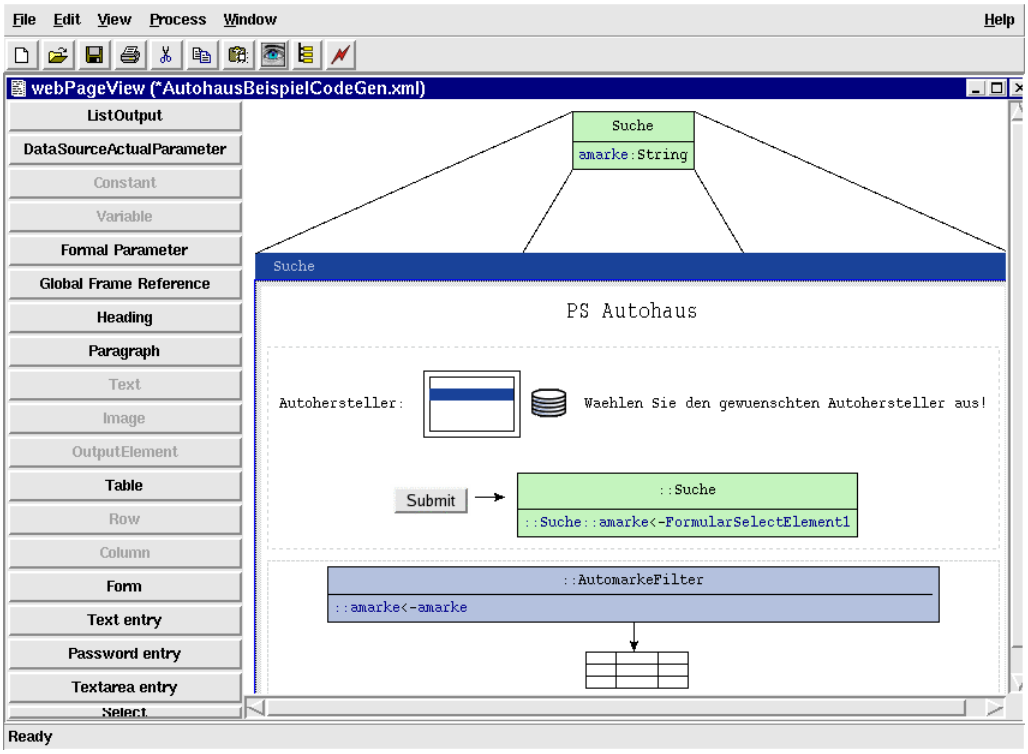

Abbildung 6.4: Suche-Seite

Diese Web-Seite enthält ein Formular, welches mit dem Sprachkonstrukt „Form“ realisiert wurde. Der Formular enthält ein „Select“-Feld. Ein solches Feld besteht aus einem „label“, „type“ und „description“. Vor dem Feld „description“ kann ein Zeichen platziert werden, anhand dessen bestimmt wird, woher die Daten für das Select-Element geholt werden. Im unteren Bereich des Formulars wird eine Seite ausgewählt, auf die nach dem Klicken auf den Submit-Button weitergeleitet wird. Der Wert, der in das Feld des Formulars eingegeben wird, wird an dieser Stelle als formaler Parameter übergeben. Nach dem Klicken auf den Submit-Button wird die Seite Suche angezeigt. Mit dem Sprachkonstrukt „ListOutput“ wird es ermöglicht, die Ergebnisse auf der selben Seite anzuzeigen. Für die Darstellung von Daten in dieser Tabelle wird der Filter „AutomarkeFilter“ benötigt.

Autoübersicht-Seite

Die Autoübersicht-Seite soll alle Autos aus der Datenbank in einer Tabelle anzeigen. Diese Darstellung wird mit dem Sprachkonstrukt „ListOutput“ ermöglicht. Alle Autodaten, die in der Datenbank „Autos“ gespeichert sind werden in einer Tabelle aufgelistet.

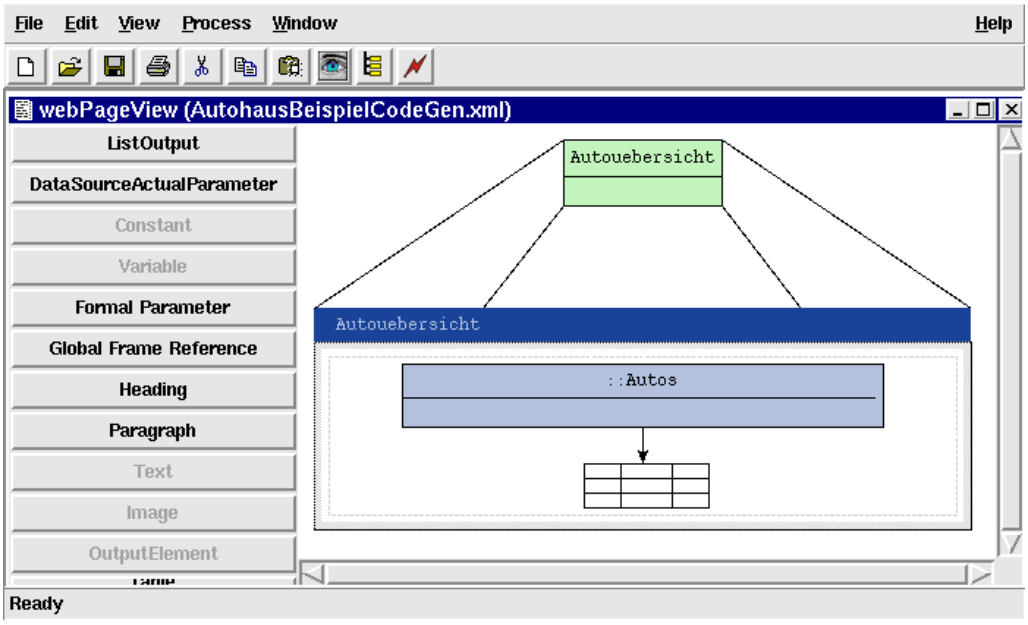

Abbildung 6.5: Autoübersicht-Seite

Mitarbeiter-Seite
Diese Seite enthält ein Login-Formular aus den Sprachkonstrukten „Text entry“ und „Password entry“, mit dem sich die Mitarbeiter in die Administratorzone einloggen können. Das bedeutet, dass das Benutzername und der Kennwort gespeichert sind und nach dem der Mitarbeiter die Daten eingegeben hat, werden diese mit den vorhandenen verglichen. Es wird eine Webaction benötigt, die ein PHP-Skript enthält, in dem die Kennwortüberprüfung stattfindet. Diese Webaction wird in der Hauptsicht mit dem Sprachkonstrukt „Webaction“ angelegt. Diese erhält einen Name „Kennwortüberprüfung“. In dem Formular wird nach den Klicken auf das Button „Submit“ die eingegebenen Daten nach „Kennwortüberprüfung“ geschickt. Dort wird das Skript ausgeführt. Nach dem erfolgreichen Ausführen werden die Mitarbeiter in die Administratorzone weitergeleitet. Bei den falschen Eingaben werden sie auf eine Errorpage verwiesen.

