JOHANNES KEPLER
UNIVERSITÄT LINZ
Netzwerk für Forschung, Lehre und Praxis

eLearning-Unterstützung mittels WebDAV

Magisterarbeit

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Magisterstudium

Informatik

Angefertigt am Institut für Informationsverarbeitung und Mikroprozessortechnik (FIM)

Eingereicht von:

Thomas Göbl

Betreuung:

o.Univ.-Prof. Dr. Jörg R. Mühlbacher

Mag. iur. Dipl.-Ing. Dr. Michael Sonntag

Beurteilung:

o.Univ.-Prof. Dr. Jörg R. Mühlbacher

Linz, März 2004

Abstract

The description of the WebDAV-Explorer for the eLearning-System WeLearn, which is developed at the FIM – Institute for Information Processing and Microprocessor Technology, is the main part of this diploma thesis. The goal was to develop a module, which implements all functions of the WebDAV-Protocol in a performant way. The main points are authorisation, namespace operations, locking, and the developing of a user-friendly web based GUI, which provides the user a common look and feel similar to those used in standard explorers of operating systems.

First part of the diploma thesis are common arguments why distance learning is useful and why eLearning-Plattforms should be developed. Afterwards a description of the eLearning-Plattform WeLearn and a detailed overview about filesystems can be found. The main part is about the functionality and the properties of the WebDAV-Protocol. Finally follows a detailed description of the WebDAV-Explorer starting with its structure and ending with a user guide.

Kurzbeschreibung

In dieser Arbeit wird die Funktionalität des WebDAV-Explorers für die eLearning-Plattform WeLearn, die am Institut für Informationsverarbeitung und Mikroprozessortechnik (FIM) entwickelt wird, beschrieben. Das Ziel war, ein Modul zu schaffen, dass die gesamte Funktionalität, die WebDAV bietet, in einer performanten Weise implementiert. Dazu gehören Autorisation, Namespace Operationen, Locking und das Erstellen einer ansprechenden Web-GUI, die dem Benutzer das Look & Feel von gängigen Explorern verschiedener Betriebssysteme auch in Browsern bietet.

Am Anfang der Arbeit werden Argumente für Fernunterricht dargestellt um eine Motivation für die Entwicklung einer eLearning-Plattform zu geben. Anschließend wird die eLearning-Plattform WeLearn vorgestellt. Es folgt eine ausführliche Übersicht über Dateisysteme. Anschließend wird die Funktionalität, Eigenschaften und Vorteile von WebDAV vorgestellt. Das 5. Kapitel ist eine genaue Beschreibung des WebDAV-Explorers, angefangen von der Struktur bis hin zu einem Handbuch für Anwender.

Danksagung

An dieser Stelle möchte ich mich bei allen bedanken, die zum Gelingen dieser Arbeit beigetragen haben.

An erster Stelle möchte ich meinen Eltern danken, die mir das Studium ermöglicht haben und mich in allen Bereichen unterstützt haben.

Für die gute Betreuung und wertvollen wissenschaftlichen und organisatorischen Tipps möchte ich Herrn Mag. iur. Dipl.-Ing. Dr. Michael Sonntag besonders danken.

Weiters danke ich Herrn o.Univ.-Prof. Dr. Jörg R. Mühlbacher für die wertvollen Tipps und Anregungen, sowie die mir gebotene Möglichkeit an dem Projekt WeLearn mitzuarbeiten.

Sehr hilfreich waren auch die vielen Anregungen der WeLearn-Mitarbeiter, besonders möchte ich mich bei Dipl.-Ing. Dr. Susanne Loidl, MSc. Alexandros Paramythis und Dietmar Stoiber bedanken.

Inhaltsverzeichnis

1. Fernunterricht

Fernunterricht ist eine Form der Weiterbildung, die nach dem deutschen Fernunterrichtsgesetz folgendermaßen definiert ist:

> Fernunterricht im Sinne des Gesetzes ist die auf vertraglicher Grundlage erfolgende, entgeltliche Vermittlung von Kenntnissen und Fähigkeiten, bei der
>
> a) der Lehrende und der Lernende ausschließlich oder überwiegend räumlich getrennt sind und
>
> b) der Lehrende oder sein Beauftragter den Lernerfolg überwacht.

Das heißt, Fernunterricht ist eine Bündelung pädagogischer Maßnahmen zur Führung von Lernenden unter Berücksichtigung ihrer Lernvoraussetzungen und ihrer Lernziele ganz oder großteils durch Informationen, Steuerung und Motivation über eine räumliche Distanz.
[Fernunterricht_Def], [Fernunterricht_Ratgeber]

1.1. Gründe für Fernunterricht

Fernunterricht, darunter fallen Fernkurse und Fernstudien, ist oft für viele Menschen der einzige vernünftige Weg, sich weiterzubilden. Es ist auf einfache Alltagsprobleme, wie die Verbindung von Beruf und Familie zurückzuführen, dass es zeitlich nicht möglich ist in Kursen, die an Universitäten oder Weiterbildungsinstituten angeboten werden, teilzunehmen. Auf Grund der Flexibilität, die dem Lernwilligen durch Fernunterricht geboten wird, nutzen heute viele Personen diese Unterrichtsform um Abschlüsse nachzuholen, oder einen neuen Bildungsweg einzuschlagen.
Unterstützt wird diese Unterrichtsform durch staatliche Kontrolle der Fernlehrinstitute sowie der Schaffung von Fernuniversitäten und dem Angebot von Fernstudien an herkömmlichen Universitäten.

1.2. Vorteile

Viele Vorteile für Fernunterricht können aufgezählt werden:

- Fernunterricht ist lernzeitflexibel: Der Lernende kann selbst bestimmen, wann er lernen möchte. Dadurch kann er den Unterricht an sein Umfeld (Familie, Beruf,…)

und seine eigene Lerngeschwindigkeit anpassen. Welche Stoffteile wie lange gelernt werden bleibt dem Lernenden überlassen.

- Fernunterricht ist ortsunabhängig: Es kann nicht nur jeder Zeit gelernt werden, sondern auch von jedem beliebigen Ort aus. Dies bietet dem Lernenden ein Höchstmaß an Flexibilität.
- Fernunterricht ist universell geeignet: Dies gilt insbesondere für das Nachholen von Abschlüssen, wenn der Lernende bereits fachliches Wissen über das zu Lernende mitbringt.
- Fernunterricht wird staatlich unterstützt: In Österreich durch Initiativen des Bundesministeriums für Bildung, Wissenschaft und Kultur [bm:bwk] (zum Beispiel der Verein eLearning Austria [eLearning_Austria]).
- Aktualität: Da Fernunterricht eine relativ neue Unterrichtsform ist, ist meist das Unterrichtsmaterial neu oder neu überarbeitet.
- Fernunterricht ist leistungsbezogen: Der Fernkurs oder das Fernstudium sind meist rein auf eine bestimmte Leistung bezogen und nicht wie herkömmlich rein nach fixen (Zeit-)Semestern geregelt.
- Angenehme Atmosphäre: Den Lernenden erwarten keine langen Anfahrtszeiten zu Universitäten oder anderen Bildungseinrichtungen, überfüllte Hörsäle und andere Stressfaktoren.
- Kostensenkung: Es entfallen die Kosten für räumliche Infrastruktur sowie Anreisekosten.
- Globales Lehrangebot: Dem Lernenden stehen durch die örtliche Ungebundenheit weit mehr Auswahlmöglichkeiten für seinen Bildungsweg zur Verfügung.

[Dissertation_Reisinger], [Fernunterricht_Def], [Fernunterricht_Gesetz]

1.3. Probleme

Natürlich birgt diese neue Unterrichtsform auch Probleme für Lernende und Lehrende:

- Fernunterricht verlangt dem Lernenden ein großes Maß an Selbstständigkeit ab, da er durch die freie Zeiteinteilung ganz auf sich selbst gestellt ist. Den meisten Lernstoff muss der Lernende selbst erarbeiten. Bei auftretenden Fragen und Problemen muss der Lehrende erst kontaktiert werden (per Telefon, eMail,…) was den Lernfluss unterbricht.

- Der Lehrende kennt die Teilnehmer an seinen Kursen meist nicht persönlich. Dadurch ergeben sich Probleme beim Einschätzen der Lerngeschwindigkeit und dem Schwierigkeitsgrad der Aufgaben.
- Nicht alle Ausbildungen sind über Fernunterricht durchführbar. Bei Berufen mit großen praktischen Anteil, wie zum Beispiel Kraftfahrer oder verschiedenen Laborberufen sowie bei Erstausbildungen ohne Vorkenntnisse auf dem unterrichteten Gebiet, ist Fernunterricht nicht die geeignete Unterrichtsform.
- Fernunterricht kann zur Isolation der Lernenden führen. Es kann zu viel weniger oder gar keinen sozialen Kontakten zwischen den Personen, die einen Kurs gemeinsam belegen, kommen.
- Der Lehrende sowie der Lernende muss ein Grundwissen im Umgang mit den technischen Grundvoraussetzungen für Fernunterricht mitbringen.
- Der Lehrende sowie der Lernende müssen über die Hard- und Software verfügen um an modernem Fernunterricht teilzunehmen.
- Das Erstellen von neuem Unterrichtsmaterial oder der Adaptierung von vorhandenem Lehrmaterial für den Fernunterricht ist zeit- und kostenaufwendig.

[Dissertation_Reisinger], [Fernunterricht_Def]

1.4. Standards

In den folgenden Kapiteln werden die wichtigsten Konsortien, die sich mit Lernstandards beschäftigen kurz vorgestellt. Hauptsächlich geht es dabei um die Standardisierung der Erstellung von Lerninhalten (Metadaten, Architektur,…), aber auch um didaktische Empfehlungen.

1.4.1. IMS

Das IMS Global Learning Consortium, Inc. wird aus Mitgliedern, die von Universitäten bzw. aus dem kommerziellen oder staatlichen Bereich kommen, gebildet. Dieses globale Konsortium arbeitet nicht gewinnorientiert sondern finanziert sich rein über die Mitgliedsbeiträge.

Das IMS Konsortium unterteilt sich in einzelne Arbeitsgruppen, die Spezifikationen und Vorschläge für Standards zur Modellierung von Daten, die vor allem im vernetzten Teil des Fernunterrichts (eLearning und mLearning) benötigt werden. Dabei wird speziell auf das

Lernmaterial, die Aufzeichnung von Lernfortschritt und –erfolg der Benutzer und dem Austausch der Daten geachtet. Das Ziel sind allgemein gültige Standards, damit einheitliche Formate eine größt mögliche Interoperabilität zwischen verschiedenen Lernplattformen schaffen.

Die Spezifikationen von IMS umfassen Datenmodelle für folgende Bereiche:

- Abschätzung, Benotung und Lernfortschritt
- Metadaten
- Lernmaterial-Struktur
- Modellierung der Lernenden
- Organisation von Kursen

[mobilearn.at], [IMS]

1.4.2. ADL SCORM

Die amerikanische Organisation Advanced Distributed Learning Initiative (ADL) versucht mit SCORM ein vollständiges Refernzmodell für die Erstellung von Lernmaterialen und webbasierten Lernplattformen zu erstellen. Es fließen dabei Standardisierungsversuche anderer Organisationen in SCORM ein.

Die wichtigsten Punkte sind Austauschbarkeit von Lerninhalten über Systemgrenzen hinweg und ein API (Application Programming Interface) zur Kommunikation zwischen Lerninhalten (Contentpaketen) und Lernplattformen.

[mobilearn.at], [SCORM]

1.4.3. EML

EML (Educational Modeling Language) ist ein Vorschlag zur standardisierten Beschreibung von Lernmaterial. Es wird der Begriff eines „learning objects models“ (Lernobjekts) eingeführt. Ein Lernobjekt enthält mehrere „units of study“ (Lerneinheiten). Zum Beispiel: Das Lernobjekt ist ein Kurs, die einzelnen Lerneinheiten sind die darin behandelten Themen.

Das Objektmodell wird in einem XML Schema abgebildet, wobei hier Metadaten und Inhalte in einem Dokument vereint werden.

Da es sich bei EML um eine reine Beschreibungssprache handelt, limitiert dies das Einsatzgebiet im Gegensatz zu anderen Standards.

[mobilearn.at], [EML]

1.4.4. IEEE LTSC

Das zum IEEE (Institute of Electrical and Electronics Engineers, Inc.) gehörende LTSC (Learning Technology Standards Committee) forscht in verschiedenen Arbeitsgruppen an Standards für die Entwicklung von Lernplattformen und der Erstellung von Lernmaterialen.
Die Arbeitsbereiche der Workgroups können in fünf Hauptgebiete gegliedert werden:

- Allgemein
- Lernerbezogen
- Inhaltsbezogen
- Daten und Metadaten
- Management-Systeme und Anwendungen

Aus den Bemühungen von LTSC gingen bis jetzt vier viel versprechende Spezifikationen hervor, die bereits in kommerziellen Produkten eingesetzt werden und auch von anderen Standardisierungsgruppen (wie zum Beispiel IMS und SCORM) verwendet werden:

- LTSA (Learning Technology System Architecture): beschreibt die Systemarchitektur von Lernplattformen.
- LOM (Learning Object Metadata): dient zur Beschreibung von digitalen und nicht-digitalen Lernunterlagen und ermöglicht dadurch eine leichtere Identifikation und den Austausch von Modulen zwischen verschiedenen Lernplattformen.
- PAPI (Public and Private Information): dient zur Beschreibung von Daten der Lernenden. Dabei wurden persönlichen Informationen, Informationen über das Lernverhalten und die Lernleistung, über Arbeiten und Abschlüsse des Lernenden sowie Benutzerpräferenzen vorgesehen.
- CMI (Computer Managed Instruction): dient zum Austausch, zur Kombination und zur Administration von Kursen.

[mobilearn.at], [IEEE_LTSC]

1.4.5. CEN/ISSS WSLT

CEN/ISSS (European Committee for Standardization/Information Society Standardization System) wurde 1997 gegründet, um der europäischen Informationsgesellschaft mit Normen und Standards einen Vorteil im internationalen Wettbewerb zu verschaffen.
Der CEN/ISSS WSLT (Learning Technology Workshop) gliedert sich in mehrere Arbeitsgruppen, die sich mit allgemeinen Fragen des Fernunterrichts befassen. Dies beginnt

bei unterschiedlicher Sprache und Schrift, verschiedenem Verständnis von Dingen in unterschiedlichen Kulturkreisen und endet bei der Versionierung von Lernmaterialen.
Besonderes Augenmerk wird auf die Wiederverwendbarkeit von Unterrichtsmaterialen gelegt, wobei hier auch auf die Frage des geistigen Eigentums geachtet wird.
CEN/ISSS WSLT verwendet den IEEE LTSC LOM-Standard zur Beschreibung der Metadaten des Lernmaterials.
[mobilearn.at], [WSLT]

1.5. Lernplattformen

Eine Lernplattform (Online Learning Plattform – OLP) ist eine webbasierte Umgebung für kooperatives Lernen und Unterrichten. Es verbindet die Präsentation von Lernunterlagen (Inhalte) mit Kommunikationssystemen (Chat, Foren,…) und interaktiven Elementen (Selbsttests,…). [Legal_Eng_Sonntag]
Beispiele für Lernplattformen sind WebCT [WebCT], Blackboard [Blackboard] und WeLearn [WeLearn], das im Kapitel 2 genau beschrieben wird.

1.6. eLearning

eLearning ist ein eine Art des Fernunterrichts, bei der verschiedene Dienste des Internets wie News, Email, World Wide Web, Chat, Internet-Telefonie, Video-Konferenzen,… verwendet werden.
eLearning ist meist multimedial, da oft mehrere der genannten Informations- und Kommunikationstechnologien gleichzeitig verwendet werden. Dabei unterscheidet man drei Arten des eLearnings:

- Teletutoring/Telecoaching: Bei dieser Form des eLearnings unterstützt ein Tutor den Lernenden durch fachliche und organisatorische Betreuung. Dabei werden meist asynchrone Kommunikationsformen verwendet (Email, Foren,…).
- Open Distance Learning: Der Lernende bekommt strukturierte, didaktisch aufbereitete Lernunterlagen auf einer Lernplattform (oder im einfachsten Fall auf einem Server) bereitgestellt. Ob und wann sich der Lernende das Unterrichtsmaterial anschaut und lernt obliegt ihm, er wird aber meist ermutigt sich in Foren oder Chats mit anderen Lernenden auszutauschen um den maximalen Nutzen aus einem Kurs zu ziehen.

- Teleteaching/Distance Lecturing: Dies ist eine Lehrveranstaltung oder ein einzelner Vortrag der per Video-Konferenz abgehalten wird. Das heißt der Vortragende und die Lernenden müssen nicht am selben Ort, aber zur gleichen Zeit anwesend sein.

[eLearning_Def], [ODL]

1.7. mLearning

Der relativ junge Begriff des mLearnings steht für Mobile Learning. Die Idee von Fernunterricht ist es, jederzeit an jedem beliebigen Ort lernen zu können. Das Konzept des eLearnings, die Unterstützung des Lernens durch PC und Internet haben diese Freiheiten beeinträchtigt.

Neueste Technologien ermöglichen es dem Lernenden jedoch wieder diese ursprüngliche Flexibilität zurückzugewinnen:

- WLAN: Die Wireless LAN (Local Area Network) Technologie bietet dem Lernenden einen Internetzugang solange man sich in der Nähe eines Hotspots befindet. Viele Universitäten, öffentliche Gebäude aber auch Cafes und Restaurants verfügen bereits über solche Internetzugänge.
- Laptops: Der tragbare PC, auf dem Lernende das Unterrichtsmaterial lernen kann und Übungen ausarbeiten kann.
- PDA und multimediafähige Handys: Auch kleine Handhelds eignen sich ideal um unterwegs etwas zu lernen. Diese dienen aber mehr als Unterstützung und werden den PC auf Grund der geringen Bildschirmgröße und Bedienungsproblemen nicht ersetzen können.

Diese Neuerungen bringen auch neue Anforderungen an Lernplattformen. Die Inhalte müssen nicht nur in einem Format für den PC, sondern auch in anderen Darstellungsarten für Handhelds bereitgestellt werden. Weiters sollten die Inhalte auch ohne Internetverbindung verfügbar sein, da diese vor allem bei Mobiltelefonen sehr teuer sein kann.

[mLearning], [mobilearn.org]

2. WeLearn

WeLearn steht für Web Environment for Learning und wird derzeit in der dritten Version (Codename Emerald) am Institut für Informationsverarbeitung und Mikroprozessortechnik [FIM] der Johannes Kepler Universität Linz [JKU] entwickelt. Der WebDAV-Explorer, der den praktischer Teil dieser Diplomarbeit darstellt, wird ein Modul dieser neuen Version sein.

WeLearn ist ein Framework, das sich bisher aus vier Hauptkomponenten zusammensetzt hat:

- Onlinefunktionalität: Die Plattform als solche, die die nötigen Grundvorrausetzungen schafft, auf die das System aufbaut.
- Zielgruppen: Es können unterschiedlich Einstellungen für verschiedene Anwendungsgebiete und Anwendergruppen getätigt werden, zum Beispiel für Schulen, Universitäten oder Erwachsenenweiterbildung.
- Templates für Kurse: Dies ermöglicht dem Lehrenden leicht Kurse, die seinen Anforderungen entsprechen, zu erstellen.
- WeLearn Offline Konverter: Dieser dient dazu Onlinekurse aus dem CPS-Format [IMS] in ein Offline-Format (zum Beispiel: (D)HTML oder Applets) zu konvertieren um die Inhalte auch ohne Internetverbindung betrachten zu können. Der Vorteil liegt darin, Kurse zum Beispiel auf CDs verteilen oder per Email verschicken zu können.

Die neue Version von WeLearn wird um folgende Eigenschaften erweitert:

- Workflowkomponente: Diese Komponente dient dazu Abläufe zu automatisieren. Zum Beispiel können die Abgaben einer Übung an Tutoren weitergeleitet werden, anschließend bekommt sie der LVA-Leiter und am Ende geht die korrigierte Version zurück an die Studenten.
- Integration von Agenten: Agenten können wertvolle Dienste bei der Suche nach Kursen bieten, zum Beispiel mittels systematischer Suche in Metadaten. Weiters kann deren Einsatz die Kursorganisation und Kursdurchführung erleichtern, indem sie etwa Kursteilnehmer verständigen sobald sich Termine ändern oder neues Kursmaterial zur Einsicht auf der Plattform vorliegt.
- Personalisierung: Dient zur Anpassung des ganzen Systems an eigene Ansprüche und Vorlieben. Dies beginnt bei Einstellungen für die GUI, geht über automatische Benachrichtigungen bei Änderungen bis zur Erkennung von Gewohnheiten des Benutzers und geeigneter Reaktionen darauf.

[Legal_Eng_Sonntag]

Das Ziel von WeLearn ist es ein Framework zu schaffen, dass sowohl dem Lehrenden als auch dem Lernenden eine spezielle Plattform bietet, die das Lernen positiv unterstützt und fördert. WeLearn ist universell einsetzbar und für Benutzer einfach zu bedienen. Navigation und Erscheinungsbild sind leicht verständlich und intuitiv bedienbar gehalten.

Abbildung 1: Erscheinungsbild von WeLearn [Dissertation_Reisinger]

Aufgrund der Architektur als Framework ist es leicht für neue Aufgaben aber auch Technologien adaptierbar und kann so genau auf Kundenwünsche und spezielle Lernsituationen hin umgebaut werden.

WeLearn ist in JAVA implementiert und wurde unter Berücksichtigung moderner eLearning-Standards entwickelt, um Plattformunabhängigkeit und Wiederverwendbarkeit zu gewährleisten.

WeLearn ist frei verfügbar.

[Dissertation_Reisinger], [WeLearn_Framework]

2.1. Architektur

Folgende Abbildung zeigt die Architektur für die neue Version von WeLearn (Emerald).

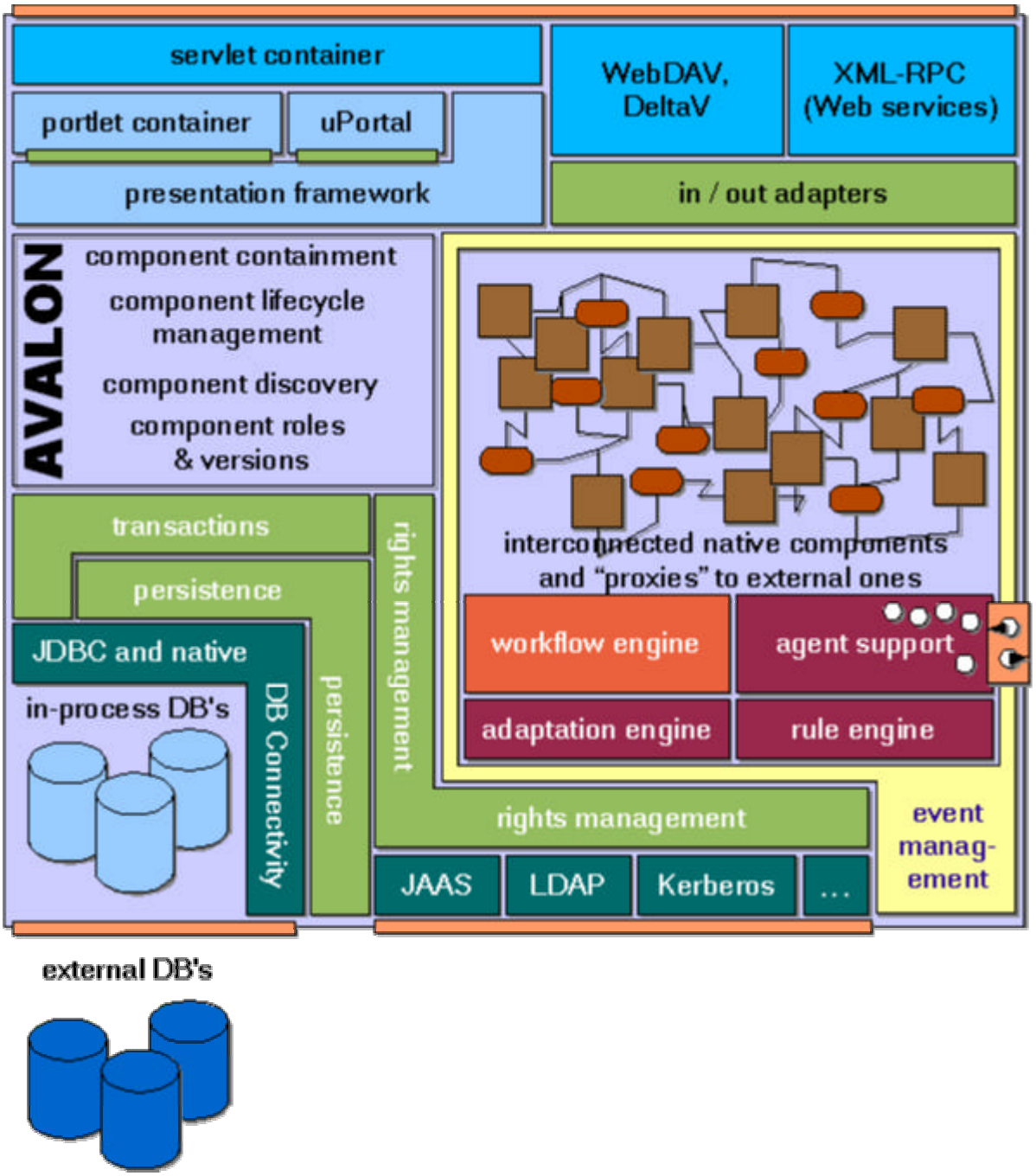

Abbildung 2: Struktur von Emerald

Diese Sicht ist hauptsächlich für die Entwickler der WeLearn-Plattform interessant und bietet einen genauen Einblick auf die verwendeten Technologien. Der Benutzer merkt nichts von diesen Modulen und Schichten nutzt aber alle Vorteile, die diese bieten. Die Kommunikation zwischen Benutzer von WeLearn (also Lernenden und Lehrenden) und der Plattform funktioniert auf dem Client-Server-Prinzip:

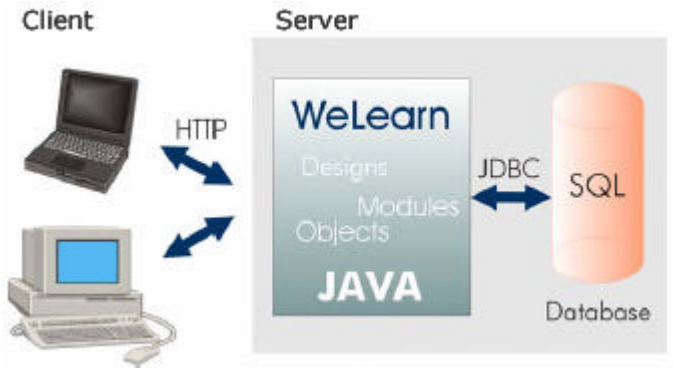

Abbildung 3: Client-Server-Sicht von WeLearn [WeLearn]

Der Benutzer kann dazu jeden gängen Webbrowser (z.B.: Internet Explorer oder Mozilla) verwenden.

Das WeLearn-System ist objektorientiert programmiert, d.h. alles wird als Objekt gesehen (Benutzer, Verzeichnisse, Dokumente, Foren,…). Dadurch wird das System modular gehalten um neue Objekte leicht hinzufügen und alte Objekte genauso leicht entfernen zu können. Bei WeLearn handelt es sich nicht einfach um eine Web-Applikation, es ist vielmehr ein Programm mit einem HTML-Interface.
Alle neu entwickelten Module müssen nach einem genau spezifizierten Interface der Klassen (API) erstellt werden, wodurch diese auch während der Laufzeit austauschbar sind.

