Draft

Realisierung einer Web-Anwendung für Geographische Daten

Scriptum SS 2014

Andrew U. Frank

Andrew U. Frank
frank@geoinfo.tuwien.ac.at
Institut für Geodäsie und Geoinformation
Technische Universität Wien

October 22, 2014

Überarbeiteter Entwurf (0.4.3) SS 2014 gz 5358 v4854

Die Beispiele, die ich zur Illustration während dem Vortrag benutzen werde sind ein wichtiger Teil des Kurses. Sie fehlen im Skript noch fast vollständig.

Inhaltsverzeichnis

I Einleitung 5

1 Gliederung der Vorlesung 11

2 Darstellung einfacher räumlicher Daten 17

II Grundlagen für einfache Webapplikationen 45

3 Das Internet als Grundlage für das World Wide Web 47

4 Das Web und die dafür nötige Basis Software 65

5 Eine einfachen Web-App auf dem Webserver 73

III Funktionen in Webseiten einbauen 81

6 Sprachen und Protokolle 85

7 Das Einbinden von Daten in eine Applikation 99

8 Einbinden von Karten 107

IV Alternativen zu den üblichen Web Architekturen 115

9 NOSQL-Datenbanken für das Web 119

10 Linked Data und Semantic Web 125

V Schluss 129

11 Datenaustausch: Sicherheit der Daten 131

12 Rechtsfragen 141

13 Literaturverzeichnis 147

Teil I

Einleitung

In dieser Vorlesung realisieren die Studenten eine Web Applikation für GIS. Im vorangehenden Semester wurde eine Idee für eine GIS Anwendung zu einer wirtschaftlich aussichtsreichen Projektidee konkretisiert. Die wirtschaftliche und technische Machbarkeit wurde positiv abgeklärt. Nun soll das Projekt technisch realisiert werden, das heißt, die Web Anwendung für GIS soll gebaut werden.

Ziel der Vorlesung und Übung ist es, dass jeder Student in einer kleinen Gruppe (typisch 2 Studenten) die Realisierung des GIS Projektes miterlebt und alle Schritte im Detail verfolgt. Auf die Frage eines zukünftigen Arbeitgebers nach dem Wissen über GIS Web Applikationen soll der Student nicht nur sagen „haben wir (theoretisch) gehabt“, sondern „habe ich (praktisch) gemacht“.

Die Projekte sind bewusst einfach zu halten. Es sind im Allgemeinen Anwendungen, bei denen der Kunde ein bezüglich seiner Standortes und seiner Präferenzen ein Objekt oder eine Dienstleistung auswählt oder spezielle Informationen erhält. Ich erwarte sehr viel Wachstum im Geoinformationsmarkt mit solchen „kleineren“, einfacheren Anwendungen, die konkrete Fragen für eine größere Gruppe von Nutzern beantworten. Zum Beispiel nutze ich eine einfache, auf ein sehr beschränktes Bedürfnis ausgerichtete, Web Anwendung für GIS auf meinem Mobiltelefon überraschend häufig: das kleine Programm bestimmt aus meiner momentanen Position, die durch GPS erfasst ist, die Haltestellen der öffentlichen Verkehrsmittel, die am nächsten liegen und gibt mir Auskunft über die nächsten Abfahrtszeiten und die Ziele, die von dort aus erreichbar sind (Abb.1 und Abb. I). Dieses Programm auf dem Mobiltelefon ist einfach zu bedienen und gibt rasch eine Antwort auf eine konkrete Frage in einer Entscheidungssituation, in der ich mich häufig befinde. Manche ähnlichen Anwendungen werden in den nächsten Jahren entstehen.

Die Gestaltung großer, auf komplexen Modellen aufbauende GIS Anwendungen mit aufwendiger räumlicher und statistischer Analyse wird in späteren Kursen des Studiengangs behandelt; aber auch solche GIS Projekte brauchen heute meist, als Benutzerinterface, die gleichen Methoden, die hier diskutiert werden. Das hier Gelernte lässt sich also später weiter verwenden und ausbauen. Andere Anwendungen sammeln Daten von Freiwilligen und zeigen sie im Web (Abb. 3).

Die Aufgabe einer Web Anwendung für GIS in einem Semester zu realisieren wäre vor wenigen Jahren noch undenkbar gewesen. Heute bietet Google mit Fusion Tables einen Dienst an, der alle einfachen Aufgaben der Visualisierung von Informationen mit einem einfachen Bezug zum Straßenraum ohne besondere Kenntnisse erlaubt. Im 2. Kapitel untersuchen wir diesen Dienst und prüfen, welche Aufgaben er übernehmen kann. Was kann “Jedermann” (oder “Jederfrau”) mit grundlegenden Kenntnissen von Tabellenkalkulationsprogrammen

Das Ziel ist PRAXIS zu vermitteln!

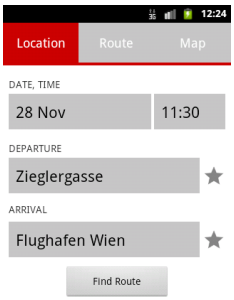


Abbildung 1: Eine Eingabe für eine einfache GIS Anwendung für Benutzer der öffentlichen Verkehrsmittel

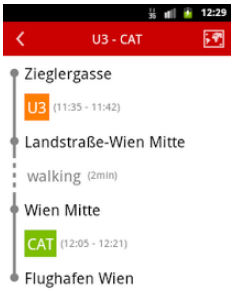


Abbildung 2: Die Ausgabe der einfachen GIS Anwendung für Benutzer der öffentlichen Verkehrsmittel Mobilephone oder Smartphone ist zu einem portablen, überall mitgeführten Web-Zugang (d.h. Web Browser) geworden.

Volunteered Geographic Information oder crowd sourcing

erreichen?

In den nachfolgenden Teilen geht es dann darum, den Studenten zu zeigen, welche verbesserten Möglichkeiten bestehen, komplexere Anwendungen zu realisieren. Die dazu notwendigen technischen Kenntnisse zeichnen Ingenieure mit Geoinformationsausbildung aus und bilden deren professionelle Grundlage.

Abbildung 3: Schlaglöcher in Madrid - eine Anwendung von Fusion Tables

Google Fusion Tables = GIS für Jedermann?

Dabei wird im ersten Teil (Kapitel 2.1 bis Kapitel 2.1.18) die Aufgabe detailliert und eine grobe Übersicht über Web Technologie gegeben. Im zweiten Teil (Kapitel 3 bis Kapitel 5)werden die Grundlagen für eine einfache Webapplikation gelegt und gezeigt, wie diese auf einfachste Weise, das heisst als statische HTML Seiten, realisiert werden kann. Der dritte Teil (Kapitel 6 bis7) führt weitere Teile der Web Technologie ein, die für das Einbinden von Benutzerdaten und geographischen Daten notwendig sind und flexible, rasch reagierende Anwendungen zu bauen. Die beiden letzten Kapitel dieses Teils zeigen jeweils praktische Beispiele, wie vorgegangen werden muss.

Mit den heutigen Hilfsmitteln, besonders in Verbindungen mit der Open Source Software Linux, Apache Web Server, MySQL Datenbank und PHP als Abfragemethodik (zusammen als LAMP Programme bekannt), ist es inzwischen relativ leicht möglich komplexe geographische Anwendungen als Webapplikationen aufzubauen. Räumliche Information wird meist durch Zugriff auf Google Map (8.2 oder alternative, ähnliche Dienstleistungen, z.B. Open Street Map 2) erbracht. Es werden auch neuere Technologien kurz vorgestellt: Datenspeicherung mit NOSQL Datenbanken (Beispiel: couchDB) und Linked Data, das sind mit RDF strukturierte Daten, die mit SPARQL abgefragt werden können.

Raumbezug durch Google Maps, Open Streetmap oder Bing Maps

Der vierte Teil (Kapitel 9 bis 12) diskutiert zuerst zwei neue, alternative Methoden, wie Web Applikationen gebaut werden können. Zum Schluss wird auf überlegungen zur Sicherheit und Rechtsfragen, die für die Erstellung von Web Anwendungen wichtig sind, hingewiesen,

Die Studenten sollen jedes dieser Hilfsmittel einmal anfassen und die grundlegende Funktionsweise erfahren. Damit erkennt der Student, welchen Teil der Aufgabe er mit welchen Hilfsmitteln erledigen muss und sieht, wie die Schnittstellen dazwischen zu organisieren sind. Leider ist es nicht möglich, die angesprochene praktische Situation in einem abstrakten Unterricht zu vermitteln und zu erlernen, sondern es ist nötig, dass die Studenten sich mit den tatsächlich benutzten Tools mit deren Einschränkungen, Besonderheiten und Fehlern auseinandersetzt. Nicht zuletzt wird dabei die wertvolle Fähigkeit erworben, mit den vorhandenen Mitteln wirkliche Aufgaben zu lösen — was nicht immer so glatt geht, wie sich das in der Theorie darstellt.

Mit realen Mitteln reale Probleme praktisch lösen!

Sicherheit der Systeme, der Daten und Programme im Web ist heute ein großes Problem, das oft vernachlässigt wird. Es wird dann

versucht, nachträglich erkannte Probleme zu beheben - wenn es eigentlich nach dem erlittenen Schaden schon zu spät ist. Um das Bewusstsein für die Gefahren im Wesentlichen offenen Internet zu schärfen, versuche ich in jedem Kapitel die in dieser Phase der Überlegungen notwendigen Hinweise zu Vorkehrungen, die die Sicherheit der Applikation erhöhen, zu geben. In einem der Kapitel am Schluss zeige ich, wie Benutzer-Identifikation gelöst werden kann und, damit zusammenhängend, der Einsatz von kryptographischen Methoden. Ähnlich wichtig sind Rechtsfragen: es muss sichergestellt sein, dass der Betreiber einer Webapplikation die Rechte an allen Texten und Bildern, die gezeigt werden hat und dass andere rechtliche Anforderungen (z.b. Impressum) erfüllt sind. Auf Aspekte von Sicherheit und Recht wird in den entsprechenden Kapiteln hingewiesen und am Schluss, im Sinne einer Checkliste, die wichtigsten Punkte zusammengefasst.

Die Bearbeitung hier gibt den Stand etwa 2014 wieder und beruht auf Open Source Programmen und Linux (Ubuntu oder eine andere auf Debian aufbauende Distribution), Ähnliches ist für andere Betriebssysteme erhältlich. Dass sich die Technik weiterentwickelt, ist selbstverständlich, wobei aber die grundsätzlichen Festlegungen, insbesondere die Standards des Web, notwendigerweise sehr langsam verändert werden und das hier Gelernte, zumindest im Prinzip, für einige Jahre Gültigkeit haben sollte.

Ein nicht unwesentlicher Lerneffekt des Kurses ist Strategien zu entwickeln, wie man mit einer zu Beginn unbekannten Software umgeht, wie man zielgerichtet die für die Aufgabe wesentliche Information erwirbt und sich bei Schwierigkeiten weiterhilft. Diese Strategien sind unverzichtbar in einer Welt die mehr und mehr auf elektronische Hilfsmittel setzt.

1
Gliederung der Vorlesung

Der Einleitungsteil beginnt mit dem Kapitel 2, in dem wir zeigen, welche Aufgaben Fusion Tables übernehmen können; für schwierigere Fragen kann Fusion Tables im Rahmen von programmierten Anwendungen eingesetzt werden, was ähnliche Methoden voraussetzt, wie sie hier im folgenden gezeigt werden.

Der erste Teil befasst sich mit der Planung des Projektes: Eine Arbeit größeren Umfangs, besonders wenn sie im Team erledigt werden soll, braucht eine Planung; im 2.1. Kapitel wird ein minimaler Projektplan erklärt und Programme zu dessen Erstellung vorgestellt. Bevor wir mit der Programmierung beginnen, muss die Benutzerschnittstelle (Kapitel 2.1.11) festgelegt werden und die Abläufe, die der Benutzer anstoßen kann, im Detail festgelegt sein. Damit die Arbeit richtig geplant werden kann, muss ein Überblick über die Technologie erreicht sein, der dann im weiteren Verlauf der Arbeit vertieft wird und mit Praxis angefüllt.

Der zweite, technische Teil erklärt die Grundlagen der Web Technologie. Welche Programme werden eingesetzt und wie wirken deren Funktionen zusammen? Zuerst eine Übersicht über das Internet (Kapitel 3) und davon unterschieden, das World Wide Web mitsamt der unterstützende Programme (Kapitel 2.1.14).

Der dritte Teil zeigt, wie Daten und Karten in die Applikation eingebunden werden. Im der grundsätzlichen Einführung dazu werden die Programmiersprachen, mit denen die Programme gesteuert werden und die Dienste, die angeboten werden, erklärt (Kapitel 6) Das Einbinden der geographischen Daten erfolgt in einem separaten Abschnitt (Abschnitt 8.1.2). Für die Speicherung der Daten, die heute meist in relationalen Datenbanken erfolgt gibt es Alternativen, die im vierten Teil ebenfalls kurz vorgestellt werden. CouchDB ist eine NOSQL Datenbank, die mit einem alternativen Transaktionskonzept besonders für mobile Anwendungen geeignet ist - im Kapitel 9. Linked Data wird in binären Tabellen strukturiert und kann weltweit an SPARQL endpoints abgefragt werden; hier sind beträchtliche Mengen von öffentlich

NOSQL = Not Only SQL, d.h. eine nicht-relationale Datenbank

Linked Data mit SPARQL Abfragesprache

verfügbaren Daten in einem einheitlichen Format nutzbar (Kapitel 10). Zum Schluss werden Hinweise gegeben, wie Applikationen abzusichern sind (Kapitel 11) und Rechtsfragen (Kapitel 12) beleuchtet.

1.1 Begriffe "Internet" und "Web"

Eine einheitliche Terminologie ist in einem jungen Feld kaum zu erwarten und in einem Bereich, der durch die rasche Entwicklung besonders in den USA geprägt ist, nicht zu erreichen. Begriffe werden durch Standards in englischer Sprache festgelegt und in rascher Folge überholt; Abkürzungen sind notwendig, um komplexe Ausdrücke (wie z.B. TCP/IP für Transmission Control Protocol/Internet Protocol) auf eine handhabbare Größe zu reduzieren. Ich bemühe mich hier in einem Feld, das durch eine mit englischen Ausdrücken durchsetzten Sprache auffällt, um möglichst klare, deutsche Begriffe - wo diese aber fehlen oder kaum bekannt sind, muss ich im deutschen Text die englischen Fachbegriffe verwenden.

Internet = Rechner, die mit einander verbunden sind

Ich werde von World Wide Web oder dem *Web* sprechen und meine damit das komplexe System von Dokumenten, die durch die verschiedenen Programme und die Infrastruktur für Kommunikation und Verarbeitung zugänglich ist. Das World Wide Web wird durch das World Wide Web Consortium (W3C) verwaltet.

Das *Internet* ist die Infrastruktur für das Web; ein Netzwerk von miteinander verbundenen Rechnern, die nach bestimmten Regeln Daten austauschen.

(World Wide) Web = System von miteinander verlinkten Dokumenten

Eine *Web Applikation* (oder Webanwendung) ist ein Anwendungsprogramm, das im Webbrowser des Benutzers abläuft und das Internet verwendet, um auf Dienste und Daten zuzugreifen.

Web Applikation = Programmierte Anwendung, die auf Web Ressourcen aufbaut

1.2 Ergebnis

Studenten sollten innerhalb eines Semesters eine einfache, räumliche Webapplikation entwickeln, die sie am Schluss des Semesters präsentieren und in einem technischen Bericht beschreiben. Während dem Semester werden Abgaben, wie z.B. ein Arbeitsplan oder eine Beschreibung des Benutzerschnittstelle erwartet; die Abgabe der Zwischenberichte wird aber nicht kontrolliert sondern ist nur ein Angebot an die Studenten, ihnen bei der Einteilung der Arbeit zu helfen und ihnen frühzeitig Hinweise geben zu können.

1.2.1 Technischer Bericht

Ein technischer Bericht hat zwei Adressaten:

- Einen späteren Programmierer, der das System erweitern, oder

verbessern soll;

- den Fachmann, der in einem Rechtsstreit abklären soll, ob die Aufgabe „richtig“ d.h. nach den zur Zeit gültigen Regeln der Kunst, gelöst wurde.

Der zweite Aspekt ist deutlich wichtiger. Der technische Bericht stellt die technische Lösung vor und liefert die sachliche Rechtfertigung für die Entscheidungen, die zu einem technischen Werk führen, zu dokumentieren. Er enthält eine Übersicht der Architektur, nicht detaillierte Beschreibungen der Programme, da solches besser als Kommentar im Programmtext eingefügt wird und dann bei Änderungen nachgeführt werden kann.

Ein technischer Bericht fängt mit der Aufgabenstellung an

Er geht von einer knappen Beschreibung der Aufgabe aus, in der Form, die vom Kunden genehmigt und unterschrieben worden ist. An dieser Beschreibung der Aufgabe wird, im Streitfall gemessen, ob die Aufgabe richtig erfüllt wurde oder nicht und daraus folgt, ob der Auftraggeber den ausbedungenen Preis bezahlen muss.

Beispiel für eine Beschreibung der Aufgabe, die dann durch die Beschreibung des Benutzer-Schnittstelle präzisiert wird:

> Es ist eine Web-Applikation zu erstellen, die Mitarbeiter und Studenten der TU Wien anzeigt, wo ein ihren Präferenzen entsprechendes Mittagessen angeboten wird; es ist Preis und Wegzeit zum anbietenden Gasthaus zu optimieren und der Weg auf Verlangen anzuzeigen.

Der technische Bericht dokumentiert die Begründung für die Entscheidungen. Die spätere Beurteilung im Streitfall, aber auch die Beurteilung im Rahmen der Übung, erfordert nicht, dass diese Entscheidungen “objektiv richtig” sind, sondern nur, dass sie, entsprechend dem Stand der Technik vertretbar sind. Stand der Technik heißt, dass berücksichtigt ist, was ein sachkundiger Ingenieur in dieser Situation und Zeit wusste; es kann nicht verlangt werden, dass die Zukunft richtig vorhergesagt wird. Stellt sich später heraus, dass wegen der späteren technische Entwicklung eine andere Wahl besser gewesen wäre, so kann das nicht als Fehler gerechnet werden.

Entscheidungen entsprechen dem Stand der Technik

Hinterher sind alle klüger!

Der technische Bericht ist - wie üblich für technische Berichte - in einer objektiven, emotionslosen Sprache abzufassen; qualifizierende Adjektive sind zu meiden. Insbesondere ist acht zu geben, dass Wörter konsistent verwendet werden - das gleiche Wort bezeichnet immer den gleichen Sachverhalt, zwei verschiedene Wörter bezeichnen nie dasselbe .

Keine qualifizierenden Adjektive!

Keine Synonyme und Keine Homonyme!

1.2.2 Demonstration

Eine Demonstration der entwickelten Anwendung auf einem Rechner mit Zugriff auf eine Datenbank auf einem anderen Rechner wird erwartet; eine Web-Applikation sollte auf jedem beliebigen Rechner im

Web Browser funktionieren (zumindest auf jedem modernen Browser wie z.B. Firefox).

Beachten sie, dass “Demo-Effekte” häufig im letzten Augenblick zuschlagen und bereiten sie eine Präsentation vor, bei dem sie Screen Shots einbauen, die sie von einer Vorbereitung der Demo gezogen haben, so dass sie nicht vollständig verloren sind wenn am Tag der Vorstellung nichts mehr geht. .

Profi's wissen, dass Demo-Effekte auftreten und bereiten Screenshots vor, die sie notfalls verwenden können.

Für die Demonstration benötigen sie Kundendaten und andere spezifische Eingaben, die meist in einer Datenbank abgelegt werden. Sie sollten genug Daten speichern, dass sie die verschiedenen Aspekte der Applikation zeigen können. Die Daten sollten realistisch sein, müssen aber für die Demo nicht exakt sein. Je nach Anwendung reichen 20 - 100 Datensätze um die Funktionsfähigkeit der Anwendung nachzuweisen.

1.2.3 Beurteilung

Die einfachste Lösung für die Aufgabe ist eine Web Applikation die auf dem Appache Server mit PHP, SQL und Google Maps funktioniert; minimal muss die Applikation Daten in einer externen Datenbank vorhalten und die Daten auf einer Karte dargestellt werden. Diese Leistung führt zu einer Note 3 (befriedigend).

Bessere Lösungen, die einen oder mehrere der folgenden Erweiterungen oder Verbesserungen enthalten, werden mit 2 (gut - zumindest eine Erweiterung) oder mit 1 (sehr gut - mehr als eine wesentliche Erweiterung) bewertet. Erweiterungen oder Verbesserungen gegenüber der Minimallösung sind z.B.:

- Lösung auf einem Server (nicht “localhost”),
- Lösung mit dynDNS und Server hinter einem Router mit NAT,
- Änderung der Daten vom Nutzer werden in der Datenbank gespeichert (mit Transaktions Konzept),
- es wird Linked Data (RDF und SPARQL) verwendet,
- Applikation läuft auf einem mobilen Gerät im Browser,
- Applikation läuft lokal auf einem mobilen Gerät,
- Verschlüsselung, sichere Benützeridentifizierung,
- besonders gute Benützerschnittstelle.

Kann die Applikation am Ende des Semesters nicht demonstriert werden, die Gruppe liefert aber eine plausible Erklärung, welche Probleme unerwarteten technischen Probleme die Lösung verunmöglicht haben haben, so wird das Ergebnis mit 4 (genügend) beurteilt. Wird die

Applikation nicht fertig so ist das Ziel der Lehrveranstaltung nicht erreicht und die Bewertung ist 5 (ungenügend) - Planung der Arbeit und Kontrolle des regelmässigen Arbeitsfortschritts ist Teil der Aufgabe.

2
Darstellung einfacher räumlicher Daten

Der wohl einfachste Fall der Darstellung räumlicher Daten ist die Ausgabe einer Karte mit eingetragenen Orten der Datenpunkte. Das wurde früher mit Stecknadeln auf Karten erreicht und hat die Standorte von Filialen von Firmen, die Verteilung der Kunden in einer Stadt oder die Sichtungen von Tieren in einem Waldgebiet gezeigt.

Mit Fusion Tables hat Google einen überall verfügbaren Dienst für einen Teil solcher Aufgaben eingerichtet und bietet ihn zur freien Nutzung an [1]. Ein kommerzieller Dienst gegen Bezahlung enthält etwas mehr Funktionalität. Der Dienst ist einfach aufgebaut und einfach handhabbar, so dass minimale Instruktionen notwendig sind. Es wird das weit herum bekannte Interface für Tabellenkalkulation als Grundlage für die Bedienung ausgenützt. [2]

[1] https://developers.google.com/maps/

Fusion Tables erwartet als Eingabe eine Tabelle - die aus einem üblichen Tabellenkalkulationsprogramm übernommen und hochgeladen werden kann und in denen in einer Kolonne z.B. Straßenadressen in einem für Google Maps verwendbaren Format enthalten sind (Abb. 8.2). Diese Tabelle wird dann als Karte angezeigt, wobei für jede Zeile eine Marke auf den angegebenen Ort gesetzt wird. Klickt man mit dem Mauszeiger auf diese Marke, werden die in der Tabelle mit diesem Ort verbundenen Daten angezeigt.

[2] Visicalc hat die elektronische Tabelle, die selber rechnet, 1979 für die damals aufkommenden kleinen Rechner (mit CP/M Betriebssystem) entwickelt und popularisiert. Microsoft hat das Konzept übernommen und in EXCELL Anfangs der 1990er realisiert und damit Visicalc und Lotus 1-2-3 aus dem Markt gedrängt.

Fusion Tables stellen Daten aus einem Tabellenkalkulationsprogramm in einer Karte dar.
csv = comma separated values; file mit einer Zeile pro Datensatz, die Werte sind durch Kommas getrennt

Die Ortsbezeichnungen können Ortsnamen, Postadressen (Straße, Hausnummer, Ort) oder geographische Länge und Breite sein. Die Tabelle kann im persönlichen Google Drive gespeichert werden und andern erlaubt werden, die Tabelle anzusehen oder zu ändern. z.B. kann eine Umfrage über das Web so organisiert werden, dass die Ergebnisse automatisch in eine Tabelle eingetragen werden und dann visualisiert werden (Abb. 3).

Eine Auswahlfunktion erlaubt, nicht alle Daten einer Tabelle, sondern nur eine Auswahl, die bestimmten Kriterien erfüllt, zu zeigen. Neben einfachen Markierungen für Orte sind Linien und Polygone möglich, die dann verschieden farbig dargestellt werden können[3].

[3] https://developers.google.com/maps/documentation/javascript/layers#fusion_table_styles

Text	Autor	Location
Strudelhofstiege	doderer	Strudlhofgasse 8, 1090 Vienna, Austria
cafe griensteidl	kraus	Michaelerplatz 2, 1010 Vienna, Austria
prater	schnitzler	Leopoldstadt, 1020 Wien, Austria
porzelangasse	doderer	Porzellangasse, 1090 Vienna, Austria
Basiliskenhaus	Sage	Schönlaterngasse 7, 1010 Vienna, Austria
Fuer Josefine Hawelka	Friederike Mayroecker	Dorotheergasse 6, Vienna, Austria
Huschhusch ins Korb!	Elfriede Jelinek	Kaffee Korb, Vienna, Austria

Tabelle 2.1: Orte in Wien, die in der Literatur erwähnt werden

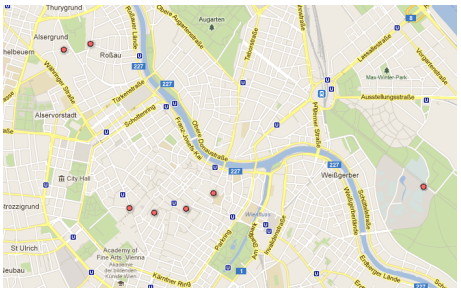


Abbildung 2.1: In der Literatur erwähnte Orte auf der Karte Wiens

Mit Fusion Tables kann man einfache Daten räumlich darstellen , um sich rasch einen Überblick zu verschaffen oder um eine einfache graphische Darstellung für andere zu produzieren. Es ist möglich, dass Nutzer der Webpage Daten eingeben. Sobald aber höhere Anforderungen gestellt werden, geht es nicht ohne Programmierung,

2.1 Fusion Tables einbinden

Fusion Tables können in Web Anwendungen eingebunden werden. Dazu dient die Schnittstelle für Programmierer.[4]); diese Schnittstelle folgt den üblichen Regeln für Web Applikationen, die in der Folge besprochen werden (2.1.18).

Fusion Tables bieten Dienste im Web an, die von Browsern genutzt werden können. Es können die wesentlichen Befehle zur Veränderung der Tabellen über das HTTP Protokoll an Google geschickt werden (Abschnitt 6.9) und dann die daraus resultierenden Karten im Browser gezeigt werden. Es ist damit möglich, auf die Verwendung einer Datenbank zu verzichten und die Daten in Tabellen zu halten. Aber auch dies Lösung erfordert Programmierung.

Für Anwendungen, bei denen nicht mehrere Nutzer gleichzeitig Daten verändern, sondern nur Abfragen - allenfalls mit unterschiedlichen Auswahlen - und die Datenmengen gering sind, dürfte dies eine brauchbare Lösung darstellen. Die Befehle zum Ändern der Daten sind ähnlich wie SQL und die Entwicklung ist wesentlich einfacher, bietet aber weniger Möglichkeiten eine sichere Anwendung zu bauen und die Daten vor Unbefugten zu schützen (nicht zuletzt, weil die Daten bei Google in der Cloud liegen müssen).

Abbildung 2.2: Wählbare Länder eingefärbt

quick & dirty mapping

[4] https://developers.google.com/fusiontables/docs/v1/getting_started

API = Application Programmers Interface

"Cloud" (Wolke) - die Daten sind an einem durch eine Web-Adresse bekannten aber nicht direkt kontrollierten Ort abgelegt; es gibt verschiedene Anbieter für solche Speicherungsdienste.

Projekt Start:Erstellen eines Arbeitsplans

Ein Projekt muss ein definiertes Ziel und einen Startpunkt haben; das Projekt ist beendet, wenn das Ziel erreicht ist. Die Planung des Projektablaufes erleichtert die Überwachung und hilft, dass das Ziel zeitgerecht erreicht wird.

Ausgangslage

Die „Geschäftsleitung“ hat, beruhend auf der Machbarkeitsstudie für eine GIS Applikation, beschlossen, das Projekt zu realisieren. Es ist damit festgelegt:

- welcher Benutzerkreis in welcher Situation welches Informationsbedürfnis hat,
- wie diese Information produziert und kommuniziert werden soll,
- welche Daten dazu benutzt werden sollen.

Es liegt ein grobes Konzept vor, wie das technisch realisiert werden kann. Nun müssen diese Unterlagen verfeinert und daraus ein technisches System von Software und Daten erstellt werden, das die Informationsbedürfnisse der potentiellen Nutzer befriedigt. Ziel ist es, ein funktionsfähiges System für die Verkaufsabteilung zu erstellen, die sich um Marketing und Vertrieb kümmern wird; Fragen der Markteinführung und des Marketings sind hier nicht eingeschlossen und werden nicht behandelt. Mit der Demonstration der Funktionsfähigkeit des Programmes mit Daten und dem technischen Bericht ist die Aufgabe erfüllt.

Zielbeschreibung

Ein Projekt beginnt mit einer Beschreibung des Zieles, die konkret und operational ist. Operational heißt hier, dass die Beschreibung des Zieles als praktischer Test verwendet werden kann. Also nicht “die Applikation hilft ein nahe gelegenes Spital zu finden” sondern “die Applikation gibt zu jeder Adresse in X das nächstgelegene Spital (in Luftlinie)”. Die zweite Beschreibung des Zieles führt zu einem Test, der nicht bestanden oder bestanden ist; bei positivem Ergebnis daraufhin wird der Programmierer bezahlt!.

Zielbeschreibung als operationaler Test

Wie geht man vor?

Ein Projekt ist zu organisieren. Dazu gehört, dass man sich zuerst überlegt:

- welche Tätigkeiten erforderlich sind, um das Ziel zu erreichen,
- wie groß der Aufwand dafür ist und
- welche Abhängigkeiten zwischen ihnen bestehen.

Erforderliche Tätigkeiten

Um die Arbeit zu planen, brauchen sie eine möglichst Liste der Arbeitsschritte. Arbeitsschritte, die von den Ergebnissen anderer abhängig sind, müssen in der richtigen Reihenfolge geplant werden; Arbeitsschritte, die nicht voneinander abhängig sind, können parallel ausgeführt werden.

Für jeden der folgenden Arbeitsschritte sind Abhängigkeiten zu erfassen und der Zeitbedarf zu schätzen.

2.1.1 Benutzerschnittstelle

Die Sicht des Benutzers muss im Detail modelliert werden. Für die Programmierung braucht es Skizzen aller Bildschirme, die der Benutzer sieht, und es muss angegeben werden, welche Eingaben erwartet werden und wie die Applikation darauf reagiert (2.1.11). Es muss somit die Darstellung der wichtigsten Bildschirme, die in der Machbarkeitsstudie erarbeitet wurden, ergänzt werden.

2.1.2 Bedienungsanleitung

Es empfiehlt sich, eine detaillierte Bedienungsanleitung zu erstellen. Beim Schreiben der Bedienungsanleitung muss die Aufgabe Schritt für Schritt durchgedacht werden. Dabei fallen meist Vereinfachungen für die Benutzerschnittstelle auf. Ziel ist, eine möglichst einfache Bedienung zu erreichen, nicht eine möglichst einfache Programmierung. Erklären sie jeden Schritt, damit nichts vergessen wird!

Ziel ist einfach für den Benutzer, nicht einfach für den Programmierer!

2.1.3 Überprüfung von Benutzerschnittstelle und Bedienungsanleitung

Testen sie das Interface durch Simulation mit verteilten Rollen und protokollieren sie die Eingaben und erwartete Ausgaben: einer spielt den Benutzer, ein Zweiter den Computer und ein Dritter protokolliert. Dieses Protokoll wird später zum Testen und zum Schluss zur Prüfung, ob die Aufgabe erfüllt ist, herangezogen.

2.1.4 Technik abklären

Es ist notwendig, sich einen Überblick über die notwendigen technischen Einwirkungen zu verschaffen. Welche Hardware wird verwendet?

Welche Software ist erforderlich? Was ist vorhanden, was muss beschaffen werden? (Abschnitt 2.1.11)

Die Komponenten sind mit einfachen Beispielen auszuprobieren, sodass zumindest die Grundfunktionen bekannt und in dieser Umgebung getestet sind.

Zu solchen Tests eignen sich von den Herstellern gelieferte oder publizierte Anwendungen. Programmiersprachen werden typischerweise mit einem Minimalprogramm getestet, das „Hello world“ (oder ähnliches) auf den Bildschirm schreibt. Für praktisch jedes Programm gibt es ein „Hello world“-artiges Programm zum Testen. Damit lässt sich ein erster grober Funktionstest durchführen und die Bestätigung erreichen, dass das Programm zumindest grundsätzlich funktioniert.

2.1.5 Welche Software braucht es?

Eine geographische Web Anwendung braucht typischerweise ein Betriebssystem, einen Webserver, eine Datenbanksoftware, eine GIS Software und eine Abfragesprache. Als GIS Software reicht oft der Zugang zu einem Map Server und eine Möglichkeit diesen zu programmieren.(2.1.14)

2.1.6 Hardware

Aus der Festlegung der Software folgt, welche Hardware benötigt wird. Es braucht meistens einen am Web angebundenen Server und einen Laptop für den Benutzer (Abschnitt 2.1.11). Welche Geräte sind vorhanden? Ist die ausgerichtete Software auf diesen Geräten vorhanden oder kann sie installiert werden?

2.1.7 Welche Daten sind erforderlich?

Es gilt aus der genauen Beschreibung der Benutzerschnittstelle abzuleiten, welche Daten erforderlich sind. Danach kann man die Strukturierung der Daten für die Datenbank festlegen (Kapitel 7). Woher beziehen wir die Daten? Wie erfolgt das? Wie formen sie die erhaltenen Daten um und legen sie in die Datenbank ab?

Es muss sichergestellt werden, dass der Betreiber der Applikation diese Daten rechtsmäßig benutzt (Abschnitt 12.0.6); bei der Gestaltung einer realen Anwendung ist der Aufwand, um die Nutzungsrechte zu erlangen und die Zeit für den Schriftverkehr dazu erheblich und muss unbedingt eingeplant werden.

2.1.8 Realisierung

Wenn die Komponenten abgeklärt sind, kann die eigentliche Realisierung starten. Hier gilt es zuerst die Architektur der Anwendung festzulegen (2.1.18). Wie werden die Aufgaben auf die Softwarekomponenten verteilt? Wie werden die Programme in Module aufgeteilt? Danach kann der Code programmiert werden.

2.1.9 Testen

Es ist unumgänglich die Software zu testen. Wenn sie Protokolle der simulierten Tests haben, können sie nun prüfen, ob die Programme wie gewünscht funktionieren.

2.1.10 Bericht

Zu jedem technischen Projekt gehört ein Bericht (Abschnitt §1.2). Er beschreibt, was man getan hat und warum man es so getan hat (Abschnitt 1.2.1).

2.1.11 Demonstration und Übergabe an die Verkaufsabteilung

Wenn alles funktioniert, demonstriert man das System und übergibt an die Verkaufsabteilung, die nun die Markteinführung erledigt.

Benötigte Hardware

Für die Entwicklung einer Webanwendung sind wenige Geräte notwendig; im Wesentlichen genügt fast ein einziger PC, auf dem man programmieren kann und der mit dem Web verbunden ist.

Zum ernsthaften Arbeiten ist erforderlich, dass der PC genügend Speicherplatz hat, so dass die notwendige Software für die Applikation und die Werkzeuge, die zum Erstellen der Programme verwendet werden, eingerichtet werden können und dann noch Platz für die benötigten Anwenderdaten frei ist. Das ist aber heute meist kein Problem - ein neuerer PC mit einem 500GB Festplatte sollte ausreichen.

Für das Testen sollte ein Gerät vorhanden sein, das so genau wie möglich dem Gerät entspricht, das die vorgesehenen Nutzer benutzen; ist die Markteinführung erst in einer entfernteren Zukunft geplant - was bei Informatikprojekten vielleicht 12 Monate meint - so ist zu überlegen, welche Geräte in 12 bis 18 Monaten aktuell sein werden und ein möglichst ähnliches, heute erhältliches Gerät zum Testen zu benutzen.

Für die Entwicklungsarbeit sollte ein PC benutzt werden, der vom Produktionssystem getrennt ist, so dass die Entwickler nicht versehentlich auf die laufenden Programme zugreifen und die Kunden stören.

Zeitplan

Möglichst realistisch wird der Zeitaufwand für jede Aufgabe geschätzt. Auch wenn man wenig Erfahrung hat, muss man versuchen den Aufwand für jede Aufgabe in Abschnitt 2.1 in Personenstunden oder -tagen zu schätzen. Diese Schätzungen werden laufend korrigiert und

am Schluss beurteilt. Die Erfahrung, die in einem Projekt gewonnen wird, fließt in die Planung des nächsten Projekts ein.

Gewisse Arbeiten können gleichzeitig ausgeführt werden, z. B. die Beschreibung der Benutzerschnittstelle und Beschaffung der Software, was bei der Planung berücksichtigt werden muss. Meilensteine bezeichnen Ereignisse bei denen definierte Teile fertig gestellt sind und werden mit operationalen Tests abgeschlossen; es sind meist Ablieferungen von Dokumenten und Tests von Software - d.h. Ergebnisse, die leicht zu prüfen sind.

milestones festlegen

Aus der Liste der Arbeitsschritte mit Zeitschätzungen ergibt sich ein Zeitplan, bei dem für jede Aufgabe ein Verantwortlicher festgelegt und ein Abschluss definiert ist. Es empfiehlt sich sehr am Schluss eine Reserve einzuplanen, um unvermeidliche Verzögerungen abfangen zu können. Für die Erstellung des Zeitplans z.B. als GANTT Diagramm (Fig. 2.3) gibt es verschiedene Hilfsmittel, z. B. www.grantproject.biz

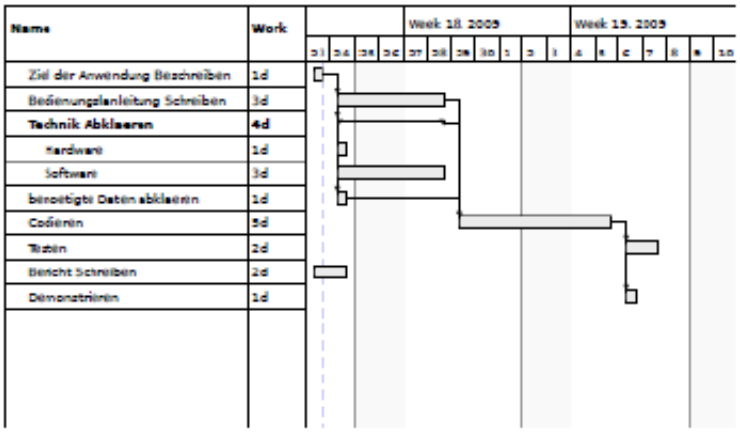


Abbildung 2.3: Zeitplan als GANTT Diagram

Zusammenfassung

Ein Projekt — auch eine Universitätsübung als Teil des Studiums — geht leichter von der Hand, wenn es gut geplant ist — wobei hier Augenmaß angesagt ist: mehr Aufwand als nötig bei der Planung bringt keine weitere Verbesserung. Die bei der Planung erstellten Dokumente geben bereits Teile des Schlussberichtes; es ist einfacher sie zu Beginn zu erstellen, als sie später zu rekonstruieren.

Planung der Aufgabe soll wohl nicht mehr als 10% des gesamten Projektaufwandes erfordern!

Muss für die Erstellung eines Softwareproduktes ein Angebot erstellt werden, so sind die überlegungen, die in die Erstellung des Zeitplanes einfliessen unerlässlich. Der Zeitbedarf in Arbeitsstunden kann mit einem guten Zeitplan geschätzt werden und daraus eine gute Kostenschätzung erstellt werden. In solchen Fällen wird der Zeitplan in Tagen oder Wochen nach dem noch unbekannten Startzeitpunkt eines

Projektes beschrieben. Die Zeit die für die Realisierung notwendig ist, d.h. die zwischen Entscheidung und Ablieferung der Anwendung verstreicht, ist meist ebenfalls ein wichtiges Beurteilungskriterium eines Angebotes, da für den Auftraggeber ein früher Eintritt in den Markt oft entscheidend ist

"time to market" ist wichtig

Beschreibung der Benutzerschnittstelle

Ziel

Der Nutzer ihrer GIS Anwendung wird nur die Benutzerschnittstelle sehen; sie entscheidet über den Erfolg, auch den kommerziellen Erfolg ihrer Anwendung und die Zukunft der Firma, die die Anwendung betreibt. Wenn ich eine Website benutze, um bei einer Firma z. B. ein Buch einzukaufen, entscheidet die Qualität der Benutzerschnittstelle, ob das Geschäft überhaupt zum Abschluss kommt. Überraschend oft scheitert der Versuch etwas zu kaufen an einem Problem des Web-Auftrittes der Firma das die Anwendung für den Nutzer nicht bedienbar ist.

Die Benutzerschnittstelle entscheidet oft über den Erfolg eines Computerprogrammes!

Der Web-Auftritt entscheidet, ob die Erfahrung des potentiellen Kunden positiv ist oder die, leider häufige, Frustration das Scheitern ist und die Firma nicht mit einem zweiten Versuch meinerseits rechnen kann - der potentielle Kunde ist für lange Zeit verloren. Die Benutzerschnittstelle ist für eine Firma so wichtig, wie die Gestaltung des Ladenlokals, Auswahl und Training des Bedienungspersonals bei einem konventionellen Geschäft. Die Gestaltung eines Ladenlokals überlässt man nicht den Handwerkern und gestaltet das Geschäft nicht so, dass die Erstellung für die Handwerker möglichst einfach ist, sondern ein Innenarchitekt bemüht sich einen Raum zu schaffen in dem der zukünftige Kunde ein positives Einkaufserlebnis hat und wiederkommt. Genauso verhält es sich mit der Gestaltung einer Web Applikation. Es gilt nicht den Programmierer die Entscheidung der Gestaltung zu überlassen, sondern aktiv aus der Sicht des Kunden zu gestalten. Ziel darf nicht einfaches Programmieren sein, sondern eine ansprechende graphische Gestaltung und eine *einfache Bedienung* ohne frustrierende Überraschungen.

GUI ist das Ergebnis eines Entwurfsprozesses, nicht der Programmierung!

GUI ist einfach für den Kunden, nicht einfach für den Programmierer!

Abbildung 2.4: Photo modernes Geschäft im Vergleich mit Dorf-Geschäft in Kuba

Mit dem Entwurf der Benutzerschnittstelle und der zugehörigen Anleitung legt man fest, wie der Kunde die Anwendung erleben wird. In diesem Kapitel zeige ich, wie man vorgehen kann, wie ein Entwurf aussehen kann und, wie der Entwurf zu testen ist bevor man mit dem Programmieren beginnt.

Die Schnittstelle muss festgelegt und getestet werden, bevor die Programmierung ernsthaft beginnt. Es mag nützlich sein, dass geprüft wird, ob sich die verwendeten Konzepte in Programme umsetzen

lassen, indem kleine sehr einfache Testbeispiele programmiert und beurteilt und vermieden wird, dass auf einem Ansatz aufgebaut wird, der sich dann leider nicht realisieren lässt.

Im Folgenden wird die übliche Technologie von Web-Anwendungen für 2014 angenommen und wenige Tools angeführt; es gibt heute eine sehr grosse Auswahl an technischen "frameworks" für die Erstellung von Webapplikationen, die aber erst nach dem logischen, benützerorientierten Entwurf des Benutzerschnittstelle einsetzen. Die Technologie entwickeln sich rasch und verschiedene Lösungen sind gegenwärtig Mode, ohne dass sich ein Konsens abzeichnet.

Entwurf

Der Entwurf der Benutzerschnittstelle umfasst alle Bildschirme oder Bildschirmteile, die der Benutzer sehen wird. Wesentlich ist, dass alle Bildschirme so entworfen werden, dass sie in graphischer Gestaltung, Terminologie und Interaktionsmuster einheitlich sind. Die Einheitlichkeit wird bei einem größeren Webauftritt mittels eines Content Management Systems z. B. www.opencms.org oder www.typo3.com gelöst.

2.1.12 Konsistenz

Das Wichtigste beim Entwurf der Benutzerschnittstelle ist Konsistenz, d.h. Einheitlichkeit[5]. Ein Nutzer merkt sich, welche Interaktionen welche Wirkungen haben. Er lernt die Begriffe, die verwendet werden und erwartet, dass das, was er auf einer Seite „gelernt" hat, auch für die nächste gilt.