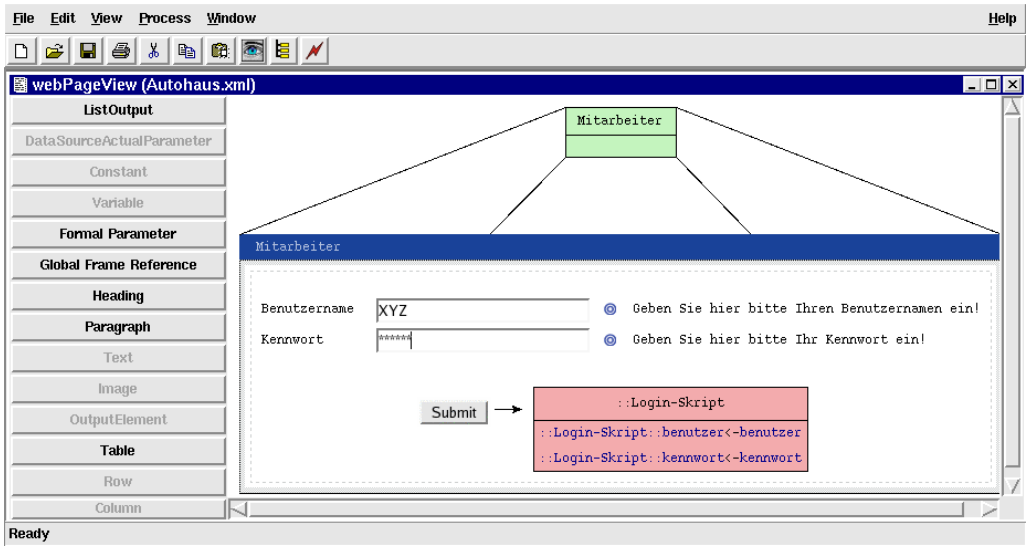

Abbildung 6.6: Mitarbeiter-Seite

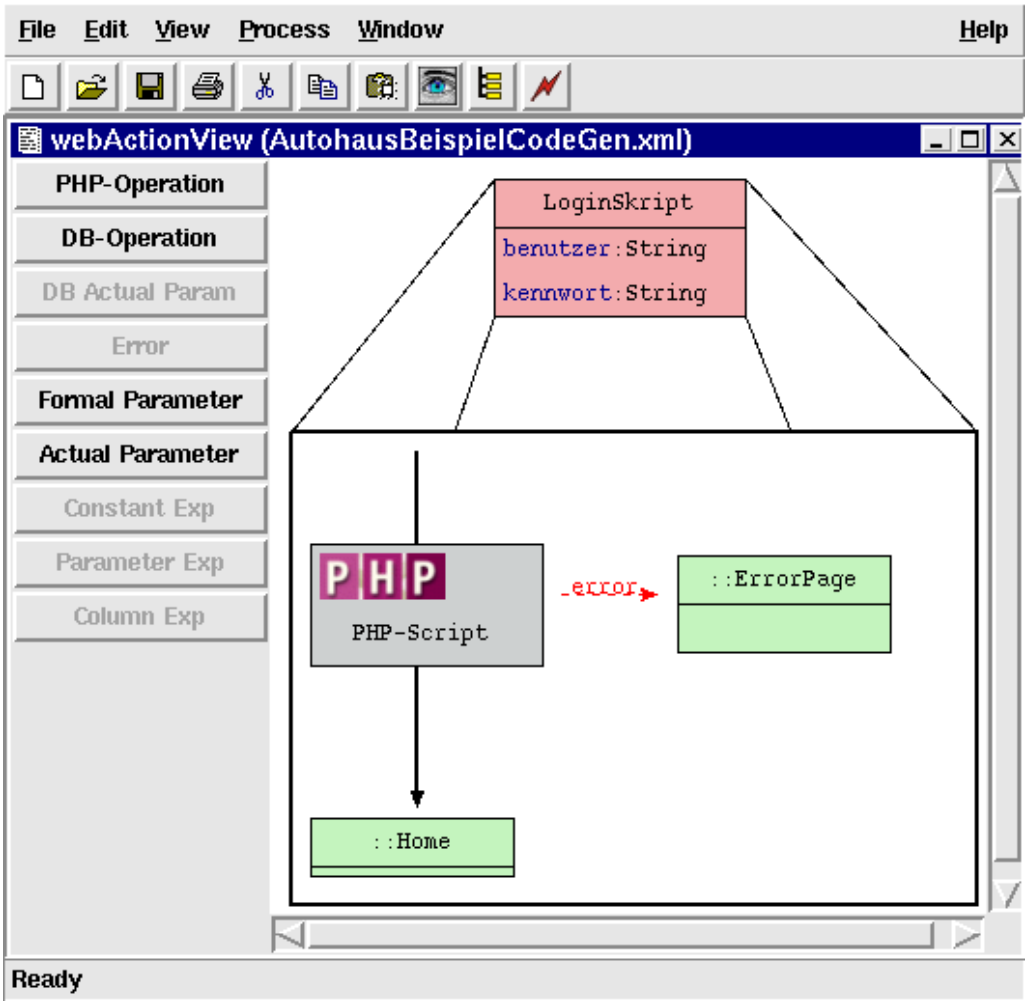

Abbildung 6.7: Login-Skript

EditModus-Seite
Diese Web-Seite enthält ein Formular, welches mit dem Sprachkonstrukt „Form“ realisiert wurde. Das Formular enthält mehrere „Text entry“-Felder. Ein solches Feld besteht aus einem „label“, „type“ und „description“. Vor dem Feld „description“ kann noch ein Zeichen für die Konsistenzüberprüfung eingefügt werden. Mit diese Konsistenzprüfung wird dafür gesorgt, dass die Felder z.B. nicht leer (Häckchensymbol) bleiben sollen. Im unteren Bereich des Formulars wird eine Webaction „AendernDB“ ausgewählt, auf die nach dem Klicken auf den Submit-Button weitergeleitet wird.