Die technischen Vorraussetzungen für den Server sind ein Linux oder Windows 2000/3 Server mit 256MB Arbeitsspeicher und installierter JAVA 2 Umgebung.
[Dissertation_Reisinger], [WeLearn], [WeLearn_Framework]

2.2. Konfiguration

Durch die Konfiguration des WeLearn-Systems kann sich die Funktionalität stark ändern, deshalb wird in diesem Kapitel auf jene Funktionen eingegangen, die üblicherweise bei eLearning benötigt werden:

- Administration
- Präsentation der Kursunterlagen
- Unterstützung des Lernprozesses

WeLearn bietet dem Administrator die Verwaltung von Usern (Lernenden, Lehrenden, Tutoren, Administratoren,…) und Gruppen. Weitere Aufgaben des Administrators sind die Konfiguration des Systems, das Management der angebotenen Kurse und die Verwaltung des Kursmaterials um die eLearning-Plattform den Anforderungen anzupassen. WeLearn ist einsetzbar in Schulen, für Universitätskurse und für Erwachsenenweiterbildung.
Aus der Vergangenheit wurde gelernt, dass Kommunikation unter den Lernenden sowie zwischen Lehrenden und Lernenden maßgeblich den Lernprozess unterstützen und beschleunigen kann. Deshalb bietet WeLearn Diskussionsforen und Chats, die an beliebigen Stellen im System eingebaut werden können.
Jeder Benutzer bekommt mittels virtuellem Dateisystem ein eigenes Verzeichnis zugewiesen, wohin er beliebige Dateien hochladen kann und diese so anderen Benutzern zugänglich machen kann. Dies fördert die kollaborative Arbeit unter den Lernenden.
WeLearn ist einerseits Benutzer-zentriert, andererseits Kurs-zentriert. Kurse, die aus verschiedenen Dokumenten, gemeinsamen Ordnern und Foren bestehen, spielen eine entscheidende Rolle innerhalb der Plattform. Genauso wichtig sind aber auch die Benutzer. Für diese bietet WeLearn Optionen zur Personalisierung, sodass nur jene Teile des Systems angezeigt werden, die für sie von Relevanz sind. Dies ist nicht nur auf die Plattform an sich beschränkt sonder die Personalisierung kann auch auf Kurse und Kursmaterial angewendet werden.
WeLearn bietet ein ausgeklügeltes und weitreichendes Rechtesystem, das den Zugriff auf alle Objekte überwacht.
[WeLearn_Framework]

2.3. Kursmaterial

WeLearn limitiert die Inhalte von Kursen nicht, es kann jeder beliebige Datentyp eingebettet werden. Falls der Browser auf der Clientseite ein geeignetes Plugin installiert hat, wird der Datentyp auch direkt mit dem Kursmaterial angezeigt. Solche Datentypen sind zum Beispiel Dokumente (pdf, doc,…), Audio und Video (avi, mpg, wav, mp3,…), Flashanimationen, Applets, etc.
Um die Wiederverwendbarkeit der Kurse (auch in anderen Lernplattformen) zu gewährleisten, werden Kurse in WeLearn nach der Content Packaging Specification des IMS Global Learning Consortiums [IMS] erstellt. Dabei wird die hierarchische Struktur der Kurse in einem XML-Manifest beschrieben.

Mittels WeLearn Offline Konverter werden die gleichen Daten, die online in der Plattform verfügbar sind, fast mit dem gleichen Aussehen offline dargestellt.
[Dissertation_Reisinger], [WeLearn_Framework]

3. Dateisysteme

Laut Informatikhandbuch [Informatik-Handbuch_99] ist der Begriff „Datei“ folgendermaßen definiert: Eine Datei ist eine Sammlung von Daten, die auf einem Permanentspeicher gehalten wird.
Um viele Dateien auf so einem Speicher (Festplatte, CD, DVD,…) zugänglich zu machen, wird eine strukturierte Anordnung benötigt. Ein Dateisystem (engl.: Filesystem) bietet diese Funktionalität. Es stellt einen Abstraktionsmechanismus dar, um die Dateien (Daten) - ohne detailliertem Wissen des Benutzers über die konkrete physische Datenablage - zur Verfügung zu stellen und bietet darüber hinaus einen Schutzmechanismus der gewährleistet, dass nur jene Benutzter Daten lesen und schreiben, die auch die erforderliche Berechtigung dazu besitzen.

Die Daten in einer Datei können in unterschiedlicher Form organisiert sein, die einfachste und am weitesten verbreitete Form der Organisation ist die einer linearen Folge von Daten (Bytes).

Dateisysteme machen meist Annahmen über die Semantik der enthaltenen Daten einer Datei (z.B.: bei Textdateien, hier wird oft ein Zeichensatz vorausgesetzt, was zu Inkompatibilität zwischen verschiedenen Dateisystemen führen kann).

Bei Dateisystemen unterscheidet man zwischen physischer Struktur und logischer Struktur:

- Die physische Struktur ist jenes Schema, das benutzt wird, um die Daten auf die Geometrie des Speichermediums abzubilden (z.B. auf Sektoren, Zylinder, Plattenoberfläche,…), sowie die Algorithmen, die erlauben diese Daten wieder zu lesen. Beim Schreiben werden zwei Schritte durchlaufen. Zuerst erfolgt die Abbildung von Dateien auf fortlaufend nummerierte Datenblöcke fester Länge. Der zweite Schritt ist die Abbildung dieser Datenblöcke auf die Geometrie des Datenträgers. Da dieser Schritt abhängig von dem Speichermedium ist, übernimmt bei moderner Hardware die Steuereinheit des Mediums (z.B.: der SCSI-Controller) diese Aufgabe. Diese Controller sind so optimiert, dass sie die schnellsten Zugriffszeiten zu erreichen versuchen.

Die Aufgaben des Betriebssystems sind also die Zuordnung von Datenblöcken zu Dateien, Buchführung über ungenutzte Datenblöcke, meiden von fehlerhaften Datenblöcken und das Verwalten von Verzeichnissen. Die Informationen über die Datenstruktur müssen ebenfalls auf dem Speichermedium gehalten werden, wobei berücksichtigt werden muss, dass diese schnell zugänglich sein sollen und im Fehlerfall der Schaden minimal sein soll.

Man unterscheidet drei Arten für das Auffinden von Datenblöcken einer Datei:

- Interne Verkettung:

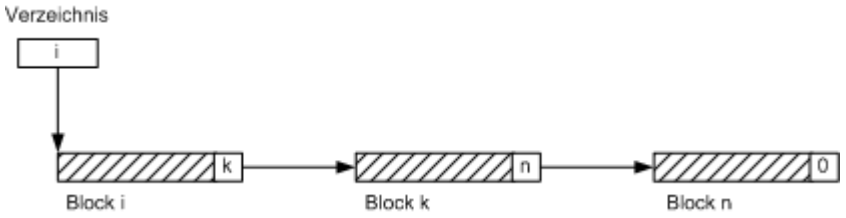

Abbildung 4: interne Verkettung [Informatik-Handbuch_99]

Bei dieser einfachen Methode ist keine zusätzliche Datenstruktur notwendig, es muss lediglich die Indexnummer (Start der Datei) und die Länge einer Datei (um zu wissen wie weit gelesen werden muss) bekannt sein um diese zu lesen, da eine Datei nach der anderen geschrieben werden kann (in variabler Länge). Dies führt jedoch bei wiederholtem Löschen und neu Beschreiben zu starker interner Fragmentierung. Der Hauptnachteil ist, dass Dateien, die in einem Verzeichnis liegen, nur sequentiell gelesen werden können. Daraus ergeben sich Performance-Probleme wenn der Benutzer mehrere Dateien gleichzeitig bearbeitet.

- Externe Verkettung:

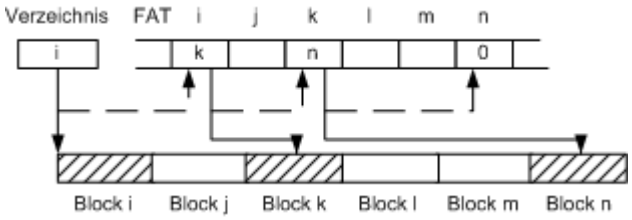

Abbildung 5: externe Verkettung [Informatik-Handbuch_99]

Dieses Schema erlaubt es, Dateien auf beliebige Datenblöcke zu verteilen. Dazu wird eine zusätzliche Datenstruktur benötigt. Dafür wird zum Beispiel eine Hilfstabelle (FAT) benutzt. Ein Beispiel hierzu siehe Kapitel 3.1.1..

- Indexblock:

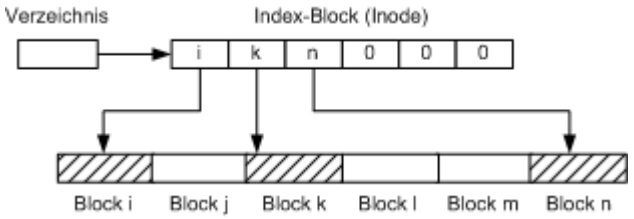

Abbildung 6: Indexblöcke [Informatik-Handbuch_99]

Hier wird keine Verkettung benutzt, sondern ein dateispezifisches Feld (Array) von Indizes, in dem die Reihenfolge der Blöcke, in denen die Daten einer Datei gespeichert sind, festgelegt ist. Der Eintrag der Datei im Verzeichnis verweist auf dieses Feld, das entweder der erste Datenblock der Datei ist (Indexblock) oder ein Verweis auf einen anderen Indexblock (da ein Indexblock nicht ausreicht). Diese Indexblöcke können verkettet oder in einer Baumstruktur angeordnet werden. Diese Methode wird bei Unix-Systemen verwendet, dort heißen die Indexblöcke Inodes (genaue Beschreibung in Kapitel 3.1.4.).

- Die logische Struktur ist die Darstellung des Dateisystems für den Benutzer. Dabei wird zwischen Verzeichnissen (directories) und Dateien (files) unterschieden, wobei meist ein Verzeichnis ebenfalls eine Datei ist, jedoch versehen mit zusätzlicher Semantik für das Betriebssystem. Die Struktur der Dateien und Verzeichnisse wird bei modernen Dateisystemen in einem Verzeichnisbaum abgebildet.

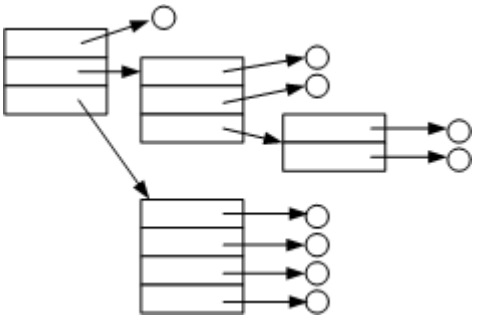

Abbildung 7: Verzeichnisbaum [Informatik-Handbuch_99]

In einem Verzeichnisbaum ist eine Datei nicht mehr durch ihren Namen eindeutig bezeichnet (es kann Dateien mit gleichen Namen geben) sondern nur noch durch ihren Pfad. Der Anwender kann eine Datei durch ihren absoluten Pfad (vom Root beginnend aus) ansprechen, aber auch über den relativen Pfad, dieser geht vom Bezugsverzeichnis (current directory) aus. Relative Angaben sind meist kürzer und bieten mehr Flexibilität für den Benutzer, werden aber bei Ausführung von Operationen vom Kommando-Interpretierer des Betriebssystems zu absoluten Pfaden gewandelt.

Die Benennung von Verzeichnissen ist Betriebssystem-abhängig, jedoch in der Regel längenlimitiert und besteht aus einem reduzierten Zeichensatz. Eine ausführliche Beschreibung der Unterschiede zwischen UNIX-Betriebssystemen und Windows ist in Kapitel 3.2.1. zu finden.

[Informatik-Handbuch_99], [Betriebssysteme_Adam], [Progtechnik_Groeller]

3.1. Übersicht über verschiedene Dateisysteme

So vielschichtig wie die Landschaft der Betriebssysteme ist, so einfallsreich waren die Entwickler auch bei der Schaffung von verschiedenen Dateisystemen, um gewissen Anforderungen (Sicherheit, Konsistenz, Zugriff über Netzwerke,…) zu genügen. Viele sind auch mit neuen Anforderungen (größere Festplatten, Sicherheitsbedenken,…) mitgewachsen und so gibt es oft verschiedene Versionen eines Dateisystems. Im Folgenden werden die wichtigsten Vertreter näher erläutert.

3.1.1. FAT

FAT (File Allocation Table - Dateizuordnungstabelle) wurde von Microsoft ab 1977 entwickelt. Zurzeit gibt es mehrere Versionen von FAT:

- FAT12 (12 bit FAT)
- FAT16 (16 bit FAT)
- HPFS (OS2 FAT)
- V-FAT/FAT32 (32 bit FAT)

FAT baut auf dem Prinzip der externen Verkettung auf. Jede Harddisk kann in eine oder mehrere Partitionen (logische Laufwerke) aufgeteilt werden. Jede Partition besteht aus einer Folge von Sektoren, wobei jedem Sektor eine logische Sektornummer beginnend mit 0 zugewiesen wird. Eine Gruppe von Sektoren bildet einen Cluster. Betriebssysteme arbeiten nur mit logischen Sektornummern, wodurch sie keinen Einfluss auf die physikalische Anordnung der Sektoren auf der Harddisk haben. Diese Aufgabe erledigt der Gerätetreiber, der für die Kommunikation zwischen Betriebssystem und Festplatte verantwortlich ist. Die verschiedenen Dateifunktionen des Betriebssystems beziehen sich auf Dateien und Verzeichnisse, so dass eine Schnittstelle zwischen diesen Ebenen benötigt wird. Die Verzeichnisstruktur und die FAT bilden diese.
Eine FAT-Partition ist folgendermaßen aufgebaut:

- zuerst kommt der Bootsektor
- dann die FAT sowie eine oder mehrere Kopien zu Sicherungszwecken
- anschließend das Wurzelverzeichnis (root)
- zuletzt der Dateien-Bereich

In der FAT ist verzeichnet, in welchen Clustern die einzelnen Dateien gespeichert sind, d.h. jede Datei wird als eine Aneinanderreihung von Clustern vermerkt, wobei diese Cluster nicht hintereinander stehen müssen (das Betriebssystem weist einer Datei beliebige Cluster zu). Dadurch entsteht das Problem der Fragmentierung. Hiervon gibt es zwei Arten bei FAT: zuerst jene, wo eine Datei verteilt auf mehrere Sektoren auf der Platte liegt, was zu starken Performance-Verlusten führt, da der Lesekopf der Harddisk oft zu verschiedenen Stellen bewegt werden muss während die Datei gelesen wird (externe Fragmentierung). Die zweite Art der Fragmentierung tritt bei sehr kleinen Dateien auf, die kleiner sind als ein Cluster. Der restliche Platz im Cluster bleibt ungenützt (interne Fragmentierung).
Weiters ist in der FAT vermerkt, welche Cluster unbelegt sind.
Die maximale Clusteranzahl für eine Partition ist beschränkt, bei FAT16 beträgt sie zum Beispiel 65536 Cluster. Bei einer Blockgröße von 512 Byte und einer maximalen Blockzahl von 64 pro Cluster ergibt sich eine maximale Partitionsgröße von 2048MB.

Das Dateisystem FAT bringt viele weitere Einschränkungen mit sich:

- Datei- und Verzeichnisnamen dürfen nur aus 8+3 Zeichen bestehen
- Bei Dateien, die kleiner als ein Cluster sind, sowie beim letzten Cluster von großen Dateien wird Speicherplatz verschwendet (interne Fragmentierung)

- Begrenzung der maximalen Partitionsgröße
- Begrenzung der maximalen Länge des Pfades zu einer Datei

Aus diesen Gründen wurde das Dateisystem FAT weiterentwickelt zu VFAT/FAT32, das lange Dateinamen (bis 255 Zeichen) unterstützt, das Erstellungsdatum sowie das letzte Änderungsdatum einer Datei speichert, Groß- und Kleinschreibung unterscheidet und die maximale Länge des Pfades zu einer Datei erhöht.
In der FAT wird auch Buch geführt, welche Datenblöcke frei oder beschädigt sind, daher ist hier kein weiterer Aufwand nötig, wie bei anderen Dateisystemen.
Das Konzept von FAT ist ein simpler aber effizienter Weg für die "linked allocation methode". Es wird vom MS-DOS (FAT16), OS/2 (HPFS), Windows 95 (FAT32) und Windows 98 (FAT32) verwendet. Auch neuere Betriebssysteme unterstützen FAT noch (Windows 2000, Windows XP,...) jedoch ist diesem Dateisystem längst von NTFS aufgrund seiner Vorteile der Rang abgelaufen worden.
[Informatik-Handbuch_99], [NTFS.com]

3.1.2. NTFS

NTFS (New Technology File System) wurde vom Microsoft zuerst für das Betriebssystem Windows NT (1993) entwickelt und später für neue Versionen von Windows (2000 und XP) weiterentwickelt (NTFS5). Der große Vorteil an NTFS gegenüber FAT ist, dass Zugriffskontrolle bis auf Dateiebene möglich ist. Weiter Sicherheitsvorteile sind das Transaktionskonzept (all-or-nothing), automatischen Rücksetzten (rollback) bei einem Systemabbruch während einer Operation (so kann kein inkonsistenter Zustand entstehen, da immer zum letzten sicheren Zustand zurückgekehrt wird) sowie die Unterstützung von RAID-1 (mirroring) und RAID-5 (block-level-parity).

NTFS behandelt kleine Dateien (kleiner als 1kb) und größere nicht gleich. Jede einzelne Datei bekommt am Speichermedium einen Bereich (record, 1 kbyte groß) zugewiesen, der in der MFT (Master File Table, dies ist eine zentrale Metadatei) angelegt wird, wo alle Attribute der Datei gespeichert werden. Es kann sein, dass eine kleine Datei noch innerhalb dieses 1 kByte-Bereiches Platz hat (resident) und nicht nach dem Index-Verfahren von der MFT aus verlinkt werden muss (nonresident). Längere Dateien bestehen aus einer Abfolgen von Läufen (runs), dies sind aufeinander folgende Blöcke auf dem Datenträger, wobei jeder dieser Läufe durch

eine Blocknummer relativ zum Datenträger (LCN), eine Blocknummer relativ zum Dateibeginn (VCN) und seiner Länge gekennzeichnet ist.

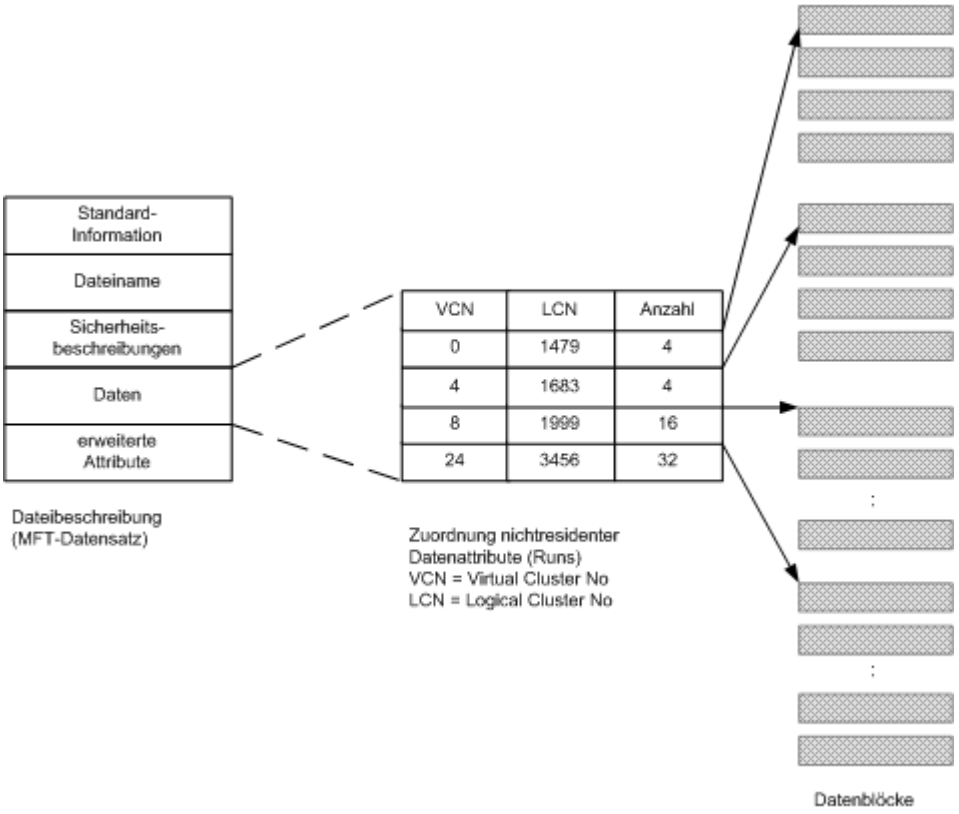

Abbildung 8: Dateibeschreibung in der MFT von NTFS [Informatik-Handbuch_99]

Damit die MFT nicht fragmentiert wird, reserviert das Betriebssystem nach dem Anlegen einer Partition 50% des vorhandenen Speichers für die MFT. Sobald die andere Hälfte mit Daten angefüllt ist, wird wieder ein Stück des Puffers freigegeben. Außerdem werden die ersten 16 Sektoren für den Bootsektor und den Bootstrap-Code reserviert.

Eine NTFS-Partition ist folgendermaßen aufgebaut:

Abbildung 9: NTFS-Partition

Der Bootsektor, der am Anfang der Partition stehen muss, verweist auf den Speicherort der MFT. Ungefähr in der Mitte der Partition wird eine Kopie der ersten vier Records der MFT gespeichert, um im Fehlerfall eine Reparaturmöglichkeit zu bieten. In der MFT wird ebenfalls

eine Logdatei mit den Änderungen von laufenden Operationen gespeichert, um wie vorhin erwähnt, im Fehlerfall diese Aktionen rückgängig zu machen und einen konsistenten Zustand zu erreichen.

Bei NTFS können beschädigte Datenblöcke nicht markiert werden wie bei FAT, deshalb ist für diese Datenblöcke eine eigene Datenstruktur notwendig. Im Fall von NTFS ist dies wie bei den meisten anderen Dateisystemen ein Bitvektor, der jeden Block als frei oder belegt/beschädigt markiert.

Jede Datei und jedes Verzeichnis wird als Set von Dateiattributen gesehen. Dies umfasst Standardinformationen (wie den Timestamp), eine Attributsliste (listet jene Plätze, wo die Attribute-Records gespeichert sind, die nicht in den MFT-Record passen, d.h. die Attribute einer Datei können mehrere nicht aufeinander folgende Records in der MFT umfassen), den Dateinamen, Sicherheitsinformationen (Dateibesitzer und Zugriffsrechte), eine Objekt ID, der Logged Tool Stream (Operationen werden in der NTFS-Logdatei gespeichert), der Reparse Point (der als Mountpunkt genutzt wird) und auch die Nutzdaten der Datei. Für spezielle Dateien gibt es noch zusätzliche Attribute:

- Verzeichnisse: Index Root, Index Allocation, Bitmap
- Volume Systemdateien: Volume Information, Volume Name

NTFS legt eine Reihe von versteckten Systemdateien an, die für den Benutzer nicht sichtbar sind. Sie liegen alle innerhalb der ersten 16 Records.

- MFT Record 0: Master File Table
- MFT Record 1: Master File Table 2
- MFT Record 2: Log File
- MFT Record 3: Volume
- MFT Record 4: Attribute Definitions
- MFT Record 5: Root File Name Index – das Rootverzeichnis
- MFT Record 6: Cluster Bitmap
- MFT Record 7: Boot Sector
- MFT Record 8: Bad Cluster File
- MFT Record 9: Security File
- MFT Record 10: Upcase Table – Konvertiert Dateinamen mit Kleinbuchstaben auf Großbuchstaben um Unicode-kompatibel zu sein.

- MFT Record 11: NTFS Extension File – Wird für optionale Extensions verwendet, zum Beispiel Quotas.

Die MFT Records 12-15 sind für zukünftige Systemdateien reserviert.

Weitere wichtigste Funktionen und Besonderheiten von NTFS sind:

- alle Systemdateien (außer dem Bootblock) sind frei verschiebbar
- die Dateinamen werden in Unicode gespeichert
- Groß- und Kleinschreibung wird gespeichert, jedoch werden die Groß- und Kleinbuchstaben nicht unterschieden (z.B.: klein.txt = KlEiN.TxT)
- lange Dateinamen werden unterstützt (bis 255 Zeichen, inklusive Erweiterung)
- das Trennzeichen zwischen Namen und Erweiterung ist ein Punkt, enthält ein Dateiname mehrere Punkte, so wird der letzte zur Trennung verwendet
- es wird FAT-Namensgenerierung aus Kompatibilitätsgründen unterstützt
- Datenkomprimierung wird unterstützt
- flexible Datenspeicherung durch LCNs und VCNs
- variable Clustergrößen: von 512 bytes bis 64 kbytes
- NTFS5 unterstützt das Verschlüsseln (Encrypting) von Dateien und Verzeichnissen
- NTFS5 bietet das Konzept von Sparse Files – hier werden nur sinnvolle Daten gespeichert; Daten die nur aus Nullen bestehen, werden nicht gespeichert, d.h. es können sehr große, aber fast leere, Dateien angelegt werden, sie belegen aber nur den Speicherplatz, den sie tatsächlich benötigen.
- Datenwiederherstellung wird unterstützt – mittels der in der Logdatei gespeicherten durchgeführten Operationen.
- mittels Disk Quotas kann der Plattenspeicher überwacht und der Zugriff limitiert werden.

NTFS ist das heute gängige Dateisystem bei Windows-Betriebssystemen.
[Informatik-Handbuch_99], [NTFS.com], [Dateiverwaltung_Heiss]

3.1.3. Virtual File System (Linux)

Das Virtual File System (VFS) entstand aus der Entwicklung von Linux. Am Anfang war Linux als Erweiterung des Betriebssystems Minix gedacht und unterstützte damals auch nur

das Minix-Dateisystem, das als effizientes und relativ fehlerfreies Stück Software galt. Dieses hat jedoch zwei Limitierungen: Erstens sind die Blockadressen in einem 16-bit Integer gespeichert, daraus resultiert eine maximale Partitionsgröße von 64mb. Zweitens haben Verzeichnis-Einträge eine fixe Größe und die Länge der Dateinamen ist auf 14 Zeichen beschränkt.

Auf Grund dieser Limitierungen wurden später zwei weitere Dateisysteme entworfen und für den Linux-Kernel implementiert:

- Extended File System (EXT) und dessen Nachfolger EXT2 und EXT3 (genaue Erklärung im Kapitel 3.1.4.)
- XIA Filesystem (basiert auf dem Minix-Dateisystem und fügt nur einige Verbesserungen hinzu, wie eine größere Partitionsgröße, lange Dateinamen und Unterstützung für die drei Timestamps)

Um auch in Zukunft leicht neue Dateisysteme in den Linux-Kernel hinzufügen zu können, wurde ein Virtual File System Layer entwickelt. Dieser Layer wurde ursprünglich von Chris Provenzano geschrieben und später von Linus Torvalds neu implementiert und in den Linux-Kernel integriert.

Das Prinzip des Virtual File System Layers ist relativ einfach: der Layer wird benutzt sobald ein System Call auf eine Datei oder ein Verzeichnis ausgeführt wird, d.h. es wird vom Kernel eine Funktion des VFS aufgerufen. Das VFS arbeitet dann als Indirection Layer, der den Datei-orientierten System Call abarbeitet und die nötigen Funktionen des physikalischen Dateisystems aufruft, um die Input-Output-Operation durchzuführen. Der Dateisystem-Code benutzt einen Buffercache bevor er die gewünschte Operation am Device durchführt. Das Schema ist in der folgenden Abbildung veranschaulicht.

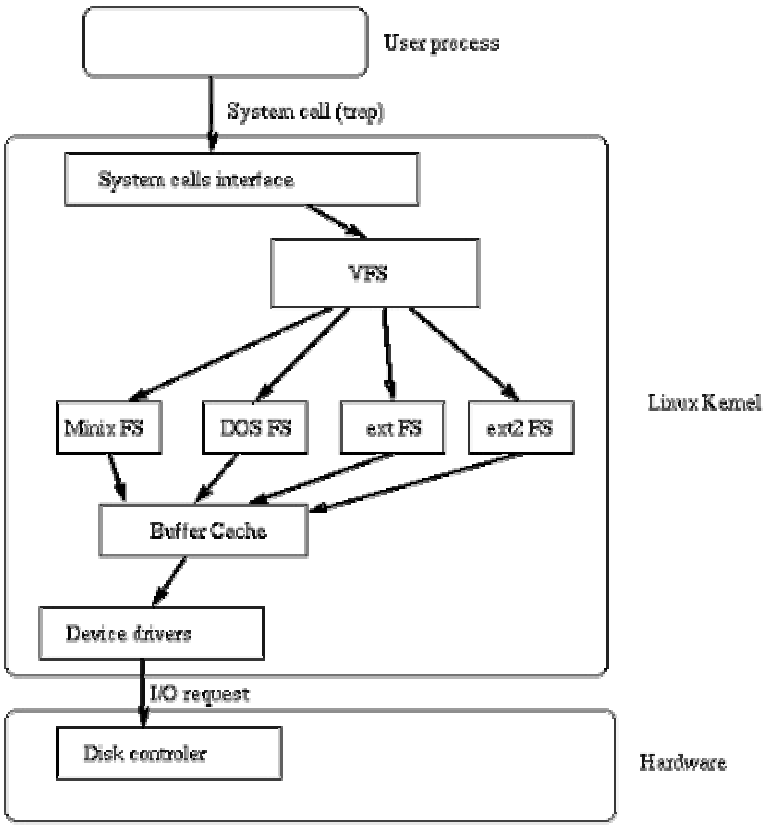

Abbildung 10: VFS-Schema [EXT2_Card]

Die Struktur des VFS definiert eine Grundmenge an Funktionen, die jedes Dateisystem implementieren muss. Dieses Interface assoziiert drei Arten von Objekte: Dateisysteme, Inodes und geöffnete Dateien.

Das VFS kennt die verschiedenen Dateisysteme aus einer Tabelle im Kernel. Jeder Eintrag dieser Tabelle beschreibt ein Dateisystem und beinhaltet den Namen des Dateisystems und einen Pointer zu einer Operation, die während des Mountens aufgerufen wird und den Superblock eines Speichermediums lesen soll, sowie interne Variablen initialisiert und einen Dateisystem-Deskriptor an das VFS zurückliefert. Der Deskriptor enthält eine Reihe von relevante Daten für das VFS:

- allgemeine Informationen die jedes Dateisystem enthält
- Pointer zu Funktionen, die der physikalische Dateisystem Kernel Code (physical filesystem kernel code) implementiert
- private Daten, die vom physikalischen Dateisystem Code gewartet werden

Nachdem das Dateisystem gemountet ist, können die VFS-Funktionen daher mit Hilfe dieser Beschreibung die internen Dateisystem-Routinen benutzen.

