HS München Fakultät 07

Studienarbeit

Big Data

Hype oder Chance ?

Verfasser: Sebastian Kraubs

Matrikelnummer: 03315808

Erstprüfer/-in / Betreuer/-in: Michael Theis

Ich, Sebastian Kraubs, versichere, dass ich die vorstehende Arbeit selbständig angefertigt und mich fremder Hilfe nicht bedient habe. Alle Stellen, die wörtlich oder sinngemäß veröffentlichtem oder nicht veröffentlichtem Schrifttum entnommen sind, habe ich als solche kenntlich gemacht.

_____ München den 30.05.2014

Inhaltsverzeichnis

Big Data

Bei „Big Data“ handelt sich um einen Begriff, der in den letzten Jahren, seit seiner Entstehung, stark gehypt wurde und welcher durchaus irreführend sein kann. Viele IT-Veteranen lehnen den Begriff daher ab. Allgemein betrachtet wird sich mit den Kriterien, Problemen und Verfahrensansätzen von Datenmanagement beschäftigt. Oftmals wird versucht mit den sogenannten „3 V‘s“, dies sind Kriterien, Herausforderungen, Probleme oder auch Dimensionen vor denen ein Unternehmen im Bereich Datenmanagement steht, den Begriff „Big Data“ zu umschreiben. Doch sollten dies die einzigen Aspekte sein, die nötig wären diesen Begriff zu definieren, so würden die aktuellen Datenmanagement- Verfahren ausreichen und der Begriff „Big Data“ nicht notwendig sein[1]. Es wird nicht oftmals nicht deutlich genug in wie weit sich „Big Data“ von den gängigen Verfahren und Definitionen unterscheidet.

Bereits 2001 beschrieb Doug Laney in einem „META Group“– Forschungsbericht die Problematik, mit Hilfe der „3 V’s“, vor der Unternehmen im Bereich Datenmanagement, ausgelöst durch das Datenwachstum, stehen oder in Zukunft stehen werden, ohne dabei von „Big Data“ zu sprechen. Erst ab 2007 finden sich erste Erwähnungen von „Big Data“ in Blogs, jedoch in Bezug auf entstehende Internettechnologien, aber bereits 2010 hatte sich der Begriff mit der heute gängigen Definition etabliert[2]. Die Definitionen, die in einer von Gartner im Jahr 2011 veröffentliche Publikation aufgestellt wurden, stellen heute die Basis dar.

So kann man von „Big Data " sprechen, wenn es sich um Datensätze handelt, deren Größe über den Möglichkeiten von typischen Datenbank-Software-Tools liegt, um diese zu speichern, zu verwalten und zu analysieren. Die Daten können dabei aus unterschiedlichsten Quellen stammen und in vielfältiger Form vorliegen. „Big Data“ hat dabei das Ziel die Analyse großer Datenmengen aus vielfältigen Quellen in hoher Geschwindigkeit, wirtschaftlichen Nutzen zu erzeugen. Bei „Big Data“ handelt sich um den Prozess aus Daten Informationen zu gewinnen und diese in Form von Wissen

[1] http://tdwi.org/articles/2012/07/24/big-data-4th-v.aspx

[2] http://it.toolbox.com/blogs/infosphere/the-origin-and-growth-of-big-data-buzz-51509

anzuwenden, oder anders ausgedrückt der Prozess der Informationsgewinnung und Informationsanwendung aus unterschiedlichsten Daten.

Die bisherigen Ansätze mit RDBMS, Data Warehouses und Reporting sind oftmals nicht mehr ausreichend um der schieren Masse und Vielfalt an Daten Herr zu werden, daher haben sich in den letzten Jahren neue Konzepte, Methoden und Technologien entwickelt, die die bestehenden erweitern oder ersetzen. Es wird schnell klar, dass „Big Data“ ein chaotischer Bereich der IT ist, der sich mit den wandelnden Anforderungen ständig und schnell ändert. Aber eins muss auch klar sein, die Daten, so wie die daraus gewonnen Informationen und die Geschwindigkeit, mit der man diese gewinnt und anwendet, werden oder sind bereits die ultimativen Erfolgsfaktoren am Markt für Unternehmen mit den richtigen Anwendungsfällen.

1 Wie wird „Big Data“ zum Erfolgsfaktor

Eine Studie aus dem Jahre 2011 von McKinsey argumentierte, dass durch den Einsatz von „Big Data“ Fünf neue Unternehmenswerte entstehen[3]:

- Schaffung von Transparenz in organisatorischen Tätigkeiten, die zu einer Erhöhung der Effizienz führt.
- Ermöglicht eine genauere Analyse der Leistungen von Mitarbeitern und Systemen. Dies ermöglicht Experimente mit exaktem Feedback.
- Segmentierung der Bevölkerung, um Maßnahmen anpassen zu können.
- Austausch / Unterstützung der Menschen bei der Entscheidungsfindung mit automatisierten Algorithmen
- innovative neue Geschäftsmodelle, Produkte und Dienstleistungen.

