

Grundlagen: Informationstechnologie in Bibliotheken

Dokument noch in Arbeit

Sven Koesling

Herbst 2017

Inhaltsverzeichnis

1 Einleitung: Von Nullen und Einsen	3
1.1 Die Entwicklung des Internets	9
1.1.1 www : am CERN wird Internet–Geschichte geschrieben	10
1.2 Server — was ist das eigentlich?	10
1.3 enorme Leistungssteigerung	14
1.4 Virtualisierung	15
2 Internettechnologien I	18
2.1 Dokumentformen	18
2.2 html	20
2.3 pdf	22
2.4 Exkurs: stateless	24
2.5 Skriptsprachen	26
2.6 Ajax	30
2.7 Responsive Web	32
3 Datenbanktechnologien I	33
3.1 Datenbanktypen	33
3.2 dokumentorientierte Datenbanken	36
3.3 Exkurs: Indexer	39
4 Internettechnologien II: von interaktiven Webseiten zu WebApps in der Cloud	42
4.1 Das DOM	42
4.2 Das DOM für Gestaltungszwecke verwenden	43
4.3 Das DOM für JavaScript-Funktionen verwenden	47
4.4 JavaScript–Libraries und –FrameWorks	50
4.5 Cloud Services	54
4.6 Das DOM für automatisiertes Testen von WebApplikationen verwenden	57
4.7 Exkurs: Das DOM für automatisiertes Testen von WebApplikationen verwenden	61
5 Datenbanktechnologien II: BigData	67
5.1 Klärung verschiedener Begriffe und Buzzwords	67
5.2 Anwendungsszenarien von Big Data, Anwendung in der ETH–Bibliothek	68
5.3 In Medias Res: DataScience am Bsp. Logfiles und Benutzerdaten, Big–Data am Bsp. OAI–Server	77
6 Von Verschlüsselung via hashing zur BlockChain	81
6.1 Verschlüsselung	81
6.2 hashing	82
6.3 BlockChain	83

1 Einleitung: Von Nullen und Einsen

Um die Konzepte hinter aktuellen Entwicklungen wie z.B. „Cloud“ und BigData verstehen zu können, benötigen wir ein wenig Grundlagenwissen. Das werden wir im Laufe der Lektionen aufbauen. Mit diesem Wissen wird uns dann die Logik hinter den Entwicklungen verständlich.

Dazu schauen wir zunächst auf die Geschichte des Internets, befassen uns mit der Technik und bauen Verständnis für die grundlegenden Konzepte auf.

Wir verschiedene Möglichkeiten zur Speicherung von Daten kennen, werden die unterschiedlichen Typen von Datenbanken besprechen und feststellen, dass die Tabelle nicht immer die ideale Form für das Ablegen von Daten ist, geschweige denn für die Formatierung bzw. Darstellung.

So kommen wir dann auf Grundlagen der Formatierung von Webinhalten, die Struktur von Webseiten und die Manipulation derselben durch Javascript, gehen auf Frameworks ein und kommen schliesslich auf Webapplikationen.

Ein Exkurs zeigt uns, wie man die bisher erlernten Konzepte und Techniken zum automatisierten Testen von Webapplikationen nutzen kann, bevor wir dann klären, was die viel gerühmte „Cloud“ ist und wo wir in unserer Branche beim Thema BigData stehen.

Zunächst wollen wir uns mit der Denkweise von Informatikern vertraut machen, die für „normale“ Menschen etwas gewöhnungsbedürftig sein kann.

Informatiker lachen beispielsweise herzlich über folgenden Satz:

Es gibt 10 Sorten von Menschen: Diejenigen, die das Binärsystem verstehen,
und die Übrigen. (Autor unbekannt)

Computer basieren darauf, dass man in Schaltkreisen den Strom an- bzw. abschalten kann. Es gibt nur die zwei Zustände „AN“ und „AUS“. Mathematisch ist das kein Problem, mit jeder Anzahl von Ziffern > 2 lässt sich zählen. Aber die Wirklichkeit lässt sich nur näherungsweise damit beschreiben.

Der Satz macht nur dann Spass, wenn man das Binärsystem kennt und versteht. Im täglichen Leben benutzen wir das Dezimalsystem. Zum besseren Verständnis nehmen wir an, dass jeder Zahl eine unendliche Anzahl von Stellen vorangestellt ist, die den Wert 0 haben. Also statt 1 nehmen wir 00000001 an (hier mit sieben Stellen vorneweg, weil sich eine unendliche Anzahl so schlecht aufschreiben lässt...).

Wir zählen die Ziffern von 0 bis 9 hoch, und wenn die Ziffern aufgebraucht sind, erhöhen wir die Stelle davor um eins und setzen die eben hochgezählte Ziffer auf 0 zurück. So wird die Zahl 9 (00000009), wenn man um eins erhöht, zu 10 (00000010)

$$\begin{array}{ccccccccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9 \\ & & & & & & +1 & \mid \text{auf 0 zurücksetzen} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}$$

Im Binärsystem stehen uns nur zwei Ziffern zur Verfügung. Aber zwei Ziffern reichen zum Zählen. Man wendet das gleiche Prinzip an. So wird die Zahl 1 (00000001), wenn man um eins erhöht, zu 10 (00000010). Jetzt ergibt der Witz einen Sinn. Der binäre Wert 10 entspricht in unserem Dezimalsystem dem Wert zwei.

Die Folge der ersten neun Binärzahlen...

0, 1, 10, 11, 100, 101, 110, 111, 1000 ...

...und die „Übersetzung“

binär	dezimal
0	0
1	1
10	2
11	3
100	4
101	5
110	6
111	7

Leibniz schreibt Ende des 17. Jahrhunderts dazu:

...deshalb ist der letzte Tag der vollkommenste und der Sabbat, denn an ihm ist alles geschaffen und erfüllt, und deshalb schreibt sich die 7 111, also ohne Null. Und nur wenn man die Zahlen bloß mit 0 und 1 schreibt, erkennt man die Vollkommenheit des siebenten Tages. (Gottfried Wilhelm Leibniz)

Ein Computer kann aktuell¹ nur zwei Zustände darstellen: „AN“ und „AUS“. Damit ist das Binärsystem für Computer zum Rechnen ideal.

¹Die zur Zeit in Entwicklung befindlichen Quantencomputer kennen auch noch „vielleicht“. Aber obwohl erste Schritte vielversprechend sind, liegen Quantencomputer noch in weiter Ferne. Und Quantenphysiker denken noch schräger als Informatiker — das möchte ich dem geneigten Leser nicht zumuten.

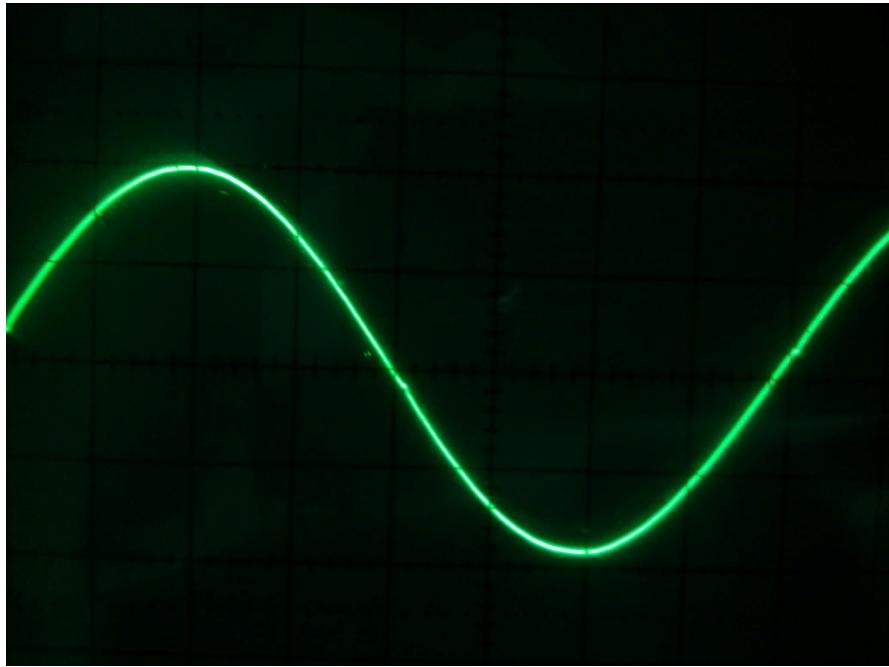


Abbildung 1: Sinuskurve, Quelle: Computer:club², Urheber: Rolf Degen

Hinweis

Für uns wäre es aus verschiedenen Gründen sehr unpraktisch, im Binärsystem zu rechnen. Die Zahlen werden z.B. sehr schnell sehr lang. So braucht die Sieben im Binärsystem schon drei Stellen (111).

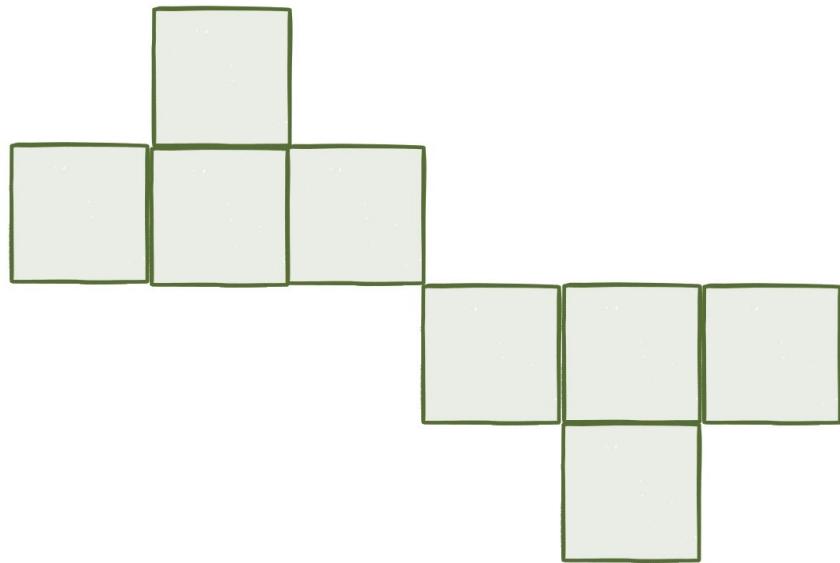
Zahlen können wir also genau umrechnen. Aber wie bilden beispielsweise Musik digital ab?

Man kann sich mit „AN“ / „AUS“ – Zuständen einer Kurve annähern. Zunächst vereinfachen wir das Problem, indem wir statt Musik eine Sinuskurve zur Veranschaulichung benutzen. Das geht, weil ein Ton eine Summe von Schwingungen ist. Sinustöne sind sehr reine Töne, deren Schwingung sehr sauber der Sinuskurve entspricht.

eine Sinuskurve

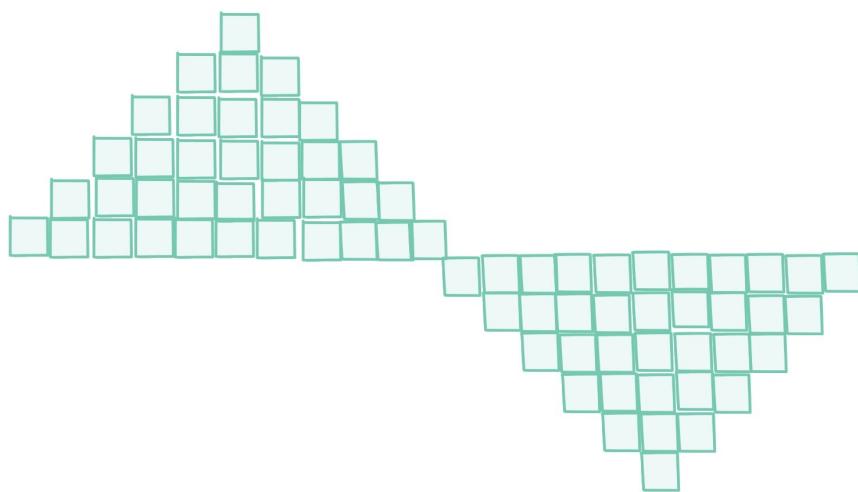
Wir wollen mal versuchen, mit unseren zwei Zuständen eine solche Kurve zu erzeugen, indem wir sie auf Karos abbilden. Nehmen wir an, dass ein Karo dem Zustand „AN“ entspricht, kein Karo dem Zustand „AUS“. Wenn man jetzt Karos zu einer Pyramide zusammenstellt und dahinter an der Basis genau so eine Pyramide nach unten zeigen lässt, erhält man eine Annäherung an eine Sinuskurve. Die Abbildung der Kurve sähe dann etwa aus wie folgt.

eine grobe Annäherung



Je mehr Informationen bzw. „AN“ / „AUS“ – Zustände wir nun verwenden, desto feiner wird die Annäherung an die Kurve:

eine feinere Annäherung



Wenn man mal von meinen mangelnden Zeichenkünsten absieht, kommt das Ganze der Kurve schon näher. Da unser Auge wie unser Ohr ein begrenzt feines Messinstrument

ist, brauchen wir das Karo– Muster nur klein genug zu machen (oder weit genug vom Augen zu entfernen) und können die Treppenstufen der Karos nicht mehr sehen bzw. das Unnatürliche im Ton nicht mehr hören.

Mit „AN“ / „AUS“ – Zuständen lässt sich also die Wirklichkeit näherungsweise beschreiben.

Hinweis

Egal, wie leistungsfähig Computersysteme sind, oder noch sein werden, sie werden immer nur eine Annäherung bieten können.

Glücklicherweise ist unser Ohr als Messinstrument so unsensibel, dass wir die Näherungen nicht mehr vom Original unterscheiden können, solange sie nur fein genug sind. So konnte Ende der siebziger Jahre die Firma Philipps der digitalen Musikproduktion und –wiedergabe mit der Einführung des CD–Spieler kräftigen Schwung geben.

Mit Bildern ist das Prinzip ähnlich: Je feiner die Treppen, je höher also die Informationsdichte, desto wirklichkeitsgetreuer wird das Bild wiedergegeben.

Je mehr, desto besser

Je mehr „an“ / „aus“ Informationen wir einsetzen, desto näher ist das Ergebnis an der Wirklichkeit. Entsprechend steigen aber auch die benötigte Rechenleistung und der Speicherplatzbedarf an. Das sind nun relativ kleine Beispielbilder gewesen. Rohbilder moderner Kameras benötigen aktuell 25 MB pro Bild, im semiprofessionellen Bereich das Doppelte, und im Mittelformat braucht es pro Bild ca. 600MB. Deshalb rechnet man mit speziellen Algorithmen „nicht benötigte“ Informationen aus den Daten heraus. Auch wenn der Leistungszuwachs von Computern aussergewöhnlich ist, stehen Speicher und Rechenpower nicht unbegrenzt zur Verfügung und sollten ganz normal als Ressource wahrgenommen werden, die man nicht verschwendet.

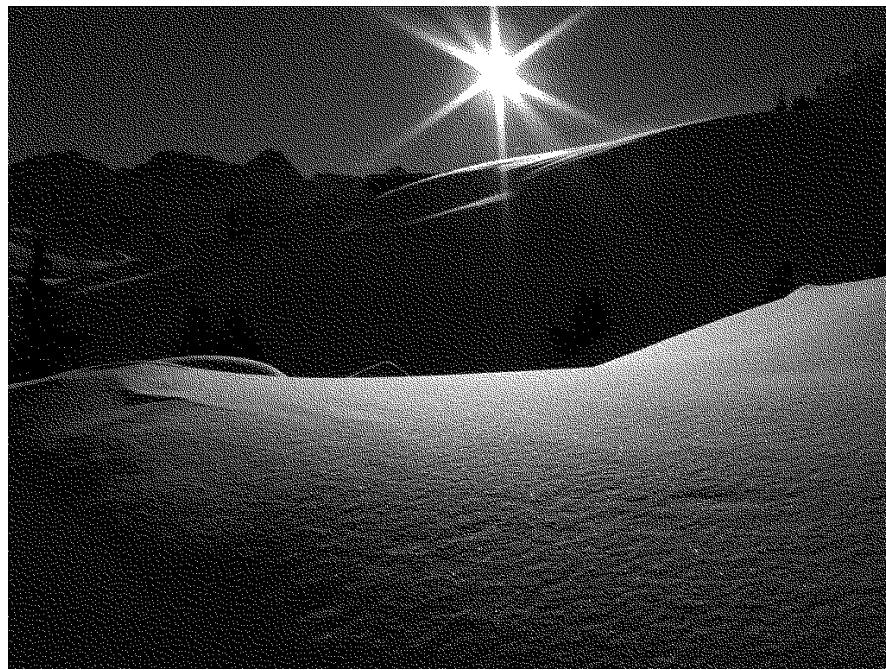


Abbildung 2: Ein Bild mit wenig Informationen (und kaum Speicherplatzbedarf: 123kb)



Abbildung 3: Ein Bild mit etwas mehr Informationen (und etwas mehr Speicherplatzbedarf: 551kb)



Abbildung 4: Ein Bild mit vielen Informationen (und noch mehr Speicherplatzbedarf:
1,1 MB)

1.1 Die Entwicklung des Internets

- ARPA
- E-Mail
- www — ein neuer Treiber
- Web Apps, Cloud Services und intelligente Kühlschränke

Der Vorläufer des Internets: Das Arpanet

Das Internet ist 1969 aus einer Zusammenarbeit des US-Verteidigungsministeriums und verschiedenen Forschungseinrichtungen entstanden. Obwohl sich das Gerücht hält, dass im kalten Krieg eine Technik aufgebaut werden sollte, die im Falle eines Atomschlages die Kommunikationsinfrastruktur erhalten kann, dürfte ein ausschlaggebender Grund für die Entwicklung die bessere Ausnutzung teurer Rechenkapazitäten gewesen sein. So entstand als Vorläufer des heutigen Internets das Arpanet. So oder so — die Idee ist genial: Man teilt Kommunikation in kleine Päckchen auf und entwickelt Protokolle,

die es ermöglichen, dass sich diese Päckchen ihren Weg selbstständig vom Sender zum Empfänger suchen. Dabei gibt es nicht nur eine Leitung von A nach B, sondern ein fein verzweigtes Netzwerk mit unzähligen Knoten. Wenn nun eine Leitung blockiert ist, nimmt das Päckchen einfach einen anderen Weg. Übertragen wurden damals Übrigens noch keine Webseiten mit Bildern.

1.1.1 www : am CERN wird Internet–Geschichte geschrieben

Das CERN spielt eine wichtige Rolle bei der Entwicklung des Internets, wie wir es heute kennen. Die Laboratorien des CERN liegen teilweise auf Schweizer Gebiet, teilweise auf französischem. Natürlich setzt jedes Land seine eigenen Computersysteme ein, was es damals unmöglich machte, Texte online auszutauschen. Mitte der achtziger Jahre nahm sich ein britischer Physiker und Informatiker namens Tim Berners-Lee dieses Problems an und entwickelte mit seinem Kollegen Robert Cailliau ein Konzept für ein weltweites Hypertext-Projekt, das sie 1990 veröffentlichten. Das daraus entstehende Protokoll „http“ und die Auszeichnungssprache „html“ sind auch heute noch die Grundlagen des **World Wide Web**. Dabei geht es darum Texte über das Netzwerk zur Verfügung stellen zu können. Die Problematik, Texte universell für verschiedenste System darstellbar zu übertragen, wird auch heute noch deutlich, wenn man z.B. eine Webseite auf einem Smartphone öffnet, die für den Desktop optimiert ist. Inzwischen sind die Texte um Bilder „bereichert“, Filme werden über das Internet gestreamt und Weltkarten in 3D betrachtet.

Das Internet der Dinge

Da man nun immer mehr Leistung in immer kleinere Chips packen kann, können kleinste Dinge Funktionen bekommen, für die man früher ganze Rechenzentren benötigte. Wecker zeigen das Wetter an, Kalender berechnen die Wegzeit automatisch in Alarne mit ein, Navigationssysteme verwenden aktuelle und zu erwartende Verkehrsdaten, um die optimale Route zu bestimmen. Für all das benötigen die Dinge eine Verbindung zum Internet. Das wirft verschiedene Probleme auf: Sicherheit und Datenschutz sind ein Thema. Aber auch technisch braucht es neue Ansätze. So muss jeder, der an ihn gerichtete Informationen bekommen will, eindeutig identifizierbar sein. Die ursprünglich für diesen Zweck gemachte Adressierung (IPv4) hat für das Internet der Dinge viel zu wenige Adressen. Eine neue Technik zur Adressierung — IPv6 — ist vorhanden, setzt sich aber nur langsam durch.