[5] Inc. Apple Computer. *Human Interface Guidelines: The Apple Desktop Interface.* Addison-Wesley, 1987

Eine Nutzung einer GIS Anwendung ist vergleichbar mit einer Konversation mit einem Menschen: wir passen uns rasch im Wortschatz an, verwenden die gleichen Wörter wie unser Gegenüber und erwarten, eine authentische Persönlichkeit, d.h. eine im Kern gleich bleibende, vorhersehbare Reaktion auf die wir uns verlassen können. Genau das Gleiche gilt für die Schnittstelle einer Webanwendung: keine negativen Überraschungen!

2.1.13 Wortschatz

Es gilt zu Beginn festzulegen, welche Wörter für die Bezeichnung der Objekte und Operationen, die in der Anwendung vorkommen, verwendet werden. Am besten macht man für sich eine Liste, die man aus einer allgemeinen Beschreibung der Anwendung (Abschnitt §2.1.13) herausziehen kann und dann vereinheitlicht. Dieser Wortschatz muss konsistent verwendet werden. Das heißt:

Keine Synonyme, keine Homonyme!

- Keine *Synonyme*; der Benutzer darf sich darauf verlassen, dass

überall wo ein Ding x gemeint ist, auch x steht.

- Keine *Homonyme*; der Benutzer darf sich darauf verlassen, dass x immer das Ding x und nie etwas anderes meint.

Selbstverständlich gilt dies für die Bezeichnungen der Objekte, die Operationen und natürlich auch für die Begriffe, die sich nicht auf die Anwendung bezeichnen, sondern die Interaktion mit dem Computer betreffen.

Beachte, dass sich der Wortschatz der Anwendungswelt deutlich von jenem der Computer-Technik unterscheidet und der Benutzer sehr verwirrt wird, wenn er plötzlich mit Computer-Jargon konfrontiert wird. Insbesondere gilt, dass das gleiche Wort nicht in der üblichen und der Computer-Bedeutung vorkommen darf, auch wenn der Sinn aus dem Zusammenhang klar ersichtlich scheint. Der Nutzer kennt den Zusammenhang nicht und kann deshalb den Unterschied nicht verstehen. [6]

Wortschatz des Anwenders wählen, nicht derjenige des Programmierers!

[6] Gefährlich sind Wörter wie “Feld” eine landwirtschaftliche Fläche oder ein Eingabe-Feld der Benutzerschnittstelle?

Allgemeine Beschreibung

Eine allgemeine Übersicht der Anwendung und des Umfeldes hilft allen Beteiligten, von Designern über die Programmierer bis zu den späteren Benutzern, zu verstehen, welche Informationen die Nutzer benötigen und wozu sie sie verwenden; dieses Beschreibung kann aus der Machbarkeitsstudie übernommen und muss nur überarbeitet werden.

Die allgemeine Beschreibung legt den Wortschatz fest, der verwendet wird — und zwar aus der Sicht der Anwendung und nicht aus der Sicht der optimalen Lösung für Computerexperten; Ziel ist, diese benutzerfreundliche Sicht für den Programmierer festzulegen auch wenn es mehr Aufwand für die Programmierer erfordert. Die Nutzer zahlen, in der einen oder anderen Form, die Rechnungen (inklusive der Löhne der EDV-Spezialisten) also müssen ihre Ansprüche Priorität haben. Aus der allgemeinen Beschreibung, allenfalls ergänzt mit Beschreibungen der Spezialfälle, ergibt sich der Wortschatz der zu verwenden ist.

Diese allgemeine Beschreibung ist nicht nur die Grundlage für die weitere Entwicklung der Web-Anwendung, sondern auch für die Arbeit der Marketing und Verkaufsabteilung. Die „Allgemeine Beschreibung“ verbindet Technik und Marketing. Deshalb ist es wichtig, dass es auf alle Fragen von Technik und Marketing, was die Anwendung bieten soll, eine ausreichende Antwort gibt. Keine Interpretation soll nötig werden, da sie Diskrepanzen erlaubt zwischen technisch realisierten Produkten (Interpretation der Techniker) und vom Marketing tatsächlich verkauften Produkten (Interpretation der Verkaufsabteilung), was schließlich beim Kunden zu Enttäuschung der Erwartungen und Frustration führt.

Beschreibung des Ablaufes

Der Ablauf einer Interaktion eines Nutzers ist im Detail festzulegen und zwar für jeden der verschiedenen Interaktionstypen, die die Anwendung vorsieht.

Für jeden Interaktionstypen beginnt die Beschreibung mit der Situation des Nutzers. Was will er wissen? Welche Informationen benötigt er? Dann ist anzugeben, welches Wissen der Nutzer hat und allenfalls andere relevante Details, wie sie bereits in der Machbarkeitsstudie diskutiert wurden

. Daraufhin beschreibt man den Ablauf: Was sind die nötigen Eingaben? Welche Ausgaben folgen? Für jeden Interaktionstyp braucht es konkrete Angaben für mehrere Beispiele; diese werden später zum Testen nützlich sein. Es muss festgelegt werden, welche Eingaben des Nutzers zu einem nächsten Bildschirm führt. Oft sind verschiedene Aktionen möglich und dann folgen verschiedene Bildschirme. Zustands-Übergangs-Diagramme (2.6), in denen jeder Bildschirm eine eindeutige Bezeichnung hat, beschreiben das präzise und sind der Ausgangspunkt der Programmierung.

Wenn alle Interaktionstypen beschrieben sind, ist zu prüfen, ob diese alle Situationen beschreiben, in denen der Nutzer auf Grund der allgemeinen Beschreibung eine Antwort von der Anwendung erwarten darf. Welche Fälle sind ausgeschlossen und warum? Allenfalls ist die allgemeine Beschreibung entsprechend anzupassen.

Beschreibung der Interaktionen Bildschirm für Bildschirm

Für jeden Interaktionstyp ist ein Ablauf „Bildschirm für Bildschirm“ zu zeichnen. Jede Skizze eines Bildschirmes zeigt, welche Informationen dem Nutzer angeboten werden; wozu vollständiger Text in der gewählten Terminologie erforderlich ist. Die Skizze zeigt welche Eingaben der Nutzer machen kann. Das umfasst Text und Zahleneingaben, aber auch andere Eingaben, wie Schieber, Kästchen zum Ankreuzen und Knöpfe zum Drücken. Mit welchen Eingaben wird eine Verarbeitung ausgelöst? Das kann z. B. ein „OK“ oder „weiter“ Knopf sein, man muss den Nutzer aber auch immer die Möglichkeit geben, zurückzugehen (meist kein separater Knopf nötig, sondern im Webbrowser als „back“ (←) Knopf enthalten [7].

Für jede mögliche Aktion des Benutzers der eine Verarbeitung auslöst, ist ein neuer Schirm zu zeichnen, der wiederum angibt, welche Informationen dem Nutzer präsentiert werden, welche Daten von vorangegangenen Schirmen wiedergegeben werden etc. Glücklicherweise baut sich eine Anwendung meist nur aus einem halben bis ganzen Dutzend Bildschirmen auf — viele Aktivitäten führen zurück auf Schirme, die schon beschrieben sind.

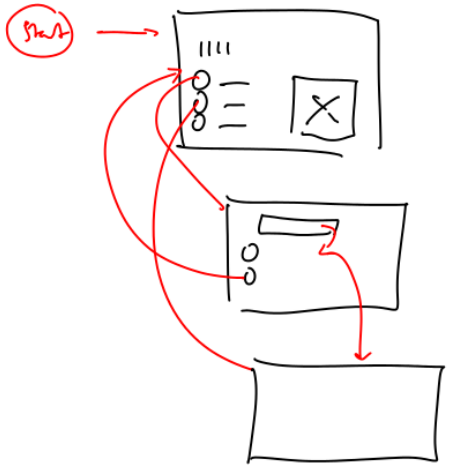


Abbildung 2.5: Jeder Bildschirm und die Übergänge werden skizziert

[7] J. Nievergelt and J. Weidert. *Methodology of Interaction*, chapter Sites, Modes and Trails : Telling the User of an Interactive System where he is, what he can do, and how to get to places, pages 327–338. North Holland, 1980

2.1.14 Zustand-Übergangs-Diagramme

Zur Überprüfung der Vollständigkeit und als Hilfsmittel für die Programmierung stellt man den Ablauf von Schirm zu Schirm in einem Zustands-Übergangs-Diagramm dar (2.6). Ein Zustand-Übergangs-Diagramm ist ein Graph, bei dem die Knoten Zustände, d.h. Bildschirme, die der Nutzer sieht, darstellen und die Kanten, die Verarbeitung auslösenden Aktionen des Nutzers sind. Diese Kanten zeigen den Übergang von einem Zustand (Schirm) zum nächsten Zustand (Schirm). Im Zustands-Übergangs-Diagramm kann leicht geprüft werden, ob die Abfolgen vollständig sind und alle Schirme beschrieben sind: zu jedem eine Aktion auslösenden Element gehört ein Pfeil der auf einen Schirm zeigt. Es kann aber überprüft werden, ob gleiche Aktionen immer die gleichen Effekte haben und ob gewisse Aktionen immer angeboten sind (z. B. Quit, Back).

Will man in einem Bildschirm Funktion einsetzen, die man noch nicht implementieren will was bei Übungen oder Prototypen vernünftig ist so kann das im Zustand-Übergangs Diagramm durch einen Übergang zu einem Bildschirm “xx noch nicht implementiert” dargestellt werden.

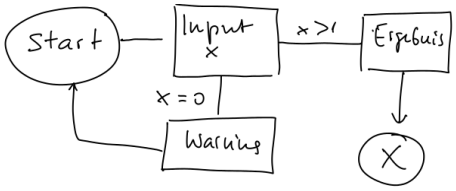


Abbildung 2.6: Zustands-Übergangs-Diagramm zeigt Benutzerablauf

Beispiel “Mittagessen”

Für die vorne beschriebene Aufgabe “Mittagessen” kann der Ablauf durch folgende Schritte beschrieben werden:

- Start der Applikation: Identifizierung des Nutzers (allenfalls durch ein Token im Rechner erfolgt und nur Kontrolle),
- Eingabe oder Änderung von Präferenzen: eine Liste von Küchen-Typ (Italiener, Hausmannskost, Asiatisch etc.),
- Ausgabe der Angebote von heute, gereiht nach Präferenz mit Angabe des Gasthauses mit Preis und Wegzeit,
- Darstellung des Weges dorthin als Karte.

für jeden dieser Schritte ist nun eine Skizze des Bildschirmes zu erstellen[8]:

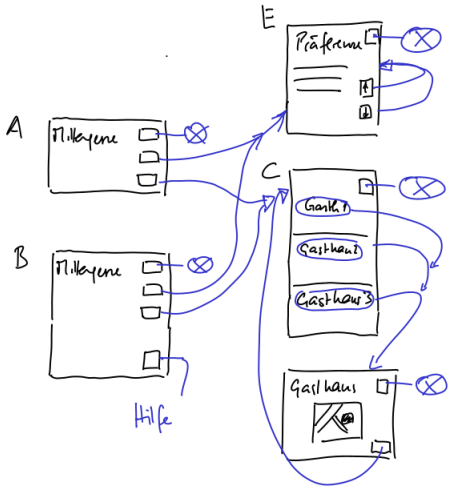


Abbildung 2.7: überblick über Ablauf

Ein einfacherer Ablauf wird später gefunden - 5.2

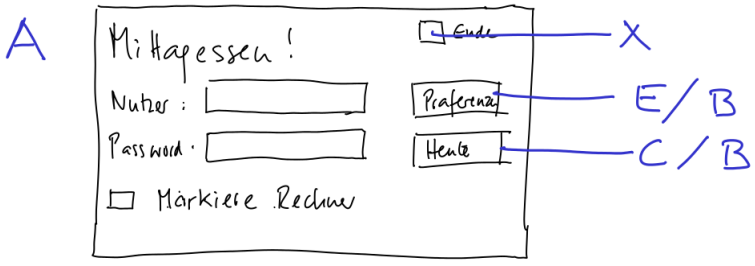


Abbildung 2.8: Bild A - Start der Applikation: sofern der Benutzer bekannt ist, geht man zu Schirm E oder C, sonst kann sich der Nutzer auf dem Schirm B nochmals identifizieren.

[8] Die Grossbuchstaben zeigen die Sprünge zwischen den Bildern an, X bezeichnet das Programm-Ende

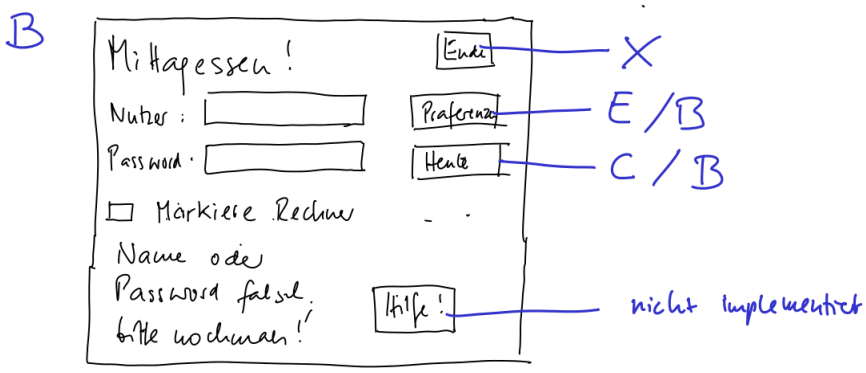


Abbildung 2.9: Bild B - Eingabe von Benutzername oder Passwort war falsch.

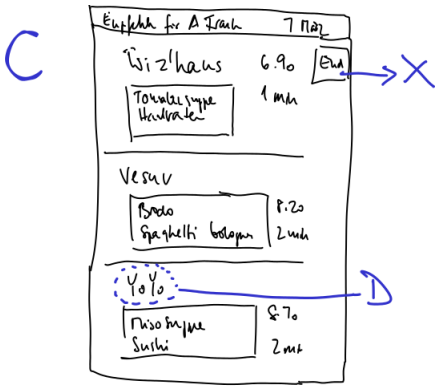


Abbildung 2.10: Bild C - Vorschlag für Mittagessen heute - mit Verweis auf Karte.

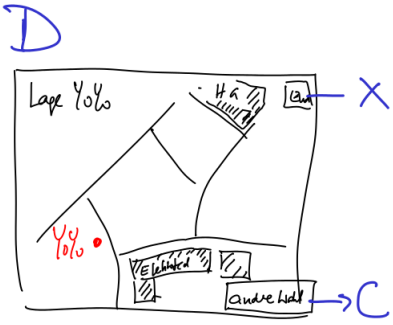


Abbildung 2.11: Bild D - Lage des Gasthauses.

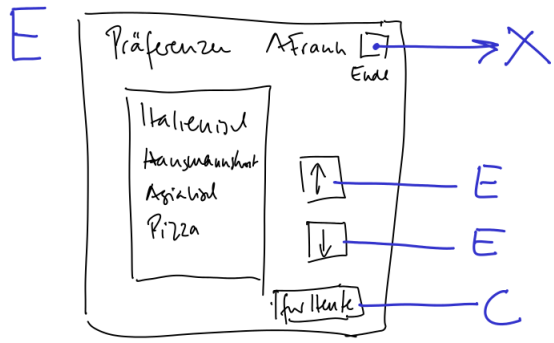


Abbildung 2.12: Bild E - Eingabe von Präferenz; die Pfeile erlauben einen Küchen-Typ höher oder niedriger einzustufen.

Test durch Simulation

Ist der Ablauf vollständig beschrieben, dann erlaubt das Zustands-Übergangs-Diagramm zu prüfen, dass alle Fälle, die ein Nutzer erwarten wird, enthalten sind. Das lässt sich bereits in diesen frühen Stadien prüfen und Fehler mit Radiergummi ausbessern; der Aufwand für Änderungen jetzt ist wesentlich geringer als wenn Fehler oder Lücken erst nach der Programmierung entdeckt werden!

Die Gestaltung des Ablaufes der Benutzer-Interaktion ist außerordentlich schwierig und es ist kaum möglich, die Logik der verschiedenen Pfade des Ablaufes im Blick zu halten. Es gilt also, den Ablauf in einem frühen Stadium zu testen, weil nur beim durchspielen der Abläufe Fehler aufgedeckt werden können.

Allgemein gilt, dass Fehler zu machen nicht vermieden werden kann, dass aber genügend Kontrollen eingebaut werden müssen, damit Fehler frühzeitig entdeckt und korrigiert werden können. Fehler die rasch aufgedeckt werden, verursachen geringe Kosten (2.13). Wird ein Fehler hingegen erst später entdeckt, so sind die Kosten oft sehr, sehr viel höher. Wenn Fehler im Interface entdeckt werden können, bevor die Anwendung programmiert ist, so vermindern sich Aufwand und Dauer der Erstellung der Anwendung.

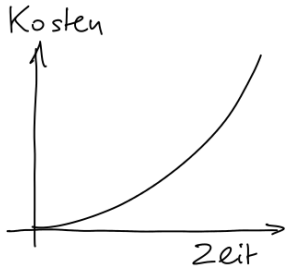


Abbildung 2.13: Die Kosten für die Korrektur eines Fehlers steigen je später er entdeckt wird

Spielen sie im Team den Ablauf mit verteilten Rollen durch: einer spielt den Benutzer, ein zweiter den Computer und ein dritter protokolliert. Dabei sollten die Rollen wechseln und versucht werden, ob ein besonders ungeschickter, unwissender, naiver oder bösartiger Benutzer das System austricksen kann. Es muss geprüft werden, ob durch Automatisierung oder Veränderung des Ablaufes die Benutzung für den Nutzer einfacher wird. Einfachheit der Benutzung ist nicht für alle Benutzer das gleiche; für einen geübten Benutzer, der die Applikation regelmäßig benutzt, ist es einfach, wenn Befehle kurz und Abläufe schnell sind; für einen „seltenen Gast“ sind Erklärungen wichtig. Man kann es nicht allen recht machen, sondern muss überlegen, was die Situation der Applikation ist—im Zweifelsfall entwirft man die Benutzerschnittstelle für den seltenen Benutzer. Das heißt, man reduziert, was der Benutzer wissen muss und vereinfacht; Abkürzungen für geübte Benutzer können dann zugefügt werden. Spezialfälle können allenfalls in speziellen Menüs „versteckt“ werden, die den Ungeübten oder technisch weniger Interessierten meist nur stören.

Prüfen sie
normaler Benutzer,
bösartiger Benutzer,
ungeschickter Benutzer, etc.

Wichtige Abkürzung: “enter” in einem Feld löst Aktion aus, ohne den Knopf zu drücken.

Der Test mit Papier und Bleistift verlangt zwei bis drei Personen, die verschiedene Rollen spielen.

- „Nutzer“, wählt eine Situation und gibt an, welche Eingaben er macht.
- „Computer“: simuliert die Antwort des Computers, d.h. er zeigt den nach dem Zustand-Übergangs-Diagramm folgenden Schirm und

gibt an, welche Felder wie gefüllt sind.

- „Protokoll“ notiert sich die Abläufe. Insbesondere die Unklarheiten, Lücken und Fehler, die entdeckt wurden, sind zu notieren, so dass sie korrigiert werden können; danach ist der gleiche Test nochmals zu durchlaufen.

Die Protokollierung kann von einem der beiden andern übernommen werden.

Die Rollen können getauscht werden. Es müsse auch spezielle Arten von Nutzen gespielt werden:

- sehr geübter Nutzer, dem es nicht schnell genug gehen kann. Sind Abkürzungen leicht einzubauen, ohne dass andere Nutzer verwirrt werden?
- naiver Nutzer, der die Texte wörtlich nimmt,
- unaufmerksamer Nutzer, der viele Angaben falsch macht und korrigieren muss. Das hilft zu erkennen, welche Tests notwendig sind, um Eingabefehler zu entdecken,
- böswilliger Nutzer, der mit Absicht falsche Eingaben macht, um die Anwendung zum Absturz zu bringen.

Die Protokolle über die Abläufe werden später beim Testen des programmierten Systems sehr nützlich sein. Verhält es sich so, wie geplant oder sind Fehler beim Programmieren entstanden?

Beispiel für ein Protokoll:

Testtyp	Ablauf	
Optimaler Ablauf	A: AF, pw - click heute	
	C: click YoYO	
	D: click End	
Mehrere Versuche	A: AF, pw - click heute	
	C: click YoYO	
	D: andere Wahl	
	C: click End	oder weitere Wahl
Nutzername falsch	A: XX,pw - click heute	
	B: AF,pw - click heute	fehlt Registrierung

Tabelle 2.2: Protokoll von Abläufen

Benutzeranleitung

Nun kann man die Benutzeranleitung schreiben. Braucht es überhaupt eine? Kann eine geschriebene Anleitung an die potentiellen Nutzer verteilt werden und werden sie sie beachten? Für Web-Anwendungen, besonders jene, die sich an einem unbestimmten Benutzerkreis wenden, sind im Allgemeinem keine Anleitungen möglich und die dem

Nutzer präsentierten Schirme, allenfalls mit einer „Help“ Funktion versehen (Übergang zu einem Hinweis-Schirm ist beim Entwurf des Benutzerschnittstelle zu berücksichtigen), müssen alle Informationen liefern, die der Benutzer wissen muss. Nützlich sind Tips, die beim überstreichen eines Feldes mit der Maus angezeigt werden - hier lassen sich einfache Erklärungen der Terminologie gut anbringen.

Lange Bedienungsanleitung schreckt vor Benutzung ab!

Für schwierigere, komplexere Anwendungen muss eine Anleitung angeboten werden, auch wenn die wenigsten Nutzer diese genau lesen werden. Beim Schreiben einer Anleitung sollte für jede Information, die man den Benutzer lernen lassen will, genau überlegt werden, ob sie zwingend notwendig ist oder ob es möglich ist, das Programm so zu verändern, dass es nicht nötig ist, dem Nutzer die Information zu vermitteln. Es ist „billiger“ etwas mehr bei der Programmierung aufzuwenden, dafür bei Erstellen, Verteilen, Lesen und Beantworten von Fragen auf der Help-Line einzusparen.

Denken sie auch an die Kunden, die durch die Bedienungsanleitung abgeschreckt werden oder durch Unverständnis oder Unwissen, weil sie die Bedienungsanwendung nicht gelesen haben, verärgert sind und das Produkt nicht nutzen werden![9]

[9] Es ist zwar verbreitet aber fördert das Geschäft nicht, auf Anfragen von Kunden diese zum Lesen der Bedienungsanleitung aufzufordern.

Zusammenfassung

Die genaue und detaillierte Dokumentation der Anwendung mit „Papier und Bleistift“ (und Radiergummi!) scheint aufwendig und oft sehr simpel. Das Ergebnis wird bei der Programmierung sehr nützlich sein, weil in der Phase des Entwurfes der Überblick vorhanden ist und sich später der Programmierer auf die Lösung der Details eines Schirmes konzentrieren kann. Wenn man versucht beide, Übersicht und Detail, gleichzeitig zu lösen, scheitert man meist mit beidem und der Aufwand ist viel größer bis mit „vor- und zurück“ alle Fehler ausgemerzt sind und etwas halbwegs Brauchbares entsteht. Generell gilt, Fehler sind umso einfacher zu korrigieren, je früher sie entdeckt werden (Figure 4).

Komplexität ist der Feind des Programmierers!

Der Prüfung der Möglichkeiten dass ein böswilliger Nutzer schaden stiften wird, ist schon in der Entwurfsphase einzugrenzen, so dass sie nachher mit technischen Vorkehrungen zu unterbinden sind. Die Darstellung “Schirm für Schirm” zeigt , was an Graphik, Fotos und Text notwendig ist und hilft, unbeabsichtigte Verwendung von Material, für das man nicht die notwendigen Rechte hat, zu vermeiden. Die Darstellung Schirm für Schirm hilft, schon jetzt zu erkennen, was die Anwendung genau anbieten kann (und was nicht!), und damit zu prüfen ob alle Versprechungen in der „Allgemeinen Beschreibung“ (2.1.13) erfüllt sind.

Die Beschreibung der Benutzerschnittstelle ist ein wesentlicher Teil des Auftrages und gibt eine genauer, detaillierte und nachvollziehbare

Beschreibung was der Auftragnehmer erwartet. Es empfiehlt sich, die Beschreibung mit dem Auftraggeber zu besprechen, durchzuspielen und am Schluss mit Unterschritt bestätigen zu lassen. Das vermeidet später Diskussion und erlaubt zu zeigen, dass das abgelieferte Produkt dem Auftrag entspricht und damit der ausgemachte Preis dafür geschuldet ist.

Betriebssystem verwaltet Hardware

Das Internet verbindet Rechner. Ein Rechner, das heisst die Hardware, besteht aus dem Prozessor (CPU), Speichermedien und Peripheriegeräten. Zur Steuerung dient ein Betriebssysteme, dass den laufenden Programmen Ressourcen zuteilt.

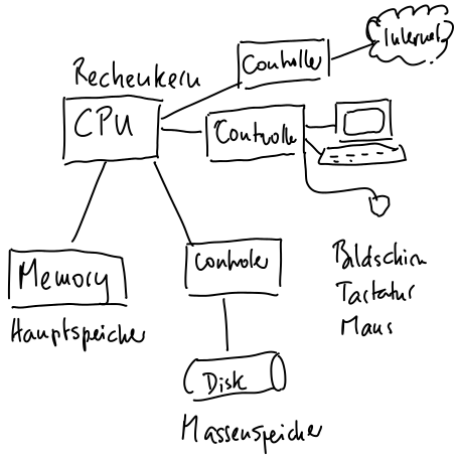


Abbildung 2.14: Das Betriebssystem verwaltet die Hardware
Prozess = laufendes Programm

Betriebssystem

Im Server und Client Rechner muss ein Betriebssystem für einen geordneten Ablauf sorgen. Das Betriebssystem verwaltet die Ressourcen des Rechners, das sind:

1. Prozesse (d.h. laufende Programme),
2. Hauptspeicher (RAM, Memory),
3. Prozessorzeit (d.h. Verwendung der CPU durch einen Prozess),
4. Platz im Dateisystem zur Speicherung von Daten und Peripheriegeräte,
5. Interrupts (kurzfristige Unterbrechungen eines Programmes).

Das heute weit-verbreitete und für Server bevorzugte Betriebssystem LINUX ist ein Abkömmling des UNIX System (Bell Labs von AT&T, etwa 1970 [10]), das seinerseits von MULTICS (MIT ab 1963) abstammt.

UNIX besticht durch wenige einfache Konzepte, die sehr breit angewendet werden;

- alles ist eine Datei (file): Tastatur, Maus, Zugang zum Netz etc. sind alles auf ein allgemeines Datei-Konzept abstrahiert;
- kleine Programme (tools), die einfache Funktionen auf Text Dateien [11] ausführen und fast beliebig zusammengefügt werden können.
- in einer höheren Programmiersprache geschrieben: Unix war wahrscheinlich das erste Betriebssystem das nicht in Assembler sondern einer höheren Programmiersprache geschrieben wurden und damit leicht auf verschiedene Rechner zu übertragen.

Diese Familie der UNIX Betriebssystemen wurde über die letzten 40 Jahre weiterentwickelt; Tanenbaum publizierte eine vereinfachte

[10] Dennis M Ritchie and Ken Thompson. The unix time-sharing system. *Communications of the ACM*, 17(7): 365–375, 1974

Sehr nützliche weiterführende Informationen über Betriebssysteme und insbesondere über LINUX findet sich in Wikipedia unter den Schlagworten UNIX, UNIX Philosophie, etc.

[11] genauer: Dateien, die nur Text im ASCII Code enthalten

Assembler = symbolische Schreibweise für Befehle, die die CPU ausführen kann
Die heute noch beliebte Sprache C wurde dafür erfunden
Brian W Kernighan, Dennis M Ritchie, and Per Ejeklint. *The C programming language*, volume 2. prentice-Hall Englewood Cliffs, 1988

Version von UNIX als Universitätslehrbuch [12] mit dem ausführlichen Programmtext. Das Interface zum UNIX Betriebssystem ist als POSIX standardisiert. Linus Torvalds (2.16)machte 1991 eine Version für den damals populären IBM PC. und erlaubt, mehreren Benutzern gleichzeitig einen Rechner zu benutzen.

[12] Andrew S Tanenbaum. Minix operating system. *Japan, Apr*, 21:328, 1989

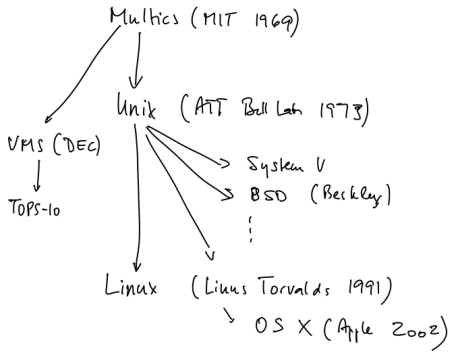


Abbildung 2.15: Entwicklungsgeschichte von LINUX

2.1.15 Betriebssysteme mit Kern

LINUX und ähnliche moderne UNIX Derivate bestehen aus einem Kern (Kernel, im Moment brauche ich LINUX 3.11.2) und (Dienst-) Programm darum herum.

Der Kern verwaltet die Ressourcen des Systems und erlaubt die Nutzung durch die laufenden Programme, die als Prozesse bezeichnet werden (2.17). Diese Prozesse sind einerseits Programme des Benutzers, aber bedienen, z.B. die Peripheriegeräte. Es wird vom Betriebssystem jeweils einem Prozess die Nutzung der (oder einer) CPU für eine bestimmte, sehr kurze, Zeitspanne zugeteilt. Danach wird die Rechenleistung der CPU einem andern Prozess, der zur Ausführung bereit ist, zugeteilt und so weiter, im Kreis herum.

Abbildung 2.16: Linus Torvalds, Schöpfer des Linux Kernels

englisch: round robin scheduler

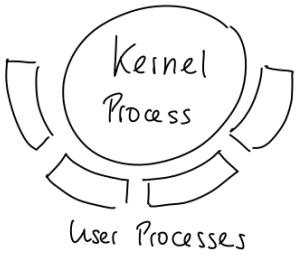


Abbildung 2.17: Aufbau eines UNIX-artigen Betriebssystems

Prozessorleistung

Ein Prozessor kann zu einer Zeit nur für einen Prozess arbeiten. Damit jeder dran kommt, wird abgewechselt. Allerdings gibt es Prozesse, die nicht warten können. Laufende Prozesse müssen unterbrochen werden (Interrupt), damit die dringlichen Prozesse rasch erledigt werden können und dann kann der erste weitergeführt werden.

Für uns ist wichtig, dass wir prüfen können, ob die für uns wichtigen Prozesse aktiv sind . Interessant ist allenfalls auch zu sehen, wie stark der Prozessor beschäftigt ist.

z. B. mit UNIX Befehl *ps*, meist gibt es aber eine benutzerfreundliche, grafische Oberfläche „System Monitor“

UNIX Befehl *top*, aber meist auch in „System Monitor“ erkennbar

Moderne Rechner verfügen oft über mehrere Rechenkerne wobei ein Prozess jeweils einem Kern zur Ausführung zugewiesen wird. Die Geschwindigkeit der Rechenoperationen eines Kerns kann seit etwa

multi core computer

einer Dekade nicht mehr leicht gesteigert werden und es ist einfacher, mehrere Kerne auf einen Chip zu plazieren; diese können dann Arbeiten parallel ausführen, sofern die Programmierung darauf ausgelegt ist - was heute für die meisten üblichen Programme nicht der Fall ist (2.1.20).

Speicher

Die Daten, die vom Prozessor verarbeitet werden, müssen vor und nach der Verarbeitung gespeichert werden. Dazu dienen verschiedene elektronische Bauteile, die unterschiedliche physikalische Phänomene ausnützen, um verschiedene Zustände darzustellen und aufzubewahren.

Speicher wird im allgemeinen in einer Hierarchie verbaut, so dass teurer, sehr schneller und kleiner Speicher näher an der CPU sitzt und Daten aus dem langsameren, aber billigeren Speicher nach Bedarf dorthin transferiert wird. Langfristige Speicherung von Daten geschieht im um einen Faktor (z.b. 5) billigeren Festplattenspeicher, der aber auch etwa 1000 mal langsamer ist.

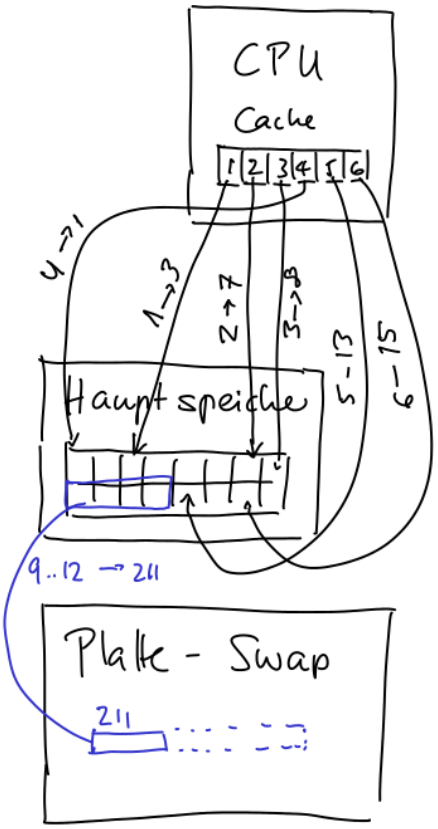


Abbildung 2.18: Speicherhierarchie

swapped out

2.1.16 Hauptspeicher

Das Betriebssystem verwaltet den Hauptspeicher und teilt Teile davon den laufenden Prozessen zu und übersetzt die lokalen Speicheradressen des Programmes in die Hardware-Adressen des Speichers (2.18). Wollen aktive Prozesse mehr Hauptspeicher als physisch vorhanden ist, so werden Teile davon auf den Plattenspeicher ausgelagert ; das verlangsamt den Wechsel zwischen Prozessen, weil zuerst der ausgelagerte Teil des Speichers wieder in den Hauptspeicher geladen werden muss, um mit der Verarbeitung weiterfahren zu können. Wesentlich für uns ist nur zu sehen, ob genügend Hauptspeicher vorhanden ist und ob der Swap Bereich benutzt ist. Allenfalls will man noch wissen, welche Prozesse wie viel Speicher beanspruchen. Beides ist mit dem “System Monitor” möglich.

Hauptspeicher ist sehr schnell und Daten können innerhalb von Nanosekunden gelesen werden; Hauptspeicher ist aber relativ teuer und nur eine beschränkte Menge kann von der CPU adressiert werden. Hauptspeicher ist flüchtig, d.h. nach einem Stromausfall ist der Inhalt verloren.

Hauptspeicher Zugriff typisch 100 Nanosec. (10^{-7})
Plattenspeicher Zugirff typisch 10 Millisec. (10^{-2})

2.1.17 Plattenspeicher

Grössere Datenmengen die langfristig verwendet werden sollen, werden deshalb auf Festplatten-Speicher (2.19) abgelegt. Zugriffszeiten sind grob 100,000 mal länger als für den Hauptspeicher aber der Inhalt geht bei Stromausfall nicht verloren.

Abbildung 2.19: Plattenspeicher

Peripheriegeräte

Alles was an den Rechner angeschlossen ist, muss dem Kern bekannt sein und wird von ihm bedient. In der UNIX Denkweise werden alle Geräte auf eine Datei abgebildet und mit einem Dateinamen (device name) angesprochen.

Das umfasst :

- Bildschirm,
- Tastatur,
- Maus (2.20) (oder ein anderes Gerät zum steuern des Zeigers),
- Internetanschluss,
- Drucker,
- etc.

File = Datei (englisch)

Abbildung 2.20: Original Maus erfunden von Douglas Engelbart 1963 (2.21)

Abbildung 2.21: Douglas Engelbart, Erfinder der Maus (geb. 1925)

2.1.18 Interrupts

Wenn Daten von einem Peripheriegerät oder einer Festplatte angefordert werden, so kann die CPU an etwas anderem arbeiten, bis die Daten eintrudeln. Wenn die Daten aber bereit sind, so muss sehr rasch reagiert werden und die Daten "übernommen" werden. Das wird durch einen Interrupt, der vom Peripheriegerät ausgelost wird, gesteuert. Der Prozess der an der Arbeit ist, wird kurzfristig unterbrochen, die Daten gesichert und weitergearbeitet. Etwas später wird dann der Prozess, der auf die Daten gewartet hat - und jetzt nicht mehr wartet, weil die Daten ja bereit sind - angestossen und verarbeitet die Daten.

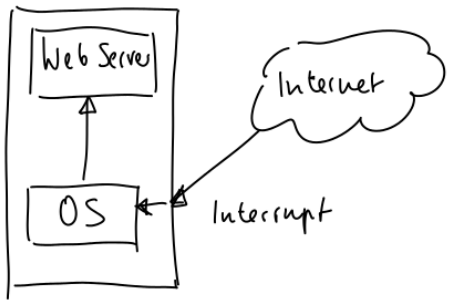


Abbildung 2.22: Interrupts vom Web werden vom Betriebssystem and den Webserver weitergegeben

Architektur einer Web-Applikation

Mit dem Wort Architektur wird im EDV Jargon eine Übersicht über die technischen Komponenten einer Anwendung und deren Verbindungen gegeben. Ein Architekturdiagramm gibt Auskunft über die Fragen: Welches sind die Komponenten? Welche Aufgaben übernehmen sie? Wie wirken sie zusammen?

Der konzeptionelle Konflikt zwischen statischem Aufbau (Grundriss der Wohnung), Ablauf (Wege in der Wohnung) und Ergebnis (Wohlfühlen in der Wohnung) ist schwierig zu überbrücken.

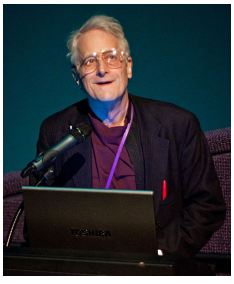


Abbildung 2.23: Ted Nelson, der Hypertext Guru (geb. 1937)

Das Web als Sammlung verlinkter Dokumente

Ein geschriebener oder gedruckter Text ist linear, er hat einen Anfang, eine Folge von Sätzen bis zum Ende. Ein von einem Rechner produzierter Text kann Verzweigungen enthalten und dem Nutzer erlauben,

den Text in einer von ihm gewählten Reihenfolge zu lesen - etwa so, wie man in einem Lexikon eine Idee durch mehrere Schlagwort-Text verfolgt und die interessierenden Verweise weiterverfolgt.

Ted Nelson hat für diese Idee (schon vorher von den Schriftstellern Borges in “El jardín de los senderos que se bifurcan” und Vannevar Bush angelegt) den Begriff *Hypertext* verwendet; in einem Hypertext sind *Hyperlinks* eingebettet, mit denen der Leser den Fluss steuern kann und auswählt, was er als nächstes lesen will. Dass kann eine Verzweigung in einer Geschichte zur Auswahl eines dem Leser richtig scheinenden Schlusses zu sein oder wie in einem Lexikon ein Verweis auf einen andern Text, der etwas zugehöriges erklärt.

Computerunterstützte Systeme für Text, aber bald auch für eine Fahrt durch Aspen (Colorado), das aus gefilmten Sequenzen zusammengesetzt war und bei jeder Kreuzung konnte der Benutzer wählen, wie er “weiterfahren”wollte und die entsprechende Filmsequenz wurde gezeigt [13]. Apple vertrieb nach 1987 ein populäres Programm HyperCard für den Macintosh beim dem kurze Texte durch Hyperlinks verbunden werden konnten.

Tim Berners-Lee (2.24) übertrug das Hypertext Konzept etwa 1990 auf das damals bestehende Internet, das fast ausschliesslich für den Transfer von Dateien ausgelegt war; er schuf ein Protokoll, nachdem in Dokumenten Verknüpfungen auf andere im Web gespeicherte Dokumente möglich waren. In älteren Hypertext-Dokumenten waren nur Verknüpfungen innerhalb eines Rechners möglich, Berners-Lee öffnete die Links für alles, was auf dem Internet erreichbar war - das World Wide Web war erfunden!
[14]

Das World Wide Web besteht aus Dokumenten, d.h. in Rechnern gespeicherten Texten, die eindeutige Bezeichnungen haben (siehe ...). In Dokumenten können Verweise (dh. Bezeichner) auf andere Dokumente enthalten sein.

[13] [ref hooper]

Macintosh war der erste weit verbreitete Computer mit einem graphischen “Windows” Benutzerinterface und einer Maus, aufbauend auf den Entwicklungsarbeiten von Xerox in ihrem Forschungszentrum Xerox PARC

Abbildung 2.24: Tim Berners-Lee 2009

[14] Eine alternative Lösung, das “Gopher Protokoll” [http://en.wikipedia.org/wiki/Gopher_%28protocol%29 vorgeschlagen von Mark P. McCahill, war in den 1990 Jahren populär, weil es angeblich einfacher (weil hierarchische Organisation der Vernetzung), weniger Resourcen verbrauchend und leichter aufzusetzen war. Unglücklicherweise versuchte die Universität von Minnesota, die das Programm erstellt hatte, Mitte 1990 dafür Lizenzgebühren einzuheben, was das weitere Wachstum stoppte.
WWW = alle im Internet verlinkten Dokumente

Client-Server-Architektur

Für eine Internet-Anwendung für GIS geht man typischerweise von einer Client-Server Architektur wie in Figure 2.25 aus:

Ein Nutzer interagiert mit seinem Computer, der über das Web (Internet) mit einem Server verbunden ist. Der Rechner des Nutzers dient zur Darstellung der Daten, die vom Server geliefert werden, und für die Annahme der Eingaben des Benutzers. Die Übermittlung der Daten erfolgt über den Technologie-Komplex der als Internet bezeichnet wird. In manchen Fällen sind mehrere zusätzliche Server beteiligt, was aber an der grundsätzlichen Struktur nichts ändert (2.26). Im Folgenden konzentrieren wir uns auf den einfachen Fall

(Figure 2.25).

Im überblick zeigt die Architektur einer Web-Anwendung drei Komponenten:

Server Ein Rechner (oder mehrere im Verband), der die Daten für eine Anwendung speichert, aufbereitet und über ein Programm zum Client überträgt.

Internet Alle Rechner sind zur Kommunikation verbunden sind; diese Verbindungen bilden ein Netzwerk fast beliebiger Topologie. Nachrichten werden darin zwischen zwei Rechnern auf einem effizienten Pfad übertragen. Die Übertragung werden meist nach den Protokollen TCP/IP abgewickelt .

Client Der Rechner des Nutzers verwaltet die Verbindung zum Server durch das Internet, nimmt Eingaben des Nutzers entgegen und kommuniziert diese an den Server und stellt im Gegenzug die Daten, die der Server übermittelt, graphisch dar (beziehungsweise als Ausgabe in einem anderen Medium, z. B. Sprache oder Musik über den Lautsprecher).

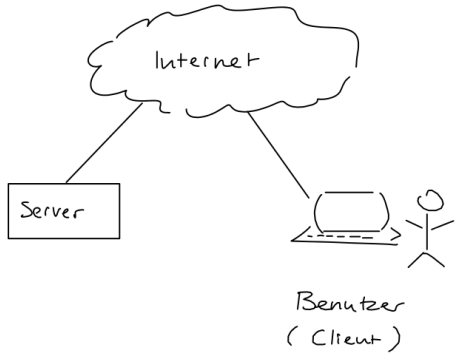


Abbildung 2.25: Benutzer nutzt Applikation auf einem Server

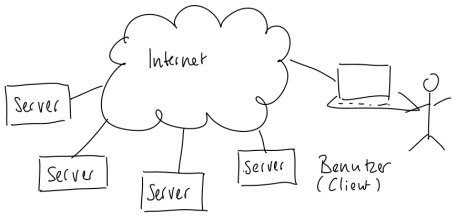


Abbildung 2.26: Architektur mit mehreren Servern und Nutzer

Web-Server und Web-Browser

Für den Zugang zum Web sind spezielle Server und Clients notwendig, die meist auf das Internet Protokoll TCP/IP spezielle Verfahren des Datenaustausches aufsetzen .

Web-Server versenden auf Anfrage Dokumente, die auf diesem Rechner gespeichert sind.

Web Das World Wide Web (www oder kurz web) ist eine immense Sammlung von verlinkten Dokumente.

Web-Browser zeigt Dokumente, die von einem Web-Server erhalten wurden, an und erlaubt dem Benutzer die in diesen Dokumenten enthaltene Verknüpfungen zu verfolgen; die vom Nutzer gewünschten Dokumente werden dann über das Internet vom im Link genannten Web-Server geholt und angezeigt.

Das Internet verbindet heute praktisch alle Rechner weltweit und viele davon enthalten einen Web-Server (Abschnitt §4.3). Diese halten die verlinkten Dokumente und erlauben es Nutzern den Links zu folgen. Die technische Installation ist komplex aber im Wesentlichen unsichtbar für den Nutzer und die meisten Nutzer wissen wenig über die Details („transparent“).