[3]Bigdata: The next frontier for innovation, competition, and productivity

2 Definition der Eigenschaften von „Big Data“

Wie bereits im ersten Abschnitt versucht wurde deutlich zumachen steht die Anforderung im Vordergrund, Informationen aus X-beliebigen Daten zu gewinnen und für sich und sein Unternehmen zu verwenden. Die Mittel, die dafür genutzt werden sind nachrangig.

Bei der Gewinnung dieser Informationen stehen Unternehmen jedoch vor bestimmten Problemen oder auch „Schmerzen“[4], die sich aus den Eigenschaften von „Big Data“ ableiten lassen. Es wird hierbei gerne von den „3 V’s“ gesprochen. Doug Laney bezeichnete die Herausforderungen des Datenwachstums als dreidimensional[5]. Die drei Dimensionen beziehen sich auf ein ansteigendes Volumen (Volume) der Daten, auf eine ansteigende Geschwindigkeit (Velocity), mit der Daten erzeugt und verarbeitet werden und auf eine steigende Vielfalt (Variety) der erzeugten Daten (siehe Abbildung 1). Je nach Anwendungsfall ist mindestens eine dieser Dimensionen betroffen, aber oftmals mehr.

Jedoch häufen sich die Stimmen, welche der Meinung sind, dass die „V’s“ nicht ausreichen um „Big Data“ in vollen Umfang zu erfassen. Für die Einordnung und für die ersten Überlegungen bzw. für einen Industrieweiten Diskurs über das Thema „Big Data“ sind sie aber ausreichend. Mit Fortschreiten eines konkreten Projekts kommen dann zusätzliche Klassifikationen hinzu, wie Chaos, Geographie, Betriebszwänge, Datenschutz usw.

Es soll noch erwähnt werden, dass mittlerweile aber auch von den 4 V’s, 5 V‘s oder gar 6 V’s die Rede ist. Weitere V’s sind die Zuverlässigkeit (Veracity) der Daten, der wirtschaftlichen Wert (Value), variability/variance, viability und victory. Diese Erweiterungen sind mal mehr oder weniger umstritten, da diese unteranderem teilweise bereits durch die 3 V’s abgedeckt werden. In den folgenden Abschnitten wird

[4] Pavlo Baron: Big Data für IT-Entscheider

[5] Doug Laney: Application Delivery Strategies

nun näher auf die einzelnen traditionellen Dimensionen, sowie Veracity und Value eingegangen.

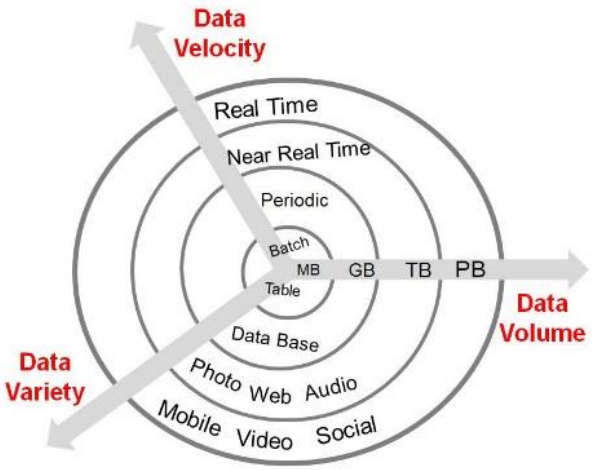

Abbildung 1: Die drei Dimensionen von "Big Data"

2.1 Volume (Datenmenge)

Im Jahre 2011 knackte das weltweite Datenvolumen die Zettabyte-Barriere (1 mit 21 Nullen) und ein Ende des Wachstums ist nicht in Sicht. 2020 sollen es bereits 35 - 40 Zettabyte sein. Es sind aber nicht die gigantischen Datenmengen, die das größte Problem darstellen, aber dazu später mehr.

Es ist keine neue Entwicklung, dass sich die Datenmengen seit Einführung des PCs um das X-fache jährlich Potenzieren, aber allein dadurch lässt sich das exponentielle Wachstum nicht erklären. Eine Vielzahl neuer unterschiedlicher Quellen ist dafür verantwortlich: Sensordaten, Maschinendaten, Log-Daten, das WorldWideWeb (vor allem Sozial Media), RFID usw.. Auch sollte nicht vergessen werden, dass 6 von 7 Milliarden Menschen inzwischen ein Handy besitzen[6]. Mit der zunehmenden Verbesserung der Infrastruktur, auch in ärmeren Regionen der Welt, wird die Nutzung von Dienstleistung exponentiell zunehmen. Als Beispiel soll hier das größte soziale Netz Facebook genannt werden welches weltweit über 1,19 Milliarde Nutzer verzeichnet, von denen allein über 870 Millionen über ein mobiles Endgerät auf das

[6] Die Zahl der Handys werde demnach 2014 voraussichtlich die Marke von sieben Milliarden erreichen, sagen die ITU-Experten voraus.

soziale Netz zugreifen (Facebook nutzt für den Anwendungsfall Nachrichtenverarbeitung HBase auf Basis des Hadoop- Frameworks). Weitere Beispiele sind die 200 Millionen Emails, die pro Minute verschickt werden oder die 175 Millionen Kurznachrichten, die über Twitter von geschätzten 500 Millionen Accounts pro Tag gepostet werden.