1.2 Server — was ist das eigentlich?

- Hardware

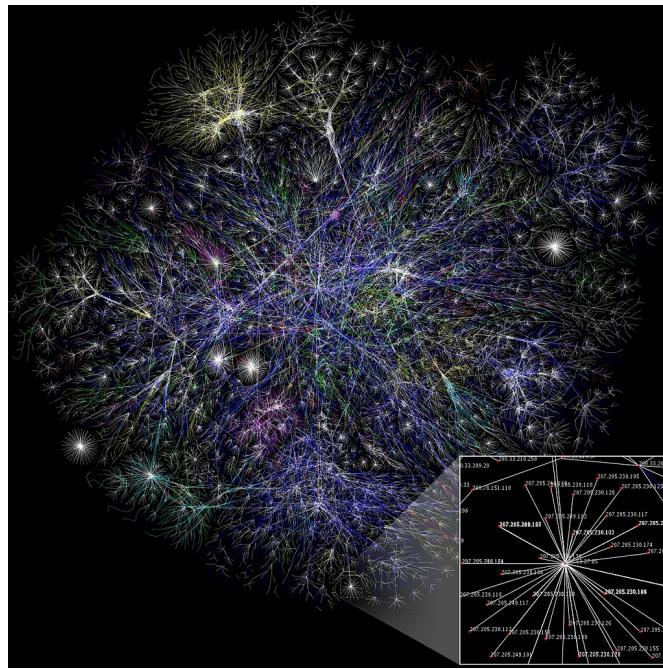


Abbildung 5: Das Internet heute; Quelle: Wikipedia, Urheber: The Opte Project

- Virtualisierung
 - Software

Hardware

Was ist ein Server? Im Grunde genommen ist jedes Gerät, das Dienstleistungen zur Verfügung stellt, ein Server. Wenn ich von meinem lokalen PC eine Webseite bereit stelle, die jemand anderes von seinem PC aufruft, ist mein PC in dem Moment ein Server.

Ursprünglich war Rechenleistung so teuer, dass es in einer Institution nur einen riesigen Rechner in einem gut gekühlten Keller gab. Terminals ohne eigene Rechenkapazität stellten für eine Anzahl von Nutzern eine Verbindung zu diesem Rechner her, dessen Leistung unter allen Nutzern aufgeteilt wurde. Ende der 70er Jahre wurde die Technik so billig, dass mit dem Aufkommen der Personal Computers jeder seinen eigenen Rechner auf seinem Schreibtisch hatte.

Im Jahr 2000 wurden für den Aufbau einer Datenbank für kurze Texte noch 40.000 DM für einen Server der Firma SUN ausgegeben. Der Vorschlag, stattdessen einen Linux-PC



Abbildung 6: MacMini Server; Quelle: Gizmodo India

für 5'000 DM einzusetzen war revolutionär... und erfolgreich waren die PCs so leistungsfähig geworden, dass es keinen Grund mehr gab, das Achtfache zu investieren.

Woraus besteht nun ein PC?

die Komponenten eines PCs

Das Herzstück des PCs ist der Prozessor — der eigentliche Rechner. Damit Menschen mit ihm kommunizieren können, gibt es Ein- und Ausgabesysteme. Bis in die achtziger Jahre wurden dafür z.B. Lochkarten gebraucht, heute benutzen wir im Wesentlichen Tastatur und Maus, immer öfter auch den Touchscreen. Monitore und Drucker sind Beispiele für Ausgabesysteme.

Die Daten müssen irgendwo gespeichert werden. Auch hierfür taugt die Lochkarte bzw. ganze Lochbänder, inzwischen abgelöst durch magnetische Medien, optische Datenträger und nichtflüchtige elektronische Speicher. „Nichtflüchtig“ leitet zu einem weiteren Speicher Über: Der Prozessor eines PCs ist so schnell, dass er einen besonderen, schnellen Zwischenspeicher benötigt, von dem er Daten laden und auf dem er seine Ergebnisse ablegen kann. Die Rede ist vom RAM, ein schneller elektronischer Speicher, der allerdings seine Informationen sofort verliert, wenn der Strom abgeschaltet wird.

Auf den meisten Heim- PCs ist heutzutage nicht mehr der Prozessor (CPU) der schnellste im Team. Den Titel hat er an den Grafikprozessor (GPU) abgegeben. Dazu muss man



Abbildung 7: die Komponenten eines PCs

wissen, dass die Spieleentwicklung einer der Treiber bei der technischen Entwicklung von PCs ist. Und für Bilder gilt genauso wie für alles Digitalisierte: Je wirklichkeitsgetreuer die Darstellung sein soll, desto mehr „AN“ / „AUS“ Informationen brauchen wir, und desto leistungsfähiger muss die Hardware sein.

Verbunden werden all die Komponenten durch das sogenannte Motherboard. Auf ihm sitzt auch ein weiterer kleiner Chip, der beim Einschalten des PCs erstmal die Komponenten sortiert und entscheidet, was gestartet werden soll. Das Problem ist nämlich, dass kein Bauteil vom anderen „weiss“. Wir brauchen eine Software, die die Bauteile miteinander verknüpft, das Betriebssystem. Das Betriebssystem übernimmt vom Chip die Hoheit über den Computer und stellt uns unsere Arbeitsumgebung — im Alltag also den Desktop mit Mail–Programm, Textverarbeitung usw. — zur Verfügung.

Server: Software

Das Wort Server wird synonym auch für die Software benutzt, die dafür zuständig ist, Services zu erbringen. Wenn wir von einem Webserver sprechen, kann sowohl die (virtuelle) Maschine gemeint sein, die die Webseiten ausliefert, als auch die Software — z.B. Apache, nginx, puma etc. — auf der Maschine, die diese Arbeit übernimmt.

Beispiele für Software– Server

- Webserver
- Mailserver
- Dateiserver
- Datenbankserver

Immer übernimmt eine entsprechende Software die Aufgabe, Daten auszuliefern bzw. zu empfangen. Auf einem Hardwareserver können mehrere Softwareserver laufen, auch wenn es im Zuge der Virtualisierung sinnvoll erscheint, für jeden Serverzweck eine eigene virtuelle Maschine zu Verfügung zu stellen. So läuft in der ETH–Bibliothek z.B. das Bibliothekssystem auf einem virtuellen Server, die Datenbank auf einem anderen.

1.3 enorme Leistungssteigerung

Das Moorsche Gesetz besagt dass sich die Rechenleistung alle ein bis zwei Jahre verdoppelt. Inzwischen haben wir mehr Leistung in einem Mobiltelefon, als noch vor zehn Jahren auf dem Desktop.

Wie konnten diese bemerkenswerten Leistungssteigerungen erzielt werden? Stark vereinfacht (und anders als beispielsweise in der Chemie) mit der Formel „viel hilft viel“.

Vor noch gar nicht so langer Zeit wurden Daten auf einer sogenannten Festplatte gespeichert. In einem luftdichten Gehäuse wurde eine beschichtete Platte zum Rotieren gebracht und mit einem Lese-/Schreibkopf die Daten darauf magnetisch abgelegt bzw. ausgelesen. Um die Geschwindigkeit zu erhöhen lies man zum einen die Platten immer schneller rotieren, zum anderen erhöhte man die Dichte der magnetisierbaren Partikel auf der Platte, so dass bei gleicher Rotationsgeschwindigkeit mehr Informationen am Kopf vorbei kamen.

Was macht man, wenn die Platte voll ist? Man kauft eine zweite.

Findige Köpfe kamen auf die Idee, die zweite Platte in das selbe Gehäuse einzubauen. Inzwischen besteht eine Festplatte aus einem Plattenstapel, in dessen Zwischenräumen die Köpfe kammartig greifen und die Informationen schreiben bzw. lesen. Durch die gleichzeitigen Schreib- und Lesevorgänge konnten neben dem erhöhten Speicherplatz auch die Geschwindigkeit weiter gesteigert werden.

Ein ähnliches Prinzip funktioniert auch bei Prozessoren: Statt einem benutzt man mehrere gleichzeitig, und da man dank des Fortschritts in der Halbleitertechnologie immer mehr Transistoren auf immer kleinerem Raum unterbringt, kann man in einem Prozessor mehrere Kerne realisieren, so dass mehr Operationen gleichzeitig ausgeführt werden können.

1.4 Virtualisierung

Hardware ist so billig — warum sollte man einen PC virtualisieren wollen?

Seit Microsoft mit seinen Produkten den PC-Markt beherrscht, stellt sich für Nicht-Windows-Nutzer ein Problem: Durch die enorme Verbreitung von Windows im PC-Markt gibt es viel Software, die ausschliesslich für Windows geschrieben wird. Was macht man nun als Benutzer eines anderen Betriebssystems, wenn man diese Software nutzen möchte?

Da seit einiger Zeit ein PC genug Leistung für zwei hat, sind kluge Köpfe auf die Idee gekommen, einen PC innerhalb eines PCs zu simulieren.

Der Clou ist, dass man alle Komponenten eines PCs entweder teilen oder komplett simulieren kann. Wenn ein PC beispielsweise 8GB RAM hat, dann kann man 4GB davon für einen virtuellen PC benutzen. Unserem PC, unserem Betriebssystem stehen dann nur noch 4GB zur Verfügung, die Übrigen 4GB „gehören“ der virtuellen Maschine.

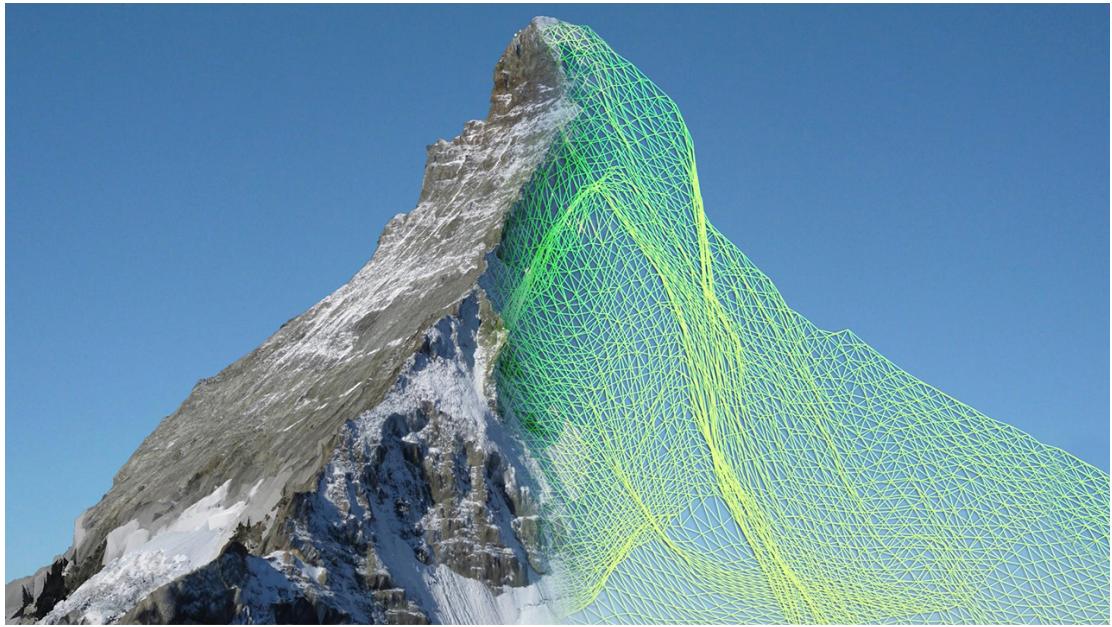


Abbildung 8: Photomontage: Jamani Caillet / © EPFL
<http://actu.epfl.ch/news/the-matterhorn-like-you-ve-never-seen-it/>

Man ruft also auf seinem echten PC ein Programm auf, das per Software nun alle Komponenten eines PCs noch einmal simuliert und diesen virtuellen PC startet. Eine Festplatte in diesem „PC“ ist dabei nur eine grosse Datei auf unserer echten Festplatte. Und in diesem virtuellen PC lässt sich dann ein eigenes Betriebssystem installieren. Auf diesem Weg hat man nun auf einem PC gleichzeitig Windows und Linux zur Verfügung.

Da die Leistungsfähigkeit der Hardware inzwischen so stark gestiegen ist, kann man auf einem Computer gleichzeitig mehrere virtuelle PCs starten.

Vorteile der Virtualisierung

- Da virtuelle Computer nur Dateien auf einer Festplatte sind, kann man sie komplett in einem Backup sichern und quasi auf Knopfdruck wieder herstellen.
- Wenn man für einen Computer kurzfristig mehr Leistung braucht, kann man einem virtuellen Computer einfach per Software mehr RAM oder weitere CPUs zur Verfügung stellen. Das geht teilweise unterbruchsfrei.
- Computer sind selten ausgelastet. Wenn man seinen Bedarf auf virtuelle Maschinen verteilt, ist die Auslastung der echten Systeme besser.

15.12.2017 : Internettechnologien I, Datenbanktechnologien I

- IntT I: Dokumentformen, Skriptsprachen, Ajax, responsive Web
- DBT I: Datenbanktypen, Technologien, Einstieg SQL

2 Internettechnologien I

2.1 Dokumentformen

Mit der Entwicklung von http ab 1989 wurde das www begründet und der Austausch von Daten aus Anwendungen heraus zwischen Rechnern ermöglicht. Http wird hauptsächlich benutzt, um mit einem Browser Webseiten zu laden. Dazu fragt der Browser den Server nach einer Seite, der Server liefert sie an den Browser aus, und dieser stellt sie dann dar.

Bei Webseiten handelt es sich zunächst einmal um mehr oder weniger simple Textdateien.

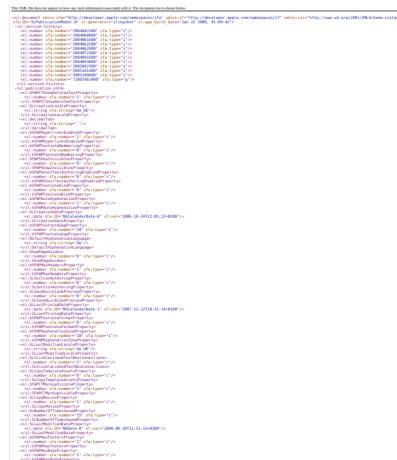
- Warum nicht Word, Pages, ...?

Das Web ist eine unvorstellbar große Sammlung von Dokumenten. Wenn man also im Internet surft, guckt man sich Dokumente an, die auf anderen Rechnern liegen. Nun haben doch alle Word — warum also ein weiteres Format (html)? Es gibt dazu viele Argumente:

1. Dokumentgrößen

 dokument.doc	28 KB
 dokument.docx	12 KB
 dokument.html	4 KB

die Dokumentauszeichnung am Beispiel “Pages”



(noch 12 mal soviel...)

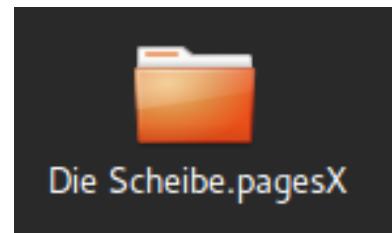
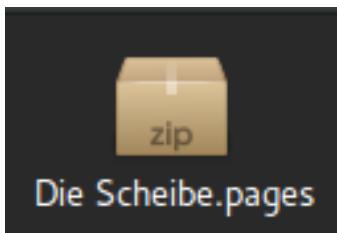
der Dokumentinhalt am Beispiel “Pages”



Bei der Übertragung über das Netz ist die Datei–Größe trotz “dicker Leitungen” nach wie vor entscheidend: Ein Word- Dokument mit dem gleichen Inhalt ist sieben mal (Word97-2003) bzw. dreimal (Word2007) grösser, als eine html-Datei gleichen Inhalts. html ist ein Format, das sehr schlank ist und schnell übertragen werden kann.

2. frei verfügbares Format

Ein proprietäres Format wie .doc wird laufend verändert, wobei diese Veränderungen nicht dokumentiert werden. Wer das Dokument lesen will, muss die Software zum Lesen kaufen (können...).



Webseiten sollten aber von jedem weltweit betrachtet werden können.

Ein weiterer Unterschied: Dokumente von Textverarbeitungen sind dafür gemacht, dass man sie leicht editieren kann. html- Dokumente dagegen sollen nach ihrer Erzeugung zunächst mal nur angezeigt, aber nicht bearbeitet werden.

Der Webbroweser schickt dazu eine Anfrage an den Server "Reich mir mal die Seite /dokument.html rüber!". Wenn die Seite vorhanden ist, gibt der Server sie heraus, sonst wird eine Fehlermeldung angezeigt. Der Webbrowserrückt die Seite raus, der Browser zeigt die Seite an - sobald man aber weiter surft, ist sie schon wieder vergessen (grund-sätzlich... Browser speichern heute Seiten in einem sogenannten Cache, damit sie beim nächsten Aufruf schneller kommen.). Bearbeiten kann man die Seite nicht. Mit speziellen Tools lässt sich der empfangene Code natürlich editieren, dann aber nur auf dem eigenen Rechner betrachten. Der Server wird es nicht erlauben, dass jeder einfach seine Änderungen auf ihm abspeichert.

Und schliesslich gibt es für Textverarbeitungen fest definierte Dokumentgrößen (A4 z.B.), Webseiten müssen aber auf den unterschiedlichsten Monitorgrößen angezeigt werden. Das bedeutet, dass z.B. Zeilenumbrüche flexibel sein müssen.

2.2 html

html ist zunächst einmal keine Programmiersprache. Man kann keinen Rechner damit füttern und ihm das Ergebnis von $2 + 2$ entlocken.

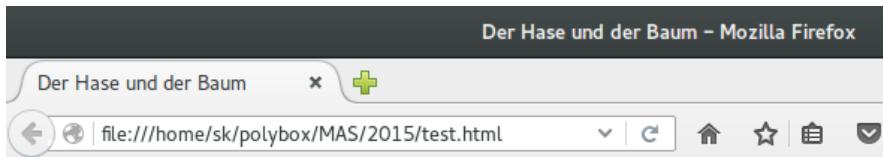
html ist eine Auszeichnungssprache, die dazu dient, Inhalt zu formatieren. Dazu versieht der Programmierer den strukturierten Inhalt mit Tags — eines zum Anfang und eines zum Ende des jeweiligen Bereichs. So kennzeichnen beispielsweise `<p> </p>` einen Absatz. Zwischen den Tags steht dann der Inhalt.

- Warum nicht Word, Pages, ...?
- html

eine einfache Seite

```
<html>
  <head>
    <title>Der Hase und der Baum</title>
  </head>
  <body>
    <h1>Der Hase und der Baum</h1>
    <h2>Kapitel 1: Der Hase</h2>
    <p>Meister Lampe hoppelt über ein Feld.</p>
    <h2>Kapitel2: In der Werkstatt</h2>
    <p>Herr K. bestellt eine Knautschzone.</p>
  </body>
</html>
```

...sieht so aus:



Der Hase und der Baum

Kapitel 1: Der Hase

Meister Lampe hoppelt über ein Feld.

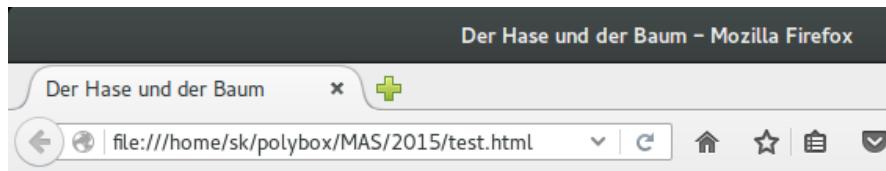
Kapitel2: In der Werkstatt

Herr K. bestellt eine Knautschzone.

...mit Umlauten:

```
<html>
  <head>
    <meta charset="utf-8" />
    <title>Der Hase und der Baum</title>
  </head>
  <body>
    <h1>Der Hase und der Baum</h1>
    <h2>Kapitel 1: Der Hase</h2>
    <p>Meister Lampe hoppelt über ein Feld.</p>
    <h2>Kapitel2: In der Werkstatt</h2>
    <p>Herr K. bestellt eine Knautschzone.</p>
  </body>
</html>
```

...sieht so aus:



Der Hase und der Baum

Kapitel 1: Der Hase

Meister Lampe hoppelt über ein Feld.

Kapitel2: In der Werkstatt

Herr K. bestellt eine Knautschzone.

2.3 pdf

Wenn man ein Dokument mit festem Layout veröffentlichen möchte, bietet sich PDF an. Es ist frei verfügbar, kann also von jedem gelesen werden, und sieht immer gleich aus. Wie auch Webseiten lässt es sich (eigentlich) nicht vom Empfänger verändern.

- Warum nicht Word, Pages, ...?
- html
- pdf

Eigenschaften von pdf

- proprietär, aber offen gelegt
- Papiergrösse, Layout und Inhalt festgelegt
- Text als Text, Bild als Bild enthalten

Wieder die Dokumentgrösse:

Django 2.0: Neue Version des Python-Webframeworks

Oliver Diedrich



Django 2.0 gibt die Unterstützung für Python 2 endgültig auf. Das URL-Routing lässt sich jetzt auch ohne reguläre Ausdrücke aufsetzen.