Web-Applikation Eine Web-Applikation ist ein Start Dokument, dessen Link dem Benutzer bekannt sein muss und in dem Verknüpfungen zu andern Dokumenten enthalten sind. Eine Web-Applikation

ist also zuerst eine Sammlung von Dokumenten; in diese Dokumente sind dann allenfalls Methoden für den Zugriff auf Daten und verarbeitende Programme eingebunden.

Verbindung Programmierung und Darstellung in Web Anwendungen

2.1.19 Model-View-Controller

Die Verbindung zwischen dem Rechner, indem ein Modell eines Ausschnittes der Realität dargestellt und allenfalls nachgeführt wird und dem Benutzer, der mit diesem Modell in einen Dialog tritt, ist schwierig und gehört zu den weniger verstandenen Teilen der Gestaltung von Computer-Anwendungen. Die Schwierigkeit liegt in der Notwendigkeit Bedeutung wie Menschen sie verstehen, in Programmen abzubilden.

Der Komfort und die "Natürlichkeit" eines Dialoges in einer Anwendung sind zum Gradmesser der Qualität geworden. Ursprung war der Benutzer durch die Anwendungen kontrolliert: der Rechner (d.h. das Anwendungsprogramm) führt den Dialog und stellt dem Benutzer Fragen.

Seit der Entwicklung der Programmiersprache Smalltalk [15] und der Rechner mit graphischem Bildschirm und Maus, wählt man Methoden, die dem Nutzer mehr Kontrolle über den Ablauf einräumen und ihm mehr Freiheit in der Reihenfolge der Arbeitsschritte überlasst. Der Benutzer kann Arbeiten anstossen und muss nicht, wie vorher, mit dem nächsten Schritt warten, bis der Rechner den ersten vollständig abgearbeitet hat.

[15] A. Goldberg and D. Robson. The smalltalk-80 system. *BYTE*, 6(8): 14ff., 1981; and Adele Goldberg and D. Robson. *Smalltalk-80*. Addison-Wesley Publ. Co., 1983

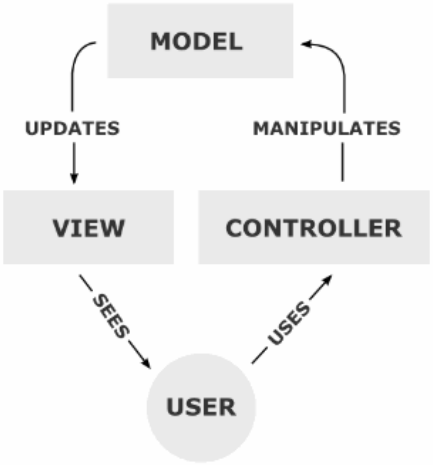


Abbildung 2.27: Model-View-Controller

In der Darstellung des Ablaufes bei der Benutzerschnittstelle sind die Arbeitsschritte und deren Reihenfolge festgelegt und diese muss nun in die Programmierung übersetzt werden. Dazu wird konzeptionell die Sicht (View) vom Modell getrennt und durch das Kontrolprogram verbunden (2.27).

In einem Programm werden Daten verarbeitet die ein Modell darstellen. Diese werden graphisch präsentiert, damit der menschliche Nutzer sie verstehen kann. Der Rechner bearbeitet das Modell und Veränderungen müssen in der Sicht möglichst rasch geändert werden. Gleich verändert der Nutzer mit Befehlen seine Sicht und diese müssen in Anweisungen an den Rechner, das Modell zu verändern; daraufhin werden diese Änderungen als Rückmeldung an den Nutzer in der Sicht des Nutzers nachgeführt.

Es wird empfohlen, zwischen dem Programm, indem die Daten ein Modell eines Ausschnittes der Realität repräsentieren und die Verarbeitungsschritte meist reale oder geplante Tätigkeiten in der Welt darstellen, einerseits und der visuellen Darstellung dieses Modells für den Benutzer zu unterscheiden. Zwischen diesen beiden Elementen ver-

mittelt ein Controller, der die Aktivitäten des Benutzers am Interface in Operationen am Model umsetzt und anderseits die Änderungen am Modell für den Benutzer sichtbar macht[16].

[16] http://en.wikipedia.org/wiki/Model%E2%80%93view
Model-View-Controller

2.1.20 Reaktive und parallele Programmierung

Die Schwierigkeit der Programmierung von interaktiven Anwendungen fällt nicht sofort auf. Sie liegt in dem asynchronen Ablauf, d.h. mehrere Aktionen können gestartet werden und quasi gleichzeitig laufen; andere Aktionen werden angestoßen, führen aber nicht sofort oder gar nie zum gewünschten Ergebnis kommen. Z.B. verlangt ein Nutzer Daten von einer Datenbank und seine Anwendung wartet auf die Antwort; diese trifft aber nie (oder stark verspätet) ein, weil die Datenbank nicht am Netz ist, außer Betrieb oder heftig überlastet. Gleichzeitig geht vom Programm eine Anfrage aus, die beim Nutzer dargestellt werden muss. (Concurrency)

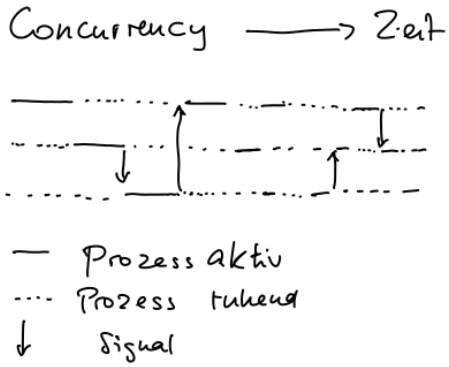


Abbildung 2.28: Nebenläufigkeit
engl. time out

Ist die Anwendung schlecht programmiert, so wartet sie “ewig”, bis der Nutzer sie beendet (z.b. durch Schließen des Web Browser Fensters); bessere Anwendungen warten eine Weile und teilen dem Nutzer dann mit, dass seine Aktion nicht durchgeführt werden konnte. Unbefriedigend, aber immerhin kann etwas anderes angestoßen werden.

Benutzer erwarten heute, dass sie die Anwendung kontrollieren und deshalb nicht warten wollen, bis eine längerdauernde Aktion zum Abschluss kommt, sondern sofort nach dem Anstoßen der einen eine andere anstoßen können.

Reaktive Programmierung ist in den heute üblicherweise gelehrten Programmiersprachen (Pascal, Java, C++ etc.) nicht im Basisumfang eingeschlossen. Ein normaler Prozeduraufruf ist “blocking”, d.h. es geschieht nichts bis die aufgerufene Aktion beendet ist.

Manche Programmiersprachen sind erweitert, so dass Prozeduren “non-blocking” aufgerufen werden können - der Benutzer kann andere Aktionen setzen, während auf die Ergebnisse der ersten Aktion gewartet werden. [17]

[17] In Javascript wird einem Aufruf deshalb häufig der Code der Aktion, die nach diesem Aufruf erledigt werden soll, mitgegeben und der Call ist ’non-blocking’, d.h. die nächste Anweisung im Programm wird ausgeführt.

HTML im Webbrowser bietet Hilfsmittel, um rasch reagierende Anwendungen mit Javascript zu gestalten: anklicken von Buttons auf der Seite löst Aktionen aus, die nicht-blockierend ablaufen und andere parallel ausgeführte Aktionen zulassen.

2.1.21 Praktische Einschränkungen

Optimal würde die graphische Darstellung der Seite mit allen Elementen, die Benutzerinteraktionen auslösen, d.h. die ganze Benutzersicht statisch mit einem Hilfsmittel zu Beschreibung einer graphischen Oberfläche beschreiben und diese mit einer Beschreibung des Ablaufes in einer Programmiersprache verbunden. Idealerweise wäre die stati-

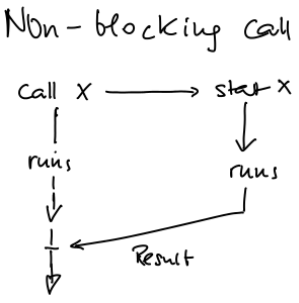


Abbildung 2.29: Non-blocking Prozessaufruf

sche Beschreibung und der Ablauf je für verschiedene Architekturen verwendbar, d.h. mit den gleichen Beschreibungen und Programmen könnte man Web-Applikationen oder Native Applications produzieren. Die grundsätzliche Schwierigkeit besteht in der notwendigen Anpassung der Benutzersicht an verschiedene Grössen des Gerätes. Es reicht bei weitem nicht aus, die Darstellung für die Grösse des Bildschirmes masstäblich zu vergrössern oder zu verkleinern: Schrift muss eine bestimmte Minimalgrösse nicht unterschreiten und zu gross ist ebenfalls unpassend; wenn der Schirm gross ist, so sollten mehrere Informationen gleichzeitig angezeigt werden, auf einem kleinen Schirm muss das in einzelne Happen zerlegt werden - das ist bisher nicht automatisch möglich, weil es ein Verständnis des Inhaltes und der Ziele des Benutzers erfordert.

Ein Entwurf einer Interaktion unabhängig vom Gerät mit einem hilfsmittel ist bisher leider nicht möglich. Ich verwende z.B. für den Entwurf einer Benutzerschnittstelle den GLADE Interface Designer [18], das zu Gtk Computer-Graphik-Umgebungen passt. Als Ergebnis liegt eine Beschreibung der graphischen Oberfläche in einer in XML kodierte Datei vor. Diese kann vom Programm, das den Ablauf steuert, eingelesen werden, z.b. durch Laden der webglade [19] Bibliothek in Javascript Code. Verschiedene andere Methoden sind verfügbar, keine lösen die Aufgabe kompromissfrei.

[18] http://en.wikipedia.org/wiki/Glade_Interface_Desig

[19] http://logand.com/sw/webglade/

Unglücklicherweise ist es mir bisher nicht gelungen, die in XML eingepackte Javascript mit Webglade Umgebung mit Google Maps in Verbindung zu bringen; diese elegante Lösung mit GLADE funktioniert also noch nicht, auch wenn Google Maps seinerseits das Modell-View-Konzept propagiert und anwendet.

Nach dem Konzept von Modell-View-Konzept bleibt den Applikationsprogrammierer die Beschreibung des Modells, d.h. der Daten der Anwendung und der Operationen, die auf diesem Modell vom Nutzer ausgeführt werden können. Dazu wird die in der Benutzersicht festgelegten graphischen Aktionen des Benutzers, z. B. ein Klick an einer Stelle oder das Drücken eines Knopfes, in entsprechende Operationen an Applikationsdaten zu übersetzen.

Alternativen zu Server - Client Web-Architektur: Native Applications

Es ist natürlich möglich, Anwendungen zu programmieren, die beim Klienten lokal gespeichert und dort ausgeführt werden und aus der Applikation (nicht aus dem Browser) auf Daten im Web zugreifen (2.30). Das hat vor und Nachteile:

Vorteile:

- Relative Sicherheit - die Applikation ist von einem Programmierer

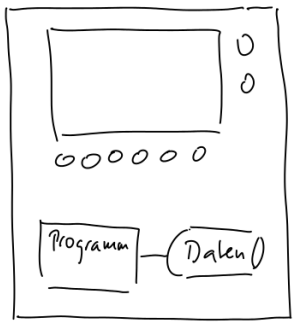


Abbildung 2.30: Native Application - Programm und Daten sind im mobilen Gerät (z.b. Smartphone) gespeichert

erstellt, dem wir vertrauen.

- Schneller, weniger Ressourcen-Verbrauch (weil kompiliertes Programm, nicht im Browser interpretiert).
- Applikations-Code kann kostenpflichtig verteilt werden (Verkauf von Lizenzen).
- Applikation kann auf Funktionen der Hardware zugreifen, die aus dem Browser nicht zugänglich sind (z.B. das GPS des Mobiltelefons)

Nachteile:

- Die Applikation muss lokal installiert werden; eine Installationsroutine muss zuerst herausfinden, was auf dem Rechner des Klienten alles vorhanden ist, das genutzt werden soll und den Programmcode daran anpassen. Relativ einfach in kontrollierten Systemen (Apple, iPhone, Android) und Linux (weil offen), schwierig unter Windows (viele verschiedene Versionen, kommerziell geschlossen).
- Programmierung ist nicht einfacher als für Browser, meist spezifisch für das Betriebssystem, auf dem der Code laufen soll. Entwicklung ist viel näher an der Hardware und muss für jede Hardware neu gemacht werden.
- Braucht spezifische Entwicklungsumgebung und Hardware spezifische Kenntnisse (2.31).
- Applikation läuft nicht bei jedem potentiellen Nutzer in seinem Browser, sondern nur auf Systemen, für die sie entwickelt wurde und installiert ist.

Eine “Native Application” wird nicht auf dem Zielgerät, was z.B. ein Smartphone ist, erstellt; dazu ist der Zugang zum Schreiben von Programm-Text auf dem Zielgerät zu beschränkt, die Speicherkapazität zu klein und die Rechnerleistung zu gering. Die Programme werden auf einem Rechner mit normaler Ausstattung (großer Bildschirm, Tastatur etc.) erstellt, dort umgewandelt, so dass sie auf dem Zielrechner ausgeführt werden können (2.31). Die Umgebung, die man auf dem Rechner für die Entwicklung einer Anwendung für ein Smartphone wird oft als SDK bezeichnet.

sogenannte SDK, Software Development Kit - Hardware spezifisch

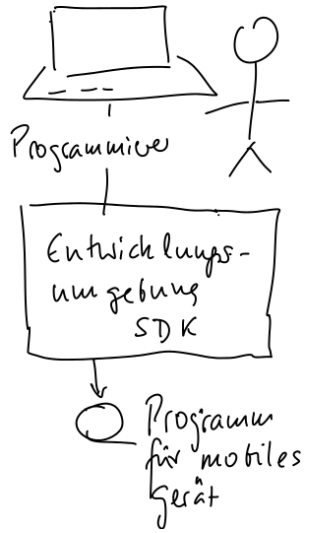


Abbildung 2.31: SDK

“cross compilation”, es wird auf einem Rechner kompiliert aber für die Ausführung auf einem andern

SDK Software Development Kit

Verteilung Aufgaben Client - Server

Die klassische Aufteilung von Web Applikationen war, dass der Browser einzelne Seiten von einem Webserver aufruft, was dem Hypertext Konzept entspricht. Alle “interessanten” Operationen finden beim Server statt und der Klient zeigt nur fertige Seiten an.

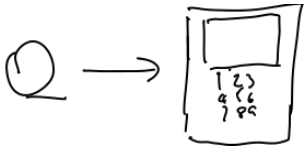


Abbildung 2.32: Laden einer Native Application

Web zum Darstellen von Dokumenten

Beim Klienten ist nur Rechenleistung für die Darstellung - was bei den frühen Web Text Seiten gering war und für die geringe Leistung der PC der 1990er ein Vorteil. Der Nachteil ist, dass die Reaktionen der Applikation auf Eingaben des Nutzers immer einen "round-trip" vom Klienten zum Server und zurück erfordern; diese Zeitverzögerung fällt heutigen Nutzern unangenehm auf, weil sie von lokalen ("native") Applikationen sofortige (dh. weniger als 1/10 sec.) Reaktionen erwarten. Im Web kann es schon Sekunden dauern, bis die Antwort vom Server zurück ist.

latency

Man nennt Klienten, die nur Seiten im Browser anzeigen "light clients", da sie auch auf sehr schwachen Rechnern laufen. Ihr wesentlicher Vorteil ist nicht vor allem die reduzierten Anforderungen an den Rechner, sondern vor allem, dass beim Nutzer nichts installiert werden muss.

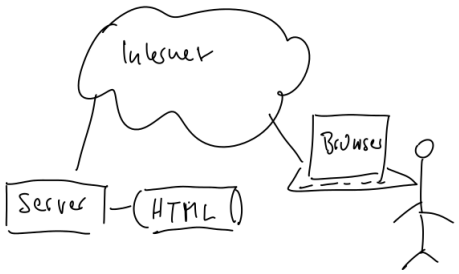


Abbildung 2.33: Web 1.0 mit HTML

Mit den viel stärkeren Rechnern heute, ist es möglich, wesentliche Aspekte des Controller-Code beim Benutzer im Browser ablaufen zu lassen. Das setzt voraus, dass im Browser Code in einer Programmiersprache ausgeführt werden kann. Seit etwa der Mitte der 1990 Jahre kann ein Web Server Seiten mit eingebettetem Javascript Code an einen Browser senden und erwarten, dass dieser den Code ausführt (Netscape seit 1994, Internet Explorer seit 1996) (2.34). Dies erlaubt, mehr und intelligentere unmittelbare Reaktionen beim Klienten, ohne die Verzögerung, die durch eine Übermittlung und Reaktion beim Server entsteht und dennoch nichts beim Nutzer installiert als der Browser, der Javascript ausführen kann, werden muss.

Interaktives Web

Die Lösung mit Code, der in den Web Seiten eingebettet ist, hat sich im Wesentlichen durchgesetzt und löst andere Verfahren, Programm-Logik zum Klienten zu bringen ab. Verbreitet waren und sind noch für spezielle Zwecke Plug-Ins; das sind Erweiterungen des Browser durch Programme, die in einem speziellen Format hochgeladen und in den Browser eingebaut werden müssen und dann (oft nach einem Neu-Start des Browsers) diesen mit zusätzlichen Funktionen erweitern. Plug-ins sind notwendig, wenn sie mit allen Seiten, die in einem Browser angezeigt werden sollen, funktionieren müssen.

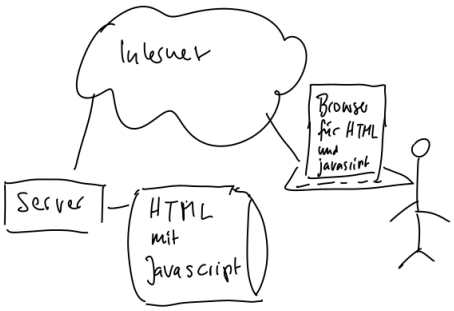


Abbildung 2.34: Web 2.0 mit Javascript
Zum Beispiel, die Erweiterung des Firefox Browsers mit dem firebug plug-in erlaubt ein komfortables Untersuchen von Web Funktionen und das Aufspüren von Fehlern in Web Seiten und deren Javascript Programmen; damit das für beliebige Seiten funktioniert, muss es im Browser geladen werden. Ähnlich funktionieren Add-Blocker plug-ins, die lästige Werbeeinschaltungen in Seiten blockieren.

Überlegungen zur Sicherheit der Anwendung

Zwei Gesichtspunkte sind bei einer normalen Client-Server Applikation zu berücksichtigen:

2.1.22 Die Sicherheit der Anwendung auf dem Server:

Welche Zugriffe könnten einen Schaden verursachen indem sie Daten löschen oder verfälschen. Manche Anwendungen leiden unter dem Speichern von unerwünschten Daten; z.B. legen manche Nutzer Re-

klametexte ab, die mit der Anwendunge in keinem Zusammenhang stehen.

Es ist zu prüfen, ob Zugriffe von außen die Anwendung (oder Teile davon) verändern könnten, so dass dem Kunden nicht die geplante Information in der gewünschten Form geliefert wird. Es kann z.B. versucht werden, Schadcode in die gespeicherten Webseiten einzubauen, der dann beim nichts-ahnenden Kunden Schaden anrichtet und bei ihm ärger über die Anwendunge verursacht.

2.1.23 Die Sicherheit des Rechners des Kunden.

Welche vom Kunden unerwünschte Vorgänge belastet die Anwendung auf seinem Rechner an. Vieles unerwünschte ist mit einer Webapplikation möglich und die Kunden sind sensibilisiert durch wiederholte Information über Schäden, die durch Anwendungen erfolgt sind.

Es muss also erstens dem Kunden versichert werden, dass der Hersteller die Anwendung kontrolliert und gegen Verfälschung durch Dritte gesichert hat und zweitens angeben, in welchem Umfang Aktivitäten auf dem Rechner des Kunden ausgeführt werden. Viele Anwendungen speichern auf dem Rechner des Kunden bestimmte Information über den Kunden, so dass sie ihn wiedererkennen und auf die Daten früherer Interaktionen zurückgreifen können (z.B. die Wunschliste oder die Liste der Interessen des Kunden, die er bei früheren Besuchen ausgebessert hat). Diese Daten werden als sogenannte Cookies auf dem Rechner des Kunden abgelegt Manche Anwendungen wollen Daten vom Kunden-Rechner erfassen, z.B. die Koordinate des Standortes bei einem mobilen Rechner, was viele Kunden lieber nicht zulassen. Der Kunde will sicher sein, dass die Anwendung nicht ein Tor öffnet, dass Dritte bei ihnen unbemerkt eindringen können.

Cookies (Kekse) sind Daten, die meist den Kunden und seine letzten Aktivitäten beschreiben und im Dateisystem des Kunden abgelegt sind. Sie werden meist gut versteckt, damit sie nicht leicht zu löschen sind.

Wird von einer Anwendung bekannt, dass sie schädlich ist oder dass sie erlaubt, Schadcode beim Kunden einzuschleusen, so werden Kunden nicht bereit sein, diese Anwendung zu nutzen. Das gilt für Webanwendungen, aber in noch viel höherem Masse für Anwendungen, die beim Kunden installiert werden müssen.

Teil II

Grundlagen für einfache Webapplikationen

3

Das Internet als Grundlage für das World Wide Web

Das World Wide Web oder kurz das *Web* meint das komplexe System von Dokumenten, die durch die verschiedenen Programme und die Infrastruktur für Kommunikation und Verarbeitung zugänglich ist. Die Infratstruktur für die Kommunikation wird durch das Internet bereitgestellt. In diesem Kapitel werden die Teile, die Internet bilden, meistens standardisierte Regeln, die zusammenwirken, vorgestellt.

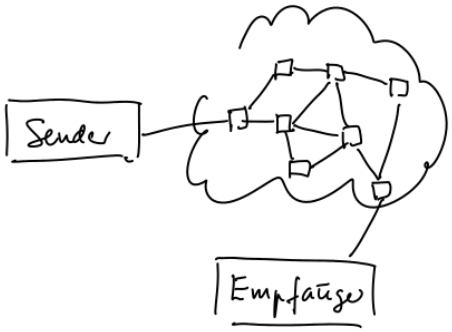


Abbildung 3.1: Store and forward im Internet

3.1 Standardisierung

Das Internet und das World Wide Web ist durch die Standardisierung geschaffen; es funktioniert nur, weil sich die meisten Programmierer an die festgelegten Normen halten, auch wenn diese nur faktisches Gewicht haben und meist nicht durch internationale Normierungsgremien festgelegt sind. Normierung ist überall dort notwendig, wo verschiedene, manchmal sogar beliebige, Lösungen möglich wären, aber das gesamte nur funktioniert, wenn sich alle an die gleichen Regeln halten, Straßenverkehr oder die Festlegung der elektrischen Spannung in Steckdosen sind gute Beispiele, bei denen beliebige Lösungen möglich sind aber nur wenn man sich auf eine einheitliche Regel einigt, das System, Straßenverkehr oder elektrische Stromverteilung, funktioniert. Gleiches gilt für Informationstechnologie, die nur funktioniert, wenn die Teile zusammenwirken. Kunst ist es, Teile so zu gestalten, dass sie zusammenwirken können .

Beide Straßenseiten sind gleich geeignet aber der Strassenverkehr funktioniert nur, wenn alle auf der gleichen Seite fahren!

Oft überleben in einem Standardisierungsprozess Lösungen, die längst ihren Sinn verloren haben, oder nur gewählt wurden, weil sie damals leicht implementierbar waren, heute aber nur noch stören. Sie bleiben aber erhalten, damit neue Geräte auch mit älteren Einrichtungen zusammenarbeiten.

In diesem Kapitel werden die technischen Grundlagen für das Internet, auf dem das Web aufbaut, erklärt und am Schluss Hinweise zum Suchen nach Fehlern gegeben.

Composition of exchangable parts

deSaussure hat bereits anfangs des 20. Jahrhunderts darauf hingewiesen, dass die Zusammensetzbarkeit einer endlichen Zahl von Wörtern zu unendlich vielen Sätzen den Reichtum und die Kraft von Sprache ausmacht.

Ferdinand de Saussure. *Cours de linguistique générale*. Payot & Rivages, 1995

backwards compatibility - das neue System funktioniert auch mit dem alten

3.2 Routing von Datenpaketen

Die Übertragung von Daten ist heute im Internet eine Übertragung von Paketen von Daten. Daten werden beim Sender in Pakete von meist fixer Länge (zB. 1 kB) zerlegt und beim Empfänger wieder zusammengesetzt. Die Pakete werden von den im Netz verbundenen Rechnern gespeichert und weitergereicht .

store & forward
Es ist nicht erforderlich, dass alle Pakete einer Nachricht den gleichen Weg durch das Netz nehmen!

Jeder Rechner entscheidet auf Grund der Adresse, zu der das Pakete geschickt werden soll, welchem seiner Nachbarn er das Paket weiterreicht, und da wird solange wiederholt, bis das Ziel erreicht ist. Rechner, die vor allem dem weiterreichen von Paketen dienen werden Router genannt; sie merken sich, für welche entferntere Rechner ihre Nachbarn erfolgreich waren, so dass das weiterreichen recht effektiv ist.

Wichtig ist aber, dass im Internet Adressierungs- und Routing-Methode so konstruiert sind, dass nicht eine einzelne Fehlstelle das ganze System zum halten bringen kann . Einzelne Ausfälle von Routern oder von Netzverbindungen verlangsamen kurzfristig die Kommunikation, führen aber nicht zu anhaltenden Ausfällen - solange der Zielrechner noch auf irgend einem Weg erreichbar ist.

Robustness principle des Internet

Bestimmte Länder konzentrieren den Verkehr von außen nach innen und umgekehrt über wenige zentral überwachte Router und kontrollieren, welche Inhalte ausgetauscht werden (3.2). Natürlich gibt es Wege, um solche Überwachung auszuhebeln...

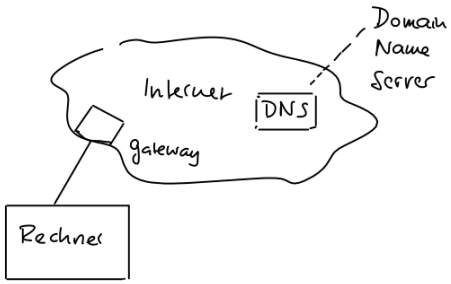


Abbildung 3.2: Gateway verbinden einzelne Rechner mit dem Internet

Netzwerk Topologie

Innerhalb des Internets müssen Pakete weitergeleitet werden; die Knoten der Verteilung sind im in einem Netz verbunden; die Struktur dieser Verbindungen wird als Netzwerk-Topologie bezeichnet.

Einfache Topologien sind Stern, wo jeder Endknoten mit dem zentralen Knoten verbunden ist; bei einer Baum-Topologie sind die Endknoten über Zwischenknoten mit einem Wurzelknoten verbunden. Stern- und Baum-Topologie enthalten viele Stellen, bei deren Ausfall die Verbindungen unterbrochen sind.

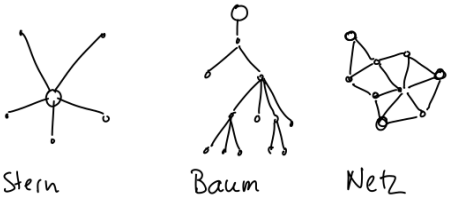


Abbildung 3.3: Netzwerk-Topologie: Stern, Baum und Netz

Redundanz

Heute sind im wesentlichen Netzwerke ueblich, weil diese gegen Störungen weniger anfällig sind, wobei häufig gegen den Rand eine Baumstruktur entsteht - jeder Randknoten ist mit einer Leitung an einen inneren Knoten gebunden, der nur mit einer Leitung angebunden ist. Zentrale Knoten sind untereinander mit mehreren Verbindungen vernetzt.

3.3 ISO Schichtenmodell

Eine Verbindung im Web kommt durch einen Stapel von Dienstleistungen zusammen, die je auf einer Ebene Kommunikation bestimmter Qualität erzeugen (3.5).

Das wichtigste Konzepte, das für den Erfolg des Web ausschlaggebend war, ist das Schichtmodell. In der Zeit der Entstehung des Internets hat die Internationale Standardisierungsorganisation (ISO) einen Vorschlag zur grundsätzlichen Organisation und Interaktion von Prozessen über Netzwerke gemacht, das Open Systems Interconnect (OSI) Modell[1].

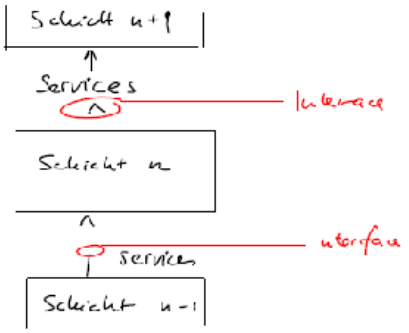


Abbildung 3.4: Prinzip des Schichtmodells

[1] Hubert Zimmermann. Osi reference model–the iso model of architecture for open systems interconnection. *Communications, IEEE Transactions on*, 28(4):425–432, 1980

3.3.1 Abstraktion durch Schichten

Ein komplexes System, wie z.B. das Internet, wird in Schichten aufgeteilt, die je bestimmte Aufgaben wahrnehmen (3.6). Diese Schichten bieten nach oben Funktionen an, die sie mittels Funktionen einer tieferen Schicht realisieren[2].

Eine Schicht abstrahiert jeweils einen Aspekt der Kommunikation und bietet dafür verschiedene Lösungen an, derart, dass in der Schicht darüber kein Unterschied (ausser allenfalls Qualitätsdifferenzen) festgestellt werden kann. Eine Schicht besorgt die Weiterleitung von Nachrichten von Quelle zum Ziel, eine andere verbessert die Qualität und korrigiert kurzfristige Ausfälle und Datenverluste, etc.

Jede Schicht ist durch ein Protokoll als übertragungsmechanismus mit dem Prozess der gleichen Schicht der Empfängerseite verbunden (3.5).

[2] D. W. Davies, E. Holler, E. D. Jensen, S. R. Kimbleton, B. W. Lampson, G. LeLann, K. J. Thurber, and R. W. Watson. *Distributed Systems - Architecture and Implementation*, volume 105 of *Lecture Notes in Computer Science*. Springer-Verlag, 1981

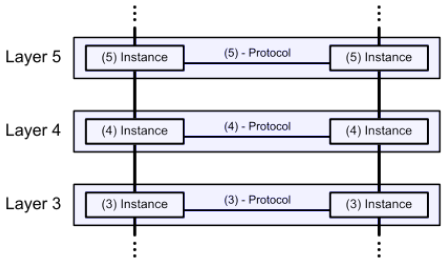


Abbildung 3.5: Schichten arbeiten zusammen und machen Dienste tieferer Schichten unsichtbar (transparent)

3.3.2 Anwendung

Das ursprüngliche ISO OSI Modell hat sieben Schichten vorgesehen, hat sich aber in dieser Form nicht durchgesetzt. Viele der im Anfang vorgeschlagenen technischen Lösungen sind überholt und weggefallen und einfache und weniger Fehler anfällige haben sich durchgesetzt. Im Folgenden wird eine vereinfachte Sicht auf die gängige Praxis — soweit für ein Projekt dieser Art nützlich — geboten. Die Funktionen, die eine Schicht anbietet, das heißt, das Interface zur nächsten Schicht, muss standardisiert werden. Dann können verschiedene Dienste gegeneinander ausgetauscht werden, sofern sie die gleichen Dienste mit dem gleichen Interface anbieten.

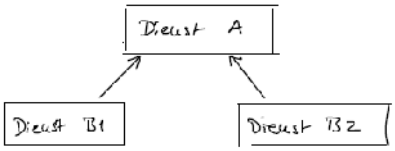


Abbildung 3.6: Ein Dienst kann mehrere verschiedene Angeboten von Dienste einer niedrigeren Schicht nutzen
Standardisierung um Auswechselbarkeit von Teilen wurde von Whitney für die Gewehre der amerikanischen Armee etwa 1800 erfunden.

3.3.3 Austauschbarkeit

Der Dienst A kann entweder durch Benutzung von Service B_1 oder B_2 erbracht werden; im Programm von A ist keine Änderung erforderlich, weil das Interface von D_1 und D_2 gleich ist (Figur 3.6).

Praktisch beobachten wir das, wenn wir ein Notebook einmal über ein Ethernet-Kabel, einmal über WiFi und einmal durch einen Mobiltelephon Dienst an das Internet anbinden: der Dienst Internet wird dabei durch sehr verschiedene darunterliegende Dienste ermöglicht, funktioniert aber an der Oberfläche gleich – allenfalls beobachten wir Unterschiede in der Qualität, z.B. die Geschwindigkeit ist meist bei Anbindung durch ein Ethernet-Kabel viel höher als über das Mobiltelephon.

Komplexität ist der Feind bei der Erstellung von Software; das OSI schichtenmodell ist eine möglichkeit, komplexität zu reduzieren und damit zuverlässigere Software zu erstellen.

Die Austauschbarkeit von verschiedenen technischen Lösungen für die gleichen Dienste ist offenkundig, wenn man sein Laptop manchmal per RJ45 mit dem Internet per Kabel verbindet

Abbildung 3.7: rj45

manchmal eine WLAN Verbindung erstellt und schließlich eine UTMS Stick verwendet, der über USB mit dem Laptop verbunden wird. Das gleiche Gerät (Laptop) wird je nach Situation über verschiedene physikalische Medien mit dem Internet verbunden und für diese Medien gelten jeweils verschiedene Standards (....).

Abbildung 3.8: utms stick

Qualitäts-Verbesserung

Eine höhere Schicht kontrolliert die Qualität der Übertragung. Drei Aspekte sind wichtig:

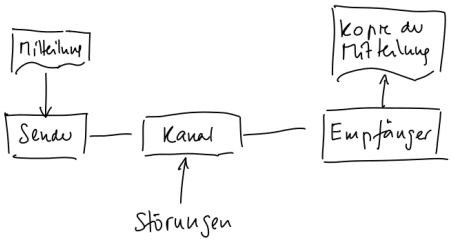


Abbildung 3.9: Shannon's model

1. Durch Einführen von Redundanz, z.b. durch Prüfbits, wird es möglich, die unvermeidlichen Fehler in der Übertragung von Sender zu Empfänger zu entdecken und bis zu einem gewissen Grad zu korrigieren (3.9).

2. Verlust von Daten; es wird geprüft, dass keine Pakete von Daten verloren gehen und gleichzeitig meist auch aufgepasst, dass der Sender nicht mehr sendet als der Empfänger empfangen kann, was zum Verlust der Daten fuehren wuerde. In einfachen Fällen quittiert der Empfänger den richtigen Empfang, bzw. verlangt eine Wiederholung (ACK oder NAK Signal des ASCII Codes); nach Eintreffen der Bestätigung sendet der Sender das nächste Paket

handshake

3. Die Pakete erreichen den Empfänger nicht unbedingt in der richtigen Reihenfolge und müssen richtig geordnet werden.

3.4 Schichten entsprechen dem Einpacken von Daten

Encapsulation principle

Die Daten werden im Internet als Pakete und nicht als fortlaufender Strom verschickt. In den unteren Schichten zwischen direkt verbundenen Rechnern, in den höheren Schichten zischen Sender und Empfänger durch das Netzwerk geleitet.

Von unten nach oben werden zusätzliche Dienste eingeführt, indem das Datenpaket der höheren Schicht eingepackt wird . Die untere

einpacken = english "wrap"

Schicht behandelt dann das eingepackte Paket entsprechend ihrem Protokoll und beim Empfänger wird das Datenpaket wieder ausgepackt und entsprechend dem Protokoll der höheren Schicht behandelt. Aus diesen Paketen wird dann auf der höheren Schicht allenfalls wieder ein kontinuierlicher Strom erstellt.

Reihenfolge der Wrapper von außen nach innen entspricht der Reihenfolge der Schichten von unten nach oben.

Abbildung 3.10: Eine Nachricht eines höheren Layers wird eingepackt

3.5 Internet Protocol Stack, vulgo TCP/IP

Das Internet ist in Schichten aufgebaut, die bestimmte Dienste erbringen. Heute werden weniger als die 7 Schichten des IOS OSI Modelles verwendet. Typisch besteht eine Internet Protokol Sammlung besteht aus den Ebenen:

1. Link layer. Transportiert Pakete von Daten zwischen den Netzwerkknoten.
2. Internet layer. Beförderung von Datenpaketen zwischen Sender und Empfänger Rechner über Zwischenstationen.
3. Transport layer. Zugang zum Internet und Austausch von Paketen zwischen Rechnern.
4. Application layer. Austausch von Nutzerdaten zwischen Prozessen (laufenden Programmen).

3.6 Link Layer

Der Link Layer beschreibt die physische Kommunikation und die Datenübertragung zwischen unmittelbar miteinander verbundenen Knoten des Internet.

Die unterste Schicht schafft die physische Verbindung, d.h. die Draht- oder drahtlose Verbindung zwischen zwei Komponenten, z.B. sind verschiedene Verfahren zwischen Rechnen vorgesehen, die jeweils elektrisch (Spannung, Signalform und Dauer) und als Stecker-Form normiert werden. Bekannt ist z.B. RJ45 ein acht-poliger Stecker zum Internet(**??**), der heute bei den meisten Rechnern vorhanden ist. Es wird auch die Art der (elektrischen) Signale, und deren Geschwindigkeit (**??**) festgelegt. Dabei werden Frequenzen etc. festgelegt.

Store and Forward Transport

Die Schicht, die Verbindungen zwischen Knotenrechnern im Internet bewerkstelligt und Pakete von Daten weiterleitet verwendet ein „store and forward“ Verfahren. Daten werden lokal beim Knoten gespeichert und sobald als möglich weitergeleitet. Dabei muss der Sender beachten, dass er nicht mehr Daten sendet als der nächste Knoten speichern kann.

3.6.1 Protokolle

Eine grosse Zahl von Standards regeln den Zugang zum Medium - wer darf wann senden - und wie der Austausch funktioniert. Klassisch ist das CSMA/CD Protokoll für den Zugang zum Ethernet, als frueher viele Stationen an einem Kabel hingen. . Vor dem Senden, prüft der Sender, ob das Medium frei ist und während dem Senden beobachtet er, ob seine Daten mit Daten von andern Sendern kollidieren. Es muss vermieden werden, dass zwei Sender gleichzeitig senden und die Daten vermischt und verfälscht werden, aber auch, dass alle Sender schweigen, weil sie andern den Vortritt lassen (6.8.1). Die Ausbreitung der Daten, die Geschwindigkeit des Sendens und die maximale Distanz zwischen Stationen muss so gewählt werden, dass bei CSMA/CD eine Kollision, d.h. wenn zwei gleichzeitig senden, sicher entdeckt wird.

CSMA/CD Carrier Sense Multiple Access/Collision Detection Protocoll

Das CSMA/CD Protokoll ist heute weniger wichtig, weil heute Router und Switches verwendet werden und zwischen den Knoten direkte Leitungen bestehen. Es kann zum Beispiel das Point-to-Point-Protocol auf einer Ethernete Verbindung verwendet werden. Zugang zum Medium Protokolle werden bei der Verteilung des Zugangs zum Internet von den Telephongesellschaften eingesetzt; bekannt ist etwa DSL, welche meist in einer assymetrischen Version betrieben werden - vom Provider zum Anscluss (downstream) ist die Bandbreite grösser als vom Kunden zum Provider.

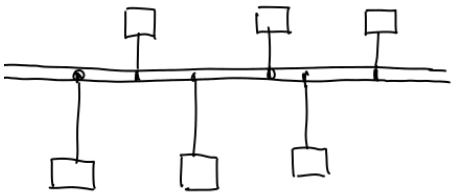


Abbildung 3.11: Die alte Ethernet Topologie mit Servern und Klienten an einem Kabel
Point-to-Point-Protocol = PPP

DSL Digital Subscriber Line (digitale Telefonabonennten-Anschluss)

3.7 Adressierung: Das Internet verbindet Computer

Der Empfänger eines Datenpaketes muss in einem einheitlichen System beschrieben werden, so dass die Zwischenstationen die Daten zum richtigen Ziel weiterleiten können. Man darf sich das ganz ähnlich wie die Zustellung von Briefen und Paketen durch die Post vorstellen: in einem Umschlag wird eine Datenmenge eingepackt und mit einer Adresse auf die Reise geschickt.Das System “Post” verwendet die Adresse nach einem bestimmten Verfahren um einen Brief vom Absender zum Empfänger zu befördern; die standardisierten Regeln sind vom Weltpostverein festgelegt.

3.7.1 Internet Protokoll (IP) Adressen

Die Datenpakete müssen von Absendenden Computer mit einer Adresse versehen werden, so dass sie richtig zugestellt werden können. Das Internet liefert das Paket dann beim Computer mit dieser Adresse aus.

3.7.2 IP heute: IPv4 Adressen

Im Internet hat jeder Rechner eine eindeutige Beziehung, Internet Protokoll (IP) Adresse genannt. Sie ist 32 bit lang und wird typischerweise mit 4 Zahlen zwischen 0 und 255 dargestellt: z. B. 198.168.1.0. Diese Internet-Adressen kennzeichnen die einzelnen Geräte, die miteinander verbunden sind. Jedes Gerät hat eine eindeutige, vom Hersteller zugewiesene, aber im Allgemeinem nicht benutzte IP Adresse (sogenannte MAC Adresse) und eine vom Netzwerk-Administrator zugewiesene IP Adresse.

IP Adressen sind hierarchisch gegliedert in weltweit festgelegte Länderblocks, daraus werden Gruppen an einzelne Organisationen zugeteilt aus diesen jedem Gerät eine zugewiesen wird. In Vorbereitung ist der Übergang auf längere IP Adressen im Internet Protocol IPv6, weil der Raum der IP Adressen fast vollständig belegt ist, d.h. alle 2^{32}IPv4 Adressen sind vergeben aber nicht unbedingt genutzt, [3] vollständig aufgebraucht ist und mit verschiedenen Tricks lokal erweitert wird; diese Tricks, als NAT (network address translation) bezeichnet, z.B. in WLAN Modems von Internet Providern, erfordern beim Anschluss eines Servers Aufmerksamkeit.

198.168.0.0 ist ein Adressbereich, der frei verwendet werden kann, z. B. für ein WLAN basierendes lokales Netzwerk.
Die TU vergibt der Geoinformation IP Adressen, wie 128.130.78.173 aus ihrem block der mit 128.130.0.0/15 beschrieben ist (d.h. die TU kann 2^{15}IP Adressen, beginnend mit 128.130.0.0 vergeben).

[3] angeblich sind nur etwa 14% wirklich in Verwendung

3.7.3 Zuteilung von IP Adressen

Jedem Rechner, der an das Internet angeschlossen werden soll, muss eine IP Adresse zugeteilt werden. Das kann statisch oder dynamisch erfolgen:

Statische Zuweisung: Dem Rechner wird eine feste IP Adresse vom Netzwerkadministrator aus seinem Pool zugeteilt und im Rechner eingetragen (z.B. in der Datei /etc/network/interfaces).

Dynamisch Zuweisung mit zentralem Dienst: IP Adressen werden in vielen Netzen dynamisch, d.h. von Fall zu Fall zugewiesen. Dafür dient ein Dienst namens DHCP (dynamic host configuration Programm) der z.B. vom Netzprovider oder vom WLAN Router angeboten wird.

DHCP (dynamic host configuration Programm)

Ein neu in ein Netz eintretender Rechner verlangt dort eine IP Adresse, die für eine bestimmte Zeit gültig ist. Zu verschiedenen Zeitpunkten kann der gleiche Rechner also verschiedene Adressen haben.

Dynamisch Zuweisung ohne zentralen Dienst: das Verfahren “Zero configuration networking” braucht kein zentraler Dienst , sondern Rechner suchen sich vorsichtig eine freie IP Adresse.[4]

zeroconf

[4] Die Provider müssen, auf Anfrage der Gerichte Auskunft geben können, wann welcher Nutzer welche IP Adresse zugeteilt worden ist; sie muessen also speichern, welchem verantwortlichen Nutzer, welcher den Internet Dienst Vertrag hält, welche IP Adresse zur vorübergehenden Nutzung zugeteilt wurde.

3.7.4 Die neuen IPv6 Adressen

Mit IPv6 stehen Adressen von 128 bit Länge zur Verfügung, also für jeden Einwohner der erde etwa eine Milliarde, was wohl ausreichen sollte, um auch Autos und Kühlschränken wie geplant eine IP zuzuteilen.