Viele solche Quelle kann oder muss ein Unternehmen nutzen, denn je mehr Daten ein Unternehmen hat, umso qualifizierter sind die Entscheidungen basierend auf der daraus gewonnen Information. Die Analyseansätze benötigen große und auch qualitativ hochwertige Mengen an Daten. In Folge dessen sind immer mehr Organisationen mit gigantische Zahlen von Datensätzen, Dateien und Messdaten konfrontiert. Dieses Volumen ist eine der größten Herausforderungen für die konventionellen IT-Strukturen. Es erfordert neue Ansätze, bei der der Speicher beliebig skalierbar ist, daher wird bei „Big Data“ massiv auf verteilte und parallele Verarbeitungsarchitekturen gesetzt.

2.2 Velocity (Geschwindigkeit)

Die Geschwindigkeit mit der sich Daten bewegen und in einem Unternehmen eintreffen hat sich ähnlich rapide erhöht wie das Datenvolumen. Aber nicht allein die Geschwindigkeit mit der die Daten eintreffen stellt ein Problem dar, vielmehr liegt das Problem darin so schnell wie möglich auf die eingehenden Daten (die Events) zu reagieren. Denn je schneller man Informationen aus den Daten extrahiert, umso schneller kann man reagieren. Und je schneller man in der Lage ist zu reagieren, anhand der produzierten Informationen, umso erfolgreicher ist man am Markt.

Für viele Systeme ist daher das Erzeugen von Informationen wichtiger, als das Volumen. Um dies zu verdeutlichen veröffentlichte IBM einen Werbespot, indem ein Gleichnis mit einer Straße dargestellt wurde. Die Kernaussage war, dass man keine Straße überqueren würde wenn man nur einen 5 Minuten alten Schnappschuss des Verkehrs zur Verfügung hätte.[7]

[7] http://www.dailymotion.com/video/xdaoae_ibm-commercial-the-road-intelligent_tech

Als weiteres Beispiel kann man sich ein Handelsgeschäft an den kritischen Tagen vor Weihnachten vorstellen, welches die Standortdaten aus den Smartphones der Kunden benutzt, die auf dem Parkplatz parken, um die zu erwartenden Verkäufe berechnen zu könne. Mit den errechneten Verkäufen könnten dann zeitnah die notwendigen Lagerbestände berechnet werden. Dadurch könnten die Lagerbestände variabel und zügig anpasst und das gebundene Kapital verringert werden.

Es gibt ganze Industriezweige welche sich schon länger auf diese Gebiete spezialisiert haben, so wird in den Finanzmärkten auf High Frequency Trading oder kurz HFT[8] bzw. Low Latency Trading[9] gesetzt. Hierbei läuft die Datenverarbeitung in (fast) Echtzeit ab und es zählt jeder Sekundenbruchteil, den ein Event an Zeit kostet. Die Konsequenz muss daher sein, dass die Verarbeitungsgeschwindigkeit mit dem Datenaufkommen Schritt zu halten hat. Doch nicht immer steht die Schnelligkeit im Vordergrund, denn diese schränkt die Konsistenz der Daten ein und bestimmte Anwendungsfälle fordern die Vollständigkeit.

Die Herausforderungen:

- Analysen großer Datenmengen mit Antworten im Sekundenbereich
- Datenverarbeitung in Echtzeit
- Datengenerierung und Übertragung in hoher Geschwindigkeit

2.3 Variety (Vielfalt)

Die Datenstrukturen haben eine Vielzahl von Formen angenommen. Mit neuen Anwendung entstehen in der Regel neue Datenformate, da diese speziell auf die Anforderungen der Anwendung oder des Benutzers angepasst sind. Nicht nur die Vielzahl an Formaten auch der Grad der Ordnung, die die Strukturen aufweisen, nimmt zunehmend ab. Selten präsentieren sich Daten daher heute in einer Form die dafür geeignet ist diese direkt zu verarbeiten. Text, SMS, Pdf, Web- Inhalte, Flash, Fotos, Audio- Inhalten, Sensordaten, GPS- Daten, Dokumente um nur einige zu nennen, stellen neue Anforderungen dar um diese effektiv verarbeiten zu können.

[8] http://de.wikipedia.org/wiki/Hochfrequenz-Handel

[9] http://en.wikipedia.org/wiki/Low_latency_(capital_markets)

Vor allem sind die Strukturen der Daten zum Großteil nicht dazu geeignet um sie in relationalen Schemen abzubilden. Tabellen und relationale Datenbanken sind nur schlecht dafür geeignet um mit „unstrukturierten“ Daten zu arbeiten. In relationalen Datenbanksystemen werden Datensätze mit Hilfe von Relationen abgespeichert. Dies kann man sich als eine Sammlung von Tabellen vorstellen, in welchen Datensätze (Tupel) abgespeichert sind. Das Relationale Schema legt dabei die Anzahl und den Typ der Attribute für eine Relation fest. Die zu speichernde Datei muss dabei dieser Struktur entsprechen.