Auch für Inodes gibt es einen eigenen Deskriptor. Dieser enthält Informationen über geöffnete Dateien und Operationen, die vom physikalischen Dateisystem Code bereitgestellt werden. Diese Funktionen betreffen nur Operationen, die auf Dateien ausgeführt werden können, z.B.: erstellen oder unlink (löscht eine Datei).

Der dritte Deskriptor ist für geöffnete Dateien vorgesehen. Er enthält, wie der Inode-Deskriptor, Informationen über geöffnete Dateien und Operationen, die vom physikalischen Dateisystem Code bereitgestellt werden. In diesem Fall sind diese Operationen jene, die auf geöffnete Dateien ausgeführt werden können, z.B.: lesen oder schreiben.

Die Idee des VFS gibt es nicht nur in der Linux-Welt, zum Beispiel gibt es von Apache [Apache] eine hervorragende VFS-Bibliothek für JAVA, die sehr viele Dateisysteme unterstützt (siehe Kapitel 3.2.2). Trotz der Namensgleichheit und der gleichen Konzeptidee sind VFS für Linux und Apache VFS nicht vergleichbar, da die Anwendungsgebiete sehr unterschiedlich sind.
[EXT2_Card]

3.1.4. EXT2/3

Jedes Linux-Dateisystem implementiert eine Grundmenge allgemeiner Konzepte, die aus dem Betriebssystem Unix stammen: Dateien werden als Inodes repräsentiert, Verzeichnisse sind ebenfalls Dateien, die Listen von Datei-Einträgen enthalten, und Devices werden angesprochen indem man eine Input-Output-Operation auf eine spezifische Datei ausführt.

Wie eben erwähnt wird jede Datei durch eine Struktur, genannt Inode, repräsentiert, wobei in dieser folgende Informationen enthalten sind:

- der Typ der Datei
- Zugriffsrechte
- Besitzer der Datei
- Timestamps (Erstellungsdatum, letztes Änderungsdatum, Zeitpunkt des letzten Zugriffs)
- Größe

- Pointer zu den Datenblöcken der Datei (auch Pointer auf andere Datenblöcke sind möglich)

Die folgende Abbildung zeigt die Struktur eines Inodes.

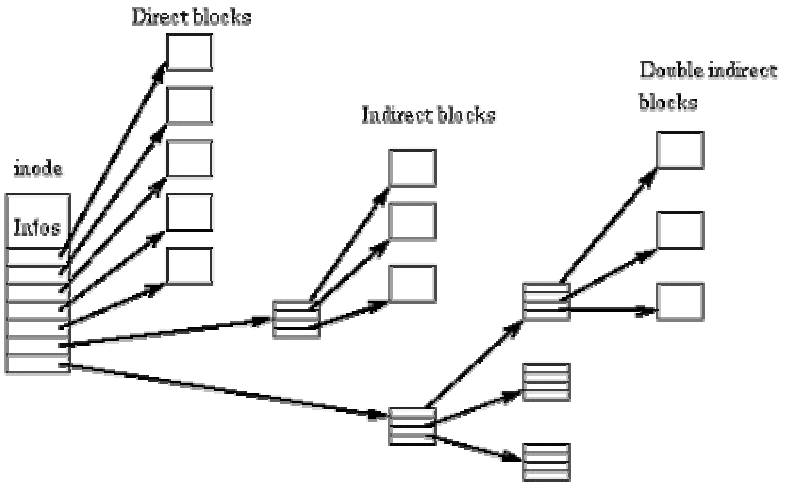

Abbildung 11: Struktur eines Inodes [EXT2_Card]

Verzeichnisse sind spezielle Dateien, die hierarchisch strukturiert sind, d.h. ein Verzeichnis kann keine, ein(e) oder mehrere Datei(en) und Unterverzeichnis(se) enthalten. Jeder Listeneintrag eines Verzeichnisses beinhaltet eine Inode-Nummer und den Dateinamen. Sobald ein Prozess den Pfadnamen benutzt, sucht der Kernel in den Verzeichnissen nach der dazugehörigen Inode-Nummer. Ist diese gefunden wird der Inode in den Speicher geladen.

Unix Dateisysteme unterstützen auch das Link-Konzept, d.h. mehrere Namen können einen Inode repräsentieren. Erstellt man einen Link wird ein Verzeichnis-Eintrag erstellt, der auf den angegeben Inode einen Pointer setzt. Weiters wird ein Link-Counter erhöht, um beim Löschen der Datei auch alle Links zu löschen. Diese Methode des Linkens wird „hard link“ genannt und unterschützt lediglich Dateien. Verzeichnisse können nicht gelinkt werden um das Problem eines Kreises (sobald man ein Eltern-Verzeichnis linkt tritt dieser Effekt auf) zu umgehen. Die meisten Unix-/Linux-Dateisysteme unterstützen auch „soft links“, auch „symbolic links“ genannt. Diese werden nicht automatisch gelöscht sobald die referenzierte Datei gelöscht wird und benötigen Speicherplatz, da es sich um eine Datei handelt.

Nachdem der VFS Layer im Linux-Kernel implementiert war, wurde im April 1992 das Dateisystem „Extended File System“ in Linux v0.96c integriert. Damit waren die Limitierungen des Mimix-Dateisystems behoben, Partitionen konnten 2 Gigabyte groß sein

und Dateinamen waren nun nur noch auf 255 Zeichen beschränkt. Trotz dieser Verbesserungen hatte EXT auch Probleme. Es unterstützt weder Mehrbenutzerzugriffe (separate access), noch die Modifikation von Inodes sowie die Modifikation der Timestaps von Dateien. EXT benutzt Link-Listen für die freien Blöcke und Inodes, was bei Benützung des Dateisystems zu einer unsortierten Liste und externer Fragmentierung führt.

Die logische Kensequenz daraus war eine Weiterentwicklung von EXT. Das Ziel war es, ein Dateisystem zu entwerfen und zu implementieren, das die gegebenen Probleme von EXT ausbessert. Weiters sollte es die Unix File Semantics implementieren, erweiterte Features bieten, eine exzellente Performance aufweisen und ein robustes Dateisystem sein, das das Risiko eines Datenverlustes minimiert. Das Ergebnis war das EXT2-Dateisystem (Ext2fs), welches auch ohne Neuformatierung das Hinzufügen von neuen Erweiterungen beinhaltet.

Die Eigenschaften von EXT2 können in zwei Kategorien aufgespaltet werden: Die „Standard"-EXT2 Features und die „Advanced"-EXT2 Features.

Die Standard-Eigenschaften:
Dies sind Eigenschaften, die alle UNIX-Dateisysteme bieten.

- EXT2 unterstützt alle Standard-UNIX-Filetypen: normale Dateien (regular files), Verzeichnisse (directories), Geräte-spezifische Dateien (device special files) und symbolische Links (symbolic links).
- EXT2 kann sehr große Partitionen erstellen und managen. Während der original Kernel-Code die maximale Partitionsgröße auf 2GB beschränkte, bietet EXT2 - auch durch Änderungen am VFS Layer - Partitionsgrößen bis 4TB. Dadurch ist es möglich große Festplatten zu verwenden, ohne viele Partitionen erstellen zu müssen.
- EXT2 unterstützt lange Dateinamen, da es variable Längen bei den Verzeichniseinträgen benutzt. Die maximale Länge eines Dateinamen beträgt 255 Zeichen, dieses Limit kann aber - falls benötigt - auf 1012 Zeichen vergrößert werden.
- EXT2 reserviert einige Blöcke für den Superuser (root). Im Standardfall werden 5% aller Blöcke reserviert. Dies ist für den Fall, wenn Benutzer die ganze Partition mit Daten angefüllen, damit immer noch genug freien Speicherplatz zur Verfügung steht und der Administrator das Speicherproblem lösen kann.

Die erweiterten Eigenschaften:

Über die normalen UNIX-Dateisystem-Features hinaus, bietet EXT2 noch eine Reihe an weiteren Eigenschaften:

- Mit Hilfe von Dateiattributen kann ein Benutzer das Kernel-Verhalten bei Dateioperationen beeinflussen. Diese Attribute können auf Dateien und Verzeichnisse gesetzt werden. Neu erstellte Dateien in solch einem Verzeichnis erben diese Attribute vom Vater-Verzeichnis.
- BSD oder System V Release 4 Semantics können beim Mounten ausgewählt werden. Damit kann der Administrator die Dateierstellungs-Semantik (file creation semantics) wählen.

  Wird das Dateisystem mit BSD-Semantik gemountet, werden Dateien mit derselben Gruppen-ID (group id) wie ihr Vater-Verzeichnis erstellt, d.h. neu angelegte Dateien und Verzeichnisse gehören nicht zu der Gruppe des anlegenden Prozesses sondern zur Gruppe des Vater-Verzeichnisses.

  Die System V-Sematik ist etwas komplexer: Hat ein Verzeichnis das setgid-Bit gesetzt, erben neu angelegte Dateien die Gruppen-ID des Verzeichnisses. Unterverzeichnisse erben die Gruppen-ID und das setgid-Bit, d.h. alle neu angelegten Dateien und Verzeichnisse gehören zu der Gruppe des Vater-Verzeichnisses.
- Es kann ein BSD-ähnlicher synchroner Update-Modus in EXT2 benutzt werden. Eine Mount-Option bietet dem Administrator die Möglichkeit, dass alle Metadaten (Inodes, Bitmap Block, Indirect Blocks, Directory Blocks) synchron auf die Platte geschrieben werden sobald sie sich ändern. Der Nutzen liegt darin, die Metadaten konsistent zu halten. Jedoch führt dies zu einer schlechten Performance. Diese EXT2-Eigenschaft wird nur wenig genützt, da sie weiters zu Fehlern bei den Benutzer-Daten führen kann.
- EXT2 erlaubt es dem Administrator, die logische Blockgröße des Dateisystems zu wählen, typische Größen sind 1024, 2048 und 4096 bytes. Eine höhere Blockgröße kann eine Beschleunigung bei I/O-Operationen bringen, da durch weniger I/O-Requests weniger Bewegungen des Plattenkopfes nötig sind um eine Datei zu lesen. Auf der anderen Seite verschwenden große Blöcke Speicherplatz, da sie diesen nicht optimal ausnützen können (beim letzten Block einer Datei) und so zu interner Fragmentierung führen.

  Die meisten Vorteile, die große Blockgrößen bringen, werden auch durch die EXT2 Preallocation Techniques geboten (an dieser Stelle sei auf die später folgenden Performance-Optimierungen verwiesen).

- EXT2 unterstützt schnelle symbolische Links (fast symbolic links). Solch ein Link benutzt keinen Datenblock des Dateisystems, sondern speichert den Zielnamen (target name) direkt im Inode. Diese Policy spart nicht nur Speicherplatz (es muss kein Datenblock allokiert werden) sondern bringt auch eine Beschleunigung bei Link-Operationen, da kein Datenblock mehr gelesen werden muss. Da der Platz für Links in Inodes begrenzt ist (auf 60 Zeichen), kann nicht jeder Link ein fast symbolic Link sein.
- EXT2 hat einen eingebauten Schutz um inkonsistente Zustände zu erkennen. Zu diesem Zweck führt es über den Dateisystem-Status Buch. Dies passiert in einem speziellen Feld im Superblock, das vom Kernel-Code geschrieben wird. Sobald ein Dateisystem in read/write-Modus gemountet wird, wird der Status auf „not clean" gesetzt. Wird es wieder freigegeben oder nur im read-only-Modus gemountet wird der Status auf „clean" zurückgesetzt. Beim nächsten Booten des Systems erkennt der Dateisystem-Checker (filesystem checker) aufgrund dieses Flags ob die Konsistenz des Dateisystems überprüft werden muss.

  Darüber hinaus werden in diesem Feld auch Fehler verzeichnet. Entdeckt der Kernel-Code eine Inkonsistenz wird das Feld auf „erroneous" gesetzt. Damit wird eine Überprüfung des Dateisystems beim nächsten Boot erzwungen, egal ob der Status auf „clean" oder „not clean" gesetzt ist.
- EXT2 bietet weiters zwei Methoden um Integritätschecks des Dateisystems in periodischen Abständen durchführen zu müssen. Im Superblock befindet sich ein sogenannter Mount-Counter, der bei jedem mounten des Dateisystems im read/write-Modus erhöht wird. Erreicht dieser Counter einen maximalen Wert, der ebenfalls im Superblock gespeichert ist, wird der Dateisystem-Checker gezwungen eine Überprüfung des Dateisystems durchzuführen. Weiters gibt es im Superblock noch zwei Felder, wo das letzte Überprüfungsdatum und die maximale Zeitspanne zwischen zwei Prüfungen gespeichert sind. Wird diese Zeitspanne überschritten wird auch ein Dateisystem-Check erzwungen. EXT2 bietet dem Administrator auch Tools um diese Werte seinen Bedürfnissen anzupassen. Zu erwähnen sei das Programm „tune2fs" womit das Fehlerverhalten (hier gibt es drei Möglichkeiten wie das System auf einen Fehler reagieren kann: 1) normal fortfahren, also den Fehler ignorieren; 2) das System im read-only-Modus remounten um Fehler im Dateisystem zu verhindern, d.h. der Benutzer kann keine Änderungen mehr speichern; 3) der Panik-Modus, indem der Kernel das System sofort neu startet um den Dateisystem-Checker auszuführen), der maximale Mount-Counter-Wert, die maximale Zeitspanne zwischen zwei

Überprüfungen und die Anzahl der logischen Blöcke, die für den Superuser reserviert sind, verändert werden können.

- EXT2 erlaubt es dem Benutzer ein Attribut zu setzen, das ein sicheres Löschen von Dateien fordert. Dies dient dazu, dass eine so geschützte Datei nicht mit Hilfe von Disk-Editoren wiederhergestellt werden kann. Dazu werden nach dem Löschen zufällige Daten über die Blöcke der gelöschten Datei geschrieben.
- EXT2 hat - inspiriert vom BSD-Dateisystem - zwei neuen Dateitypen integriert. Gekennzeichnet sind diese Dateien durch spezielle Flags:
  unveränderliche Dateien (immutable files), Flag „i“: Diese Dateien können nur gelesen werden, niemand kann sie schreiben oder löschen. Sie können als wichtige Konfigurationsdateien genutzt werden. Im Falle eines Verzeichnisses können Dateien, die in diesem Verzeichnis bereits existieren, verändert werden. Weder das Löschen noch das Anlegen von Dateien ist möglich.
  Dateien, an die nur angefügt werden darf (append-only files), Flag „a“: Diese Dateien können zwar geschrieben werden, jedoch werden die Daten nur am Ende der Datei angehängt. Sie können weder gelöscht noch umbenannt werden. Ein sinnvoller Einsatz für Append-only-Dateien sind Logdateien. Wird das Flag auf einem Verzeichnis angewendet, dürfen darin keine Dateien angelegt oder gelöscht werden.
  Um eine dieser Dateien wieder löschen zu können, muss ein Benutzer, der die nötigen Rechte besitzt (meist nur der Administrator), das Flag wieder umsetzen und so eine klassische Datei aus der Spezialdatei machen.

Die physikalische Struktur von EXT2 wurde stark vom Layout des BSD-Dateisystems beeinflusst, daher sind die EXT2-Blockgruppen analog zu den BSD FFS (Fast Filesystem) Zylindergruppen. Diese Blockgruppen haben nichts mit dem physischen Layout der Blöcke auf der Festplatte zu tun, da moderne Laufwerke für den sequentiellen Zugriff (sequential access) optimiert sind und ihre physikalische Geometrie vor dem Betriebssystem verstecken.
Die physikalische Struktur besteht also aus einem Bootsektor und einer Reihe von Blockgruppen.

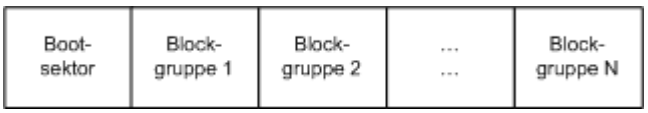

Abbildung 12: Struktur des Dateisystems bei EXT2 [EXT2_Card]

Jede dieser Blockgruppen enthält eine redundante Kopie von kritischen Dateisystemkontrollinformationen (Superblock und Dateisystem-Descriptor) und einen Teil des Dateisystems (eine Block-Bitmap, eine Inode-Bitmap, ein Stück der Inode-Tabelle und Datenblöcke).

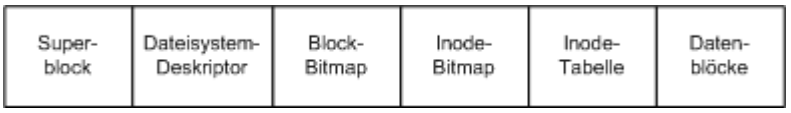

Abbildung 13: Struktur einer Blockgruppe bei EXT2

Die Ausfallsicherheit wird durch dieses Prinzip drastisch erhöht, da die Kontrollstrukturen in jeder Blockgruppe repliziert werden und so im Fehlerfall, dass der Superblock korrupt ist, dieser leicht wiederherzustellen ist. Weiters bringt diese Struktur eine bessere Performance, da der Lesekopf des Datenträgers zwischen Inode-Tabelle und Datenblöcken nicht so weit bewegt werden muss um I/O-Operationen auf Dateien auszuführen. Der Nachteil ist ein leicht erhöhter Speicherverbrauch.

Dateien in einem Verzeichnis werden in EXT2 als Linkliste von längenvariablen Einträgen gespeichert, wobei jeder Eintrag die Inode-Nummer, die Länge des Eintrags, die Länge des Namens und den Dateinamen enthält. Die längenvariablen Einträge haben den Vorteil, dass kein Speicherplatz verschwendet wird.

Als quasi-Nachfolger von EXT2 gilt EXT3, das als größte Neuerung Journaling unterstützt. Der Vorteil von Journaling-Filesystemen ist, dass nach einem Systemabsturz nicht die ganze Platte überprüft werden muss, sondern nur die letzten Transaktionen am Dateisystem rückgängig gemacht werden müssen.
EXT3 unterstützt drei Modi:

- writeback: Dies ist der schnellste Journaling-Modus. Hier wird nur geloggt was bei den Metadaten geändert wurde, es verlässt sich aber beim Schreiben der Daten einer Datei auf das Dateisystem.
- journal: Dies ist der langsamste Journaling-Modus. Es werden sowohl alle Änderungen der Daten des Dateisystem sowie der Metadaten geloggt.

- ordered: Dies ist der Standardmodus bei dem auch nur die Metadaten im Logbuch (=Journal) gespeichert werden. Jedoch werden hier die Daten zuerst auf der Platte geändert bevor die Metadaten geschrieben werden.

EXT3 hat den großen Vorteil gegenüber anderen Journaling-Filesystemen, dass es abwärtskompatibel zu EXT2 ist, d.h. im Notfall kann man von einer Rettungsdiskette/-CD starten und das vorhandene EXT3-Dateisystem mit EXT2 mounten. Auch kann man aus einem bestehenden EXT2-Dateisystem in wenigen Schritten ein EXT3-Dateisystem erstellen. Weiters sind alle EXT2-Tools auch auf das EXT3-Dateisystem anwendbar.
EXT3 ist auch Teil des Linux-Kernels und aufgrund seiner vielen Vorteile eines der derzeit beliebtesten Linux-Dateisysteme.
[EXT2_Card], [EXT3_Expl], [EXT3_Tweedie_00], [EXT_Setuid], [FS_Perf_Bryant]

3.1.5. Weitere Dateisysteme

Nachdem in den vorangegangen Kapitel einige Dateisysteme genauer beschrieben wurden, wird an dieser Stelle ein Überblick über die wichtigsten weiteren verfügbaren Dateisysteme gegeben, wobei diese in drei Kategorien eingeteilt werden.

1) Dateisysteme für Blockdevices (Festplatten, Disketten, CDs,…)

Klassische Linux-Dateisysteme:

- Minix: veraltet, aber aus Platzgründen teilweise noch auf Disketten zu finden.
- XIA: veraltet, wird kaum mehr verwendet.
- EXT: veraltet, wird kaum mehr verwendet, siehe Kapitel 3.1.4
- EXT2: siehe Kapitel 3.1.4
- UMSDOS: wurde benutzt um Linux auf einer MS-DOS-Partition zu installieren.

Journaling-Linux-Dateisysteme:

- EXT3: siehe Kapitel 3.1.4
- ReiserFS: erstes voll funktionsfähiges Journaling-Dateisystem für Linux; Journaling nur für Metadaten, d.h. für die Verzeichnisse und Verwaltungssektoren, nicht jedoch für die Nutzdaten in den Dateien selbst, dies kann jedoch durch einen Patch behoben werden, der bald auch in den Linux-Kernel einfließen wird.

- XFS: ist eine Portierung des XFS-Dateisystems von SGI IRIX, die Vorteile liegen in der hohen Geschwindigkeit, besonders im Umgang mit großen Dateien und der Unterstützung von Access Control Lists, Quotas und Resizing. Ist im Linux-Kernel 2.6 direkt enthalten.
- JFS: JFS steht zwar allgemein für Journaling-Filesystem, gemeint ist aber IBM's JFS, das eine Neuimplementierung des JFS von OS/2-Warp-Server sowie der IBM AIX darstellt; unterstützt auch Access Control Lists; ab Linux-Kernel 2.4.20 direkt enthalten.

Microsoft-Dateisysteme:

- FAT (FAT12, FAT16, FAT32, VFAT): siehe Kapitel 3.1.1
- NTFS/NTFS5: siehe Kapitel 3.1.2

OS/2-Dateisysteme:

- HPFS: High Performance File System, ab OS/2 v1.2, ähnlich zu FAT mit einigen Verbesserungen.

MacOS-Dateisysteme:

- HFS: Dateisystem für Apple Macintosh
- HFS+: neuere Variante des HFS-Dateisystems, das den Speicherplatz des Datenträgers durch kleinere Clustergrößen und mehr Dateien pro Volume besser nützt.
- UFS bzw. FFS: wird auch von SunOS und BSD genützt, MacOS X enthält es nur aus Kompatibilitätsgründen.

Kommerzielle Unix-Dateisysteme:

- VxFS
- Xenix
- System V-Dateisystem
- Coherent-Dateisystem

CD-Dateisysteme:

- ISO-9660-Dateisystem: Wird für CD-ROMs benützt, früher auch als High Sierra Filesystem bekannt.

- Joliet-Dateisystem: Erweiterung des ISO-9660-Dateisystem, das auch Unicode-Dateiennamen versteht; wurde von Microsoft entwickelt.
- UDF: CD im PacketWriting Format, um CDs wie eine Festplatte anzusprechen (mit allen Funktionen wie Lesen, Schreiben, Umbenennen, Löschen).

Amiga-Dateisysteme:

- AFFS: Dateisystem für Standard-Amiga-Dateisystempartitionen, jedoch nicht für Amiga-Disketten.

Acorn-Dateisysteme:

- ADFS: Dateisystem für Acorn-Partitionen.

2) Netzwerk-Dateisysteme

Klassische „exporte“ oder „shares“ (dienen zum Freigeben von Dateien und Verzeichnissen im lokalen Netzwerk):

- NFS: Network File System ist die übliche Art im UNIX-Bereich Verzeichnisse zu ex- und importieren um auf die Dateien und Verzeichnisse anderer Rechnern zugreifen zu können; NFS ist ein relativ unsicheres Protokoll (da unverschlüsselt) und soll daher nur im lokalen Netz eingesetzt werden.
- SMBFS: Das SMB-Protokoll (Server Message Block), auch als CIFS (Common Internet File System) bekannt, dient dazu in Microsoft Windows Dateien und Verzeichnisse freizugeben, Drucker im LAN (Local Area Network) gemeinsam zu nutzen und sich an einer Windows-Domäne anzumelden. Das SMBFS ist auch für Linux implementiert und heiß dort Samba. Dies ist ein Datei-, Druck- und Domain-Anmelde-Server unter Linux für die Anbindung von Windows-Clients. Es sind die wichtigsten Funktionen eines Windows NT-Servers nachgebildet. Ab Version 3.0 unterstützt Samba Active Directorys, kann an Windows-Domänen teilnehmen und Benutzer können sich über LDAP/Kerberos authentifizieren.
- NCPFS: NCP ist ein Protokoll für Novell-Fileserver um ein Dateisystem zu exportieren; es wird nicht TCP/IP verwendet sondern das Novell-eigene IPX/SPX.

Verteilte Dateisysteme (distributed Filesystems):

- InterMezzo: Ideal um seine Daten auf verschiedenen Rechnern zu bearbeiten, die nicht ständig miteinander verbunden sind. Um die Daten zu synchronisieren mountet man das Standarddateisystem über den Typ „intermezzo“, anstatt beispielsweise als „ext3“ oder „reiserfs“. Danach führt das InterMezzo-Kernelmodul Buch über alle Änderungen des Dateisystems. Die Protokolldateien werden von dem InterSync-Dämon ausgewertet und mit denen der anderen Rechner verglichen. Treten Unterschiede auf und besteht gerade eine Netzwerkverbindung, werden die Veränderungen synchronisiert.
- AFS: Andrew File System, ein sehr gutes Netzwerk-Dateisystem, das von IBM als freie Software (OpenAFS) zur Verfügung gestellt wird.
- Coda: Nachfolger von AFS mit vielen nützlichen Features (Network Bandwith Adaption, Security Modell für Authentifizierung, Verschlüsselung und Access Control, Server Replication,…).

3) Spezielle Dateisysteme für bestimmte Anwendungen

Kernel:

- PROCFS: Dient unter Linux zum Zugriff auf Kernel-Parameter und Status-Informationen; /proc ist ein virtuelles Dateisystem, d.h. ihm wird kein Speicherplatz zugeordnet. Es bietet Informationen über den Prozessor, Speicher, Interrupts, PCI-Busse und -Geräte, Systemlast, benutzte Geräte, verfügbare Dateisysteme, DMA-Kanäle, gemountete Laufwerke, Rechnerlaufzeit,…

Gerätetreiber:

- DEVFS: Zugriff auf Gerätetreiber

Embedded Systems:

- JFFS: Journaling Flash File System, ein Dateisystem für Flash-Speicherkarten.
- JFFS2: Ein log-structured Dateisystem, das für Flash-Devices in Embedded Systems ausgelegt ist.

Virtuelle Dateisysteme:

- AVFS: A Virtual Filesystem, virtuelles Dateisystem zum Zugriff auf Archivdateien (gzip, tar, zip,..) und entfernte Dateisysteme (via ftp, http, WebDAV – siehe Kapitel 4.,…).
- CVSFS: virtuelles Dateisystem zur Abbildung von CVS-Repositories, das CVS-Projekt wird direkt in ein Verzeichnis eingeblendet.
- LUFS: Dateisystem außerhalb des Kernels (Userspace), um entfernte Dateisysteme über verschiedene Wege (ftp, ssh,…) anzusprechen und die entfernten Dateien lesen und bearbeiten zu können als wären sie lokal.

Speicher (auch im virtuellen):

- TMPFS bzw. SHMFS: Linux-Dateisystem, das in einer Art dynamischen RamDisk (virtuelles Laufwerk im Arbeitsspeicher) liegt. Vorteile: gegenüber einer normalen RamDisk wächst oder schrumpft die Größe des belegten Speichers; ein schnelles Dateisystem, das sich für temp-Dateien eignet; die Daten brauchen nicht entsorgt werden; Nutzung: temporäre Dateien oder speichern von Session-Daten wie zum Beispiel bei PHP.

Read-Only-Dateisysteme:

- ROM-Dateisystem: ein sehr kleines Dateisystem, das keine Schreibvorgänge unterstützt und hauptsächlich bei RAM-Disks für die Systemkonfiguration, beim Booten oder für EPROMS gedacht ist.
- Squashfs: komprimiertes read-only-Dateisystem (Daten, Inodes und Verzeichnisse werden komprimiert), gedacht für Archive aber auch für abhängige Block-Device/Memory Syteme (z.B. Embedded Systems) wo ein geringer Overhead benötigt wird.

Cluster Dateisysteme:

- OpenGFS: nützlich in SANs (Storage Area Networks), kann mit iSCSI, HyperSCSI und Fireware benutzt werden.
- OCFS: Oracle Cluster File System
- Lustre: Lustre steht für Linux + Cluster und ist für sehr große Cluster entworfen, Ziel ist es eine Cluster-Dateisystem für 10.000e Nodes mit Petabytes von Speicher zu entwickeln.

[LinuxWiki.org], [Linux_Wegweiser_00],[Coda_FS], [HFS+], [HPFS], [JFFS], [JFFS2], [LUFS], [Lustre], [OCFS], [SQUASHFS], [OpenGFS]

3.2. Dateisystem-Modelle in Java

Da Java eine plattformübergreifende Programmiersprache ist, muss es Ansätze geben, mit verschiedenen Dateisystemen zurechtzukommen. Sun liefert mit java.io.File eine gute Bibliothek um Operationen auf Dateien und Verzeichnisse durchzuführen. Weit über diesen Ansatz hinaus geht Apache mit seiner vfs-Bibliothek (Virutal File System). Die beiden Bibliotheken werden in den kommenden Kapiteln behandelt.