Ernsthaftes Webentwicklung bedeutet heutzutage in der Regel den Einsatz eines Webframeworks, das dem Entwickler viele Routinetätigkeiten abnimmt. In der Python-Welt ist Django, benannt nach dem Jazz-Gitarristen Django Reinhardt, seit zehn Jahren das Webframework der Wahl. Konkurrenten wie Flask konnten bislang nur Nischen besetzen.

Django setzt konsequent das Model-View-Controller-Konzept um, kapselt Datenbankzugriffe über eine objektrelationale Abbildung und generiert automatisch eine Oberfläche zur Verwaltung von Datenbank und Benutzern. Das Framework enthält eine leistungsfähige Template-Sprache und bietet eine flexible URL-Konfiguration.

Die frisch erschienene Version 2.0 ist trotz des großen Versionssprungs weitgehend rückwärtskompatibel zur Vorversion 1.11 – mit einer Ausnahme: Django 2 arbeitet nur noch mit Python 3.4, 3.5 und 3.6 zusammen. Wer noch Python 2.7 nutzt, muss bei Django 1.11 bleiben, das als LTS-Version noch bis April 2020 Sicherheits- und kritische Bugfixes erhalten soll. Django 1.10 hat mit dem Erscheinen der neuen Version das Ende des Supports erreicht.

als Bild:

178.8 kB als PDF: 67.7 kB

Fazit: Welche Form wofür?

Textverarbeitung: Alles zum Weiterverarbeiten

html: Inhalt geht über Form

PDF: Form soll erhalten bleiben

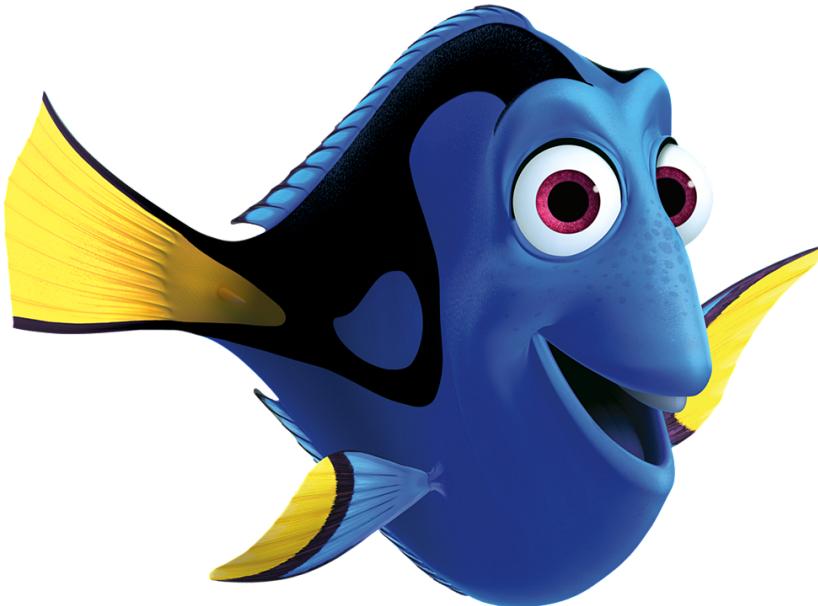
alles Quatsch?

Satz 1. *Im Web liest doch niemand mehr. Da sollte man nur knappe Infos unterbringen.*

2.4 Exkurs: stateless

Ein Problem bei der Entwicklung von Web–Apps war, dass ein Webserver eigentlich nichts vom User weiss. Er liefert die geforderte Webseite aus und hat dann den User schon wieder “vergessen”.

http is stateless



...some web applications may have to track the user's progress from page to page...Solutions for these cases include:

- the use of HTTP cookies.

- server side sessions,
- hidden variables (when the current page contains a form), and
- URL-rewriting using URI-encoded parameters, e.g., /index.php?session_id=some_unique_session_code.

(wikipedia)

Um das zu ändern gibt es verschiedene Techniken wie z.B. Cookies und Sessions. Dabei handelt die Web-App einen eindeutigen Identifier aus und kann so wiederkehrende User identifizieren. „Dank“ der Werbeindustrie wurden diese Techniken so weit optimiert, dass wir im Internet gläsern sind. Man kann feststellen, woher wir kommen, welche Seiten wir gesehen haben, welche wir wegklicken, wie lange wir wo verweilt haben und, und, und.

privacy?

You have zero privacy anyway. Get over it.
Microsystems, 1999)

(Scott McNealy, Sun

2.5 Skriptsprachen



Während man mit Auszeichnungssprachen wie z.B. html oder L^AT_EX Dokumente formatiert, dienen Skriptsprachen wie Perl, Ruby oder Javascript dazu, einfache, wiederkehrende Aufgaben am Computer zu übernehmen.

Beim Publishing von Aleph erhalten wir beispielsweise unzählige komprimierte Dateien, die ihrerseits unzählige XML-Dateien enthalten. Jede dieser XML-Dateien enthält einen Datensatz.

Das erste Programm

Number of Operations	Statement of Operation	Variables acted upon	Variables receiving results	Indication of change in the value of any Variable	Statement of Results										Working Variables		Result Variables			
					Data				Working Variables						Result Variables					
					V ₁	V ₂	V ₃	V ₄	V ₅	V ₆	V ₇	V ₈	V ₉	V ₁₀	V ₁₁	V ₁₂	V ₂₁	V ₂₂	V ₂₃	
1	= V ₂ × V ₃		V ₁₄ , V ₁₅ , V ₁₆	{V ₁₄ = V ₁₅ } = 2n	1	2	n	2n	2n	2n										
2	- V ₁₄ - V ₁₅		V ₁₄	{V ₁₄ = V ₁₅ } = 2n - 1	1															
3	+ V ₁₄ + V ₁₅		V ₁₄	{V ₁₄ = V ₁₅ } = 2n + 1	1															
4	- 2V ₁₄ + 2V ₁₅		V ₁₄	{V ₁₄ = V ₁₅ } = $\frac{2n-1}{2}$				0	0											
5	+ V ₁₁ - V ₂		V ₁₁	{V ₁₁ = V ₂ } = $\frac{2n+1}{2}$				0	0											
6	- 2V ₁₁ + V ₂		V ₁₁	{V ₁₁ = V ₂ } = $\frac{2n-1}{2}$ - 3 = $\frac{2n-7}{2}$ = A ₀				2												
7	- V ₁₀ - V ₁		V ₁₀	{V ₁₀ = V ₁ } = n - 1 (n > 3)	1															
8	+ V ₉ + V ₂		V ₉	{V ₉ = V ₂ } = 2 + 0 = 2				2												
9	- V ₉ + V ₁₁		V ₁₁	{V ₉ = V ₁₁ } = 2n - 2					2n	2										
10	+ V ₁₂ × V ₁₃		V ₁₂	{V ₁₂ = V ₁₃ } = 0, $\frac{2p}{2} = B_1 A_1$																
11	- V ₁₂ + V ₁₃		V ₁₂	{V ₁₂ = V ₁₃ } = $\frac{2p-1}{2}$ - 3 = $\frac{2p-7}{2}$ = B ₁																
12	- V ₁₀ - V ₁		V ₁₀	{V ₁₀ = V ₁ } = n - 2 (n > 2)	1															
13	{+ V ₈ - V ₁		V ₈	{V ₈ = V ₁ } = 2n - 1						2n - 1										
14	{+ V ₁ + V ₇		V ₇	{V ₁ = V ₇ } = 2 + 1 = 3	1						3									
15	{+ V ₉ + V ₂		V ₂	{V ₉ = V ₂ } = $\frac{2p-1}{2}$						2n - 1	n	$\frac{2p-1}{2}$								
16	- V ₁₀ × V ₁₁		V ₁₁	{V ₁₀ = V ₁₁ } = 2p - 2							0									
17	- V ₈ - V ₁		V ₈	{V ₈ = V ₁ } = 2n - 2	1					2n - 2										
18	{+ V ₄ + V ₅		V ₅	{V ₄ = V ₅ } = 3 + 1 = 4	1						4									
19	{+ V ₆ + V ₇		V ₇	{V ₆ = V ₇ } = $\frac{2p-1}{2}$						2n - 2	4	$\frac{2p-1}{2}$								
20	{+ V ₈ + V ₁₁		V ₁₁	{V ₈ = V ₁₁ } = 2p - 2							0									
21	- V ₁₂ × V ₁₃		V ₁₂	{V ₁₂ = V ₁₃ } = B ₂ , $\frac{2p}{2} = B_2 A_2$											B ₂ A ₂					
22	- V ₁₂ + V ₁₃		V ₁₂	{V ₁₂ = V ₁₃ } = $\frac{2p-1}{2}$ - 3 = $\frac{2p-7}{2}$ = B ₂ A ₂											0					
23	- V ₁₀ - V ₁		V ₁₀	{V ₁₀ = V ₁ } = n - 3 (n > 1)	1															
24	+ V ₁₃ + V ₂₄		V ₂₄	{V ₁₃ = V ₂₄ } = B ₂															B ₂	
25	+ V ₁ + V ₉		V ₉	{V ₁ = V ₉ } = n - 1 = 4 + 5 by a Variable-card. by a Variable-card.	1			n + 1			0	0								

Here follows a repetition of Operations thirteen to twenty-three

Spielereien in Perl und Ruby

Die Aufgabe lautet, ein Programm zu schreiben, das die Zahlen von 1 – 100 hoch zählt und

- bei Zahlen, die durch **drei** teilbar sind, „Fizz“ ausgibt,
- bei Zahlen, die durch **fünf** teilbar sind, „Buzz“ ausgibt
- bei Zahlen, die durch **beides** teilbar sind, „FizzBuzz“ ausgibt.

Ansatz in Perl

```
#! /usr/bin/perl
for my $zahl (1 .. 100){
    if ($zahl % 15 == 0) {
        print "Fizz Buzz\n";
    }
    elsif ($zahl % 5 == 0) {
        print "Buzz\n";
    }
    elsif ($zahl % 3 == 0) {
        print "Fizz\n";
    }
    else {
        print "$zahl\n";
    }
}
```

Ansatz in Ruby

```
#!/usr/bin/ruby
(1..100).each do |z|
  if z.modulo(15) == 0
    puts "FizzBuzz :-)"
  elsif z.modulo(5) == 0
    puts "Buzz"
  elsif z.modulo(3) == 0
    puts "Fizz"
  else
    puts z
  end
end
```

Ein Programm

Das folgende Ruby-Skript (Auszug) dient zum Durchsuchen von tar-files:

```
Dir.foreach('./') do |item|
  next unless item =~ /\.tgz$/ or item =~ /tar.gz$/
  puts "=====
cmd = "tar tf #{item}"
ergebnis = %x[ #{cmd}]
ergebnis.each_line do |file|
  suchstring.each do |s|
    if file =~/#{s}/
      puts "#{file.chomp} gefunden."
      print "Soll die Datei extrahiert werden? [j/N] :
antwort = STDIN.gets
if antwort =~ /[jJyY]/
  puts "OK, ich extrahiere #{file.chomp} aus #{item}"
  if File.exist?(file.chomp)
    ts = Time.now.to_i
    %x[/bin/tar --transform 's/$/#{$ts}-/' -xzf #{item}]
  else
    %x[/bin/tar xzf #{item} #{file}]
  end
end
end
end
end
end
end
```

2.6 Ajax

Ajax

Interaktion mit dem User durch:

Asynchronous JavaScript And XML

dynamische Webseiten



Neueinschreibungen

Nicht freigeschaltet

alle Benutzer (40)

Tagesg

Sie sind als Administrator eingeloggt

Nachname

Suchen

Nachname, Vorname

Geb.

E-Mail

Test, Lars (Kein Entitlement)

27.06.1978



dalars27@gmail.com

Lüttringhaus, Rahel (Kein
Entitlement)

08.03.1982



rahel.luettringhaus@library.ethz.ch

Garkeiner, Niemand (Kein
Entitlement)

01.01.2000



mcvsvenster@googlemail.com

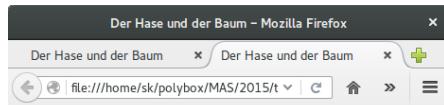
ein Benutzer mit verifizierter EMail–Adresse:

```
<div class="col-md-3">
  <a href="mailto:rahel.luettringhaus@library.ethz.ch" class="verified"><span class="glyphicon glyphicon-ok-sign">
</div>
```

...und einer mit einer unbekannten EMail–Adresse:

```
<div class="col-md-3">
  <span class="unknown"><span class="glyphicon glyphicon-question-sign">
</div>
```

2.7 Responsive Web



Der Hase und der Baum

Kapitel 1: Der Hase

Meister Lampe hoppelt über ein Feld.

Kapitel2: In der Werkstatt

Herr K. bestellt eine Knautschzone.

 Lorem ipsum dolor sit amet, consectetur adipisicing elit,
 labore et dolore magna aliqua. Ut enim ad minim veniam,
 laboris nisi ut aliquip ex ea commodo consequat. Duis au
 voluptate velit esse cillum dolore eu fugiat nulla pariatur.
 proident, sunt in culpa qui officia deserunt mollit anim id

So soll es sein:



Der Hase und der Baum

Kapitel 1: Der Hase

Meister Lampe hoppelt über ein Feld.

Kapitel2: In der Werkstatt

Herr K. bestellt eine Knautschzone.

 Lorem ipsum dolor sit amet, consectetur adipisicing
 elit, sed do eiusmod tempor incididunt ut labore et
 dolore magna aliqua. Ut enim ad minim veniam, quis
 nostrud exercitation ullamco laboris nisi ut aliquip ex
 ea commodo consequat. Duis aute irure dolor in
 reprehenderit in voluptate velit esse cillum dolore eu
 fugiat nulla pariatur. Excepteur sint occaecat cupidatat
 non proident, sunt in culpa qui officia deserunt mollit
 anim id est laborum.

3 Datenbanktechnologien I

Warum Datenbanken?

- geringer Speicherplatzbedarf
- gleichzeitiger Zugriff durch viele Nutzer

Da Speicher früher sehr teuer war, entwickelte man Strategien, um Speicherplatz zu sparen. Ein gutes Beispiel sind Normalisierungen bei relationalen Datenbanken.

3.1 Datenbanktypen

Welche Typen von Datenbanken gibt es?

- K/V– Stores
- relationale Datenbanken
- Spaltenorientierte Datenbanken
- Dokumentorientierte Datenbanken
- Graphendatenbanken

K/V– Stores

K/V– Stores sind — wie der Name schon sagt — schlichte Schlüssel / Wert– Speicher. Sie sind bei minimalem Speicherplatzbedarf sehr schnell, was sie für das Caching von Werten prädestiniert.

relationale Datenbanken

Nehmen wir zur Verdeutlichung eine Versicherung, die die Adressen ihrer Kunden in einer Datenbank speichern möchte. Es wird eine grosse Tabelle angelegt, je eine Zeile pro Kunde. Dabei stellt sich heraus, dass die Versicherung 1000 Kunden hat, die in der Hauptstrasse wohnen. Das bedeutet, dass wir 1000 Mal den Speicherplatz für das Wort “Hauptstrasse” benötigen.

relationale Datenbanken

id	Nachname	Vorname	Strasse	Stadt
1	Muster	Hans	Hauptstrasse	Zürich
2	Meier	Heinrich	Hauptstrasse	Zürich
3	Müller	Hubert	Hauptstrasse	Zürich
4	Schulze	Herbert	Hauptstrasse	Zürich

Wenn man nun die Strassen in eine extra Tabelle schreibt, eine Zeile pro Strasse, dann muss man in der Tabelle der Kunden nur die ID der Strasse hinterlegen. Wenn die Hauptstrasse z.B. die ID 1 hat, dann benötigen wir nur noch 1000 Mail den Speicherplatz für den Integer 1 — wesentlich weniger, als für den String “Hauptstrasse”. Analog kann man z.B. auch mit Städten verfahren.

relationale Datenbanken

id	Strasse
1	Hauptstrasse
2	Nebenstrasse
3	Seitenstrasse

relationale Datenbanken

id	Stadt
1	Zürich
2	Basel
3	Bern

relationale Datenbanken

id	Nachname	Vorname	Strassen_ID	Stadt_ID
1	Muster	Hans	1	1
2	Meier	Heinrich	1	1
3	Müller	Hubert	1	1
4	Schulze	Herbert	1	1

relationale Datenbanken

Eine relationale Datenbank dahingehend zu optimieren, dass es möglichst wenig Redundanzen gibt, nennt man „normalisieren“. Es gibt fünf Normalformen.

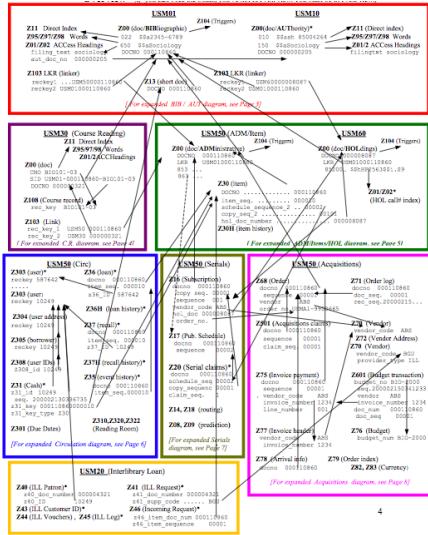


Abbildung 9: Relationen in der DB von Aleph

relationale Datenbanken

```
-rw-rw-r-- 1 sk sk 37K Dez 9 16:32 testdaten_full.csv  
-rw-rw-r-- 1 sk sk 20K Dez 9 17:02 testdaten_normalisiert.csv
```

Für die Datenbank-Spezialisten war es eine komplexe Aufgabe, Daten optimal zu normalisieren.

Die Normalisierung hat auch Nachteile. So liegen nicht alle Informationen in einer Tabelle, was Abfragen komplexer macht. Damit einher geht ein höherer Leistungsbedarf des Servers.

relationale Datenbanken

relationale Datenbanken

```
select substr(z103_rec_key,6,9) || 'EHO60'  
from eho60.z103  
where substr(z103_lkr_text_n,1,3) = 'E04'  
and z103_lkr_library = 'EBI01'  
and substr(z103_rec_key,1,5) = 'EHO60'  
;
```

Ein Nachteil von relationalen Datenbanken ist die feste Grösse und Anzahl von Feldern. Wenn ich beispielsweise für die Tabelle der Kunden die Spalten

Vorname Nachname Strasse Hausnummer Postleitzahl Ort

festgelegt habe, und die Versicherung sich entscheidet, ein internationales Geschäft aufzubauen, ist es einiger Aufwand, die Spalte "Land" hinzu zu fügen.

relationale Datenbanken

Das Feld für Inventarnummern darf in Aleph nicht mehr als 20 Zeichen haben.

relationale Datenbanken

UTF8-Codierungszeichen zaehlen einzeln, auch wenn daraus ein einziges Unicode-Zeichen entsteht.

Ein Beispiel dafuer ist das "Ä", das im Inventarnummernfeld zwei VARCHAR2 Zeichen aufbraucht, weil es aus 0xc3 und 0x84 besteht.

Ich habe nicht geschaut, ob es noch weitere solche Fälle gibt. [Mathias Weyland]

relationale Datenbanken

3.2 dokumentorientierte Datenbanken

```
{ "_id": 1,  
  "Vorname": "Sven",  
  "Nachname": "Koesling",  
  "email": "sven.koesling@library.ethz.ch"  
}
```



Abbildung 10: <http://houseofbrick.com/the-oracle-parking-garage/>

ein weiteres Dokument in derselben collection

```
{ "_id": 2,  
  "Vorname": "Harry",  
  "Nachname": "Hirsch",  
  "email": "hh@lustich.com"  
  "Beruf": "Reporter"  
}
```

noch ein Dokument in derselben collection

```
{ "_id": 3,  
  "Vorname": "Martha",  
  "Nachname": "Graham",  
  "Beruf": ["Tänzerin", "Choreographin"]  
}
```

und noch ein weiteres Dokument in derselben collection

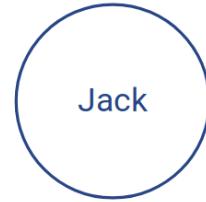
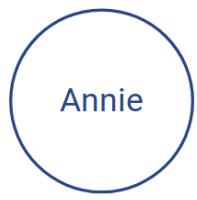
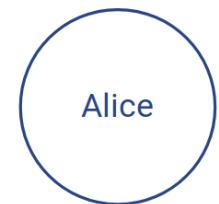
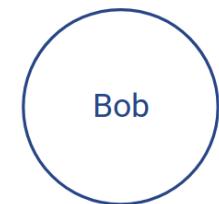
```
{ "_id": 4,  
  "Vorname": "Niemand",  
  "Nachname": "Nixda",  
  "email": "nn@none.org"  
  "Hund": {  
    "Name": "Nero",  
    "Geschlecht": "male"  
  }  
}
```

Wenn wir an Dokumente denken, kommen wir schnell auf die „Panama–Papers“.