Zwar kann man einen Text zum Beispiel als Blob als Teil eines Tupel abspeichern, jedoch wäre eine Auswertung in traditionellen relationalen Datenbanken nicht möglich. Es wurde und wird immer noch versucht Datenformate und er deren Inhalte so umzuwandeln, dass diese in vorgegebene Schemen passen. Mit diesem Übergang von Quelldateien zu Dateien, die das verarbeiten erlauben, geht aber meist ein Verlust von Informationen einher. Ein Grundsatz von „Big Data“ ist jedoch, wenn möglich alle Daten zu behalten. Es muss sich mit dem Chaos abgefunden werden welche diese Quelldaten mit sich bringen und neue Ansätze in Betracht gezogen werden um den Informationsverlust, wenn möglich, zu vermeiden. Um den richtigen Ansatz (Datenhaltung, Datenverarbeitung) zu finden muss daher klar sein um welche Art von Daten es sich handelt. Man unterscheidet dabei grob Drei Kategorien (siehe Abbildung 2):

- strukturierte Daten: Datensatz (Tupel) in einer relationalen Datenbank
- semi- strukturierte Daten: Email, bestehen aus Anschrift, Sender und Absender (strukturiert) und Text (unstrukturiert)
- unstrukturierte Daten: Ein Text kann eine beliebige Struktur aufweisen

„Im Rahmen von „Big Data“ werden alle vorhandenen Daten, ob strukturiert oder nicht, zusammengefasst und gemeinsam analysiert.“[10] Dabei stellen diese Kategorien jedoch unterschiedliche Anforderungen an die Technologien um ein optimales Ergebnis zu erreichen, sei es bei der Verwaltung oder der Analyse. Für die Analyse von Daten wird

[10] http://www.gi.de/nc/service/informatiklexikon/detailansicht/article/big-data.html

verstärkt auf Ansätze wie Statistik, Maschine Learning und Natural Language Processing gesetzt.

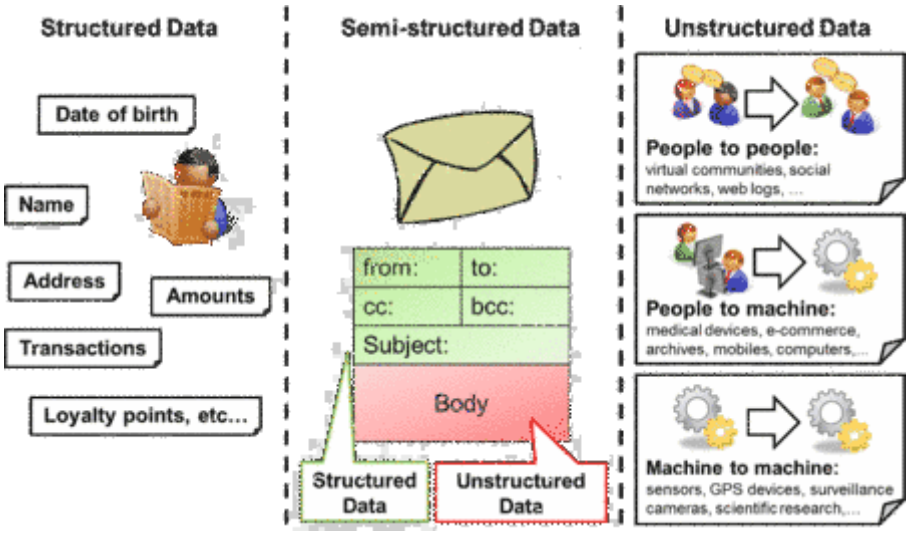

Abbildung 2: Kategorien von Datenstrukturen

2.4 Veracity (Zuverlässigkeit)

Wie bereits erläutert wurde, je größer der „Datentopf“ ist aus dem ein Unternehmen Informationen gewinnt, desto genauere sind diese. Daher sollte auch versucht werden so viele Quellen wie möglich zu nutzen. Das Problem ist, dass die Daten welche das Unternehmen verstehen möchte, dann möglicherweise verunreinigt sind mit Daten, welche ungenau oder falsch sind, und Daten, welche für das Unternehmen nicht von Interesse sind.

Als Beispiel soll hier Twitter genannt werden, das dazu genutzt werden kann Nutzerfeedbacks zu Produkten zu analysieren. Es gibt eine große Anzahl an Spambots auf dieser Plattform, deren Daten uninteressant für das Unternehmen sind. Auch können Konkurrenzunternehmen gezielt Falschinformationen einbetten um die eigene Analyse zu erschweren.