Eine IPv6 Adresse wird mit Hexadezimalzahlen (0-9, a-f) geschrieben, z.b. 2001:0db8:0000:0000:0000:ff00:0042:8329. Da aber soweit als möglich vermieden wird, dass Menschen solche Zahlen eingeben, spielt die Länge keine Rolle. Für die TU Wien sind 2^{98} Adressen vorgesehen; Institute können ipv6 Adressierung brauchen, aber bisher sind in Österreicher erst wenig Provider bereit, ipv6 zu unterstützen (z.B. hat A1 noch keine veröffentlichten Pläne dafür!).

3.8 Transport Layer

Der Transport Layer erlaubt die Kommunikation zwischen Rechnern, die indirekt durch das Internet verbunden sind. Für die Übertragung von Daten im Internet werden vor allem zwei Protokolle verwendet: TCP und UDP.

3.8.1 Transport Control Protokol TCP

Beim TCP Protokoll wird eine Verbindung zwischen Sender und Empfänger aufgebaut und der Fluss der Pakete kontrolliert. Aus der darunterliegenden Transportschicht wird, die nicht zuverlässig ist wird eine zuverlässiges und fehlerkorrigiertes Medium, bei dem die Daten in der richtigen Reihenfolge und vollständig geliefert werden. Im "Umschlag" von TCP steckt die Adresse des Senders und Empfängers, Reihenfolge Information.

TCP protokol ist verbindungsorientiert
best effort

Das Protokoll ist aufwendig, weil eine Verbindung auf- und abgebaut werden muss und der Aufwand um die Qualität der Übertragung zu garantieren produziert Verzollungen in der Übertragung

latency

TCP stellt sicher, dass größere Nachrichten vollständig übertragen werden. Der Strom der Daten wird in Pakete aufgeteilt . Jedes Datagram enthält die Port Nummern von Sender und Empfänger und eine Sequenznummer. Das Protokoll sorgt für Zusammensetzung der Datagramme in der richtigen Reihenfolge zur Rekonstruktion des Datenstromes. TCP braucht IP um die einzelnen Pakete zu übertragen und die an der Übertragung beteiligten Router sind über die Verbindung nicht informiert - nur der Sender und Empfänger kümmern sich um die Verbindung.

Datagram = Paket von Daten

3.8.2 User Datagram Protocol UDP

Das UDP Protokoll ist einfacher und produziert weniger Aufwand; es ist "connectionless", dh. ohne ein Verbindungskonzept. Es werden Mitteilungen (Datagrams) von einem Sender an einen Empfänger geschickt, ohne das zuvor eine Verbindung aufgebaut worden wäre. Die Übermittlung kontrolliert nur die richtige Übermittlung der eingeschlossenen Daten, ohne aber Verluste von Daten, Verzögerungen und falsche Reihenfolgen etc. zu korrigieren.

UDP wird immer dann angewandt, wenn nicht die Qualität, sondern die rasche Ausführung und geringe Verzögerung einer Übermittlung im Vordergrund stehen. Also z.b. für die Übermittlung von Sprache, kodiert und paketiert, durch das Internet geschickt. Geringe Verluste von Paketen werden als kleine Störungen wahrgenommen und stören die Verständlichkeit nicht, hingegen würden Verzögerungen, wie sie TCP bringen könnte, deutlich Probleme bei der Interaktion zwischen den Personen erzeugen. VoIP ist ein Beispiel, bei dem eine höhere Schicht ein verbindungsorientiertes Protokoll auf UDP aufsetzt; weil Verbindung auf höherer Ebene erledigt ist es überflüssig auf einer tieferen Schicht das nochmals zu besorgen.

VoIP Voice over IP = Telefongespräche durch das Internet

3.9 Menschen-lesbare Web Adressen (URL und IRI)

Im allgemeinen tippen wir keine IP Adressen ein, sondern Ausdrücke, z.B. einen Namen, der auch Menschen verständlich ist. Die TU Wien Homepage ist unter www.tuwien.ac.at zu finden. Mein Rechner hat z.B. den Namen gi33.geoinfo.tuwien.ac.at.

Diese Web Adressen, als Uniform Resource Locator bezeichnet, die alle Ressourcen irgendwo im Web eindeutig bezeichnen. Ein Teil davon sind host addresses, d.h. Adressen, die ein Gerät bezeichnen, auf dem Prozesse laufen können.

URI - Uniform Resource Locator

Umwandlung von URL/IRI in IP

Der Teil der URL der den Host (den Rechner) bezeichnet, wird von einem Domain Name Server in eine IP Adresse umgewandelt. Für jeden Aufruf einer Web Ressource, die mit einem URL gekennzeichnet ist, muss beim DNS in eine numerische IP Adresse umgewandelt werden. Die numerische IP Adresse des DNS muss also dem Client bekannt sein. Die vom DNS zurückgemeldete IP Adresse wird dann zum versenden der Datenpakete verwendet.

DNS (Domain Name Server)

Unterschied URL und IRI

Die Zeichen in einer URL sind beschränkt auf das (amerikanische) ASCII Alphabet ; das sind 127 Zeichen, 0-9, a-z, A-Z und wenig Sonderzeichen. Die neuere, internationalisierte Version IRI verwendet Unicodes, was Umlaute, Sonderzeichen, japanische Kanji, chinesische Schriftzeichen etc. etc. erlaubt. [5]

ASCII American Standard Code of Information Interchange

IRI International Resource Locator

URL und IRI sind von ihrer Konstruktion her, weltweit eindeutig: der weltweite, hierarchische Aufbau der IP Adressen und die zugehörigen URL/IRL sorgen für Eindeutigkeit des letzten Teils (der Länderkennung oder anderer Top Level Domains weltweit. TLD sind z.b. .at, .de, .ch, die für URL für hosts in Österreich, Deutschland oder der Schweiz bezeichnen; allgemeine TLD sind .com, .net oder .org, unter denen sich jedermann registrieren kann. Eine Liste der TLD findet sich unter http://en.wikipedia.org/wiki/List_of_Internet_top-level_domains.

[5] Unglücklicherweise wird diese Freiheit öfters zum Verschleiern einer Adresse verwendet. So ist für einen Menschen www.ebay.com nicht von www.ebay.com zu unterscheiden - für einen DNS sind die zwei verschieden (im zweiten ist das “a” ein kyrillischer Buchstabe!). wenn ein Nutzer meint, mit einem Ebay Rechner zu kommunizieren, in Wirklichkeit aber mit einem Rechner, der nur vortäuscht von Ebay zu sein, kann der Nutzer leicht betrogen werden. das ist als “IDN homograph attack” bekannt und wird von kriminellen Gruppen ausgenützt.

Top Level Domain TLD, zb. Länderkennung

Die Eindeutigkeit innerhalb einer Organisation, der vom TLD verwalter eine subdomain zugewiesen ist (z.B. “ac” und danach “tuwien”) wird wiederum durch die technische Konstruktion der web Server garantiert. URL/IRL werden deshalb häufig als eindeutige Namen verwendet, auch wenn nur Beziehungen zwischen Namen hergestellt werden sollen (in RDF). .

Alternativ weltweit eindeutig sind auch Email Adressen, die oft in Anwendungen verwendet werden, um Personen zu identifizieren (Beachte: eine Email Adresse identifiziert eine Mailbox auf einem host, nicht eine Person!)

URL escaping

Eine URL enthält auch den Namen des Files, der angefordert wird, und oft zusätzliche Argumente, die vom Web-Server verarbeitet werden. In einer URL dürfen keine Leerzeichen vorkommen und alle Zeichen müssen als ASCII kodiert sein - d.h. Zeichen ausserhalb des ASCII Zeichensatzes müssen durch spezielle, numerische Kodierung dargestellt werden. Die Umwandlung einer Zeichenkette mit beliebigen Zeichen in eine URL wird als “URL escaping” bezeichnet.

Leerstellen werden durch “+” ersetzen und Sonderzeichen numerisch zwischen % ersetzt. Programmierte Routinen um eine Benutzereingabe umzuwandeln existieren (siehe Google Geographic Code Services).

Als Beispiel die folgende Abfrage an Google Maps, bei der nach der Web-Adresse nach einem “?” verschiedene Codes für den benutzten Webbrowser (iceweasel) und dem Zeichensatz (UTF-8) auch ein Titel “Hotel San Jose” mit Leerzeichen folgt:

```
https://maps.google.com/maps?client=iceweasel-a&ie=UTF-8&q=Hotel+San+Jose&fb=1&hq=hotel+san
```

3.10 Application Layer: Kommunikation zwischen Prozessen: Ports

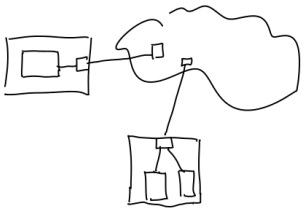


Abbildung 3.12: Verbindung zwischen Ports

Datenaustausch soll nicht zwischen den Rechnern stattfinden, sondern zwischen den Prozessen, die innerhalb der Rechner ablaufen. Die Prozesse melden sich beim host operating system an und teilen mit, auf welchen ports sie 'horchen', dh. Pakete, die mit dieser port Nummer beim OS ankommen werden an diesen Prozess weitergeleitet.

Damit eine Verbindung zwischen zwei laufenden Programmen (Prozesse genannt) auf zwei Rechnern (z. B. Server und Client) zustande kommt, werden an die Web-Adresse Port Nummern angehängt. Ein Programm (Service) lauscht typischerweise auf einem bestimmten Port, ob dort Aufträge abgegeben werden — vergleichbar mit einem Schalter bei dem Bestellungen entgegengenommen werden. In der Wikipedia gibt es eine (lange) Liste der üblicherweise welchen Diensten zugeteilte Port-Nummer [6]. Typisch sind z.b. port 22 für ssh Verbindungen, port 80 für web Browser etc.

[6] (Wiki „Port“)

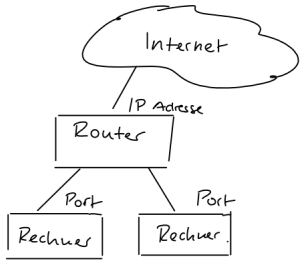


Abbildung 3.13: Ports adressieren Prozesse

Netzzugang durch Dienstleister

Dienstleister bieten gegen Gebühren zugang zum Internet. Kunden wird meist eine IP Adresse zur Verfügung gestellt und ein Verbindungspunkt, d.h. ein Gerät, an das mit einem Kabel oder drahtlos angeshlossen werden kann.

Müssen in einer Firma oder einem Haushalt mehr als ein Gerät angeschlossen werden, so erhält jedes heute eine eigene IP im lokalen Netz. Der Router vermittelt dann den Verkehr mittels NAT, so, dass die verschiedenen Geräte unabhängig im Internet agieren können [7]

Sollen Geräte von außen erreichbar sein, so muss der lokale Router eingehenden Verkehr an diese weiterleiten (im allgemeinen wird eingehender Verkehr nicht durchgelassen - siehe Firewall ..), das heißt port forwarding.

Sollen mehrere verschiedene Geräte den gleichen Dienst hinter einem Router anbieten, so müssen diese unterschiedliche Ports für den gleichen dienst verwenden.

Telephongesellschaften, Kabelfernseh- und Stromversorger vermieten Privaten und Geschäften Leitungen und bieten auch den Zugang zum Internet an.

[7] IPv6 (3.7.4) stellt genügend Adressen zur Verfugung so dass NAT in Zukunft nicht mehr nötig sein wird.

Dynamic DNS service

Provider

Firmen, die Zugang zum Internet anbieten teilen ihren Kunden IP Adressen zu und ändern diese regelmäßig (häufig alle 24 Stunden); dafür wird der Dienst kurzzeitig unterbrochen, was nicht weiters merkbar ist. Soll aber ein Dienst von außen erreichbar sein, so muss für die URL des Dienstes (z.B. ein SSH Zugang zu einem Rechner) nach jedem Wechsel der IP Adresse die Übersetzung der URL bei den DNS

geändert werden. Dazu dienen dynamische Domain Name Services. Bei diesen wird für eine gewählte URL (meist eine subdomain einer URL, welcher der DynDNS Organisation gehört) eine IP hinterlegt, damit die Umsetzung der URL in die IP gelingt. Wenn die IP wechselt, so muss die neue IP hinterlegt werden. Das geschieht sehr einfach durch eine regelmäßige Anfrage des zu verbindenden Gerätes beim DynDNS, wobei der DynDNS Dienst feststellt, ob die Anfrage von der bereits registrierten IP ausgeht oder ob eine neue zugeteilt worden ist und diese registriert werden muss.

Übertragung von Daten zwischen Rechnern - FTP

FTP ist ein Verfahren zum Transfer von Dateien (files) zwischen Rechners - es wurde 1971 entworfen und wird immer noch benutzt. Es gibt verschiedene Graphische Benutzeroberflächen (z.b Filezilla [8]), die sowohl das Filesystem meines Rechners als des entfernten (remote) rechners zeigen; es ist dann einfach, Dateien von einem zum andern Rechner zu schieben. FTPS ist die "sichere", d.h. verschlüsselte Version des gleichen Protokolls.

File Transfer Protocol

[8] https://filezilla-project.org/

3.11 Sicherheit des Zugangs vom Web - Firewalls

Da das Internet weltweit offen ist und sich auch übelwollende Teilnehmer herumtreiben, sind Sicherheitsmaßnahmen nötig. Diese sind nicht Teil der Protokolle, denn die Protokolle sind mit Absicht neutral zum Inhalt — legales oder kriminelles wird gleich übertragen, denn wer sollte international entscheiden, was legal und was nicht legal sei. Deshalb muss sich jede Organisation selber schützen.

Zum Beispiel werden beim Übergang vom weltweiten Web in das interne Netz einer Organisation Filter eingesetzt, sogenannte Firewalls, die z. B. Datenpakete für verschiedene Ports blockieren (3.11). Gesperrt werden heute eigentlich alle ausser die wenigen, allgemein genutzten Ports und Ports, die von der Organisation absichtlich genutzt werden.

für die allgemein genutzen Ports (web browser, email) bemüht man sich, Programme sichere zu machen. für andere Dienste muss oft bei einer zentralen Stelle der Organization (bei der TU ist das beim Zentralen Informatik Dienst) um öffnung des Ports nach aussen gebeten werden. Alle andern Ports sind gesperrt.

Portsperren können sowohl auf Seiten der Organisationen, bei der der Server oder der Client angeschlossen ist, vorgenommen werden (Figure 3.11) und müssen, wenn man eine neue Anwendung einrichtet, gezielt geöffnet werden.

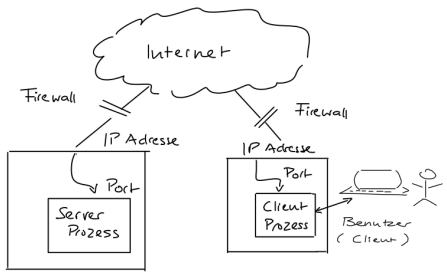


Abbildung 3.14: Firewalls blockieren unerwünschten Verkehr auf dem Web

Kryptographie

Das wichtigste Verfahren zur Erreichen von Sicherheit im Web ist die Verschlüsselung. Im Internet verkehren alle Datenpakete im gleichen Medium und können (mehr oder weniger problemlos) bei allen Routern gelesen werden. Dagegen hilft die Verschlüsselung des Inhaltes, so dass der Inhalt nur vom Adressaten, nicht aber von einem Dritten gelesen werden kann. Die Verschlüsselung hilft aber auch, um sicherzustellen, dass ein Text von einer bestimmten Quelle (z.B. eine Person, Behörde oder Firma) stammt.

Public Key

Grundlage dieser Verschlüsselungs-Methoden sind Funktionen, die nur in einer Richtung leicht zu berechnen sind; typisch, die Faktorisierung einer großen Zahl. Es ist einfach, zwei große Primzahlen zu multiplizieren, es ist hingegen sehr aufwendig, das Resultat in die zwei Primzahlen zu zerlegen, wenn diese nicht bekannt sind.

Das Programm SSH enthält eine Routine zum generieren von zwei Schlüsseln, der eine wird als *public key*, der andere als *private key* bezeichnet. Diese Schlüssel arbeiten als Paar, so dass der eine zum Verschlüsseln, der andere zum Entschlüsseln verwendet werden kann.

Bei der Verschlüsselung von Nachrichten kann das Ziel sein:

Sicherung Übertragung: Wenn Alice eine Nachricht mit dem public key von Bob verschlüsselt, kann nur Bob mit seinem private key die Nachricht entschlüsseln, niemand anderes. Den public key des Partners ist allgemein bekannt, aber nicht nützlich um die gesicherte Nachricht zu entschlüsseln.

Sichern des Ursprungs: wenn Alice dagegen eine Nachricht mit ihrem private key verschlüsselt, kann sie jedermann mit ihrem public key entschlüsseln und ist dann sicher, dass die Nachricht von Alice ist - niemand anderes kann eine Nachricht so verschlüsseln, dass sie mit ihrem public key zu öffnen ist.

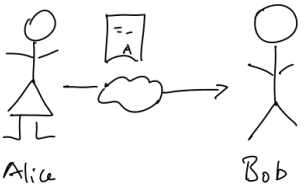


Abbildung 3.15: Alice sendet Bob eine geheime Nachricht

Sicherheit der Übertragung

Die Verschlüsselung des Datenstromes zwischen zwei Ports wird üblicherweise durch das TLS Protokoll ; TLS kann zu einem bestehenden Anwendungsprotokoll dazugeschaltet werden: z.B. HTTP mit TLS ist HTTPS. Email abfragen werden ebenfalls über mit TLS verschlüsselte Verbindungen gemacht.

Transport Layer Security

Sicherung von Verbindungen

Es ist oft notwendig, einen Rechner über das Netz fernzusteuern; das ist potentiell gefährlich, insbesondere weil beim Login Benutzername und Passwort ungeschützt übertragen werden und also von andern

mitgehört werden könnten. Sicherer ist die gegenseitige Identifizierung durch public key cryptography und durchgängig Verschlüsselung der übertragenen Inhalte. Das gesicherte SSH Protokoll für den Zugang zu einem andern Rechner erreichen dies.

https: ist das kryptographisch gesicherte http Protokoll

Bei einer ersten Verbindung wird der public key übertragen und bei jeder späteren Verbindung wird damit geprüft, ob die Aufnahme der Verbindung durch den autorisierten Nutzer erfolgt. Dach wird mit Hilfe dieser asymmetrischen Schlüssel eine symmetrischer Schlüssel für diese Verbindung erzeugt und verwendet. Danach werden alle Daten über diese Verbindung verschlüsselt.

Localhost

Für die Programmierung und die Entwicklungsarbeit insgesamt kann es ausreichen, wenn auf einem Rechner gleichzeitig der Server und der Browser läuft. Man vermeidet damit viele Probleme, die von Routern, Firewalls, Sicherheitseinstellungen etc. im Netz verursacht werden können, bzw. man kann die einen Probleme von den andern trennen und sich bei der Fehlersuche auf eine kleiner Zahl von Komponenten konzentrieren.

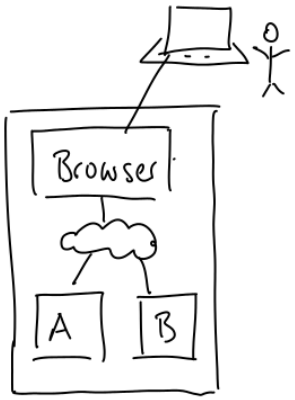


Abbildung 3.16: Localhost: Server und Klient sind alle im gleichen Rechner, aber dennoch über Internetprotokolle verbunden.

Es entsteht dabei keine richtige Verbindung durch das Internet, weil die gesendeten Pakete direkt an den Empfänger übergeben werden, was einen kleinen overhead ausmacht.

Localhost verbindungen werden aus zwei Gründen verwendet:

Verminderung von Fehlerquellen: Diese Architektur wird sehr oft bei der Entwicklung gewählt, weil damit viele Fehlerquellen ausgeschaltet sind und sowohl Client als Server direkt am gleichen Rechner kontrolliert werden können.

Flexible Architektur: Wir statt einesmonolithischen Programmes eine aufgabe auf einen Klienten und einen Serverprozess aufgeteilt, so können diese zuerst auf dem gleichen Rechner installiert werden. Später, wenn z.B. viele Anfragen den einen Rechner in die Kniezwingen würden, werden ddie einzelnen Aufgaben auf mehrere Rechner verteilt.

Für jeden Rechner gilt, dass der lokale Server mit einer fiktiven IP von “localhost” angesprochen werden kann (steht für 127.0.0.1 IPv4 oder für 0:0:0:0:0:0:0:1 in IPv6).

Qualität einer Verbindung

Drei Parameter beschreiben die Qualität einer Verbindung im Netz:

1. Bandbreite - wie groß ist der maximale Datendurchsatz pro Zeiteinheit, wird in Mbit/sec (Mbps) gemessen. Für die Beurteilung,

wie lange die Übertragung eines Datensatzes von x MByte dauern wird, sind die Kosten der Netzprotokolle und der schichtenweise Übersetzung einzubehalten (>10%).

2. Verlustrate - wie sicher ist die Übertragung, wie viele Fehler werden eingetragen und wie oft gehen Daten verloren.

3. Verzögerung - wie lange dauert es bis ein minimales Datenpaket vom Client zum Server geschickt ist und die Antwort des Servers wieder beim Client.

Verschiedene Anwendungen verlangen unterschiedliche Qualitäten und reagieren unterschiedlich, wenn Verbindungen "schlecht" sind. Extremfälle sind Skype (und andere Telefonie-über-das-Internet Dienste): Verzögerungen machen sich sehr unangenehm bemerkbar weil sie den natürlichen Wechsel zwischen Sprechen und Zuhören stören, Fehler sind tolerierbar - es rauscht in der Leitung - aber eine bestimmte minimale Bandbreite ist erforderlich, damit eine realistische Stimmdarstellung möglich ist und die Verständigung klappt. Spiele sind ähnlich anspruchsvoll in Bezug auf die Verzögerung. Anderseits spielt beim Herunterladen von Filmen oder ähnlichem die Verzögerung keine Rolle, entscheidend ist die Bandbreite. Für die Übermittlung komplexer Dateien, z.B.. .doc files, dürfen keine Fehler auftreten, bzw. auftretende Fehler müssen korrigiert werden - was für die Übermittlung von Telefonie keine Rolle spielt und bei Filmen toleriert wird.

VIP Voice over Internet Protocol

Es gibt eine politische Diskussion ob die Internet Provider die Vermittlung von Daten nach der Qualität beeinflussen dürfen. Ist es zulässig, bestimmte Inhalte oder Daten aus bestimmten Quellen bevorzugt zu behandeln? Die Frage ist strittig, weil die Provider nicht nur im Geschäft mit der Vermittlung von Daten zwischen Rechnern agieren, sondern selber auch Inhalte anbieten und diese allenfalls bevorzugt behandeln. Im Geschäft mit der Vermittlung von Daten sind die Provider häufig in einer Monopolsituation, die sie dann ausnützen können, um auch Monopolsituationen im Verkauf von Inhalt zu erreichen. Wenn Filme, die ich bei meinem Provider einkaufe rasch und fehlerfrei, aber Filme, die ich von einem andern Anbieter kaufe langsam und fehleranfällig übermittelt werden, gibt es einen starken Druck, dass der Provider auch beim Verkauf von Filmen ein Monopol erhält.

QOS quality of service

Fehlersuche bei Internet Verbindungen

Für die Suche nach Fehlern ist systematisch vorzugehen. Man kann zum Beispiel von unten aufsteigend jede Protokollschicht testen, oder rasch größere Teile testen, damit der Fehler eingegrenzt werden kann.

Die Behebung eines Fehlers in einem computerisierten Systems ist häufig einfach, wenn einmal der Fehler genau lokalisiert ist, was in einem komplexen, dynamischen System oft der schwierige Teil der Fehlerbehebung ist. Viele Komponenten im Web sind fehlertolerant und funktionieren auch wenn andere Komponenten falsch eingestellt sind — Fehler sind dann oft vom korrigierenden System maskiert und doppelt schwierig zu finden. Ohne ein systematisches Vorgehen geht es nicht!

Divide et imperare - teile und herrsche!

Ist der Rechner im Netz angemeldet und verbunden?

Die Yeichnung 3.11 zeigt, wo Fehler auftreten können. Ich teste meist zuerst mit dem Webbrowser, ob der Anschluss ans Web funktioniert, z. B. mit einer Suche mittels Google [9]. Wenn das klappt, ist der Rechner wohl richtig im Web eingebunden. Wenn der webbrowser einen Fehler anzeigt ("can not reach address xxx") muss man die Fehlermeldung genau lesen und interpretieren und in tieferen Schichten des Protokolls suchen. Das braucht meist Kommandos, die auf der Kommandozeile oder einem Terminal-Fenster eingegeben werden:

[9] weil google.com ein einfach zu merkender Dienst ist, der kaum einmal ausfällt

Mit dem Hilfsprogramm *ping* kann ein benachbarter Rechner angestoßen werden, der eine einfache Antwort zurückschickt. (Achtung: ping ist manchmal aus „Sicherheitsüberlegungen", z.B. bei der TU, unterdrückt). Dazu verwende ich zuerst eine numerische IP Adresse für ping. Wenn das funktioniert, kann *ping* mit einer symbolischen Web Adresse zeigen, ob der Namensdienst funktioniert (alternativ kann man host brauchen).

- Google search funktioniert - Rechner hat IP Adresse, DNS funktioniert
- Ping mit numerischer Adresse (z.b ping 8.8.8.8) - Rechner hat IP Adresse
- ping mit URL (z.B.. ping google.com) - DNS funktioniert

Laufen die notwendigen Prozesse?

Danach ist zu prüfen, ob der anzusprechende Prozess auf dem angesprochenen Rechner tatsächlich läuft. Das ist in einer Liste der laufenden Prozesse des Rechners sichtbar. Der Prozess muss mit dem Port verbunden sein. Das kann geprüft werden, wenn man den Dienst von diesem Rechner aus mit der Web Adresse localhost anspricht.

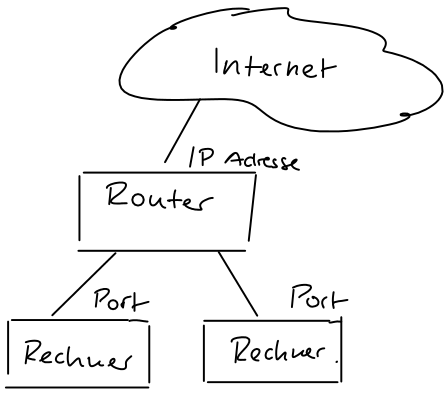


Abbildung 3.17: Verbindung mehrerer Rechner mit dem Internet über einen Router — Adressierung über unterschiedliche Ports

Sind die Ports offen?

Zuletzt untersuche ich, ob bei einer Verbindung zwischen Programmen die Ports offen sind oder ob dabei ein Feher aufgetreten ist. Das prüft man mit einem Hilfsprogramm, das erlaubt zu sehen, welche Ports offen sind (bei Linux ist das Network Tool „gnome-nettool"). Leider kann dieser Test negative ausgehen, weil das Abfragen ob ein Port offen ist, von einem sicherheits-(über-)bewussten Administrator abgestellt wurde. Es ist möglich, dass der Rechner über einen Router indirekt mit dem Web verbunden ist und dieser die Verbindung zum

Rechner für diesen Port nicht richtig weiterleitet (Figure 3.11).

4
Das Web und die dafür nötige Basis Software

Diese Kapitel gibt einen Überblick über die für unser Projekt notwendig Software und die dabei verwendeten Sprachen; Details werden später im Teil III besprochen. Die Basis-Software wird installiert und Tests, um die Funktionen zu prüfen gezeigt, wobei die Installation der Datenbank und PHP auf später verschoben werden kann, da sie im folgenden Kapitel noch nicht verwendet werden. Diese Kapitel schafft die Voraussetzung für die im nächsten Kapitel beschriebene Realisierung des Mock-up der Anwendung, der aus Webseiten mit festem Text aber ohne Daten und Karten besteht.

Das Internet bewerkstelligt Verbindungen zwischen Rechnern und den darin laufenden Programmen (genauer Prozesse). Diese werden mit IP Adressen und Portnummern identifiziert und zwischen ihnen werden Daten ohne Interpretation ausgetauscht. Auf dieser Grundlage können nun "intelligentere" Austausche erfolgen:

Das Web tauscht Dokumente aus und adressiert Dokumente (4.1). Die erste und wichtigste Anwendung des Web ist der Browser, der Dokumente von einem Server anfordern kann und diese dann anzeigt; das geht über das ursprüngliche Übertragen von Dateien hinaus, indem Inhalte angezeigt werden und in diesen Inhalten wiederum auf andere Dokumente verwiesen werden kann, die dann vom Nutzer leicht aufgerufen und angeschaut werden können. Das ist eine spezielle Form der Client-Server-Architektur (Abschnitt §2.1.18 auf Seite 37), bei der der Client Dokumente anzeigt und der Server Dokumente liefert.

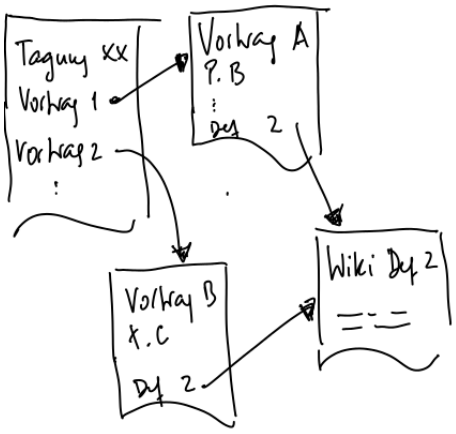


Abbildung 4.1: Programm einer Tagung, Vorträge und verwendete Definitionen sind verlinkt aber möglicherweise auf verschiedenen Rechnern gespeichert

Web - ein Netz von Dokumenten

Das Web beruht auf zwei grundsätzlichen Methoden:

- das HTTP Protokoll für den Zugang und die Übertragung von Daten und
- die HTML Sprache zur Beschreibung der Darstellung des Inhaltes.

HTTP - Datenaustausch
HTML - Darstellung des Inhaltes

Eine Web Applikation besteht aus einem Teil der in einem Client lauft - ein web Browser - und einem Teil der im Server lauft - ein web Server (2.25 auf Seite 38).

4.1 Realisierung der Client - Server Architektur

Ein Klient für einen Web Dienst hat zumindest ein “Web Browser” laufen - das kann Internet Explorer oder Firefox auf einem stationären Windows Rechner sein, das kann aber Opera oder Chrome auf einem Android Mobiltelefon sein. Auf der Seite des Servers sind mehrere Pakete notwendig (4.2).

Als Kürzel ist „LAMP-Server“ für diese Gruppe von freier Software üblich; LAMP steht für Linux, Apache, MySQL und PHP. Diese Programme funktionieren auf anderen Betriebssystemen (Windows, Mac OS). Selbstverständlich funktionieren andere ähnliche Programme und bieten dieselben standardisierten Dienste an.

Diese Programme werden gestartet und die laufenden Prozesse werden als Server bezeichnet (z.B. Apache Server, MySQL Server).

Im Folgenden wird die LAMP-Umgebung beschrieben; die allgemeine Überlegungen gelten für andere ähnliche Systeme. Anschließend werden alternative Lösungen dargestellt, die einfachere Lösungen versprechen.

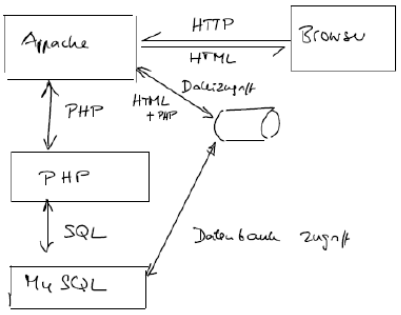


Abbildung 4.2: Software bei Server und Client
Ein moderner Server wird typischerweise mit der folgenden freier Software ausgerüstet und betrieben:

- Linux Betriebssystem
- Apache http Server
- MySQL Datenbank
- PHP Sprache.

Server = laufendes Programm, das auf Anforderungen reagiert

Verbindung Rechner - Internet

Auf jedem Rechner, der ans Internet angeschlossen wird, muss ein Prozess aktiv sein, der die Verbindung zum Netz hält. Dieser muss zumindest die IP Adresse von sich selbst kennen, die des Rechners über den er im Netz eingebunden ist, Gateway genannt, und diejenige des Nameservers (Abbildung 4.3). Dies wird meist (aber nicht an der TU) mit DHCP (3.7.3) automatisch eingerichtet.

Der Prozess, der das Internet bedient, muss die oben besprochenen Protokolle implementieren.

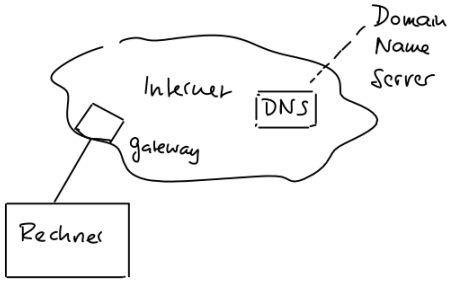


Abbildung 4.3: Jeder Rechner am Netz kennt zumindest die IP Adresse eines Gateways und eines Domain Name Servers

4.2 Browser

Auf Seiten des Klienten ist ein beliebiger Browser als Gegenstück zum Programm auf der Seite des Servers möglich; die beiden Programme müssen nur zusammenpassen, d.h. das gleiche Protokoll verwenden. HTTP (auf Seite 95) ist standardisiert und bildet die funktionierende Grundlage für Webdienste.

Der Browser läuft im Betriebssystem und muss für dieses angepasst sein. Firefox ist für viele Systeme erhältlich (z.B. Windows, Linux, Android für Tabletts und Mobiltelefone); es gibt versuche, den Browser als das Betriebssystem zu verwenden (z.B. das Cloud Operating System [1] oder Google Chrome OS).

Eine Anwendung, die von jedermann und überall benützt werden kann, darf nicht viel voraussetzen und möglichst keine Installation

[1] http://en.wikipedia.org/wiki/Cloud_%28operating_s

von Programmen erfordern. Heute ist auf praktisch jedem Rechner ein moderner Webbrowser installiert. Ein Webbrowser und ein Webserver sind darum die verbreitete Grundlage für Webanwendungen.

Der Web-Browser nimmt Eingaben des Nutzers entgegen und schickt passende Befehle an den Server. Die vom Server gelieferten Daten werden grafisch angezeigt. Das HTTP Protokoll beschreibt diese Interaktion (auf Seite 95), wobei für die Beschreibung der Gestaltung der Bildschirme die HTML Sprache verwendet wird (auf Seite 74) .

Der Webbrowser stellt eine Webseite dar, wie sie in HTML beschrieben und über das Web geliefert wurde. HTML erfasst bestimmte Interaktionen des Benutzers und leitet sie an den Server weiter: die Ergebnisse der Interaktion werden als Anforderung neuer Webseiten, allenfalls um Parameter erweitert, an den Server geschickt.

Mehrere Browser, die auf den ersten Mosaic (als Firma dann Netscape) Browser von 1993 zurückgehen, sind heute verfügbar (Internet Explorer von Microsoft[2], Firefox, Opera und ähnliche, die frei verfügbar, oft sogar Open Source, sind). [3]

[2] Netscape hatte 1994 eine Dominante Stellung, die durch die Gratisverteilung von Internet Explorer durch Microsoft innert weniger Jahre zerstört wurde und durch ein Quasi-Monopol von Microsoft abgelöst wurde.

[3] Problematisch waren bestimmte Versionen des Internet Explorers von Microsoft, die nicht die Vorgaben es W3C Standards folgten. Internet Explorer hatte einige Jahre praktisch eine Monopolstellung, die weiter ausgebaut werden sollte, indem Abweichungen vom Standard, auf die die Web-Designer Rücksicht nehmen müssen, andere Browser Anbieter behindert und aus dem Geschäft drängen sollten. Diese Bemühungen scheinen nicht erfolgreich und andere Browser, (besonders Firefox) haben Marktanteile gewonnen. Gleichzeitig wurde Microsoft von der Europäischen Kommission zu riesigen Busen wegen unfairen Marktverhaltens verurteilt.

4.2.1 Gefahren der Browser Mono-Kultur

Durch die weite Verbreitung weniger Browser und weniger Betriebssysteme kann sich Schadcode verbreiten, indem von einem Rechner, auf dem der Schadcode eingenistet ist, andere Rechner angesteckt werden; die Ausbreitung erfolgt sehnlich wie bei einer Vireninfektion und man spricht deshalb anschaulich von Computer Viren. Viren nutzen zu ihrer Verbreitung Fehler in Programmen, die häufig eingesetzt werden, aus; Rechner mit seltenen Betriebssystem und seltener verwendete Browser sind weniger gefährdet, weil sich eine Infektion nicht leicht ausbreiten kann.

4.3 Server

Server und Browser sind standardisiert in ihrer Kommunikation durch HTTP (auf Seite 95) und HTML Standards (auf Seite 74), so dass im Prinzip fast jeder Browser mit fast jedem Server zusammenarbeiten kann. haben. Hier verwenden wir zuerst einen der üblichen Server für Text, der als HTTP kodiert ist. Am weitesten Verbreitet ist der Apache HTTP Server.

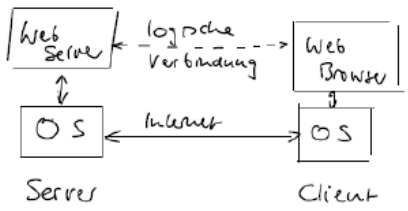


Abbildung 4.4: Server und Browser Prozess kommunizieren durch das Internet

Der Server Prozess ist das Gegenstück beim Server zum Browser Prozess beim Client. Ein http Server im Internet wird von einem Browser über http angesprochen und muss entsprechend reagieren. Im einfachsten Fall wird eine Webadresse und ein Dateiname an einem bestimmten Port mit dem Befehl GET übermittelt und der dort wartende Server sendet die entsprechende Datei, die HTML Code enthält,

an den Browser, der diese als Webseite darstellt. Diese Grundfunktion dient schon als erster Test, ob eine Apache Installation erfolgreich ist).

4.4 Datenspeicherung beim Server

Zur Speicherung dient traditionell eine relationale Datenbank - die quelloffene Datenbank MySQL ist sehr verbreitet und dient hier als Beispiel.

4.5 MySQL Datenbank

Wir verwenden die frei erhältliche Open Source Datenbanksoftware MySQL, die unter Linux und andern Betriebssystemen läuft. Andere RDB funktionieren im Prinzip gleich und werden ähnlich mit einem Webserver verbunden (z.B. Access, Postgres, Filemaker). Auf dem Webserver wird die RDB installiert und mit lokalen Programmen mit Daten gefüllt; wozu verschiedene Hilfsmittel zur Verfügung stehen, um kleinere oder größere Datenmengen zu laden und interaktiv zu bearbeiten, abzufragen, etc.

Zum Bearbeiten der Datenbank, Einfüllen von Daten etc. brauche ich MySQL Workbench; die über das Web den Zugang zur Datenbank erlauben. Ähnliche Hilfsmittel stehen für andere RDB (z.B. Access, Postgres) zur Verfügung. Funktionstests der RDB und später Tests, ob die SQL Abfragen (7.3.2) richtig sind, werden mit diesen Werkzeugen ausgeführt. Die Verbindung zwischen Datenbank und Web Server wird durch eine Erweiterung des Webservers mit der PHP Sprache erreicht.

4.6 PHP

Apache wird durch einen Zusatz erweitert, sodass Anweisungen in der Sprache PHP in die Webseite eingebaut werden dürfen (4.2). Diese werden nicht wie die HTML Texte an den Browser weitergeleitet, sondern werden auf dem Server ausgeführt und können z.B. Abfragen an die RDB enthalten. Die Ergebnisse der Verarbeitung werden als HTML kodiert und an die entsprechende Stelle in den HTML Text eingefügt.

Apache mit Erweiterung filtert PHP Anweisungen aus den verlangten Webseiten heraus, führt diese aus und ersetzt die PHP Anweisungen durch das Ergebnis als HTML (4.5). Als Resultat steht eine Webseite nur in HTML beschrieben bereit, die an den Browser geschickt wird und dort angezeigt werden kann.

PHP ist eine vollständige Programmiersprache (6.7) und wird in Webanwendungen sehr häufig für die Verbindung zwischen Webserver und Datenbank herangezogen. Datenbank abfragen und Umwandlung

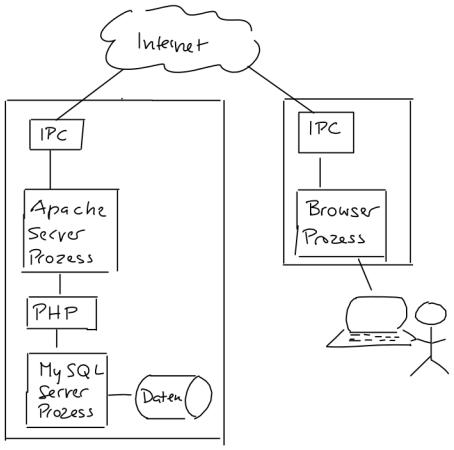


Abbildung 4.5: PHP verbindet den Apache Dienst mit dem MySQL-Dienst und erlaubt so den Zugriff vom Browser auf die Datenbank (IPC = Prozess, der mit Internet verkehrt)

der Ergebnisse in HTML kodierte Darstellungen sind damit leicht zu programmieren.

4.7 Dynamische Anwendungen

Nach jeder Interaktion durch den Benutzer eine neue Web-Seite die in HTML kodiert ist, zu laden, erlaubt keine flüssige Interaktion, die Wartezeiten bis die Seite übertragen und dargestellt ist, stört den Benutzer. Eine bessere Lösung erfordert, dass auf der Klient-Seite sofort ohne Übertragung über das Netz reagiert wird. Das erfordert, dass im Browser eine Programmiersprache läuft - das ist heute üblicherweise Javascript (6.9).

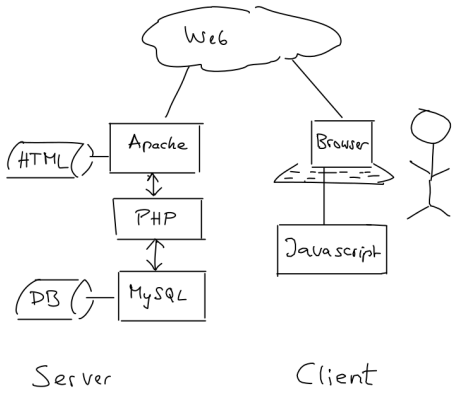


Abbildung 4.6: Das Einbinden von Karten in die Applikation

4.8 Installation

Das Installieren von LAMP ist einfach, besonders , weil die Programme über das Web gratis heruntergeladen werden können — was an der Universität meist rascher geht als zu Hause.

Für die Installation finden sich Anleitungen auf dem Web. Auf meinem Linux Rechner muss ich nur in der Paket Verwaltung *synaptic* die Pakete apache2, mysql-server, php5 ankreuzen, um die Installation anzustoßen. Werden andere Pakete für die Installation gebraucht, so sorgt synaptic automatisch, dass diese in der richtigen Version geladen werden und kontrolliert die Abhängigkeiten.

Auf der Seite https://library.linode.com/lamp-guides/debian-7-wheezy finden sich Instruktionen für die Details der Installation:

4.8.1 Installation von Apache

Es ist zu prüfen, dass die Berechtigungen richtig gesetzt und die minimal notwendigen Öffnungen von Ports gemacht sind. Meist sind frisch installierte Systeme vorsichtigerweise nach außen vollständig geschlossen. Es muss allenfalls festgelegt werden, wo die ins Web gelieferten Seiten gespeichert werden sollen und Nutzer, die Web-Seiten schreiben oder ändern können sollen, müsen entsprechende Berechtigungen für diesen Ordner (directory) erhalten.

In einem Debian basierten Linux system ist in einem Terminal einzutippen:
do apt-get update
sudo apt-get install apache2

Die Installation eines andern Webservers ist ebenso einfach und in 5.6.2 beschrieben.

4.8.2 Installationstest für Web Server

Nach der Installation eines Servers ist es am einfachsten, auf dem gleichen Rechner einen Browser aufzumachen und dort die Seite “localhost/index” aufzurufen. Man sollte dann den Inhalt der Seite im Directory /var/www/index.html angezeigt bekommen. Die Antwort sollte sein „It works!“, denn im im Directory /var/www steht eine Datei index.html, die die Zeile

```
<HTML><body><h1>It works!</h1></body></HTML>
```

enthält. Diese Datei wird an dem Browser übermittelt und schreibt "it works" auf den Schirm. Dieser einfache Test zeigt, dass der Apache Server richtig installiert ist und lokal auf den default Port horcht.

Default Port für HTTP: port 80

Es ist sicherzustellen, dass nach einem Neustart des Betriebssystems der Prozess des Apache Servers automatisch gestartet wird; das wird meist bei der Installation automatisch eingestellt und muss überprüft werden, am einfachsten, indem man den Server neu startet und schaut, ob die Seite erneut angezeigt werden kann.