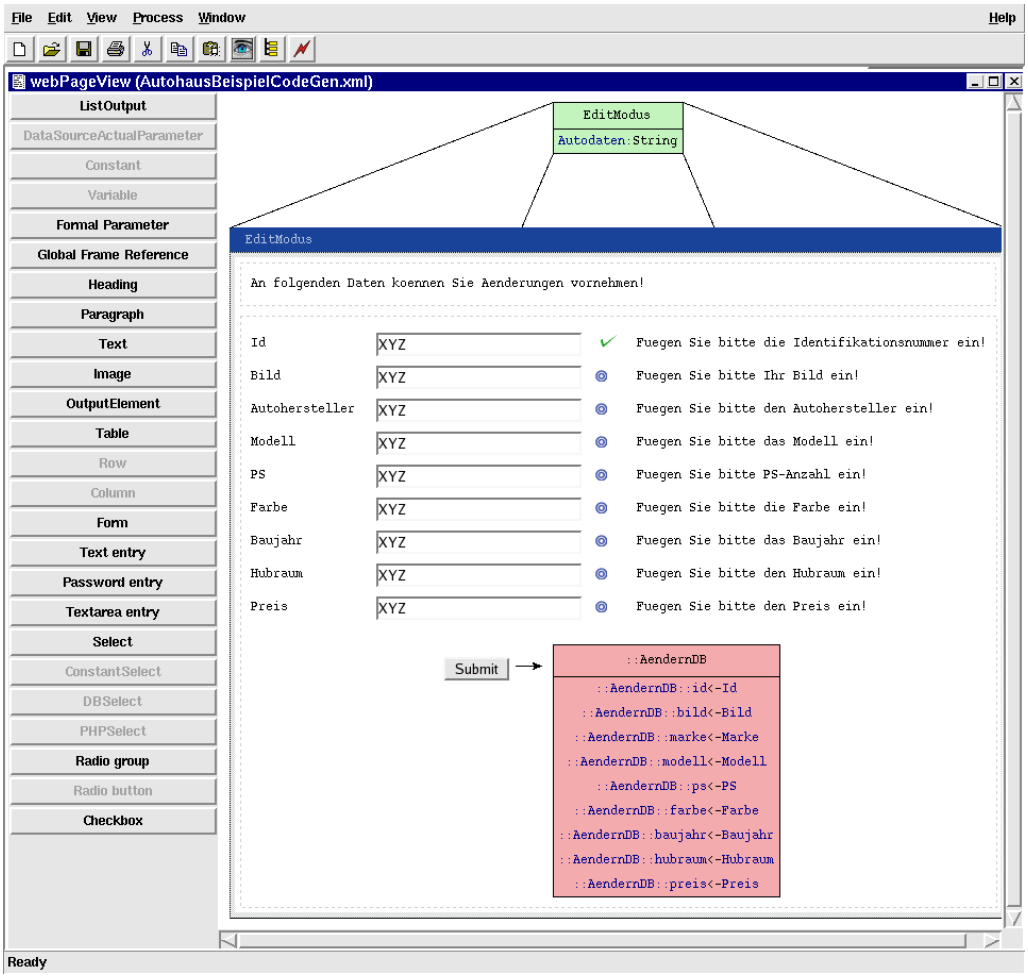

Abbildung 6.8: EditModus-Seite

An dieser Webaction werden alle geänderten Autodaten weitergegeben. Die vorhandene Daten werden in Datenbank-Tabelle „Autos“ mit Hilfe von Update-Operation „DataSourceUpdate1“ mit neuen Daten überschrieben.

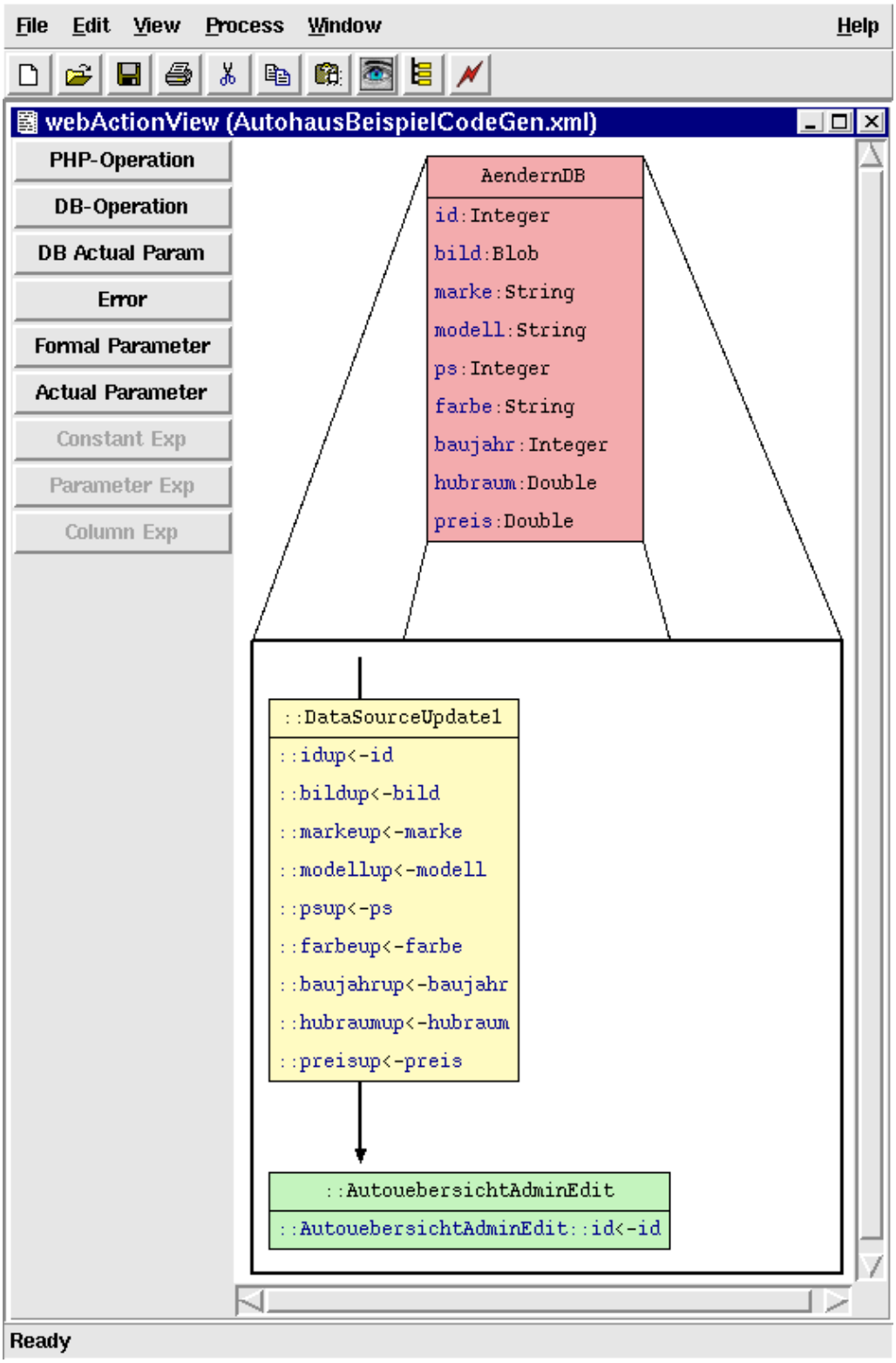

Abbildung 6.9: Webaction „AendernDB“

Navigationview

In der Navigationview wird die Navigationsleiste zusammengestellt. Die Ausrichtung der Navigationsleiste kann per Doppelklick auf die dunkelblaue Fläche im Dialogfenster eingestellt werden. Als weiteres enthält die Navigationsleiste die Web-Seiten. Die Web-Seiten werden mit dem Sprachkonstrukt „Webnode Reference“ hinzugefügt.