Diese Daten müssen bereinigen werden, oder es entstehen gewisse Unsicherheiten und Ungenauigkeiten. Wie jedoch bereits in Abschnitt 1.2 beschrieben wurde ist die Geschwindigkeit mit der die Informationen aus den Datensätzen gewonnen werden

ein Marktvorteil. So bleibt oft keine Zeit bzw. die Ressourcen sind zu begrenzt um die Datensätze mit geeigneten Analysemethoden zu bereinigen.

Abbildung 3: Verfügbare Datenmenge in Relation zur Verarbeitungskapazität

2.5 Value (Wert)

Dann gibt es noch ein weiteres V (Value) zu berücksichtigen, wenn man von Big Data spricht. Allein der Zugang zu großen Daten und die Möglichkeit diese auszuwerten ist nutzlos, wenn kein neuer Wert daraus geschaffen werden kann. Es ist wichtig, dass die Unternehmen die einzelnen Business Cases konkretisieren und damit bestimmen in welchem Zusammenhang große Daten gesammelt und analysiert werden sollen. Dies sollte eigentlich eine Grundvoraussetzung sein wenn ein „Big Data"- Projekt gestartet wird. Es sollte als erstes ein Verständnis dafür aufgebaut werden, welche Kosten und welcher Nutzen aus einer „Big Data"-Initiativen entstehen soll.

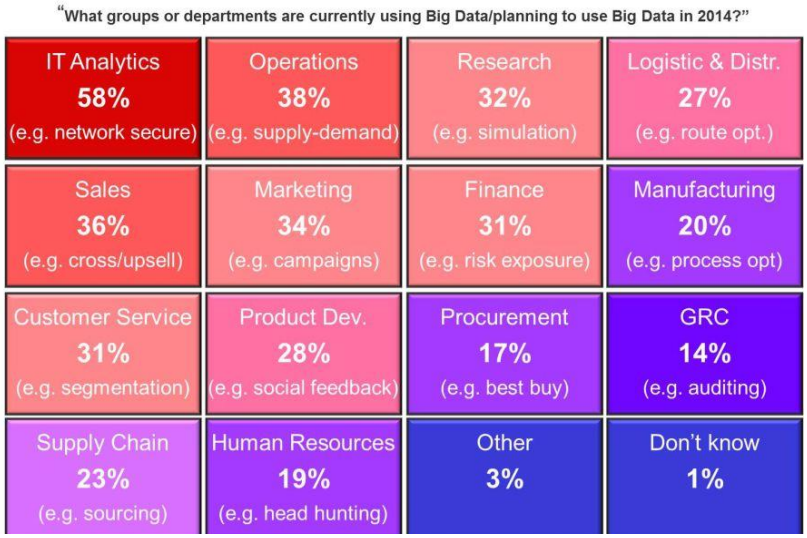

Abbildung 4: Entscheidungsträger aus Business und IT wurden zu verschiedenen Aspekten ihrer Software-Unternehmensstrategie befragt

3 Die Technologieansätze

Je nach Anwendungsfall unterscheiden sich die Anforderungen im Bereich „Big Data“ und es werden verschiedene Architekturen bzw. Kombinationen von Architekturen genutzt um ein Lösung zu bilden. Die Dimensionen von „Big Data“ erlauben dabei eine Einordnung, welcher Technologischer Ansatz zielführend ist (siehe Abbildung 6). Im folgenden Abschnitt wird auf ein die wichtigsten Technologieansätze eingegangen. „Die Vielfalt an Datentypen und Big-Data-Einsatz -Szenarien erfordert auch vielfältige Werkzeuge auf jeder Schicht einer Technologie- Landschaft.“[11](siehe Abbildung 5)