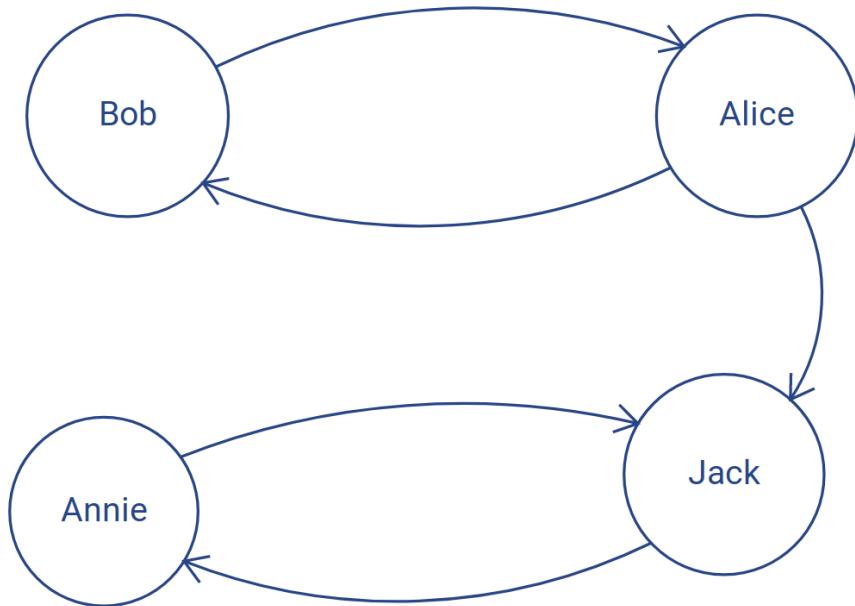
Graphendatenbanken

Graphendatenbanken sind quasi Knoten und Kanten mit Attributen.

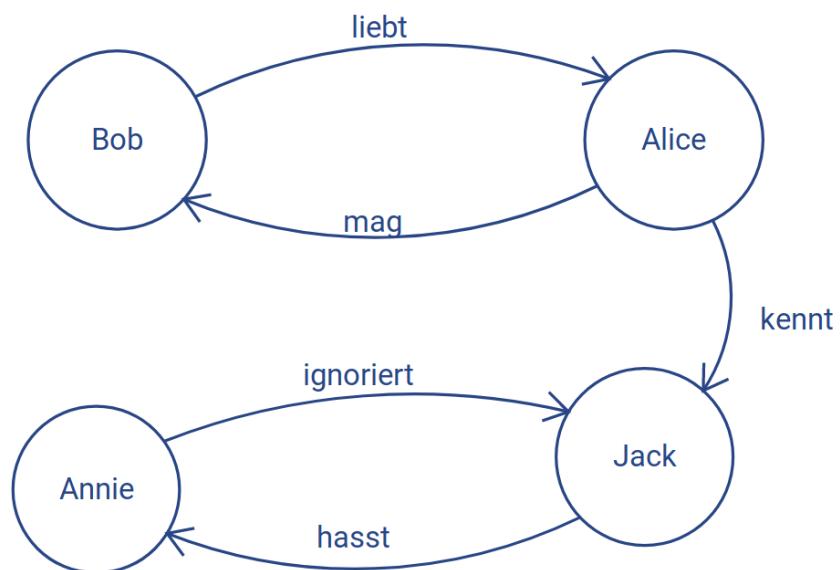
ein paar Knoten



Knoten und Beziehungen



Knoten, Beziehungen und Attribute



3.3 Exkurs: Indexer

Abbildung 11: Suche nach “Landquart” in einem aktuellen Bibliothekskatalog (420.000 Dokumente): 1072 Treffer – ohne Facetten: completed in 2845ms

The screenshot shows a library catalog interface with the following details:

- Header:** Includes the logo of the Swiss Confederation (a heraldic shield), links for "Ihr Bibliothekskonto", "Neustart", "Hilfe", "Suche", "Ergebnisliste", "Suchverlauf", "Ihre Literaturliste", "Drucken", "Teilkataloge BGR", and "Andere Kataloge".
- Search Bar:** Shows the search term "Stichwort= Landquart"; Sortiert nach: Jahr, dann Autor. It also indicates "Sätze 1 - 10 von 1072".
- Buttons:** "Gehe zu #", "Vorige Seite", and "Nächste Seite".
- Table:** The search results table has columns for #, Autor, Titel, Jahr, Bestand / davon ausgeliehen, and Photo. The results are as follows:

#	Autor	Titel	Jahr	Bestand / davon ausgeliehen	Photo
1	Gantenbein, Köbi	Das Tor zu Graubünden	2010		
2	Radio- und Fernsehgesellschaft der Deutschen und der Rätoromanischen Schweiz (Zürich)	[am Bündner Ländlerkapellen-Treffen in Landquart] [Ton]	2009	CHUR-Kantonsbibliothek(2 / 0)	
3	Schmid, Christian	Die Bahn fährt durch den Unterschnitt	2009		
4	Uffer, Rita	Luzia Vonmoos-Simonett, Igis [Ton]	2009	CHUR-Kantonsbibliothek(1 / 0)	
5	Vischer, Daniel	Der Bündner Theologe Lucius Pol und die Landquarkorrektion	2009	CHUR-Kantonsbibliothek(1 / 0) CHUR-Staatsarchiv(1 / 0)	
6	ÖKK (Landquart)	ÖKK dossier per aziende	2009		
7	ÖKK (Landquart)	ÖKK Dossier für Unternehmen	2009	CHUR-Kantonsbibliothek	
8		Alles, was in der Ostschweiz gehegt und gepflegt wird. [Bildmaterial]	2009		
9	Brunold-Bigler, Ursula	Arbeiterschaft und Kapuziner	2008	CHUR-Kantonsbibliothek(2 / 0) CHUR-Staatsarchiv(1 / 0) CHUR-Theologische Hochschule(1 / 0)	
10	Herrmann, Arno	Eine Kirche, die redet ...	2008	CHUR-Staatsarchiv(1 / 0)	

Abbildung 12: Die gleiche Suche mit einem Indexer: 1.700 Treffer mit drei Facetten: completed in 792ms

Bibliographisches Portal Graubünden
eine Seite Ihrer Kantonsbibliothek



Regionaler Einstieg Bündner Personen Autorenregister Epochen Bibliographie 2009
→ Albula Bernina Hinterhein Imboden Inn Landquart Maloja Moesa Plessur Prättigau / Davos Surseva Feedback

Ergebnis für "Landquart"
(1700 Treffer)

« Previous 1 2 3 4 5 6 7 8 9 ... 113 114 Next »

Kreis
Fünf Dörfer
Maienfeld

Autoren (47)
Ferdmann, Jules (2)
Hartmann, Reto (4)
Härtli, Peider (2)

Urheber (31)
Elmer-Cantieni, Anna Maria (2)
Mooser, Ueli (3)
Tschümperlin, Beat (8)
Weber, Johann (2)

Schlagworte (26)
Eisenbahnlinie : Landquart-Davos (7)
Eisenbahnlinie : Landquart-Davos-Filisur (2)
Graubünden (3)
Iglis (3)
Landquart (29)
Landquart (Fluss) (2)

"...und wenn nötig, dieselben zu allem Fleiss anhalten"
Autor / Urheber: Bartli, Hans
Vorkommen: Dorfchronik Jenins..., 11(1998); S. 26-30, 11/1998/26-30, 000150885
ISBN: 0
geograph. Schlagwort: Jenins
zum [Katalog](#)

"Alligator Malans" vor dem Unihockey-Eurocup - Start
Vorkommen: Bündner Zeitung, Chur, 1997/12/30, p. 19, 1997/12/30/19, 000048258
ISBN: 0
zum [Katalog](#)

"Ausgelöffelt" : "Hirschen" in Maienfeld
Vorkommen: Südoestschweiz, Chur, 2002/05/17, Apropos, 2002/05/17, 000167664
ISBN: 0
zum [Katalog](#)

"Bock auf Rock!"
ISBN: 0
geograph. Schlagwort: Zizers
zum [Katalog](#)

"Cucci Travel" Reisebüro neu in Landquart
Vorkommen: Bündner Zeitung, Chur, 1997/3/1, p. 4, 1997/3/1/4, 000048258
ISBN: 0
zum [Katalog](#)

"Das Repertoire möglichst breithalten"

4 Internettechnologien II: von interaktiven Webseiten zu WebApps in der Cloud

4.1 Das DOM

Eine Webseite gliedert sich — wie „normale“ Textdokumente auch — in verschiedene Abschnitte wie z.B. Überschriften, Absätze, es gibt Listen usw. Dazu kommen Bereiche im Layout wie z.B. Der Kopf, Randnotizen oder das Menü. All diese Elemente sind im DOM (document object model) beschrieben und darüber ansprechbar. Das DOM ist eine Spezifikation eines API, einer Programmierschnittstelle, mit der man html—Elemente ansprechen kann. Damit das funktioniert, müssen die Elemente im html—Dokument entsprechend ausgezeichnet sein.

Beispiele für Elemente im html—Dokument

- Überschriften : <**h1**>Überschrift</**h1**>
- Absätze : <**p**>Absatz</**p**>
- Tabellen : <**table**><**tr**><**td**>Zelle</**td**></**tr**></**table**>

Eine Unart war, die Dokumente mit Tabellen zu „gestalten“ oder besser: zu verunstalten. Die Designer hatten erkannt, dass man mit Tabellen Text auf einer Webseite gut positionieren kann. Damit wird die Seite jedoch eher statisch, ist auf unterschiedlichen Monitorgrößen nicht mehr gleich gut lesbar, und unübersichtlich im Quelltext und damit schlecht zu pflegen. Diese Unart ist — bis auf beim OPAC von Aleph – inzwischen aus dem Netz verschwunden.

die Formatierung im alten OPAC

 Benutzungskonto | Einstellungen | Geführte Suche | Hilfe | Abmelden
Andere Kataloge E | F
Ergebnisliste | Suchverlauf | Liste | Korb
Suche | Erweiterte Suche | Expertensuche | Blättern >Zeitschriften >Signaturen >Sachbegriffe >ISBN

Stichwortsuche:
Stichwörter aus allen Feldern
Titel enthält Stichwörter...
Autor/Körperschaft enthält Stichwörter...
Serie/Reihe enthält Stichwörter...
 OK

Beispiele:
frisch andora
frisch andorra
anthropolog? china
(herz or cardio?) and therap?

Blättern in einer Liste:
Autor/Körperschaft beginnt mit...
Titel beginnt mit...
Zeitschriftentitel beginnt mit...
Sachbegriff NEBIS (D,E,F) beginnt mit...
 OK

Beispiele:
frisch m
frisch max
schweizerische afrika gesellschaft
conference on acid rain
Tipp: bei Autoren Nachname zuerst

Informationen zum Katalog
Ferien- und Inventurschliessungen

Feedback/Anregungen - FAQ

die Formatierung im alten OPAC (Quelltext)

```
<table style="margin-top:0.4em" width="100%" border="0" cellspacing="0" cellpadding="0">
<tr class="topbar">
<td valign="middle" width="120" rowspan="3">
<a href="#">

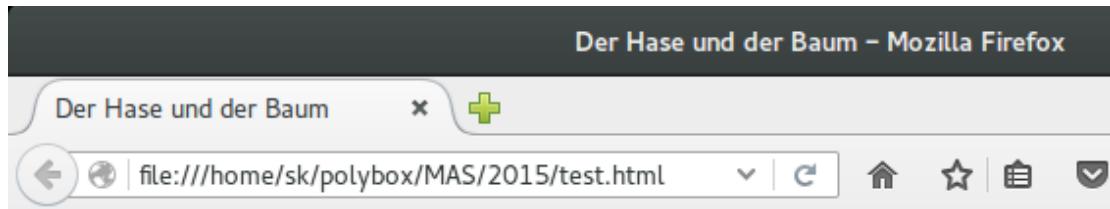
</a>
</td>
<td valign="middle" align="left" nowrap>
<a href="#" class="konto">Benutzungskonto</a> |
<a href="#" class="black">Einstellungen</a> |
<a href="#" class="black">Gef&uuml;hrte Suche</a> |
...
</td>
<td valign="middle" align="right" nowrap>
<a href="#" class="black">Andere Kataloge</a>
...
</td>
</tr>
</table>
```

4.2 Das DOM für Gestaltungszwecke verwenden

Dennoch bleibt natürlich der Wunsch, den hoffentlich wertvollen Inhalt auch ansprechend gestalten zu können. Dazu lassen sich die Elemente des DOMs mit Formationsinformationen versehen.

unsere Webseite bis jetzt

```
<html>
  <head>
    <meta charset="utf-8" />
    <title>Der Hase und der Baum</title>
  </head>
  <body>
    <h1>Der Hase und der Baum</h1>
    <h2>Kapitel 1: Der Hase</h2>
    <p>Meister Lampe hoppelt über ein Feld.</p>
    <h2>Kapitel2: In der Werkstatt</h2>
    <p>Herr K. bestellt eine Knautschzone.</p>
  </body>
</html>
```



Der Hase und der Baum

Kapitel 1: Der Hase

Meister Lampe hoppelt über ein Feld.

Kapitel2: In der Werkstatt

Herr K. bestellt eine Knautschzone.

Eine Überschrift wird mit <h1> (bzw. h2,h3 usw.) ausgezeichnet. Dieser Auszeichnung kann man z.B. Formatierungsinformationen übergeben:

unsere Webseite bis jetzt

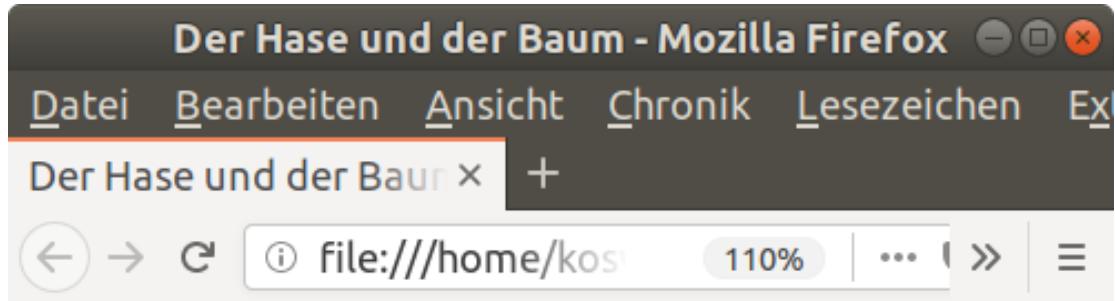
Die Anweisung

```
"
```

lässt das entsprechende Element in grau erscheinen.

erste eigene Formatierung

```
<html>
  <head>
    <meta charset="utf-8" />
    <title>Der Hase und der Baum</title>
  </head>
  <body>
    <h1 style="color: grey;">Der Hase und der Baum</h1>
    <h2>Kapitel 1: Der Hase</h2>
    <p>Meister Lampe hoppelt über ein Feld.</p>
    <h2>Kapitel2: In der Werkstatt</h2>
    <p>Herr K. bestellt eine Knautschzone.</p>
  </body>
</html>
```



Der Hase und der Baum

Kapitel 1: Der Hase

Meister Lampe hoppelt über ein Feld.

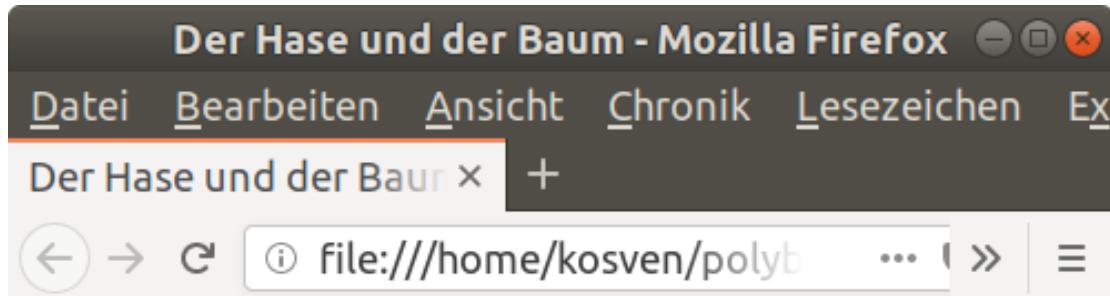
Kapitel2: In der Werkstatt

Herr K. bestellt eine Knautschzone.

Will man nun alle Überschriften in grau setzen, wird es etwas umständlich, wenn man die Informationen in jedes Element einzeln setzen müsste. Stattdessen kann man eine Formatierung, die für alle Elemente eines Typs gelten soll, zentral definieren:

Formatierung im head

```
<html>
  <head>
    <meta charset="utf-8" />
    <title>Der Hase und der Baum</title>
    <style>
      h1,h2 { font-family: sans-serif; }
      p { color: gray; }
    </style>
  </head>
  <body>
    <h1>Der Hase und der Baum</h1>
    <h2>Kapitel 1: Der Hase</h2>
    <p>Meister Lampe hoppelt über ein Feld.</p>
    <h2>Kapitel2: In der Werkstatt</h2>
    <p>Herr K. bestellt eine Knautschzone.</p>
  </body>
</html>
```



Der Hase und der Baum

Kapitel 1: Der Hase

Meister Lampe hoppelt über ein Feld.

Kapitel2: In der Werkstatt

Herr K. bestellt eine Knautschzone.

Da einzelne Elemente mehrfach vorkommen können, kann man sie zur eindeutigen Identifizierung mit einer ID versehen. Außerdem kann man auch Klassen mit Formatierungs-informationen definieren, um so verschiedenen Elementen eine konsistente Formatierung

zuweisen zu können. Ein kompletter Webauftritt besteht nun selten nur aus einer Seite. Will man über mehrere Seiten ein konsistentes Layout erzielen, müsste man die Formatierungsinformationen im Header in jede Seite kopieren.

EXKURS „DRY“

Das DRY–Prinzip verbietet es, die Formatierungsinformationen in jeder einzelnen Seite redundant vorzuhalten und jeweils einzeln pflegen zu müssen. Stattdessen lagert man sie in eine eigene Datei aus, die als stylesheet (css) im head der Seite verlinkt wird.

externes Stylesheet

```
<head>
  <link rel="stylesheet" media="all" href="/assets/application-
    e77e74d6189491bd233e87f81af0db0315d8ed4a4e7b0070d13b6180200
    ce1a7.css">
</head>
```

Auszug aus dem css von libraries.ch

```
a, label, .btn {
  font-size: 1em
}

a {
  color: #20758c
}
...
body {
  font-size: 1em
  overflow-y: scroll
}
...
footer h3 {
  font-size: 1.2em
  margin-top: 0
  margin-bottom: 0
  height: 1.4em
}
```

Da wir aber in unserem Beispiel nur eine einzelne Seite haben, lasse ich die Formatierungsinformationen ausnahmsweise im head.

4.3 Das DOM für JavaScript-Funktionen verwenden

Man kann nun das DOM auch verwenden, um Elemente in der Seite vom Client, also vom Browser des Nutzers dynamisch anpassen zu lassen. Das probieren wir im Folgenden mal mit Hilfe der Entwicklertools im FireFox. Doch zunächst geben wir einer Tabellenzeile eine id.

id vergeben

```
<table>
  <tr>
    <th>Kapitel</th><th>Titel</th>
  </tr>
  <tr>
    <td>1</td><td>Der Hase</td>
  </tr>
  <tr id="kapitel2">
    <td>2</td><td>In der Werkstatt</td>
  </tr>
</table>
```

Element mit JavaScript ausblenden

Der JavaScript-Befehl

```
document.getElementById("kapitel2").style.display = 'none';
```

blendet das Element mit der id „kapitel2“ aus.

Um den Befehl eingeben zu können, müssen wir die Entwicklertools öffnen (Strg+Umschalt+I).

Die Entwicklertools

The screenshot shows the developer tools interface in a browser window. The title bar includes tabs like Inspekt., Konso, Debugg., etc. The main area displays an

Inhaltsverzeichnis

 and a table with two rows. The table structure is shown in the DOM tree on the left. On the right, the 'Box-Modell' panel is open, showing a visual representation of the element's layout. The element has an outer yellow border (Außenabstand 8px), a dark grey inner border (Rand 0px), and a light blue center (Innenabstand 0px). The overall dimensions are 931x132.433. Below the visual representation, it says 'Keine CSS-Eigenschaften gefunden.' (No CSS properties found.).