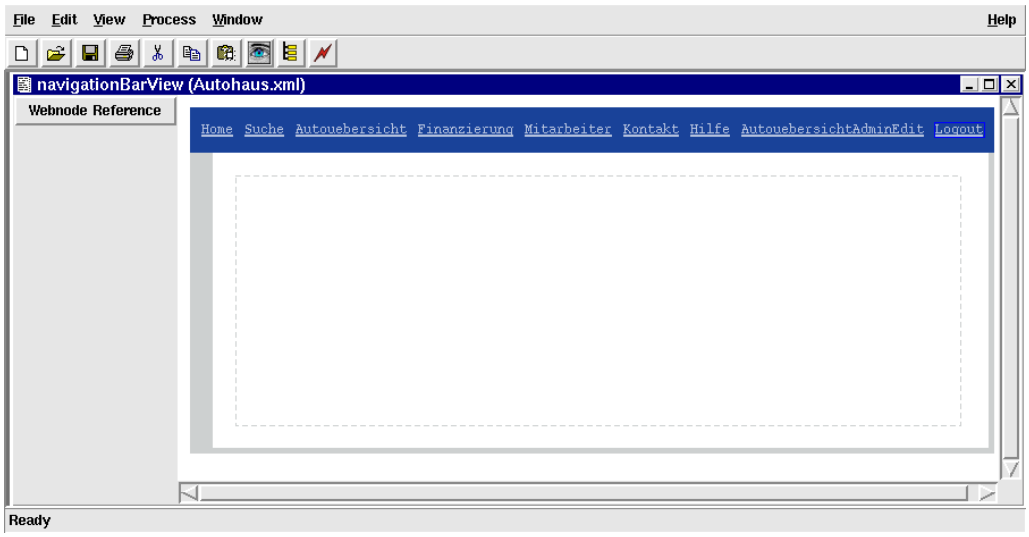

Abbildung 6.10: Navigationsleiste

Nachdem die Seiten für die Navigationsleiste referenziert wurden, werden diese in der Hauptsicht mit einem Navigationszeichen versehen (siehe Abbildung 6.1). Anhand diesen Zeichens kann erkannt werden, welche Seiten in der Navigationsleiste erscheinen werden. Wenn ein Benutzer als Besucher die Web-Anwendung besucht, dann werden alle Seiten außer AutoübersichtAdminEdit und Logout angezeigt, weil diese Seiten nur für den Administrator sichtbar sind. Der Administrator kann aber alle Seiten besuchen.

Stylesheetview

In der Stylesheetview werden die verschiedene Stylesheetblöcke für die Web-Anwendung „Autohaus“ definiert. Bei dem „Default Block“ existieren schon vorgefertigte Blöcke für unterschiedliche Teile einer Web-Anwendung, wie beispielsweise „body“. Dieser Block enthält alle Attribute, die für den Body dieser Anwendung notwendig sind. Bei dem „Custom Block“ werden der Name und die Attribute, darstellbar mit dem Sprachkonstrukt „Attribute“, per Hand eingegeben bzw. ausgewählt.

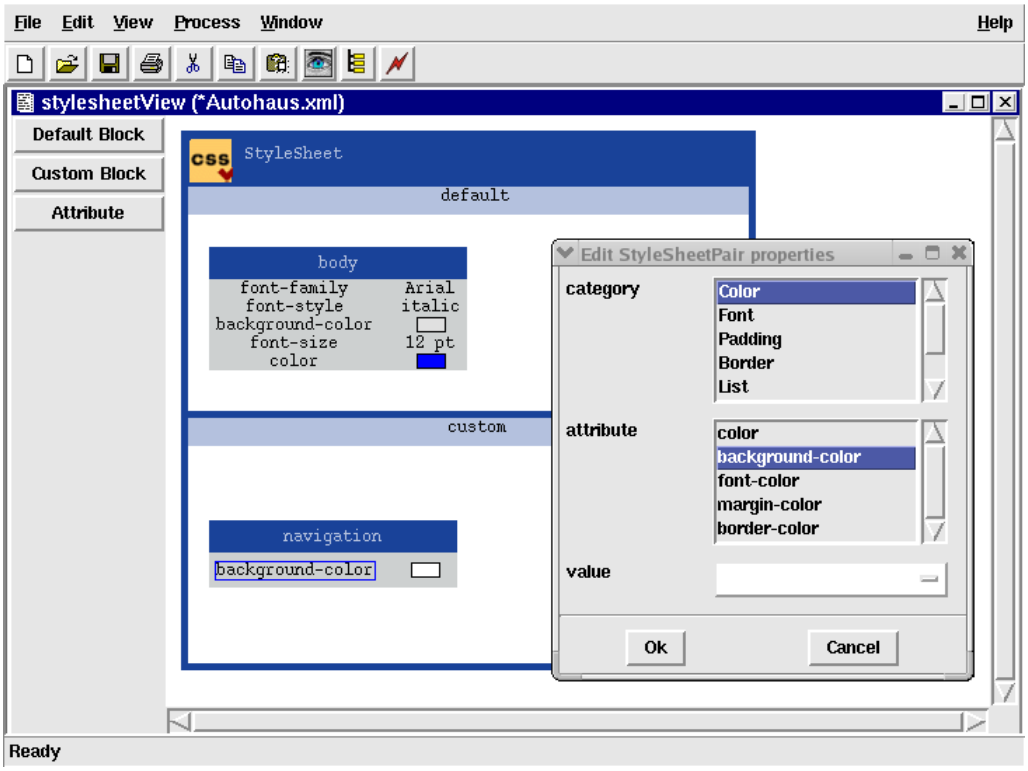

Abbildung 6.11: Stylesheetview

Die Stylesheetblöcke können den Komponenten dieser Web-Anwendung zugewiesen werden.

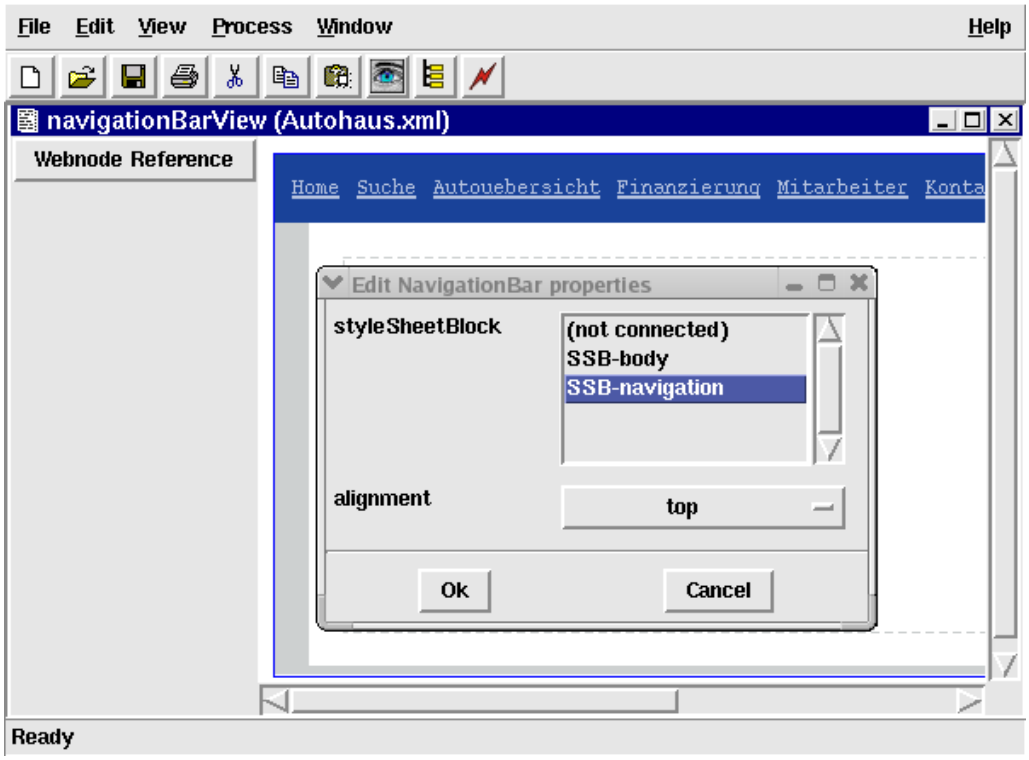

Abbildung 6.12: Referenz auf ein Stylesheetblock

7 Aspekte der Realisierung

In diesem Kapitel werden Details der technischen Realisierung erläutert. Als erstes werden in Abschnitt 7.1 Methoden und Techniken erläutert, die in dem generierten Anwendungs-Code zum Einsatz kommen. Im anschließenden Abschnitt 7.2 werden wichtige Aspekte der visuellen Sicht beschrieben.