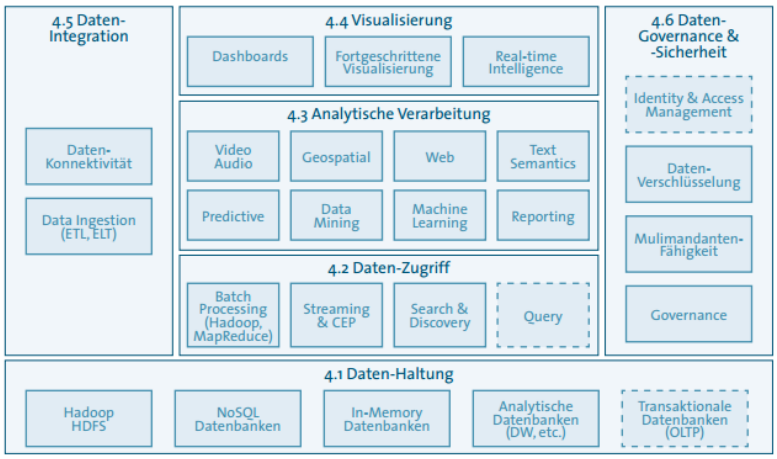

Abbildung 5: Taxonomie von „Big Data“- Technologien

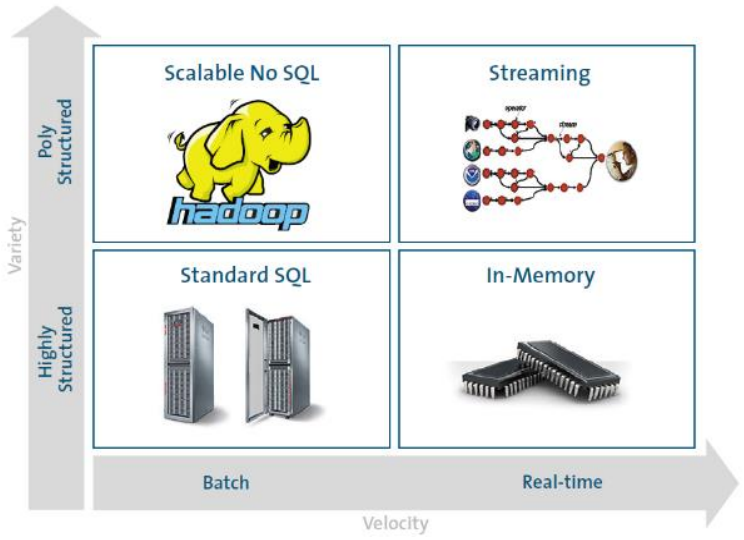

Abbildung 6: Die Dimensionen und Technologieansätze

[11] BITKOM: Leitfaden Big-Data-Technologien-Wissen für Entscheider

3.1 Datenhaltung

Bei der Datenhaltung ist der Zweck das entscheidende Kriterium. Die einzelnen Lösungen unterscheiden sich hinsichtlich der Performance und der Skalierbarkeit. Da die Schichten auf die Datenhaltungs- Schicht aufbauen, stellt diese die Basis dar für alle weiteren Überlegungen. Die wichtigsten Architekturen sollen hier nun erläutert werden.

3.1.1 RDBMS

Das Relationale Datenbanksystem hat auch in „Big Data“ einen festen Platz, jedoch nur dort wo es Sinn macht, insbesondere wenn die Struktur der Daten klar ist und sich nicht häufig ändert, baut man auf RDBMS. Auch bei klassischen Anwendungen ist RDBMS immer noch Maßstab aller Dinge. Wie bereits im Verlauf dieser Arbeit deutlich geworden sein sollte sind RDBMS schlecht darin chaotische und große Daten effektiv und effizient zu verwalten, da diese Systeme strukturierte Daten voraussetzen (spaltenorientiert) und bedingt durch ihre Architektur nur eingeschränkt eine horizontale Skalierung (scale out) erlauben. Genauso sind der vertikalen Skalierung (scale up) Grenzen gesetzt, was den Hauptgrund für den Erfolg der verteilten Systeme (NoSQL) darstellt.

Transaktionale Datenbank

Die Transaktionalen Datenbanken sind für das Einsatz-Szenario Online Transaction Processing (OLTP) optimiert. Hauptschwerpunkt liegt auf schnellen und verlässlichen Operationen, sogenannten ACID-Transaktionen[12], zur Einfügung, Löschung und Aktualisierung von Datensätzen. Diese Datenbanken sind dann effizient, wenn sie für häufige aber kleine Transaktionen oder für große Batch-Transaktionen mit seltenen Schreibzugriffen optimiert sind.

[12] Transaktionen sollen die Anforderungen Atomicity, Consistency, Isolation und Durability erfüllen (Konsistenzmodell).

Die klassischen Analytischen Datenbanken (Data Warehouses)

Analytische Datenbank sind auf das Einsatz- Szenario Online Analytical Processing (OLAP) optimiert. Unternehmen setzen ETL-Technologien ein um in bestimmten Zeitabständen mehrere gegebenenfalls unterschiedlich strukturierte Datenquellen, meist OLTP-Datenbanken (CRM, ERP, usw.), in einem Data Warehouse zu vereinigen. Dort werden sie in Datenwürfeln (OLAP-Würfel) für die Datenanalyse verarbeitet. Das Data Warehouse bildet die Back-end-Infrastruktur für ein breites Spektrum von Technologien zum Management von Kunden, Produkten, Mitarbeitern und Geschäftsprozessen.

3.1.2 NoSQL

NoSQL ist eine Bewegung, welche im Jahre 2009 ihren Anfang nahm. Die Bewegung hat sich zur Aufgabe gemacht die Problemstellung des modernen Datenmanagement ohne relationalen Datenbanken zu lösen. So bedarf es bei einer NoSQL- Datenbanken meist keiner festen Tabellen- Schemata um Daten zu speichern. Festzuhalten ist, der Begriff NoSQL ist keineswegs gegen SQL gerichtet. Dem Begriff wurde daher nachträglich die Bedeutung „not only SQL" zugewiesen. Es geht vielmehr um Alternativen zum relationalen Modell und vor allem zu ACID. In der Zwischenzeit hatte sich durch das CAP-Theorem nämlich herausgestellt, dass die Erfüllung der ACID-Eigenschaften bei verteilten Datenbanken zwangsläufig zu einer Reduktion der Verfügbarkeit führt. Diese Art der Datenbank-Infrastruktur ist jedoch sehr gut an die hohen Anforderungen von „Big Data“ angepasst.