3.2.1. Das FS-Modell von Java

Die Klasse „File“ ist eine abstrakte Repräsentation einer Datei oder eines Verzeichnisses. Da User-Interfaces und Betriebssysteme systemspezifische Pfadnamen verwenden um die Pfade zu Dateien und Verzeichnissen anzugeben, bietet die File-Klasse eine abstrakte, System-unabhängige Sicht von hierarchischen Pfadnamen.
Ein abstrakter Pfadname besteht aus zwei Komponenten:

- ein optionaler, System-abhängiger Prefix-String (wie ein Laufwerks-Specifier), „/“ für das UNIX-Root-Verzeichnis oder „\\“ für einen Microsoft Windows UNC Pfadnamen
- eine Sequenz von keinem oder mehreren Stringnamen

Wobei alle diese Strings Verzeichnisse sein müssen, bis auf den letzten. Dieser kann eine Datei- oder ein Verzeichnisname sein.

Da verschiedene Betriebssysteme verschiedene Fileseparatoren haben, werden in der File-Klasse Informationen aus dem System-Properties übernommen (file.separator). Mit dieser Information kann dann der Pfadname aufgebaut werden. Prinzipiell gibt es zwei Arten wie die File-Klasse Pfadnamen zur Verfügung stellt:

- relativer Pfad: Standardmäßig wird der relative Pfad, also jener Pfad relativ zum aktuellen Arbeitsverzeichnis (workingdirectory), ausgegeben. Dieses bekommt Java wieder aus den System-Properties (user.dir).

- absoluter Pfad: der gesamte Pfad vom Root-Verzeichnis an, keine weiteren Informationen sind mehr nötig.

Das Prefix-Konzept wird benutzt um Root-Verzeichnisse auf UNIX-Systemen sowie Drive-Specifier, Root-Verzeichnisse und UNC-Pfadnamen in Windows-Systemen darzustellen.

- Bei UNIX ist der Prefix für absolute Pfade „/“, relative haben keinen Prefix. Der abstrakte Pfad für das Root-Verzeichnis ist „/“.
- Bei Windows wird als Prefix zuerst der Laufwerksbuchstabe angegeben, gefolgt von einem „:“ und einem „\“ falls der Pfadname absolut ist. Der Prefix für UNC Pfadnamen ist „\\“, der Hostname und der Name des Shares sind die zwei ersten Namen in der Sequenz. Ein relativer Pfad, der kein Laufwerk spezifiziert hat keinen Prefix.

Instanzen der File-Klasse sind nach der Erstellung unveränderlich, d.h. der Pfadname der durch das File-Objekt repräsentiert wird, kann sich nicht mehr ändern.
[java.io.File]

3.2.2. Das VFS-Modell von Apache

Die Commons Virtual File System Bibliothek (org.apache.commons.vfs [Commons_VFS]) des Apache Jakarta Projects [Jakarta] bietet ein API (Schnittstelle) für den Zugriff auf eine große Anzahl von Dateisystemen. Diese Bibliothek ist in Java implementiert und frei verfügbar. Sie kommt auch im WebDAV-Explorer zum Einsatz.

Die wichtigsten Funktionen des Common VFS sind:

- ein einziges, konsistentes API für das Ansprechen von Dateien auf verschiedenen Dateisystemen
- unterstützt Caching von Datei-Informationen; diese werden in der JVM gehalten und können auch optional vom Dateisystem abgefragt werden.
- Event delivery: Wird durch eine Operation ein Event ausgelöst, wird dieses den Listeners übermittelt.
- unterstützt auch logische Dateisysteme, die auf mehrere Dateisysteme aufbauen

- Utilitys zum Integrieren der Common VFS in Applikationen (z.B.: VFS-aware ClassLoader und URLStreamHandlerFactory)
- Ein Set von VFS-enabled Ant-Tasks [ANT] (kopieren, löschen, verschieben, Verzeichnisse erstellen, synchronisieren)

Unterstütze Dateisysteme:

- Lokale Dateien: bietet den Zugriff auf das lokale, physische Dateisystem über einen absoluten Pfad
  Zugriff: [file://]absolute-path (wobei „absolute-path“ einen gültiger Dateipfad am lokalen Dateisystem darstellt)
  Beispiel „file:///home/usr/dir“ oder „c:\dir1\dir2“
- CIFS: Zugriff auf Dateien eines CIFS Server (Windows Shares, Samba-Server,…)
  Zugriff: smb://[username[:password]@]hostname[:port][absolute-path]
- FTP : Zugriff auf Dateien eines FTP-Servers
  Zugriff: ftp://[username[:password]@]hostname[:port][absolute-path]
- SFTP: Zugriff auf einen sicheren FTP-Server (SSH- oder SCP-Server)
  Zugriff: sftp://[username[:password]@]hostname[:port][absolute-path]
- HTTP und HTTPS: Zugriff auf Dateien auf einem HTTP-Server
  Zugriff: http[s]://[username[:password]@]hostname[:port][absolute-path]
- Temporäre Dateien: Zugriff auf ein temporäres Dateisystem oder Zwischenspeicher (scratchpad), das nach dem Beenden von Common VFS gelöscht wird. Das temporäre Dateisystem befindet sich am lokalen Dateisystem.
  Zugriff: tmp://[absolute-path]
- WebDAV: Zugriff auf Daten eines WebDAV-Servers
  Zugriff: webdav://[username[:password]@]hostname[:port][absolute-path]
- ZIP und JAR: nur Lesezugriffe sind verfügbar, falls eine Zipdatei eine andere unterstützte Datei enthält, müssen alle „!“ des äußeren Zipdatei-Pfades durch „%21“ ersetzt werden.
  Zugriff: (zip | jar)://zip-file-uri[!absolute-path]
  Beispiel: „jar:../lib/classes.jar!/META-INF/manifest.mf“ oder
  „jar:zip:http://somehost/dl/outer.zip%21/nested.jar!/somedir“

Geplante Erweiterungen:

- Verbesserte Dokumentation

- Unterstützung für mehr Dateisysteme (nfs, tar, gzip, bzip2, rsnyc, vcs, imap, jdbc filesystem, xml filesystem, ClassLoader resources,…)
- Erweiterung der vorhanden Dateisysteme (um zusätzliche Funktionen, aber auch Bugfixes,…)
- Verbessertes In-Memory-Caching

Diese umfangreiche Bibliothek ist eine enorme Erleichterung für den Programmierer, da dieser nicht auf die Dateisystem-Spezifika eingehen muss.
[Commons_VFS]

3.3. Gemeinsamkeiten und Vergleich von Dateisystemen

Das Auffinden der Gemeinsamkeiten von Dateisystemen ist die Grundlage für Konzepte, wie sie im vorangegangen Kapitel beschrieben wurden, trotzdem muss man auch auf Performanceeigenschaften der jeweilige Dateisysteme achten (z.B.: lokales Dateisystem vs. WebDAV-Server mit einer langsamen Verbindung).

Gemeinsamkeiten und Vergleich von lokalen Dateisystemen:

- Linux-Dateisysteme:

Kriterien	EXT2/3	ReiserFS	JFS	XFS
Betriebssystem	Linux	Linux (Kernel 2.4.1)	Linux (Kernel 2.4.20)	Linux (Kernel 2.6)
Limitierungen				
Max Datenträger Größe	4TB	17.6TB	512TB-4PB (Blockgrößen-anhängig)	18 Millionen TB
Max Anzahl Datei pro Datenträger	unlimitiert	unlimitiert	unlimitiert	unlimitiert
Max Dateigröße	2GB	17.6TB	kein Limit	9 Millionen TB
Max Länge eines Dateinamen	Standard - 255 Extended - 1012	Bis 255	Bis 255	Bis 255

Dateisystem-Features				
Unicode Datei Namen	Unicode Character Set	Unicode Character Set	Unicode Character Set	Unicode Character Set
Dateiattribute	Standard und benutzerdef.	Standard und benutzerdef.	Standard und benutzerdef.	Standard und benutzerdef.
Kompression	Ja	Ja	Ja	Ja
Verschlüsselung	Ja	Ja	Ja	Ja
Object Permissions	Ja	Ja	Ja	Ja
Disk Quotas	Ja	Nein	Ja	Ja
Sparse Files	Ja	Ja	Ja	Ja
Volume Mount Points	Ja	Ja	Ja	Ja
Performance				
Built-In Security	Nein	Nein	Ja	Ja
Wiederherstellbar	Ja	Ja	Ja	Ja
Performance	Durchschn.	Durchschn. Schnelles Löschen vieler Dateien	Durchschn. Schneller Umgang mit großen Dateien	Durchschn.
Speicherplatz-Economy	Max	Max	Max	Max
Fehlertoleranz	Max	Max	Max	Max

Abbildung 14: Linux-Dateisysteme

- Windows-Dateisysteme:

Kriterien	NTFS5	NTFS	FAT32	FAT16
Betriebssystem	Windows 2000 Windows XP	Windows NT Windows 2000 Windows XP	Windows 98 Windows ME Windows 2000 Windows XP	DOS Alle Versionen von Microsoft Windows

Limitierungen				
Max Datenträger Größe	2TB	2TB	2TB	2GB
Max Anzahl Datei pro Datenträger	unlimitiert	unlimitiert	unlimitiert	~65000
Max Dateigröße	kein Limit	kein Limit	4GB	2GB
Max Anzahl Cluster	unlimitiert	unlimitiert	268435456	65535
Max Länge eines Dateinamen	255	255	255	Standard - 8.3 Extended - 255
Dateisystem-Features				
Unicode Datei Namen	Unicode Character Set	Unicode Character Set	System Character Set	System Character Set
System Records Mirror	MFT Mirror Datei	MFT Mirror Datei	Zweite Kopie der FAT	Zweite Kopie der FAT
Boot Sector Location	Erster und letzter Sektor	Erster und letzter Sektor	Erster Sektor	Erster Sektor
Dateiattribute	Standard und benutzerdef.	Standard und benutzerdef.	Standard Set	Standard Set
Alternate Streams	Ja	Ja	Nein	Nein
Kompression	Ja	Ja	Nein	Nein
Verschlüsselung	Ja	Nein	Nein	Nein
Object Permissions	Ja	Ja	Nein	Nein
Disk Quotas	Ja	Nein	Nein	Nein
Sparse Files	Ja	Nein	Nein	Nein
Reparse Points	Ja	Nein	Nein	Nein
Volume Mount Points	Ja	Nein	Nein	Nein
Performance				
Built-In Security	Ja	Ja	Nein	Nein
Wiederherstellbar	Ja	Ja	Nein	Nein
Performance	Niedrig bei kleinen Datenträgern, hoch bei großen	Niedrig bei kleinen Datenträgern, hoch bei großen	Hoch bei kleinen Datenträgern, niedrig bei großen	Sehr hoch bei kleinen Datenträgern, niedrig bei großen

Speicherplatz-Economy	Max	Max	Durchschn.	Minimal bei großen Datenträgern
Fehlertoleranz	Max	Max	Minimal	Durchschn.

Abbildung 15: Windows-Dateisysteme [NTFS.com]

Gemeinsamkeiten von entfernten Dateisystemen:

- Performance: Hier kommt es lediglich auf die Geschwindigkeit des Internet-anschlusses beider Seiten (Server und Client) an. Verschlüsselte Verbindungen benötigen etwas mehr Zeit.
- Sicherheit: Viele entfernte Dateisysteme unterstützen Autorisation, Access Control und Verschlüsselung. Jene die dies nicht bieten, sollten nur in sicheren Umgebungen verwendet werden. Das Problem bei verteilten Dateisystemen, die von UNIX- und Windows-Clients genutzt werden, ist, dass zwei verschiedene Authentifizierungs-verfahren verwendet werden, d.h. der Administrator muss jeden Benutzer-Account doppelt anlegen (SAMBA löst dieses Problem teilweise).
- Geschwindigkeit: Mittels Client-seitigem Caching kann die Geschwindigkeit erhöht werden. Dabei muss aber auf die Cachekonistenz geachtet werden. Bei entfernten Dateisystemen kommen die verschiedensten Caching-Methoden vor, bei SMB/CIFS kommt zum Beispiel Read-Ahead-Caching und Write-Behind-Caching zum Einsatz, NFS verwendet ein Timestamp-Verfahren.
- Namensauflösung: Hierfür werden bei den entfernten Dateisystemen die verschiedensten Methoden verwendet. SMB/CIFS hat zwei Arten Namen aufzulösen (mittels Broadcast-Anfrage oder NetBIOS). NFS verwendet intern RPC-Funktionen sowie den normalen DNS Dienst zur Namensauflösung.

[LinuxWiki.org], [NTFS.com], [XFS], [XFS_for_Irix], [Inode], [JFS], [Linux_FS_Vergleich]

4. WebDAV

WebDAV steht für Web-based Distributed Authoring and Versioning und wird in den RFCs 2518 [RFC_2518] (HTTP Extensions for Distributed Authoring -- WebDAV aus dem Jahr 1999) und 3253 [RFC_3253] (Versioning Extensions to WebDAV aus dem Jahr 2002) spezifiziert. Es ist ein noch junges Protokoll mit großem Potential in der nahen Zukunft. Einfach gesprochen ist WebDAV eine Erweiterung des HTTP-Protokolls. Diese Erweiterung erlaubt es nicht nur Dateien von einem Webserver zu laden, sondern auch in die umgekehrte Richtung auf dem Server zu speichern.
Bisher war die Hauptaufgabe eines Webservers auf Anfragen die gewünschten Dokumente zu liefern. Mittels WebDAV ist es möglich, auf Dateien und Verzeichnisse direkt am Server Operationen auszuführen. Vor allem durch die Einbindung in alle modernen Betriebssysteme sowie Programmen, die älteren Betriebssystemen WebDAV-Unterstützung bieten, wird sich dieses Protokoll durchsetzen.
HTTP-Unterstützung befindet sich auf so gut wie allen Systemen, d.h. die notwendige Infrastruktur ist bereits vorhanden. Für den HTTP-Transfer gibt es bereits ausgefeilte Caching-Verfahren, sowie bei Bedarf transparente Verschlüsselung mittels SSL/TLS und Zugriffsschutz. Auch HTTP wird weiterentwickelt und ist zurzeit als Version 1.1 (RFC 2616 [RFC_2616]: Hypertext Transfer Protokoll -- HTTP/1.1) verfügbar.
[DAV_FAQ], [Slide]

4.1. Die Ziele von WebDAV

Das Hauptziel der WebDAV Working Group war die Entwicklung eines Protokolls das zum HTTP-Protokoll jene Erweiterungen hinzufügt, um es als verteiltes Web Entwicklungswerkzeug verwenden zu können, das völlig kompatibel zu HTTP ist.
Viele Entwickler sahen aber bald ein viel größeres Potential für WebDAV als nur ein simples Webpage Authoring Tool. Das Ziel einiger Entwickler war es, WebDAV als verteiltes Dateisystem für das Internet zu entwickeln. Andere sahen WebDAV als Protokoll zum Manipulieren der Inhalte von Document Management Systemen über das Web. Ein weiteres Ziel war es Groupwork zu unterstützen, da es als erstes Protokoll eine große Menge an kollaborativen Applikationen unterstützte. Weiters sollte WebDAV den Erfolg des HTTP-

Protokolls als Standard-Access-Layer für eine Reihe von Speicher-Repositories positiv beeinflussen.
Zwar gehen die Ziele der einzelnen Entwickler in ganz verschiedene Richtungen, trotzdem bietet WebDAV alle diese Eigenschaften.
[WebDAV_Wiki]

4.2. Die Eigenschaften und Funktionen von WebDAV

WebDAV fügt Eigenschaften und Collections zum HTTP Data Model hinzu und bietet zahlreiche Möglichkeit für:

- Eigenschaften: list, add, remove
- Namespace Operationen: move, copy
- Schreibschutz: lock, unlock
- Collections: mkcol, hierarchy operations

Es gibt mehrere Workinggroups, die an WebDAV arbeiten, deren Aufgabengebiete im Folgenden kurz aufgelistet werden.
[WebDAV_Intro_Pr]

4.2.1. Collaboration Infrastructure

Diese Infrastruktur eignet sich zur Entwicklung von asynchronen, verteilten, „hypertext-awaren", kollaborativen Editing Tools. Anwendungsgebiete sind zum Beispiel das gemeinsame Erstellen von Webseiten mit dazugehörigen Bildern oder das Arbeiten an einem beliebigen Mediatyp (Textverarbeitung, Präsentation) mit externen Programmen. Dazu wird das Sperren (lock/unlock) von Ressourcen benötigt.
Status: Standardisiert im RFC 2518 [RFC_2518]
[WebDAV_Intro_Pr]

4.2.2. Metadata Recording Infrastructure

Mit Hilfe dieser Infrastruktur können Metadaten von Webdaten bearbeitet werden. Mittels WebDAV können Eigenschaften (Properties) einer Web-Ressource als Name-Value-Tupel erzeugt, modifiziert, gelöscht und gelesen werden.

Die Konsistenz der Eigenschaften kann entweder von Server oder vom Client gewartet werden.
Sämtliche Eigenschaften sind im XML-Format verfügbar.
Status: Standardisiert im RFC 2518 [RFC_2518]
[WebDAV_Intro_Pr]

4.2.3. Namespace Management Infrastructure

Diese Infrastruktur bietet das entfernte Erstellen von Collections (Container, der Ressourcen und weitere Collections enthalten kann). Auf Collections und einzelne Ressourcen können Namespace-Operationen ausgeführt werden. Beispiele hierfür sind das Kopieren oder Verschieben von einzelnen Ressourcen, aber auch von Hierarchien von Ressourcen. Weiters kann man (geordnete) Collections von Ressourcen erstellen oder modifizieren und Ressourcen bei einem Referenzaufruf hinzufügen oder entfernen.
Status: Standardisiert im RFC 2518 [RFC_2518]
[WebDAV_Intro_Pr]

4.2.4. Versioning Infrastructure (Delta V)

Dies ist einer der Hauptpunkte von WebDAV, der Remote Versioning von Web-Ressourcen erlaubt. Dabei werden folgende Funktionen geboten: check-out, check-in (inklusive Kommentare), Version Graph History, durchsuchen von alten Versionen, automatische Versionierung für Clients, die kein Versioning unterstützen, und einfache sowie high-level Konfigurationsoperationen.
Status: Standardisiert im RFC 3253 [RFC_3253]
[WebDAV_Intro_Pr]

4.2.5. Access Control Infrastructure

Dient zur entfernten Erstellung von kollaborativen Gruppen und um zu kontrollieren, wer welche Ressourcen lesen oder schreiben darf. Das Schwierigste hierbei ist einerseits die Access Control Fähigkeiten des Repositorys herauszufinden und andererseits das clientseitige Userinterface so einfach wie möglich zu halten.
Status: Draft (draft-ietf-webdav-acl-13), Stand Dezember 2003 [WebDAV_ACL_Draft]
[WebDAV_Intro_Pr]

4.2.6. Searching Infrastructure (DASL)

Diese Infrastruktur bietet entferntes Suchen (remote searching). Gesucht kann nach Ressourcen werden, von denen eine Eigenschaft oder ein Wert bekannt ist. Weiters kann auch nach einem String in einer Ressource gesucht werden. Der Bereich der Suche kann beschränkt werden, entweder auf einen ganzen Server, eine Hierarchie von Ressourcen, eine Collection von Ressourcen oder eine einzelne Ressource.
Status: Draft (draft-reschke-webdav-search-latest), Stand November 2003 [DASL_Draft] [WebDAV_Intro_Pr]

4.3. WebDAV Applikationen

Ältere Betriebssysteme wie Windows 95/98 oder Mac OS 9, die noch keine WebDAV-Unterstützung mitbringen, können durch Applikationen WebDAV-Server nutzen.
In diesem Kapitel werden zwei solche Applikationen vorgestellt. Einerseits das frei verfügbare „Goliath“ für Mac OS und eine kommerzielle Variante „WebDrive“ für Microsoft Windows.

4.3.1. Goliath

Goliath wurde für Mac OS entwickelt und ist zurzeit als Version 1.0 verfügbar. Das Tool ist gedacht zum Erstellen und Editieren von Webseiten und nutzt dabei WebDAV um die Änderungen am WebServer zu speichern. Es war die erste Anwendung, die das WebDAV-Protokoll für den Macintosh implementierte.

Funktionalität:

- Verbinden zu einem WebDAV-Server, auch über einen Proxy

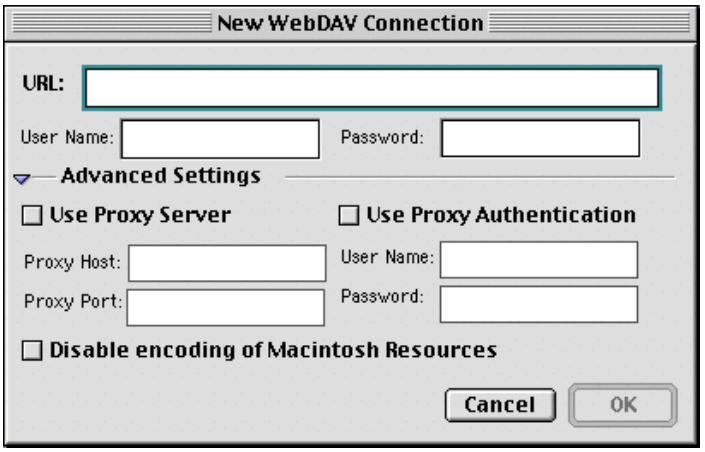

Abbildung 16: Verbindungs-Dialog [Goliath]

- Anzeigen des Inhalts eines Verzeichnisses wie im Windows-Explorer (mit Größe,…)

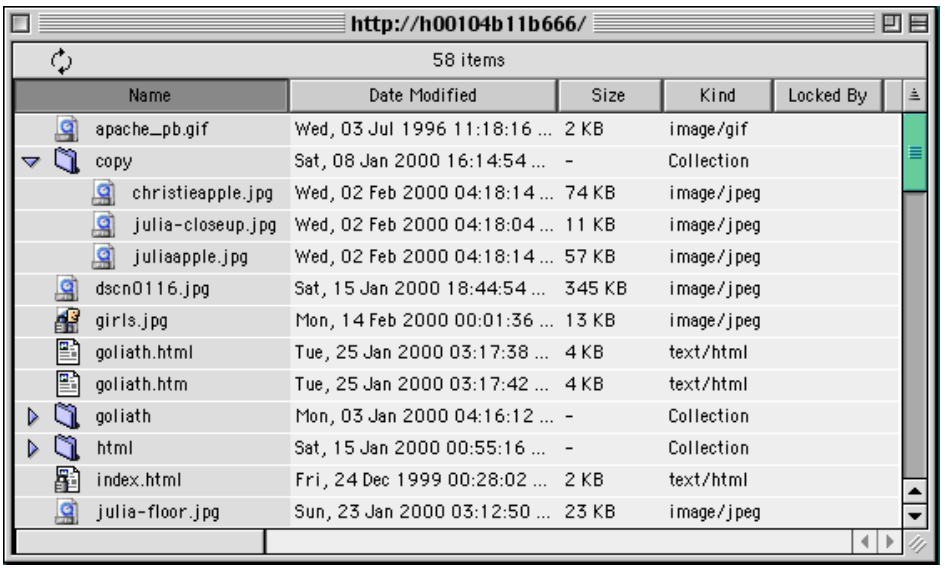

Abbildung 17: Hauptfenster [Goliath]

- Upload von Dateien auf die Webseite
- Neue Verzeichnisse erstellen
- Löschen von Dateien und Verzeichnissen
- Umbenennen von Dateien und Verzeichnissen
- Herunterladen von Dateien mittels Drag & Drop

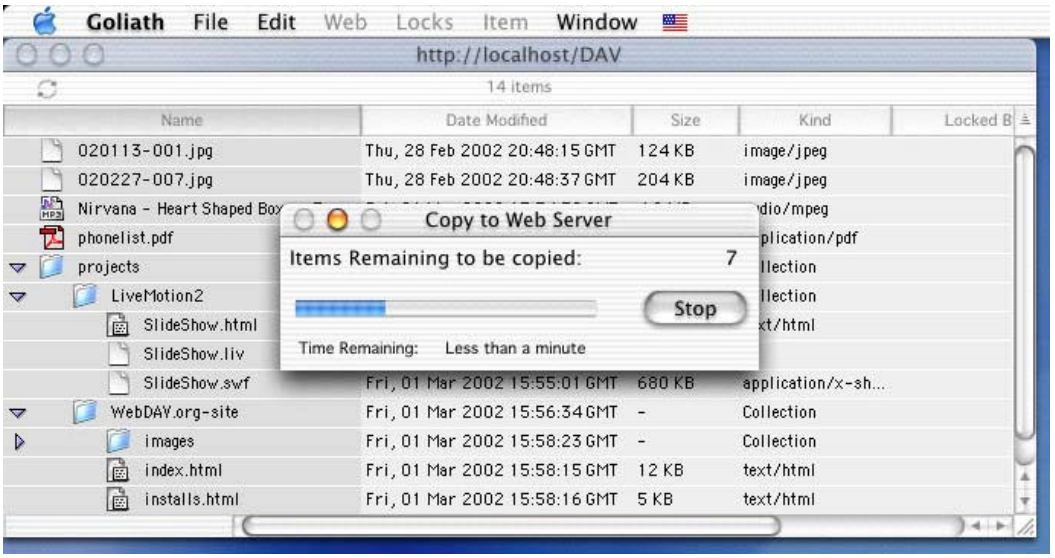

Abbildung 18: Dateidownload unter Mac OS X [Goliath]

- Bookmarks von WebDAV-Ressourcen für eine leichte Navigation
- Sperren von Dateien auf dem Web-Server um diese zu ändern (mit einer beliebigen anderen Applikation)
- Anzeigen und ändern von Properties
- Duplizieren von Items am Web-Server

Geplante Funktionalität (jedoch ohne konkrete Zeitangabe):

- Kopieren und verschieben von Dateien und Verzeichnissen
- WebDAV Searching (DASL)
- AppleEvents scriptable
- Zukünftige WebDAV-Entwicklungen implementieren (z.B.: Delta V, ACL-Unterstützung)

[Goliath]

4.3.2. WebDrive

WebDrive integriert einen WebDAV, HTTP oder FTP Server in den Windows-Desktop indem er ihn als Laufwerk abbildet. Dadurch wird ermöglicht, dass der Benutzer mit jeder beliebigen Applikation auf die am Server gespeicherten Dateien zugreifen und diese ändern kann.

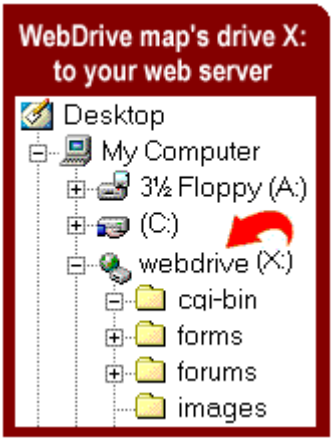

Abbildung 19: WebDrive im Windows-Explorer [WebDrive]

Mit Hilfe des WebDrive Site Managers kann man auswählen, welchen Server man mit welchem Laufwerksbuchstaben verbinden will. Es ist ebenfalls möglich Unterverzeichnisse von einem Server zu mappen, nicht nur dessen root-Verzeichnis.
Bei WebDAV-Servern können Dateien gesperrt werden, bevor Änderungen vorgenommen werden. Dadurch wird die Datenintegrität gewahrt (bei http oder ftp ist dies nicht möglich).
Im Bereich Sicherheit wird SSL-Verschlüsselung geboten.
[WebDrive]

4.4. WebDAV Server

Viele Webserver haben in den letzen Versionen eine WebDAV-Unterstützung erhalten, was auch zur großen Verbreitung und Akzeptanz des neuen Protokolls beiträgt. In den folgenden Kapiteln werden vier verschiedene WebDAV-Server vorgestellt.

- Slide vom Apache Jakarta Projekt: Frei verfügbarer WebDAV-Server, der auf Tomcat 4 aufbaut
- Zope: Open Source Web Application Server
- Mircosoft IIS: Der Web-Server von Mircosoft Windows
- Apple iDisk: Das Konkurrenzprodukt von Apple

4.4.1. Slide

Slide ist frei verfügbare Software aus dem Apache Jakarta Project und bietet einen WebDAV-Aufsatz auf Tomcat 4. Laut Ankündung auf der Homepage steht in Kürze Version 2.0 zur Verfügung.
Die Installation gestaltet sich zu Beginn ein bisschen kompliziert, da Slide die Möglichkeit bietet, verschiedene Repositories zum Speichern der Dateien und Verzeichnisse zu verwenden. Zurzeit werden als Repositories Datenbanken und Dateisysteme unterstützt.
Das Anlegen neuer Benutzer erfolgt über die Modifikation einer xml-Datei.