Die Entwickertools : Konsole



Natürlich lässt man das nicht den Endbenutzer von Hand machen. Stattdessen wird ein entsprechendes JavaScript in die Seite eingebaut, das beispielsweise durch Klick auf ein Element ausgeführt wird. Im Beispiel verwenden wir einen einfachen Textlink:

Element mit script ausblenden

```
<tr>
  <td colspan="2"><a href="#" onclick="verstecken('kapitel2'); return false;">
    Kapitel 2 verstecken</a></td>
</tr>
<tr id="kapitel2" style="display: table-row;">
  <td>2</td><td>In der Werkstatt</td>
</tr>
</table>

<script type="text/javascript">
  function verstecken(id) {
    document.getElementById(id).style.display = 'none';
  }
</script>
```

Und so sieht es im Browser aus:

Element mit script ein- und ausblenden

```
<tr>
  <td colspan="2"><a href="#" onclick="einausblenden('kapitel2'); return false;">
    zeige</a></td>
</tr>
<tr id="kapitel2" style="display: none;">
  <td>2</td><td>In der Werkstatt</td>
</tr>
</table>

<script type="text/javascript">
  function einausblenden(id) {
    var meinElement = document.getElementById(id);
    if (meinElement.style.display === "none") {
      meinElement.style.display = "table-row";
    } else {
      meinElement.style.display = "none";
    }
  }
</script>
```

Und so sieht es im Browser aus:

Übung / Spiel: Wie verhindern wir das Springen in der Tabellenkopfzeile? th text-align: left;

Viele Probleme kommen immer wieder in ähnlicher Form vor. Man vereinfacht sich das, indem man Elemente so programmiert, dass sie wieder verwendbar sind. Ein guter Punkt, der sich mit der Entwicklung der objektorientierten Programmierung stark verbreitet hat.

Auf entsprechenden Plattformen werden Code-Schnipsel, Lösungen und komplette Programme ausgetauscht. (Github, Gitlab, Bitbucket...)

4.4 JavaScript–Libraries und –FrameWorks

Zurück zur Webseite: Auch hier kommen viele Elemente immer wieder vor und werden von vielen Entwicklern benötigt. Das Ein- und Ausblenden von Elementen ist eine häufig gestellte Anforderung. Damit nun nicht jeder den Javascriptcode wieder schreiben (oder von Github kopieren) muss, gibt es Leute, die Lösungen zu ganzen Sammlungen zu einem Thema zusammen stellen, die sogenannten Bibliotheken bzw. „Libraries“.

Noch umfangreicher als Libraries sind FrameWorks, bei denen verschiedene Libraries, unterstützende Software und sogar Skriptsprachen zusammengepackt werden, um das Softwaredevelopment im entsprechenden Gebiet erheblich zu vereinfachen.

Die Grenzen sind fliessend und nicht eindeutig zu bestimmen. Man kann sicher sagen, dass FrameWorks wiederverwertbare Strukturen mit komplexen Anwendungen zur Programmierung und ein abstraktes Design bereit stellen, wohingegen eher Libraries Funktionssammlungen zu konkreten Aufgaben sind.

zwei verbreitete JavaScript Libraries

- jquery / jquery UI
- bootstrap

zwei verbreitete JavaScript FrameWorks

- AngularJS / Angular 2
- ReactJS

Libraries und FrameWorks bindet man in den head seiner Webseite ein. Der Nachteil ist, dass jeder Nutzer der Webseite sich einmal die komplette Library bzw. das komplette FrameWork (im Hintergrund) herunterladen muss, auch wenn man von tausenden Funktionen nur zwei benötigt.

Es gibt verschiedene...

Möglichkeiten, JS einzubinden:

- eine Version direkt auf dem eigenen Server zur Verfügung stellen
- eine bestimmte Version als Link direkt einbinden
- eine Minimalversion einbinden und benötigte Funktionen dynamisch nachladen (problematisch, wenn das Gerät nicht dauerhaft online ist)

Beginnen wir mit einem Beispiel, bei dem wir die Library „jquery“ direkt über einen Link zum Anbieter in der jeweils neuesten Version einbinden.

die Einbindung von jquery

```
<html>
  <head>
    <meta charset="utf-8" />
    <script src="https://code.jquery.com/jquery-latest.js"></script>
  ...

```

Der Link im Beispiel oben zeigt immer auf die aktuellste Version („jquery-latest.js“), was problematisch sein kann, falls es in einer neueren Version Änderungen gegeben hat, die nicht mit dem Code bzw. den Skripts in der Webseite kompatibel sind. Deswegen kann es sinnvoll sein, stattdessen auf eine konkrete Version zu verlinken bzw. diese direkt auf dem eigenen Webserver zur Verfügung zu stellen.

jquery erlaubt nun ein viel einfacheres Ansteuern der DOM–Elemente, als pures JavaScript. Auch sind — wie beschrieben — oft benötigte Funktionen gleich integriert. So will man oftmals ein Element nicht nur einmal einblenden bzw. ausblenden, sondern wiederkehrend ein– und ausblenden. jquery stellt hierfür die Funktion „toggle“ zur Verfügung.

Das Ansteuern der Elemente ist viel einfacher

Der Befehl

```
$("#kapitel2").toggle();
```

blendet das Element mit der id „kapitel2“ ein und aus.

→ Test im Browser

ein– / ausblenden mit jquery

```
...
<tr>
  <td colspan="2"><a href="#" id="schalter">zeige</a></td>
</tr>
<tr id="kapitel2" style="display: none;">
  <td>2</td><td>In der Werkstatt</td>
</tr>
</table>

<script type="text/javascript">
$(document).ready(function() {
  $("#schalter").click(function() {
    $("#kapitel2").toggle();
  });
});
</script>
...
```

Userinterface mit jquery UI

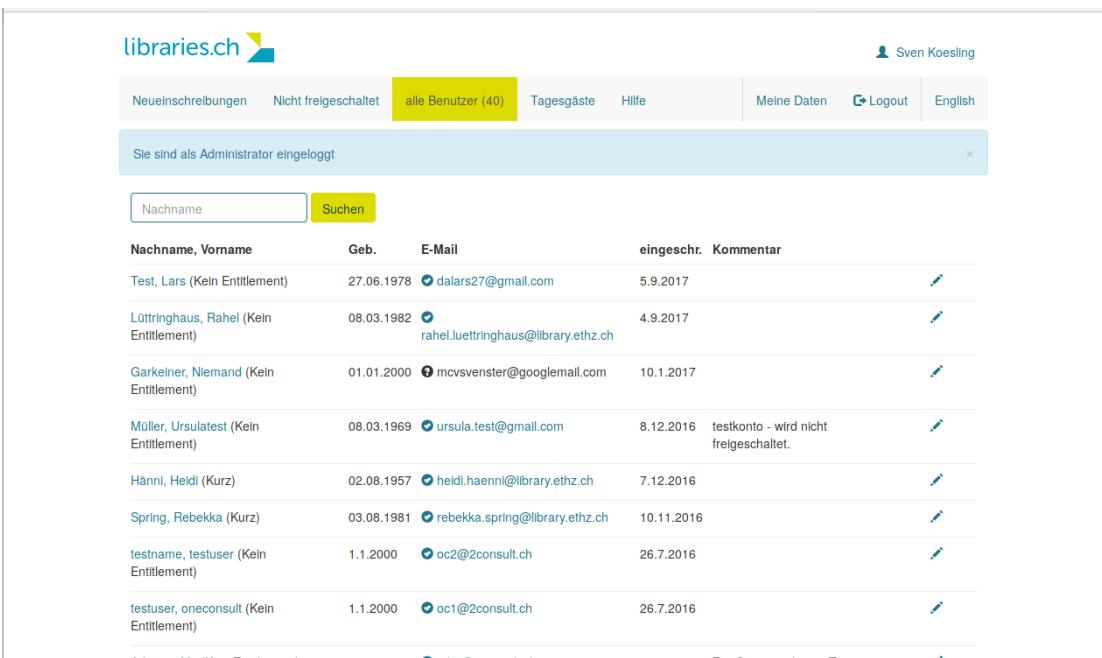
Mit der Bibliothek „jquery UI“ lassen sich schnell einheitliche Oberflächen bauen. Hier ein Button:

```
<head>
  <meta charset="utf-8" />
  <script src="https://code.jquery.com/jquery-latest.js"></script>
  <link rel="stylesheet" href="https://ajax.googleapis.com/ajax/libs/jqueryui/1.12.1/themes/smoothness/jquery-ui.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jqueryui/1.12.1/jquery-ui.min.js"></script>
...
<body>
...
  <button>zeige Kapitel 2</button>
```

bootstrap statt Tabellenlayout

```
<div class="row">
  <div class="col-md-3">...</div>
  <div class="col-md-1">...</div>
  <div class="col-md-3">...</div>
  <div class="col-md-1">...</div>
  <div class="col-md-3">...</div>
  <div class="col-md-1">...</div>
</div>
```

auf einem normalen Monitor



The screenshot shows a web application interface for managing user profiles. At the top, there is a navigation bar with links for 'Neueinschreibungen', 'Nicht freigeschaltet', 'alle Benutzer (40)', 'Tagesgäste', 'Hilfe', 'Meine Daten', 'Logout', and 'English'. A user profile picture and the name 'Sven Koesling' are also visible. A blue banner at the top indicates that the user is logged in as an administrator. Below the banner, there is a search bar with fields for 'Nachname' and 'Suchen'. The main content area displays a table of user data:

Nachname, Vorname	Geb.	E-Mail	eingeschr.	Kommentar
Test, Lars (Kein Entitlement)	27.06.1978	dalars27@gmail.com	5.9.2017	edit
Lütringhaus, Rahel (Kein Entitlement)	08.03.1982	rahel.luettringhaus@library.ethz.ch	4.9.2017	edit
Garkeiner, Niemand (Kein Entitlement)	01.01.2000	mcvsvenster@googlemail.com	10.1.2017	edit
Müller, Ursulatest (Kein Entitlement)	08.03.1969	ursula.test@gmail.com	8.12.2016	testkonto - wird nicht freigeschaltet.
Hänni, Heidi (Kurz)	02.08.1957	heidi.haenni@library.ethz.ch	7.12.2016	edit
Spring, Rebekka (Kurz)	03.08.1981	rebekka.spring@library.ethz.ch	10.11.2016	edit
testname, testuser (Kein Entitlement)	1.1.2000	oc2@2consult.ch	26.7.2016	edit
testuser, oneconsult (Kein Entitlement)	1.1.2000	oc1@2consult.ch	26.7.2016	edit

auf einem mobilen Endgerät

The screenshot shows a web application interface for 'libraries.ch'. At the top, there is a logo and the name 'Sven Koesling' with a profile icon. A blue banner displays the message 'Sie sind als Administrator eingeloggt' (You are logged in as administrator) with a close button. Below this is a search bar with the placeholder 'Nachname' (Last name) and a yellow 'Suchen' (Search) button. The main content area lists three users with their details:

- Test, Lars (Kein Entitlement)
27.06.1978
dalars27@gmail.com
5.9.2017
- Lüttringhaus, Rahel (Kein Entitlement)
08.03.1982
rahel.luettringhaus@library.ethz.ch
4.9.2017
- Garkiner, Niemand (Kein Entitlement)
01.01.2000
mcvsvenster@qooalemail.com

Each user entry includes a small edit icon.

Die vielfältigen UI–Elemente und umfangreichen Funktionen der aktuellen JavaScript–Bibliotheken und –FrameWorks ermöglichen nicht nur komplexe Interaktionen des Benutzers mit der Webseite. In Kombination mit Ajax lassen sich echte Webapplikationen realisieren, die gewöhnlicher Desktop–Software nichts nachsteht. So werden neue Geschäftsmodelle wie Software as a Service erst möglich.

4.5 Cloud Services

der Einsatz von JavaScript ermöglicht neue Services

- SaaS
- IaaS
- PaaS

Beispiel für Saas: iCloud

The screenshot shows a document in iCloud Pages. The title is "Fortschrittsbericht August 2015". The document contains several sections:

- Sitzungsnummer:** 16
- Datum:** 27.08.2015, 14:00 - 15:15 Uhr, HG H 59.2
- Anwesende:** Judith Bissegger, Barbara Wittwer, Marianne Wolff
- Entschuldigt:** Erica Pfister, Madeleine Vollmin

Ergebnisse

- Nachbearbeitung E-Book Anreicherung: Analyse der angereicherten Daten und notwendige Bereinigungen, sowie Erstellung von SYS-Nr.-Listen für MDM.
- Inhalte der Felder 008/906/907 genauer analysiert und Gruppen von fehlerhaften Codes korrigiert.
- Prüf-Routinen und Auswahllisten von in Aleph hinterlegten Werten (CtlF8-Listen) auf EB105 in Aleph Test eingespielt und geprüft. Arbeit wird noch fortgesetzt.
- Safari E-Books (2. Einspielung mit 7000 Titeln) analysiert und maschinell aufgearbeitet. Einzelne Aufnahmen wurden manuell angepasst und mit Verknüpfungen zu Serien und Nameinträgen ergänzt. Die Aufnahmen konnten damit auf ein ausreichendes Level gebracht werden, dass sie im Katalog keine weitere Auferarbeitung durch MDM benötigen. Der Code-Zusatz „MDM“ wurde daher gestrichen.
- E-Book Aufnahmen wurden im Hinblick auf RDA analysiert und ein Kriterium wurde gesucht, anhand dessen eine E-Book Aufnahme nach RDA erkannt werden kann. Dies ist nötig für künftige E-Book Einspielungen und den Abgleich mit Aleph.

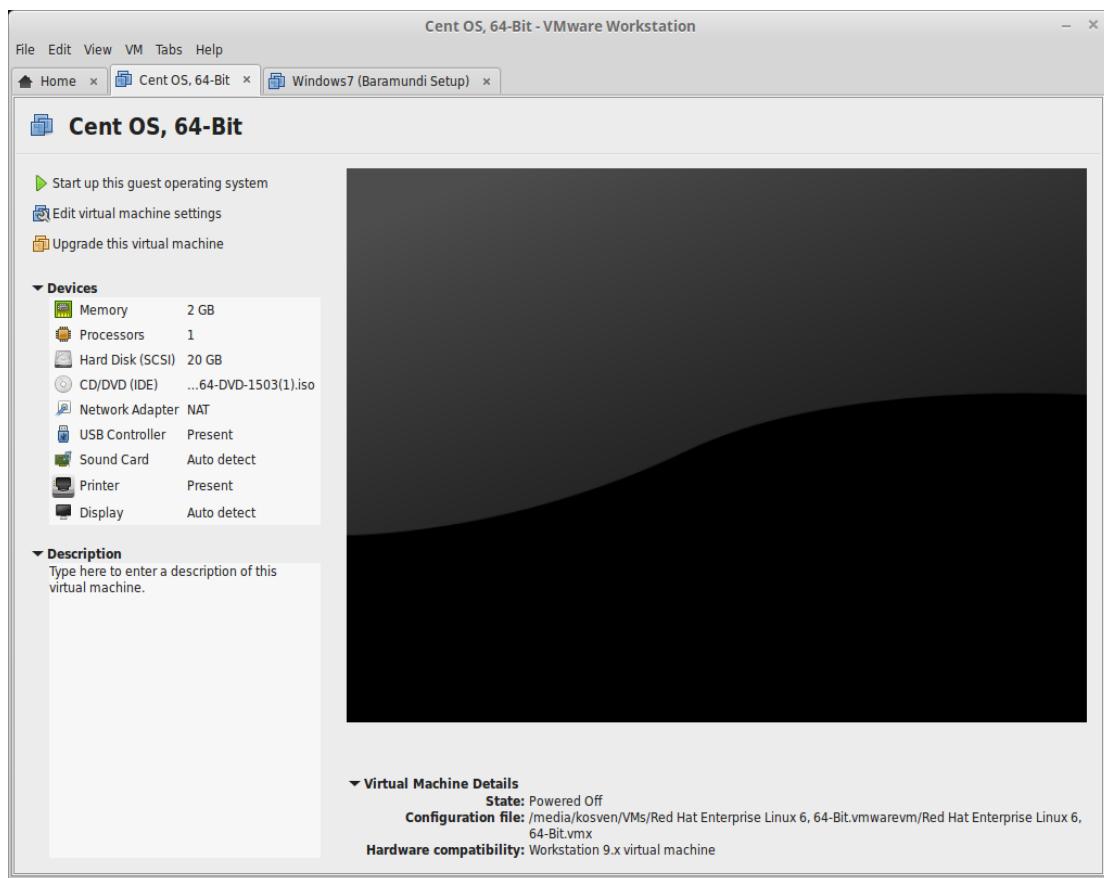
Erkenntnisse

- Arbeiten, die mit der Einführung von RDA anfallen, eignen sich hervorragend für das MDM-Team. Das bisher erlangte Wissen ist notwendig für bestimmte Arbeiten, wie z.B. Datenanalysen, Angleichungsstrategien, Expertenabfrage, sowie den allg. Umgang mit GUI. Ebenso ist das Datenflussdiagramm auch für die Einführung der RDA ein Gewinn, da es aufzeigt, welche Systeme Daten aus Aleph beziehen und daher auch von der Umstellung auf RDA betroffen sind.
- Die erstellte Liste mit KoFoFo-Beschlüssen zu den einzelnen MARC-Feldern ist eine hervorragende Hilfe für die Konfiguration der Aleph Tabellen. Damit kann sichergestellt werden, dass die Konfigurationen alle KoFoFo-Beschlüsse berücksichtigen und auf dem neuesten Stand sind.
https://projekte.ethbib.ethz.ch/rda/bit/Dokumente/AP2.1.1,_Änderungen_MARC_.pdf

The right side of the screen shows a sidebar with various text styling options:

- Absatzstil: Normal
- Schrift: Helvetica, Normal, 8 pt
- Ausrichtung: Left, Center, Right, Justify
- Zellenabstand: 1
- Absatzabstand: Vor dem Absatz: 0 pt, Nach dem Absatz: 0 pt
- Listen & Zeichen: Ohne
- Einrücken: Left, Center, Right, Justify

Beispiel für IaaS



Beispiel für Paas

The screenshot shows the Primo application's overview page. At the top left is a green circular icon with a white checkmark. To its right, under the heading "Health", it says "Green" and has a "Causes" link. In the center, under "Running Version", it shows "app-6eed-151222_104543" and a "Upload and Deploy" button. To the right, there is a "Ruby" logo with a stylized "R" and a "Configuration" link. Below the configuration link, it says "64bit Amazon Linux 2015.09 v2.0.4 running Ruby 2.2 (Passenger Standalone)" and a "Change" button. At the bottom left, there is a table titled "Recent Events" with columns for "Time", "Type", and "Details". The table contains two rows: one for an INFO event at 2016-01-03 05:57:41 UTC+0100 stating "Environment health has transitioned from RED to GREEN", and another for a WARN event at 2016-01-03 05:56:50 UTC+0100 stating "Environment health has transitioned from GREEN to RED". At the top right of the page is a "Refresh" button.

Time	Type	Details
2016-01-03 05:57:41 UTC+0100	INFO	Environment health has transitioned from RED to GREEN
2016-01-03 05:56:50 UTC+0100	WARN	Environment health has transitioned from GREEN to RED

4.6 Das DOM für automatisiertes Testen von WebApplikationen verwenden

Nachdem wir das DOM verwendet haben, um vielfältige Funktionen auf einer Webseite zu realisieren, können wir es auch benutzen, um diese Funktionen automatisch vom Computer testen zu lassen.

Beispielsweise kann man Elemente einer Webseite mit Skriptsprachen auf den korrekten Inhalt prüfen.

Mittels Ruby und den FrameWorks „Cucumber“ und „Capybara“ testen wir nach jedem Update von Primo, ob die Features noch so funktionieren, wie wir es geplant haben:

automatisierte Tests von Primo

- Wir prüfen nach einem Update, ob sich die Suchalgorithmen verändert haben, indem wir die Trefferzahl zwischen den beiden Systemen mit identischem Datenstand vergleichen.
- Der Date-Slider hat in der Vergangenheit Probleme gemacht. Wir checken, ob er plausible Ranges liefert.
- Wir testen, ob nicht-lateinische Schriftzeichen gefunden werden.