Eigenschaften von NoSQL-Databanken (Diese können je nach Implementierung und Data Store abweichen):

- Skalieren horizontal
- Nicht relational
- Verteiltes System
- Eventually Consistent/ CAP / BASE (nicht wie bei den relationalen Datenbanken ACID)
- Speicherung von großen Datenmengen
- Open- Source
- Commodity Hardware

In verteilten Datenbanken, wie bereits angedeutet wurde, kommt es zu Problemen, wenn alle ACID-Eigenschaften erfüllt werden sollen. Diese Probleme wurden in dem CAP-Theorem von Eric Brewer formuliert. Die Kernaussage des Theorems besagt, dass es drei wesentliche Systemanforderungen für die erfolgreiche Konzeption,

Implementierung und Bereitstellung von Anwendungen in verteilten Computersystemen bestehen. C in CAP beschreibt die Konsistenz, A beschreibt die Verfügbarkeit und P die sogenannte Partition Tolerance. Partition Tolerance beschreibt was passiert, wenn sich die einzeln Knoten in einem verteilten Data Store nicht „sehen“. Nach dem CAP-Theorem ist P konstant und je nach Anforderung wird C oder A priorisiert, so unterscheidet man meist zwischen AP- und CP- Systemen.

CAP:

- Consistency: Alle Replikate von Daten sind identisch und zu einem Zeitpunkt sind für alle Knoten die Daten im gleichen Zustand sichtbar.
- Availability: Ist das System verfügbar wenn eine Anfrage erfolgt? Hat jede Anfrage eine Antwort erhalten?
- Partition Tolerance: Das verteilte System arbeitet auch dann weiterhin wenn einzelne Knoten ausfallen. Das Versagen einzelner Knoten sollte also nicht dazu führen, dass das gesamte System kollabiert. Dies ist eine Zusage von P, die konstant ist. Variabel ist jedoch wie das System auf den Ausfall reagiert.

„Mit CAP geht das sogenannte BASE einher: Basically Available, Soft-state, Eventually consistent.“[13] BASE ist Vergleichbar mit den Zusagen von ACID in einem RDBMS. In verteilten System können meist nur schwächere Garantien in Bezug auf die Konsistenz gegeben werden, wenn die Verfügbarkeit nicht eingeschränkt werden soll, dies wird anhand von BASE deutlich.

BASE:

- **B**asically **A**vailable: Diese Bedingung besagt, dass das System die Verfügbarkeit der Daten in Bezug auf CAP-Theorem garantiert; Es gibt eine Antwort auf jede Anfrage. Die Antwort könnte jedoch falsch sein, da die angeforderten Daten sich in einem inkonsistenten Zustand befinden können.
- **S**oft state: Der Zustand des Systems kann sich im Laufe der Zeit ändern, sogar dann wenn keine Eingaben gemacht werden. Der Grund dafür liegt in der Eventual Consistency. Damit ist der Zustand des Systems immer "Soft" (weich).

[13] Pavlo Baron: Big Data für IT-Entscheider, S.140

- **E**ventual Consistency: Das System wird schlussendlich einen konsistenten Zustand einnehmen, sobald es keinen Input mehr bekommt. Die Daten werden früher oder später an alle, oder verantwortlichen, Knoten propagiert. Konkret macht Eventual Consistency noch mehr Zusagen in verschiedener Form.

Eventual Consistency[14]:

- Read Your Write Consistency: Hat der Client mal eine Version der Daten geschrieben, liest er nur seine Version der Daten und keine fremden, oder er kann die neueste Version sehen, wenn er diese anfordert.

- Monotonic Read Consistency: Client liest nur die Version der Daten, die er zuletzt gelesen hat oder neuere.

- Monotonic Write Consistency: Das System lässt den Client nur aktuellere Daten fortschreiben bzw. niemals ältere Daten schreiben, als die Versionen auf dem System.

Es gibt eine Vielzahl an NoSQL- Datenbanken auf dem Markt, welche auf unterschiedlichste Anforderungen optimiert sind und so z.B. BASE implementieren oder gar ACID, obwohl diese mehr der NewSQL- Bewegung zuzuordnen sind, aber dazu später aber mehr[15]. Natürlich versuchen die Hersteller mit jeder neuen Version neue Features zu implementieren und den möglichen Einsatzbereich der eigenen Applikation zu erweitern. Es gibt jedoch keine „All in One"- Lösung um alle Probleme von „Big Data" zu lösen. Verschiedene Produkte und Technologien müssen miteinander kombiniert werden. NoSQL-Datenbanken sind in der Regel Open Source Softwares und in unterschiedlichen Programmiersprachen geschrieben, dies erschwert oftmals den Einstieg, aufgrund mangelnder Übersichtlichkeit und hochwertiger Einführungen, vor allem für weniger erfahrene Nutzer. Bei NoSQL Data Stores unterscheidet man grundsätzlich zwischen mehreren Arten, auf die hier nun eingegangen wird.