Slide bietet drei verschiedene Sichten auf die Daten:

- Client-Sicht: Die Funktionen beschränken sich auf das Ausführen von Web Applikationen und das Suchen nach Informationen.
  Bei der Standardkonfiguration von Slide kann die Client-Sicht über den Port 8080 erreicht werden.
  Im Prinzip bietet die Client-Sicht dieselbe Funktionalität wie ein Standard-Tomcat-Server, mit der Einschränkung, dass das Konzept des Virtual Hostings auf Grund von Designüberlegungen in Slide nicht vorhanden ist.
  Für jeden Namespace der in der Slide-Domäne definiert wird, legt Tomcat eine separate Web Applikation im Servlet-Container an. Sobald zum Beispiel ein „foo“-Namespace in Slide definiert wird, erstellt Tomcat einen „foo“-Kontext. Dem Kontext wird ein Bereich (realm) zugewiesen. Weiters wird die Authentisierung gegenüber den Principals aus dem Slide-Namespace überprüft.
- Editor-Sicht: Der Benutzer kann Web Applikationen ändern und konfigurieren sowie Informationen ändern. Sie kann bei einer Standardinstallation über den Port 8081 erreicht werden.
  Aus Sicherheitsgründen gibt es in der Standardkonfiguration keinen Benutzer, der sich in dieser View anmelden kann. Die Berechtigung muss zuerst über einen Eintrag in der Domain.xml vergeben werden. Um die Sicherheit weiter zu erhöhen sollte der Port, auf dem der Host läuft, vom Internet aus nicht erreichbar sein (z.B.: mit einer Firewall sperren).
  Wie bei der Client-Sicht wird jeder Slide-Namespace mit einem Kontext assoziiert, der vom Slide WebDAV-Servlet verwaltet wird. Jeder beliebige WebDAV-Client kann benutzt werden um den Inhalt dieses Kontexts zu ändern. Die Authentisierung funktioniert wie in der Client-Sicht.

Alle Ressourcen in der Web Applikation können gelesen und geändert werden, sofern die Benutzerrechte dies erlauben.

Da Slide im Hintergrund läuft werden alle Services, die Slide zur Verfügung stellt, benutzt, um die Web Applikation zu verwalten. Das beinhaltet ein ACL System, Versioning, Locking uvm.

- Administrator-Sicht: Der Administrator kann seine Benutzer verwalten, Rechte vergeben, Rollen zuteilen und den Server administrieren. Diese Management-Komponente ist wie die Client-Sicht über Port 8080 zu erreichen, nur muss der Benutzer der Rolle „manager“ (von Tomcat, nicht von Slide) zugeteilt sein. Mit Hilfe der Administrator-Sicht können aktive Sessions eingesehen werden, Contexts gestartet, gestoppt und neu geladen werden, Rechte von Benutzern und Gruppen geändert werden, uvm. Folgender Screenshot zeigt die Administrator-Sicht.

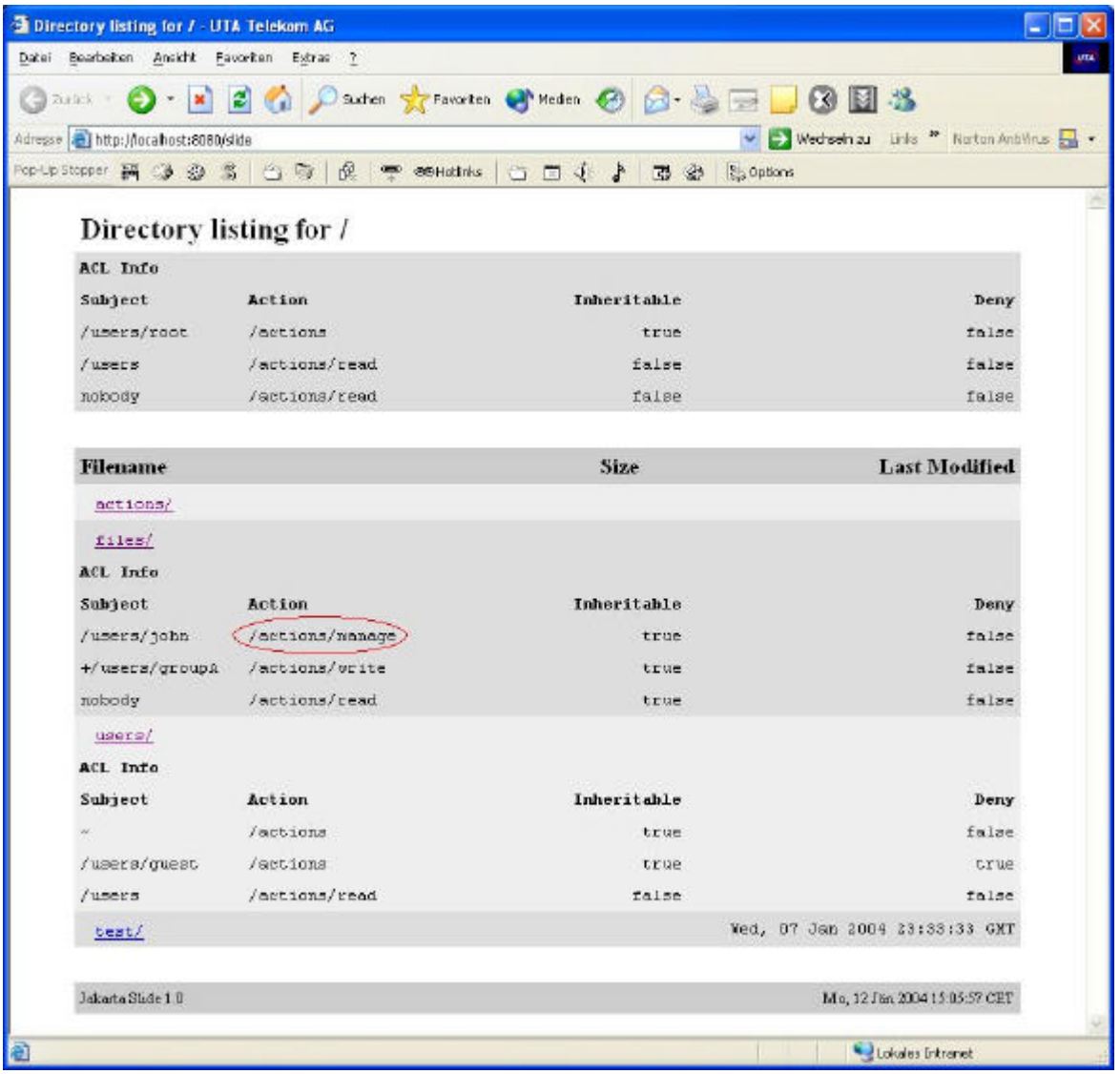

Abbildung 20: Slide Administrator-Sicht

Das Mounten eines Slide-Servers wird im Kapitel 4.5. beschrieben.
[Slide]

4.4.2. Zope

Zope ist ein Open Source Web Applikation Server, der hauptsächlich in Python geschrieben ist. Seine Transaction Object Database (d.h. Änderungen sind so lange temporär bis sie durch ein Commit bestätigt werden) kann nicht nur Inhalte und Zugangsdaten verwalten, sonder auch dynamische HTML-Templates, Skripte, eine Suchmaschine und eine Relationale Datenbank. Bei der Entwicklung wurde besonders auf Sicherheit und Datenintegrität geachtet. Zope unterstützt HTTP, FTP, WebDAV und XML-RPC.
[Zope], [Zope_WebDAV]

4.4.3. Microsoft IIS

Der Microsoft Internet Information Service hat vier Hauptziele:

- effektive Informationsverteilung über das Web
- erstellen von Web-basierten Applikationen
- Serverfunktionalitäten über das Web nutzen
- sichere Web-Services anbieten

Die WebDAV-Unterstützung, die seit Version 5.0 in den IIS integriert ist, fällt in die erste Kategorie.
Auch andere Microsoft-Produkte unterstützen WebDAV, zum Beispiel Microsoft Office seit Version 2000. Vorteile sind unter anderem in Microsoft Outlook zu finden, wo man Emails wie bei IMAP abrufen kann (die eigentliche Nachricht bleibt am Server und wird nur bei Bedarf geladen, es werden nur die Header übertragen).
[IIS]

4.4.4. Apple iDisk

Apple iDisk ist ein in den Mac OS X Finder integrierter Online-Speicherdienst auf der Basis des WebDAV-Protokolls. Die Dateien und Verzeichnisse auf der iDisk können auch offline bearbeitet werden. Sobald eine Internetverbindung besteht, werden die Daten mit den Apple

Servern synchronisiert, d.h. die Daten werden redundant gehalten. Dies eignet sich für synchronisiertes Kopieren im Mehrbenutzerbetrieb.
Das Prinzip der iDisk ist einfach in der Anwendung und bietet eine Vielzahl an Möglichkeiten für Apple-Benutzer. Einziger Nachteil: zur Verwendung benötigt man eine .Mac-Mitgliedschaft.
[iDisk]

4.5. WebDAV Clients

Mit Hilfe von WebDAV-Clients können sich Benutzer zu WebDAV-Servern verbinden und die Dateien und Verzeichnisse - je nach Rechten - einsehen, ändern und löschen. Alle neuen Betriebssysteme haben bereits WebDAV-Clients integriert. Zwei davon werden in den nächsten Kapiteln vorgesellt.

4.5.1 Windows-Explorer

Seit Windows 2000 ist der Windows-Explorer WebDAV-fähig. Das Verbinden zu einem WebDAV-Server ist einfach. Nach dem Mounten kann der WebDAV-Server wie ein lokales Laufwerk benutzt werden. Beim Testen fällt auf, dass der Windows Explorer mitunter sehr lange zum Ausführen von Operationen braucht.

Zum Mounten eines WebDAV-Servers werden in Windows nur wenige Schritte benötigt. In der Netzwerkumgebung fügt man ein neues Netzwerk hinzu, wo man als URL den Pfad zum gewünschten WebDAV-Server angibt. Nach Benennung der Netzwerkressource wird eine Verknüfung erstellt und die Daten am WebDAV-Server sind mittels Windows-Explorer anzusehen und zu manipulieren.

Abbildung 21: Verknüfung mit dem WebDAV-Server

4.5.2 Mac OS X Finder

Ähnlich wie beim Windows-Explorer kann auch im Mac OS X Finder ein WebDAV-Server eingebunden werden. Der WebDAV-Server ist wie ein lokales Verzeichnis benutzbar.

Abbildung 22: Mac OS X Finder nutzt den WebDAV-Server [Slide]

4.6. Interna

Durch WebDAV wird das HTTP Datenmodel erweitert. Diese Erweiterungen und deren genaue Funktionen werden in den folgenden Kapiteln erläutert.

4.6.1. Datenstrukturen

WebDAV erweitert das HTTP Data Model um zwei Komponenten:

- Eigenschaften, die eine Web Ressource beschreiben
- Collections, die Verzeichnisse nachbilden

WebDAV führt den Begriff einer WebDAV-Ressource (WebDAV Object Model) ein. Dabei handelt es sich um eine Properties-Body-Tupel.

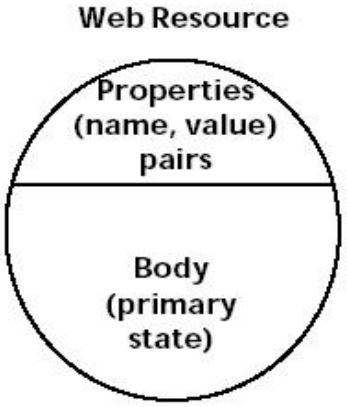

Abbildung 23: WebDAV Object Model [WebDAV_Intro_Pr]

Die Properties (Eigenschaften) beschreiben die Metadaten der Web Ressource. Dazu gehören Informationen über Name, Größe, Autor, Erstellungsdatum, Sperrstatus und andere, je nach verwendetem Schema.

WebDAV-Eigenschaften sind name-value-Tupel, wobei der Name ein URI (assoziiert mit einem Schema, das Informationen über Syntax und Semantik bietet) ist und der Wert in XML (Extensible Markup Language) beschrieben wird. Die Vorteile dieser Architektur liegen klar auf der Hand: Eine URI/URL als Name erlaubt es leicht, neue Properties hinzuzufügen, ohne diese zuvor zentral registrieren zu müssen, und ist global eindeutig. XML bietet eine erweiterbare Struktur, die leicht um neue Tags erweitert werden kann. Da der Inhalt in XML mit einem Start- und Endtag gekennzeichnet ist, können auch zusätzliche Elemente leicht hinzugefügt werden. Ein weiterer Vorteil von XML ist die Internationalisierung, die durch das UTF-8 und UTF-16 Encoding gewährleistet wird. Schlussendlich wird durch die Benutzung von XML auch die Möglichkeit geboten, andere XML-basierte Metadaten in die WebDAV-Eigenschaften einfließen zu lassen, wie zum Beispiel RDF (Resource Description Framework, W3C).

Großen praktischen Nutzen haben die Properties bei der Suche etwa nach dem Autor. Durch die Metadaten werden viele unrelevante Ergebnisse vermieden, da durch die genaue definierte Bedeutung jedes Eintrags in den Metadaten kaum Missverständnisse bei der Interpretation auftreten können. Weiters ist die Entwicklung von Metadaten enorm wichtig für die Zukunft des Webs. Es wird heute nicht nur von vielen Menschen genutzt, auch viele Maschinen (Spiders, Agenten,…) sollten sich zurechtfinden.

Die Beschreibung der Daten in XML unterliegt keinem Schema, die WebDAV Workinggroup ging davon aus, dass andere Communities (wie zum Bespiel die Dublin Core Group) eigene Metadaten-Sets entwickeln. Deshalb wurden Erleichterungen in WebDAV eingebaut um

Metadaten zu erstellen, diese zu modifizieren, zu löschen und zu empfangen. Dadurch ist es möglich, Metadaten von verschiedenen Schemata zu manipulieren.
Im Body sind die Daten der Ressource gespeichert, die klassisch über GET und PUT ausgelesen bzw. beschrieben werden können.

Mit Hilfe von Collections können mehrere Ressourcen zusammengefasst werden. Die Ressourcen werden durch ihre URIs beschrieben, die auch auf verschiedenen Servern liegen können. Eine Collection hat ebenfalls Properties und dient im Allgemeinen zur Nachbildung von Verzeichnissen wie in anderen Dateisystemen. Dadurch ermöglicht WebDAV eine hierarchische Navigation sowie hierarchische Operationen (depth-Operationen). Die Tiefe der Operationen kann mit den Flags 0 (nur die Collection), 1 (Collection und ihre Member-URIs) und „infinity“ (rekursiv alle Ressourcen, die die Collection enthält) angegeben werden.
WebDAV bietet auch Advanced Collections, die Referenzen zu Ressourcen (vergleichbar mit den Symbolic Links bei Dateisystemen) enthalten und vom Client sortiert werden können. Normalerweise sortiert WebDAV die Ressourcen hierarchisch. Weiters können Advanced Collections auch non-HTTP Ressourcen enthalten.
[WebDAV_Intro], [WebDAV_Intro_Pr], [RFC_2518]

4.6.2. WebDAV Operationen

Aufbauend auf die im vorangegangen Kapitel beschriebenen Erweiterungen bietet WebDAV folgende neue Möglichkeiten um Operationen auf Web-Servern durchzuführen:

- Eigenschaften (properties): list, add, remove
- Namespace Operationen: move, copy
- Schreibschutz (overwrite prevention): lock, unlock
- Collections: mkcol, hierarchische Operationen

Folgende Abbildung zeigt den Scope der WebDAV-Methoden und wie die Operationen die Ressource sowie die Properties oder den Body der Ressource beeinflussen können. Jene Operationen, die durch eine Sperroperation beeinflusst werden, sind speziell gekennzeichnet.

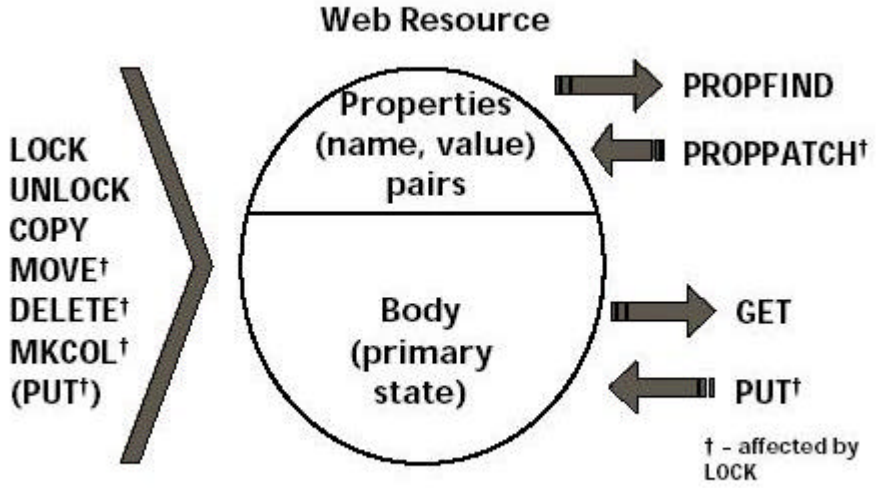

Abbildung 24: Operationen auf eine WebDAV-Ressource [WebDAV_Intro_Pr]

Wird im Folgenden von einer Ressource gesprochen, ist immer eine einzelne Ressource (non-Collection) gemeint. Logisch gesehen ist auch eine Collection eine Ressource, wird jedoch über eine Collection gesprochen wird dies jeweils erwähnt.

PROPFIND dient zum Abfragen der Properties einer Ressource. Es ist möglich alle Eigenschaften, eine einzelne Eigenschaft oder eine Liste von Eigenschaften abzufragen. Der Vorteil ist, mit einer Abfrage über das Netzwerk alle benötigten Properties zu erhalten. Falls die Ressource nicht gefunden wird, übermittelt der WebDAV-Server den Status-Code 404 (Not Found). Die Ergebnisse dieses Befehls sollen nicht in einem Cache gespeichert werden.
Wird PROPFIND auf eine Collection durchgeführt, liefert dies die URIs der Kinder sowie die Eigenschaften der Collection. Dabei kann angegeben werden ob nur die URIs der Collections zurückgegeben werden sollten (Depth 1) oder rekursiv auch die in der Collection enthaltenen Collections und deren Ressourcen zurückgegeben werden sollten (Depth infinity).
Der Vorteil der PROPFIND-Operation ist, dass Verzeichnis-Auflistungen (dir, ls,…) rein auf den Metadaten der Ressourcen beruhen, da sich dort alle nötigen Informationen befinden.

PROPPATCH dient zum Hinzufügen und Löschen von Properties einer Ressource. Dies wird mit den XML-Elementen „create“ und „remove“ durchgeführt. Dabei werden die Rechte des Benutzers sowie der Lock-Status der Ressource beachtet. Folgende Status-Codes können vom WebDAV-Server übermittelt werden:

- 200 (OK): Die Operation wurde erfolgreich durchgeführt.

- 403 (Forbidden): Der Client darf nicht auf die Ressource zugreifen. Dies ist eine allgemeine Fehlermeldung, wenn der Server nicht mehr Informationen hat/liefern will.
- 409 (Conflict): Der Inhalt des Propertys ist semantisch nicht richtig oder dieses Property ist read-only.
- 423 (Locked): Die Ressource ist gesperrt und der Client ist nicht der Lock-Owner.
- 507 (Insufficient Storage): Der Server hat zu wenig Speicherplatz um das Property zu speichern.

Die Vorzüge dieser Operation sind einfaches Handling der Modifikation von Eigenschaften, es können mit einem Netzwerkzugriff mehrere Properties gleichzeitig geschrieben werden und die Properties werden automatisch in einem konsistenten Zustand gehalten.

MKCOL erstellt eine neue Collection. Die MKCOL-Operation kann auch mit einem Request-Body aufgerufen werden. Ohne Request-Body wird eine leere Collection erstellt, mit Request-Body wird der zukünftig gespeicherte Medientyp definiert. Eine neu erstellte Collection wird automatisch zu ihrer Eltern-Collection hinzugefügt. Folgende Status-Codes können vom WebDAV-Server übermittelt werden:

- 201 (Created): Die Collection wurde erfolgreich angelegt
- 403 (Forbidden): Es gibt zwei Möglichkeiten: 1) der Server erlaubt es nicht an dieser Stelle in seinem Namespace eine Collection zu erstellen oder 2) die Eltern-Collection existiert zwar, kann aber keine URIs akzeptieren.
- 405 (Method Not Allowed): MKCOL kann nur auf eine gelöschte oder noch nicht existierende Ressource ausgeführt werden.
- 415 (Unsupported Media Type): Der Server unterstützt den Typ des Request-Bodys nicht.
- 507 (Insufficient Storage): Der Server hat zu wenig Speicherplatz um die Collection zu speichern.

GET kann auf die Daten einer Ressource, aber auch auf eine Collection ausgeführt werden und liefert den Inhalt. Die Semantik ist dabei die gleiche und unterliegt der Definition der Request-URI (RFC 2068).

Wird ein PUT auf eine existierende Ressource ausgeführt, wird dadurch der GET-Response der Ressource verändert. Properties der Ressource können bei dieser Operation neu definiert werden (zum Beispiel der Content-Typ), die anderen bleiben unverändert. Wird mit PUT eine

neue Ressource erstellt, so müssen alle Eltern-Collections existieren, ansonsten wird vom WebDAV-Server der Status-Code 409 (Conflict) zurückgegeben.
PUT kann auch auf Collections durchgeführt werden, wobei hier nach der HTTP/1.1 Spezifikation (RFC 2068) vorgegangen wird. Darin wird verlangt, dass die Ressource, auf die die Operation durchgeführt wird, unter dem angegebenen URI gespeichert ist. Das Übertragen einer Entität, die eine Collection repräsentiert, impliziert automatisch auch das Erstellen und Löschen einer Collection mit Hilfe der PUT-Operation. Laut RFC ist dies jedoch nicht erwünscht und es wird auf die MKCOL-Operation zum Erstellen und die DELETE-Operation zum Löschen von Collections verwiesen.

DELETE löscht eine Ressource und sorgt dafür, dass der Eintrag der URI aus jeder Collection gelöscht wird.
Wird DELETE auf eine Collection ausgeführt wird automatisch Depth auf „infinity“ gesetzt und alle Ressourcen und Collections (und deren Ressourcen) werden gelöscht. Tritt ein Fehler beim Löschen auf, darf keine Ressource gelöscht werden um die Konsistenz des Namespaces zu gewährleisten.

COPY dupliziert eine Quelle, die durch ihren URI angegeben wird, und speichert diese an ein ebenfalls durch einen URI angegebenes Ziel. Das Verhalten der COPY-Operation variiert je nachdem was die Quelle ist (Ressource oder Collection).
Wird die COPY-Operation auf eine Ressource durchgeführt, resultiert dies in einer exakten Kopie der Originalressource am gewünschten Ziel, d.h. es werden auch alle Properties kopiert.
Wird eine COPY-Operation auf eine Collection durchgeführt und kein Depth-Header angegeben wird dieser automatisch auf „infinity“ gesetzt. Erlaubte Werte für den Depth-Header sind „0“ und „infinity“.

- Bei einer COPY-Operation mit „Depth: 0“ wird nur die Collection und ihre Properties kopiert.
- Bei einer COPY-Operation mit „Depth: infinity“ wird die Collection mit allen Ressourcen und Collections, die in der Collection gelistet sind, rekursiv zum Ziel kopiert. Dadurch bleiben die relativen Pfade am Ziel gleich.

Am Ende der Operation muss der Namespace in einem konsistenten Zustand sein. Tritt ein Fehler beim Kopieren auf, darf keine Ressource kopiert werden.

Der Overwrite Header kann gesetzt werden, um Ressourcen oder Collections am Ziel zu überschreiben, falls diese bereits existieren. Ist der Header auf „T“ gesetzt, wird zuerst ein DELETE auf die vorhandene Ressource/Collection durchgeführt, bevor die COPY-Operation durchgeführt wird. Ist der Header auf „F“ gesetzt und die Ressource/Collection existiert bereits, so bricht die COPY-Operation ab.
Folgende Status-Codes können bei einer COPY-Operation vom Server übermittelt werden:

- 201 (Created): Die Ressource/Collection wurde erfolgreich kopiert.
- 204 (No Content): Die Quelle wurde erfolgreich über eine existierende Ressource kopiert.
- 403 (Forbidden): Quell- und Ziel-URI sind gleich.
- 409 (Conflict): Eine Ressource kann nicht erstellt werden, solange nicht eine oder mehrere Eltern-Collections erstellt wurden.
- 412 (Precondition Failed): Es ist entweder ein Fehler beim Kopieren der Properties aufgetreten oder der Overwrite Header war auf „F” gesetzt und die Ziel-Ressource/Collection existiert bereits.
- 423 (Locked): Die Ziel-Ressource ist gesperrt.
- 502 (Bad Gateway): Dieser Fehler kann auftreten, wenn das Ziel auf einem anderen Server liegt und dieser das Kopieren der Ressource/Collection nicht gestattet.
- 507 (Insufficient Storage): Der Server hat zu wenig Speicherplatz um die Ziel-Ressource/Collection zu speichern.

Da die COPY-Operation rein am WebDAV-Server durchgeführt wird, wird die Ressource/Collection gar nicht über das Netz übertragen.

MOVE ist logisch äquivalent eine COPY-Operation, gefolgt von einer Konsistenz-Überprüfung und anschließender DELETE-Operation auf die Quelle, nur in einem Schritt ausgeführt. Bei der Konsistenz-Überprüfung werden Updates durchgeführt, damit auch andere Collections, die verschobene Ressourcen/Collections enthalten die neuen URIs verlinken.
Wird MOVE auf eine Collection ausgeführt, so muss der Depth-Header auf „infinity“ gesetzt sein. Die Collection und alle darin gelisteten Ressourcen und Collections werden auf den Ziel-Pfad verschoben.
Tritt beim Verschieben ein Fehler auf, darf keine Ressource/Collection verschoben werden um die Konsistenz des Namespaces zu garantieren.

Wie bei der COPY-Operation kann auch bei der MOVE-Operation ein Overwrite Header gesetzt werden. Die Funktion ist bei beiden Operationen die gleiche.
Folgende Status-Codes können bei einer MOVE-Operation vom Server übermittelt werden:

- 201 (Created): Die Ressource/Collection wurde erfolgreich verschoben.
- 204 (No Content): Die Ressource/Collection wurde erfolgreich verschoben und hat eine vorhandene Ressource/Collection überschrieben.
- 403 (Forbidden): Quell- und Ziel-URI sind gleich.
- 409 (Conflict): Eine Ressource kann nicht erstellt werden, solange nicht eine oder mehrere Eltern-Collections erstellt wurden.
- 412 (Precondition Failed): Es ist entweder ein Fehler beim Kopieren der Properties aufgetreten oder der Overwrite Header war auf „F" gesetzt und die Ziel-Ressource/Collection existiert bereits.
- 423 (Locked): Die Quell- oder Ziel-Ressource/Collection ist gesperrt.
- 502 (Bad Gateway): Dieser Fehler kann auftreten, wenn das Ziel auf einem anderen Server liegt und dieser das Speichern der Ressource/Collection nicht gestattet.

Vorteile der MOVE-Operation sind, dass sie performanter ist als ein COPY mit einem gefolgten DELETE und dass die Ressource/Collection beim Verschieben auch umbenannt werden kann.
[WebDAV_Intro], [WebDAV_Intro_Pr], [RFC_2518]

4.6.3. Locking

Die Möglichkeit eine Ressource sperren (lock) zu können bietet einen Mechanismus für den konfliktfreien Zugriff auf Ressourcen. Benutzt ein Client eine Sperre bekommt er die Garantie, dass kein anderer Client diese Ressource modifizieren kann, bis er sie wieder freigegeben hat. Damit kann das sogenannte „lost update"-Problem vermieden werden.
Das WebDAV-Protokoll sieht nur eine Art von Sperren vor: Write Locks. Es ist aber modular gehalten, sodass bei Bedarf in Zukunft auch andere Locktypen hinzugefügt werden können. Ein Write Lock erlaubt es anderen Clients Leseoperationen (PROPFIND, GET) auf die gesperrte Ressource durchzuführen.
Es werden auch zwei Lock-Level eingeführt, wie der Client eine Ressource sperren kann: Exclusive und Shared Locking.
Die einfachste Form des Sperrens ist der Exclusive Lock. Dabei werden nur einem Client Zugriffsrechte auf eine Ressource gewährt. Der Vorteil ist, dass der Server keine Ergebnisse

zusammenfügen muss, es kann ja immer nur ein Client die Ressource ändern. Die folgende Abbildung zeigt einen typischen Ablauf eines Exclusive Locks:

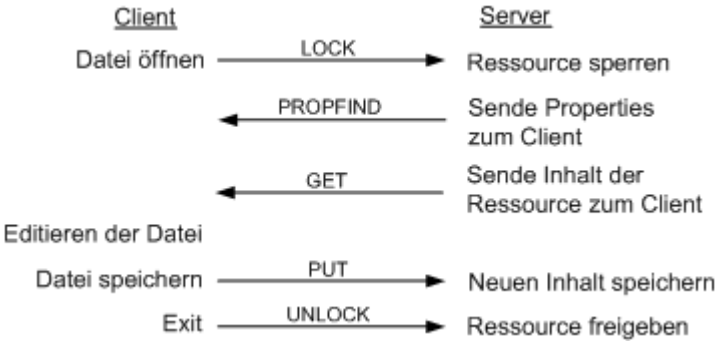

Abbildung 25: Exclusive Lock [WebDAV_Intro_Pr]

Oft ist aber das Ziel einer Sperre nicht das generelle Schreibverbot anderer Clients auf eine Ressource, sondern nur um anzuzeigen, dass man von seinen Schreibrechten gebrauch machen möchte. Dafür wurden die Shared Locks eingeführt. Dieser erlaubt es mehreren Clients, gleichzeitig dieselbe Ressource zu locken. Aus den zwei verschiedenen Sperrvorgängen ergibt sich für zwei Clients folgende Tabelle:

Lock Request / Lock Status	Shared Lock	Exclusive Lock
Kein Lock	ja	ja
Shared Lock	ja	nein
Exclusive Lock	nein	nein

Abbildung 26: Lock-Kompatibilität

Das Prinzip der Shared Locks basiert auf dem Vertrauen zwischen den Clients, dass diese keine Daten eines Anderen überschrieben. Zieht man die Shared Locks in Betracht, ergeben sich drei Benutzergruppen, die verschiedene Rechte auf eine Ressource haben. Jede Benutzergruppe ist eine Teilmenge der vorigen:

- die Menge aller Benutzer (des Internets)
- jene Benutzer, die Schreibrechte auf die Ressource haben
- jene Clients, die Shared Lock Holders sind

Wie die Clients, die Shared Locks nutzen, sich untereinander koordinieren (Face-to-Face, Telefon, Email, ICQ,…) obliegt ihnen selbst und wird nicht von WebDAV gesteuert.

