Comes — Spezifikation

Stefan Wehr
Prof. Dr. Peter Thiemann

14. Dezember 2005

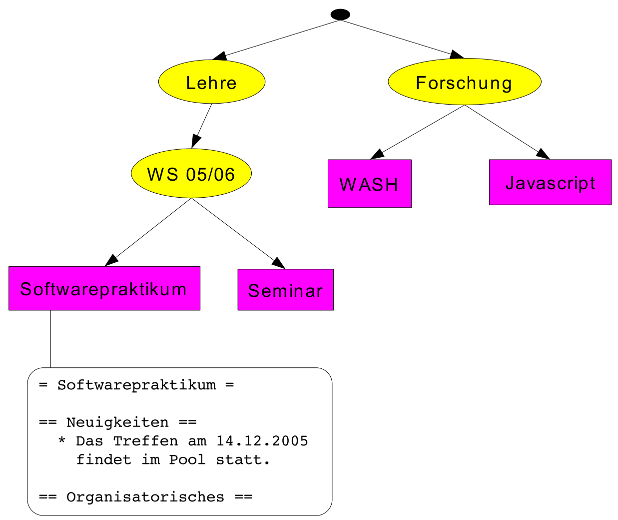
Komponenten

- Dokumentenbaum und Dokumententransformation
- Versionierung und Persistenz
- Zugriffskontrolle
- User Interface

Dokumentenbaum

- Repräsentiert die Seitenstruktur
- Besteht aus Dokumentencontainern (innere Knoten) und Dokumentenquellen (Blätter)
- Jeder Knoten wird durch einen Namen identifiziert
- Navigierbar mittels XPath Ausdrücken

Beispiel



Dokumentenquellen

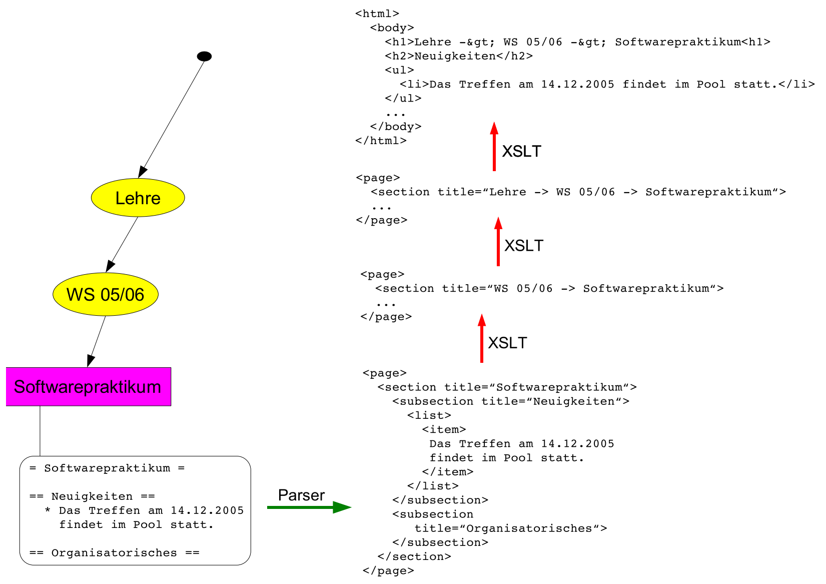
- Statisch: Quellcode liegt in Wiki Markup vor, Parser erzeugt daraus XML.
- Dynamisch: XML wird zur Laufzeit generiert. Damit können Navigationsmenüs etc. erzeugt werden.

Vorerst: Einschränkung auf statische Dokumentenquellen

Dokumententransformation

- An Knoten im Dokumentenbaum können XSLT Stylesheets gehängt werden
- XSLT Stylesheets transformieren den XML Code einer Dokumentenquelle
- Die Stylesheets auf dem Pfad von einer Dokumentquelle zur Wurzel des Dokumentenbaums werden nacheinander ausgeführt, um die endgültige Darstellung zu erhalten.

Beispiel



Versionierung

- Jeder Dokumentenquelle wird eine Version $n \in \mathbb{N}$ zugeordnet.
- Für ein Dokumentenbaum kann ein *Working Set* angelegt werden:
  - Kopie des Dokumentenbaums
  - Nur ausgewählten Benutzern zugänglich
  - Identifiziert durch einen eindeutigen Namen
  - *Commit*: Im Working Set vorgenommene Änderungen werden auf den ursprünglichen Dokumentenbaum übertragen (Konflikte möglich).
  - *Rollback*: Working Set wird verworfen.