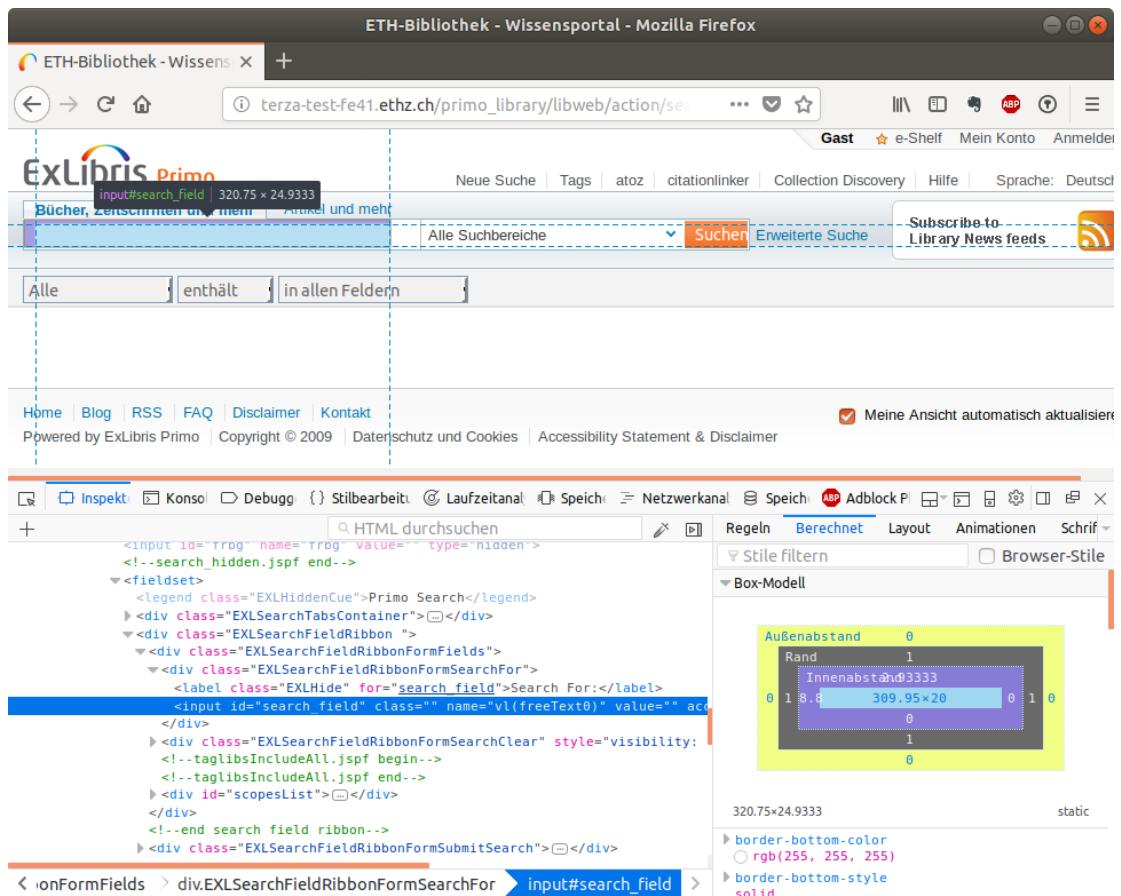
Das Gute an dem verwendeten Framework ist, dass die Anforderungen nahezu natürlichsprachig formuliert werden. So stellen wir sicher, dass IT und „normale“ Menschen vom Gleichen sprechen.

Anforderung in Gherkin formuliert

Szenario: Eine Suche ergibt auf den beiden Prod Systemen eine ähnliche Anzahl Treffer
Wenn ich die Seite "http://terza-prod1-fe41.ethz.ch/primo-explore/search?vid=DADS&sortby=rank&lang=de_DE" aufrufe,
Und ich in den Suchschlitz "Wald" eingebe,
Und die Anzahl der Treffer nehme
Und dann die Seite "http://terza-prod2-fe41.ethz.ch/primo-explore/search?vid=DADS&sortby=rank&lang=de_DE" aufrufe,
Wenn ich in den Suchschlitz den Suchbegriff "Wald" eingebe,
Und dort die Anzahl der Treffer nehme
Dann sollten die Treffermengen ähnlich, d.h. die Abweichung unter 1%, sein.

Schauen wir uns dazu eine Seite im UI von Primo an. Wir wollen in den Suchschlitz das Wort „Wald“ eingeben.

Der Suchschlitz im alten UI



The screenshot shows a Firefox browser window with the ETH-Bibliothek Wissensportal URL. The page displays the ExLibris Primo search interface. A search field containing "Wald" is highlighted with a blue border and has dimensions of 320.75 x 24.9333. The Firebug developer tool is open, showing the HTML structure of the search form. The search input field is selected in the DOM tree. The Firebug panel shows the element's position, style, and computed styles.

```
<input id="search_field" name="search_field" value="" type="text">
```

The Firebug panel shows the following computed styles for the search input field:

- Außenabstand: 0
- Rand: 1
- Innenabstand: 3333
- 0 1 8.8 389.95×20 0 1 0
- border-bottom-color: #000000
- border-bottom-style: solid
- border-bottom-width: 1px

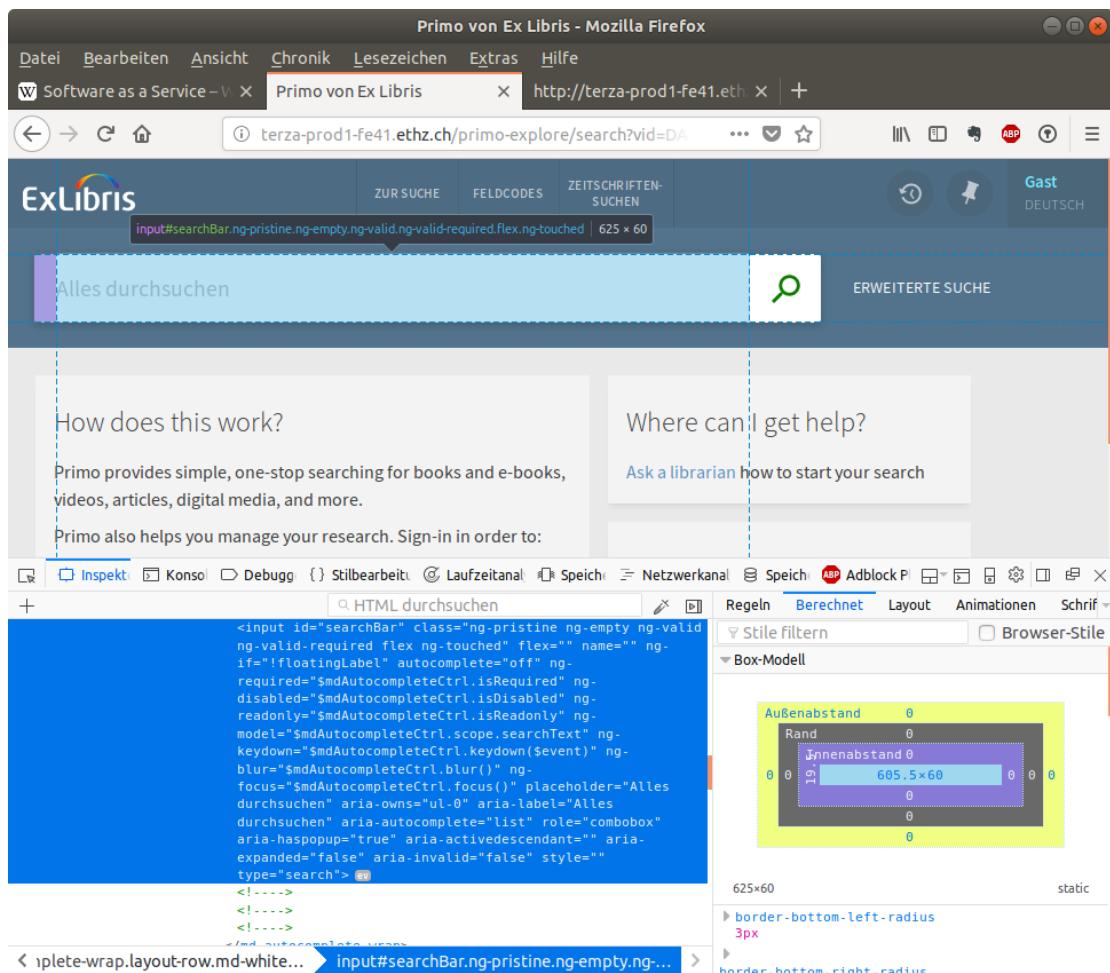
Quellcode des Suchschlitzes (altes UI)

```
<input name="vl(freeText0)" class="" value="" id="search_field" accesskey="s" type="text">
```

Den Suchschlitz können wir also ansteuern, indem wir im DOM nach der id „`search_field`“ suchen.

Ein Problem ist, dass sich ids nach Updates ändern können.

Der Suchschlitz im neuen UI



Quellcode des Suchschlitzes (neues UI)

```
<input flex="" id="searchBar" name="" ng-if="!floatingLabel" autocomplete="off" ng-
required="$mdAutocompleteCtrl.isRequired" ng-disabled="$mdAutocompleteCtrl.
isDisabled" ng-readonly="$mdAutocompleteCtrl.is Readonly" ng-model="
$mdAutocompleteCtrl.scope.searchText" ng-keydown="$mdAutocompleteCtrl.keydown(
$event)" ng-blur="$mdAutocompleteCtrl.blur()" ng-focus="$mdAutocompleteCtrl.focus
()" placeholder="Alles durchsuchen" aria-owns="ul-0" aria-label="Alles durchsuchen
" aria-autocomplete="list" role="combobox" aria-haspopup="true" aria-
activedescendant="" aria-expanded="false" class="ng-pristine ng-empty ng-
valid ng-
valid-required flex ng-touched" aria-invalid="false" style="" type="search">
```

Skript, um „Wald“ in den Suchschlitz zu schreiben

```
Wenn(/^ich in den Suchschlitz "([^\"]*)" eingebe,$/) do |q|
  fill_in('#searchBar', with: q)
  find(".button-confirmed").send_keys(:enter)
end
```

Und der Testablauf

```
cucumber features/suche.feature:50
# language: de
Funktionalität: Suche

  Grundlage: # features/suche.feature:3
    # Gegeben sei , dass die Seite "http://terza-test-fe43.ethz.ch" aufgerufen ist
    Gegeben sei , dass die Startseite aufgerufen ist # features/step_definitions/suche_steps.rb:1

  Szenario: Slider bietet eine plausible Zeitangabe zur Einschränkung der Trefferliste # features/suche.feature:48
    Wenn ich in den Suchschlitz den Suchbegriff "cucumber" eingebe, # features/step_definitions/
    vollansicht_steps.rb:1
      Dann sollte der Slider plausible Anfangs- und Endwerte haben. # features/step_definitions/
    suche_steps.rb:196
      **** alt: 1922--> DS-Startdatum: 1920
      **** jung: 2017--> DS-Enddatum: 2017
      **** Normalisiertes Startdatum: 1920 **** Normalisiertes Enddatum: 2017

1 scenario (1 passed)
3 steps (3 passed)
0m16.334s
```

4.7 Exkurs: Das DOM für automatisiertes Testen von WebApplikationen verwenden

Bei jedem Update einer Software stellt sich uns das Problem, dass wir überprüfen wollen, ob sie noch so funktioniert, wie bestellt. So testen wir beispielsweise bei unserem Discovery Portal, ob sich die Indexierung verändert hat, oder ob der Date-Slider noch funktioniert. Aber warum sollten wir das manuell machen?

Nachdem wir das DOM verwendet haben, um vielfältige Funktionen auf einer Webseite zu realisieren, können wir es auch benutzen, um diese Funktionen automatisch vom Computer testen zu lassen. Das wollen wir uns im Folgenden mal anschauen.

Unser Ziel ist es, dass ein Computer die Webseite wie ein normaler Benutzer aufruft, die Services nutzt und dabei testet, ob alles wie gewünscht funktioniert.

Da taucht übrigens die erste Schwierigkeit auf: Was ist „wie gewünscht“? Fragen wir drei Personen nach einem Feature, bekommen wir mindestens vier verschiedene Antworten.

Indem man sich nun über die zu testenden Funktionen abspricht, deckt man quasi nebenbei derartige Missverständnisse schnell auf. Mehr dazu in meinem Artikel „Automatisiertes Testen von Webapplikationen“ [Link].

Was kann der Computer nun tun?

Beispielsweise lassen sich Elemente einer Webseite mit Skriptsprachen auf den korrekten Inhalt prüfen. Wir hatten das in dem Beispiel schon gesehen, in dem es darum ging, eine Tabellenzeile ein- bzw. auszublenden.

den Inhalt eines Elements einer Webseite abfragen

```
<script type="text/javascript">
    function einausblenden(id) {
        var meinElement = document.getElementById(id);
        if (meinElement.style.display === "none") {
            meinElement.style.display = "table-row";
        } else {
            meinElement.style.display = "none";
        }
    }
</script>
```

Mittels Ruby und den FrameWorks „Cucumber“ und „Capybara“ testen wir nach jedem Update von Primo, ob die Features noch so funktionieren, wie wir es geplant haben:

automatisierte Tests von Primo

- Wir prüfen nach einem Update, ob sich die Suchalgorithmen verändert haben, indem wir die Trefferzahl zwischen den beiden Systemen mit identischem Datenstand vergleichen.
- Der Date-Slider hat in der Vergangenheit Probleme gemacht. Wir checken, ob er plausible Ranges liefert.
- Wir testen, ob nicht-lateinische Schriftzeichen gefunden werden.
- ...

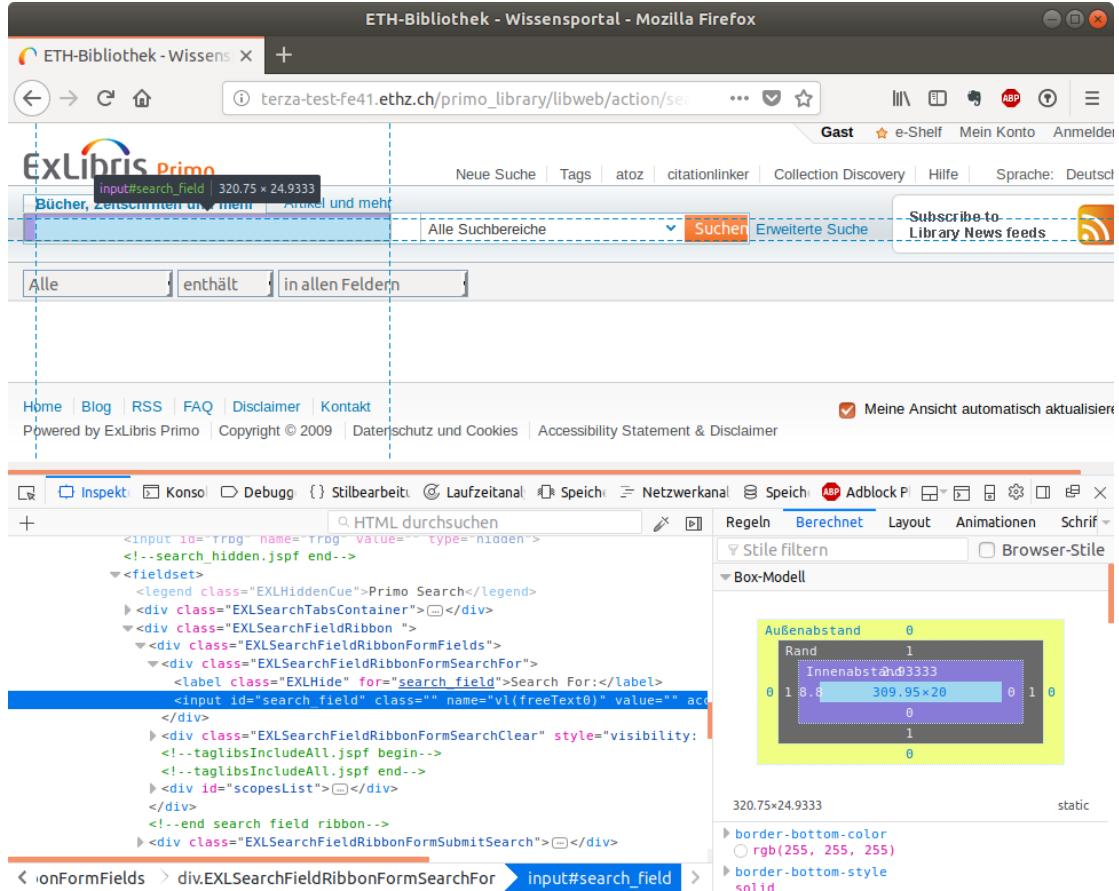
Das Gute an dem verwendeten FameWork ist, dass die Anforderungen nahezu natürlichsprachig formuliert werden. So stellen wir sicher, dass IT und „normale“ Menschen vom Gleichen sprechen.

eine Anforderung in Gherkin formuliert

Szenario: Eine Suche ergibt auf den beiden Prod Systemen eine ähnliche Anzahl Treffer
Wenn ich die Seite "http://terza-prod1-fe41.ethz.ch/primo-explore/search?vid=DADS&sortby=rank&lang=de_DE" aufrufe,
Und ich in den Suchschlitz "Wald" eingebe,
Und die Anzahl der Treffer nehme
Und dann die Seite "http://terza-prod2-fe41.ethz.ch/primo-explore/search?vid=DADS&sortby=rank&lang=de_DE" aufrufe,
Wenn ich in den Suchschlitz den Suchbegriff "Wald" eingebe,
Und dort die Anzahl der Treffer nehme
Dann sollten die Treffermengen ähnlich, d.h. die Abweichung unter 1\%, sein.

Schauen wir uns dazu eine Seite im UI von Primo an. Wir wollen in den Suchschlitz das Wort „Wald“ eingeben.

Der Suchschlitz im alten UI



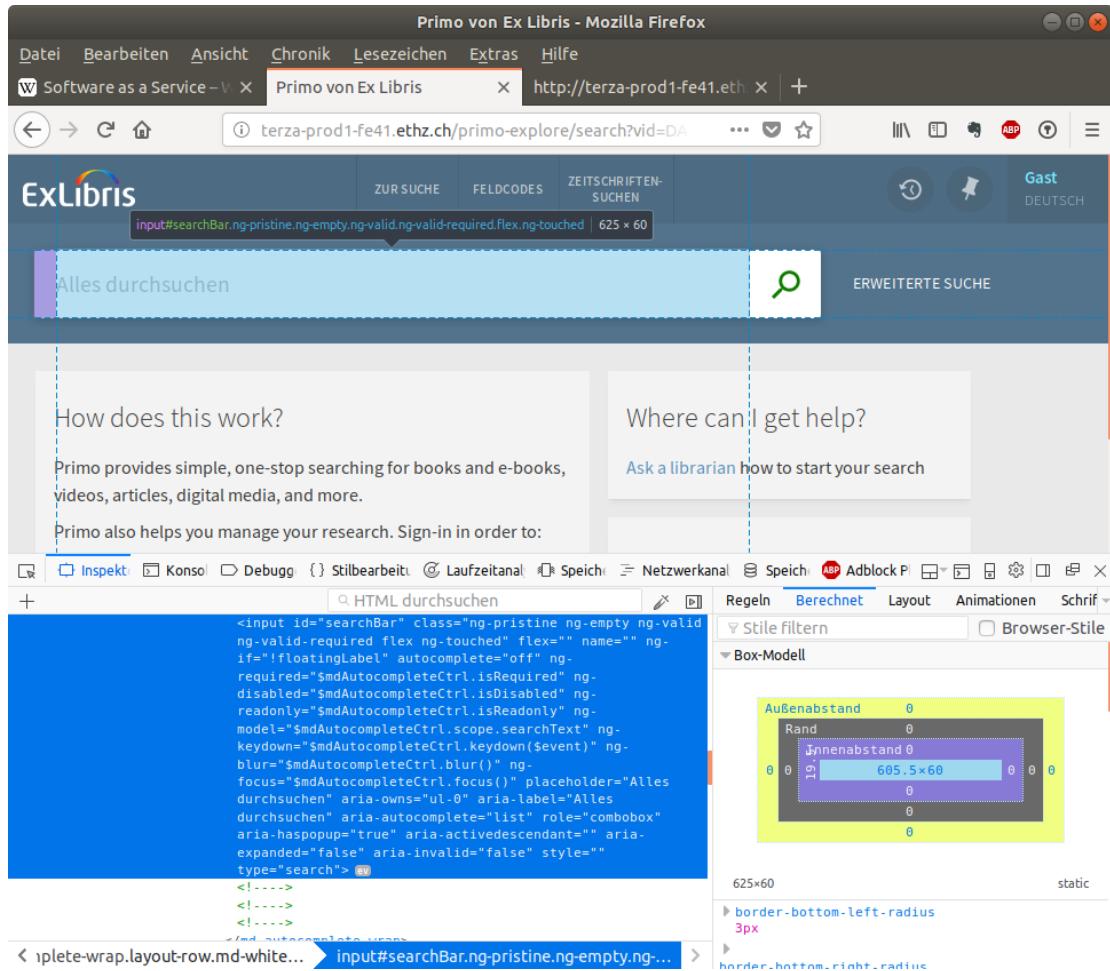
der Quellcode des Suchschlitzes (altes UI)

```
<input name="vl(freeText0)" class="" value="" id="search_field" accesskey="s" type="text">
```

Den Suchschlitz können wir also ansteuern, indem wir im DOM nach der ID „search_field“ suchen.

Ein Problem ist, dass sich IDs nach Updates ändern können.