Um einen Lock zu repräsentieren wurden Lock Token eingeführt. Dieser wird bei einer erfolgreichen Sperre einer Ressource zum Client übermittelt. Der Token wird durch einen URI repräsentiert, die eindeutig über alle Ressourcen und die Zeit ist. Auch soll der Client beim Lock Request einen Besitzernamen für die Sperre übermitteln, damit andere Clients herausfinden können welcher Client zurzeit eine spezifische Ressource gesperrt hat um ihn eventuell auffordern zu können, diese zu entsperren.

Nachdem der Client die Ressource geschrieben hat, sollte er diese wieder entsperren (unlock). Führt der Client diese Operation nicht durch gibt es entweder einen Lock Timeout, nachdem die Ressource wieder freigegeben wird, oder der Administrator des Servers kann die Ressource freigeben. Daraus ergibt sich für eine Sperre folgender Lifecycle:

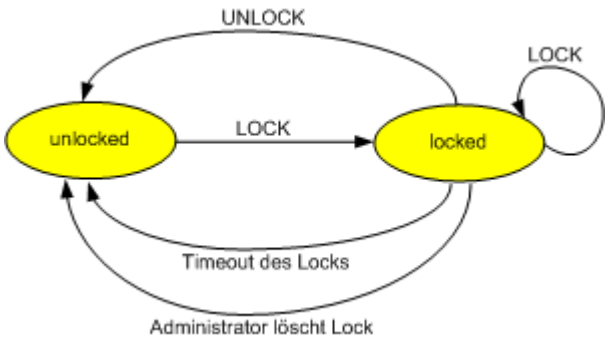

Abbildung 27: Lock Livecycle [WebDAV_Intro_Pr]

Es ist auch möglich, Sperren auf Collections durchzuführen. Man spricht dann von einem Hierarchy Lock. Dafür wird der Depth-Header auf „infinity“ gesetzt. Es wird nur ein Lock Token für die gesamte Ressource zurückgegeben. Werden Ressourcen in einer Collection hinzugefügt, verschoben oder gelöscht, kann dies die Sperre verändern. Hier gibt es folgende Möglichkeiten:

- COPY/MOVE IN: Eine Ressource wird in eine gesperrte Hierarchie hineinkopiert oder hineinverschoben. Diese Ressource wird zu dem Lock hinzugefügt.
- COPY/MOVE WITHIN: Eine Ressource wird innerhalb der Hierarchie kopiert oder verschoben. Die Ressource bleibt im Lock.
- COPY OUT: Eine Ressource wird kopiert und außerhalb der Hierarchie gespeichert. Die Quelle bleibt im Lock, die Kopie ist nicht im Lock.
- MOVE OUT: Ein Ressource die außerhalb der Hierarchie verschoben wird, wird aus dem Lock entfernt.
- DELETE: Die gelöschte Ressource wird aus dem Lock gelöscht.

[WebDAV_Intro], [WebDAV_Intro_Pr], [RFC_2518]

4.6.4. Versioning

Der Sinn von Versioning ist es, frühere Versionen von Dokumenten zu speichern um diese auch später noch einsehen zu können. Weiters soll Versioning auch kollaboratives Erstellen von Dokumenten unterstützen, indem mehreren Personen gleichzeitig ein Dokument editieren können.
Die Vorteile von Versioning im WWW sind:

- Jede Ressource hat eine explizite Geschichte und eine persistente Identität über die verschiedenen Stadien ihrer Entwicklung. So ist es leicht, alte oder alternative Versionen einer Ressource einzusehen. Oft sind auch die Modifikations- und Autorenliste eine wichtige Information.
- Jede Version einer Ressource hat einen eindeutigen, bleibenden Namen. So kann sie auch verlinkt werden, ohne das Problem, dass bei der nächsten Version der Link ungültig wird.

WebDAV bietet einfache und erweiterte Versioning-Funktionen.
Einfaches Versioning (basic versioning) erlauben dem Benutzer:

- Eine Ressource unter Version-Control zu setzen
- Abfragen ob eine Ressource unter Version-Control steht
- Abfragen ob eine Änderung einer Ressource automatisch als neue Version gespeichert wird
- Erstellen und abfragen einzelner Versionen einer Ressource

Erweitertes Versioning bietet Funktionen für paralleles Arbeiten an einem Set von Ressourcen.

Im Folgenden werden zwei Begriffe bei der Erklärung von Properties beigefügt, die an dieser Stelle kurz erläutert werden:

- protected: Ist ein Property geschützt, kann der Wert nicht verändert werden.
- computed: Der Wert der Eigenschaft wird errechnet und kann nur vom WebDAV-Server geändert werden.

Um eine WebDAV-Ressource Versioning-fähig zu machen, müssen einige Properties hinzugefügt werden:

- comment: Ein Kommentar warum diese Version erstellt wurde.
- creator-displayname: Der Name des Autors der vorliegenden Version.
- supported-methode-set (protected): Listet alle unterstützten Methoden der Ressource.
- support-live-property-set (protected): Listet alle Live-Properties der Ressource.
- support-report-set (protected): Listet alle unterstützten Reports.
- check-in (protected): Dieses Property tritt bei checked-in version-controlled Ressourcen auf und kennzeichnet eine Version, die den gleichen Inhalt, aber veraltete Properties (dead properties) als die version-controlled Ressource hat. Dieses Property wird entfernt sobald die Resource ausgecheckt (checked-out) wird. Das heißt das check-in identifiziert eine neue Version einer Ressource.
- auto-version: Dieses Property dient zum automatischen Versioning, wobei unterschiedliche Werte dieses Properties verschiede Ergebnisse liefern. Folgende Werte sind möglich: checkout-checkin, checkout-unlocked-checkin, checkout, locked-checkout.
- checked-out (protected): Dieses Property kennzeichnet die Version, die vom checked-in-Property gekennzeichnet wurde als dieses ausgecheckt wurde. Dieses Property wird entfernt sobald die Ressource wieder eingecheckt (check-in) wird.
- predecessor-set: Dieses Property beschreibt die Vorgängerversion. Jede Version - außer der „Root-Version“ - hat mindestens einen Vorgänger.
- successor-set (computed): Dieses Property listet jene Versionen, deren predecessor-set-Property diese Version kennzeichnen, d.h. es ist eine Liste der direkten Nachfolger.
- checkout-set (computed): Dieses Property listet jene Versionen, deren checked-out-Property diese Version kennzeichnen.
- version-name (protected): Dieses Property enthält einen vom Server generierten String, der für jede Version verschieden ist. Er ist rein für den Benutzer gedacht.

Um Versioning in WebDAV zu erlauben, wird zu allen WebDAV-Operationen und zum Workspace neue Semantik hinzugefügt. Dies sind neue Pre- und Postconditions, die bei der Durchführung der einzelnen Operationen erfüllt werden müssen, sowie die MKWORKSPACE-Operation um eine neue Workspace-Ressource zu erstellen.

WebDAV-Versioning bietet eine Vielzahl an Methoden:

- Die Version-Control-Methode erklärt an einem Beispiel: Versioning für eine html-Datei

  Request:

  ```
  VERSION-CONTROL /foo.html HTTP/1.1
  Host: www.test.org
  Content-Length: 0
  ```

  Response:

  ```
  HTTP/1.1 200 OK
  ```

  Die Datei foo.html wird unter Version-Control gestellt. Es wird eine Kopie der Datei erstellt mit gleichem Inhalt und „dead Properties“. Das check-in-Property von foo.html kennzeichnet diese Version.
- Die Report-Methode liefert Informationen über die Ressource. Anders als Metadaten, die nur einen Wert haben, kann so ein Report variable Länge haben.
- Der Version-Tree-Report liefert die gewünschten Properties aller vorhandener Versionen.
- Der Expand-Property-Report bietet einen Mechanismus um Properties von jenen Ressourcen zu bekommen, die in einem Property gelinkt sind (z.B.: predecessor-set).
- Die Update-Methode bietet einen Mechanismus um den Status einer checked-in Ressource auf eine andere Version (in der Versions-History) zu ändern.
- Mit Hilfe des Label-Features kann jeder Version ein eindeutiger String zugewiesen werden, der sie von allen anderen Versionen unterscheidet (in der gleichen Version-History). Dieses Label kann entweder automatisch vom Server erstellt werden oder (um sinnvolle Namen zu vergeben) vom Client gesetzt werden. Durch die Label-Methode kann das Label geändert werden.
- Das Workspace-Feature erlaubt es mehreren Benutzern, gleichzeitig neue Versionen zur gleichen Version-History hinzuzufügen. Es wird der Begriff einer Workspace-Ressource eingeführt. Dies ist eine Collection, die eine Liste von verwandten version-controlled und non-version-controlled Ressourcen enthalten kann. Es werden mehrere Workspaces benutzt um verschiedene Versionen und Konfigurationen von einem Set von (version-controlled) Ressourcen parallel zu erstellen. Um die Änderungen in einem Workspace auch in einem anderen Workspace sichtbar zu machen, muss diese Ressource eingecheckt werden um ein Update durchzuführen. Danach sind der Inhalt der Ressource sowie deren Properties im anderen Workspace in der neuen Version dargestellt. Das heißt hier werden die Konfiguration der Versionen und das Auschecken der Ressourcen vom Server durchgeführt.

- Das Working-Resource-Feature bietet eine Alternative zum Workspace-Feature. Hier wird die Konfiguration am Client durchgeführt. Dies erleichtert zwar die Server-Implementierung, erlaubt es dem Benutzer aber nicht, auf Konfigurationen auf anderen Clients zuzugreifen (in einem anderen Büro, zu Hause oder unterwegs). Beim Working-Resource-Feature arbeiten alle Clients mit einem Set von gemeinsam genutzten Ressourcen, d.h. jeder Client sieht sofort alle Änderungen sobald sie durchgeführt werden.

Die erweiterten Versioning Features (advanced versioning features) beschäftigen sich mit den Problemen bei parallelem Entwickeln und Configuration Management von multiplen Sets interagierender Ressourcen. Es wird die Infrastruktur geboten, um große Websites koordiniert parallel zu verwalten.

Um bei Updates keine Daten zu verlieren, wurden bei WebDAV Versioning Merge Features eingeführt. Sobald ein Benutzer die Änderungen, die ein anderer Benutzer auf eine version-controlled Ressource ausgeführt hat, akzeptiert, ist es wichtig, dass nicht nur ein Update der Ressource durchgeführt wird, da dies zu einem Überschreiben der Änderungen des Benutzers führt. Die Version aus dem anderen Worksspace sollte vielmehr mit der Version des Benutzers verbunden (merge) werden. Über die Versions-History wird festgestellt welche Version neuer ist (line of descent von der Root-Version aus). Falls es dem Server möglich ist die Verschmelzung der Dokumente automatisch durchzuführen, wird dies gemacht. Der Benutzer muss aber anschließend die Korrektheit überprüfen und bestätigen. Ist der Server nicht in der Lage die Dokumente zu verschmelzen muss dies der Benutzer durchführen.

Da eine Configuration (dies ist ein Set von Ressourcen - nicht zu verwechseln mit einer Collection, die nur eine einzige Ressource ist) eine große Anzahl von Ressourcen beinhalten kann, ist es oft sinnvoll, diese auf kleine Configurations aufzusplitten. Dies kann mit der Baseline Operation durchgeführt werden.

WebDAV bietet auch eine Methode, um einzelne logische Änderungen auf mehrere Versionen einer Ressource durchzuführen. Diese Änderungen werden mit Hilfe des Activity Features durchgeführt. Ein Activity ist eine Ressource die mehrere Versionen einer Ressource selektiert, die sich auf einer einfachen durchgehenden Versions-Linie (line of descent) befinden (Das heißt die Versionen sind durch das successor-Property miteinander verbunden).

Wenn eine Collection unter Version-Control gestellt wird, wird wie für jede andere Ressource auch eine Version History angelegt. Um die Standard-Versioning-Semantik zu bewahren soll eine Collection-Version nur Informationen aufnehmen, die die version-controlled Bindings einer Collection betreffen. Um klar eine Modifikation des Namespaces mit einer Modifikation des Inhaltes der Collection unterscheiden zu können, hat eine Version einer Collection keine Members (sprich die URIs zu den einzelnen Ressourcen), sondern enthält in einem Property (version-controlled-binding-set) alle Binding-Namen und Version-History-Ressource von jedem version-controlled Member der Collection.
[RFC_3253], [WebDAV_Versioning_Pr]

4.6.5. Access Control

Das Ziel der WebDAV Access Control Extensions ist es, eine Zugriffkontrolle für Inhalt und Metadaten von WebDAV Ressourcen zu bieten. Diese Erweiterung von WebDAV ist noch nicht standardisiert und liegt zurzeit als 13. Draft vor (Stand Jänner 2004).
WebDAV Access Control kann auf jedem (Content) Repository mit Sicherheitsfeatures implementiert werden, zum Beispiel auf einem UNIX-Dateisystem. Um Access Control durchführen zu können, muss das System wissen, wer man ist (Benutzer, Client, Software, Server, eine ganze Gruppe,…). Dieser Identifier wird „Principal“ genannt. Eine Gruppe (group) ist eine Menge von Principals, die die gleichen Rechte haben. Die Operationen, die ein Principal durchführen darf, werden in der Access Control List (ACL) gespeichert (im Zusammenhang mit einer Ressource). Eine ACL enthält als einzelne Einträge Access Control Entries (ACEs), wobei jedes ACE einen Principal mit seinen Privilegien (erlaubte und verbotene Operationen auf eine Ressource) spezifiziert. Das heißt, sobald ein Principal eine Operation auf eine Ressource ausführen will, evaluiert der Server anhand der ACEs in der ACL ob der Benutzer die Rechte besitzt um die Operation durchführen zu dürfen.
Um sich als ein Principal authentifizieren zu können, muss die Clientsoftware dem Benutzer erlauben sich am Server als solcher anzumelden. In WebDAV werden hierfür die http(s)-Schema URLs genutzt.
[WebDAV_ACL]

4.6.6. DAV Searching and Locating (DASL)

WebDAV bietet auch eine Möglichkeit, um nach Inhalten und Properties von Ressourcen zu suchen. WebDAV SEARCH ist ein Suchprotokoll zum Transport von Abfragen (queries) und Ergebnissen (results), das es dem Client ermöglicht, Server-seitige Suchfunktionen zu nützen. Dies soll die Komplexität bei der Implementierung von Clients vermindern und trotzdem eine mächtige Suchfunktion bieten.

DASL besteht aus folgenden Komponenten:

- die SEARCH Methode: Der Client nutzt diese Methode, um eine Server-seitige Suche auszulösen. Der Body der Anfrage definiert die Suchabfrage. Das heißt die SEARCH Methode spielt eine Rolle beim Transport von Abfrage und Ergebnis, aber es definiert nicht die Semantik der Abfrage. Der Typ der Anfrage definiert die Semantik.
- der DASL Response Header: In dem Header stehen Informationen über die vom Server unterstützten Abfragegrammatik. Der Wert ist ein URI, der die Art der Grammatik spezifiziert.
- dem DAV:searchrequest XML Element: Teil des Request-Bodys.
- dem DAV:query-schema-discovery XML Element: Teil des Request-Bodys. Dient zum Herausfinden des verwendeten Abfrage-Schemas.
- dem DAV:basicsearch XML Element und Abfragegrammatik: Erlaubt es dem Benutzer Suchabfragen zu formulieren, die bei WebDAV-Szenarien allgemein nützlich sind (select, from, where, orderby, limit).
- dem DAV:basicsearchschema XML Element: Schema für DAV:basicsearch

Ein hypothetisches Beispiel soll die Funktion verdeutlichen: Gesucht wird in natürlicher Sprache nach einem thailändischen Lokal in Los Angeles.
Request:

```
SEARCH / HTTP/1.1
Host: example.org
Content-Type: application/xml
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<D:searchrequest xmlns:D="DAV:" xmlns:F="http://example.com/foo">
  <F:natural-language-query>
    Find the locations of good Thai restaurants in Los Angeles
  </F:natural-language-query>
</D:searchrequest>
```

Response:

```
HTTP/1.1 207 Multi-Status
Content-Type: text/xml; charset="utf-8"
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<D:multistatus xmlns:D="DAV:"
   xmlns:R="http://example.org/propschema">
  <D:response>
    <D:href>http://siamiam.test/</D:href>
    <D:propstat>
      <D:prop>
        <R:location>259 W. Hollywood</R:location>
        <R:rating><R:stars>4</R:stars></R:rating>
      </D:prop>
      <D:status>HTTP/1.1 200 OK</D:status>
    </D:propstat>
  </D:response>
</D:multistatus>
```

[DASL], [DASL_Draft]

4.6.7. Request-Response Flow

Bei jeder Operation, die ein WebDAV-Client ausführt, kommt es zu einem Request-Response-Flow zwischen Client und Server. Anhand von drei Beispielen wird dieser erklärt.

Request-Response-Flow bei OPTION:

Die OPTION-Operation liefert alle Operationen, die auf dem WebDAV-Server ausgeführt werden können.

Request:

```
OPTIONS /somefolder HTTP/1.1
Host: www.example.org
```

Response:

```
HTTP/1.1 200 OK
Allow: OPTIONS, GET, HEAD, POST, PUT, DELETE, TRACE, COPY, MOVE
Allow: MKCOL, PROPFIND, PROPPATCH, LOCK, UNLOCK
```

Request-Response-Flow bei PROPFIND (auf eine Collection):

In diesem Beispiel wird die PROPFIND-Operation auf die Ressource http://www.foo.bar/container/ ausgeführt. Der Depth Header wird auf 1 gesetzt, d.h. die Anfrage betrifft die Collection und ihre Kinder. Das Propfind XML Element enthält das allprob XML Element, d.h. es sollen Name und Wert aller Properties jeder einzelnen Ressource zurückgegeben werden.

Die Ressource (Collection) http://www.foo.bar/container/ hat 6 definierte Properties. Die ersten zwei (bigbox und author) sind unter http://www.foo.bar/boxschema/ definiert, die restlichen vier (creationdate, displayname, resourcetype und supportedlock) WebDAV-spezifische Properties. Die GET-Operation wird auf diese Ressource nicht unterstützt, daher können auch die get* Properties (z.B.: getContentLength) nicht abgefragt werden.
Die Ressource http://www.foo.bar/container/front.html hat 9 definierte Properties: bigbox (eine andere Instanz des „bigbox“-Property-Typs), creationdate, displayname, getcontentlength, getetag, getlastmodified, resourcetype, supportedlock.

Request:

```
PROPFIND  /container/ HTTP/1.1
Host: www.foo.bar
Depth: 1
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:propfind xmlns:D="DAV:">
  <D:allprop/>
</D:propfind>
```

Response:

```
HTTP/1.1 207 Multi-Status
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:multistatus xmlns:D="DAV:">
  <D:response>
       <D:href>http://www.foo.bar/container/</D:href>
       <D:propstat>
            <D:prop xmlns:R="http://www.foo.bar/boxschema/">
                 <R:bigbox>
                      <R:BoxType>Box type A</R:BoxType>
                 </R:bigbox>
                 <R:author>
                      <R:Name>Hadrian</R:Name>
                 </R:author>
                 <D:creationdate>
                      1997-12-01T17:42:21-08:00
                 </D:creationdate>
                 <D:displayname>
                      Example collection
                 </D:displayname>
                 <D:resourcetype><D:collection/></D:resourcetype>
                 <D:supportedlock>
                      <D:lockentry>
                           <D:lockscope><D:exclusive/></D:lockscope>
                           <D:locktype><D:write/></D:locktype>
                      </D:lockentry>
                      <D:lockentry>
                           <D:lockscope><D:shared/></D:lockscope>
                           <D:locktype><D:write/></D:locktype>
```

```
                    </D:lockentry>
               </D:supportedlock>
          </D:prop>
          <D:status>HTTP/1.1 200 OK</D:status>
     </D:propstat>
  </D:response>
  <D:response>
     <D:href>http://www.foo.bar/container/front.html</D:href>
     <D:propstat>
          <D:prop xmlns:R="http://www.foo.bar/boxschema/">
               <R:bigbox>
                    <R:BoxType>Box type B</R:BoxType>
               </R:bigbox>
               <D:creationdate>
                    1997-12-01T18:27:21-08:00
               </D:creationdate>
               <D:displayname>
                    Example HTML resource
               </D:displayname>
               <D:getcontentlength>
                    4525
               </D:getcontentlength>
               <D:getcontenttype>
                    text/html
               </D:getcontenttype>
               <D:getetag>
                    zzyzx
               </D:getetag>
               <D:getlastmodified>
                    Monday, 12-Jan-98 09:25:56 GMT
               </D:getlastmodified>
               <D:resourcetype/>
               <D:supportedlock>
                    <D:lockentry>
                         <D:lockscope><D:exclusive/></D:lockscope>
                         <D:locktype><D:write/></D:locktype>
                    </D:lockentry>
                    <D:lockentry>
                         <D:lockscope><D:shared/></D:lockscope>
                         <D:locktype><D:write/></D:locktype>
                    </D:lockentry>
               </D:supportedlock>
          </D:prop>
          <D:status>HTTP/1.1 200 OK</D:status>
     </D:propstat>
  </D:response>
</D:multistatus>
```

Request-Response-Flow bei COPY (mit Overwrite):

Der Statuscode 204 (siehe auch Kapitel 4.6.2.) zeigt, dass die Ressource erfolgreich über eine bereits vorhandene Ressource kopiert worden ist. Will man das Überschrieben beim Kopieren nicht erlauben, muss dies explizit angegeben werden (Overwrite: F), standardmäßig ist der Overwrite Header auf „T“ gesetzt.

Request:

```
COPY /test/index.html HTTP/1.1
Host: www.example.org
Destination: http://www.example.org/otherfolder/index.html
```

Response:

```
HTTP/1.1 204 No Content
```

Mit Hilfe eines Sniffers (z.B.: Ethereal) kann man die Kommunikation zwischen WebDAV-Client und Server beobachten. Folgende Abbildung zeigt einen Ausschnitt aus einem Paket, das der Response auf eine PROPFIND-Operation war:

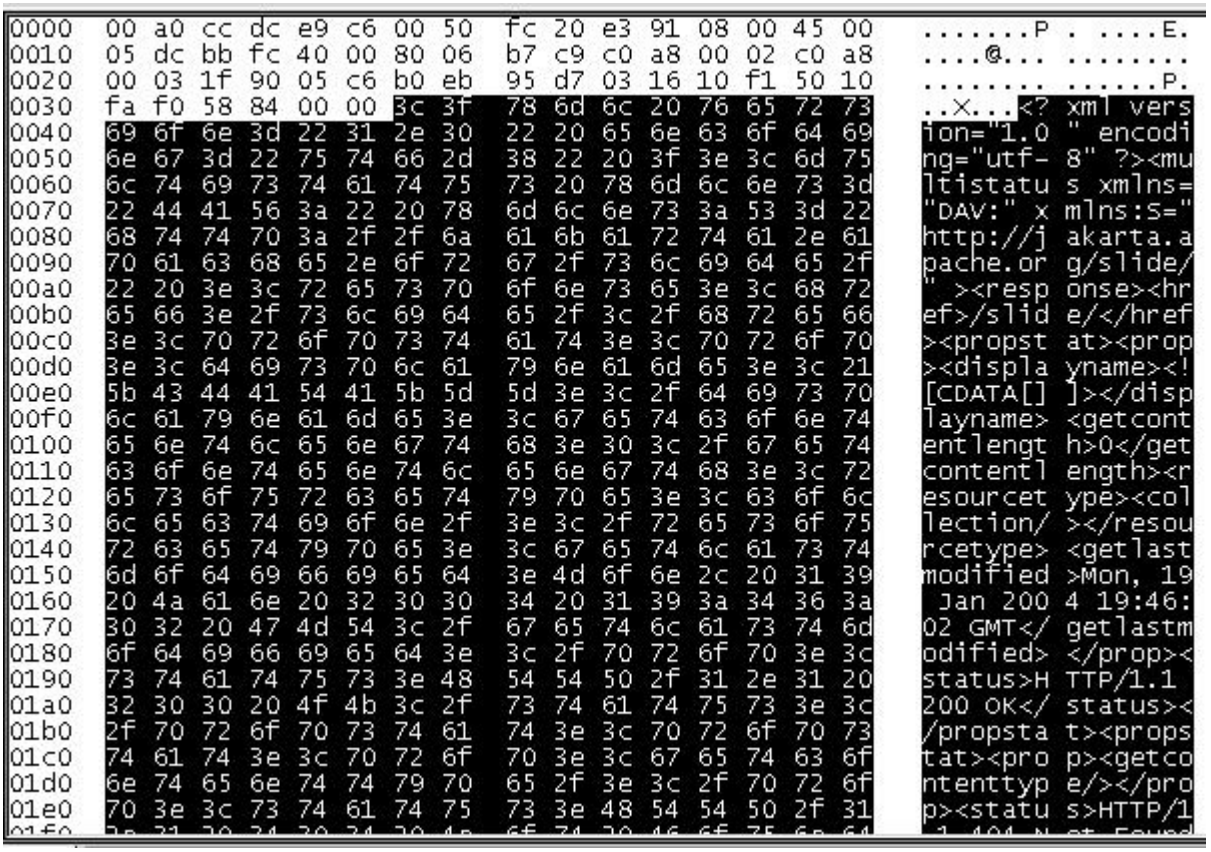

Abbildung 28: Server-Response auf eine PROPFIND-Abfrage

[RFC_2518]

4.6.8. Fehlerbehandlung

In RFC 2068 werden Status-Codes für das HTTP-Protokoll definiert. Diese werden auch von WebDAV-Servern genutzt. Zusätzlich werden weitere Status-Codes hinzugefügt, die für

WebDAV notwendig sind, um dem Client Informationen über Erfolg oder Misserfolg einer Operation bieten zu können:

- 102 (Processing): Mit diesem Status-Code wird der Client informiert, dass sein Request akzeptiert, jedoch noch nicht abgearbeitet wurde. Es wird empfohlen diesen Status-Code zu übermitteln, wenn das Abarbeiten des Befehls erst in 20 Sekunden oder später beendet wird. Der Server muss, nachdem die Operation abgearbeitet wurde, noch einen anderen Status-Code über den Ausgang der Operation an den Client schicken. Der Sinn dieses Status-Codes ist, dass ein Client bei langem Warten kein Timeout durchführt.
- 207 (Multi-Status): Dieser Status-Code enthält mehrere andere Status-Codes, die in einem XML Element namens „multistatus“ gelistet sind. Es dürfen nur Status-Codes der 200er, 300er, 400er und 500er Serien enthalten sein.
- 422 (Unprocessable Entity): Dieser Status-Code wird retourniert falls der Server zwar den Content-Typ versteht (andernfalls gibt es einen 415 - Unsupported Media Type Fehler) und die Syntax der Request-Entity korrekt ist (andernfalls gibt es einen 400 - Bad Request Fehler), aber die enthaltene Instruktion nicht ausführen kann. Dieser Fehler tritt bei semantisch fehlerhaften XML Instruktionen auf.
- 423 (Locked): Dieser Status-Code weist darauf hin, dass entweder die Quell- oder Ziel-Ressource gesperrt ist und die Operation deshalb nicht durchgeführt werden kann.
- 424 (Failed Dependency): Der Server sendet diesen Status-Code, wenn eine Operation nicht auf eine Ressource ausgeführt werden kann, da die auszuführende Aktion von einer anderen Aktion abhängt, die fehlgeschlagen ist.
- 507 (Insufficient Storage): Dieser interne Serverfehler tritt auf, wenn eine Operation nicht durchgeführt werden kann, weil der Server die benötigte Repräsentation einer Ressource/Collection/Properties nicht speichern konnte.