7.1 Aspekte der Codegenerierung

7.1.1 Post- & Get-Methode

Innerhalb einer Web-Anwendung werden zur Übergabe von Daten vom Browser des Benutzers an den Server die im HTTP-Protokoll spezifizierten Methoden POST und GET verwendet.
Die GET-Methode ist dazu gedacht, Ressourcen von einem Server zu holen. Die Ressource wird dabei durch eine Ortsangabe und optionale Anfrage-Parameter spezifiziert. Eine GET-Anfrage ist nicht dazu gedacht, den Status der Web-Anwendung zu ändern, obwohl dies technisch möglich wäre.
Statusänderungen sollten immer mit Hilfe der POST-Methode herbeigeführt werden. Jedoch kann auf Grund der Verwendung der POST-Methode in Web-Anwendungen das so genannte „Double-Submit-Problem“ auftreten. Dieses Problem beschreibt die Möglichkeit, dass ein Benutzer, der Daten in ein Formular eingegeben hat und diese an den Server sendet (Submit), mehrere Möglichkeiten hat, die Daten erneut abzusenden. Dazu gehören:

- Benutzung des Aktualisieren-Knopfes des Browsers (explizites aktualisieren der Web-Seite, implizitites nochmaliges Absenden der Formulardaten)
- Benutzung der Zurück- und Vorwärts-Knöpfe des Browsers (implizites aktualisieren der Web-Seite, implizitites nochmaliges Absenden der Formulardaten)
- Zurückkehren zum Formular und erneutes Absenden der Formulardaten durch den Submit-Knopf des Formulars (explizites nochmaliges Absenden der Formulardaten)

Bei schlecht entworfenen Web-Anwendungen kann dieses Vorgehen zu Inkonsistenzen in der Datenhaltung führen.

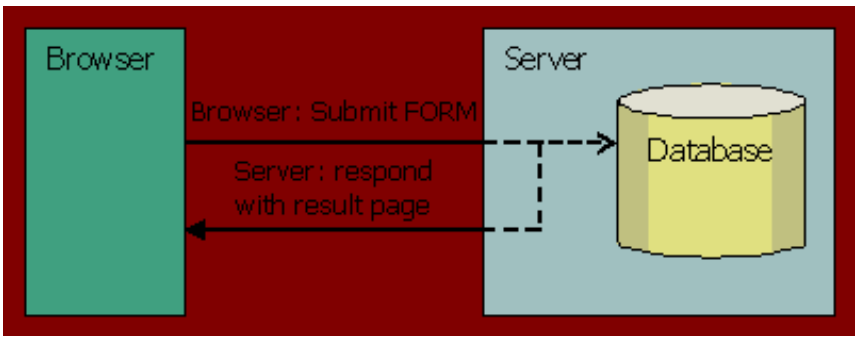

Abbildung 7.1: POST-Redirect-GET-Problem

Um dieses Problem zu umgehen, wurde in PaderWAVE das so genannte PRG-Muster (POST-Redirect-GET) [11] implementiert. Hierbei kommen die drei HTTP-Methoden POST, Redirect und GET zum Einsatz um ein Auftreten des „Double-Submit-Problem“ zu vermeiden. Der Benutzer sendet auch in diesem Fall Daten aus einem Formular mittels der POST-Methode an den Server. Dieser liefert jedoch nicht direkt eine Ergebnisseite aus, sondern führt einen Redirect auf die Ergebnisseite aus, die dann mittels der GET-Methode vom Server geholt wird.
Ergebnisseiten werden in einer mit PaderWAVE modellierten Anwendung immer durch ein HTTP-Redirect aufgerufen. Dabei spielt es keine Rolle, ob die Ergebnisseite direkt aus einem Formular aufgerufen wird, oder die Ergebnis- bzw. Fehlerseite einer Web-Action ist.

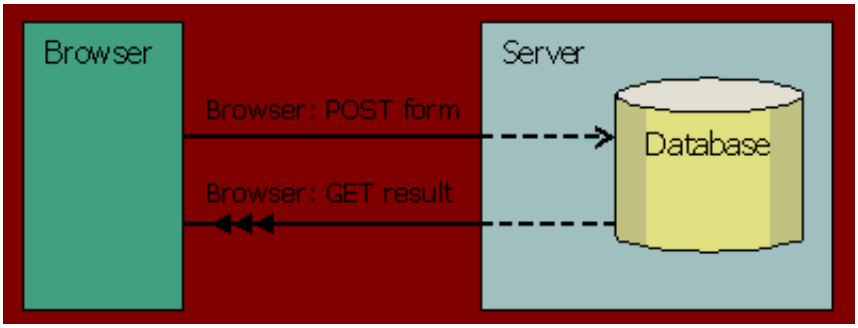

Abbildung 7.2: PRG-Muster

Durch diese Vorgehensweise wird das „Double-Submit-Problem“ umgangen. Der Benutzer hat also nicht mehr die Möglichkeit mittels einer der oben genannten Methoden Daten mehrfach an eine Web-Anwendung zu senden. Der Entwickler muss dazu keinen zusätzlichen Code in die Anwendung einfügen. Der Generator von PaderWAVE erstellt auf Basis der vom Entwickler spezifizierten Art der Datenübergabe den entsprechenden Code für das PRG-Muster.

7.1.2 PHP-Sessions

Um Daten während der gesamten Nutzung einer Web-Anwendung speichern und nutzen zu können, bietet PHP die Integration von Sessions an. Hierbei wird dem Benutzer der Web-Anwendung beim Aufruf einer Web-Seite eine eindeutige ID zugeordnet, die als Session-ID bezeichnet wird. Innerhalb der Session können definierte Variablen registriert, abgerufen und bei Bedarf gelöscht werden.

PaderWAVE nutzt Session beispielsweise bei der Verwaltung von Benutzerdaten und -rollen. Die Erstellung der Sessions und die Registrierung der Daten wird beim Login-Vorgang automatisch bei der Verwendung des Templates zur Authentifikation in Gang gesetzt, wobei hier das Ergebnis der Datenbankabfrage bezüglich des anfragenden Benutzers gespeichert wird. Über die gespeicherten Daten kann bei dem Aufruf jeder Web-Seite geprüft werden, ob der Benutzer den Login-Vorgang bereits erfüllt hat und ob seine Rechte ausreichend sind, um die angefragten Operationen ausführen zu können. Auf diese Art und Weise wird eine mehrmalige Nutzung der Datenbanktabelle vermieden und die Geschwindigkeit der Verifikation erhöht.
Beim Logout-Vorgang werden alle benutzerspezifischen Variablen automatisch gelöscht, so dass die unbefugte Weiternutzung der Anwendung ausgeschlossen ist.

7.1.3 DB-Init-Skripte

PaderWAVE generiert neben den eigentlichen Webseiten und Skripten sowie den dazugehörigen Bildern auch ein Skript um das Datenbankschema anzulegen. Die Datei dbinit.php wird bei Projekterzeugung in das Projektverzeichnis geschrieben. Wird diese Datei auf einen Web-Server geladen und ausgeführt, so wird automatisch das Datenbankschema erstellt. Dabei ist zu beachten, dass die Datenbank schon existieren muss. Eventuell bereits vorhandene, gleich benannte Tabellen werden überschrieben.