Der Suchschlitz im neuen UI



der Quellcode des Suchschlitzes (neues UI)

```
<input flex="" id="searchBar" name="" ng-if="!floatingLabel" autocomplete="off" ng-
required="$mdAutocompleteCtrl.isRequired" ng-disabled="$mdAutocompleteCtrl.
isEnabled" ng-readonly="$mdAutocompleteCtrl.is Readonly" ng-model="
$mdAutocompleteCtrl.scope.searchText" ng-keydown="$mdAutocompleteCtrl.keydown(
$event)" ng-blur="$mdAutocompleteCtrl.blur()" ng-focus="$mdAutocompleteCtrl.focus()
()" placeholder="Alles durchsuchen" aria-owns="ul-0" aria-label="Alles
durchsuchen" aria-autocomplete="list" role="combobox" aria-
haspopup="true" aria-activateddescendant="" aria-
expanded="false" aria-invalid="false" class="ng-pristine ng-empty ng-valid ng-
valid-required flex ng-touched" aria-invalid="false" style="" type="search">
```

Die ID heisst nun „searchBar“. Die Tests müssen darauf angepasst werden. Im nächsten Schritt verwenden wir nun die ID, um ein Suchwort in den Suchschlitz zu schreiben.

Ruby-Snippet, um „Wald“ in den Suchschlitz zu schreiben

```
Wenn(/^ich in den Suchschlitz "(*)" eingebe,\$/) do |q|
  fill_in('#searchBar', with: q)
  find(".button-confirm").send_keys(:enter)
end
```

Mit dieser Technik lassen sich alle Elemente auf einer Webseite ansteuern, auslesen und bedienen. So lassen sich komplexe Tests schreiben und dann automatisch vom Computer durchführen.

Der Testablauf auf der Kommandozeile

```
cucumber features/suche.feature:50
# language: de
Funktionalität: Suche

  Grundlage:                                     # features/suche.feature:3
    # Gegeben sei , dass die Seite "http://terza-test-fe43.ethz.ch" aufgerufen ist
    Gegeben sei , dass die Startseite aufgerufen ist # features/step_definitions/suche_steps.rb:1

  Szenario: Slider bietet eine plausible Zeitangabe zur Einschränkung der Trefferliste # features/suche.feature:48
    Wenn ich in den Suchschlitz den Suchbegriff "cucumber" eingebe,                                # features/step_definitions/
    vollansicht_steps.rb:1
      Dann sollte der Slider plausible Anfangs- und Endwerte haben.                         # features/step_definitions/
suche_steps.rb:196
    **** alt: 1922--> DS-Startdatum: 1920
    **** jung: 2017--> DS-Enddatum: 2017
    **** Normalisiertes Startdatum: 1920 ***** Normalisiertes Enddatum: 2017

1 scenario (1 passed)
3 steps (3 passed)
0m16.334s
```

und als Webseite:

The screenshot shows a web-based Cucumber test report. At the top, it displays the goal: "Ziel: http://terza-test-fe43.ethz.ch", the comparison system: "Vergleichssystem: http://terza-prod1-fe43.ethz.ch", and the test date: "Test vom 16.04.2017, 06:00 Uhr". To the right, it shows the test results: "47 scenarios (16 failed, 3 skipped, 28 passed)", "249 steps (16 failed, 9 skipped, 224 passed)", and "Finished in 10m38.108s seconds". Below this, there are sections for "Funktionalität: Erweiterte Suche" and "Beispiele".

Funktionalität: Erweiterte Suche

Grundlage

Gegeben sei, dass die Startseite aufgerufen ist	features/step_definitions/suche_steps.rb:1
Und die Erweiterte Suche geöffnet ist	features/step_definitions/erweiterte_suche.steps.rb:1

Szenariogrundriss: Einschränkung einer Suche mit dem Filter "Erscheinungsjahr"

Wenn ich in das erste Suchfeld das Suchwort "Hasenpfeffer" eingebe	features/erweiterte_suche.feature:11
Und im Erscheinungsjahr-Filter die Option "<Option>" wähle und die Suche abschicke	features/erweiterte_suche.feature:12
Und ich bei den Sortieroptionen die Option "Älteste zuerst" wähle, um das älteste Exemplar zu sehen	features/erweiterte_suche.feature:13
Dann darf das Erscheinungsjahr des ersten Treffers nicht früher sein als die gewählte Option	features/erweiterte_suche.feature:14

Beispiele

Option

Letztes Jahr	Null Treffer gefunden, Szenario übersprungen.
Letzte 2 Jahre	Null Treffer gefunden, Szenario übersprungen.
Letzte 5 Jahre	Null Treffer gefunden, Szenario übersprungen.
Letzte 10 Jahre	2007 ist \geq 2007 und liegt im Zeitraum Letzte 10 Jahre
Letzte 20 Jahre	2007 ist \geq 1997 und liegt im Zeitraum Letzte 20 Jahre
Letzte 50 Jahre	1993 ist \geq 1967 und liegt im Zeitraum Letzte 50 Jahre

Szenariogrundriss: Einschränkung einer Suche mit dem Filter "Ressource"

Wenn ich in das erste Suchfeld das Suchwort "cucumber" eingebe	features/erweiterte_suche.feature:28
Und im Ressource-Filter die Option "<Option>" wähle	features/erweiterte_suche.feature:29
Dann muss der "<Ressourcetyp>" der Treffer dem gewählten Typ entsprechen	features/erweiterte_suche.feature:30

Beispiele

Zeitschriften	Option	Ressourcetyp
Bücher	Zeitschrift	Buch

```
Unable to find css "#resultsNumbersTitle"
./features/step_definitions/erweiterte_suche.steps.rb:68:in `^muss der "([^"]*)" der Treffer dem gewählten Typ entsprechen$'
features/erweiterte_suche.feature:36:in `Dann muss der "Buch" der Treffer dem gewählten Typ entsprechen'
features/erweiterte_suche.feature:30:in `Dann muss der "<Ressourcetyp>" der Treffer dem gewählten Typ entsprechen'
```

Da man die Ausgabe der Test-Ergebnisse auch als Datei — beispielsweise als html-Datei — abspeichern kann, bietet es sich an, diese Tests regelmässig auf einem Server auszuführen und die Ergebnisse als Webseiten den Productownern zur Verfügung zu stellen.

5 Datenbanktechnologien II: BigData

5.1 Klärung verschiedener Begriffe und Buzzwords

Beginnen wir mit der berühmten Cloud. Wer heute etwas in der IT zu entscheiden hat, geht in die Cloud. Die Cloud ist die Lösung für alle IT-Probleme!

Oder?

Die Cloud bietet für viele Anwendungszenarien praktische Lösungen.

Dienstleistungen in der Cloud

Wir erinnern uns:

- SaaS
- IaaS
- PaaS

Ein Problem ist, dass die Services stark standardisiert sind. Ein System, in dem alle vollkommen unabhängig von den eigenen Anforderungen dieselben Services beziehen, ist nicht erfolgreich. So gab es beispielsweise in der DDR für alle nur Trabis, obwohl sicher nicht alle das gleiche Auto fahren wollten.

Ein anderes ist die Frage, welche Daten herausgegeben werden. Die Free Software Foundation formuliert es sehr klar:

It's not the Cloud, it's just other people's computers.

Nochmal deutlich: Cloud bedeutet, dass uns jemand einen Teil der Arbeit abnimmt und dafür unsere Daten nutzt! So bietet Google hervorragende Services kostenlos an — im Tausch für unsere Daten.

Im Falle von Bibliotheksssoftware bekommt der Anbieter zusätzlich zu unseren Daten noch eine Menge Geld.

Um nun entsprechende Systeme aufzubauen, mit denen man derartige Dienstleistungen anbieten kann, benötigt man besondere Techniken mit denen man sehr grosse Datens Mengen verarbeiten und speichern kann.

So kommen wir zu BigData.

5.2 Anwendungsszenarien von Big Data, Anwendung in der ETH-Bibliothek

V

V

V

eine Definition

Der aus dem englischen Sprachraum stammende Begriff Big Data [b̩ dətə] (von englisch *big* ‚groß‘ und *data* ‚Daten‘) bezeichnet Datenmengen, welche

- zu groß,
- zu komplex,
- zu schnelllebig
- zu schwach strukturiert

sind, um sie mit manuellen und herkömmlichen Methoden der Datenverarbeitung auszuwerten.

[Seite „Big Data“. In: Wikipedia, Die freie Enzyklopädie. Bearbeitungsstand: 21. Oktober 2017, 06:34 UTC. URL: <https://de.wikipedia.org/w/index.php?title=BigData&oldid=170175574> (Abgerufen: 26. Oktober 2017, 14:31 UTC)]

V o l u m e

V e l o c i t y

V a r i a n c e

Was ist nun BigData?

DataScience	BigData	MachineLearning
Wissensgewinnung <ul style="list-style-type: none">• Sichtung• Umwandlung• Analyse• Visualisierung	Datenhaltung <ul style="list-style-type: none">• Speicherung• Sicherung• Abfrage	KI <ul style="list-style-type: none">• Training• Klassifizierung• Vorhersage

Umgangssprachlich wird BigData für alle drei Themen genutzt. So bedeutet die innovative Idee „Lasst uns mal BigData machen“ in der Regel nicht, uns eine neue Datenbank zuzulegen.

Um uns den Möglichkeiten in Bibliotheken weiter zu nähern, fangen wir mit einer Selbst-einschätzung an.

Datenmengen an der ETH–Bibliothek

- Bildarchiv Online: ca. 600'000 Datensätze
- Aleph: ca. 7,9 Millionen Datensätze
- Primo: ca. 10,1 Millionen Datensätze

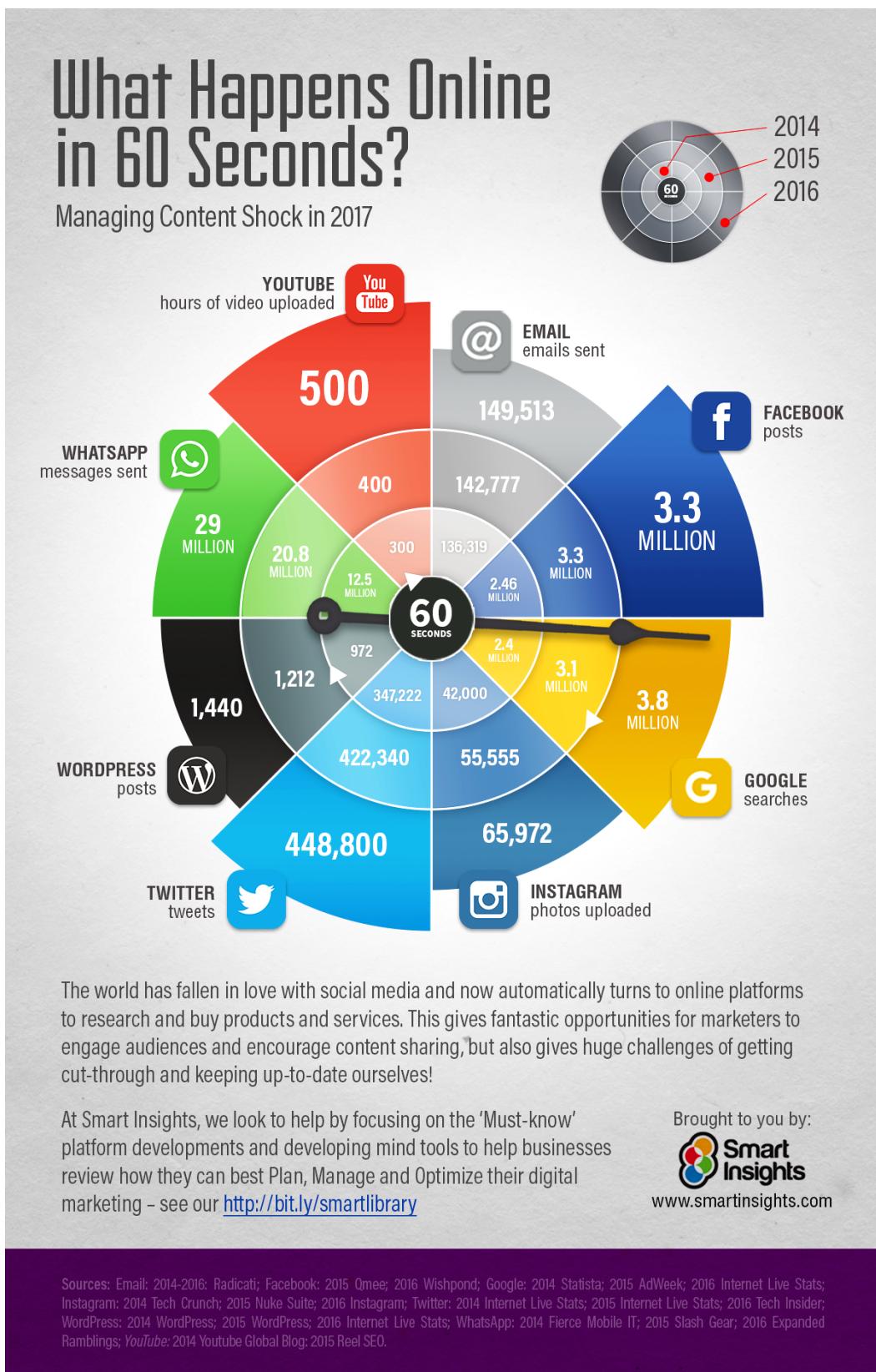
Obwohl 10,1 Millionen Datensätze in einem Bibliothekskatalog schon eine ganze Menge sind, kommt es bei Weitem nicht an die Mengen heran, die üblicherweise mit BigData bezeichnet werden.

ein frustrierender Vergleich:

Ein Beispiel für Datensatzzahlen im BigData–Bereich sind die ca. 15 Milliarden Tweets die über Twitter in einem Monat versendet werden.

Auch sind die Datenzuwächse und –veränderungen bei uns nicht mit denen im BigData–Business vergleichbar.

Hier ein paar Zahlen aus dem Web:



Auch die Varianz ist bei unseren Metadaten — hoffentlich — nicht gross. Im Gegenteil, unsere Daten sind nach klaren Regeln erfasst und idealerweise über Systemgrenzen austauschbar.

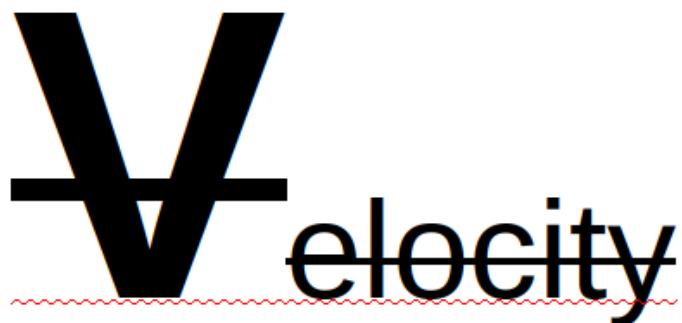
Varianz

```
<recordid>ebi01_prod010616366</recordid>
<type>book</type>
<title>Sophocles: four tragedies</title>
<creator>Sophocles, v497-v407</creator>
<creator>Oliver Taplin, 1943-</creator>
<edition>First edition</edition>
<publisher>Oxford, United Kingdom : Oxford University Press</publisher>
<creationdate>2015</creationdate>
<subject>Sophocles -- Translations into English</subject>
<subject>Oedipus, Greek mythological figure Drama</subject>
```

Varianz

```
<recordid>ebi01_prod010103021</recordid>
<type>image</type>
<title>[Aias und Kassandra]</title>
<creator>Heinrich Meyer, 1760-1832</creator>
<creator>Johann Heinrich Lips, 1758-1817</creator>
<publisher>[Weimar]</publisher>
<creationdate>[1794]</creationdate>
<format>1 Druckgraphik : Aquatinta und Radierung in Schwarz und Braunrot ; Bild 18,5 x
    23,1 cm, Blatt 23,4 x 32,2 cm</format>
<identifier><b>DOI: </b>10.3931/e-rara-37966 ; $$CBIBL$$VJoachim Kruse: Johann
    Heinrich Lips 1758-1817, Coburg 1989, S. 220 ; $$CBIBL$$VErschienen in: Heinrich
    Meyer/Carl August Böttiger (Hg.): Über den Raub der Cassandra auf einem alten Gefä
    sse von gebrannter Erde, Weimar 1794, S. 93</identifier>
<subject>Kassandra, Fiktive Gestalt</subject>
<subject>Athena, Göttin</subject>
```

ergo



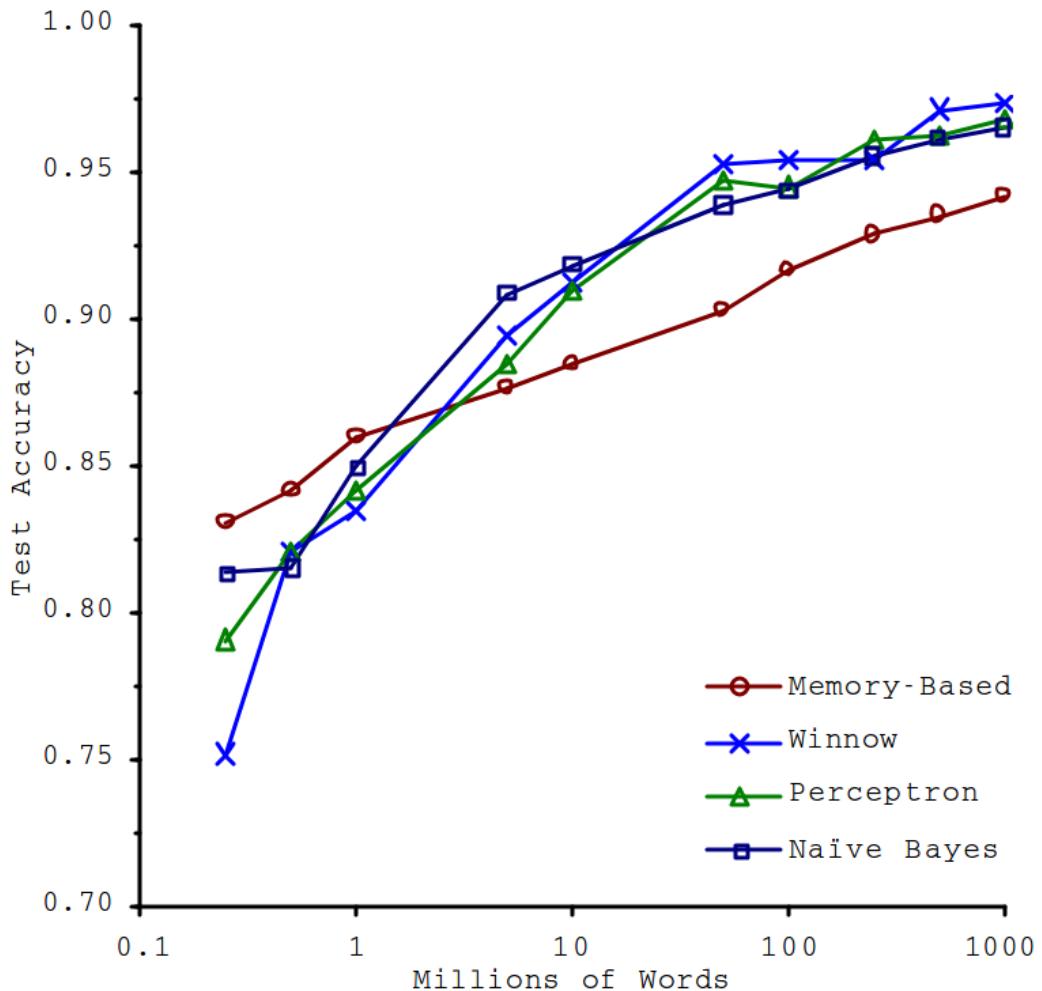
BigData ohne Big Data?

Wir haben so wenige, so gut strukturierte Daten, dass wir kein Einsatzszenario für BigData entwickeln können, bei dem die Kosten/Nutzen-Rechnung stimmt.

Auch für eine sinnvolle Anwendung von MachineLearning reichen unsere Daten kaum

aus. Microsoft Research hat das gut verdeutlicht. Um angemessen akkurate Aussagen treffen zu können, benötigt man mehr als 50 Millionen Datensätze.

machine learning



Eine Studie von Microsoft Research kommt bei einem Vergleich verschiedener Klassifizierungsalgorithmen zu dem Ergebnis, das die Tests erst ab ca. 50 Millionen Datensätzen eine befriedigende Genauigkeit aufweisen.

[Michele Banko and Eric Brill, Microsoft Research: Scaling to Very Very Large Corpora for Natural Language Disambiguation]

5.3 In Medias Res: DataScience am Bsp. Logfiles und Benutzerdaten, BigData am Bsp. OAI–Server

Allerdings können wir einzelne Technologien aus dem BigData–Bereich durchaus gewinnbringend einsetzen. So lassen sich beispielsweise grosse Mengen an Log–Daten mit modernen DataScience–Methoden gut sichten und für erste Auswertungen nutzen.