Danach ist zu prüfen, ob der Server von andern Rechnern angesprochen werden kann. Senden sie mit dem Browser ihres Rechners z.B. http://gi34.geoinfo.tuwien.ac.at/index und ihr Browser zeigt den File der auf dem Server gi34.geoinfo.tuwien.ac.at mit dem Namen index.html steht. http ist nicht unbedingt erforderlich, da der Browser das als default einsetzt. Beim diesem Rechner, den wir für Tests benutzen ist Apache frisch installiert.

Ist mit localhost (d.h. der Browser adressiert direkt den Server auf den gleichen Rechner) die Seite auf dem Server selbst erreichbar, aber nicht von einem anderen Rechner aus, so ist etwas mit den Adressen und Ports nicht in Ordnung. Es ist dann zu überprüfen, ob der Port 80 freigegeben ist und kein Firewall dazwischen steht.

Es ist nun vorsichtig, mit einem Text Editor den file in /var/www/-index.html zu verändern (z.B. zu "It still works !!!") und zu sehen, ob die änderung angezeigt wird. Was kann schief gehen?

- der angezeigte file ist irgendwo anders gespeichert (weil in der Konfiguration von Apache ein anderes wurzelverzeichnis für www eingetragen),
- der file lässt sich nicht ändern (weil er vor dem überschreiben durch gewöhnliche User geschützt).

Es ist übrigens nicht erforderlich, HTML zu erlernen, denn statistische Bildschirme können mit den üblichen Texteditoren MSware oder Open Office Writer erstellt werden und dann statt als .doc oder .odt Datei einfach als .html Datei gespeichert werden. Mit einem Webbrowser und diesen Funktionen von Apache können wir statische Bildschirme an den Browser des Nutzers liefern, was wir in Kapitel 5 für die Erstellung eines Mock-up der Anwendung ausnützen werden.

4.8.3 Installation der Datenbank

Die Datenbank und der Zugang über PHP erfordert möglicherweise ebenfalls Anpassungen. In der Grundeinstellung ist meist nur ein lokaler Zugang zur Datenbank erlaubt, so dass MySQL workbench möglicherweise nur lokal (d.h. auf dem Rechner, auf dem MySQL läuft)

funktioniert. Das ist ausreichend, da der Zugriff auf die Datenbank vom lokalen Apache-Server her erfolgt und die Nutzer der Datenbank nicht direkt darauf zugreifen dürfen.

Im Terminal eintippen:
sudo apt-get install mysql-server libapache2-mod-auth-mysql php5-mysql mysql-workbench

4.8.4 Funktionstest der Datenbank

Zum Funktionstest, bauen Sie eine einfache Tabelle z.B. ein halbes Dutzend Namen und Telefonnummern in ihrer Datenbank und versuchen mit einer SQL Abfrage Daten herauszusuchen. Brauchen Sie Zugang von aussen (z.B. mit SQL Browser) so müssen sie sich entsprechende Rechte einräumen und den DB Server nach außen öffnen.

Es gilt zu prüfen, dass der SQL Server nach einem Betriebssystem Neustart wieder startet, was üblicherweise bei der Installation eingestellt wird, aber geprüft werden muss.

4.8.5 Funktionstest PHP

Schliesslich ist PHP zu installieren und zu testen, ob die PHP Erweiterung in Apache eingebaut und aktiviert ist. Das geschieht, indem ein einfacher PHP Befehl in einen File eingebaut wird und dieser über einen Browser vom Apache Server angefordert wird.

Zur Installation von PHP
sudo apt-get install php5 libapache2-mod-php5 php5-mcrypt

Ein minimales PHP Programm ist z.B.

```
<?
Phpinfo();
?>
```

Dieses ist typisch in einer Datei testphp.html in /var/www bei der Installation von PHP schon eingestellt. Wenn sie in ihrem Browser *localhost/testphp* öffnen, sehen sie einen Text der zeigt, dass PHP mit Apache zusammen funktioniert. Es wird eine sehr detaillierte Beschreibung der PHP Installation angezeigt.

4.9 Spezialisierte Editoren für Webseiten

Für die Entwicklung einer Webanwendung, die im Browser läuft, dh. die hier anvisierte Architektur, braucht einen Editor mit dem HTML Seiten erstellt werden können, und allenfals PHP und Javascript programmiert werden kann.

Statische HTML Seiten können mit einem Texteditor wie z.B. Word erstellt und als .html (statt .doc) gespeichert werden.Es gibt zwei typen von Editoren um Web Seiten zu erstellen:

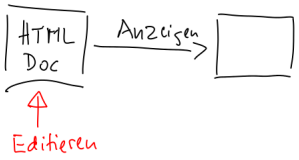


Abbildung 4.7: Editor für HTML

WYSIWYG Editoren Die Seite wird dargestellt, wie sie sich präsentiert und diese Darstellung wird direkt verändert. Die Nutzung eines Texteditors um HTML kodierte Dokumente zu erstellen entspricht diesem Verfahren; BlueGriffon ist ein auf die Erstellung

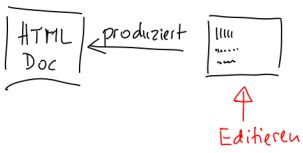


Abbildung 4.8: WYSIWYG Editor für HTML
WYSIWYG what you see is what you get

von Webseiten zugeschnittener Editor, der aber ähnlich wie Word funktioniert.

Code Editoren Es wird HTML Code bearbeitet und es gibt eine bequeme Art, zu sehen, wie das Ergebnis dargestellt wird. BlueFish ist ein Beispiel für diesen Zugang.

Wir verwenden hier BlueGriffn, das kostenlos unter den meisten Betriebssystemen installiert werden kann [http://bluegriffon.org/].

4.10 Entwicklungsumgebung - Programmierwerkzeuge

Für Tests mit einem Server auf einem zweiten Rechner sind Hilfsmittel, die es erlauben, den andern Rechner vom ersten aus zu steuern, nützlich. Am üblichsten ist der Aufbau einer SSH Verbindung zum andern Rechner, was Zugang wie bei einer lokalen Konsole (Kommandozeile oder Terminal) erlaubt. Will man einen graphischen Zugang mittels eines Fensters mit z.B. RDP, das mir von meinem Client einen Zugang als Benutzer zum Server gibt — ganz als ob ich dort an der Tastatur sitzen würde und den Bildschirm mit Maus bedienen würde. An vielen Servern sind deshalb gar keine Schirme und Tastaturen angeschlossen und sie werden immer mit einem solchen Fernzugang gewartet.

Mögliche Probleme: die nötigen Prozesse für Internet, Apache oder MySQL sind nicht am Laufen; das wird überprüft, indem die Liste der laufenden Prozesse angeschaut wird. Wenn die Installation richtig gemacht wird, starten diese automatisch bei jedem Neustart des Servers; auch das ist zu testen!

Fast unumgänglich ist die Erweiterung des Browsers um ein Werkzeug um Fehler zu finden und zu erkennen, ein sog. Debugger erforderlich. Für Mozilla Firefox ist das ein Plug-in namens Firebug; für andere Browser gibt es ähnliches. Firebug ist hilfreich für das Suchen von Fehlern in HTML und CSS: es wird damit möglich, zu kontrollieren, was beim Browser als Text ankommt, nachdem Apache, PHP und anderes den ursprünglich gespeicherten Text bearbeitet haben.

5
Eine einfachen Web-App auf dem Webserver

In diesem Kapitel erstellen wir eine Web-App mit den bisher eingeführten Methoden und mit Betonung auf Einfachheit. Ziel ist die Demonstration der dazu notwendigen Schritte. Es ist günstig, wenn Apache (oder ein ähnlicher Web Server) lokal installiert ist und die aufrufbaren Seiten in einem Folder /var/www liegen und wir darin ein Unterfolder angelegt haben (hier /var/www/word) in den wir files schreiben können - üblicherweise ist /var/www nur für root zum schreiben offen.

Abbildung 5.1: Eine einfache Applikation nur mit HTML Seiten

Das Beispiel implementiert das Skeleton der "Mittagessen!" Applikation, für die die Benutzeroberfläche beschrieben wurde (2.1.14). Um zu zeigen, wie einfach die Installation und wie gering die Ansprüche an die Hardware sind, wird die gesamte Installation auf einem Raspberry Pi durchgeführt, ohne dass sich eine änderung zu einem andern Rechner, der als Webserver dient ergibt.

5.1 Gestaltung der Webseiten in Editor

Mit einem üblichen Editor (word, libreoffice) erstellen wir die Seiten und speichern sie in HTML format ab. Libreoffice kann die Seite auch in der Web Form Darstellen, was bei dem trivialen Format nicht viel unterschied macht. für die Seite A_Mittagessen.html habe ich eingetippt:

```
Mittagessen!                    Ende
Nutzer          Andre           Präferenzen
Password        xx              Heute

Markiere  Rechner
```

und danach als HTML gespeichert. Dann kann die Seite mit "localhost/word/A_Mittagessen.html" im Browser angezeigt werden. Die andern Text Seiten (Empfehlungen und Präferenzen) werden gleich erstellt. Die Darstellung ist nicht überzeugend, aber zumindest wird der Text angezeigt.

Die Lage des Restaurantes kann auf einer Skizze angezeigt werden. Zum Beispiel zeigt man das Restaurant in Google Maps im Browser und zieht einen Screenshot. Aus diesem kann dann eine Graphik herausgeschnitten werden (z.B.. mit GIMP oder einem andern Graphik Programm) und diesen in die HTML Seite eingeklebt.

5.2 Verlinken der Seiten

Die Links, die im Browser stehen, können nun im Text Editor in die HTML files kopiert werden. für den File für die Präferenzen-Wahl tippt man ein:

```
Präferenzen Frank                Ende: http://localhost/word/X_ende.HTML

Italienisch
Hausmannskost    Pfeil aufwärts
Asiatisch              Pfeil abwärts
Pizza
                                  Heute: http://localhost/word/C_Empfehlung.HTML
```

Wenn im Browser auf die Links geklickt wird, öffnet sich der entsprechende File und wir können prüfen, ob die geplanten Abläufe durch zuspielen sind und sehen, was fehlt. Dazu ziehen wir die Protokolle, die zur überprüfung der Abläufe gemacht worden sind, heran (2.1.14).

5.3 HTML

Die produzierten Seiten sind vom Text Editor in HTML kodiert und können mit einem Editor angeschaut werden, indem man sie als Text öffnet, so dass der Code nicht beachtet, sondern dargestellt wird. Die erste Seite enthält:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
        <META HTTP-EQUIV="CONTENT-TYPE" CONTENT="Text/HTML; charset=utf-8">
        <TITLE></TITLE>
        <META NAME="GENERATOR" CONTENT="LibreOffice 4.1.3.2 (Linux)">
        <META NAME="AUTHOR" CONTENT="Andrew Frank">
        <META NAME="CREATED" CONTENT="20140315;134823599870923">
        <META NAME="CHANGEDBY" CONTENT="Andrew Frank">
        <META NAME="CHANGED" CONTENT="20140315;143636371664098">
        <STYLE TYPE="Text/css">
        <!--
                @page { margin: 0.79in }
                P { margin-bottom: 0.08in }
                A:link { so-language: zxx }
        -->
        </STYLE>
</HEAD>
<BODY LANG="en-US" DIR="LTR">
<P STYLE="margin-bottom: 0in">Mittagessen!                                Ende:
<A HREF="http://localhost/word/X_ende.HTML">http://localhost/word/X_ende.HTML</A></P>
<P STYLE="margin-bottom: 0in">Nutzer          Andre                      Präferenzen:
<A HREF="http://localhost/word/E_Präferenzen.HTML">http://localhost/word/E_Präferenzen.HTML</A></P>
<P STYLE="margin-bottom: 0in">Password  xx                          Heute:
<A HREF="http://localhost/word/C_Empfehlung.HTML">http://localhost/word/C_Empfehlung.HTML</A>  </P>
<P STYLE="margin-bottom: 0in"><BR>
</P>
<P STYLE="margin-bottom: 0in">Markiere Rechner</P>
```

```
</BODY>
</HTML>
```

Man könnte jetzt von diesem Gerüst aus weitergehen und den HTML Code editieren und schönere Webseiten generieren., wir können aber auch einen mehr für Web layout geeigneten Editor instalieren und einsetzen. Der HTML Code, der von normalen Text Editoren erstellt wird, ist im allgemeinen nicht besonders günstig für die manuelle Weiterverarbeitung.

5.4 Vereinfachung des Ablaufes

Um die Implementierung zu vereinfachen, kann man den Ablauf am Anfang so verändern, dass das Login als erster Schritt geschieht und dann zu einem ersten (informativen) Screen führt:

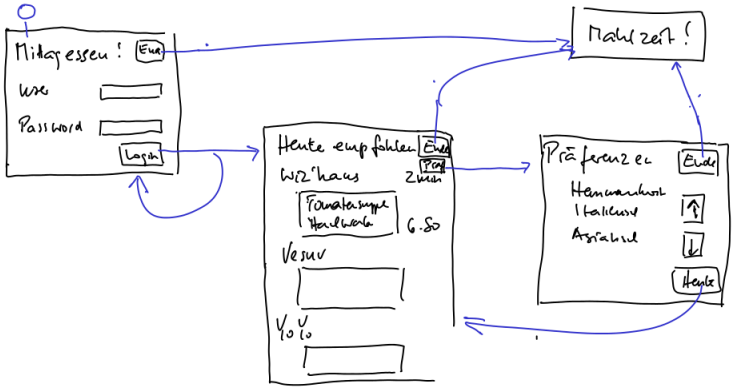


Abbildung 5.2: Der vereinfachte Ablauf, vergleiche mit 2.7

Das ist einfacher zu implementieren als die erste Idee (2.7), weil der Login-Ablauf isoliert wird.

5.5 Layout

Das Layout ist anspruchsvoll, weil die Darstellung der Seiten sich automatisch der Grösse des Schirmes anpassen muss; eine einfache Skalierung ist nicht ausreichend, weil damit Schrift entweder viel zu gross oder viel zu klein dargestellt wird; wird die Seite Grösser muss Schriftgrösse leicht angepasst werden, Text Felder sollten grösser werden, damit mehr Text ohne Scroll operation angezeigt werden kann und Abstände zwischen Textbestandteilen dürfen grösser werden. Diese Anpassungen dürfen aber nicht dazu führen, dass Ausrichtung (alignement) verloren geht - sonst sieht die Seite entsetzlich aus.

Für das Layout der Elemente auf der Seite werden wir eine *Tabel* Struktur wählen. Das ist nicht die beste Art aber es ist eine einfaches Verfahren, damit Ausrichtung sowohl horizontal als auch vertikal erreicht werden kann. Besser an verschieden große Schirme anpassbar

Abbildung 5.3: Der Startbildschirm mit Login

Abbildung 5.4: Wie Empfehlungen dargestellt werden

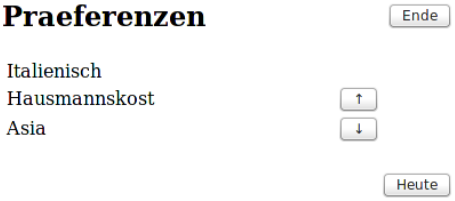


Abbildung 5.5: Präferenzen können

wäre der Einsatz von CSS (Cascading Style Sheets), was aber schwierig zu lernen ist - nicht zuletzt, weil es zu viele Möglichkeiten gibt, ähnliches zu erreichen.

Bei *Tables* ist es schwierig zu entscheiden, ob man sich auf eine minimale Grösse festlegt und damit vermeidet, dass kleine Darstellungen unsinnig aufgeblasen werden aber damit auch die Anpassung an ein Window, das kleiner als das festgelegte, verhindert - Benutzer müssen dann scrollen. Wenn man hingegen keine Grösse festlegt, so passt sich die Seite kleinen Fenstern an, wird aber auf großen sehr auseinander gezogen.

Die drei wichtigsten Fenster der Anwendung sind am Rand dargestellt und für alle Seiten haben wir HTML Code. Wenn in den Empfehlungen auf den Namen des Gasthauses getippt wird, so geht ein Fenster mit einer Karte, in der das Gasthaus eingezeichnet ist, auf.

Mit dem BlueGriffon Editor kann man die Qualität der Webseite gegenüber den Empfehlungen des W3C prüfen.

Der Ablauf der Applikation kann nun getestet werden, auch wenn keine echten Daten dabei sichtbar werden und dem Auftraggeber gezeigt werden. Eine Demonstration macht meist deutlicher, was und wie die Applikation funktionieren wird und bringt viele wertvolle Anregungen zur Verbesserung; allenfalls ist der Auftrag und die Entschädigung den korrigierten Anforderungen anzupassen!

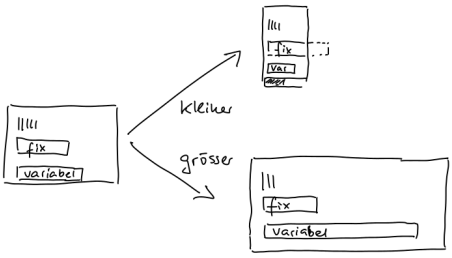


Abbildung 5.6: Veränderung der Dartellung eines Elementes mit fixer und eines mit variabler Breite

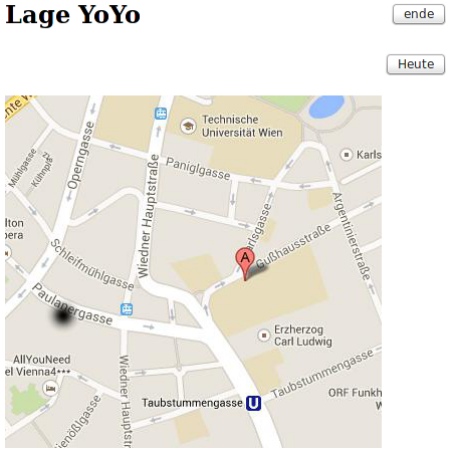


Abbildung 5.7: Die Lage eines Gasthauses
Technisch wird eine solche Darstellung ohne echte Daten Mock-up genannt.

5.6 *Installation auf web Server*

Als webserver verwende ich hier einen Raspberry Pi [1] - ein Einplattinen-Rechner mit ARM Prozessor der “nackt” ca. Euro 30 kostet. Das Betriebssystem und alle Daten werden auf einer SD Karte gespeichert (ich habe 16 GB, was für ziemlich viel web content reichen sollte - kostet 15 Euro); als Stromversorgung dient ein Mobiltelefon-Ladegerät (6 Euro), zusammen also ungefähr 50 Euro - nicht viel für ein System zum Testen eines web auftrittes und allenfalls sogar für eine kleine Homepage.

Betriebssystem ist Raspbian Linux [2] - eine Debian Distribution. Als Web Server installiere ich ngingx (“engine x”); auch apache ist verfügbar:

[1] http://en.wikipedia.org/wiki/Raspberry_Pi oder http://www.raspberrypi.org/

[2] http://www.raspbian.org/

5.6.1 *Internet Anbindung*

Zuweisung einer statischen WebAddress im Bereich, der mein A1 Modem versorgt (im File /etc/network/interface - Beispiel 10.0.0.103)

```
auto lo

iface lo inet loopback
iface eth0 inet static
```

```
        address 10.0.0.103
        netmask 255.255.255.0
        gateway 10.0.0.138

iface default inet dhcp
```

Alternativ wäre eine dynamische Zuweisung der Adresse und eine Lokalisierung des Raspberry Pi mittels des avahi service (entpsricht bonjour für Macs) möglich; das vermindert Probleme mit doppelt belegten statischen Adressen und die Notwendigkeit, die Adresse zu memorisieren, bringt aber ein zusätzlicher Dienst, der konfiguriert werden muss.

Jetzt kann man prüfen, ob der Server auf "ping 10.0.0.3" (in einem Terminal zu einzugeben) reagiert - d.h. der Server hat die gewünschte IP und ist lokal von einem andern Rechner am gleichen Modem erreichbar. wenn der Rechner mit ssh ("ssh pi@10.0.0.103" - verlangt das Passwort für den User "pi") erreichbar ist braucht er weder Bildschirm noch Tastatur sondern wird "ferngesteuert".

Dann muss unbedingt der Hostname in /etc/hostname und in /etc/hosts eingetragen werden (hier "pi103").

```
127.0.0.1       localhost
127.0.0.1       pi103
```

Von meinem Rechner übertrage ich mein lokales Zertifikat auf den Raspberry PI mit "ssh-copy-id pi@10.0.0.103"; beim erstellen einer ssh Verbindung werde muss ich dann nicht mehr das Passwort eingeben.

5.6.2 Installation eines alternativen Web servers: nginx

Ich wähle nicht Apache sonder nginx als Webserver; beide sind für den Raspberry Pi erhältlich und sind im Betrieb ähnlich.

Der web Server wird mit "sudo apt-get install nginx" installiert. Der Service wird mit "sudo service nginx start" gestartet. Im Web Browser eines andern Rechners soll dann für 10.0.0.103 eine Web Seite mit "Welcome to nginx!" erscheinen ; damit sind wir sicher, dass der Server installiert und läuft. Diese Seite ist in "/usr/share/nginx/www" gespeichert (nicht in /usr/share/www wie für Apache), was sich aber durch einen eintrag in der Konfigurationsdatei von nginx ändern lässt.

Hello world für nginx!

Am besten mit "reboot" prüfen, ob nach einem restart des Betriebsystems noch da; wenn nicht, muss der Service in die liste der automatisch startenden Programme eingebunden werden (z.B.. mit "sudo insserv nginx default").

Man kann nginx konfigurieren, für den ersten Start reicht die Default konfiguration.

5.6.3 Installieren der Applikation auf dem Server

Es gilt nun die files, die wir local erstellt haben (bei mir in Code/web/Script) auf den Server in den ngingx Folder zu kopieren. Zuerst erstellen wir einen link für den www folder /home/pi durch "ln -s /usr/share/nginx/www www" und darin einen folder Script, für den pi Schreibrechte hat. in diesen kopieren wir vom hauprechner die files mit "scp -r Script pi@10.0.0.103:www/Script". Aufpassen, dass nicht nur alle *.HTML files mitkommen, sondern auch das Bild mit der Lage eines Gasthauses (ein pgn file, dessen Adresse im HTML file steht) kopiert wird!

Danach kann von jedem Browser (im lokalen Netz) mit "10.0.0.103/Script/m2.HTML" die Startseite der Anwendung aufgerufen werden. Wenn alle links in allen Seiten und der Verweis auf die eingebundene Karte relativ waren, so funktionieren alle Buttons und zeigen die richtigen Seiten; wird ein Link nicht ausgeführt, so kann entweder der File fehlen oder der Link nicht relataiv zur aufrufenden Seite (nur der filename) sondern absolut (ganzer Pfad), sein.

5.6.4 Zugriff von Außerhalb des lokalen Netzes

Damit der zugriff nicht nur vom lokalen Netz (dh. alle Geräte am gleichen Modem) sondern von außen (dh. vom ganzen Internet) möglich ist, müssen wir verkehr gesendet an die von außen sichtbare IP Adresse des Modems auf den lokalen Web Server "forwarden". Das geschieht mit einer "port forwarding" einstellung am Modem. Ihr Modem kann im allgemeinen mit dem web Browser aufgerufen werden (bei A1 ist das üblicheweise auf 10.0.0.138) und dann können die einstellungen dort verändert werden.

Es muss ein port forwarding für HTTP protokol (dh. port 80) auf die lokale Adresse des web servers eingestellt werden, und zwar so, dass der port 80 vom Modem auf den port 80 des Servers geleitet wird (es wäre möglich, einen eingehenden port auf einen andern port eines gerätes umzuleiten).

5.6.5 Stabile web Adresse für den Server

Der Server ist jetzt aus dem web unter der Adresse des Modems zugänglich. Diese ändert sich aber bei den meisten provider regelmässig und ist schwierig zu merken. Eine symbolische Adresse bekommt man - ohne kosten - von einem service für dynamische DNS . Ein einfaches Angebot bietet https://freedns.afraid.org/. Man kann dort zu einer der vorhandenen Domains (z.b. us.to) eine subdomain eingeben (ich habe auf.us.to, bitte etwas anderes wählen!) und zu dieser dann weitere subdomains. Für den Server mit dem Beispiel habe ich pg.auf.us.to

dyndns = dynamischer DNS dienst

gewählt.

Bei dem service trage ich für jede subdomain eine IP ein - die jetzt gültige des Modems (im sogenannten A record). Damit der service immer weiss, welche IP meinem Modem im moment zugeteilt ist, schlägt mir die webseite vor, dass mein Server regelmässig eine Anfrage mit einem bestimmten (geheimen) schlüssel startet. Der dazu nötige Code wird im file crontab eingetragen und dann alle paar Minuten ausgeführt. Crontab ist in einem Linux system der file, wo regelmässig erfolgende Aktionen eingetragen und dann ausgelöst werden.

```
# You might need to include this path line in crontab, (or specify full paths)
PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin

0,5,10,15,20,25,30,35,40,45,50,55 * * * * sleep 47 ;
    wget --no-check-certificate -O - https://freedns.afraid.org/dynamic/update.php?SZXZzB3.....OjEwOTY2NzQ0
    >> /tmp/freedns_pg_auf_us_to.log 2>&1 &
```

Mit diesem Methoden sind die Seiten des Mockup unter pg.auf.us.to/Script/mt2.HTML erreichbar (meistens - es ist für einige minuten möglich, dass die IP Adresse des Modems geändert worden ist und der dyndns das noch nicht mitgekriegt hat).

5.7 Zusammenfassung

Wir haben eine Demoversion der Applikation erstellt, die zeigt, wie der Webserver funktionieren wird und diese auf einem externen Server installiert und getestet. Der Mock-up enthält keine Daten und ist damit nicht funktionsfähig. Dennoch hilft eine solche Demoversion um Fehler im Ablauf zu erkennen, den Ablauf zu verbessern oder zu vereinfachen. Eine Demonstration beim Auftraggeber hilft diesem frühzeitig zu erkennen, ob die bestellte Anwendung seinen Bedürfnissen entsprechen wird und Änderungen in die Wege zu leiten; wenn mehr als Details geändert werden, so ist das unbedingt als Änderung des Auftrages wiederum schriftlich festzuhalten und es muss vielleicht der Preis angepasst werden - das mag schwierig sein, ist aber besser in diesem Stadium, wenn noch wenig Kosten aufgelaufen sind, zu erledigen als später. Der Auftraggeber eines Werkes kann den Auftrag zurücknehmen, muss aber alle aufgelaufenen Kosten und für den entgangenen Gewinn (vielleicht 5% des Gesamtpreises), wenn das Werk fertig erstellt worden wäre, aufkommen; einen Werkvertrag nach der Demonstration, die zeigt, dass das Werk nicht dem entspricht, was der Auftraggeber braucht, aufzulösen ist viel klüger, als nach Ablieferung des Werkes zu streiten.

Im nächsten Teil werden diese “stubs” (dh. HTML Seiten ohne Funktionen) mit programmierten Funktionen gefüllt.

Teil III

Funktionen in Webseiten einbauen

In diesem Teil erweitern wir den Mock-up um Funktionen, die Daten aus einer Datenbank heraussuchen und Anzeigen oder Karten, von Google Maps, damit verbinden.

6
Sprachen und Protokolle

Die oben beschriebenen Software-Komponenten wirken zusammen, indem sie Daten austauschen: der Browser schickt Daten zum Server und erhält von dort Daten zurück etc. Damit die Zusammenarbeit funktioniert, müssen die ausgetauschten Daten in vereinbarten Formaten kodiert sein und der Austausch muss festgelegten regeln folgen, s.g. Protokolle. Ein Protokoll legt fest, wie die Interaktion zwischen zwei Teilen erfolgen soll und welche Mitteilungen dabei ausgetauscht werden. Eine Sprache gibt die Möglichkeit, etwas zu beschreiben; die Mitteilungen die nach einem Protokoll ausgetauscht werden, müssen in einer Sprache abgefasst werden.

Protokoll - Regel über den Ablauf einer Interaktion, bei der Mitteilungen ausgetauscht werden

Sprache - Regeln, wie Zeichen zusammengefügt werden um eine bestimmte Bedeutung zu transportieren

Meiner Meinung nach braucht die Entwicklung einer Webapplikation zu viele verschiedene Sprachen (6.1). In meinem Blog "Thinking in Geras" [1]habe ich geschrieben:

> An adventure in real world programming with 5 languages
>
> I wanted to build a tool which I can use on my mobile phone (Android, HTC Desire); not finding a path to compile Haskell, my prefered language to anything executable on the mobile phone, I decided to follow advice from Building iPhone Apps with HTML, CSS, and JavaScript to build a web application using Haskell to construct the web Server with Yesod .
>
> What I did not expect was the number of un-coordinated and slightly different languages I had to learn. The project included a total of 5 different computer languages: - Haskell, for programming the application, - CouchDB (fortunately accessible from Haskell), - Javascript, to Code the building of the index in couchDB, - HTML, to describe the structure of the output, - CSS, to describe the appearance of the web page. Modern computer languages are quite similar, but differ in small ways: how do they deal with white space and what is white space? how to insert a comment? how to 'escape' a character, which has a particular meaning?
>
> In the process I was reading documentation, and could not, for example, find an advanced, well structured description how to use HTML and CSS; the specification is readable, but dös not indicate how the various features, which are producing seemingly the same effects, are

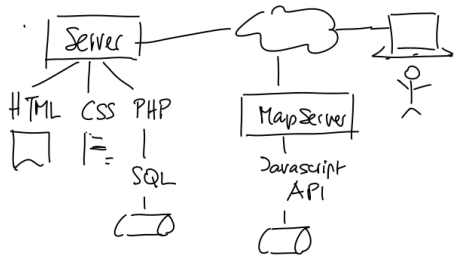


Abbildung 6.1: The different languages used to connect the components

[1] andrewufrank.blogspot.com

best used, to avoid their quirks. Not only had I to learn the languages, but also to find specialized editors and tools for debugging (e.g. a Firefox extension to see what HTML and CSS arrived at the Browser) - each tool with its own set of 'small differences'.

I succeeded, but two questions remain, one existential, one social:

- is it possible, to build a web application in a single language (specifically: in Haskell)?
- why do the 'real programmers' accept this sorry state of the world?

The first question leads to a testable hypothesis, which I hope to report about soon; the second reminds me of an observation of a social science researcher (I do not remember who it was), who pointed out that social institutions oscillate to a middle ground of complexity. If they are so simple that everybody can understand them, ingenuous people make them more complex; if they are so complex that nobody can use them any more, efforts to simplify are made. I guess, social systems need complexity to differentiate between amateurs and experts; complexity assures income to experts. This seems not only to apply to the legal system, with layers and tax advisers, but also to programmers?

Abbildung 6.2: John Backus

Formale Sprache besteht aus
- Zeichenvorrat (Alphabet)
- Regeln (Syntax)

6.1 Formale Sprachen

Anweisungen für Rechner müssen in einer formalen Sprache ausgedrückt werden. Formale Sprachen können sehr einfach sein (zwei Kommandos: ein oder aus) oder komplex, wie die heutigen Programmiersprachen. Eine formale Sprache besteht aus einem Vorrat von Zeichen (Alphabet) und Regeln (Syntax), wie diese Zeichen zusammengesetzt werden können, so dass das Ergebnis Teil der Sprache ist.

Zum Beispiel werden üblicherweise Zahlen aus den Ziffern $0, 1, ..9$ zusammengesetzt, wobei aber nicht jede Kombination zulässig ist; die 13, 204 und 1003 sind "richtige" Zahlen, 073 ist im allgemeinen nicht akzeptiert als ganze Zahl. Für moderne Programmiersprachen prüft der Compiler, ob der eingegebene Programmtext den Regeln folgt und damit interpretierbar und ausführbar ist, oder ob er syntaktische Fehler enthält und damit kein akzeptierbares (legal) Programm darstellt.

Der Compiler "parst" einen Text um zu prüfen Test ob syntax richtig ist

Die Regeln für die Syntax einer Sprache werden typisch in der Backus-Naur-Form (BNF) beschrieben und aus einer derartigen Beschreibung lässt sich der erste Teil des Compilers, nämlich der Parser, automatisch konstruieren.[2] Der *Parser* prüft das Programm auf syntaktische Richtigkeit und wandelt syntaktisch richtige Programme in eine interne Form um, die dann weiterbearbeitet wird und schließlich ausgeführt wird.

Syntaxregeln in BNF (Backus-Naur-Form)

Für jede Sprache müssen bestimmte Verfahren festgelegt werden um z.B.

[2] John W. Backus hat den ersten FORTRAN Compiler als Mitarbeiter von IBM 1954 geschaffen; er wurde 1977 mit dem Turing-Award ausgezeichnet (6.2). Seine Turing-Award Lecture ist lesenswert!

John Backus. Can programming be liberated from the von neumann style?: A functional style and its algebra of programs. *CACM*, 21: 613–641, 1978

Ein syntaktisch richtiges Programm ist keine Garantie, dass auch das gewollte ausgeführt wird!

- Kommentare als nicht zum Text gehörig markiert zu werden,
- Zeichen, die sonst eine besondere Bedeutung haben, als solche (d.h. ohne die besondere Bedeutung) in den Text einzufügen,
- Linearisierung von Baumstrukturen,
- Referenzen in nicht-hierarchischen Strukturen.

Unglücklicherweise wurden viele der Sprachen, die im Web verwendet werden, zur Beschreibung scheinbar sehr einfacher Sachverhalte gebaut und sind mit der Einsicht in die Komplexität der Aufgabe langsam und nicht systematisch gewachsen. Oft kann die gleiche Aufgabe auf verschiedene Art erreicht werden, was für den Programmierer vielleicht einfacher ist und viele "literarische Stile" erlaubt. Das Lesen und Verstehen von Programmen, die andere geschrieben haben, macht es aber sehr viel schwieriger.

Kommentare sind Texte, die nicht zum Programm gehören und meist Anmerkungen für menschliche Leser sind. Typisch werden "–", "#" oder"##" am Zeilenanfang als Kommentar-Zeichen verwendet, die eine ganze Zeile als nicht zum Programm gehörig markieren und die vom Compiler nicht beachtet wird. Das wir häufig benutzt, um Anweisungen auszuschalten ohne dass man sie aus dem Text entfernen muss.

escape sequences: wird ein Zeichen in der Sprache mit besonderer Bedeutung verwendet, z.B. Anführungszeichen um eine Zeichenkette (String) zu begrenzen, so kann dieses Zeichen nicht mehr im Text erscheinen. Um es dennoch einfügen zu können, werden "escape sequences" definiert, die erlauben, das Zeichen seiner speziellen Funktion zu entbinden und es als solches in eine Zeichenkette einzufügen. z.B. um Anführungszeichen in eine Zeichenkette, die durch Anführungszeichen begrenzt wird, einzufügen, müssen sie doppelt geschrieben werden: "er sagte """Ja"""!". In andern Sprachen können einfache Anführungszeichen innerhalb doppelter (und umgekehrt) verwendet werden.

Linearisierung von Baumstrukturen: Es ist vielfach üblich, einen Sachverhalt als hierarchisch gegliedert zu verstehen (von der Gliederung eines Studiums, die Gliederung Firmen und Universitäten oder die Gliederung eines Textes); eine hierarchische Gliederung kann als Baumstruktur dargestellt werden.
In einer Datei müssen Daten aber in eine lineare Reihe gebracht werden, sollen Daten über eine Leitung übermittelt werden, so werden sie linear in der Zeit dargestellt. Es ist möglich, durch das einfügen von Klammern eine Baumstruktur in eine lineare Struktur umzuwandeln und die Baumstruktur daraus wieder verlustfrei herzustellen (6.3).
In jeder Sprache muss festgelegt werden, in welcher Art die Klammern in den Text eingefügt werden, so dass sie sich nicht mit andern Zeichen im Text überschneiden (siehe oben escape sequences!). Glücklicherweise zeichnet es sich ab, dass wenige Linearisierungsmethoden allgemein verwendet werden; XML (6.4.1) und JSON (6.4.2) sind die gebräuchlichsten.

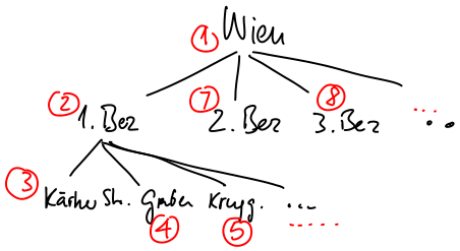


Abbildung 6.3: Der Baum wird linearisiert als: Wien(1.Bez,(Kärtnerstr. Graben,Kruggasse), 2.Bezirk(...), 3.Bezirk (..),...).

Referenzen: Die Linearisierung mit Klammern geht nur für Bäume, dh. Hierarchien, nicht aber für allgemeine Graphen! müssen diese linearisiert werden, also z.b. in einer Datei abgelegt werden, so muss eine Form von Referenzen und Referenzierungen zwischen teilen verwendet werden.
Sofern nicht hierarchische Strukturen linearisiert werden sollen, so müssen Referenzen so ausgestaltet werden, dass sie nicht mit dem Inhalt kollidieren (wiederum "escape sequences"). RDF (10) ist eine allgemeine Art, komplexe Strukturen darzustellen und in eine lineare Form zu bringen.

6.2 Kodierung von Zeichen

Üblich war eine Kodierung von Text als Zeichen mit 8 bit (7 bit + prüfbit), also ein Byte pro Zeichen (ASCII Standard 1963). Damit lassen sich die üblichen Zeichen des lateinischen Alphabetes, Zahlen und eine Anzahl Sonderzeichen darstellen, total 127 Zeichen, davon 32 reserviert für Signale (z.B.. Klingelton, neue Zeile, Tab etc. 6.4).

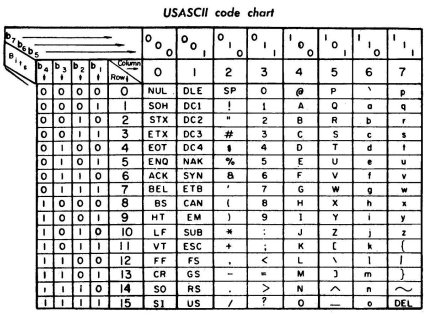
Erweiterungen für Spezialzeichen, die in andern Ländern (d.h. nicht USA) üblich sind, (Umlaute, Akzente, Pfundzeichen etc.) haben zu verschiedenen Code-Tabellen geführt, selbstverständlich für jeden Hersteller (Windows, Apple) und jedes Land verschieden, so dass Texte beim Transfer von einem zum andern Computer oft entstellt werden. Diese Verwirrungen verschwinden langsam, da der neue Unicode Standard zunehmends angewendet wird.

Unicode (UTF) soll dem abhelfen. UTF-8 ist eine Kodierung von Zeichen, wobei die Codes unterschiedlich lang sind. Für die üblichen Zeichen wird die gleiche Kodierung wie ASCII verwendet und ein Zeichen in 8 bit dargestellt. Für die (europäischen) Sprachen typische Akzente, das griechische und kyrillische Alphabet etc. wird mit einem 2 Byte Code erfasst, der durch eine 10 Start Sequenz im ersten Byte angekündigt wird. Das gibt etwa 2^{16}= 65,536 darstellbare Zeichen, was nicht für alle Zeichen der Welt (denken sie an Japan, China und Korea) ausreicht. Für diese Sprachen werden Zeichen mit drei oder vier Bytes dargestellt.

Es ist zu hoffen, dass der UTF Standard überall greift (auch bei den Programmiersprachen) und die Texte, die durch falsche Interpretation der Zeichenkodierung entstehen, langsam verschwinden. Dennoch ist es höchst empfehlenswert, in Namen von Web Services, Email Adressen, Betreff-Zeilen von Email, Filenamen etc. etc. nur die Standard ASCII Zeichen zu verwenden; alles andere kann zu Überraschungen führen oder wird kriminell ausgenützt (5) .

Der Nachteil von UTF-8 ist, dass die Anzahl Zeichen und der benötigte Speicherplatz nicht in einem einfachen Verhältnis zueinander

ASCII American Standard Code for Information Interchange enthält
!"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
`abcdefghijklmnopqrstuvwxyz{|}~



Row	b_7 b_6 b_5	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
0	b_4 b_3 b_2 b_1	0	1	2	3	4	5	6	7
0	0 0 0 0	NUL	DLE	SP	0	@	P	`	p
1	0 0 0 1	SOH	DC1	!	1	A	Q	a	q
2	0 0 1 0	STX	DC2	"	2	B	R	b	r
3	0 0 1 1	ETX	DC3	#	3	C	S	c	s
4	0 1 0 0	EOT	DC4	$	4	D	T	d	t
5	0 1 0 1	ENQ	NAK	%	5	E	U	e	u
6	0 1 1 0	ACK	SYN	&	6	F	V	f	v
7	0 1 1 1	BEL	ETB	'	7	G	W	g	w
8	1 0 0 0	BS	CAN	(	8	H	X	h	x
9	1 0 0 1	HT	EM	)	9	I	Y	i	y
10	1 0 1 0	LF	SUB	*	:	J	Z	j	z
11	1 0 1 1	VT	ESC	+	;	K	[	k	{
12	1 1 0 0	FF	FS	,	<	L	\	l	|
13	1 1 0 1	CR	GS	-	=	M	]	m	}
14	1 1 1 0	SO	RS	.	>	N	^	n	~
15	1 1 1 1	SI	US	/	?	O	_	o	DEL

Abbildung 6.4: Die klassische Kodierung von Zeichen nach ASCII

stehen. können

6.3 Auszeichnungssprachen (Markup Language)

Leslie Lamport (6.6) 1994 weiterentwickelt; beide erhielten den Turing Award.

Markup languages lösen das allgemeine Problem, dass ein Text für die Darstellung aufbereitet werden muss. Es muss festgelegt werden, welche Größe die Zeichen haben, Farbe, Schriftgrad und -typ müssen bestimmt werden und die Verteilung des Textes auf der Seite (Layout) fixiert.

Eine markup language ist eine Auszeichnungssprache, mit der Text ausgezeichnet wird, das hieß früher wurde dem Buchdrucker Anweisungen an den Rand geschrieben, welchen Schrifttype (z.B.. Times Roman), welche Größe [3]die Schrift haben soll und aus welcher Stil (Fett=bold, kursiv=italics, etc.). Diese Angaben haben die Vorläufer von HTML ursprünglich gemacht.

Damit ein Text gleichmäßig strukturiert wird, ist es vorteilhaft, nicht jeden Titel einzeln mit einer Auszeichnung zu versehen, sondern nur zu markieren, dass dies ein Titel einer bestimmten eben (z.B. ein Abschnitt) ist, und dann für alle Abschnitte festzulegen, wie diese darzustellen sind (dies wird heute von CSS übernommen 6.6). Wir erwarten z.B., dass ein Buch Schriftgrad und -typ für Titel, Text und Fußnoten konsistent verwendet.

Es ist vorteilhaft, den Text und die Auszeichnung, die nur die Darstellung nicht aber den Inhalt beeinflusst, getrennt zu beschreiben. Das erlaubt einerseits, Texte durch ändern der Auszeichnung anders Darzustellen (z.B. für verschiedene Größen eines Bildschirmes anzupassen) aber auch verschiedene Texte in eine einheitliche Darstellung zu bringen.

Eine markup language besteht aus Anmerkungen, die in den Text eingefügt werden und einem style sheet, dass allgemein beschreibt, wie die Anmerkungen im Text ausgeführt werden müssen. Die Anmerkungen im Text müssen wiederum so codiert sein, dass sie nicht mit normalem Text zusammenfallen; eine übliche Konvention war, dass Zeichen, die mit einem Punkt beginnen am Anfang der Zeile markup ist (z.b. “.hl 1” oder “.p”).

LaTEX ist eine der am häufigsten verwendeten Markup language (neben HTML), wobei LATEX Auszeichnungen oft noch “von Hand” geschrieben wird, aber zunehmens verwendet man für die Generierung von ausgezeichneten Texten (HTML oder LaTEX) WYSIWYG Edidorten (4.9), z.B. Lyx (wie hier). LaTEX baut auf TEX auf, das 1978 von Donald Knuth (6.5) bereitgestellt wurde und dann von

Abbildung 6.5: Donald Knuth

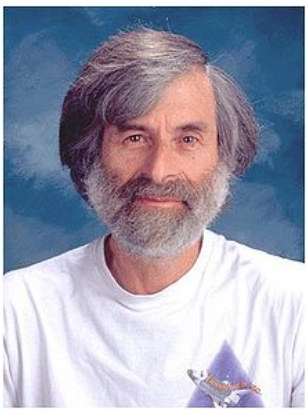


Abbildung 6.6: Leslie Lamport

Markup Languages trennen
- Kodierung von Darstellungstypen (Title, Author, Zeilenabstand) und
- Anwendung der Darstellung im Text.

[3] Schriftgrössen wurden im Buchdruck in 12 punkten - das traditionelle Maß für Schriftgrößen angegeben. 1 pt = 0.3527 mm, 72 pt = 1 inch; alte Maße variieren.

(What you see is what you get - was man sieht ist was man hat)

6.4 Kodieren von strukturierten Daten

6.4.1 XML

XML ist eine allgemeine, weit verbreitete Methode, strukturierte Daten in eine lineare Form überzuführen und so darzustellen, dass andere Programme sie verwenden können.

Baumstrukturen können eindeutig linearisiert werden (durch sogenannte Traversals, hier 'depth first') indem im linearen Text durch klammern die Baumstruktur gezeigt wird (6.1).