Persistenz

Dokumentenbaum wird auf das Dateisystem abgebildet:

- Dokumentencontainer → Verzeichnis
- Dokumentenquelle → Verzeichnis, das als Inhalt die einzelnen Versionen des Dokuments enthält

Zugriffskontrolle

Rollenbasierte Zugriffskontrolle

- Jedem Benutzer werden ein oder mehrere Rollen zugeordnet.
- Mit jeder Rolle werden Berechtigungen verknüpft.

Benutzer

- Identifiziert durch eindeutigen Namen. Bsp.: `Helge`
- Assoziiert mit einer oder mehreren Rollen.

Rollen

- Identifiziert durch eindeutigen Namen. Bsp.: `admin`, `guest`, `student-sopra05`, `betreuer-sopra05`, `Helge`
- Können hierarchisch angeordnet werden. Bsp.: `student-sopra05` ist eine Subrolle von `betreuer-sopra05`.

Globale Berechtigungen

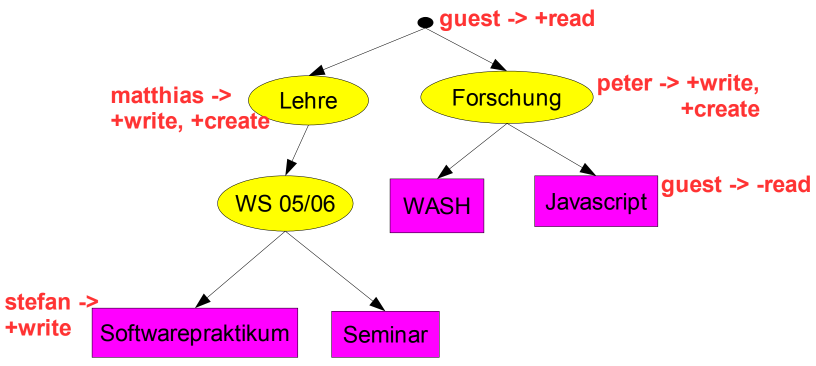
Die Berechtigungen für folgende Operationen sind systemweit gültig:

- Anlegen eines Working Sets: Rolle^*
- Sichtbarkeit eines Working Sets: $\mathit{Rolle} \to \mathit{Working\ Set}^*$
- Commit eines Working Sets: $\mathit{Rolle} \to \mathit{Working\ Set}^*$
- Rollback eines Working Sets: $\mathit{Rolle} \to \mathit{Working\ Set}^*$

Lokale Berechtigungen

- Folgende Rechte können von Dokument zu Dokument verschieden sein:
  - Lesen
  - Schreiben
  - Erstellen
- Mit jedem Knoten im Dokumentenbaum kann eine Abbildung $\mathit{ACL} : \mathit{Rolle} \to (\pm\ \mathit{Recht})^*$ assoziiert werden.
- Bestimmung der Rechte für ein Dokument d: Durchlaufe den Dokumentenbaum von der Wurzel zu d und addiere/subtrahiere die Rechte entsprechend.

Beispiel



User Interface

Aufgaben

1. Beantworten von Benutzeranfragen
2. Administrationsoberfläche

Eingesetzte Technologie:
Java Servlets ("CGI für Java")

Beantworten von Benutzeranfragen

Bsp.: Benutzer möchte Dokument
`http://proglang.informatik.uni-freiburg.de/lehre/ws0506/softwarepraktikum`
sehen

Vorgehen

- Werte XPath Ausdruck `lehre/ws0506/softwarepraktikum` auf dem Dokumentenbaum aus, erhalte dadurch Dokumentenquelle *d*.
- Frage *d* nach seiner Darstellung.
- Sende Darstellung an den Browser.

Administrationsoberfläche

- Verwalten von Dokumenten
- Verwalten von Working Sets
- Verwalten von Benutzer und Rollen

Servlets — "CGI für Java"

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class HelloWorld extends HttpServlet {
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
           throws ServletException, IOException {
        PrintWriter out = response.getWriter();
        out.println("Hello World");
    }
}
```