Mit der Auswertung von Logfiles lassen sich Aussagen zum Nutzerverhalten treffen. Dabei kommen zunächst Methoden der IT zum Einsatz:

anonymisierte Logdaten aus dem Discovery–Portal

So werden aus insgesamt 158'203'686 Zeilen aus 617 Logfiles 6'405'616 relevante Zeilen extrahiert:

```
awk 'match($7,/(\./.+)+\/search\.do.+freeText/) {sub(/^(.,"NEBIS"\t",$4);sub(/(\./.+)+\/search\.do/, "",$7);print $1"\t"$4"\t"$7}' access_log.2016* > suchanfragen.log
```

anonymisierte Logdaten aus dem Discovery–Portal

Mit der Sprache Python werden die Daten in ein Dataframe geladen und können so in beliebigen Ausschnitten gesichtet werden, um anhand von Stichproben die weiteren Bearbeitungsschritte festzulegen...

	IP	country	Date	FE	url	query_type	query
0	3644786527210799	Germany	01/Jan /2016:01:07:13	NEBIS	?srt=title&srtChange=true&&dscont=0&vl(21516851...	Basic	['Verbalphrase']
1	3640797120747109	Switzerland	01/Jan /2016:01:10:20	NEBIS	?fn=search&ct=search&vl(freeText0)=kadhija&se...	[]	['kadhija']

anonymisierte Logdaten aus dem Discovery–Portal

...oder erste Analysen zu machen:

```
len(logframe[logframe['wcount'] == 1 ].index) * 100 / len(logframe.index),  
len(logframe[(logframe['wcount'] == 1) & (logframe['query_type'] == "Advanced")].index) * 100 / len(logframe)
```

96.3 Prozent aller einfachen Suchanfragen beinhalten nur ein Suchwort. 62.7 Prozent aller erweiterten Suchanfragen beinhalten ebenfalls nur ein Suchwort. 8.4 Prozent aller Suchanfragen werden über die erweiterte Suche durchgeführt.

Na, erscheinen uns die Werte plausibel? Hier liegt der Vorteil in den Methoden der DataScience: Erste Auswertungen und Übersichten kosten nichts. So brauchte die Auswertung zu den Suchanfragen über sechs Millionen Zeilen nur knapp drei Sekunden. Zum Vergleich: Das Laden aller Daten ins den DataFrame dauert ca. 30 Sekunden, ein Versuch, dieselben Daten in Excel zu laden bringt die Software zum Absturz.

Wir können also jederzeit in die Daten schauen und sie bei jedem Schritt auf Plausibilität überprüfen. Im obigen Beispiel gab es einen Fehler bei der Aufteilung des Strings in Suchwörter.

anonymisierte Logdaten aus dem Discovery–Portal

Die korrekten Zahlen:

23.86981985807454 Prozent aller einfachen Suchanfragen beinhalten nur ein Suchwort. 26.29683387412833 Prozent aller erweiterten Suchanfragen beinhalten ebenfalls nur ein Suchwort. 8.357088529815087 Prozent aller Suchanfragen werden über die erweiterte Suche durchgeführt. Die Auswertung dauerte 1.91 Sekunden.

Der Einsatz von DataScience Methoden ist nicht immer optimal. So war es wesentlich einfacher, die 150 Millionen Zeilen mit awk auf die relevanten sechs Millionen einzudampfen, als alle Daten erst in ein Dataframe zu laden und dann die relevanten zu extrahieren.

DataScience für alles?

Bsp. Benutzerdatensplitting: Automatisiertes Splitten von 100 Datensätzen, die aus zwei Elementen bestehen, in Vor- und Nachname:

mit Python und nltk:	16.73	Sekunden
mit Python ohne nltk:	0.69	Sekunden
mit Ruby:	0.39	Sekunden

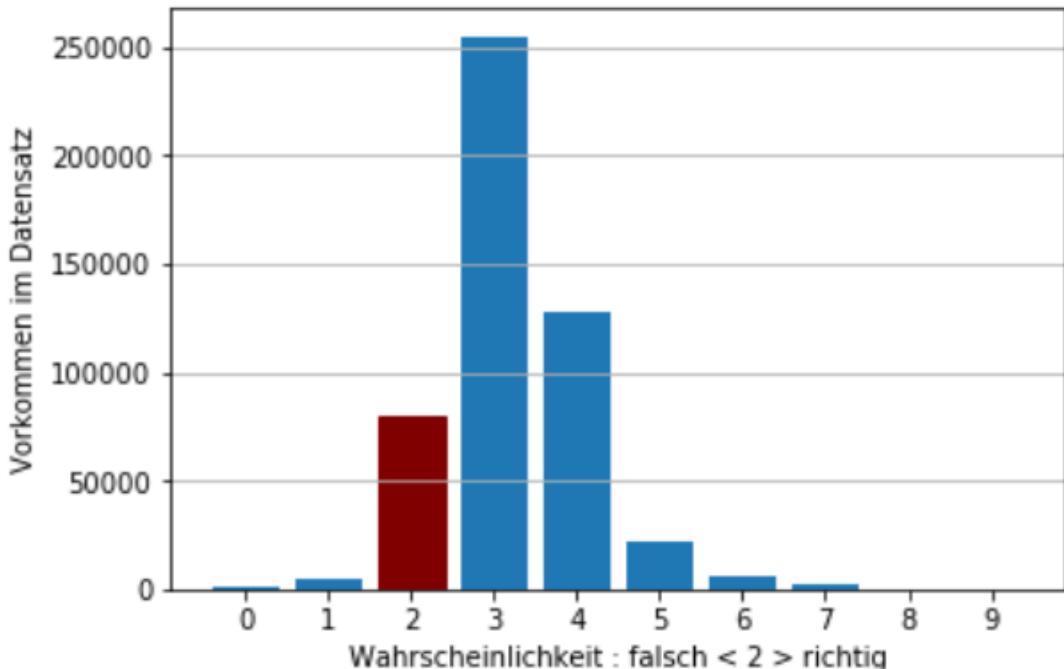
Das wirkt sich bei 650'000 Datensätzen schon deutlich aus.

DataScience–Methoden zum Sichten

Auch bei „wenigen“ Daten können Methoden aus der DataScience genutzt werden, um sich schnell einen Überblick zu verschaffen und danach das weitere Vorgehen aufgrund von Fakten besser planen zu können.

DataScience–Methoden zum Sichten

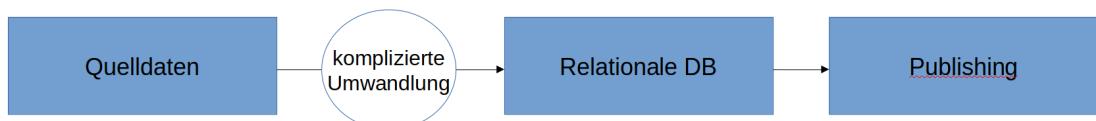
Wieder am Beispiel Benutzerdatensplitting: Visualisierung der Ergebnisse einer maschinellen Bearbeitung:



BigData Technologien für small data:

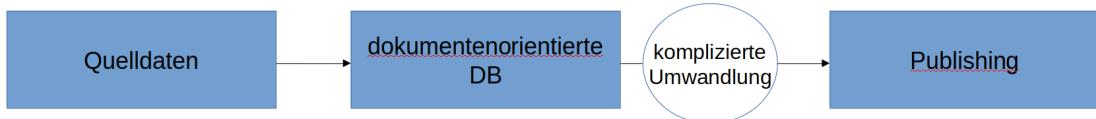
Aufbau eines OAI–Servers für die Daten der graphischen Sammlung: 23‘275 Datensätze, annähernd OAI_{DC} , Veränderung in den letzten Monaten nicht mehr ein Datensatz/Woche

gestern: Alle Quelldaten werden bereinigt und konvertiert, so dass sie beim Publishen direkt aus der DB gezogen und korrekt ausgegeben werden können.



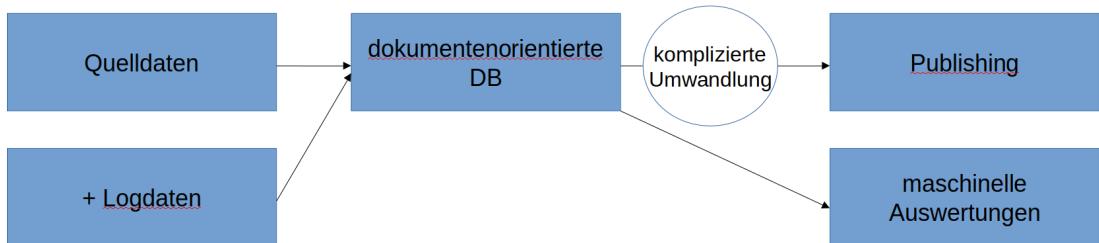
BigData Technologien für small data:

heute: Alle Quelldaten werden unverändert in eine dokumentenorientierte DB, die gut skalierbar ist, geladen. Die Umwandlung passiert erst beim Publishing.



BigData Technologien für small data:

morgen: Dadurch dass die Umwandlung erst beim Publishing passiert und somit alle Daten komplett und unverändert in der DB vorhanden sind, lassen sich aus den Quelldaten und Verknüpfungen mit weiteren Daten umfangreiche Auswertungen ziehen.



6 Von Verschlüsselung via hashing zur BlockChain

6.1 Verschlüsselung

Das Problem bei allen Anwendungen im Netz — seien es Cloud–Applikationen oder einfach nur der Versand einer E–Mail — ist, dass die Daten auf ihrem Weg mitgelesen werden können. Tatsächlich fliegt eine Mail so offen durch das Netz, wie eine Postkarte vom Urlaubsort zum Empfänger.

Damit im Postverkehr nicht jeder meine Gedanken verfolgen kann, packe ich sie in einen verschlossenen Umschlag. Mails müsste man verschlüsseln. Das ist jedoch immer noch relativ kompliziert, so dass der meiste E–Mail–Verkehr immer noch offen durch's Netz fliegt.

Leider hinkt auch sonst der Vergleich mit der guten alten Post: Während im Postwesen unsere Kommunikation in vielen Ländern durch das Fernmeldegeheimnis geschützt ist, nehmen sich die Staaten bei digitaler Kommunikation das Recht, im grossen Stil mitzulesen.

Was man für sich behalten möchte, gehört nicht ins Netz.

Damit ergibt sich nun ein Problem: Wenn ich etwas verschlüsseln und mit jemand anderem austauschen möchte, müssen wir uns über den Schlüssel verstständigen. Das ist Einfach, wenn der Kommunikationspartner nebenan wohnt. Ich gehe einfach rüber und wir bauen uns einen digitalen Schlüssel, mit dem wir unsere Texte zukünftig ver- und wieder entschlüsseln.

Jeder, der an unserer Kommunikation teilnehmen möchte, bekommt den Schlüssel und kann von da ab alles Mitlesen.

Das wird nun schwierig, wenn einer der Partner nicht am selben Ort ist. Man kann sich den erstellten Schlüssel ja schlecht per Mail schicken. Die Mail kann gelesen werden, und jeder, der den Schlüssel hat, kann den gesamten Verkehr entschlüsseln.

Darum wurde die asymmetrische Verschlüsselung entwickelt. Sie basiert darauf, dass jeder Teilnehmer einen eigenen Schlüssel zum entschlüsseln und einen sogenannten öffentlichen Schlüssel von jedem seiner Kommunikationspartner zum verschlüsseln hat.

Nochmal: Jeder hat einen privaten Schlüssel, mit dem er die an sich selber gerichtete Kommunikation entschlüsseln kann, und einen öffentlichen Schlüssel, mit dem alle anderen die an ihn gerichtete Kommunikation verschlüsseln können.

Der Ablauf ist nun folgender: Ich möchte eine Mail an meinen Freund, nennen wir ihn Hans Heimlich, schicken. Hans Heimlich schickt mir ganz offen seinen öffentlichen Schlüssel. Er liegt auch auf diversen Schlüsselservern, damit jeder, der mit ihm in Kontakt treten möchte, den Schlüsseln finden kann.

Diesen öffentlichen Schlüssel von Hans Heimlich benutze ich nun, um den an ihn gerichteten Text zu verschlüsseln. Nachdem ich den Text verschlüsselt habe, kann ich ihn selber nicht mehr lesen. Denn mit dem öffentlichen Schlüssel kann man zwar verschlüsseln, aber nicht entschlüsseln! Ich kann also den verschlüsselten Text über das Netz versenden, und niemand kann ihn (mit vertretbarem Aufwand — bei aktuellen Schlüsseln brauchen Supercomputer Tage bis Wochen, um eine Mail zu entschlüsseln) mitlesen.

Der verschlüsselte Text kommt also bei Hans Heimlich an. Er nimmt nun seinen *privaten* Schlüssel, um den Text zu entschlüsseln. Sein privater Schlüssel ist der einzige, mit dem man den Text wieder entschlüsseln kann. Deshalb achtet Hans Heimlich darauf, ihn nicht zu verlieren, aber auch darauf, dass er keinesfalls im Netz landet! Also kein Backup des Schlüssels auf DropBox, Google— oder iCloudDrive!

Und das macht die Verschlüsselung so sicher wie unpraktisch: Heute lesen wir Mails nicht mehr nur auf einem Gerät. Wir müssten den privaten Schlüssel aber auf jedem Gerät haben, mit dem wir Mails empfangen — also auch auf dem Smartphone. Keine gute Idee!

Praktikable Lösungen für dieses Problem gibt es noch nicht.

6.2 hashing

Bei Dateien, die wir aus dem Netz laden, seien es öffentliche Schlüssel, Dokumente oder Software, ergibt sich immer die Frage, ob ich auch wirklich die Datei bekommen habe, die ich haben wollte. Das schauen wir uns an einem Beispiel an.

Ein Programmierer, Mr. Goodcode unterhält eine Webseite, auf der er seine Software anbietet, die Dokumentation und eine FAQ-Seite pflegt. Die Software kann man sich herunter laden. Da für Webseiten die Datenmenge, die herauf und herunter geladen werden kann üblicherweise begrenzt und teuer ist, liegt die Software nicht in einem Unterverzeichnis der Webseite, sondern bei einem File-Hoster. Der Benutzer wird also bei Klick auf den Download-Link auf die Seite des File-Hosters weiter geleitet, von wo aus der Download dann direkt startet.

Nun wechseln wir mal die Perspektive und schlüpfen in die Rolle des Hackers Mr. Badcode. Badcode hat einen Schadcode entwickelt und möchte ihn auf möglichst viele Rechner verteilen. Er hackt sich in die Server eines File-Hosters ein, entpackt die dort liegenden Downloads, fügt den Programmen seinen Schadcode hinzu und verpackt alles wieder. Niemand bemerkt seinen Einbruch und ab sofort lädt jeder, der von dem Filehoster Software bezieht, den Schadcode von Badcode mit auf seinen PC.

Mr. Goodcode möchte das verhindern und sicherstellen, dass die Kunden seine Software unverändert bekommen, wenn sie sie irgendwo aus dem Netz herunter laden. Glücklicherweise gibt es eine Lösung: Mr. Goodcode berechnet auf seinem Computer von seiner

Software mit einem weit verbreiteten Algorithmus eine Prüfsumme, einen sogenannten hash.

Ein hash ist vergleichbar mit einer Quersumme einer Zahl. Allerdings wird durch den Algorithmus sichergestellt, dass mit einer sehr grossen Wahrscheinlichkeit keine zwei unterschiedlichen Codes die gleiche Prüfsumme ergeben können.

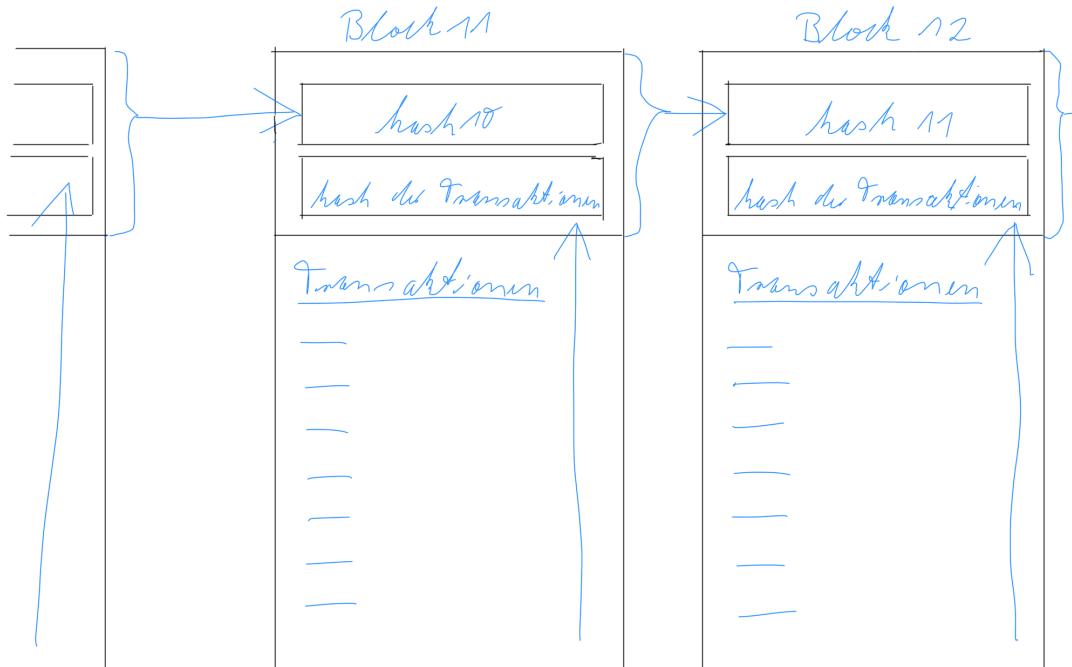
Mr. Goodcode verteilt nun seine Software im Netz und veröffentlicht auf seiner Webseite die Prüfsumme. Wer sich jetzt die Software irgendwo herunter lädt, erzeugt von dem erhaltenen Download mit dem gleichen Algorithmus die Prüfsumme und vergleicht sie mit der auf der Webseite veröffentlichten. Stimmt sie überein, ist die Software unverändert. Mr. Badcode müsste sich nun in den File-Hoster und zusätzlich in die Webseite von Mr. Goodcode hacken, um sowohl den Download als auch den veröffentlichten hash zu manipulieren. Und diese Manipulation auf seiner Webseite würde Mr. Goodcode schnell auffallen.

6.3 BlockChain

In der ersten Lektion hatten wir uns ein Konzept angesehen, mit dem man wachsenden Datenmengen begegnet: verschiedene Techniken um Daten in mehreren Speichersystemen — Festplatten oder Datenbanken — gleichzeitig abzulegen. Das Problem dabei ist, sicher zu stellen, dass die Daten in allen Speichern konsistent sind. Sie zu vergleichen würde viel zu viel Rechenzeit kosten. Stattdessen hasht man sie (bzw. Teile davon) und vergleicht die hashes. Dafür braucht es aber einen „Master“, der die Transaktionen überwacht und die Verantwortung dafür übernimmt, dass alle Daten über alle Systeme einen konsistenten Zustand haben. Mit diesem Master haben wir aber auch wieder einen Single Point of Failure — fällt er aus, ist das ganze System unbrauchbar.

Mit der BlockChain wird dieses Problem gelöst. Hier sind alle Geräte gleichberechtigt dafür verantwortlich, dass alle denselben Stand haben. Eine einfache Mehrheit genügt, um zu bestätigen, dass eine Transaktion so gelaufen ist, wie die gelaufen ist. Oder umgekehrt: Um eine Transaktion zu fälschen, muss man 51% aller Ressourcen unter seine Kontrolle bringen.

Dazu kommt ein weiteres Konzept: Die Transaktionen werden in Blöcke zusammengefasst. Dann wird über den Block ein hash gebildet. Über diesem hash und dem hash der vorangegangenen Blocks wird ein weiterer hash gebildet, der wieder im nächsten Block abgelegt wird. So würde das Fälschen einer Transaktion alle folgenden ungültig machen.



Das Konzept hat natürlich auch Nachteile: Zum einen liegen immer alle Daten auf allen Systemen redundant vor. Bitcoin beispielsweise belegte im Januar 2018 rund 150 GB auf der Platte, und das redundant auf jedem teilnehmenden System. Zum anderen wird das Berechnen der hashes immer aufwendiger, je länger die Kette wird. Der Energiehunger von BitCoin liegt laut des Bitcoin Energy Consumption Index bei rd. 50 TWh, das entspricht dem Strombedarf von Portugal.

Und gerade bezüglich digitaler Währungen gibt es ein nicht zu unterschätzendes Risiko: Die Sicherheit der Transaktionen beruht auf der Annahme, dass ein einzelner nicht genügend Rechenleistung zur Verfügung hat, um die Hashes zu knacken und Manipulationen schneller vornehmen zu können, als alle anderen benötigen, die Kette weiter zu verlängern. Es ist inzwischen recht wahrscheinlich, dass Quantencomputer in absehbarer Zeit Realität werden. Der erste Quantencomputer wäre so schnell, dass er die Blockchain aushebeln könnte. Wenn sich bis dahin das Konzept nicht noch weiter optimiert haben wird, wäre die Währung auf einen Schlag wertlos.