[RFC_2518]

5. Dokumentation des WeLearn-Moduls

Dieses Kapitel beschäftigt sich mit dem WebDAV-Explorer für WeLearn. Dies ist ein als Java-Servlet implementiertes Modul, das als WebDAV-Client dient. Es ist möglich den WebDAV-Explorer als allein stehendes Programm zu nutzen (mittels eines Browsers) oder ihn in ein anderes Programm als Modul zu integrieren.

Der WebDAV-Explorer bietet Funktionen um Namespace Operationen, Datei Up- und Download und andere Operationen auf einem WebDAV-Server durchzuführen.

Folgende Grafik zeigt ein Szenario, indem der WebDAV-Explorer zum Einsatz kommt:

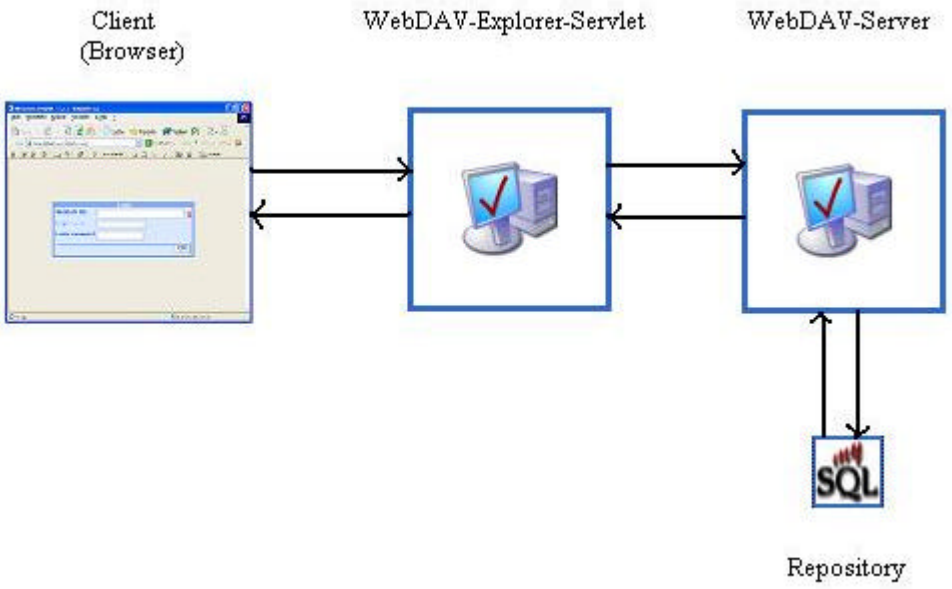

Abbildung 29: Szenario

5.1. Designziele

Ausgangspunkt bei der Entwicklung waren folgende Überlegungen:

- Modul: Zur leichten Integrierung in ein großes Projekt muss jeder Teil modular gehalten werden um leicht austauschbar zu sein, falls es in Zukunft neuere, bessere Technologien gibt oder Anpassungen an andere Systeme erforderlich werden.
- Verwendung bestehender Bibliotheken: Im WeLearn WebDAV-Explorer kommen mehrere externe, frei verfügbare Bibliotheken zum Einsatz: Slide Client Bibliothek

(aus dem Apache Jakarta Project) [Slide], Millstone (Model-View-Controller) [Millstone], Log4j (Logging) [Log4j] und i18n (Internationalisierung) [i18n].

- Internationalisierung: Alle Strings des Programmcodes werden von einer Textdatei ausgelesen. Diese ist leicht auszutauschen und bringt die Möglichkeit, schnell die Sprache zu wechseln, ohne den Programmcode ändern und neu kompilieren zu müssen.
- Plattformunabhängigkeit: Als Java-Servlet ist das Modul plattformunabhängig und kann über jeden üblichen Webbrowser bedient werden.
- Behindertengerecht: Auch sehbehinderten Menschen darf der Zugang zu Lernplattformen nicht verwehrt werden, deshalb ist der WebDAV-Explorer so ausgelegt, dass er auch ohne Java-Script ausführbar ist und so in einem reinen Textbrowser angezeigt werden kann.
- Intuitive Bedienbarkeit: Alle Betriebssysteme haben Explorer (Windows Explorer, Mac OS Finder,…) integriert und der Benutzer ist mit diesen vertraut. Darum ist es naheliegend, ein ähnliches Look & Feel für den WebDAV-Explorer zu benutzen.
- Hilfe: Sobald eine Eingabe des Benutzers erforderlich wird, bekommt er eine Hilfestellung, warum diese nötig ist (zum Beispiel wenn eine hochgeladene Datei mit demselben Namen bereits existiert und er diese umbenennen muss).

5.2. Architektur

Die Hierachie der Klassen:

GUI-Elemente: Diese Klassen betreffen Objekte zur Visualisierung.

welearn\millstone\extensions\AbstractDialog
welearn\millstone\extensions\ButtonPanel
welearn\millstone\extensions\Dialog
welearn\millstone\extensions\DialogPanel
welearn\millstone\extensions\EventListenerList
welearn\millstone\extensions\MessageDialog
welearn\millstone\extensions\PerformableAction
welearn\millstone\extensions\PerformableActionButton
welearn\millstone\extensions\PerformableActionException

welearn\millstone\extensions\TextFieldDialog

Hauptklassen: Dieses Package beinhaltet die Hauptklassen des WebDAV-Explorers.

welearn\modules\external\wdavex\Clipboard
welearn\modules\external\wdavex\ExplorerException
welearn\modules\external\wdavex\FileFolderObj
welearn\modules\external\wdavex\InitializationProbe
welearn\modules\external\wdavex\TreeObj
welearn\modules\external\wdavex\WebDAVException
welearn\modules\external\wdavex\WebDAVExplorer
welearn\modules\external\wdavex\WebDAVFile
welearn\modules\external\wdavex\WebDAVFileContainer

Aktionen: Dieses Package beinhaltet alle Operationen, die auf Dateien und Verzeichnisse ausführbar sind.

welearn\modules\external\wdavex\actions\CopyAction
welearn\modules\external\wdavex\actions\CopyPasteAction
welearn\modules\external\wdavex\actions\CutAction
welearn\modules\external\wdavex\actions\CutPasteAction
welearn\modules\external\wdavex\actions\DeleteAction
welearn\modules\external\wdavex\actions\LoginAction
welearn\modules\external\wdavex\actions\NewDirectoryAction
welearn\modules\external\wdavex\actions\PasteAction
welearn\modules\external\wdavex\actions\RefreshAction
welearn\modules\external\wdavex\actions\RenameAction
welearn\modules\external\wdavex\actions\UnzipAction
welearn\modules\external\wdavex\actions\UploadAction

Dialoge zur Interaktion mit dem Benutzer:

welearn\modules\external\wdavex\dialogs\DeleteDialog
welearn\modules\external\wdavex\dialogs\FileReplaceDialog
welearn\modules\external\wdavex\dialogs\LoginDialog
welearn\modules\external\wdavex\dialogs\UploadDialog
welearn\modules\external\wdavex\dialogs\UploadRenameDialog

Internationalisierung: Mit Hilfe dieser Klasse kann die Sprache leicht über eine Textdatei ausgetauscht werden.

welearn\modules\external\wdavex\i18n\Resources

Panels: Dieses Package beinhaltet alle Panels, die im WebDAV-Explorer vorkommen.

welearn\modules\external\wdavex\panels\ActionsPanel
welearn\modules\external\wdavex\panels\BaseExplorerPanel
welearn\modules\external\wdavex\panels\ClipboardPanel
welearn\modules\external\wdavex\panels\DirectoryTreePanel
welearn\modules\external\wdavex\panels\FilePanel
welearn\modules\external\wdavex\panels\MessagesPanel
welearn\modules\external\wdavex\panels\WebDAVFileNode

Diverse Hilfklassen:

welearn\modules\external\wdavex\util\Area
welearn\modules\external\wdavex\util\FilenameValidator
welearn\modules\external\wdavex\util\FileOperation
welearn\modules\external\wdavex\util\LoginInfo
welearn\modules\external\wdavex\util\Refreshable
welearn\modules\external\wdavex\util\Unzipper
welearn\modules\external\wdavex\util\Uploader
welearn\modules\external\wdavex\util\URLUTF8Encoder
welearn\modules\external\wdavex\util\WebDAVFileComparators
welearn\modules\external\wdavex\util\WebDAVURLValidator
welearn\modules\external\wdavex\validators\BasicFilenameValidator
welearn\modules\external\wdavex\validators\UniqueFilenameValidator
welearn\modules\external\wdavex\validators\WebDAVURLValidator

5.3. Verwendung

Folgende Punkte müssen berücksichtigt werden, bevor WebDAV-Explorers genutzt werden kann.

Server: Der WebDAV-Explorer ist ein Java-Servlet und benötigt einen Container um ausgeführt werden zu können. Dafür wird Tomcat [Tomcat] (getestet mit Version 4.1) genutzt. Nach der Installation von Tomcat muss nur ein Link auf das Verzeichnis gesetzt werden, wo sich der WebDAV-Explorer befindet (Server.xml).
Weiters muss das Sun JRE (Java 2 Runtime Environment; getestet mit Version 1.4.2) installiert sein.
Die Logdateien des WebDAV-Explorers werden standardmäßig in dem Verzeichnis „logs“ im Tomcat-Verzeichnis gespeichert. Der Pfad kann über die Log-Konfigurationsdatei „logger.properties“ im WebDAV-Explorer-Verzeichnis geändert werden.

Client: Um den WebDAV-Explorer zu nutzen ist nur ein gängiger Browser nötig (getestet mit Internet Explorer 6, Netscape 7, Mozilla 1.5). Die Installation von Plugins ist nur nötig, wenn spezielle Dateien direkt im Browser angezeigt werden sollten (zum Beispiel pdf-Dokumente). Um den ganzen Umfang des WebDAV-Explorers nutzen zu können, sollte Java-Script aktiviert sein.

5.4. Benutzerhandbuch

In diesem Kapitel wird beschrieben, wie der WeLearn WebDAV-Explorer verwendet werden kann, welche technischen Voraussetzungen nötig sind sowie welche Probleme und Fehler eventuell auftreten können.

5.4.1. Voraussetzungen

Die Voraussetzungen um den WebDAV-Explorer nutzen zu können:

- Browser: Es werden alle neuen Browser unterstützt, ebenso wie ältere Versionen dieser und Textbrowser zum Beispiel für sehbehinderte Personen.
- Zugang zu einem WebDAV-Server (lokal, im LAN oder über das Internet)

5.4.2. Login

Der WebDAV-Explorer bietet Ihnen zwei Möglichkeiten sich an einem WebDAV-Server anzumelden.

Erstens über eine Login-Box (siehe Abb. 5.4.2-1): Bei dieser Methode kann man die Adresse des WebDAV-Servers, Benutzername und Passwort in die dafür vorgesehenen Felder eintragen.

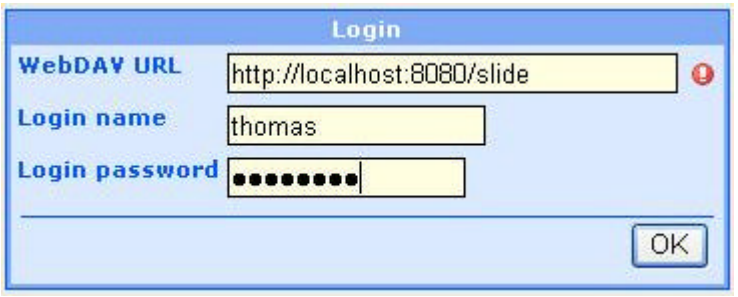

Abbildung 30: Login-Box

Zweitens über Parameter in der URL (siehe Abb. 5.4.2-2): Die WebDAV-Server-URL sowie die Authentifizierungsdaten können als Paramenter in der URL übergeben werden. Dies ist sinnvoll für Bookmarks, wenn man oft den gleichen WebDAV-Server nutzt, aber auch wichtig, wenn andere Module den WebDAV-Explorer nutzen.

Abbildung 31: Login über Paramter

Die zweite Variante birgt natürlich in dieser Form ein Sicherheitsrisiko, da das Passwort im Klartext in der URL steht. Gedacht ist diese Version aber lediglich zur Verwendung aus anderen Modulen heraus.

5.4.3. Der Explorer

Nach dem erfolgreichen Login sieht man links die Treeview (ausgehend vom Wurzelverzeichnis des WebDAV-Servers), rechts davon eine Liste der Dateien und Verzeichnisse des aktuellen Verzeichnisses, darüber eine Liste von Operationen, die auf selektierte Dateien und Verzeichnisse ausgeführt werden können (jene Operationen die nicht durchgeführt werden können, sind grau dargestellt), sowie am unteren Ende den Inhalt des Clipboards (der nach dem Login noch leer ist).

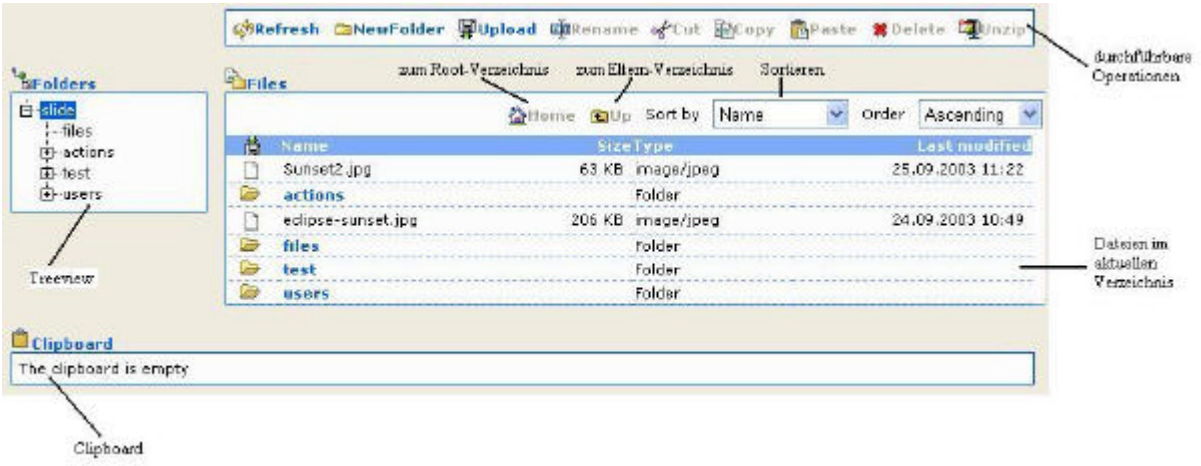

Abbildung 32: WebDAV-Explorer nach dem Login

Treeview: Die Treeview bietet Ihnen eine Übersicht über die Verzeichnisstruktur am WebDAV-Server. Verzeichnisse können auch direkt in der Treeview ausgewählt werden. Deren Inhalt wird dann in der Tabelle angezeigt. Wird ein Verzeichnis entweder in der Treeview oder in der Tabelle ausgewählt, werden automatisch die Kinderverzeichnisse sichtbar, falls welche existieren. Durch einen Klick vor den Verzeichnisnamen kann ein Verzeichnis auch per Hand expandiert oder wieder geschlossen werden.

Filetable: Diese Tabelle zeigt die Dateien und Verzeichnisse des aktuellen Verzeichnisses an. Es können eine Reihe von Operationen in der Tabelle durchgeführt werden:

- Home: Ein Klick auf dieses Icon bringt Sie zur Ansicht des Root-Verzeichnisses.
- Up: Ein Klick auf dieses Icon bringt Sie in der Verzeichnishierarchie um einen Schritt nach oben (zum Eltern-Verzeichnis).
- Sort By: Mit dieser Funktion können Sie die Dateien und Verzeichnisse in der Tabelle sortieren. Es ist möglich nach Dateiname, Größe, Typ und letztem Änderungsdatum auf- oder absteigend zu sortieren.
- Download: Durch einen Klick auf das Icon vor dem Dateinamen kann jede Datei geladen werden. Kann der Browser diese anzeigen (z.B.: Bilder oder pdf-Dokumente), wird die Datei in einem neuen Browserfenster geöffnet. Ansonsten öffnet sich ein Download-Window des jeweiligen Browsers.
- Verzeichnis wechseln: Einerseits über die Funktionen Up und Home, sowie der Treeview. In der Tabelle selbst können Sie nur in Kinderverzeichnisse wechseln, indem sie entweder auf das Icon vor dem Verzeichnis oder dem Verzeichnisnamen selbst klicken.

- Selektieren: Durch einen Klick in eine Zeile der Tabelle wird diese selektiert. Es können mehrere Zeilen gleichzeitig ausgewählt werden um eine Operation auf mehrere Dateien und Verzeichnisse gleichzeitig auszuführen.
- Un-Selektieren: Durch einen erneuten Klick auf eine bereits selektierte Zeile der Tabelle wird diese wieder un-selektiert.

Funktionsleiste: Operationen die Sie mit Hilfe des WebDAV-Explorers durchführen können. Operationen die im Moment nicht verfügbar sind (da diese Operation nicht auf die selektierte(n) Datei(en) oder Verzeichnis(se) durchführbar ist), sind grau und können nicht ausgewählt werden. Folgende Operationen stehen zur Verfügung:

- Refresh: Aktualisiert das Verzeichnis (immer verfügbar)
- New Folder: Anlegen eines neuen Verzeichnisses (immer verfügbar)
- Upload: Hochladen einer lokalen Datei auf den WebDAV-Server in das derzeit aktive Verzeichnis (immer verfügbar)
- Rename: Selektierte Datei oder Verzeichnis umbenennen (verfügbar, sobald mindestens eine Datei oder ein Verzeichnis selektiert ist)
- Cut: Selektierte Datei(en) oder Verzeichnis(se) ausschneiden (verfügbar, sobald mindestens eine Datei oder ein Verzeichnis selektiert ist)
- Copy: Selektierte Datei(en) oder Verzeichnis(se) kopieren (verfügbar, sobald mindestens eine Datei oder ein Verzeichnis selektiert ist)
- Paste: Inhalt des Clipboards einfügen (verfügbar, sobald das Clipboard nicht leer ist, d.h. eine Cut- oder Copy-Operation vorangegangen ist)
- Delete: Selektierte Datei(en) oder Verzeichnis(se) löschen (verfügbar, sobald mindestens eine Datei oder ein Verzeichnis selektiert ist)
- Unzip: Selektierte Zip-Datei(en) entpacken (verfügbar, sobald eine Zip-Datei selektiert wurde)

Clipboard: Im Clipboard werden jene Dateien und Verzeichnisse angezeigt, die ausgeschnitten oder kopiert wurden. Werden Dateien oder Verzeichnisse verschoben, werden sie nach erfolgreicher Beendigung aus dem Clipboard gelöscht. Kopierte Dateien bleiben im Clipboard. Ist das Clipboard nicht leer und werden neue Dateien oder Verzeichnisse kopiert bzw. ausgeschnitten, werden die alten Einträge aus dem Cipboard gelöscht.

Statuszeile: Hier werden Meldungen über durchgeführte Operationen bzw. Fehlermeldungen angezeigt.

5.4.4. Funktionen

Um die Operationen Rename, Cut, Copy, Delete und Unzip durchführen zu können, müssen Sie eine oder mehrere Dateien bzw. Verzeichnisse in der Tabelle auswählen. Refresh, New Folder und Upload sind zu jedem Zeitpunkt verfügbar. Die Operation Unzip ist nur durchführbar, wenn eine oder mehrere Zip-Dateien ausgewählt sind.
Tritt bei einer Operation ein Fehler auf (z.B.: Sie haben nicht die nötigen Rechte diese Operation durchzuführen), werden Sie in der Statuszeile über den Fehler informiert.

Refresh: Mit dieser Funktion können Sie das aktuelle Verzeichnis neu vom WebDAV-Server laden und sich anzeigen lassen. Diese Funktion ist sinnvoll für den Mehrbenutzerbetrieb falls gleichzeitig andere Benutzer im selben Verzeichnis arbeiten.

Abbildung 33: Die Refresh-Funktion

New Folder: Mit dieser Funktion können Sie einen neuen Ordner im aktuellen Verzeichnis anlegen.

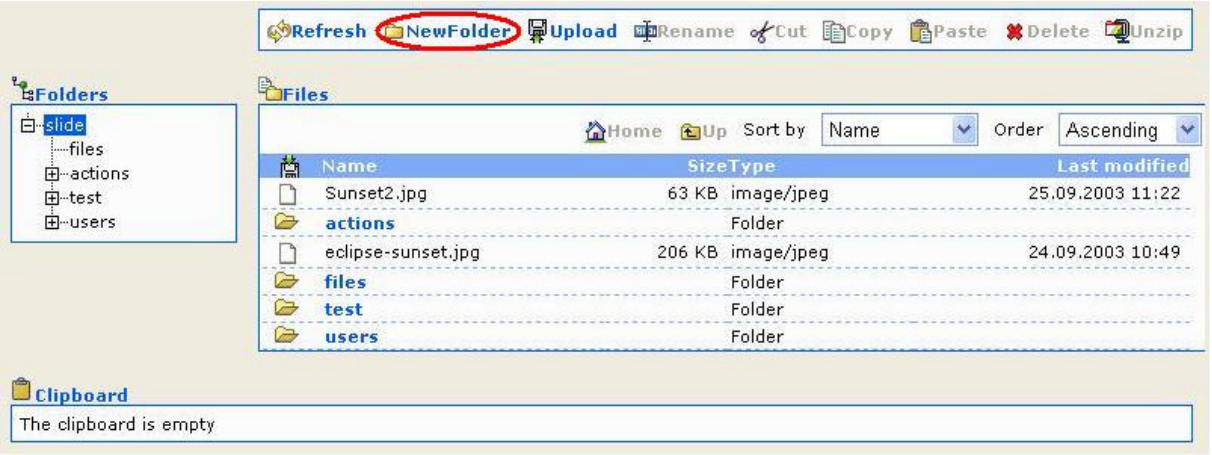

Abbildung 34: Anlegen eines neuen Ordners

Anschließend erscheint eine Box, in der Sie den Namen des neuen Ordners angeben müssen.

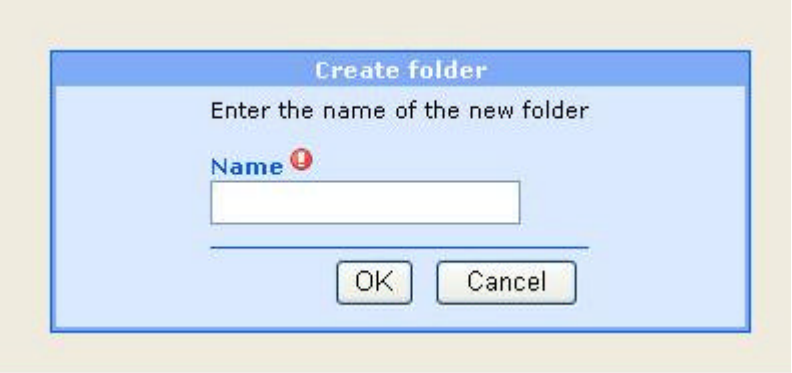

Abbildung 35: Create Folder-Box

Solange sich ein kleines rotes Rufzeichen über dem Textfeld befindet, ist der Eintrag für ein neues Verzeichnis noch nicht korrekt oder ein gleichnamiges Verzeichnis existiert bereits. Durch einen Klick auf das Rufzeichen (Java-Script muss aktiviert sein) erfahren Sie Details zum aufgetretenen Fehler.

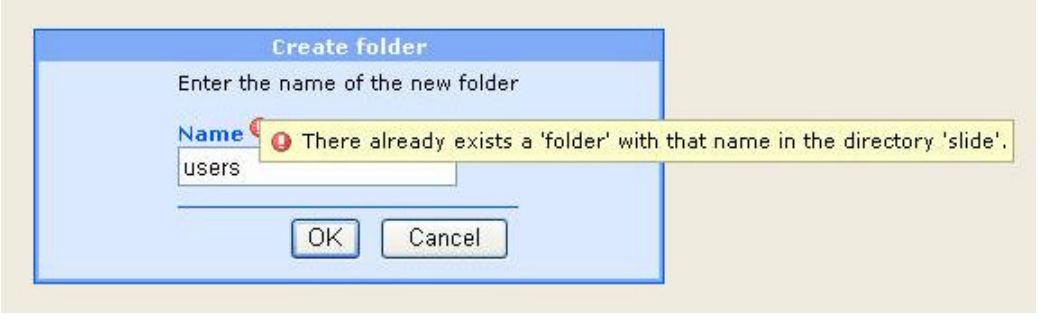

Abbildung 36: Fehler: der angegeben Verzeichnisname existiert bereits

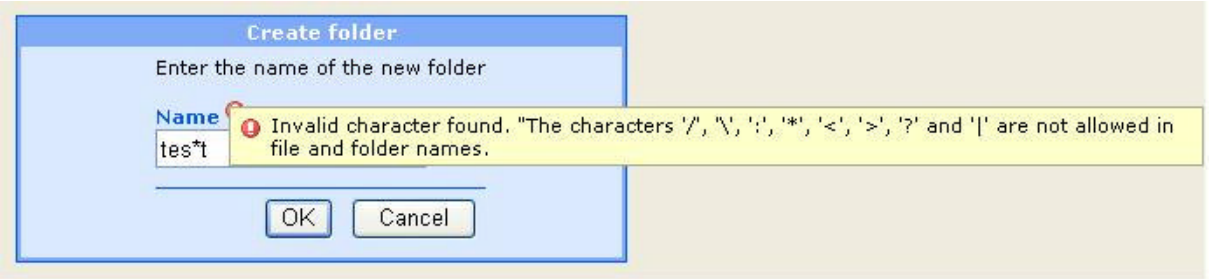

Abbildung 37: Fehler: der angegebene Verzeichnisname enthält nicht erlaubte Zeichen

Durch einen Klick auf den Cancel-Button kann die Operation zu jedem Zeitpunkt abgebrochen werden. Sie kehren dann zum Hauptfenster zurück.

War das Anlegen des Verzeichnisses erfolgreich, kehren Sie zurück zum Hauptfenster und bekommen eine Erfolgsmeldung:

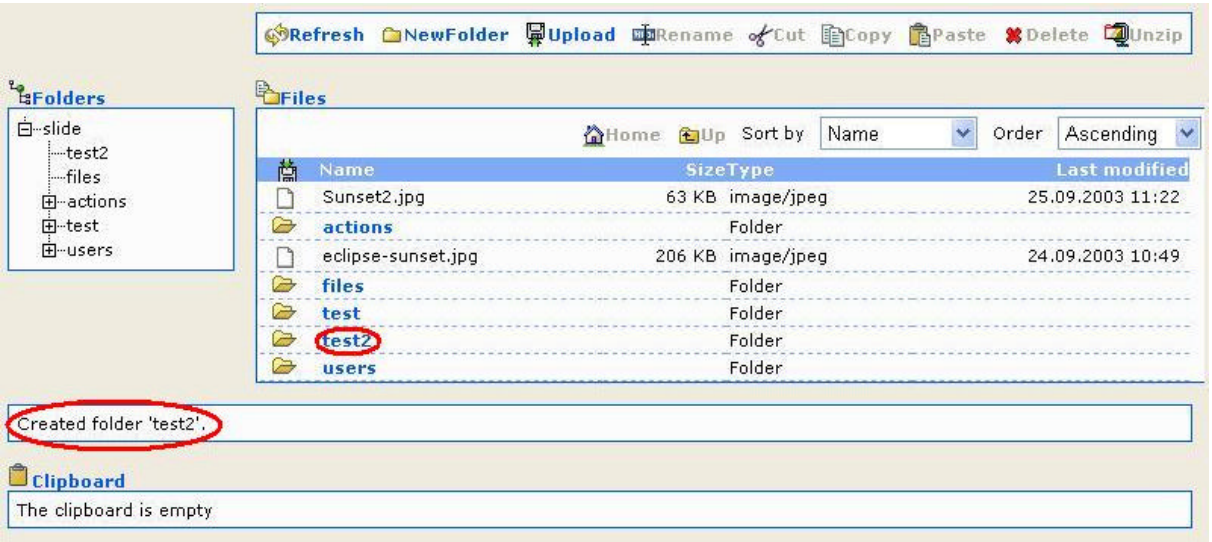

Abbildung 38: erfolgreiches Anlegen eines neuen Ordners

Upload: Mit dieser Funktion können Sie eine lokale Datei (die sich auf dem Rechner, auf dem der Browser läuft, befindet) auf den WebDAV-Server hochladen.

Abbildung 39: Upload einer lokalen Datei in das aktuelle Verzeichnis

Mit Hilfe dieses Upload-Dialogs können Sie die lokale Datei auswählen. Durch einen Klick auf den Button „Durchsuchen…“ öffnet sich ein Dateibrowser des Betriebssystems, mit dem

sie die gewünschte Datei auswählen können. Alternativ können sie auch den Pfad der Datei in das Textfeld eingeben.

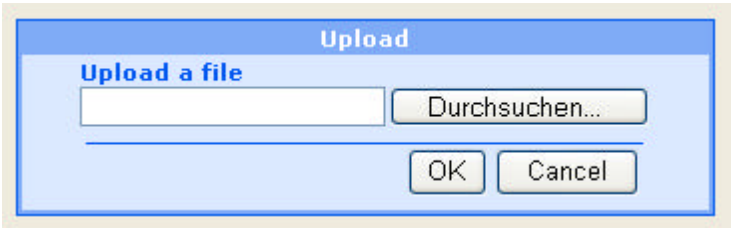

Abbildung 40: Upload-Dialog

Existiert bereits eine gleichnamige Datei am WebDAV-Server werden Sie aufgefordert, die neue Datei umzubenennen oder können die vorhandene Datei überschreiben.