```
<?xml version="1.0" encoding="utf-8"?>
<Address xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:noNamespaceSchemaLocation="SimpleAddress.xsd">
  <Recipient>Mr. Walter C. Brown</Recipient>
  <House>49</House>
  <Street>Featherstone Street</Street>
  <Town>LONDON</Town>
  <PostCode>EC1Y 8SY</PostCode>
  <Country>UK</Country>
</Address>
```

Abbildung 6.7: Beispiel XML
(http://en.wikipedia.org/wiki/XML_Schema_%28W3C

Diese Klammern werden mit dem Typ des Elements markiert und zwar zu Beginn und am Ende; damit wird deutlich wo eine Klammer beginnt und endet. Im allgemeinen wird das auch durch einrücken graphisch deutlich gemacht, in XML spielt aber die graphische Anordnung für die Interpretation keine Rolle (in andern, moderneren Sprachen, wird das verwendet.

Daten, die in XML kodiert werden sollen, können durch ein Schema beschrieben werden, dass angibt welche Datenfelder vorhanden sein müssen und welche Werte sie haben können. Dieses Schema ist wiederum in XML kodiert:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element Name="Address">
    <xs:complexType>
      <xs:sequence>
        <xs:element Name="Recipient" type="xs:string" />
        <xs:element Name="House" type="xs:string" />
        <xs:element Name="Street" type="xs:string" />
        <xs:element Name="Town" type="xs:string" />
        <xs:element Name="County" type="xs:string" minOccurs="0" />
        <xs:element Name="PostCode" type="xs:string" />
        <xs:element Name="Country" minOccurs="0">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="IN" />
              <xs:enumeration value="DE" />
              <xs:enumeration value="ES" />
              <xs:enumeration value="UK" />
              <xs:enumeration value="US" />
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Abbildung 6.8: Das XML Schema zum obigen Datensatz

6.4.2 JSON

Eine alternative Art der Linearisierung von Daten, die nahe der Strukturierung von Daten in Java ist und vor allem zusammen mit Javascript verwendet wird, ist JSON. Sie ist kompakter als XML und etwas leichter zu lesen.

```
{
    "firstName": "John",
    "lastName": "Smith",
    "age": 25,
    "address": {
        "streetAddress": "21 2nd Street",
        "city": "New York",
        "state": "NY",
        "postalCode": 10021
    },
    "phoneNumbers": [
        {
            "type": "home",
            "number": "212 555-1234"
        },
        {
            "type": "fax",
            "number": "646 555-4567"
        }
    ]
}
```

Abbildung 6.9: Eine Person mit Adresse und zwei Telefonnummern in JSON

6.5 Die Web-Sprache HTML

Die meisten Ressourcen im web werden in der Sprache Hypertext Markup Language (6.3) kodiert, die den Browser anweist, wie der Text darzustellen ist. Diese Festlegungen von Tim Berners-Lee 1991 (2.24) festgelegt, sind die Grundlage des Erfolges von WWW.

Die Darstellung des Text “Hello World!” in einem Fenster mit dem Titel “Hello HTML” sieht so aus:

```
<!DOCTYPE HTML>
<HTML>
  <head>
    <title>Hello HTML</title>
  </head>
  <body>
    <p>Hello World!</p>
  </body>
</HTML>
```

Es fällt auf, dass der Text eingerahmt ist von Klammerausdrücken, die paarweise den Beginn und das Ende einer Einheit ankündigen. Das Dokument hat einen Kopf (<head> bis </head>) und einen Körper (<body> bis </body>). Diese Kodierung entspricht (ziemlich) dem

HTML = Hypertext Markup Language

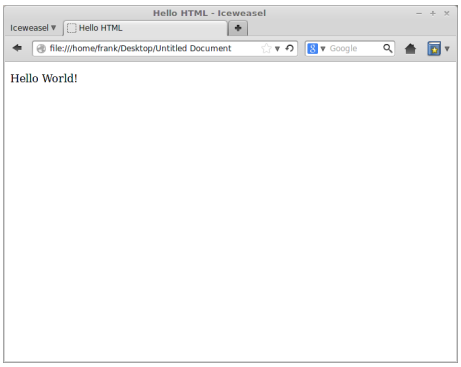


Abbildung 6.10: Ergebnis von “hello World”

XML Format (6.1).

HTML ist eine auszeichungssprache für Hypertext , das heißt, Text der nicht linear dargestellt ist, sondern vom Benutzer die Fortsetzung eines Textes aus verschiedenen alternativen weiterlesen kann (2.1.18). In HTML werden Referenzen auf andere web Ressourcen als URL (3.9) eingebaut; wenn der Leser auf diese klickt (meist blau und unterstrichen ausgezeichnet) wird ein GET request (im HTTP Protokol 6.9) für diese Ressource gestartet und die entsprechende Seite vom Server angefordert und dann vom Browser angezeigt.

6.5.1 DOM Document Object Model

Eine Sprache beruht auf einem Modell des zu beschreibenden Sachverhaltes; zu den Sprachelementen in HTML, die die Speicherung von Daten erlauben gehört ein Modell, wie die Daten strukturiert sind. Zu HTML gehört eine objekt-orientierte Struktur der Objekte auf der Seite, mit der Objekte die in HTML (und XML) strukturiert sind, bearbeitet werden können. Es beschreibt eine Baumstruktur, die aus dem linearen HTML Text aufgebaut wird und zeigt, wie die Teile des Baumes mit Programmen in Javascript manipuliert werden können. Es ist möglich, nach Teilen des Baumes zu suchen, die durch Eigenschaften beschrieben sind oder die bestimmte Identifiers haben. Im DOM können auch Daten im Browser gespeichert werden.

Mir scheint DOM übermäßig kompliziert, besonders weil viele Spezialfälle und Ausnahmeregeln beachtet werden müssen. Es reicht meistens, Daten in variablen auf der äußersten Ebene abzulegen. Die üblichen Hilfsmittel, web seiten zu erstellen übernehmen das für den Programmierer meist (z.B. jQuery [4]

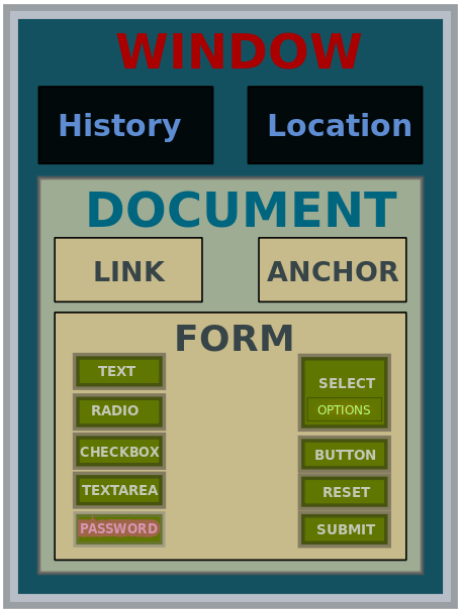


Abbildung 6.11: Das hierarchische DOM (quelle wikipedia Document Object Model)

[4] http://jquery.com/).

6.6 CSS - Cascading Style Sheets

CSS ist der Teil der HTML Markup Sprache, indem die Darstellung der Elemente beschrieben wird. Um in umfangreichen Sammlungen von web Seiten für eine einheitliche Darstellung zu sorgen wird die Darstellung allgemein in einem CSS Dokument beschrieben und dann in die Webseite hereingeladen. Inhalt und Darstellung sind getrennt.

CSS bestimmt, wie die Elemente einer HTML Seite dargestellt werden, wobei verschiedene Aspekte des Elements berücksichtigt werden können.

CSS scheint recht einfach, ist aber in der Anwendung sehr schwierig. Es ist fast unmöglich, die Interaktionen zwischen verschiedenen CSS ausdrücken zu erfassen und entsprechend schwierig zu implementieren und es hat 8 Jahre gedauert, bis vollständige Implementierung vorhanden waren. Es empfiehlt sich, CSS Dateien, die von andern

erstellt worden sind, vorsichtig anzupassen.

Als Beispiel[5]

[5] http://de.wikipedia.org/wiki/Cascading_Style_Sheets

```
p.info {
  font-family: arial, sans-serif;
  line-height: 150%;
  margin-left: 2em;
  padding: 1em;
  border: 3px solid red;
  background-color: #f89;
  display: inline-block;
}
p.info span {
  font-weight: bold;
}
p.info span::after {
  content: ": ";
}
```

Diese Regeln werden auf eine HTTP Seite angwendet:

```
<p class="info">
  <span>Hinweis</span>
  Sie haben sich erfolgreich angemeldet.
</p>
```

und produzieren 6.12.

Hinweis: Sie haben sich erfolgreich angemeldet.

Abbildung 6.12: Das Ergebnis der CSS Anweisung

6.7 *PHP - Die Verbindung von der Datenbank zur Webdarstellung*

Daten in einer Datenbank auf dem Server müssen herausgelesen und in die HTML Form gebracht werden. Das wird nicht vom Apache Server selber sondern mit einer Erweiterung, die mit der Sprache PHP gesteuert wird, erledigt.

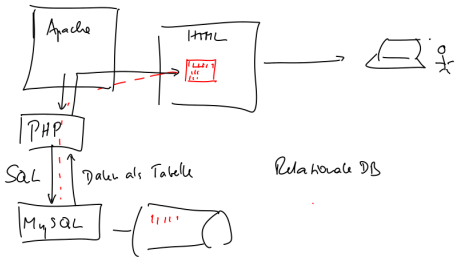


Abbildung 6.13:
PHP Personal Home Page Tools, von Rasmus Lerdorf

```
<!DOCTYPE HTML>
<meta charset=utf-8>
<title>PHP Test</title>
<?php
 echo 'Hello World';
?>
```

Wie aus dem Beispiel hervorgeht, werden im HTML Text teile eingefügt, die nicht mit üblichen HTML klammern "<Name>" sondern mit php klammern "<?php....?> ausgezeichnet werden.

```
<?
$username="username";
$password="password";
$database="your_database";

$first=$_POST['first'];
$last=$_POST['last'];
$phone=$_POST['phone'];
$mobile=$_POST['mobile'];
$fax=$_POST['fax'];
$Email=$_POST['email'];
$web=$_POST['web'];

mysql_connect(localhost,$username,$password);
@mysql_select_db($database) or die( "Unable to select database");

$query = "INSERT INTO contacts VALUES ('','$first','$last','$phone','$mobile','$fax','$email','$web')";
mysql_query($query);

mysql_close();
?>
```

In solchen PHP Klammern können Operationen für die Datenbank eingeschlossen werden. Zum Beispiel werden Werte, die in Feldern der Seite eingegeben worden aus diesen in Variablen (markiert mit vorangestelltem $) eingelesen und dann in eine query eingebaut, die an die Datenbank übergeben wird.

Der Code zwischen <? und ?> wir vom Apache Server an den PHP Prozessor weitergereicht. Dieser führt SQL Operationen aus, stellt die Ergebnisse als HTML dar und liefert das Ergebnis an den Apache Server Prozess weiter, der das Ergebnis an der Stelle, wo der PHP Befehl stand, in die Seite einfügt. PHP ersetzt also teile des HTML Textes durch Daten aus der Datenbank. PHP kann aber viel mehr und ist eine (auch eine) voll funktionsfähige Skripting Sprache. Manchmal ist mehr nicht besser.. und der Trend geht zu kleineren Werkzeugen, die nur eine Funktion möglichst gut erfüllen und sich leicht mit andern zusammensetzen lassen.

6.8 Protokolle

Ein Protokoll legt für eine Kommunikation fest, in welcher Reihenfolge die Partner welche Mitteilungen machen. Gute Sitten legen z.B. fest, wer, wem in welcher Reihenfolge vorgestellt wird, wer wen zuerst zu grüßen hat, etc. Das sind Protokolle und der Protokollchef im Kanzleramt [6] ist zuständig, dass solche Regeln bei offiziellen Anlässen befolgt werden.

[6] im Rang eines Diplomaten, also nicht eine unwichtige Person!

6.8.1 Datenverkehr im Ethernet - CSMA/CD

Bei der Diskussion der untersten Schicht, der Datenübertragung (Transport layer) ist uns das CSMA/CD Protokoll begegnet (3.6.1) ; es regelt, wie mehre Sender ein gemeinsames Medium (z.B. einen Funk-Kanal) nutzen:

Carrier Sense - der Sender hört, ob das Medium frei ist, und sendet nur, wenn frei

Multiple Access - mehrere können den Kanal nutzen

Collision detection - Konflikte werden entdeckt und das Senden wiederholt.

Mit dieser untersten Schicht des WWW hat man üblicherweise keinen Kontakt, da sie von hardware-nahen Treibern implementiert wird und kaum Anlass zu Problemenen gibt.

6.9 Javascript

Javascript ist eine interpretierte Sprache, dh. im Browser wird der Code zur Ausführungszeit in Instruktionen für die Hardware über-

setzt. Die Sprache entspricht den heute üblichen Standards für blockstrukturierte Sprachen und liest sich etwa wie Pascal [7]

C++ oder Java. Mittels der AJAX (Asynchronous JavaScript And XML) Technik werden unter Kontrolle des Programms Seiten geladen.

Javascript ist eine standardisierte [8] funktional aufgebaute Programmiersprache, mit einer Java artigen Syntax. Sie ist objektorientiert, dynamisch Typisiert und wird zur Ausführungszeit ausgewertet, d.h. interpretiert, nicht compiliert [9].

Zum Einbau von Javascript in eine HTML Seite wird der Code in speziellen XML Klammern eingefügt (`<Script> ... </Script>`). Am besten beginnt man mit einem von Google bereitgestellten Beispiel, denn es ist nicht erforderlich den vollen Umfang von Javascript zu erlernen um eine Karte in eine HTML Seite einzubinden.

Aus Sicherheitsgründen kann die Ausführung von Javascript beim Browser verhindert werden. Obgleich Javascript keine vollen Zugang zum System des Klienten gibt, sondern abgeschaltet in einer Sandkasten (Sandbox) ausgeführt wird, enthält es potentielle Sicherheitsrisiken[10], manche Benutzer schalten es generell aus oder erlauben nur vertrauenswürdigen Code vom Web Server auszuführen.

Für die Programmierung von web Applikationen stehen umfangreiche Bibliotheken von Javascript Routinen zur Verfügung (z.b. Couchapp couchapp.org).

Das folgende Dokument zeigt wie eine einfache Javascript Funktion in eine web page eingebunden wird. Nach dem drücken des Knopfes dauert es drei Sekunden und es wird “Hello” ausgeben.

```
<!DOCTYPE HTML>
<HTML>
<body>

<p>Click the button to wait 3 seconds, then alert "Hello".</p>
<button onclick="myFunction()">Try it</button>

<Script>
function myFunction()
{
setTimeout(function(){alert("Hello")},3000);
}
</Script>

</body>
</HTML>
```

```
function factorial(n) {
  if (n == 0) {
  return 1;
  }
  return n * factorial(n - 1);
  }
```

[7] Kathleen Jensen and Niklaus Wirth. *PASCAL User Manual and Report.* Springer-Verlag, second edition edition, 1975

Abbildung 6.14: Niklaus Wirth hat Pascal 1969 beschrieben ; er erhielt den Turing Award 1984.

[8] Standardisiert als ECMAScript 1997

[9] http://en.wikipedia.org/wiki/Javascript

[10] http://en.wikipedia.org/wiki/JavaScript#Browser_a

HTTP Hypertext Transfer Protocol

HTTP ist wohl das wichtigste Protokoll für den Datenaustausch im web und ist die Grundlage für das world wide web. Es ist sehr einfach (darum der Erfolg!) und beruht auf dem Klient - Server Model und einer Anfrage-Antwort Struktur. HTTP setzt das TCP Protokoll (3.8.1) ein, um Verbindungen zu verwalten.

Eine Nachricht besteht aus dem Kopf (message header) und der

eigentlichen Nachricht (message body).

6.9.1 Nachrichtentypen:

Die wesentlichen Nachrichten Typen sind

- GET - Anforderung nach Daten
- POST - Übergabe von Daten an den Server
- PUT - Aufforderung, Daten zu speichern
- DELET - Löschung von Daten verlangen

HTTP ist eine state-less Protokoll; der Server bearbeitet jeweils eine Anforderung oder einen Auftrag und behält keine Information über die Arbeitsschritte. DELETE und PUT sind so zu implementieren, dass sie idempotent sind (dh. eine Aufforderung mehrfach gemacht hat den gleichen Effekt, wie wenn sie nur einmal gemacht worden wäre.) POST ist nicht unbedingt idempotent. GET ist "safe" und hat keine Seiteneffekte, dh. es ist eine reine anfrage nach Daten, die vom Server geliefert werden und verändert den Server nicht, im allgemeinen sollten wiederholte anfragen die gleiche Antwort liefern.

idempotent $f(a) = f(f(a))$

Mit einer CONNECT anfrage, kann die Kommunikation in einen Tunnel gepackt werden, auf dem dann ein anderes Protokoll angewandt wird - meist verwendet um vom normalen HTTP auf HTTPS zu wechseln, dass kryptographisch verschlüsselt ist und als sicher gegen abhören und Beeinflussung von außen gilt.

HTTP wird üblicherweise auf port 80, HTTPS auf port 443; HTTPS ist im Wesentlichen HTTP auf einem SSL/TLS verschlüsselten Übertragungsprotokoll.

Beispiele:

Mit einem GET request wird der Server aufgefordert eine bestimmte Ressource zu liefern. der Körper der Nachricht enthält die Adresse der Ressource als URL. Die Antwort des Servers beginnt mit Metadaten über den Server und die angeforderte Ressource, meist gefolgt vom Text.

Der einfachste GET request ist nach einer Datei (das ist eine web Ressource), die auf einem Server liegt. Diese Ressource wird durch Servername und Pfad zur Datei beschrieben. Enthält die Datei Text, der in HTML kodiert ist, so wird der Browser diesen Text anzeigen. Wenn sie im Browser Feld eine web Adresse eintippen, so wird diese mit einem GET request angefordert.

6.10 Sicherheit

Die Ausführung von Code, der in einer von einem Server gelieferten Seite steht, auf meinem Rechner ist potentiell gefährlich. Der Code könnte bei mir Schaden anrichten oder Geheimnisse ausspionieren; beides wird so weit wie möglich unterbunden.

HTML enthält bindet Code in eine "Sandkiste" ein und verhindert, dass Code außerhalb Schaden anrichten kann. Das gilt insbesondere für Javascript Code. Weiters gilt eine "same origin policy", d.h. ein Script kann nur Daten beeinflussen, die von einem Script, das von der gleichen Web Adresse geladen wurde, beschrieben wurden.

sandbox

7
Das Einbinden von Daten in eine Applikation

Eine statischer Mock-up einer Applikation ist erstellt und getestet. Nun soll die Applikation mit Funktion gefüllt werden. In einem ersten Schritt binden wir eine Datenbank an, in der die applikationsspezifischen Daten gespeichert und nach den Wünschen des Nutzers herausgesucht werden.

Die Datenbank wird üblicherweise auf dem Server betrieben und mittels der PHP Erweiterung im Webserver angebunden (6.7). Der Webserver fragt die Daten von der Datenbank ab und wandelt sie in eine HTML Darstellung um, so dass die dem Browser gelieferte Seite nur statische HTML Angaben für die graphische Ausgabe enthält. Diese Architektur hat Vorteile, weil Rechenleistung nur auf Seiten des Servers benötigt wird und die Darstellung unabhängig von der Geschwindigkeit des Clients ist und sie hat Sicherheitsvorteile, weil alle aktiven (programmierten) Teile im kontrollierten Server ablaufen und auf Seiten des Clients damit kaum Angriffspunkte eröffnet werden.

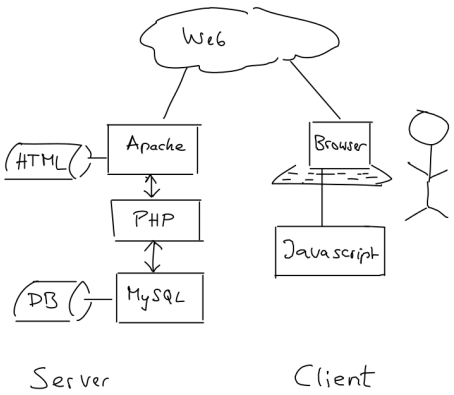


Abbildung 7.1: Das Einbinden von Daten aus Datenbank in die Applikation

7.1 Entwurf der Datenstruktur

Bevor eine Datenbank eingerichtet werden kann, muss man klarstellen, welche Entitäten (Objekte) mit welchen beschreibenden Daten (Felder) und welchen Beziehungen dazwischen, notwendig sind. Entitäten bilden gedankliche Einheiten der Anwendung mit ihren Eigenschaften ab (z.b. Person mit Vorname, Name und Geburtsdatum). Eine Entität ist, was von der Anwendung als konzeptionelle Einheit gesehen wird [1]. Die Grundlagen dazu ergeben sich aus der Machbarkeitsstudie, nun müssen aber die Angaben formalisiert und präzisiert werden (z.B. müssen die Datentypen für die Felder festgelegt werden und alle Felder und Relationen mit Namen versehen). Typisch verwendet man dazu die Darstellung in einem Entity-Relationship- Diagram, wie es Peter Chen vorgeschlagen hat.

Will man zeigen, dass eine Entität PERSON an einem WOHNORT wohnt (als Relation), so zeichnet man als Pfeil von PERSON

[1] William Kent. *Data and Reality - Basic Assumptions in Data Processing Reconsidered.* North-Holland, 1978


Abbildung 7.2: Peter Chen

zu WOHNORT (7.3). Dabei wird angemerkt, welche *Incidenc properties* die Relation hat: eine Person hat einen Wohnort, aber an einem Wohnort kann mehr als eine Person wohnen (als$m$ oder n bezeichnet).

Schwierig wird die Modellierung von Relationen des Typ$m : n$, d.h. dem einen Objekt sind viele andere zugeordnet und umgekehrt auch. Dann muss eine Zwischenentität eingefügt werden; haben die Personen im vorstehenden Beispiel mehrere Wohnorte (z.b. Nebenwohnsitze) so ergibt sich 7.4. ähnlich wird die Beziehung zwischen Studenten und Kursen modelliert: ein Student besucht mehrere Kurse, ein Kurs wird von vielen Studenten besucht.

Mit Werkzeugen, wie z.B. die MySQL Workbench können Datenstrukturen rasch graphisch beschrieben

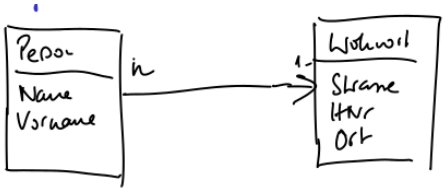


Abbildung 7.3: ER Diagramm

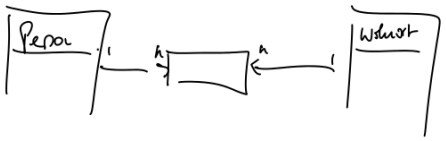


Abbildung 7.4: ER Diagramm mit m:n Relation

7.2 Relationale Datenbanken

Administrative Daten werden heute vorwiegend in Relationalen Datenbanken abgelegt (kleine Datenmengen oft auch in Spreadsheets wie Excell). Relationale Datenbanken organisieren Daten konzeptionell ähnlich wie ein Spreadsheets, nämlich als in Tabellen (2.1). In einer Tabelle steht pro Zeile ein Datensatz, der in jeder Spalte ein beschreibendes Datenelement enthält.

Relationale Datenbanken sind eine gut entwickelte Technologie, die auf grundsätzliche Forschungsarbeiten aus 1970 zurückgehen [2]. Die in der Praxis eingesetzten Systeme unterscheiden sich leicht von der Theorie und sind auf hohen Datendurchsatz getrimmt, um Anwendungen in großen Verwaltungen, die für diese Software sehr viel bezahlen, zu ermöglichen. Systeme wie Postgres und MySQL sind den kommerziellen Systemen für meisten Anwendungen meist ebenbürtig und frei erhältlich.

[2] Edgar F. Codd. *The Relational Model for Database Management.* Addison-Wesley, 1991; E. F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970; and E. F. Codd. Relational data base: A practical foundation for productivity. *Communications of the ACM*, 25(2): 109–117, 1982. Source: Max Egenhofer

Seit etwa 2000, seit dem WWW, das neue Anforderungen an die Datenspeicherung stellt, gibt es zwei alternative Methoden zur Speicherung und Strukturierung von Daten, die andere Ziele verfolgen: NOSQL Datenbanken mit einem andern Konzept für die Verarbeitung von Änderungen (9) und RDF oder Linked Data (10), bei dem Daten und Datenbeschreibung in einem gespeichert und verarbeitet werden. Diese neuen Methoden werden in den anschließenden zwei Kapiteln diskutiert.

7.3 Relationale Algebra - Operationen mit Tabellen

Die Grundidee der relationalen Datenbanken ist, dass Daten häufig natürlicherweise als Tabellen dargestellt werden. In Büros findet man Listen von Personal, mit Wohnadresse und Telefonnummer, in Geographietexten findet man listen von Orten mit Einwohnerzahl und Land

etc. E.F.Codd

hat Tabellen und die Operationen auf Tabellen als relationale Algebra zur Grundlage der relationalen Datenbank Theorie gemacht. Tabellen entsprechen der Sammlung von Werten, die bestimmte Entitaeten beschreiben; dabei wird jeder Entitaet eine Zeile zugeordnet und in den Spalten stehen die beschreibenden Werte für verschiedene Eigenschaften (7.6).

Aus einer Tabelle kann ein Datensatz, der einer Zeile (engl. row) der Tabelle entspricht mittels *Selektion* herausgesucht werden, eine Tabelle kann vereinfacht werden, indem mit *Projektion* Kolonnen (engl. column) weggelassen werden.

Tabellen können miteinander *verknüpft* (engl. join) werden, indem Zeilen mit dem gleichen wert in einer gemeinsamen Kolonne als "Scharnier" verwendet werden und die werte aus beiden Tabellen in eine neue Tabelle kopiert werden. Im Beispiel (7.6) wird eine Tabelle, die Personen beschreibt mit einer zweiten Tabelle, die den Ort mit Postleitzahl enthält, so verknüpft, dass der Wohnort von Personen in einer Tabelle steht.

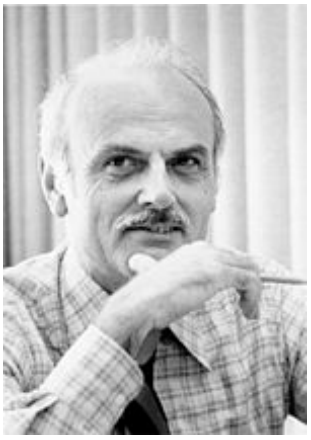


Abbildung 7.5: E.F.Codd - der Erfinder der Relationalen Datenbanken (Turing Award 1981)

Drei Operationen für Relationale Datenbanken:

- select
- project
- join

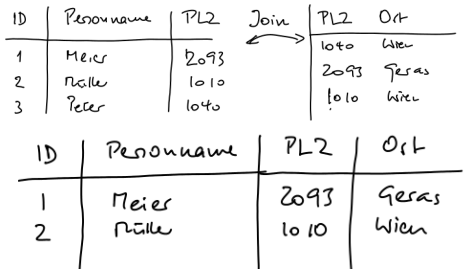


Abbildung 7.6: Verknüpfung zweier Tabellen

7.3.1 Relationale Datenbanken verarbeiten Werte

Relationale Datenbanken sind Wertorientiert, nicht objektorientiert. Zwei Felder mit dem gleichen Namen (z.B. Müller) werden als gleich gesehen, auch wenn zwei verschiedene Personen gemeint sind. Jede Zeile in einer Tabelle muss von jeder andern Zeile verschieden sein - in der Theorie sind doppelte Zeilen verboten, praktische System lassen sie aber zu.

Um duplizierte Zeilen zu vermeiden, werden für Objekte Kennzeichen (engl. identifier) gewählt, damit diese als Werte das Objekt eindeutig bezeichnen, auch wenn mehrere Objekte (z.b. Personen) mit dem gleichen Namen vorkommen. Für jedes Objekt (z.b. Person) wird ein Schlüssel gewählt (z.b. Sozialversicherungsnummer); dann stört es nicht mehr, wenn mehrere Objekte gleiche Eigenschaften (z.b. gleicher Familienname) haben. Die Verknüpfungen erfolgen dann über die Schlüssel. Im obigen Beispiel wären also für die Familien ein Familien-Kennzeichen zu wählen und die Verknüpfung über dieses zu bewerkstelligen; das löst auch das Problem, dass nicht alle Personen in einer Familie den gleichen Familiennamen haben (und bei weitem aus gleichem Familienname nicht auf gleiche Familie geschlossen werden kann).

Codd hat bereits 1979 auf die praktische Erweiterung des Relationalen Modells mittels Schlüssel, die für Entitäten stehen, hingewiesen [3].

[3] E. Codd. Extending the database relational model to capture more meaning. *ACM TODS*, 4(4):379–434, 1979

7.3.2 Abfragesprache SQL

Mittels der Abfragesprache SQL (Structured Query Language) können Daten ausgewählt, gefiltert und Tabellen miteinander verknüpft werden. Im untenstehenden Beispiel werden alle Felder (alle Kolonnen) gewählt (*) der Tabelle Book und zwar werden nur die Zeilen gezeigt, bei denen der price über 100 ist; diese werden nach dem Titel geordnet.

```
SELECT *
 FROM Book
 WHERE price > 100.00
 ORDER BY title;
```

Abbildung 7.7: Michael Stonebreaker, verantworlich für Ingres und Postgres

Ursprünglich wurde SQL als Vereinfachung der vorher vorgeschlagenen Abfragesprachen entwickelt und angenommen, dass Benützer solche Anfragen schreiben würden. Inzwischen ist SQL zum “intergalactic data speak” (Micheal Stonebraker)

verkommen und wird vor allem zur Kommunikation zwischen Programmen eingesetzt. Eine Webapplikation setzt die Eingaben des Benutzers in eine SQL abfrage um, die dann an die Datenbank zur Erledigung geschickt wird.

Das Ergebnis einer SQL Abfrage ist immer eine Tabelle, die in Zeilen Datensätze, die sich aus einzelnen Datenfelder zusammensetzen, enthält. Es ist möglich, das Ergebnis einer Frage in einer zweiten Frage zu verwenden, also z.b. um zuerst den Durchschnittspreis der Bücher zu bestimmen und dann nur diejenigen herauszusuchen, deren Preis geringer als der Durchschnitt ist [4].

[4] http://en.wikipedia.org/wiki/Sql

```
SELECT isbn, title, price
 FROM Book
 WHERE price < (SELECT AVG(price) FROM Book)
 ORDER BY title;
```

7.3.3 SQL zum ändern von Daten

SQL ist nicht nur Abfragesprache, sondern enthält auch Befehle zum Ändern, Einfügen und Löschen von Daten und Datenstrukturen. Mit SQL Befehlen definiert man die Tabellen und die Kolonnen (Datenfelder) und deren Datentypen. Technisch anspruchsvoll ist die Repräsentation der Werte, deren Verarbeitung auf jedem Computer der SQL anbietet, gleich sein muss, obgleich die Repräsentationen oft nicht gleich sind.

Im untenstehenden Beispiel wird eine Tabelle mit Create Table geschaffen und in dieser werden drei Kolonnen mit den Namen my_field1, my_field2 und my_field3 definiert, das erste mit Zahlen (Int) werten, das zweite mit Zeichenketten bis zur Länge 50 und

das dritte mit einem Datum, das nicht den Nullwert haben darf; der Schlüssel, mit dem in dieser Tabelle gesucht werden kann ist die Kombination des ersten und zweiten Feldes:

```
CREATE TABLE My_table(
 my_field1 INT,
 my_field2 VARCHAR(50),
 my_field3 DATE NOT NULL,
 PRIMARY KEY (my_field1, my_field2)
);
```

Der folgende Befehl fügt in eine andere Tabelle My_table2 eine Zeile mit drei Feldern (Kolonnen) mit den Namen field1, field2 und field3 ein und weist die Werte “test”, “N” und einen Nullwert zu.

```
INSERT INTO My_table2
 (field1, field2, field3)
 VALUES
 ('test', 'N', NULL);
```

7.3.4 Editieren von Daten in Datenbanken

Mit Werkzeugen wie z.B. der MySQL-workbench können Datensaetze für Entitaeten als Zeilen in eine Datenbank eingefügt werde, Daten veraendert und SQL Abfragen abgesetzt werden.

7.4 ODBC Schnittstelle (Open Database Connectivity)

Um die geringen Unterschiede zwischen verschiedenen relationalen Datenbanken auf verschiedenen Rechnern auszugleichen, ist eine einheitliche Schnittstelle geschaffen worden. Damit wird vermieden, dass für den Zugriff aus einer Programmiersprache (z.b. PHP) auf eine Datenbank für jede Datenbank eine eigene Schnittstelle geschaffen werden muss und Programme, die diese proprietären Schnittstellen verwenden, beim Wechsel der Datenbank durch den Programmierer angepasst werden müssen. Es genügt, für die ODBC Schnittstelle zu programmieren und die Anpassung erfolgt dann in tieferen Schichten durch Laden von Treibern und Einträgen in einem odbc.ini file.

7.5 Räumliche Daten

Für Postgres gibt es ausgefeilte Zugriffsmethoden für räumliche (geometrische) Daten (PostGIS); ähnliches gilt für das kommerzielle Datenbanksystem Oracle. Die Interfaces (API) sind im allgemeinen den Spezifikationen der vom Open GIS Consortium erarbeiteten ISO Standards (SQL/MM/simple features) entsprechend.

Für eine Datenbank mit raumbezogenen Daten braucht es - wie ich in einem SIGGRAPH paper 1982 vorgeschlagen habe [5] - Erweiterungen für die Abfragesprache. In der Zwischenzeit ist eine Standardisierung erfolgt, die wesentlich auf Arbeiten unserer Gruppe an der University of Maine abstellt:

[5] Andrew U. Frank. Mapquery: Database query language for retrieval of geometric data and its graphical representation. *ACM SIGGRAPH*, 16 (3):199–207, 1982

```
SELECT superhero.Name
FROM city, superhero
WHERE ST_Contains(city.geom, superhero.geom)
AND city.Name = 'Gotham';
```

Damit solche Fragen effizient beantwortet werden können, werden die Daten raumbezogen indiziert, meist mit R-trees [6,7]. Für jedes räumliche Objekt wird das Minimial Bounding Rectangle (MBR, kleinstes umschreibendes axparalleles Rechteck) zur Lokalisierung und bei der Suche herangezogen.

[6] Antonin Guttman. R-trees: A dynamic indexing structure for spatial searching. 1984

[7] C. Zaniolo and C. Delobel, editors. *Application of DBMS to Land Information Systems*, Seventh International Conference on Very Large Data Bases VLDB, 1981

7.6 Verbindung DB zu Webserver

Die Verbindung zum Webserver ist nicht direkt, sondern es werden SQL Abfragen in einem Prozedure-Aufruf in einer Programmiersprache eingebettet (7.8). Dazu dient bei Web-Applikationen meist die PHP Progrmmiersparche (6.7). Vor der Abfrage muss in einem PHP Programm die Datenbank geöffnet werden und damit eine Verbindung zwischen Programm und Datenbank hergestellt werden.

Es wird das SQL Statement als Zeichenkette übergeben, und die zurückgelieferten Daten (in Form einer Tabelle, als Array Zeilenweise geliefert) müssen dann in die HTML Form übersetzt werden. Das übernimmt PHP, das in den Apache Server eingebunden ist und die speziellen PHP befehle erkennt [8].

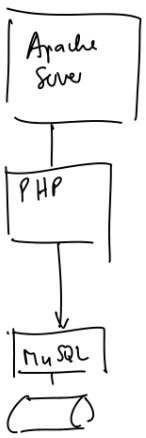


Abbildung 7.8: Verbindung Server - Datenbank

[8] http://en.wikipedia.org/wiki/Php

```
<?php
// Dies könnte z.B. durch einen Nutzer angegeben werden
$firstname = 'fred';
$lastname = 'fox';

// Formuliere Abfrage
// Dies ist die beste Art, eine SQL Abfrage durchzuführen
$query = sprintf("SELECT firstname, lastname, address, age FROM friends
        WHERE firstname='%s' AND lastname='%s'",
                mysql_real_escape_string($firstname),
        mysql_real_escape_string($lastname));

// Führe Abfrage aus
$result = mysql_query($query);

// Prüfe Ergebnis
// Dies zeigt die tatsächliche Abfrage, die an MySQL gesandt wurde und den
// Fehler. Nützlich bei der Fehlersuche
if (!$result) {
        $message = 'Ungültige Abfrage: ' . mysql_error() . "\n";
        $message .= 'Gesamte Abfrage: ' . $query;
        die($message);
        }

// Nutze Ergebnis
// Der Versuch $result auszugeben, erlaubt keine Zugriff auf die Informationen
// der Ressource.
// Eine der MySQ result Funktionen muss genutzt werden
```

```
// Siehe auch: mysql_result(), mysql_fetch_array(), mysql_fetch_row(), etc.
        while ($row = mysql_fetch_assoc($result)) {
        echo $row['firstname'];
        echo $row['lastname'];
        echo $row['address'];
        echo $row['age'];
        }

// Gebe Ressourcen, die mit der Ergebnismenge assoziiert sind, frei
// Dies geschieht am Ende eines Skriptes automatisch
mysql_free_result ($result);
?>
```

PHP ist eine Programmiersprache mit voller Funktionalität, die auch den Zugang zu Datenbanken und die Umformung der Daten zu HTML übernimmt - aber auch vieles andere tun könnte. Das bedeutet leider auch, dass sie etwas komplex ist; es gibt aber mehrere vereinfachende, auf PHP aufbauende und für den Datenbank Zugang spezialisierte Programme (z..B. http://en.wikipedia.org/wiki/Adminer).

7.7 Sicherheit

Es muss sichergestellt werden, dass der Zugang vom Web zu einer Datenbank diese nicht vollständig nach außen öffnet sondern es dürfen nur die Daten abgefragt und geändert werden, welche dafür vorgesehen sind.

Ein beliebiger Zugang wird über mehrere Ebenen verhindert und muss an mehreren stellen ausdrücklich und für einzelne Datenelemente oder Operationen freigegeben werden.

7.7.1 Autorisierung von Nutzern

Verschiedene Nutzer einer Datenbank erhalten verschiedene UserID und mit diesen werden bestimmte Rechte verknüpft (11.5.1). Rechte legen fest, welche Daten wie vom Nutzer mit dieser UserID bearbeitet werden können; dabei wird grob zwischen Zugriff, Veränderung und Änderung der Metadaten (dh. der Tabellenstruktur) unterschieden. Es muss auch festgelegt werden, ob der Nutzer seine Rechte weitergeben kann. Er kann aber einem andern nicht mehr Rechte einräumen, als er selber hat.

7.7.2 Zugang von Außen

Es ist im Prinzip möglich, auf eine MySQL Datenbank von außen über einen bekannten Port zuzugreifen. Das wird im allgemeinen unterbunden, so dass nur SQL Abfragen von innen, d.h. vom Server selber zulässig sind, was im allgemeinen den Zugriff auf PHP beschränkt.

7.7.3 Zugang über Abfragen

Es ist leider möglich, in Webseiten (HTML) in Feldern, in denen Nutzer eingaben machen können, nicht nur vernünftige Eingaben zu machen, sondern auch bösartigen SQL Code einzuschleusen. Wenn der Programmierer der Webseite die Eingabe einfach an eine SQL abfrage anhängt (oder als Text einfügt), so gelangt möglicherweise schädlicher Code zur Ausführung.

Es muss darauf geachtet werden, dass Eingaben des Nutzers vom web Browser nicht ungefiltert an die Datenbank übergeben werden, sondern gefiltert werden. Es ist bei manchen Webapplikationen möglich, in Eingabefelder nicht nur die erwartete Eingaben zu machen, sondern auch noch zusätzliche SQL befehle einzuschleusen (z.B.Delete *). Man schützt sich dagegen, indem man die SQL Statement als Prozeduren ausbildet, die Input Parameter erwarten und in diesen die Eingabe des Nutzers übergibt. [9]. [10]

[9] http://en.wikipedia.org/wiki/SQL_Injektion

[10] Erstaunlicherweise funktioniert die neuere Methode, Web Seiten mit Cross-site scripting anzugreifen, nach ungefähr dem gleichen Prinzip. https://en.wikipedia.org/wiki/Cross-site_scripting

7.8 Was kann schiefgehen?

Die Vorkehrungen gegen böswillige Nutzer, insbesondere die verschiedenen Stellen, an denen der Zugang offen sein muss, stellen alles auch mögliche Fehlerquellen dar, die verhindern, dass eine zugelassene Anwendung nicht auf die Daten zugreifen kann. Es gilt hier - wie anderswo - jeden einzelnen Teil der Kette von Web Server bis zum SQL Server einzeln zu prüfen. Zum Beispiel:

1. Funktioniert der SQL Server? Ist der UserName und das Passwort richtig und der Nutzer mit den richtigen Rechten autorisiert? - Am einfachsten mit einer SQL Workbench, die auf dem gleichen Rechner läuft, prüfen.

2. Kann der Apache Server über PHP auf den MySQL Server zugreifen? auf die interessierenden Daten? Besonders schwierig, wenn MySQL und Apache nicht auf dem gleichen Rechner laufen.

8
Einbinden von Karten

Für das Einbinden von Karten ist es vernünftig, eine zweiten Server zu verwenden - meist ein spezialisierter Server, der nur Karten für Webapplikationen liefert (das ist eine “mehre Server Architektur” 2.26). Dafür gibt es zwei gründe:

- eine Karte oder ein Ausschnitt aus einer Karte als Bild ist ein recht großer Datensatz, den man nicht mehrfach übermitteln will (d.h. vom Karten-Server zum Web-Server und von dort zum Browser),
- viele Webapplikationen brauchen die gleichen Karten als Hintergrund - es drängt sich auf, dafür spezialisierte Server bereitzustellen, die viele Anwendungen bedienen.

Die Architektur für die Einbindung von Karten ist anders als für die Einbindung von Datenbanken, eine Erweiterung des Browsers auf der Seite des Klienten nötig(4.7). Wie Programme für den Zugang zur Datenbank wird Code in die Webseite eingefügt, aber nicht in PHP geschrieben, sondern in Javascript, das auf Seiten des Clients im Browser ausgeführt werden kann.

Dieser Code wird mit der Seite vom Apache Server an den Browser mit geschickt und von diesem ausgeführt (6.9).

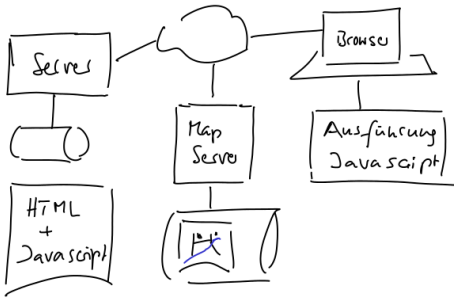


Abbildung 8.1: Applikation mit Einbindung von Karten-Daten von einem andern Server

8.1 Server für Karten

Die Verwaltung räumlicher Daten schien anfänglich kompliziert, weil spezielle Datenstrukturen für die Speicherung [frank vldb] und Erweiterungen für die Abfragesprache nötig waren - wie ich in schon in den 1980ern gezeigt habe.

Ein Durchbruch gelang erst nach 1996, als Mapquest auf Grund der vorhandenen Daten in den USA einen allgemeinen Navigationsdienst im Web anbot. Dieser Service war sehr populär und wurde sehr viel genutzt - um sich Anweisungen zur Fahrt zu unbekannten Zielen auszudrucken und während der Autofahrt zu konsultieren.

Dieser Service und dessen Popularität dürft mit beigetragen haben, dass Google etwas 2005 eine Firma für die Produktion von Karten für die Navigation und eine andere Firma für die Verteilung von Satellitenbilder aufkauften und Google Maps und Google Earth ankündigten.

8.1.1 Kommerzielle Karten-Server

Google scheint geeignet, um möglichst rasch und einfach die Grundzüge der Einbindung von Karten in web Anwendungen zu erlernen. Das API ist ausgefeilt. Nachteilig hingegen ist, dass das System geschlossen und von der Firma Google kontrolliert wird. Diese Kontrolle folgt den wirtschaftlichen Interessen der Firma, nicht derjenigen der Nutzer. [1]

Außer Google bieten auch andere Dienste ähnliche Karten an. Ich habe mich auf die Dienste von Google konzentriert, weil sie das umfassendste und das am besten dokumentierte Angebot bereitstellen.

Microsofts Virtual Earth bietet wohl ähnliche wie Google Maps, aber Zugang zur Dokumentation drängt den potentiellen Nutzer schon die Installation eines (kostenlosen) Microsoft Produkts auf. Das schreckt manche Anwender ab.

[1] Darüber hinaus gibt es wesentliche Kritik an Google's policy für die Speicherung von Nutzerdaten und mangelnde Rücksicht auf die Privatsphäre; es werden offensichtlich alle Anfragen gespeichert und zu einem Profil des Nutzers, das Werbetreibenden verkauft wird, zusammengesetzt. Sucht man einmal nach einem Ort erhält man danach eine Weile Reklame mit diesem Ziel.

8.1.2 Nicht-kommerzielle Karten-Server

Alternativ zu Google Maps gibt es Open Street Map; diese weltweite Organisation versucht offene, zu jeglicher Benutzung freigegebene Karten anzubieten. Die Daten werden von freiwilligen Nutzern gesammelt.Open Street Maps bietet scheinbar 2014 noch nicht ganz die gleichen Dienste wie Google in der gleichen Komfortstufe an. Ich werde das Angebot weiterverfolgen und ein Wechsel ist in Zukunft wahrscheinlich, um die Beschränkungen der Google Dienste (Web-Auftritt muss öffentlich sein) zu vermeiden.

OpenStreetMap ist unter der "Open Data Commons Open Database License" (ODbL). Die Daten dürfen kopiert, verteilt, übertragen und adaptiert werde, solange ein Verweis auf den Ursprung von OpenStreetMap und deren Mitarbeiter. Sofern die Daten verändert oder auf den Daten aufgebaut wird, muss das Resultat mit den gleichen Lizenzbedinungen weitergeben werden.
[2]

Die Verwendung der fertigen Kartenausschnitte ist unter der "Creative Commons Attribution - ShareAlike" (CC-BY-SA) erlaubt.[3]

[2] OpenStreetMap is open data, licensed under the Open Data Commons Open Database License (ODbL).
You are free to copy, distribute, transmit and adapt our data, as long as you credit OpenStreetMap and its contributors. If you alter or build upon our data, you may distribute the result only under the same licence. The full legal Code explains your rights and responsibilities.

[3] The cartography in our map tiles, and our documentation, are licensed under the Creative Commons Attribution-ShareAlike 2.0 license (CC-BY-SA).
http://switch2osm.org/using-tiles/
http://leafletjs.com/examples.htmlhttp://switch2osm.tiles/

8.2 Anfrage von Google Maps

Google stellt auf Webseiten ausführliche Anleitungen mit vielen Beispielen, wie Google Maps eingebaut werden kann, zur Verfügung.

Google hat für ihren Dienst ein Application Programmers Interface

(API) publiziert. In einer API Beschreibung werden die verschiedenen Routinen, die dem Programmierer der Applikation zur Verfügung stehen, festgelegt. Zu jedem Aufruf wird die Funktion beschrieben und die notwendigen Parameter und ihre Bedeutung erklärt. Google stellt mit dem API nicht nur ein Interface für die Darstellung von Karten, mit bestimmten Zeichen und bestimmten Detaillierungsgrad (Level of Detail, LoD) bereit, sondern auch Routinen um Karten mit Wegen zu zeichnen. Geographische Koordinaten werden in φ und λ dezimal angegeben. Eine spezielle Routine erlaubt die Umwandlung von Straßenadressen in Koordinaten und umgekehrt [4].

[4] Rich Gibson and Schuyler Erle. *Google maps hacks*. O'Reilly Media, Inc., 2006

Die Schnittstelle hat in den letzten Jahren beachtliche Entwicklung gemacht und ist nun in Version 3 viel einfacher zu nutzen. (aber nicht für Internet Explorer 6 oder Firefox 2, für diese wäre Version 2 des API zu nutzen).

Es sind verschiedene Formen der Nutzung möglich:

- Einbinden statischer Karten mittels eines HTML Befehls ohne Javascript code
- Einbinden von Karten mittels HTML und Javascript
- Dynamische Karten für mobile Anwendunge (nur Android).

Alle bieten die gleichen Methoden für die Spezifizierung von Karten und die Bedeutung der Parameter ist im wesentlichen auch die gleiche.

8.2.1 Statische Karte

An der Stelle, an der die Karte erscheinen soll wird ein HTML Befehl eingefügt, der die Details angibt. Der Ort kann durch einen Point of Interest oder eine Adresse angegeben werden, die Grösse und der Kartentyp spezifiziert und auch noch verschiedene farbige Marker auf dem Kartenbild plaziert werden.

Auch nationale politische Empfindlichkeiten werden beachtet mit einem zusätzlichen Paramter: "region (optional) defines the appropriate borders to display, based on geo-political sensitivities. Accepts a region code specified as a two-character ccTLD ('top-level domain') value"

```
http://maps.googleapis.com/maps/api/staticmap
?center=Brooklyn+Bridge,New+York,NY&zoom=13&size=600x300
&maptype=roadmap
&markers=color:blue%7Clabel:S%7C40.702147,-74.015794
&markers=color:green%7Clabel:G%7C40.711614,-74.012318
&markers=color:red%7Clabel:C%7C40.718217,-73.998284
&sensor=false
```

Dieser Zugang ist neu; die offensichtliche Beschränkung ist, dass eine solcher Kartenaufrauf nicht länger als 2048 Zeichen (nach URL escaping) sein darf. Das bedeutet, das einfache Fälle mit wenigen Markern, unkomplizierten Pfaden etc. mit dieser Methode erledigt werden können und fuer andere Javascript verwendet werden muss.

Für die Verwendung von Google Maps muss immer angegeben werden, ob man einen Sensor für die Ortsbestimmung (d.h. ein GPS fähiges Gerät) oder nicht; Google benötigt diese Angabe, damit sie den Herstellern der Kartengrundlagen (das sind z.B. die nationalen Vermessungsbehörden) ihre Gebühren richtig bezahlen.

8.2.2 Einbinden von Karten mittels HTML und Javascript

Das folgende Beispiel zeigt wie in einer Seite eine Karte eines bestimmten Ortes, gegeben durch φ und λ angezeigt werden soll.