7.1.4 Synchronisation von Parametern

Die Synchronisation von formalen und aktuellen Parametern in PaderWAVE wird automatisch sichergestellt. Diese Synchronisation tritt dabei sowohl immer an den Schnittstellen zwischen Web-Seiten und Web-Actions also auch bei der Übergabe von Parametern an die Datenbank auf. Die Synchronisation soll dem Benutzer unnötige Aktionen ersparen und mehr Sicherheit gewährleisten. So wird immer sichergestellt, dass es zu einem aktuellen Parameter auch immer einen formalen Parameter gibt. Wird einer der Parameter gelöscht, so wird automatisch das ihm zugeordnete Pendant ebenfalls entfernt. Dieses Konzept ist intuitiv und vermeidet Fehler, die ansonsten mit zusätzlichen Konsistenzchecks mühsam hätten überprüft werden müssen.

7.1.5 Konsistenzprüfungen der Spezifikation

DEViL bietet von Haus aus Konsistenzprüfungen an. Dem Benutzer wird es somit erleichtert, konsistente Web-Anwendungen zu schreiben. Falls bei der Generierung etwas schief läuft, wird diese abgebrochen und eine Fehlermeldung angezeigt. Wird auf diese geklickt, so wird der Benutzer zu dem Kontext geleitet, in dem ein Fehler entstanden ist. Bereits auf der Spezifikationsebene ist es möglich, anzugeben welche Kardinalität bestimmte Elemente haben sollen. Wird eine Kardinalität verletzt, so wird die Erzeugung des Projekts nicht ermöglicht. Desweiteren wurde PaderWAVE um zusätzliche Konsistenzchecks erweitert. Insbesondere wurde dabei auf eine korrekte Modellierung der Datenquellen Wert gelegt. Hier wird zum Beispiel darauf geachtet, dass sich aktuelle Parameter auch auf die entsprechenden korrekten formalen Parameter beziehen und dass auch alle benötigten Attribute gesetzt sind.

7.1.6 Konsistenzprüfungen der Formulareingaben

In Web-Anwendungen kann der Fall auftreten, dass in einem Formularfeld nur bestimmte Werte eingegeben werden dürfen, oder dass diese Werte einer bestimmten Formatierung genügen müssen. Aus diesem Grund wurden in PaderWAVE Konsistenzprüfungen für Eingaben in Formularfelder eingeführt.
Damit hat der Entwickler von Web-Anwendung die Möglichkeit, jedem Eingabefeld in einem Formular eine Funktion zur Konsistenzüberprüfung zuzuweisen. Einige dieser Funktionen sind bereits in einer Bibliothek definiert. So existieren bereits Funktionen, mit deren Hilfe überprüft werden kann, ob der eingegebene Wert eine gültige E-Mail-Addresse ist, oder ob überhaupt ein Wert eingegeben wurde.
Die Überprüfung der Werte findet nach dem Absenden der Formulardaten statt. Der Generator fügt die spezifizierten Funktionen für die Konsistenzüberprüfung in die Web-Seite ein, in der sich auch das Formular befindet. Bei Abschicken der Formulardaten wird zunächst nicht das eigentliche Ziel aufgerufen, sondern erneut die Web-Seite mit dem Formular. In einem PHP-Skript wird dann zunächst festgestellt, ob überhaupt Daten übergeben wurden. Ist dies der Fall, so werden die Funktionen zur Überprüfung der Konsistenz aufgerufen. Entspricht ein eingegebener Wert nicht den vorgegebenen Spezifikationen, so liefert die Funktion ein Tupel, bestehend aus dem Namen des Eingabefeldes in dem der Fehler aufgetreten ist, und einer Fehlermeldung zurück. Diese Rückgabewerte werden gesammelt und zusammen mit dem Formular erneut ausgegeben. So kann der Benutzer seine Eingaben korrigieren und die Formulardaten erneut abschicken. Tritt während der Konsistenzüberprüfung kein Fehler auf, so wird das vom Entwickler spezifizierte Ziel aufgerufen. Wie der Aufruf dieses Ziel geschieht, hängt von der Art des Ziels ab. Ist das Ziel wieder eine Web-Seite, so wird diese direkt per HTTP-Redirect aufgerufen. Ist das Ziel jedoch eine Web-Action, so wird diese direkt in die Formular-Seite per PHP-Include-Anweisung eingebunden und ausgeführt.
Der Entwickler der Web-Anwendung kann auch eigene Funktionen für die Konsistenzüberprüfung definieren. Hierzu muss er in die Baumansicht der Web-Anwendung

wechseln und im Unterbaum 'Library' ein neues FormularCondition-Element einfügen. Es besteht nun die Möglichkeit zusätzliche formale Parameter für die Funktion zu definieren. Ein Beispiel wären hier Parameter für die untere und obere Grenze eines Zahlenbereichs in dem der zu überprüfende Wert liegen soll.
Weiterhin muss die eigentliche Überprüfung des Wertes in PHP programmiert werden. Hierbei kann es sich um eine einfache if-Anweisung handeln. Innerhalb dieser Anweisung muss der Parameter $fieldValue, der bereits definiert ist, ausgewertet werden. Für den Fall, dass $fieldValue nicht den gewünschten Wert enthält, muss die Variable $error mit einer textuellen Fehlermeldung erzeugt werden. Diese Variable wird bei der späteren Ausführung der Funktion im Fehlerfall als Rückgabewert übergeben.

7.2 Aspekte der visuellen Sicht

Während der Spezifikation von PaderWAVE wurden beim Entwurf von visuellen Sichten einige Besonderheiten beobachtet, die hier kurz vorgestellt werden.

7.2.1 Unterschiedliche Farbgebung

Einzelne Sprachelemente wurden in der visuellen Darstellung durch Farben visualisiert, um sie besser und schneller voneinander unterscheiden zu können. Z.B. haben die Web-Seiten (Webpages) die Farbe grün, und die Web-Actions sind rosa. Für die Datenbank-Änderungsoperationen wird gelbe, für die Datenquellen blaue Farbe verwendet. Der Einsatz von Farben findet sich ebenfalls bei den Rollen in der Hauptsicht wieder. Jeder Rolle wird nach Wahl des Benutzers eine Farbe zugewiesen. In der Web-Zone kann man Rollen referenzieren, und sie werden in entsprechender Farbe angezeigt.

7.2.2 Link-Visualisierung

Texte und Bilder können als Links dargestellt werden, indem man das Attribut *link* von diesen Sprachelementen auf eine Web-Seite, eine Web-Zone aus der Web-Anwendung oder einen externen Link setzt. Externe Links müssen dafür in der Hauptsicht vorhanden sein. Alle textuellen Links werden blau und unterstrichen visualisiert. Bei Bildern verändert sich das Aussehen nicht.