Abbildung 41: Fehler: Datei mit diesem Namen existiert bereits

Nach erfolgreichem Upload der Datei bekommen Sie eine Erfolgsmeldung und die Datei ist in der Tabelle zu sehen.

Abbildung 42: Erfolgsmeldung

Rename: Mit dieser Funktion können Sie eine Datei oder ein Verzeichnis am WebDAV-Server umbenennen.

Abbildung 43: Umbenennen einer Datei oder eines Verzeichnisses

In dem Textfeld dieses Dialogs geben Sie den neuen Namen ein. Das Ausrufezeichen informiert Sie über etwaige Fehler (der Dateiname enthält nicht zulässige Zeichen oder eine gleichnamige Datei existiert bereits).

Abbildung 44: neuen Namen eingeben

Nach erfolgreicher Durchführung der Operation kommen Sie zurück zum Hauptfenster und sehen eine Erfolgsmeldung.

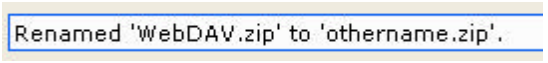

Abbildung 45: Erfolgsmeldung

Clipboardoperationen: Der WebDAV-Explorer visualisiert das Clipboard, sodass Sie immer den aktuellen Inhalt des Clipboards sehen.

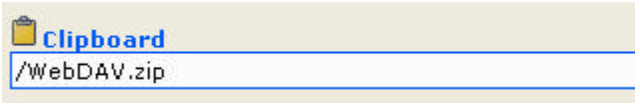

Abbildung 46: Clipboard

Sobald eine oder mehrere Dateien ausgewählt sind, können sie Copy oder Cut anklicken. Paste ist erst aktivierbar wenn sich Dateien im Clipboard befinden.

Abbildung 47: Clipboard-Operationen

- Cut: Mit dieser Operation können Sie Dateien und Verzeichnisse verschieben. Mit der Operation Paste werden diese dann in einem anderen Verzeichnis eingefügt. Die Einträge werden dann aus dem Clipboard gelöscht.

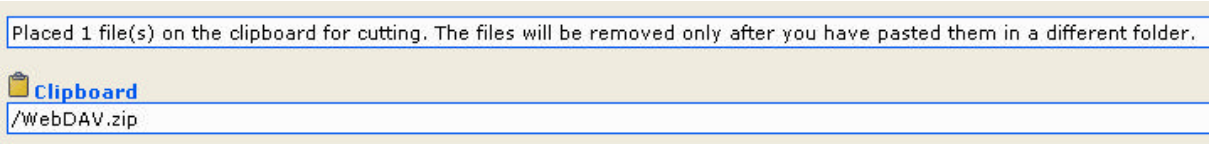

Abbildung 48: Datei wurde ausgeschnitten

Wird nie eine Paste-Operation durchgeführt, bleiben die Dateien im ursprünglichen Verzeichnis.

- Copy: Mit dieser Operation können Sie Dateien und Verzeichnisse kopieren. Mit der Operation Paste werden diese dann in einem anderen Verzeichnis eingefügt. Der Inhalt des Clipboards kann in beliebig viele Verzeichnisse kopiert werden.

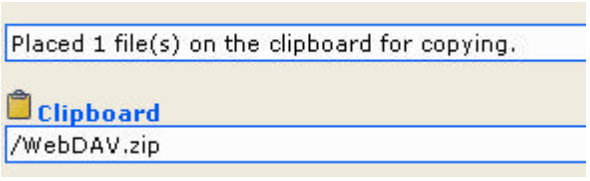

Abbildung 49: Datei wurde ins Clipboard kopiert

- Paste: Diese Operation schließt eine Cut- oder Copyoperation ab. Ist eine Datei mit dem gleichen Namen, wie eine eingefügt/verschobene Datei, bereits vorhanden, kann diese überschrieben werden.

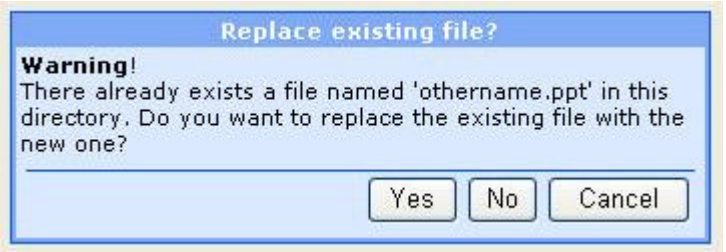

Abbildung 50: Fehler: eine Datei mit diesem Namen existiert bereits

Delete: Mit dieser Funktion können Sie Dateien und Verzeichnisse vom WebDAV-Server löschen. Wird ein Verzeichnis gelöscht, wird automatisch auch der gesamte Inhalt gelöscht.

Abbildung 51: Löschen von Dateien und Verzeichnissen

Sie müssen das Löschen der ausgewählten Datei(en) bestätigen.

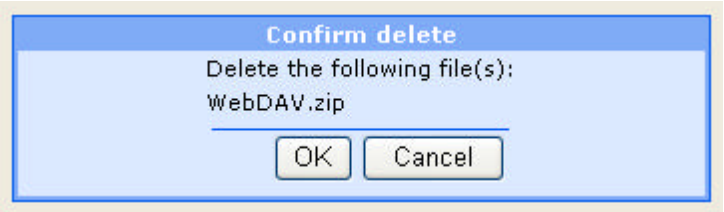

Abbildung 52: Löschen bestätigen

Nach dem Durchführen der Operation bekommen Sie in der Statuszeile eine Rückmeldung wie viele Dateien und Verzeichnisse gelöscht wurden.

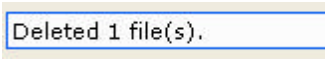

Abbildung 53: Erfolgsmeldung

Unzip: Mit dieser Funktion können Sie Zip-Dateien am WebDAV-Server entpacken.

Abbildung 54: Entpacken von Zip-Dateien

In einem Dialog können Sie einen Verzeichnisnamen angeben, in den die Zip-Datei entpackt werden sollte.

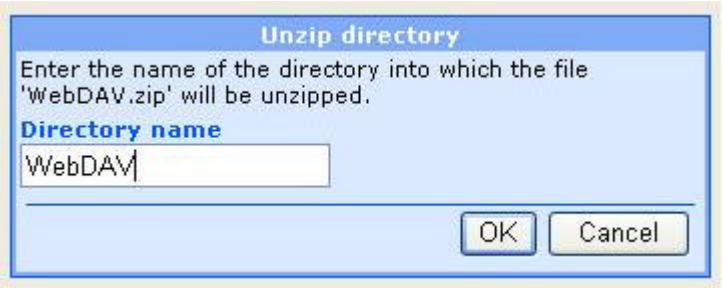

Abbildung 55: Verzeichnis zum Entpacken angeben

Enthält die Zip-Datei eine Verzeichnisstruktur, wird diese beim Entpacken ebenfalls nachgebildet.

5.4.5. Probleme

Bekannte Probleme, die bei der Benutzung des WebDAV-Explorers auftreten können:

- Tomcat Out of Memory Exception: Wird ein Verzeichnis mit sehr vielen Dateien (500+) geladen, wirft Tomcat oft eine Out of Memory Exception. Der Grund des Fehlers ist bei Tomcat zu suchen, startet man Tomcat mit dem Parameter „–Xmx…m“ (… steht für eine beliebige Megabyte-Anzahl, zum Beispiel 512) und gibt ausreichend viel Speicher an, tritt dieses Problem nicht mehr auf. Der verwendete Parameter ist eine non-standard Erweiterung der Sun Java VM.
- Millstone Bilder Problem: Manchmal werden eingebundene Bilder (Icons) falsch angezeigt. Dies liegt daran, dass Millstone alle Ressourcen intern durchnummiert bevor es den HTML-Code erzeugt, den es zum Client schickt. Da sich Bilder im Tomcat-Cache befinden können, werden manchmal falsche oder gar keine Bilder angezeigt. Ein Neustart von Tomcat behebt das Problem.

6. Fazit

Das WebDAV-Protokoll hat bereits in alle modernen Betriebssysteme Einzug genommen, es ist daher nur logisch seine Funktionen auch für das eLearning zu nutzen.
Es wird versucht bei WeLearn immer die neuesten Technologien und Standards zu verwenden, die es sowohl dem Lernenden als auch dem Lehrenden erleichtern diese Lernplattform optimal zu nutzen. Anstatt viel Zeit in das Erlernen einer Lernplattform zu investieren bietet WeLearn nach dem Motto „keep it simple“ den Benutzern einerseits eine mächtige Lernplattform, andererseits eine einfach bedienbare Umgebung um sich auf das Wesentlich konzentrieren zu können: lehren und lernen. An diese Philosophie knüpft auch der WeLearn WebDAV-Explorer an, indem er dem Benutzer einen Explorer mit dem gewohnten Look & Feel aus gängigen Betriebssystemen bietet. Der Benutzer merkt nicht, dass er auf einem WebDAV-Server arbeitet, er bedient ihn wie einen Explorer auf seinem lokalen Dateisystem. Trotzdem werden alle Funktionen (wie etwa Locking und Versioning), die der WebDAV-Server bietet genutzt.

In dieser Arbeit wurden zuerst grundlegende Informationen über Fernunterricht, eLearning und Lernplattformen gegeben. Dies sollte den Leser motivieren sich eingehender mit dieser neuen Unterrichtsform zu beschäftigen. Anschließend wurde der Lernplattform WeLearn ein Kapitel gewidmet um die Eigenschaften und Vorzüge dieser vorzustellen. Außerdem war es wichtig sich eingehend mit WeLearn zu beschäftigen um ein Modul dafür erstellen zu können.
Weiters geben die aufgeführten Dateisysteme einerseits einen geschichtlichen Rückblick auf die Entwicklung und andererseits einen Überblick und Vergleich der heute verwendeten Dateisysteme in modernen Betriebssystemen.
Die Begriffe Netzwerkdateisystemen und Protokollen werden oft synonym benutzt, so ist auch WebDAV für viele Experten mehr als nur ein Protokoll, sondern wird auch als Dateisystem gesehen. Welche Ziele die Entwickler bei der Standardisierung von WebDAV verfolgten, welche Eigenschaften und Funktionen dieses Protokoll hat und vor allem wie viele verschiedenen Designansätze WebDAV vereint, wurde im darauf folgenden Kapitel genau erläutert. Weiters wurden einige WebDAV Applikationen, WebDAV Server und WebDAV Clients evaluiert. Der Hauptteil befasste sich mit den Interna des Protokolls, angefangen von den Datenstrukturen bis hin zu den einzelnen Funktionen, dem http-Request-Response-Flow sowie der Fehlerbehandlung.

Schlussendlich folgte die Dokumentation des WebDAV-Explorers, wobei auf die Designziele, die Architektur und Verwendung eingegangen wurde. Ein ausführliches Benutzerhandbuch rundete diese Arbeit ab.

Es hat sich gezeigt, dass WebDAV eLearning vor allem durch seine kollaborativen Eigenschaften, sowie der Sicherheitsfunktionen (Locking, Versioning) unterstützen kann. Durch die Weiterentwicklung von WebDAV – vor allem im Bereich der Suche (DASL) – wird WebDAV auch weiterhin immer mehr Bedeutung im Bereich des eLearnings und der Lernplattformen bekommen.

Die Erstellung dieser Arbeit war für mich inhaltlich sehr interessant, da ich mich in drei großen Bereichen bewegte: Erstens das große Gebiet des Fernunterrichts mit den für die Arbeit relevanten Teilen über eLearning, mLearning und Lernplattformen. Zweitens die Auseinandersetzung mit verschiedenen Dateisystemen, deren Eigenschaften sowie Vor- und Nachteile. Drittens ein genaues Studium des WebDAV-Protokolls inklusive all den technischen Details. Aber viel mehr als das theoretische Studium von verschiedenen Fachgebieten hat die praktische Mitarbeit an einem großen Projekt wie WeLearn meinen Horizont erweitert.

Ziel dieser Arbeit war es zu argumentieren, warum das WebDAV-Protokoll eLearning unterstützen kann und diese Eigenschaften im WeLearn WebDAV-Exploerer zu implementieren. Dabei entstanden drei Hauptpunkte, die diese Arbeit für Leser interessant machen:

- WebDAV Übersetzung: Es gibt einige Kurzbeschreibungen des WebDAV-Protokolls in deutscher Sprache, jedoch ist das in dieser Arbeit enthaltene Kapitel das einzige mir bekannte in diesem Detailierungsgrad.
- WeLearn-Modul: Im praktischen Teil dieser Arbeit ist der WebDAV-Explorer entstanden, der hauptsächlich als Modul für die nächste Version der eLearning-Plattform WeLearn entwickelt wurde, aber auch als standalone Webanwendung benutzt werden kann, um Operationen auf WebDAV-Server durchzuführen.
- Benutzerhandbuch: Mit dem 5. Kapitel entstand ein detailiertes Handbuch für die Nutzung des WeLearn WebDAV-Explorers, das alle durchführbaren Operationen sowie eventuell auftretende Fehler behandelt.

7. Literatur

[ANT] The Apache ANT Project; Link: http://ant.apache.org/ (2004-01-21)

[Apache] The Apache Software Foundation; Link: http://www.apache.org/ (2004-03-15)

[Betriebssysteme_Adam] Jetzt lerne ich endlich… Betriebssysteme; Hans-Joachim Adam; 2003; Link: http://www.informationstechnikadam.de/inft/skripte/Betriebssysteme%20Adam.pdf (2004-01-21)

[Blackboard] Bb – Blackboard; Link: http://www.blackboard.com (2004-03-15)

[bm:bwk] Bundesministerium für Bildung, Wissenschaft und Kultur (bm:bwk); Link: http://www.bildung.at (2004-03-15)

[Coda_FS] Coda File System; Link: http://www.coda.cs.cmu.edu/index.html (2004-01-21)

[Commons_VFS] Commons Virtual File System; Link: http://jakarta.apache.org/commons/sandbox/vfs/index.html (2004-01-21)

[DASL] DAV Searching and Locating; Link: http://www.webdav.org/dasl/dasl-charter.html (2004-01-21)

[DASL_Draft] Julian Reschke's latest working draft; 2002; Link: http://greenbytes.de/tech/webdav/draft-reschke-webdav-search-latest.html (2004-01-21)

[Dateiverwaltung_Heiss] Dateiverwaltung; H.-U. Heiß, TU Berlin; Link: http://kbs.cs.tu-berlin.de/teaching/ws2001/bs/bs10.pdf (2004-01-21)

[DAV_FAQ] DAV Frequently Asked Questions; Greg Stein; 2000; Link: http://www.webdav.org/other/faq.html (2004-01-21)

[Dissertation_Reisinger] Mobile intelligente Agenten als Wegweiser im Distance Teaching/Coaching/Learning; Dissertation; Susanne Reisinger; 2002; Link: http://www.fim.uni-linz.ac.at/Diplomarbeiten/dissertation_reisinger/Inhalt/dissertation_reisinger.zip

[eLearning_Austria] Verein eLearning-Austria; Link: http://www.blended-elearning.at/ (1004-03-15)

[eLearning_Def] Telelearning - Definition, Bedeutung, Erklärung im Lexikon; Link: http://www.net-lexikon.de/Telelearning.html (2004-03-15)

[EML] Educational Modeling Language; Link: http://eml.ou.nl (2004-03-16)

[EXT_Setuid] Setuid, Setgid und Sticky Bit; Bernhard Winkler; 1995; Link: http://www.rz.uni-bayreuth.de/lehre/unix_rz/vorlesung/dateibaum/setuid.html (2004-01-21)

[EXT2_Card] Design and Implementation of the Second Extended Filesystem in First Dutch International Symposium on Linux, ISBN 90-367-0385-9; Rémy Card, Theodore Ts'o, Stephen Tweedie;

[EXT3_Expl] Exploring the ext3 Filesystem; Bill von Hagen; 2002; Link: http://www.linuxplanet.com/linuxplanet/reports/4136/5/ (2004-01-21)

[EXT3_Tweedie_00] EXT3, Journaling Filesystem; Dr. Stephen Tweedie; 2000; Link: http://olstrans.sourceforge.net/release/OLS2000-ext3/OLS2000-ext3.pdf (2004-01-21)

[Fernunterricht_Def] Landesportal Sachsen-Anhalt – Fernunterricht; Link: http://www.sachsen-anhalt.de/rcs/LSA/pub/Ch4/fld5rtwrmy1xr/pgic4w4ibfkd/index.jsp (2004-03-15)

[Fernunterricht_Gesetz] Fernunterrichtsschutzgesetz und Weiterbildung; Diana Kiesecker; 2002; Katholische Universität Eichstätt-Ingolstadt; Link: http://www1.ku-eichstaett.de/PPF/FGPaed/Basar/Handout%20fernunterrichtsschutzgesetz.PDF (2004-03-10)

[Fernunterricht_Ratgeber] Fernunterricht Ratgeber; Link: http://www.zfu.de/Download/Ratgeber%201%202002.pdf (2004-03-15)

[FIM] FIM – Institute for Information Processing and Microprocessor Technology; Johannes Kepler University of Linz, Austria; Link: http://www.fim.uni-linz.ac.at (2004-03-16)

[FS_Perf_Bryant] Filesystem Performance and Scalability in Linux 2.4.17; Ray Bryant, Ruth Forester, John Hawkes; 2002; Link: http://oss.sgi.com/projects/xfs/papers/filesystem-perf-tm.pdf (2004-01-21)

[GNU] The GNU Project; Link: http://www.gnu.org (2004-03-16)

[Goliath] Goliath; Link: http://www.webdav.org/goliath/ (2004-01-21)

[HFS+] HFS+ Dateisystems; Link: http://www.at-mix.de/hfs.htm (2004-01-21)

[HPFS] High Peformance File System; Link: http://www.gbt.ch/_forum/0000019e.htm (2004-01-21)

[i18n] W3C Internationalization Activity; Link: http://www.w3.org/International/ (2004-03-15)

[iDisk] Mac OS X iDisk; Apple; Link: http://www.apple.com/de/macosx/features/idisk/ (2004-01-21)

[IEEE_LTSC] IEEE LTSC; Link: http://ltsc.ieee.org (2004-03-16)

[IIS] Internet Information Services Features; Microsoft; 1999; Link: http://www.microsoft.com/windows2000/server/evaluation/features/web.asp (2004-01-21)

[IMS] IMS Global Learning Consortium, Inc.; Link: http://www.imsglobal.org (2004-03-16)

[Informatik-Handbuch_99] Informatik-Handbuch / hrsg. von Peter Rechenberg und Gustav Pomberger. - 2., aktualisierte und erw. Aufl. - München ; Wien : Hanser, 1999

ISBN 3-446-19601-3

[Inode] Das I-Node System; F. Kalhammer; 2001; Link: http://www.linux-praxis.de/linux1/filesystem3.html (2004-01-21)

[Jakarta] The Apache Jakarta Project; Link: http://jakarta.apache.org/ (2004-01-23)

[java.io.File] Sun JAVA API - java.io.File; Link: http://java.sun.com/j2se/1.4.2/docs/api/java/io/File.html (2004-01-21)

[JFFS] JFFS: The Journalling Flash File System; David Woodhouse; Link http://sources.redhat.com/jffs2/jffs2.pdf (2004-01-21)

[JFFS2] The Journalling Flash File System, Version 2; David Woodhouse; 2003; Link: http://sources.redhat.com/jffs2/ (2004-01-21)

[JFS] Journaled File System; Steve Best (IBM); 2000; Link: ftp://www6.software.ibm.com/software/developer/library/jfs.pdf (2004-01-21)

[JKU] Johannes Kepler Universität (JKU) Linz; Altenberger Str. 69, A-4040 Linz, Austria; Link: http://www.jku.at (2004-03-16)

[Linux_FS_Vergleich] Scalability and Performance in Modern Filesystems; Philip Trautman (SGI); 2000; Link: http://www.sgi.com/pdfs/2668.pdf (2004-01-21)

[Legal_Eng_Sonntag] Legal Engineering; Michael Sonntag; Link: http://www.fim.uni-linz.ac.at/Publications/Sonntag/Legal_Engineering.pdf (2004-03-15)

[Linux_Wegweiser_00] Linux - Wegweiser zur Installation & Konfiguration, 3. Auflage (Online Version); Matt Welsh, Matthias Kalle Dalheimer, Lar Kaufmann; 2000; O'Reilly Verlag GmbH & Co.KG; ISBN 3-89721-133-5 Link: http://www.oreilly.de/german/freebooks/rlinux3ger/linux_wegIVZ.html (2004-01-21)

[LinuxWiki.org] LinuxWiki; Link: http://www.linuxwiki.org/ (2004-01-21)

[Log4j] Log4j Project; Link: http://logging.apache.org/log4j/docs/index.html (2004-03-15)

[LUFS] LUFS; Link: http://lufs.sourceforge.net/lufs/intro.html (2004-01-21)

[Lustre] Lustre; Link: http://www.lustre.org/ (2004-01-21)

[Millstone] Millstone (IT Mill); Link: http://www.millstone.org/ (2004-03-15)

[mobilearn.at] MobiLearn: Mobile Learning: Medieninformatik - Any-Time Any-Where; Link: http://www.mobilearn.at (2004-03-15)

[mobilearn.org] MOBIlern: The wings of learning; Link: http://www.mobilearn.org (2004-03-15)

[mLearning] mLearning Consortium; Link: http://www.mcgrawhill.ca/college/mlearning/ (2004-03-15)

[NTFS.com] NTFS.com NTFS File System General Information; Link: http://www.ntfs.com/ (2004-01-21)

[OCFS] Oracle Cluster File System; Link: http://oss.oracle.com/projects/ocfs/ (2004-01-21)

[ODL] Open Distance Learning - Definition, Bedeutung, Erklärung im Lexikon; Link: http://www.net-lexikon.de/ODL.html (2004-03-15)

[OpenGFS] The OpenGFS Project; Link: http://opengfs.sourceforge.net/index.php (2004-01-21)

[Progtechnik_Groeller] IT/PT: Programmiertechnik und theoretische Grundlagen (Teil II); Eduard Gröller, TU Wien; 2003

[RFC_2291] HTTP Requirements for a Distributed Authoring and Versioning Protocol for the World Wide Web; Network Working Group; 1998; Link: http://www.ietf.org/rfc/rfc2291.txt (2004-01-21)

[RFC_2518] HTTP Extensions for Distributed Authoring -- WEBDAV; Network Working Group; 1999; Link: http://www.ietf.org/rfc/rfc2518.txt (2004-01-21)

[RFC_2616] Hypertext Transfer Protocol -- HTTP/1.1; Network Working Group; 1999; Link: http://www.ietf.org/rfc/rfc2616.txt (2004-01-21)

[RFC_3253] Versioning Extensions to WebDAV (Web Distributed Authoring and Versioning); Network Working Group; 2002; Link: http://www.ietf.org/rfc/rfc3253.txt (2004-01-21)

[SCORM] ADL SCORM; Link: http://www.adlnet.org (2004-03-16)

[Slide] Jakarta Slide; Apache Software Foundation; Link: http://jakarta.apache.org/slide/ (2004-01-21)

[SQUASHFS] SQUASHFS – A compressed FS for Linux; Link: http://sourceforge.net/projects/squashfs/ (2004-01-21)

[Tomcat] Apache Tomcat; Link: http://jakarta.apache.org/tomcat/index.html (2004-01-21)

[WebCT] WebCT; Link: http://www.webct.com (2004-03-15)

[WebDAV.org] WebDAV Resources; Link: http://www.webdav.org/ (2004-01-21)

[WebDAV_ACL] WebDAV Access Control Protocol; Link: http://www.webdav.org/acl/ (2004-01-21)

[WebDAV_ACL_Draft] Access Control Extensions to WebDAV Draft 13; Link: http://www.ietf.org/internet-drafts/draft-ietf-webdav-acl-13.txt (2004-01-21)

[WebDAV_Intro] WEBDAV: IETF Standard for Collaborative Authoring on the Web; E. James Whitehead, Jr., Meredith Wigg; IEEE Internet Computing, Vol. 2, No. 5, September/October, 1998, Seite 34-40; Link: http://www.ics.uci.edu/pub/ietf/webdav/intro/webdav_intro.pdf (2004-01-21)

[WebDAV_Intro_Pr] The Web as a Collaborative, Writeable Medium - An Introduction to the IETF WebDAV Standard; Jim Whitehead; Link: http://www.cs.unibo.it/~fabio/webdav/WebDAV.pdf (2004-01-21)

[WebDAV_Versioning_Pr] WebDAV und subversion – Weg von der Web-Einbahnstrasse; /ch/open; 2002; Link: http://www.ch-open.ch/html/events/2002/chopen-webdav.pdf (2004-01-21)

[WebDAV_Wiki] WebDAV aus Wikipedia, der freien Enzyklopädie; Link: http://de.wikipedia.org/wiki/WebDAV (2004-01-21)

[WebDrive] WebDrive - FTP, WebDAV and Internet File Management Software; South River Technologies; Link: http://www.webdrive.com/index.html (2004-01-21)

[WeLearn] Web Environment for Learning; Link: http://www.fim.uni-linz.ac.at/research/welearn/ (2004-03-15)

[WeLearn_Framework] The WeLearn Distance Teaching Framework; Roman Divotkey, Jörg R. Mühlbacher, Susanne Reisinger, Doris Remplbauer; 2002; Link: http://www.fim.uni-linz.ac.at/research/DistanceEducation/publications/The_WeLearn_Distance_Teaching_Framework.pdf (2004-03-15)

[WSLT] CEN/ISSS WSLT; Link: http://www.cenorm.be/isss/Workshop/lt/ (2003-03-16)

[XFS] Open Source XFS for Linux; SGI (Silicon Graphics Inc.); 2000; Link: http://www.sgi.com/pdfs/2508.pdf (2004-01-21)

[XFS_for_Irix] XFS for Irix; SGI (Silicon Graphics Inc.); 1999; Link: http://www.sgi.com/pdfs/xfs_irix.pdf (2004-01-21)

[Zope] Zope; Link: http://www.zope.org (2004-01-21)

[Zope_WebDAV] Zope Builds on its Success - A second look at Digital Creations' open-source application server now that it supports WebDAV; Cameron Laird, Kathryn Soraiz 1999; Link: http://webserver.cpg.com/wa/4.4/ (2004-01-21)

8. Abbildungsverzeichnis

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Magisterarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Linz, 29. März 2004 _____

Lebenslauf

Thomas Göbl
Thürheimerstr. 27
4020 Linz
Tel.: 0650/6706755
e-mail: thomasg@aon.at

Schulbildung:

09/91 - 06/99	Bundesrealgymnasium Landwiedstraße, 4020 Linz Matura im Juni 1999

Studium:

10/99 - dato	Informatik an der JKU Linz
04/03 - dato	Diplomarbeit am Institut für Informationsverarbeitung und Mikroprozessortechnik (FIM), Betreuer: Mag. iur. Dipl.-Ing. Dr. Michael Sonntag
02/04	Kurzstudium in Thailand im Rahmen des ASEA-UNINET Programms

Ferialpraxis, Studienbegleitende Tätigkeiten:

07/97	Ferialpraxis bei Druckerei und Zeitungshaus J. Wimmer GmbH, Oberösterreich Online, Linz, Internetdienst
08/97	Ferialpraxis bei Allg. SPK OÖ Bank AG, Geschäftsstelle Keferfeld, Linz
07/98 - 09/02	sporadisch Ferialpraxis bei Holub, Steiner + Partner GmbH, Fundraising für das Rote Kreuz in Deutschland und Österreich
08/99	Ferialpraxis bei Allg. SPK OÖ Bank AG, Geschäftsstelle Keferfeld, Linz
03/02 - 06/02	Tutor am Institut für Praktische Informatik für die LVA Embedded Systems, Universität Linz
10/02 - 01/03	Tutor am Institut für Praktische Informatik für die LVA Telekooperation, Universität Linz
03/03 - 06/03	Tutor am Institut für Praktische Informatik für die LVA Embedded Systems, Universität Linz
03/04 - dato	Tutor am Institut für Praktische Informatik für die LVA Embedded Systems, Universität Linz

Berufspraxis:

03/03 – 09/03	Mitarbeiter bei dem Projekt „MobiLearn“ am Institut für Praktische Informatik (SOFT), Projektleiter: o.Univ-Prof. Mag. Dr. Alois Ferscha
04/03 - dato	Mitarbeiter im Rahmen meiner Diplomarbeit bei dem Projekt „WeLearn“ am Institut für Informationsverarbeitung und

Mikroprozessortechnik (FIM), Projektleiter: o.Univ-Prof. Dr. Jörg R. Mühlbacher

Zusatzqualifikationen:

Sprachen:	Englisch in Wort und Schrift
	Französisch Maturaniveau

Interessen:

Sport:	Schwimmen und Badminton
Sonstiges:	Musik, Film und Computer

Persönliche Daten:
Geboren am 09.04.1981 in Linz, ledig

Linz, 14.03.2004