```
<!DOCTYPE HTML>
<HTML>
  <head>
    <title>Simple Map</title>
    <meta Name="viewport" content="initial-scale=1.0, user-scalable=no">
    <meta charset="utf-8">
    <style>
      HTML, body, #map-canvas {
        margin: 0;
        padding: 0;
        height: 100%;
      }
    </style>
    <script src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false"></script>
    <script>
var map;
function initialize() {
  var mapOptions = {
    zoom: 8,
    center: new google.maps.LatLng(-34.397, 150.644),
    mapTypeId: google.maps.MapTypeId.ROADMAP
  };
  map = new google.maps.Map(document.getElementById('map-canvas'),
      mapOptions);
}

google.maps.event.addDomListener(window, 'load', initialize);

    </script>
  </head>
  <body>
    <div id="map-canvas"></div>
  </body>
</HTML>
```

5

[5] https://developers.google.com/maps/documentation/javascript/tutorial-simple

8.3 Einbinden statischer Karten mittels Javascript Code

Für viele Anwendungen, die geographische Informationen erwarten, ist eine Karte mit markierten Punkten, allenfalls ein Weg oder eine Umfangslinie über den normalen Karteninhalt gelegt, ausreichend. Solche Anforderungen sind mit statischen Karten in Google Maps, die als URL beschrieben werden, einfach zu erfüllen.

Es muss eine Anfrage mit passenden Parametern abgesetzt und die Karte, mit markierten Punkte und Linien, wird als Bild zurückgeliefert und kann im Web Browser angezeigt werden.

Etwas Kommentar zum obigen Beispiel:

Die URL für die Karte wird in einen `<div id="map-canvas"></div>` in HTML eingebaut. Dieser, durch beschriebene Raum auf der HTML Seite wird mit der ElementId im Javascript Code gefunden und dort die gelieferte Karte eingeführt:

```
map = new google.maps.Map(document.getElementById('map-canvas'),mapOptions);
```

Wesentlich sind die möglichen Parameter, die auf von Google ausführlich dokumentiert sind und die in den mapOption record im JSON

Format (6.4.2) dargestellt werden.

```
var mapOptions = {
    zoom: 8,
    center: new google.maps.LatLng(-34.397, 150.644),
    mapTypeId: google.maps.MapTypeId.ROADMAP
  };
```

Der Ort der Karte, die Orte von Markierungen oder die Punkte einer Linie könne in verschiedenen Formen beschrieben werden, am Wichtigsten sind LatLong (d.h. geographische Koordination in dezimaler Schreibweise) oder Straßenadressen mit Hausnummer oder Straßennamen soweit sie Google bekannt sind. Beachte, dass alle Angaben "URL escaped" sein müssen, d.h. keine Leerstellen (durch "+" ersetzen) oder andere verbotene Sonderzeichen enthalten (3.9); programmierte Routinen um eine Benutzereingabe umzuwandeln existieren (siehe Google Geographic Code Services).

8.4 Geokodierung

Die Umwandlung einer Adresse in eine LatLong Koordinate ist ein wesentlicher, aufwendiger Arbeitsschritt. Es ist deshalb zu geocodieren und das Ergebnis mit den zugehörigen Daten zu speichern. Das Beispiel von Google [6] zeigt,wie man auf einer Seite eine Eingabe einer Strassenadresse einbaut und dann die Karte mit einem Marker an der gewünschten stelle zeigt (große Teile des Codes sind gleich wie das erste Beispiel, ich zeige sie hier dennoch vollständig).

[6] https://developers.google.com/maps/documentation/.../simple

```
<!DOCTYPE HTML>
<HTML>
  <head>
    <meta Name="viewport" content="initial-scale=1.0, user-scalable=no">
    <meta charset="utf-8">
    <title>Geocoding service</title>
    <link href="/maps/documentation/Javascript/examples/default.css" rel="stylesheet">
    <script src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false"></script>
    <script>
var geocoder;
var map;
function initialize() {
  geocoder = new google.maps.Geocoder();
  var latlng = new google.maps.LatLng(-34.397, 150.644);
  var mapOptions = {
    zoom: 8,
    center: latlng,
    mapTypeId: google.maps.MapTypeId.ROADMAP
  }
  map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);
}

function codeAddress() {
  var address = document.getElementById('address').value;
  geocoder.geocode( { 'address': address}, function(results, status) {
    if (status == google.maps.GeocoderStatus.OK) {
      map.setCenter(results[0].geometry.location);
      var Marker = new google.maps.Marker({
          map: map,
          position: results[0].geometry.location
      });
    } else {
      alert('Geocode was not successful for the following reason: ' + status);
    }
  });
```

```
}

google.maps.event.addDomListener(window, 'load', initialize);

    </script>
  </head>
  <body>
    <div id="panel">
      <Input id="address" type="textbox" value="Sydney, NSW">
      <Input type="button" value="Geocode" onclick="codeAddress()">
    </div>
    <div id="map-canvas"></div>
  </body>
</HTML>
```

Google bietet auch an, aus einer LatLong Koordinate, z. B. einen Punkt, den der Benutzer angeklickt hat, eine verständliche Ortsbezeichnung für den am nächst gelegenen Punkt mit einer Bezeichnung zu generieren; dies wird als "umgekehrtes Geokodieren" (reverse geocoding) bezeichnet[7].

[7] https://developers.google.com/maps/documentation/...reverse

```
function codeLatLng() {
  var Input = document.getElementById('latlng').value;
  var latlngStr = Input.split(',', 2);
  var lat = parseFloat(latlngStr[0]);
  var lng = parseFloat(latlngStr[1]);
  var latlng = new google.maps.LatLng(lat, lng);
  geocoder.geocode({'latLng': latlng}, function(results, status) {
    if (status == google.maps.GeocoderStatus.OK) {
      if (results[1]) {
        map.setZoom(11);
        Marker = new google.maps.Marker({
            position: latlng,
            map: map
        });
        infowindow.setContent(results[1].formatted_address);
        infowindow.open(map, Marker);
      } else {
        alert('No results found');
      }
    } else {
      alert('Geocoder failed due to: ' + status);
    }
  });
}
```

8.5 Wegbeschreibung und andere Dienste

Die Produktion von *Wegbeschreibung,* die den Benutzer gezeigt und in der Karte markiert werden können, ist ein anderer Dienst von Google, der über eine URL in HTML Code eingebettet werden kann.

```
function codeAddress() {
  var address = document.getElementById('address').value;
  geocoder.geocode( { 'address': address}, function(results, status) {
    if (status == google.maps.GeocoderStatus.OK) {
      map.setCenter(results[0].geometry.location);
      var Marker = new google.maps.Marker({
          map: map,
          position: results[0].geometry.location
      });
    } else {
      alert('Geocode was not successful for the following reason: ' + status);
    }
  });
```

}

Zu Wegbeschreibung können auch *Höhenkarten* aufgerufen werden, so dass Höhenprofile gezeichnet werden können (allgemein, Höhenwerte zu beliebigen Punkten können angefragt werden). Neu ist eine, stärker eingeschränkte, Anwendung um zu einem Punkt die dort befindlichen, Google bekannten, Plätze von Interesse (sogenannte Points of Interest POI) zu finden; da finden sich oft Hotels, Restaurants, Bahnhöfe, Museen, etc.

8.6 Dynamic Maps und andere Ortsbezogene Dienstleistungen

Google bietet - besonders in Verbindung mit dem von ihnen mitentwickelten Android Betriebssystem für Mobilgeräte - verschiedene weiterführende Anwendungen an. Diese werden im Moment (2014) rasch ausgebaut; sie beruhen oft auf sozialen Netzwerken und finanzieren sich durch Werbung.

Teil IV

Alternativen zu den üblichen Web Architekturen

Der Aufbau einer Webapplikation wie in 2.25 gezeigt, entspricht der heute üblichen Praxis (oder der Praxis der letzten Jahre). Es entspricht der Vorstellung, dass eine Firma ein Webportal operiert und darüber die bei ihr vorhandenen Daten zur Verfügung stellt. Das Web war zuerst eine Methode Texte anzuzeigen, allenfalls als Hypertext - d.h. in einem vom Leser gewünschten Reihenfolge - und wurde mit der Zeit um Funktionen erweitert, andere Daten aus einer Datenbank abzufragen und vor der Darstellung zu bearbeiten. Diese Funktionen sind alle auf dem Server ablaufend, weil nur dort Rechenleistung in genügendem Umfang vorhanden war.

Inzwischen ist es möglich, auch beim Clienten programmierte aktionen im Browser durchführen zu lassen (in javasript zu programmieren, siehe ..). Das erlaubt von der strengen Ein-Server-Aarchitektur wegzugehen.

Mehr-Server-Architektur

Inzwischen wird in einer Information auch unterlagen von andern eingebunden, die ebenfalls über das Web beschafft werden. Das geschieht vorallem mit Werbung die als Werbebanner von andern Stellen geliefert werden aber auch mit Karten und ähnlichem Material. Nun wäre es unsinnig, die Daten zuerst von dieser dritten Stelle zum Server liefern zu lassen und dann von dort weiter zu liefern.

Es ist offensichtlich einfacher, wenn die Daten, die von Dritten geliefert werden, nicht erst zum Server geschickt werden und dann von dort an den Client, sondern wenn der client Daten von mehreren Servern direkt erhalten kann. Das kann zu einer moderneren Architektur einer web applikation führen, bei der ein client Daten von verschiedenen Servern anfordert und als eine zusammenhängende Seite darstellt; dies wird z.b. bei der Einblendung von Werbung in Seiten genützt, wo die seiten von verschiedenen servern geliefert und erst beim client zusammengesetzt werden.

Das wurde bereits beim Einbinden von Karten, die von Google geliefert werden und direkt zum Browser geschickt werden, beschrieben (8); es fragt sich nun, ob nicht auch die Datenbank selber als Webserver dienen kann.

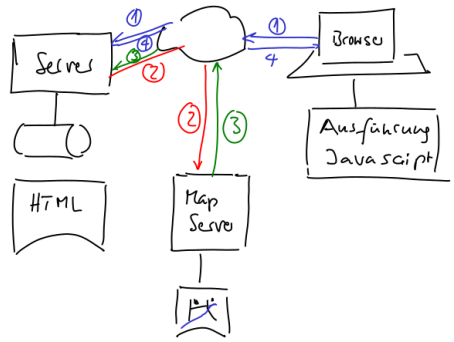


Abbildung 8.2: Datenfluss in einer Ein-Server-Architektur

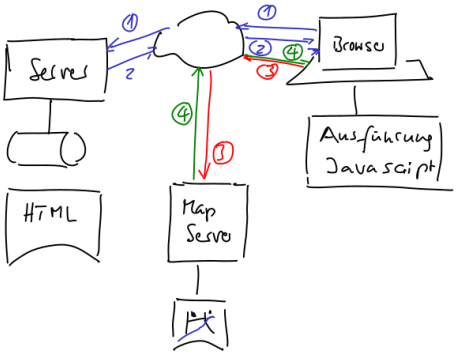


Abbildung 8.3: Datenfluss in einer Mehr-Server-Architektur

Dokumentendatenbank als Server

Die Anbindung einer Datenbank an einen Dokumentenserver (d.h. MySql mittels PHP an den Apache-Server) kann auch umgekehrt werden: eine Dokumenten-Datenbank wird als Server gebaut und liefert Dokumente und Datensätze, die als kleine Dokumente aufgefasst werden. Damit kann zumindest die Datenbank-Anbindung mit PHP

eingespart werden, was die Komplexität wohl vermindert.

Service Architektur

Generell werden zunehmend Programme als Dienste im Web konzipiert; ihre Nutzung erfolgt immer über eine HTTP Schnittstelle und das Userinterface ist als Webpage in HTML programmiert. Beispiele sind: Datenbank CouchDB, SPARQL-endpoint als Datenbank abfrage, etc. In diesen Fällen wird ein Dienst direkt aus dem Web zugreifbar gemacht ohne dass ein spezieller Server dazwischengeschaltet wird

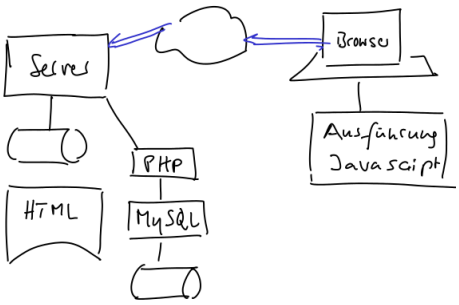


Abbildung 8.4: Die klassische und eine Datenbank als Webserver Architektur

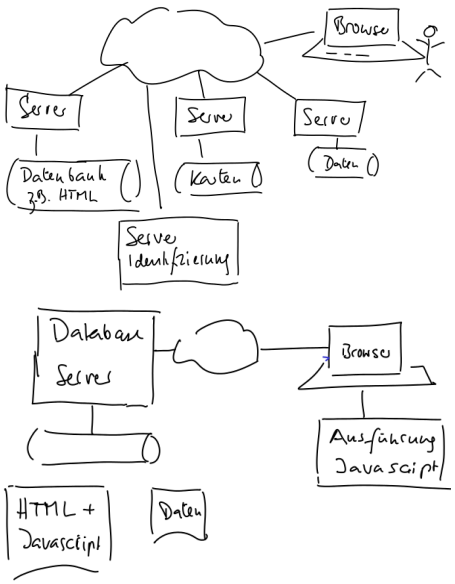


Abbildung 8.5: Viele Server, jeder für eine spezifische Funktion

Es werden mehrere Vereinfachungen erwartet:

- zum Dienstprogramm gehört keine komplexe graphische Schnittstelle sondern der überall vorhandene Browser und HTML werden zur Programmierung der Schnittstelle herangezogen.
- zu einem Dienst können mehrere Schnittstellen erstellt werden ohne dass die substantielle Programmierung verändert wird.

Wenn Dienst und Browser auf dem gleichen Rechner laufen, sind nur unwesentliche Verzögerungen durch den Wechsel des Prozesses zu erwarten.

9
NOSQL-Datenbanken für das Web

Die Kombination von Browser und Datenbank mittels einer zusätzlichen Sprache (PHP) ist offensichtlich kompliziert und ruft nach einer Verbesserung. Im Zuge verschiedener Versuche, die Beschränkungen der relationalen Datenbanken zu überwinden und wurden NOSQL Datenbanken gebaut [1]. Populäre Dienste im rasch gewachsenen WWW müssen sehr viele Anfragen verarbeiten - Google verzeichnet durchschnittlich 40,500 Anfragen pro Sekunde (3.5 Milliarden pro Tag), twitter verzeichnete ein Maximum von 140,000 Tweets pro Sekunde. Firmen wie Google, Twitter oder Facebook haben Ansprüche an Datenspeicherung, die eine relationale Datenbank nicht liefern kann. Michäl Stonebreaker ist anderer Meinung, nachzulesen iin einem regelmässigen Blog[2].

für mobile Anwendungen sind die NOSQL Datenbanken, die eine "eventually consistent" Policy (9.1.2) verfolgen, wesentlich, weil sie erlauben, dass mobile Geräte Daten erfassen und speichern, auch wenn sie nicht mit dem Internet verbunden sind. CouchDB ist ein Beispiel für eine solche Datenbank.

[1] Stefan Edlich, Achim Friedland, Jens Hampe, and Benjamin Brauer. *NoSQL: Einstieg in die Welt nichtrelationaler Web-2.0-Datenbanken.* Hanser, 2010
NOSQL steht für "not only SQL"

[2] http://cacm.acm.org/blogs/blog-cacm/50678-the-nosql-discussion-has-nothing-to-do-with-sql/fulltext

9.1 Datenbanken für das World Wide Web

Datenbanken, die die Millionen von Anfragen pro Sekunde an Google, Twitter oder Facebook, beantworten und gleichzeitig die vielen Änderungen eintragen können, sind nicht mit der Technik der relationalen Datenbanken zu schaffen. Zwei wesentliche Veränderungen:

9.1.1 Map.Reduce Verarbeitung

Eine Anfrage wird aufgeteilt, so dass der eine teil parallel auf verschiedenen Rechnern, die je nur einen teil der Daten zur Verfugung haben, erledigt wird (Map). Im zweiten Schritt (Reduce) werden diese Teilergebnisse - die viel kleiner sind, als die gesamten Daten, die im ersten schritt abgesucht wurden - zusammengeführt und abschließend

ausgewertet und dann dem Kunden zugestellt. Als Beispiel: im ersten schritt werden die in Stücke geteilten Daten je nach einer Bedingung abgesucht und im zweiten Schritt die jeweils gefundenen Ergebnisse nach einer Bewertung gereiht. Das entspricht ungefähr der Arbeitsweise beim abarbeiten einer google anfrage.

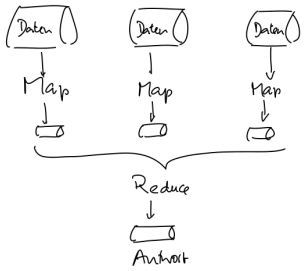


Abbildung 9.1: Map-Reduce: nach einem ersten parallel ausgeführten Map schritt werden die wenigen gefundenen Daten in einem Reduce Process zusammengeführt.

9.1.2 Eventually consistent

Relationale (traditionelle) Datenbanken verarbeiten Änderungen nach dem Transaktionskonzept. Es wird erreicht, dass nur abgeschlossene Änderungen andere gleichzeitig agierende Nutzer beeinflussen können und dass nur Änderungen, die die Datenbank in einen neuen konsistenten zustand bringen, eingebracht werden können. Wenn die Datenbank mehrfach gespeichert ist, so haben alle Kopien immer den gleichen Zustand und anfragen geben bei jeder Kopie das gleiche Ergebnis. Damit das funktioniert, müssen Sperren (locks) gesetzt werden, damit vermieden wird, dass zwei Nutzer gleichzeitig die gleichen Daten ändern wollen.

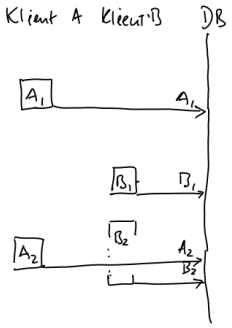


Abbildung 9.2: Die übliche Verarbeitung von Änderungen erfordert Kontakt mit der zentralen Datenbank, in die jede Änderung je einzeln eingetragen wird

manche NOSQL Datenbank geben dieses harte Konzept auf und erlauben vorübergehend unterschiedliche Zustände von Kopien von (replizierten) Datenbanken. Es wird erlaubt, dass Änderungen lokal durchgeführt werden, ohne zentrale Steuerung, und damit lokale Datenbanken unterschiedliche Daten enthalten. Sind die Datenbanken über das Netz miteinander verbunden, so werden die neu gespeicherten Versionen von Dokumenten ausgetauscht und “eventually” sind alle Datenbanken im gleichen Zustand mit den neuesten Versionen der Dokumente. Es wird riskiert, dass zwei lokale Änderungen das gleiche Dokument unterschiedlich verändern und dieser Konflikt nicht automatisch gelöst werden kann. Es wird gehofft, dass (a) solche Konfliktfälle selten vorkommen, (b) einfache Regeln aufgestellt werden können welche Änderung gewinnt oder (c) dass der Nutzer die Bereinigung interaktiv vornimmt.

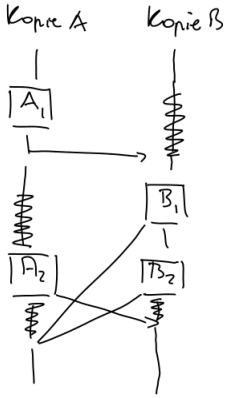


Abbildung 9.3: Zwei Nutzer haben je eine Datenbank, die sie unabhängig voneinander ändern; zeitweise sind die gespeicherten Daten nicht gleich (inkonsistent - mit einer Schlangenlinie markiert); wenn Verbidungen zwischen den Datenbanken bestehen, werden die Änderungen ausgetauscht und die Datenbanken sind gleich (konsistent).

Änderungen nach dem Prinzip “eventually consistent” sind nicht für alle Anwendungen geeignet, wohl eher nicht für Buchungen bei Banken, aber wohl für Fluggesellschaften, die ohnehin Überbuchungen zulassen und eben für Twitter, Facebook und Google - kleine Differenzen und Fehler haben keine weiteren Folgen.

9.2 CouchDB als Beispiel für eine alternative Architektur

CouchDB, eine neue NOSQL Datenbank [3] ist direkt als web Server konstruiert und bietet ein http Protokoll (GET, POST etc.) an, um Datenbank befehle entgegen zu nehmen und zu bearbeiten. das macht erstens das Programm viel schlanker, weil keine User Interface Rou-

[3] http://CouchDB.Apache.org/

tinen eingeführt werden müssen, und erlaubt auch eine wesentlich einfachere Einbindungen in Web Applikationen.

CouchDB unterscheidet sich von relationalen Datenbanken in wesentlichen punkten:

Datenmodell Die Einheit ist ein Dokument, was eine beliebig große oder kleine Menge von strukturierten Daten, die in JSON codiert sind, meint.

DocumentID Jedes Dokument hat eine ID, über die es rasch gefunden werden kann.

DocumentVersion Dokumente werden nie überschrieben, sondern immer eine neue Version des Dokumentes mit der gleichen ID aber einer höheren Versionsnummer abgelegt. (alte Versionen werden nicht automatisch gelöscht und keinesfalls einfach überschrieben).

Index Beliebige Index Dateien können angelegt werden, die alle oder Teile der Dokumente nach bestimmten Werten in effizienten Indexstrukturen erfassen und so einen raschen zugriff auf Dokumente erlauben.

Die Architektur ist anders als für die Einbindung von Datenbanken, es ist eine Erweiterung des Browsers auf der Seite des *Klienten* nötig (Abbildung 4.7). Wie Programme für den Zugang zur Datenbank wird Code in die Webseite eingefügt, aber nicht in PHP geschrieben, sondern in Javascript. Dieser Code wird mit der Seite vom Apache Server an den Browser mit geschickt und vom Browser im Client ausgeführt (Abbildung 4.7).

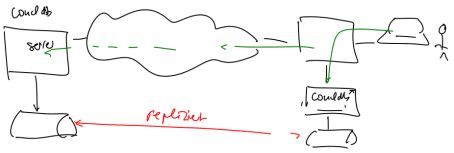


Abbildung 9.4: Architektur einer Webanwendung mit CouchDB

9.3 Räumlich Daten

Für CouchDB gibt es einen Zusatz, der räumliche Daten indiziert und den Zugriff raumbezogen erlaubt um Kartenausschnitte anzuzeigen etc.

9.4 Datenbank als eigener Server

Eine CouchDB ist ein web Service, der auf einem Port auf Anweisungen wartet. (üblich ist 5984). Das Interface ist RESTful wo? und anfragen werden als GET (Abfrage der Datenbank), POST (einbringen eines Dokumentes oder einer neuen Version eines Dokumentes), PUT und DELETE um eine neue Datenbank zu schaffen, bzw. eine zu löschen. Daten und Anfragen werden üblicherweise als JSON Datenstrukturen übergeben.

CouchDB enthält keine eigenen Methoden, um Daten auf dem Bildschirm darzustellen und keine graphischen Interfaces.

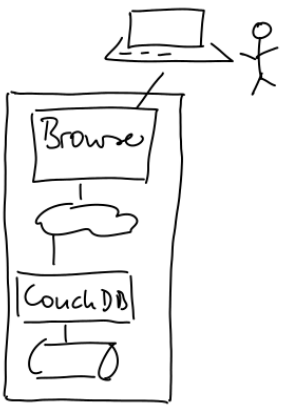


Abbildung 9.5: CouchDB benutzt den Browser als Interface.

Futon ist eine Webapplikation, die im Browser läuft [4] und mit der Datenbanken interaktiv geschaffen, befüllt und abgefragt werden können - der Form nach ist das aber eine ganz normale Web Applikation, die das gleiche REST Interface wie jede andere CouchDB Anwendung braucht.

CouchDB ist - auch wegen dieser Beschränkung auf ein einziges web Interface - sehr klein und kann auch auf winzigen Rechnern (z.B. Mobiltelefonen) laufen. das erlaubt die lokale Speicherung (Replikation) von Datenbanken, die dem Nutzer auch zur Verfugung stehen, wenn kein Zugang zum web möglich ist (Ausland, Versorgungslücke, Netzausfall) aber dennoch das “scharen” von Daten über Replikation erlaubt.

[4] mit http://127.0.0.1:5984/_utils/ aufzurufen

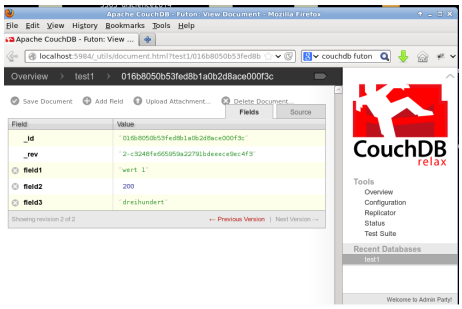


Abbildung 9.6: Das Webinterface zu einer neuen CouchDB

9.5 Applikationen in der Datenbank gespeichert

Webapplikationen sind Dokumente, im allgemeinen HTML Dokumente, die von Apache Server vom Filesystem des Servers geholt und an den web Browser des Client geschickt werden. Solche HTML Dokumente kann man aber auch als Dokumente in einer CouchDB speichern und von dort an eine web Browser schichten; die Datenbank enthält dann nicht nur die Daten, sondern auch die zugehörigen Programme, was das leidige Problem der Synchronisation von Anwendung und Daten bei Änderungen löst. die Verteilung einer neuen Version einer Applikation erfolgt durch einspeisen einer neuen Version der entsprechenden Dokumente und diese werden über Replikation an alle verbundenen Datenbanken automatisch weitergegeben - diese sind immer auf dem neusten stand, ohne dass der benutzter davon etwas merkt.

9.6 User Interface Tools

Auch für NOSQL Datenbanken sind Hilfsmittel notwendig, um die Daten aus dem Datenbank Format in eine ansprechende graphische Form zu überführen. Das geschieht mit Javascript und auf Javascript aufbauenden Hilfsmittel. Häufig verwendet wird jquery (von dem es eine Version für mobile Anwendungen gibt jMobile) und spezifisch für CouchDB Couchapp, welches die Interaktion zwischen den eingaben des Nutzers und den Aktionen der Javascript Funktionen regelt. Beachten sie, dass die eingaben des Benutzers interaktiv, d.h. asynchron zur Abarbeitung erfolgen. Das Programm muss auch auf Aktionen reagieren während eine andere Aktion noch in Verarbeitung ist.

Ein gutes Beispiel ist das Werkzeug, das das einrichten und bearbeiten einer Datenbank erlaubt. Es ist als Webapplikation geschrieben und in jeder Instanz einer CouchDB schon enthalten. Nach dem ein-

richten von CouchDB auf einem Rechner muss nur http://localhost:5984/_utils/ im web Browser aufgerufen werden. Dann kann man gleich beginnen eine Datenbank anzulegen und einzelne Dokumente abzulegen:[5]. In der Figur 9.6 ist ein Dokument zu sehen, das einmal geändert wurde (_rev ist 2-...) und das drei werte enthält.

[5] http://wiki.Apache.org/CouchDB/Getting_started_v

10
Linked Data und Semantic Web

Die Integration von Daten aus unterschiedlichen Quellen ist eine wichtiges Ziel in räumlichen Anwendungen. Während der Machbarkeitsstudie haben wir erkannt, dass der Wert für den Nutzer oft aus der Kombination von Daten entsteht, die einzeln jederzeit zur Verfügung stehen aber erst in einer intelligenten Verbindung wertvoll werden.

Sehr viele Daten sind in unterschiedlich strukturierter Form auf dem Web erhältlich. Wikipedia ist nur eine der grössten Sammlungen, neben den öffentlichen Sammlungen von geographischen Daten in Ländern uns Städten (z.B. die Stadt Wien [1].

Linked Data [2] ist eine Methode, Daten, die im Web frei verfügbar sind in einer offenen, mit andern Datenquellen verknüpfbaren Form zu veröffentlichen und damit nutzbar zu machen. Linked Data ist der wesentlichste Teil der Semantic Web Initiative des World Wide Web Consortiums. Ein neuer, deutscher Text dazu [3].

[1] https://open.wien.at/site/datenkatalog/)

[2] http://de.wikipedia.org/wiki/Linked_Open_Data

[3] Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, and York Sure. Semantic web. *Berlin, Heidelberg*, 2008

10.1 Metadaten

Um Daten, die in einer relationalen oder einer Dokumentendatenbank gespeichert sind, zu verwenden, muss man die verwendeten Datenstrukturen und die Kodierung verstehen. Um die Verwendung von Daten, die andere gesammelt hatten, zu erleichtern sollten Daten auch mit Beschreibungen ihrer Bedeutung versehen werden. Das sind die sogenannten Metadaten (von griechisch meta, über, jenseits von); diese werden nach einem standardisierten Verfahren beschrieben [dublin core] und können von Webservern bezogen werden.

Metadaten sollen zur Entscheidung, ob ein Datensatz der von einem Server bezogen werden kann, für eine bestimmte Aufgabe eingesetzt werden kann. Das funktioniert in der Praxis meist nicht, weil die Beschreibung der Daten aus der Sicht des Produzenten erfolgt und die Fragen von potentiellen Benutzern nicht ausreichend beantworten [4]; weil Praktiker aber wissen, dass Metadaten selten benutzt werden, sind sie oft nicht vollständig insbesondere nicht die besonders wichtige

[4] *Experiences with Metadata*, volume 2 of *7th Int. Symposium on Spatial Data Handling, SDH'96*, 1996. Faculty of Geodectic Engineering, Delft University of Technology

Information über die Datenqualität [5].

[5] A. T. Boin and G. J. Hunter. Facts or fiction: Consumer beliefs about spatial data quality. Spatial Science Conference 2007, 2007a; and A. T. Boin and G. J. Hunter. What communicates quality to the spatial data consumer? International Symposium on Spatial Data Quality 2007, 2007b

10.2 Linked Data

Linked data beruht auf drei einfachen Ideen, die das Internet ausnützen:

- alle Daten werden als binäre Relationen dargestellt,
- URI (genauer IRI) sind nach Konstruktion im web eindeutige identifiziert,
- Daten und Metadaten werden am gleichen Ort mit dem gleichen Verfahren abgelegt.

10.2.1 Binäre Relationen

Relationen mit zwei stellen sind ausreichend, um alle Arten von Fakten zu beschreiben. Eine binäre Relation Vater mit zwei Einträgen:

Vater (Peter, Anna).

Vater (Heinrich, Maria).

Fakten aus verschiedenen Relationen können als Triple gespeichert werden:

(Peter, Vater, Anna), (Heinrich, Vater, Maria), ... (Anna, Mutter, Maria), ...

Zu binären Relationen gibt es eine einfache und starke mathematische Theorie [6,7]: binäre Relationen können verknüpft werden (eine Vereinfachung des joins der relationalen Datenbank-Operationen).

[6] Erwin Schröder. *Vorlesungen über die Algebra der Logik (Exakte Logik).* Teubner, 1890

[7] Richard Bird and Oege de Moor. *Algebra of Programming.* Prentice Hall International Series in Computer Science. Prentice Hall Europe, 1997

10.2.2 Identifier

Im Unterschied zur relationalen Datenbank-Theorie, die auf Werte abstellt, verwenden linked data eindeutige identifizierer. In einer relational Datenbank kann es nur einen Eintrag mit einem Namen geben; zwei Personen, die zufällig den gleichen Namen tragen, sind nicht unterscheidbar und werden zu einer zusammengefasst; in der Praxis werden eindeutige Identifizierungsschlüssel benutzt (zb. die Sozialversicherungsnummer) um dieses Problem zu vermeiden - um den Preis einer Verwaltung eindeutiger Schlüssel.

RDF (Resource Descriptor Framework) nützt die Konstruktion des WWW aus, die automatisch eindeutige Identifizierer für die Ressourcen (z.b. files auf die mit dem Browser zugegriffen werden kann) zuweist. Der Beschreibung eines Objektes wird eine IRI (International Ressource Identifier, eine Generalisierung des URI) zugewiesen; in einer Relation “Label” oder “Name” kann dann der String mit dem Namen der Person gespeichert werden - zwei Personen mit dem gleichen Namen sind aber immer zwei verschiedene Datensätze.

10.2.3 Daten und Metadaten gleich gespeichert

Daten und Metadaten werden gleichartig gespeichert und darauf kann auch gleichartig zugegriffen werden; es wird keine Unterscheidung gemacht.

10.2.4 Abfragesprache

Abfragen werden in einer SQL nachempfundenen, aber eigentlich einfacheren Sprache geschrieben und über eine web Schnittstelle an Server, sogenannte SPARQL-endpoints geschickt. es gibt verschiedene Implementierungen, einige davon sind Open-source und unter verschiedenen Betriebssystemen verfügbar.

Eine Erweiterung zur abfrage geographischer Fakten ist GeoSPARQL (http://www.opengeospatial.org/Standards/geosparql)

10.2.5 Open Linked Data

Ein großer Teil von frei verfügbarer Information ist im RDF Format zugänglich (z.b. viele Daten aus der Wikipedia), darunter auch geographische Ortsnamen. Diese sind auf verschiedenen Servern zugänglich und können in Abfragen beliebig verwendet werden (die web Adressen der Server sind bereits in den IRI enthalten).

Beispiel: “Find all schools within a 5km radius around a specific location, and for each school find coffeeshops that are closer than 1km.” (from http://linkedgeodata.org/OnlineAccess/SparqlEndpoints)

```
Prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
Prefix ogc: <http://www.opengis.net/ont/geosparql#>
Prefix geom: <http://geovocab.org/geometry#>
Prefix lgdo: <http://linkedgeodata.org/ontology/>

Select ?school ?schoolLabel ?coffeeShop ?coffeeShopLabel
From <http://linkedgeodata.org> {
  ?school
    a lgdo:School ;
    rdfs:label ?schoolLabel ;
    geom:geometry [
      ogc:asWKT ?schoolGeo
    ] .

  ?coffeeShop
    a lgdo:CoffeeShop ;
    rdfs:label ?coffeeShopLabel ;
    geom:geometry [
      ogc:asWKT ?coffeeShopGeo
    ] .