7.2.3 Stylesheet-Editor

Um dem Benutzer die Möglichkeit der Trennung zwischen dem Inhalt und der Formatierung beim Modellieren seiner Web-Anwendung zu geben, wurde eine Sicht zum Festlegen des Layouts, der so genannte Stylesheet-Editor, definiert. In dieser Sicht

können unterschiedliche Stylesheet-Blöcke angelegt und an vielen Stellen in der Web-Anwendung referenziert werden. Ein Stylesheet-Block enthält Attribut-Werte-Paare. In PaderWave hat der Modellierer die Wahl zwischen dem Default-Stylesheet-Block und dem Custom-Stylesheet-Block. Mit dem rechten Mausklick auf einen Stylesheet-Block gelangt man zu seinen Eigenschaften.
Beim Default-Stylesheet-Block werden mehrere Attribut-Werte-Paare gleichzeitig eingefügt, je nachdem welches Sprachelement man bei den Eigenschaften ausgewählt hat. Es kann ein Absatz, eine Tabelle oder eine Überschrift sein. Fehlt dem Modellierer bei dieser Default-Angabe ein Attribut-Werte-Paar, weil das von ihm zu formatierende Element anders aussehen soll, kann er es problemlos über die Schaltfläche 'Attribute' hinzufügen und die konkreten Werte dafür im Stylesheet-Dialog bestimmen. Der Stylesheet-Dialog wird geöffnet, sobald man das Attribut entweder doppelt angeklickt hat oder über die rechte Maustaste den Eintrag 'edit attributes' im Kontextmenü ausgewählt hat.
Beim Custom-Stylesheet-Block hat der Modellierer die volle Freiheit die von ihm benötigten Attribute und Werte selber zu setzen. Er kann sie eins nach dem anderen hinzufügen und legt deren Anzahl somit selber fest. Mittels der Schaltfläche 'Attribute' - analog zum Default-Stylesheet-Block - werden Attribut-Werte-Paare eingefügt. Über den rechten Mausklick auf das Attribut des Attribut-Werte-Paares bekommt man einen Stylesheet-Dialog angezeigt, in dem die Kategorie, das Attribut und der Wert bestimmt werden können. Möchte man z.B. die Hintergrundseite einer Webseite festlegen, so wählt man zuerst die Kategorie 'color', dann das Attribut 'background-color' und schließlich die gewünschte Farbe aus der Farbpalette. Im Stylesheet-Dialog werden die vorhandenen Attribute in Abhängigkeit von der vorher ausgewählten Kategorie dynamisch angezeigt. Genauso verhält es sich mit den Werten, die eventuell Maßeinheiten haben, bzgl. der Attribute. Die Liste der Attribute passt sich somit der vorher festgelegten Kategorie an, und die Liste der Werte passt sich dem vorher ausgewählten Attribut an. Per rechten Mausklick auf den Custom-Stylesheet-Block kann man diesem einen aussagekräftigen Namen vergeben.

7.2.4 Realisierung von HTML-Tabellen

Bei der Visualisierung der vorhandenen Sprachkonstrukte in PaderWAVE wurde oft Gebrauch von visuellen Mustern wie Menge, Liste, Formular etc. gemacht. Jedes visuelle Muster besteht aus Berechnungsrollen, die in den Attributberechnungen der Sprachelemente an einzelne Klassen vererbt werden. Möchte man z.B. eine Webzone spezifizieren, die einzelne frei angeordnete Webseiten enthält, so können eine Webzone als eine Menge und eine Webseite als ein Mengenelement angesehen werden. Somit würde die Klasse für die Webzone von der Berechnungsrolle 'VPSet' und die Klasse für die Webseite von der Berechnungsrolle 'VPSetElement' erben.
Das Sprachelement Tabelle wurde z.B. mit dem visuellen Matrix-Muster realisiert. Das Matrix-Muster besteht aus den folgenden Berechnungsrollen: VPMatrix, VPMatrixRowList, VPMatrixRow, VPMatrixCell, VPMatrixColumnRef, VPMatrix-

ColumnList und VPMatrixColumn.
Eine Tabelle an sich ist als eine Matrix anzusehen, deshalb erbt die Klasse dieses Sprachelements von der Berechnungsrolle 'VPMatrix'. Da eine Tabelle über eine Menge an Zeilen und Spalten verfügt, werden zwei Attribute rows und columns spezifiziert, die Container für Zeilen und Spalten darstellen. An diese Attribute werden die Berechnungsrollen 'VPMatrixRowList' und 'VPMatrixColumnList' vererbt. Für den Zeilen- bzw. Spaltencontainer werden des Weiteren einzelne Zeilen bzw. Spalten spezifiziert, indem an die Klassen einer Zeile bzw. einer Spalte die Berechnungsrollen 'VPMatrixRow' bzw. 'VPMatrixColumn' vererbt werden. Eine Tabellenzelle ist genau einer Zeile und einer Spalte zugeordnet und wird durch die Berechnungsrolle 'VPMatrixCell' repräsentiert. Da im Matrix-Muster die Berechnungsrolle 'VPMatrixCell' unterhalb von 'VPMatrixRow' und nicht unterhalb von 'VPMatrixColumn' zu finden ist, ist lediglich die Zuordnung zu einer Zeile und nicht zu einer Spalte gegeben. Um zu wissen, zu welcher Spalte die entsprechende Tabellenzelle gehört, gibt man mit der Berechnungsrolle 'VPMatrixColumnRef' die Zuordnung der Zelle zu einer Spalte an.

7.2.5 Drag & Drop zur Instanziierung

Zur Vereinfachung und wegen der Benutzerfreundlichkeit wurde die Drag-and-Drop Technik eingefügt. Z.B. kann diese Technik beim Anlegen von Rollen und zugehörigen Referenzen verwendet werden. Hat man einige unterschiedliche Rollen angelegt, so kann man die benötigten Referenzen in einer Web-Zone per Drag and Drop mit der Maus anlegen. Dazu klickt man einfach die Farbe der entsprechenden Rolle an und zieht sie an die dafür vorgesehene Stelle in einer Web-Zone.

8 Zusammenfassung und Erfahrung

8.1 Zusammenfassung

Aufgabe der Projektgruppe war es, eine visuelle Umgebung (Generator) zu entwickeln, in der man Web-Anwendungen mit Hilfe einer *visuellen* Sprache spezifiziert, um damit dann den ausführbaren Programm-Code generieren zu lassen.
Zu Beginn der ersten Projektphase wurden die Ziele des Systems formuliert. Ein wichtiges Ziel war es, dass Entwickler von Web-Anwendungen bei der Benutzung von PaderWAVE über kein spezielles Expertenwissen im Bereich der Web-Programmierung verfügen müssen. Der Benutzer soll sich ganz auf die Spezifizierung des eigentlichen Web-Inhalts konzentrieren können. Mit der Entwicklung einer visuellen Spezifikationssprache haben wir die Entwicklung von Web-Anwendungen auf eine höhere Abstraktionsebene geschoben. Damit ist es dem Benutzer möglich, Web-Anwendungen unabhängig von deren technischen Realisierung zu entwerfen, um im Anschluss den Programm-Code automatisch generieren zu lassen.

Bei dem Entwurf der visuellen Spezifikationssprache wurde darauf geachtet, dass spezielle Probleme der Web-Entwicklung, z.B. das Synchronisieren von externen Links, in der Spezifikation leicht zu ändern sind oder komplett von der visuellen Umgebung gekapselt werden.

Für die Spezifizierung von Web-Anwendungen gibt es in PaderWAVE zahlreiche Sprachkonstrukte. Die statische Struktur kann mit Hilfe von Web-Seiten und Web-Zonen spezifiziert werden. Mit deren Hilfe lassen sich auch unterschiedliche Zugriffsbereiche in der Web-Anwendung umsetzen. In diesem Zusammenhang wurde das Rollen-Konzept vorgestellt. Es gibt Konstrukte für die Spezifizierung von visuellen Web-Elementen (Komponenten), wie zum Beispiel Texte, Überschriften, Bilder und Formulare. Neben diesen gibt es Sprachkonstrukte, um den Austausch von Daten zwischen Web-Seiten und Skripten zu realisieren. Zum Beispiel können die Eingabedaten eines Formulars an ein Skript zur weiteren Verarbeitung weitergeleitet werden.

Darstellungseigenschaften von Komponenten lassen sich mit Stylesheets festlegen. Durch die Referenzierung dieser lassen sie sich von beliebig vielen Komponenten

referenzieren und erhöhen damit deren Wiederverwendbarkeit. Darüber hinaus hat dies auch den Vorteil, dass Änderungen in der Darstellung nur an einer zentralen Stelle, dem Stylesheet, vorgenommen werden müssen.

Um Daten persistent zu halten, gibt es diverse Sprachkonstrukte für Datenbankoperationen. Neben Standardoperationen wie das Einfügen und Löschen von Datensätzen gibt es Konstrukte um Daten zu filtern, zu vereinen oder zu schneiden.

Damit Benutzer Web-Anwendungen möglichst einfach und schnell spezifizieren können, war es wichtig, neben der visuellen Spezifikationssprache eine intuitiv bedienbare Benutzungsoberfläche für PaderWAVE zu entwickeln. Dies wurde unter anderem mit Hilfe der unterschiedlichen visuellen Sichten in PaderWAVE gelöst. Zusätzlich werden Sprachkonstrukte unterschiedlich farblich dargestellt. Abhängigkeiten zwischen den wichtigsten Konstrukten, zum Beispiel Web-Seiten, werden durch Pfeile symbolisiert.

Um das System möglichst flexibel und offen zu halten, war ein weiteres Ziel, die Möglichkeit der Einbindung von benutzerdefinierten PHP-Skripten. Dazu wurden entsprechende Schnittstellen geschaffen, um die benutzerdefinierten Skripte und den generierten Skript-Code miteinander zu verbinden. Zum Beispiel für den Austausch von Daten in Form von Variablen. Diese Möglichkeit der Funktionserweiterung von Web-Anwendung wird auch von PaderWAVE selbst genutzt. So zum Beispiel bei der Umsetzung des Web-Zonen-Konzeptes. Die dafür erforderlichen Skripte liegen in einem Template als Web-Actions vor und können vom Benutzer ihren Bedürfnissen entsprechend angepasst werden.

Die Verwendung eines Generators ist nur dann sinnvoll, wenn sichergestellt wird, dass der generierte Anwendungs-Code ausführbar, fehlerfrei und möglichst robust ist. Durch eine starke Typisierung der Spezifikationssprache wird die Möglichkeit der Spezifizierung von fehlerhaften Web-Anwendungen erheblich eingeschränkt. Des Weiteren werden mit Hilfe von Konsistenzüberprüfungen zur Übersetzungszeit weitere Fehler in der Spezifikation vorzeitig aufgedeckt. Zum Beispiel werden fehlende Referenzen oder nicht initialisierte Variablen erkannt. Um die Robustheit der verwendeten Code-Muster zu testen, wurden Test-Anwendungen erstellt und auf verschiedenen Plattformen mit verschiedenen Web-Browsern ausgeführt.

Der generierte Code erfüllt ebenso bestimmte Qualitätsansprüche. Neben der Verwendung von Standards wie HTML, PHP und wurden auch Cascading Style Sheets (CSS) eingesetzt. Damit lässt sich der Inhalt von den Darstellungseigenschaften trennen. Durch die Referenzierung von CSS innerhalb von HTML-Seiten erzielt man eine zusätzliche Kompaktheit des generierten Codes. Zusätzlich ist der erzeugte HTML-Code XML-konform.

Das weiterführende Beispiel (Autohaus) hat die Praxistauglichkeit von PaderWAVE

gezeigt. Es lassen sich größere Web-Anwendungen einfach und schnell entwickeln und das ganz ohne Expertenwissen.

8.2 Erfahrungen der Projektgruppe

Die Projektgruppe neigt sich langsam dem Ende zu. Deshalb wollen wir an dieser Stelle ein kurzes Fazit über die Projektgruppenarbeit selbst ziehen. Die Eindrücke aller Teilnehmer sind durchweg positiv. Wenn man sich mit Kommilitonen unterhält, die an anderen Projektgruppen teilgenommen haben, ist dies keine Selbstverständlichkeit. Im Gegensatz zu vielen anderen Prüfungen während des Informatik-Studiums, ist die Projektarbeit sehr stark von der Gruppenarbeit geprägt. Um das vorgegebene Projektziel zu erfüllen, ist es wichtig Teilaufgaben zu planen, zu verteilen und zu koordinieren. Um dies erfolgreich umzusetzen, sind neben den fachlichen Kenntnissen besonders *Soft-Skills* wie Teamarbeit, Kommunikationsfähigkeit und Kooperationsfähigkeit gefragt.

Zu Beginn der Projektgruppe wurden Aufgaben wie Planung und Koordination noch von den Betreuern übernommen. Mit der Zeit wurden diese Aufgaben immer mehr von Mitgliedern der Gruppe übernommen. Dies hat uns den Einstieg in die Selbstorganisation stark erleichtert. Wichtig war auch das Verteilen von Verantwortlichkeiten. Dabei wurden die persönlichen Interessen und Fähigkeiten der einzelnen Mitglieder berücksichtigt. Jedes Mitglied war aber auch bereit, seine persönlichen Interessen, zum Beispiel bei der Findung von Terminen, zurückzustellen. Dies hat die Koordination der Projektgruppe stark vereinfacht. Innerhalb der Gruppe herrschte immer eine gute Atmosphäre. Auch gegen Ende der Projektphase, als es stressig wurde, nahm die Motivation keinen Abbruch.

An dieser Stelle möchten wir uns auch bei den beiden Betreuern der Projektgruppe Carsten Schmidt und Michael Thies für die ausgezeichnete Betreuung bedanken.

Literaturverzeichnis

[1] Projektgruppe: *Paderborn Web Application builder in a Visual Environment* Online: http://ag-kastens.upb.de/lehre/paderwave

[2] Fachgruppe Prof.Dr.Uwe Kastens: *Programmiersprachen und Übersetzer* Online: http://ag-kastens.upb.de

[3] M. Lohmann, S. Sauer, T.Schattkowsky, ProGUM-Web: Tool Support for Model-Based Development of Web Applications, Universität Paderborn, 2003

[4] K. Jamroendararasame, T. Suzuki, T. Tokuda, A Diagram Approach to Automatic Generation of JSP/Servlet Web Applications, Tokyo Institute of Technology

[5] T. Shimomura, A page-transition framework for image-oriented Web programming, University of Tokushima

[6] DEViL *Development Environment for Visual Languages* Online: http://ag-kastens.uni-paderborn.de/forschung/devil/index.php

[7] HTML: *HyperText Markup Language* Online: http://www.w3.org/MarkUp/

[8] PHP-Homepage: *PHP: Hypertext Preprocessor* Online: http://www.php.net

[9] Struts-Homepage: *The Apache Struts Web Application Framework* Online: http://jakarta.apache.org/struts/

[10] Cascading Style Sheets Level 2 Specification, W3C Recommendation 12-May-1998, Online: http://www.w3.org/TR/REC-CSS2

[11] Redirect after POST, TheServerSide.com August 2004, Online: http://theserverside.com/articles/article.tss?l=RedirectAfterPost