  Filter (
    bif:st_intersects (?schoolGeo, bif:st_point (4.892222, 52.373056), 5) &&
    bif:st_intersects (?coffeeShopGeo, ?schoolGeo, 1)
  ) .
}
```

Diese Abfrage wird nun in eine “URL escaped” Format umgewandelt (3.9) und kann dann mit Copy-Paste in die Adresszeile des Browsers kopiert werden. Das Ergebnis ist die gewünschte Liste!

http://linkedgeodata.org/sparql?default-graph-uri=http%3A%2F%2Flinkedgeodata.org&query=Pre

school	schoolLabel	coffeeShop	coffeeShopLabel
http://linkedgeodata.org/triplify/node848763411	"auf dem Brinke - music school"	http://linkedgeodata.org/triplify/node1237571316	"Star Bucks"
http://linkedgeodata.org/triplify/node249329160	"Berlage Lyceum"	http://linkedgeodata.org/triplify/node1237571316	"Star Bucks"
http://linkedgeodata.org/triplify/node237812173	"4e montessorischool"	http://linkedgeodata.org/triplify/node1237571316	"Star Bucks"
http://linkedgeodata.org/triplify/node409609569	"Maas en Waalschool"	http://linkedgeodata.org/triplify/node1237571316	"Star Bucks"
http://linkedgeodata.org/triplify/node158700775	"Theo Thijssen"	http://linkedgeodata.org/triplify/node254039743	"La Canna"
http://linkedgeodata.org/triplify/node158700391	"Theo Thijssen"	http://linkedgeodata.org/triplify/node254039743	"La Canna"
http://linkedgeodata.org/triplify/node158700775	"Theo Thijssen"	http://linkedgeodata.org/triplify/node254039746	"Grasshopper"
http://linkedgeodata.org/triplify/node409609569	"Maas en Waalschool"	http://linkedgeodata.org/triplify/node469234383	"Kaffee Schmole"
http://linkedgeodata.org/triplify/node237812173	"4e montessorischool"	http://linkedgeodata.org/triplify/node469234383	"Kaffee Schmole"
http://linkedgeodata.org/triplify/node249329160	"Berlage Lyceum"	http://linkedgeodata.org/triplify/node469234383	"Kaffee Schmole"
http://linkedgeodata.org/triplify/node848763411	"auf dem Brinke - music school"	http://linkedgeodata.org/triplify/node469234383	"Kaffee Schmole"
http://linkedgeodata.org/triplify/node31240605	"Christelijke Basisschool Dr. J.T.H. de Visser"	http://linkedgeodata.org/triplify/node593725286	"Hornig Kaffee"
http://linkedgeodata.org/triplify/node679091166	"Winterkoninkje"	http://linkedgeodata.org/triplify/node593725286	"Hornig Kaffee"
http://linkedgeodata.org/triplify/node1794924147	"OBS Corantijn"	http://linkedgeodata.org/triplify/node593725286	"Hornig Kaffee"
http://linkedgeodata.org/triplify/node1432277900	"De Roos"	http://linkedgeodata.org/triplify/node593725286	"Hornig Kaffee"
http://linkedgeodata.org/triplify/node1343137836	"Joop Westerweelschool"	http://linkedgeodata.org/triplify/node593725286	"Hornig Kaffee"
http://linkedgeodata.org/triplify/node1343137837	"As-Siddieq"	http://linkedgeodata.org/triplify/node593725286	"Hornig Kaffee"

Abbildung 10.1: Das Ergebnis der Abfrage

Teil V

Schluss

11
Datenaustausch: Sicherheit der Daten

In einer Web Applikation werden Daten mit den Nutzern ausgetauscht. Dabei stellen sich drei Arten von Fragen:

- Bedeutung: der Empfänger soll die Daten richtig, d.h. mit der Bedeutung, die ihnen der Sender gibt, versteht.
- Berechtigung: nur berechtigte Personen dürfen die Daten erhalten und allenfalls verändern.
- Zugang: Daten müssen vor Zugriff, Veränderung oder Zerstörung durch andere geschützt werden und den berechtigten Nutzern der Zugang möglichst einfach gemacht werden.

Die Sicherstellung der korrekten Übertragung von Bedeutung, das heißt, richtige Interpretation der Daten beim Empfänger, muss bereits beim Entwurf des Graphical User Interfaces angelegt werden. Es muss darauf geachtet werden, dass die Applikation in einer einheitlichen Terminologie präsentiert wird; begriffe in allen Dokumenten, die Nutzer sehen, aber auch in allen Ausgaben auf dem Schirm, müssen einheitlich sein[1]. Menschliche Benutzer sind gewohnt, sich an abweichende Terminologien in Konversationen anzupassen, erwarten aber, dass der Partner die Bedeutung von Begriffen konsistent verwendet.

[1] Inc. Apple Computer. *Human Interface Guidelines: The Apple Desktop Interface.* Addison-Wesley, 1987

Schwierig ist es, sicherzustellen, dass die Bedeutung, d.h. die Klassifizierung von Objekten in Daten aus verschiedenen Quellen, vergleichbar sind. Landesgrenzen sind auf Karten oft leicht auszumachen, weil die Darstellungen der Objekte auf der Karte unterschiedlich sind. Ähnliche Unterschiede finden sich zwischen Datensammlungen verschiedener Länder.

In diesem Kapitel werden zuerst die Gefahren analysiert und dann übergreifende Fragen angegangen, wie z.b. die Sicherheit von Programmen und die grundsätzliche wichtige Verschlüsselungstechnik. Danach werden die beiden wichtigen Themen Berechtigung und Zugang besprochen.

11.1 Gefahren

Mit dem Anschluss ans Internet wird im Prinzip der Rechner für jeden von außen zugänglich; das wird leider ausgenützt, einerseits um „Geheimnisse“ auszuspionieren und andererseits den Rechner zu missbrauchen. Es ist deshalb notwendig — auch und vielleicht besonders in einer Universität — den Internetzugang zu beschränken. Die Sicherheitseinstellungen sind zahlreich, nicht leicht zu durchschauen und nicht immer optimal zusammenwirkend. Oft wird versucht durch Kompliziertheit und Leichtsinnigkeit Sicherheit zu erleichtern („Security by Obfuscation“) — das scheint mir nicht zielführend.

Wenn Programme, die mit einer web Seite geladen werden, auf einem Rechner ablaufen, wird vieles möglich, was bei einer reinen Darstellung von Text der vom Browser geladen wird, nicht passieren kann. Web Seiten von unbekanntem Ursprung - z.b. durch unterschieben einer falschen Web-Adresse (5) - werden geladen und der darin eingebettete Kode ausgeführt.

Es wird immer wieder versucht, durch Fehler in der Browser Implementierung, schad-kode beim Benutzer ablaufen zu lassen, der dann z.b. den Rechner unter die Kontrolle des Schädigers bringt (sogenannte bot Netze), um ihn später zu missbrauchen.

Den gefahren wird begegnet durch:

Sandbox - der Javacode läuft in einer beschränkten Umgebung im Browser und hat nur die Möglichkeit web-bezogene Aktivitäten anzustoßen, nicht allgemeine Befehle beim lokalen Rechner laufen zu lassen (also z.b. nicht files zu löschen).

“same origin policy” - der Code hat nur Zugang zu Daten, die mit der web Adresse des Senders des Codes verbunden sind (kann also nicht die Cookies eines anderen Dienstes ausspionieren).

Diese Vorkehrungen bannen leider nicht alle gefahren und die starke Verbreitung von Browser und Betriebssystem Kombinationen (besonders IE und Windows) erlauben Fehler auszunützen, die sich dann “Viren-erkrankungsartig” weiter verbreiten. Schwierig zu verstehen ist, dass Code auf einem Rechner ausgeführt werden kann, der nicht zuverlässig ist und dann schaden beim Sender anrichtet. Windows Betriebssysteme erlauben auch, Javacode außerhalb der Sandbox ausführen zu lassen, was potentiell beliebige Schaden anrichten kann.

11.2 Sicherheit der Programme

Linux und Apache gelten als sehr sicher und werden automatisch mit Sicherheitsupdates versorgt. Es ist kein Fall einer Sicherheitslücke, die virusartig ausgenützt worden ist, bekannt. Den eigenen Rechner, oder den aufgesetzten Server, zu schützen, erfordert aber etwas Vorsicht,

Überlegung und Genauigkeit bei der Durchführung — besonders bei einem MS Windows Rechner, der unbedingt einen aktuellen Virenschutz braucht.

Beobachtungen zeigen, dass Rechner, die im Web sichtbar sind, innerhalb kürzester Zeit zum Ziel von Angriffen werden. Im allgemeinen reichen aber Passwörter, besser Zertifikate zur Abwehr aus. Schad-Kode versteckt in HTML Seiten kann gefährlicher sein und schwieriger zu erkennen.

ich habe Angriffe innerhalb von Stunden erlebt!

Programmen sollte man nur von Quellen, denen man vertrauen kann, laden. Es wird im allgemeinen auch ein Hashwert angegeben, mit dem man überprüfen kann, dass die Kopie, die bei mir angekommen ist genau gleich ist wie die ursprünglich im web zugänglich gemachte.

11.3 Grundlagen der Verschlüsselung

Damit Daten - Passwörter und andere Daten, die nicht öffentlich gemacht werden sollen - nicht abgehört werden können, werden sie verschlüsselt. Die gleichen kryptographischen Verfahren werden auch benutzt, um Verfälschungen der Daten zu verhindern. Zwei verschiedene Verfahren werden eingesetzt:

11.3.1 symmetrische Verschlüsselung

Sender und Empfänger verwenden den gleichen Schlüssel (oder leicht auseinander abgeleitete Schlüssel) für die Ver- und Entschlüsselung. Diese Verfahren sind rasch und sofern genügend lange Schlüssel, und sichere verfahren angewendet werden, nach dem heutigen stand des Wissens sicher (und bleiben es noch eine weile). Voraussetzung ist, dass der Schlüssel geheim ist - was die frage aufwirft, wie man den Schlüssel vom Sender zum Empfänger überträgt (z.b. auf dem Postweg, als SMS...); das geschieht meist mit public-key methder der Verschlüsselung.

11.3.2 asymmetrische (Public key) Verschlüsselung

Bei diesen Verfahren produziert jeder Teilnehmer zwei Schlüssel: einen öffentlichen, den er publiziert (z.b. auf einem spezialisierten public key Server ...) und einen privaten, den er geheim hält. Geheime Nachrichten, die man dem Empfänger senden will werden mit dem publizierten öffentlichen Schlüssel verschlüsselt und vom empfänger mit seinem geheimen privaten Schlüssel entziffert. Nach der Verschlüsselung mit dem öffentlichen Schlüssel des Empfängers kann weder der Sender noch irgendjemand anderes den verschlüsselten Text lesen - nur der Empfänger kann ihn mit seinem Schlüssel entziffern.

11.4 Authentifizierung

Vertrauliche Daten dürfen bestimmten Personen zugänglich gemacht werden; Authentifizierung stellt sicher, dass die Kommunikation mit einer bestimmten Person erfolgt, deren Berechtigung unüberprüft worden ist. Die Überprüfung geschieht meist durch Passwort oder besser durch publik key Verschlüsselung:

11.4.1 Authentifizierung der Benutzer durch Passwort

Das Passwort ist die verbreitetste aber wenig sichere Methode ungebetene Gäste auszuschreiben. Passworte müssen in anderen Programmen eingetragen werden, wo sie für andere sichtbar sind, oder werden einfach auf Zettel beim Bildschirm aufgeschrieben, damit sie nicht vergessen werden. Im übrigen sind 6-stellige Passwörter mit heutigen PCs leicht zu finden und längere mit etwas Aufwand und besseren Rechnern auch knackbar. Beim Entwickeln der Anwendung werden verschiedene Teile mit Passwörtern geschützt werden müssen; ohne genaue Listen, welche Passwörter wo verwendet wurden, führt das zu frustrierendem Suchen. Ein Tagebuch über die Entscheidung der Anwendung, in dem Passwörter und andere relevanten Details gesammelt werden, bewährt sich!

11.4.2 Authentifizierung mittels Publik key Verschlüsselung

wenn ich von einem Sender ein Dokument erhalte das ich nur mit seinem öffentlichen Schlüssel entziffern kann, so weiß ich, dass dieser das Dokument mit seinem geheimen, privaten Schlüssel verschlüsselt hat. Damit kann man Authentifizierungsverfahren aufbauen, bei denen einer dem andern ein Dokument schickt und dieser es verschlüsselt zurückschickt; aktuelle Verfahren sind noch etwas komplizierter, weil man auch ausschließen muss, das die Kommunikation in der Mitte durch einen Dritten abgehört und verfälscht wird (sogenannte “man in the middle attack”).

11.5 Berechtigungen

Nicht alle Daten in einer Applikation beschreiben offen sichtbare Fakten oder Fakten, die allgemein bekannt sind und deshalb keines besonderen Schutzes bedürfen. Viele Fakten werden von öffentlichen Stellen für bestimmte Aufgaben gesammelt und müssen diesen mitgeteilt werden (z.b. Steueramt). Diese Daten dürfen nur für die bestimmten Aufgaben verwendet werden und dürfen nicht weitergegeben werden oder von den sammelnden Stellen veröffentlicht werden.

Andere Daten sind von einer Organisation mit Kosten gesammelt

und aufbereitet worden und sollen nur von Personen eingesehen werden, die den für die Benutzung der Daten festgelegten Preis bezahlt haben.

Um den Personenkreis, der Zugang zu den Daten haben soll einzuschränken, muss eine Verbindung zwischen den natürlichen, realen Personen und den Prozessen, die für diese Personen agieren, eine Verbindung hergestellt werden. Unter berechtigter Person ist hier allenfalls auch eine Organisation mit mehreren Personen zu verstehen, die als ganzes Berechtigt ist und die Verteilung der Berechtigungen an ihre Mitglieder selber prüfen muss.

Es muss erstens dem Prozess, der im Auftrag für eine Person operiert eine Markierung mit der Identität der Person, für den der Prozess ausgeführt wird, zugeordnet werden. Dann kann zweitens für jeden Zugriff dieses Prozesses auf Daten geprüft werden, ob für diese Person (d.h. für die Marke) der Zugriff für diese Daten erlaubt ist.

11.5.1 Identifikation von Personen oder Organisationen

Für jede Person, der Rechte eingeräumt werden sollen, muss eine eindeutige Bezeichnung gewählt werden; diese wird oft als "Benutzername" ("User Name") beschrieben und wird von der Person gewählt. Das produziert die Schwierigkeit, dass das System prüfen muss, ob der vom Nutzer gewählte Namen nicht schon vergeben ist (dh. nicht eindeutig wäre); in diesem Fall muss vom Benutzer ein anderer Name gewählt werden - was dazu führt, das ein Benutzer bei verschiedenen Systemen verschiedene Benutzernamen braucht und sich diese merken muss.

Einfacher ist die Lösung, dass der Benutzername die Email Adresse des Nutzers ist; diese ist automatisch, durch die Konstruktion des Email Systems eindeutig.

Es ist mit diesen Methoden nicht möglich zu verhindern, dass eine Person mehr als eine "Identität" hat oder dass eine Gruppe von Personen mit nur einer Identität auftritt. Beides kann in bestimmten Fällen nützlich und erwünscht sein; will man es verhindern, sind wesentlich aufwendigere Methoden notwendig (z.b. Bestätigung des namens, Geburtsdatum und Geburtsort durch eine Behörde....) oder ein Vertrauensnetz aufgebaut werden.

Network of Trust

11.5.2 Überprüfung der Identität der Person für einen Prozess

Wird ein Prozess zum Zugriff auf Daten gestartet muss diesem Prozess eine Person (genauer eine Identifikator einer Person) zugeordnet werden. das geschieht durch Abfrage des Benutzernamens.

Es muss zweitens sichergestellt werden, dass es sich wirklich um die betreffende Person handelt und nicht um einen unberechtigten, der

sich als diese Person ausgibt, indem er deren Benutzername benutzt. Da Benutzernamen nicht geheim sind wird ein zusätzliches wissen, im allgemeinen Passwort genannt, abgefragt, das der Benutzer geheim halten muss.

Hier entstehen mehrere Probleme:

- Passwörter sind leicht weiterzugeben, absichtlich oder unabsichtlich (zb. als Post-It Notiz am Bildschirm), auch weil die verschiedenen Passwörter, die für jedes System unterschiedlich sein sollen schwer zu merken sind.
- Passwörter, die offen von der Applikation zum Server übertragen werden, können abgehört werden.
- Passwörter werden leicht weitergegeben. Die Rechte sind mit einer UserID verbunden, nicht direkt mit der Person. Wenn die Person ihre UserID und das zugehörige Passwort andern mitteilt, so können auch diese die Rechte ausnützen.

Sicherer sind “credentials” oder “certificates”, welche für berechtigte Besucher ausgestellt werden und mit denen diese sich anmelden - ohne Benutzername oder Passwort eingeben zu müssen, nur das certificate muss auf dem Computer gespeichert sein (z.b. als Cookie). Es werden kryptographische Methoden (siehe ..) eingesetzt und die Credentials werden nicht direkt ausgetauscht, sondern nur verwendet - so dass auch eine abgehörte Verbindung die notwendige Information nicht enthält.

11.5.3 Netz von Vertrauen

Die Zuordnung einer “web Existenz”, die im wesentlichen aus einer Benutzeridentifizierung besteht und einer natürlichen Person, kann nicht rein im IT Bereich erledigt werden, sondern die Verbindung zwischen realer Welt und IT Welt muss explizit hergestellt werden. Das geschieht nach zwei verschiedenen Methoden:

Zertifizierung Gewissen Stellen im web wird vertraut (z.B. VeriSign, die auch zentrale aufgaben in der Verwaltung des Internets übernehmen) und diese zertifizieren andere, denen sie vertrauen und erlauben ihnen, ihre Zertifizierungsrechte weiter zu übertragen. Zertifizierungsstellen bieten Dienste an, die es erlauben, Zertifikate, die ein dienst vorweist, zu überprüfen und die meisten Browser enthalten listen von zertifikaten, denen vertraut wird (und manchmal auch listen von bekannten ungültigen oder gefälschten Zertifikaten). Zertifikate haben eine bestimmte Gültigkeitsdauer und es ist möglich, dass Institutionen sich selber Zertifikate ausstellen (so eigenartigerweise die TU), denen man dann vertrauen muss, da keine weiter Überprüfung möglich ist.

Netzwerk Nutzer zertifizieren andere ihnen persönlich bekannte Nutzer und bei der Überprüfung eines Zertifikates wird sichtbar, wer diesen Nutzer bestätigt hat. Die Bestätigung eines Nutzers (dh. der mit ihm verbundenen Schlüssel in einem public key System und einer oder mehrere Email Adressen) erfolg in der Regel nur nach Vorlage und Prüfung von amtlichen Dokumenten.

11.5.4 OpenID

OpenID ist ein weiterverbreitetes Verfahren, um die Identität eines Benutzers zu bestimmen, das die üblichen, sehr lästigen Methoden mit Benutzername und Password vermeidet. Ein Benutzer kann sich mit einer Identität, die er bei einem weit verbreiteten Dienst (Google, Facebook, Github etc.) hat, anmelden und dieser Dienst bestätigt die Identität des Nutzers. Der Nutzer kann dabei den zu verwendenden Dienst frei wählen - er kann auch einen eigenen Server unterhalten, der die Identität bestätigt - es muss ja nur ausgeschlossen werden, dass ein anderer sich mit der gleichen Identität anmeldet. Der nutzende Server erhält die Zugangsdaten beim bestätigenden Dienst nicht, sondern erhält nur eine kryptographisch gesicherte Bestätigung, über die Identität des Benutzers.

11.5.5 OAuth

OAuth bietet einer Applikation an, dass ein anderer Provider die Authorisierung eines Nutzers bestätigt. Beim Anbieter des OAuth Dienstes hinterlegt der Ersteller einer Applikation eine Liste der zugelassenen Nutzer und dier OAuth Dienst bestätigt, dass es sich um einen dieser Benutzer handelt; der Benutzer wird bei der Anmeldung auf den OAuth Dienst umgeleitet und muss sich dort mit seinen üblichen Nutzerdaten anmelden. Google bietet z.b. einen OAuth Dienst an.

11.6 Sicherheit bei der Übertragung von Daten

Das wichtigste Verfahren zur Erreichen von Sicherheit im Web ist die Verschlüsselung. Im Internet verkehren alle Datenpakete im gleichen Medium und können (mehr oder weniger problemlos) bei allen Routern gelesen werden. Dagegen hilft die Verschlüsselung des Inhaltes, so dass der Inhalt nur vom Adressaten, nicht aber von einem Dritten gelesen werden kann. Die Verschlüsselung hilft aber auch, um sicherzustellen, dass ein Text von einer bestimmten Quelle (z.B. eine Person, Behörde oder Firma) stammt.

11.6.1 Verschlüsselung im Email Verkehr

Das Public Key Verfahren kann zur Identifizierung oder Zertifizierung eines Textes durch einen Absender verwendet werden: einen Text, den ich mir eindeutig zuordnen lassen will (auf deutsch, den ich unterschreibe) verschlüssele ich mit meinem privaten geheimen Schlüssel. jeder andere kann ihn dann mit meinem publizierten öffentlichen Schlüssel entziffern und ist dann sicher, dass der Text von mir ist, denn niemand anderes kann einen Text produzieren, der mit meinem öffentlichen Schlüssel entschlüsselt werden kann.

Dieses verfahren ist in Mail Programmen standardisiert eingebaut und wird leider zu wenig genutzt [ref]; damit lassen sich Mails signieren, so dass ein Emfänger sicher weiß, von wem die email stammt und dass der Text nicht verändert wurde. email kann auch automatisch mit dem bekannten Schlüssel des Empfängers verschlüsselt werden, so dass nur er den Inhalt lesen kann. Den abhörenden Stellen bleibt dann nur die Information über den Austausch einer Email zwischen Sender und Empfänger, der Inhalt bleibt geheim - auch Experten nehmen an, dass richtig eingesetzt, diese Verfahren nicht geknackt werden können (OpenPGP, S/MIME).

Methoden der asymmetrischen Verschlüsselung beruhen auf “trapdoor functions” (deutsch etwa “Falltürfunktionen” oder Einwegfunktionen), das sind Funktionen, die leicht zu berechnen aber praktisch unmöglich zu invertieren. Die Theorie ist schwierig; als Beispiel mag dienen: die Multiplikation zweier Primzahlen ist einfach, die Zerlegung einer großen Zahl in Primfaktoren hingegen aufwendig, d.h. nicht in nützlicher zeit zu schaffen.

Metaphorisch kann man sich ein Publik-Key Verschlüsselungssystem wie einen Briefkasten vorstellen: jeder kann einen Brief einwerfen aber herausholen kann ihn nur, wer den Schlüssel hat.

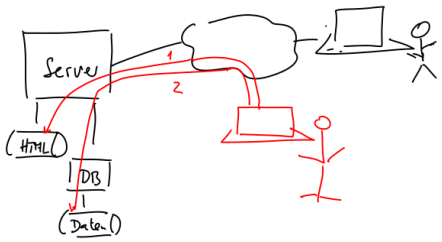


Abbildung 11.1: Angriff von aussen auf den Webserver

11.7 Beschränkung des Zuganges von außen

Nicht alle Prozesse, die auf einem Rechner laufen, müssen von außen zugänglich sein. Zum Beispiel ist es erforderlich, dass Apache zugänglich ist, nicht aber die RDB. Diese wird ja nur lokal vom Apache Server über PHP und SQL angesprochen, nicht von entfernten Nutzern. In diesem Fall wird

- der Port der RDB von außen geschlossen (Achtung RDB Hilfsprogramme funktionieren dann nicht mehr über das Web)

- die RDB nimmt nur Anfragen von lokalen Systemen entgegen, nicht aber von außen (das lässt sich auch auf bestimmte Rechner, d.h. IP Adressbereiche, beschränken).

11.7.1 Sicherung von Verbindungen

Es ist möglich, dass ein Dritter die Verbindung abhört und insbesondere Passwörter und andere geheime Codes mitbekommt und dann diese nutzt. Besonders gefärdet sind Identifizierungsmethoden, die bei Finanztransaktionen eingesetzt werden. Beispielsweise werden die Kreditkarten-Daten, die im Web häufig für die Bezahlung von Dienstleistungen verwendet werden müssen, nur in gesicherten Webverbindungen abgefragt und übertragen (HTTPS Protokoll).

Es ist oft notwendig, einen Rechner über das Netz fernzusteuern; das ist potentiell gefährlich, insbesondere weil beim Login Benutzername und Passwort ungeschützt übertragen werden und also von andern mitgehört werden könnten. Sicherer ist die gegenseitige Identifizierung durch Public key cryptography. Das gesicherte SSH und SSH2 Protokoll für den Zugang zu einem andern Rechner erreichen dies.

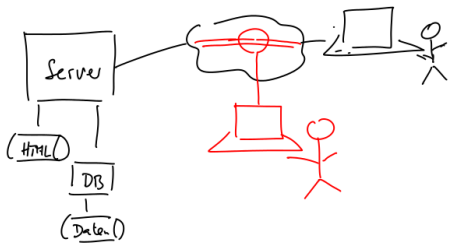


Abbildung 11.2: Ein Dritter belauscht die Verbindung und merkt sich geheime Codes

11.7.2 Sicherung des richtigen Partners

Kriminelle versuchen oft, Nutzer mit Mails zu überzeugen, eine Seite aufzurufen, die scheinbar Vertrauenswürdig ist (z.b. eine Regierungsstelle, der Telekommunikationsunternehmen oder eine Bank) und leiten dann die Verbindung zu ihnen um. Das kann durch die verwendung von alternativen Zeichen in URL (IRI) sein, kann auber auch durch Manipulation einer Domain Name Table bei einem Provider erreicht werden.

Bei einer ersten Verbindung wird der public key übertragen und bei jeder späteren Verbindung wird damit geprüft, ob die Aufnahme der Verbindung durch den autorisierten Nutzer erfolgt.

Besondere Aufmerksamkeit braucht es, um Angriffe abzuwehren, bei denen ein Dritter zwischen den Browser und den angesprochenen Server tritt und den Verkehr zum und vom Server durchleitet und allenfalls verändert. Solche Angriffe beginnen oft mit falschen Einträgen in Domain Name Servern.

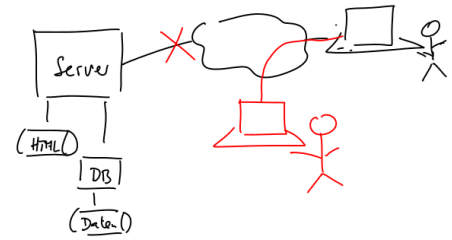


Abbildung 11.3: Umgeleitete Verbindung

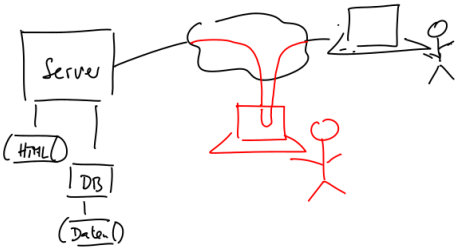


Abbildung 11.4: Man in the middle attack

11.7.3 Public Key Certificates

In eine Certificate wird ein public key mit einem Name - z.b. einer Firma oder einer Person - verknüpft und die Verknüpfung wiederum mit dem private key einer “trusted authority” verschlüsselt. Dann kann jeder empfänger des Certificates mittels des public key der “trusted authority” prüfen, ob das Zertifikat für die richtige Person steht.

Es wird dabei eine Kette von “Trust” erstellt, ich vertraue der Stelle, die das Zertifikat ausgestellt hat - oder ich überprüfe deren public key mittels ihres Zertifikates, das von einer höheren vertraubaren Stelle ausgestellt worden ist. Die oberste Stelle ist oft Verisign, eine Firma,

die auch mit der Kontrolle der Internet top level domains ausübt (.com und .net) und die obersten Domain Name Servers (sogenannte Root Servers).

Solche Zertifikate verden bei HTTPS Verbindungen geprüft und die Browser zeigen an, dass die Verbindung geprüft und gesichert ist. Zertifikate sind für eine bestimmte Zeit gültig und können auch zurückgezogen werden. In den meisten Browsern ist eine Reihe von "trusted" Zertifikaten eingebaut - und es hat sich immer wieder herausgestellt, dass manche von ihnen gefälscht oder von unbefugten verwendet worden sind.

11.7.4 Veränderungen und Zugriffe protokollieren

Wenn sensitive Daten, das sind persönliche Daten, besonders aus z.b. dem medizinischen Bereich, andern zur Verfugung gestellt werden, so muss sichergestellt werden, dass die Daten nur von befugten Personen und zu den vorgesehenen Zwecken benutzt werden. Im Zweifelsfall ist zu protokollieren, welcher Nutzer welche Daten gelesen und welche Daten verändert wurden.

11.7.5 Verhindern des Zugangs durch Absichtliche Überlastung (Denial of service)

Wenn man durch Sicherheitsmaßnahmen an einem Angriff in einen Server hinein gehindert wird, ist es immer noch möglich, die Arbeit des Servers zu behindern, so dass er die eigentlichen Kunden nicht bedienen kann. Das wird erreicht, indem man den Server durch eine sehr große Zahl von anfragen (oft nur triviale Ping anfragen) dermaßen überlastet, dass er kaum noch berechtigte Nutzer bedienen kann. Die große Zahl von anfragen wird durch die Gleichschaltung von einer großen Zahl von Rechnern nichtsahnender Nutzer erreicht, die vorher unter die Kontrolle eines sogenannten Bot-netzes geraten sind. Das ist illegal aber mit den komplexen Rechtsfragen im globan web scheinbar nicht zu verhindern [2].

[2] http://en.wikipedia.org/wiki/Denial_of_service

Eine neue Variante dieser Störung sind versuche, eine andere Ressource auszuschöpfen. Denial of service schöpft die Möglichkeit des Servers auf anfragen ablehnend zu reagieren aus; alternativ können parallel zu verarbeitende Threads, von denen nur eine beschränkte zahl zur Verfugung steht (typisch 1000), durch unkooperatives verhalten blockiert werden.

12
Rechtsfragen

12.0.6 Copyright

Wer Karten in eine Webanwendung einbaut, muss immer rechtliche Aspekte berücksichtigen. Karten — wie Bilder, Fotos — sind durch Copyright geschützt und dürfen nicht einfach verwendet werden, sondern es ist eine Erlaubnis des Inhabers der Rechte zu erlangen (mit wenigen Ausnahmen), das ist insbesondere bei Web Applikationen wichtig, weil die Resultate einer unbeschränkten Anzahl Personen zugänglich gemacht werden.

Die Verwendung von Karten der nationalen Verwaltungsbehörden für Webapplikationen, besonders wenn diese kommerzielle Ziele haben, erfordern einen — meist langwierigen Bewilligungsprozess. Dienste, wie Google Maps enthalten detaillierte, aber akzeptable, Nutzungsbedingungen. Ist die angebotene Webapplikation gratis, so ist auch die Verwendung von Google Maps frei, kommerzielle Dienste erfordern eine Lizenz, die zur Zeit $400 kostet. Es ist selbstverständlich, dass die Copyright Angaben in den Bildern nicht verdeckt werden dürfen und Google möchte keine Nutzung im anrüchigen Umfeld (z.B. Werbung für illegale Drogenverkaufsstellen). Eine Lizenz wird von Google automatisch abgegeben und gilt für eine Web Adresse und alle darunterliegenden Seiten. Die Lizenz wird auf einer Webseite beantragt; als Ergebnis erhält man eine scheinbar zufällige Zeichenkette, die im Programm einzukopieren ist. Sie muss für die Web Adresse gelten, auf der die Web Seiten (HTML) eingestellt sind.

Eine raumbezogene Webapplikation muss fast immer dem Benutzer den Ort der Handlung in einer Karte oder einem Satellitenbild zeigen. Am einfachsten ist es einen Dienst, wie Google Maps, Google Earth oder die entsprechenden Dienste von Microsoft wie Virtual Earth zu verwenden; im Folgenden verwende ich die Google Dienste; in Zukunft werden wohl die offenen Dienste, wie Open Street Map bevorzugt werden. Die Verwendung von solchen Diensten ist technisch einfach, präsentiert dem Benutzer ein bekanntes Erscheinungsbild, sind welt-

weit fast in der gleichen Qualität vorhanden und erlauben auch die kommerzielle Nutzung.

Gosse Vorsicht ist bei der Verwendung von Fotos auf Webseiten nötig. Bilder, die man selber gemacht hat oder bei denen man die Rechte von einem Freund erhalten hat (aber das bitte schriftlich bestätigen lassen), sind problemlos, sofern sie nicht andere Personen wiedererkennbar und als mehr als Staffage zeigen (es ist ja schließlich nicht möglich, den Stephansdom in Wien zu fotografieren, ohne dass zwangsläufig einige Passanten im Bild sind); abgebildete Personen haben ein "recht am eigenen Bild", dass sie klage weise (teuer!) einfordern können. Photos, die man auf dem web findet, darf man nicht einfach verwenden - das führt zu kostspieligen Briefen von den Anwälten, welche die Rechte der Bildautoren vertreten.

12.0.7 Andere Rechte

Verschiedene andere Rechtsregeln schützen Inhalte und Geschäftsideen; im allgemeinen gilt, das was nicht anständiges verhalten ist, auch nicht zulässig ist. Die Details des Schutzes von Datensammlungen (ein Sui generis Schutz), von Patent- und Markenrecht etc. sind sehr komplex. Das Buch von Christian Twaroch [ref]

Abbildung 12.1: Bild

fasst das für geographische Anwendungen, besonders web Anwendungen, zusammen.

12.0.8 Konsumentenschutz

Verschiedene Regeln beschränken die Art des Abschlusses von Verträgen im Web und räumen den Konsumenten besondere Rechte, insbesondere das Recht von einem Vertrag zurückzutreten ein. In diesem zusammenhang wird auch verlangt, dass für geschäftliche Angebote im Web eine Angabe über die Rechtsperson, mit der ein Vertrag zustande kommt, erfolgen muss (Impressum).

12.1 Klare Anforderungen

Eine klare und detaillierte Abklärung der Anforderungen an das Programm ist wesentlich. Es lohnt sich, die Machbarkeitsstudie genau zu machen und sie nachher beim Projektstart nochmals zu überarbeiten (2.1.11). Eine sichere Methode, dass ein Projekt kein Ziel erreicht, unendliche Diskussionen und oft gerichtliche Auseinandersetzungen produziert und für alle beteiligten ein riesiger Verlust wird, ist unklare und im Laufe des Projektes ändernde Zielvorgaben.[1]

[1] Scheinbar war das auch ein wesentlicher Teil der Ursache, dass das Projekt Obamacare zu einem Misserfolg wurde: http://althouse.blogspot.co.at/2013/11/this-is-post-where-i-paraphrase-10.html#more

12.2 Implementierungs Strategie

Die wichtigste Strategie für die Entwicklung und das Fehler beheben ist “divide et imperare” - “teile und herrsche”. Die Komplexität einer Webanwendung ist groß, auch wenn jeder Teil einzeln relativ einfach ist; wenn es nicht läuft, kann das verschiedene Ursachen haben - oft mehrere gleichzeitig. Diese werden nur gefunden und korrigiert, wenn wir Teil für Teil prüfen und dann stückchenweise zusammenbauen. Komponenten prüft man, indem man das einfachste Programm (oft ein Programm das “Hello world!” ausgibt) Ausfuhren lässt. Das gibt Gewissheit, dass ein Service zumindest im Grundsatz funktioniert.

Für einen Test ist es oft am einfachsten, wenn Server und Client auf dem gleichen Rechner laufen. Ein Dienst auf dem lokalen Rechner wird vom Browser mit „local host“ als Web Adresse angesprochen. Wenn alles läuft, werden die Seiten von lokalem Web Directory (z. B. /var/www) auf eine anderen Rechner, der im Web eingebunden ist, übertragen.

12.3 Die Entwicklung des Web

In diesem Kurs ist indirekt die Entwicklung des Internet und des Web nachgezeichnet: die technischen Lösungen, die heute verwendet werden bauen auf Entscheidungen auf, die seit der Mitte des 20. Jahrhunderts getroffen wurden. Die Folgen vieler dieser Entscheidungen sind heute noch wirksam, auch wenn vieles Vergessen ist und auch vergessen bleiben darf. Welche Entscheidungen wirken noch heute? und warum?

Ein rationaler Optimist wird annehmen, dass sich die richtigen Entscheidungen durchgesetzt haben und dass unglückliche Entscheidungen korrigiert worden sind. Das stimmt wohl auch zu einem guten Teil: die Gemeinschaft der am Web beteiligten hat gelernt und vorallem eine Abstimmung mit den Füssen hat geholfen, dass unpraktisches verschwunden ist. Gleich wie in der Wissenschaft, die sich nicht dadurch weiterentwickelt, dass Anhänger falsche Auffassungen vom Gegenteil überzeugt würden, sondern dadurch, dass Denkschulen mit falschen Ideen weniger Nachfolger haben und mit der Zeit aussterben[2]. Unpraktische und teuere Lösungen verschwinden schließlich, weil sie niemand kaufen und anwenden will. Die Liste der verschwundenen Ideen, denen wohl niemand eine Träne nachweint, im Web ist lang: Protokolle wie Token-Ring, Netzwerk- und Hierarchische Datenbanken, Programmiersprachen mit Goto[3] und vieles andere mehr.

In andern Fällen kann man erkennen, dass wirtschaftliche Interessen und nicht technische überlegenheit eine Entscheidung beeinflusst haben. Wie bei der Entscheidung zwischen Wechsel- und Gleichstrom mit den Verfechtern Edison und Westinghouse, bei dem es um Markt-

[2] Nachzulesen beim Wissenschaftshistoriker Thomas Kuhn
Thomas Kuhn. *The Structure of Scientific Revolutions*. University of Chicago Press, 1962

[3] Edsger Dijkstra. Go to statement considered harmful. *Communications of the ACM*, 11(3):147–148, 1968

anteil und Einkommen, nicht um Technik ging[4] sind manche Entscheidungen bei der Web-Technologie nur im Lichte der beteiligten Firmen und ihrer Interesse zu sehen. Wie bei der Elektrifizierung von Nordamerika versuchen Firmen Monopolstellungen aufzubauen, weil diese grosse Gewinne über länger Zeiten versprechen.

[4] http://en.wikipedia.org/wiki/War_of_Currents

Zu beginn des Computerzeitalters hatte IBM ein Quasi-Monopol, das sie erst beim Aufkommen des Personal Computers an Microsoft verloren haben[5]. In dieser Zeit verdiente IBM viel Geld mit sehr schwierig zu verwendenden Datenbanken und behinderte die Verbreitung der Relationalen Datenbanken, die von ihrem eigenen Personal (E.F.Codd) erarbeitet worden war, bis ihnen schliesslich von andern Firmen und der Universität Berkley (Michael Stonebraker mit Ingress) zum Erfolg verholfen wurde.

[5] Ironischerweise, weil der Verkauf von PC Hardware, die nicht geschützt war, IBM wenig einbrachte und IBM das Betriebssystem MS-DOS von Microsoft erstellen lies aber nicht kaufte sondern Microsoft erlaubte, es als Lizenz selber zu vertreiben. Eine unglückliche Entscheidung zum Out-Sourcing wichtiger Leistungen. Angeblich brauchte es ein Jahr um das System einigermassen zu kennen; vier Jahre, um es zu beherrschen!

Unix spielte bis zu Linux keine grosse Rolle, weil AT&T (die Telefongesellschaft der USA bevor deren Zerschlagung wegen Monopolstellung) es nicht selber verkaufen wollte aber von andern Geld für die Nutzung verlangte - die unklare Rechtslage und wohl auch die mangelnde Leistung der Rechner behinderten die breite Nutzung; der wichtigste Grund war aber wohl die praktische Monopolstellung von Windows als Betriebssystem.

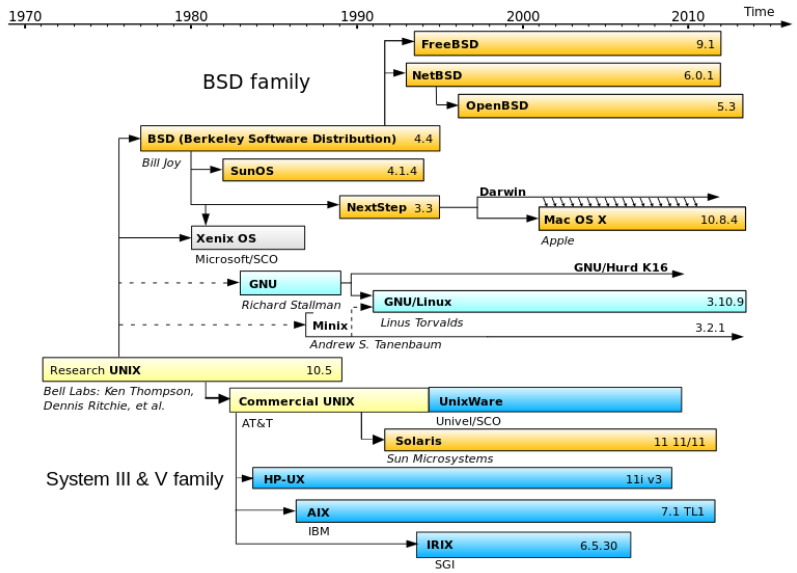


Abbildung 12.2: Die Abstammung der Unixe

Relationale Datenbanken, erfunden und entwickelt zuerst bei IBM wurden von IBM nicht auf den Markt gebracht, weil sie die Einkünfte aus dem lukrativen Geschäft mit den IMS/DB beeinträchtigt hätten; Nachfrage von Kunden und Entwicklungen bei Konkurrenten und den Universitäten brachten den Durchbruch.

Um eine Monpolstellung zu erreichen hat Microsoft mehrmals am

Markt etablierte Anbieter durch “Billigangebote” unterboten und aus dem Markt gedrängt (Visicalc und Lotus bei Tabellenkalkulation, Netscape beim Browser). Um das Monopol bei den Browsern zu halten, hat Microsoft während der “Browser Wars” Standardisierungen ignoriert und damit Mitbewerber behindert (eine Technik, die IBM und andere für FORTRAN compiler früher benutzt haben).

Heute, 2014, präsentieren sich mehrere grosse Anbieter, die je auf einem Teilmarkt nahezu ein Monopol geniessen:

Microsoft führt beim Betriebssystem für PC in Firmen und zu hause.

Apple punktet beim Benutzerinterface und dabei besonders beim Mobiltelephonen. Entscheidend ist aber wohl der iTunes store für Musik.

Google dominiert mit der Suchprogramm den Werbemarkt und versucht mit einem Linux abkömmling (android) die Software für Mobiltelefone zu beherrschen.

Amazon

Was lernt man aus diesem Mist?

- Manche Entwicklungen werden behindert und sind langsamer als erwartet, weil die Firma, die sie besitzt, sie nicht vermarkten kann (Unix, Windows Technologie bei Xerox) oder nicht vermarkten will (Relationale Datenbanken).
- Software-Entwicklung ist zunehmend eine Open-Source Bewegung, bei der grosse Firmen massive Unterstützungen bieten, aber die Grundleistung der Software im Open-Source Bereich belassen. Google bietet eigene Android Lösungen an, schneller als andere - aber die Grundlage ist offen. Gleiches gilt für die Entwicklungsumgebung Eclipse, von IBM stark unterstützt, aber Open Source.

12.4 Soziale Effekte der Web Entwicklung

Eine Technologie, die sich so rasch über die gesamte Welt verbreitet und von einer enormen Zahl von Menschen täglich verwendet wird, hat auch ökonomische und soziale Effekte. Die neue Technologie schafft Möglichkeiten für neue Geschäftspläne und damit neue Gewinnchancen; oft ist, wegen der Neuigkeit der Technik und der raschen Entwicklung eine Monopolsituation möglich mit entsprechenden Gewinnen. Es ist auffällig, wie sich die Liste der grössten Firmen der Welt verändert hat und den technischen Wandel widerspiegelt.

Die rasche Entwicklung der Kommunikationm verändert sowohl das Wirtschaftsleben als auch das Privatleben. Kommunikation ist “augenblicklich” und damit ist auch die Erwartung nach “augenblicklicher”

Antwort verbunden - was im Geschäftsleben einen rund-um-die-Uhr Zwang erzeugt, der die Abgrenzung zum Privaten zum Opfer fällt. Der unbeschränkte Austausch mit Personen, die entfernt wohnen, erlaubt den Zusammenhang in Familien, die an verschienen Orten leben mindestens zum Teil aufrecht zu halten - macht es aber nicht nötig, dass man mit den Menschen, die am gleichen Ort leben in einen ernsthaften Austausch tritt. Gemeinschaften werden virtuell und reale Gemeinschaften ausgezehrt.

Die Idee, dass alles dokumentiert werden kann und jedes Detail meines Lebens auf “ewig” aufbewahrt - z.B. mittels einer Google Glass Brille - hat nur im ersten Augenblick etwas attraktives.

Sicherheitsexperten haben sich verfahren ausgedacht, die eine private Konversation produzieren - wobei privat nicht nur bedeutet, dass ich sicher bin, mit wem ich spreche und der Austausch geheim bleibt, sondern auch, dass keiner der beiden Beteiligten beweisen kann, was gesagt worden ist. Eine überraschende Forderung, die aber wesentliches der Privatheit eines Gespräches ausmacht: der andere darf nicht den Inhalt an dritte weitergeben können.

13
Literaturverzeichnis

Experiences with Metadata, volume 2 of *7th Int. Symposium on Spatial Data Handling, SDH'96*, 1996. Faculty of Geodectic Engineering, Delft University of Technology.

Inc. Apple Computer. *Human Interface Guidelines: The Apple Desktop Interface*. Addison-Wesley, 1987.

John Backus. Can programming be liberated from the von neumann style?: A functional style and its algebra of programs. *CACM*, 21: 613–641, 1978.

Richard Bird and Oege de Moor. *Algebra of Programming*. Prentice Hall International Series in Computer Science. Prentice Hall Europe, 1997.

A. T. Boin and G. J. Hunter. Facts or fiction: Consumer beliefs about spatial data quality. Spatial Science Conference 2007, 2007a.

A. T. Boin and G. J. Hunter. What communicates quality to the spatial data consumer? International Symposium on Spatial Data Quality 2007, 2007b.

E. Codd. Extending the database relational model to capture more meaning. *ACM TODS*, 4(4):379–434, 1979.

E. F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.

E. F. Codd. Relational data base: A practical foundation for productivity. *Communications of the ACM*, 25(2):109–117, 1982. Source: Max Egenhofer.

Edgar F. Codd. *The Relational Model for Database Management*. Addison-Wesley, 1991.

D. W. Davies, E. Holler, E. D. Jensen, S. R. Kimbleton, B. W. Lampson, G. LeLann, K. J. Thurber, and R. W. Watson. *Distributed Systems - Architecture and Implementation*, volume 105 of *Lecture Notes in Computer Science*. Springer-Verlag, 1981.

Ferdinand de Saussure. *Cours de linguistique générale*. Payot & Rivages, 1995.

Edsger Dijkstra. Go to statement considered harmful. *Communications of the ACM*, 11(3):147–148, 1968.

Stefan Edlich, Achim Friedland, Jens Hampe, and Benjamin Brauer. *NoSQL: Einstieg in die Welt nichtrelationaler Web-2.0-Datenbanken*. Hanser, 2010.

Andrew U. Frank. Mapquery: Database query language for retrieval of geometric data and its graphical representation. *ACM SIGGRAPH*, 16(3):199–207, 1982.

Rich Gibson and Schuyler Erle. *Google maps hacks*. O'Reilly Media, Inc., 2006.

A. Goldberg and D. Robson. The smalltalk-80 system. *BYTE*, 6(8): 14ff., 1981.

Adele Goldberg and D. Robson. *Smalltalk-80*. Addison-Wesley Publ. Co., 1983.

Antonin Guttman. R-trees: A dynamic indexing structure for spatial searching. 1984.

Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, and York Sure. Semantic web. *Berlin, Heidelberg*, 2008.

Kathleen Jensen and Niklaus Wirth. *PASCAL User Manual and Report*. Springer-Verlag, second edition edition, 1975.

William Kent. *Data and Reality - Basic Assumptions in Data Processing Reconsidered*. North-Holland, 1978.

Brian W Kernighan, Dennis M Ritchie, and Per Ejeklint. *The C programming language*, volume 2. prentice-Hall Englewood Cliffs, 1988.

Thomas Kuhn. *The Structure of Scientific Revolutions*. University of Chicago Press, 1962.

J. Nievergelt and J. Weidert. *Methodology of Interaction*, chapter Sites, Modes and Trails : Telling the User of an Interactive System where he is, what he can do, and how to get to places, pages 327–338. North Holland, 1980.

Dennis M Ritchie and Ken Thompson. The unix time-sharing system. *Communications of the ACM*, 17(7):365–375, 1974.

Erwin Schröder. *Vorlesungen über die Algebra der Logik (Exakte Logik)*. Teubner, 1890.

Andrew S Tanenbaum. Minix operating system. *Japan, Apr*, 21:328, 1989.

C. Zaniolo and C. Delobel, editors. *Application of DBMS to Land Information Systems*, Seventh International Conference on Very Large Data Bases VLDB, 1981.

Hubert Zimmermann. Osi reference model–the iso model of architecture for open systems interconnection. *Communications, IEEE Transactions on*, 28(4):425–432, 1980.