[14] „Pavlo Baron: Big Data für IT-Entscheider, S.142/143" befinden sich Skizzen welche die Vorgänge detaillierte erläutern.

[15] Auf http://nosql-database.org/ findet man eine gute Übersicht der wichtigsten NoSQL-Datenbanken.

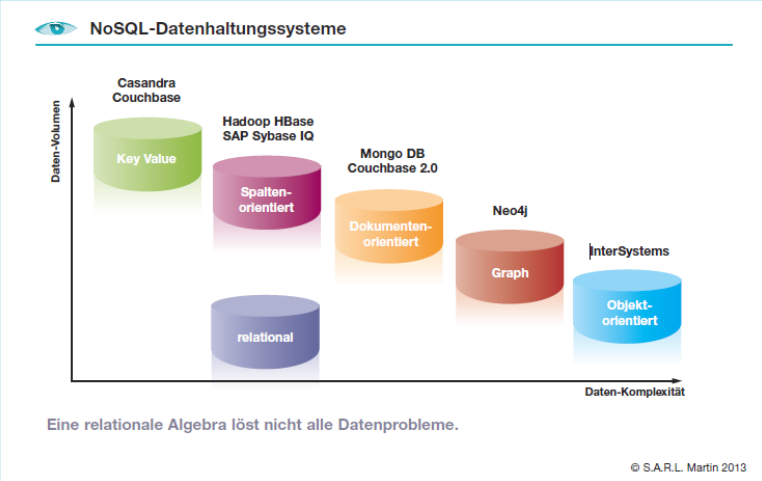

Abbildung 7: Konkrete Tools und der NoSQL-Datenbankentypus

3.1.2.1 Key/Value Stores

Die Key/Value- Datenbanken sind Hashtabellen. Daten besitzen einen bestimmten Schlüssel (Key), einen Hashwert. Dieser Hashwert wird aus der Datei, mit Hilfe von Algorithmen wie SHA- 1, errechnet und ist eindeutig. Mit diesem identifizierenden Hashwert lassen sich spezielle Datenstrukturen aufbauen, die einen sofortigen Zugriff erlauben, ohne dass eine Suche nötig ist. So ist es möglich, eine geeignete Datenstruktur vorausgesetzt, über den Hashwert mit bestimmten Algorithmen den Speicherort zu bestimmen. Dieses Haching kann sogar mit dem Datenspeicher und dem Knoten kombiniert werden (UUID). Die Datenstruktur spielt bei Key/Value-Datenbanken keine Rolle. Ein Key/Value- Datenbank muss nicht auf einem verteilten System arbeiten.

Zu den verbreitersten Vertreter gehört Riak[16]

- Programmiersprache: Erlang
- BASE Implementierung (AP-System, kann jedoch auch anders justiert werden)

[16] http://basho.com/products/riak-overview/

3.1.2.2 Spaltenorientierte Datenbanken

In einer relationalen Datenbank werden Datensätze zeilenorientiert abgespeichert. Bei der spaltenorientierten Speicherung hingegen werden die Werte einer Spalte fortlaufend abgespeichert.

Spaltenorientierte Speicherung ist in analysierenden Applikationen vorteilhaft, weil in diesen Anwendungen häufig eine größere Datenmenge angefragt wird und auf dieser eine statistische Berechnung stattfindet. Weiterhin hat spalten-orientierte Speicherung bei sehr breiten Tabellen einen deutlichen Vorteil, wenn in der Anfrage nur wenige Attribute benötigt werden. Schließlich lässt sich Kompression in zeilenorientierten Systemen nur schwer durchführen, weil in einem Datensatz typischer-weise Werte vieler verschiedener Datentypen gespeichert sind. Durch die Spaltenorientierung hingegen stehen die sehr ähnlichen Werte einer Spalte auch physisch beieinander und bieten eine gute Basis für Kompression.

Zu den bekanntesten Vertreter gehört Cassandra.

3.1.2.3 Document Stores

Dokumentorientierte Datenbanken speichern Dateien weder in Zeilen, noch in Spalten, sondern in einzelnen Dokumenten/Datensätzen. Ein Dokument hat kein festes Schema, das bestimmt, welche Information an welcher Stelle steht. Es gibt auch keine richtige Normalisierung[17] wie bei relationalen Datenbanken. Man versucht im Gegensatz dazu möglichst alle zusammengehörenden Daten in einen Datensatz zu speichern. Dokumentenorientierte Datenbanken sind auf das Speichern von halbstrukturierten Daten ausgelegt und erlauben ein Durchsuchen der Dokumentinhalte.

Beispiele sind MongoDB und Apache CouchDB:

- CouchDB
    - Programmiersprache: Erlang
    - Einschränkungen bei der Lizenz
    - In – Database – MapReduce Funktionen

- MongoDB
    - Programmiersprache: C++
    - Völlig frei einsetzbar

[17] Unter Normalisierung versteht man das Vermeiden von nichtfunktionalen Redundanzen in einer relationalen Datenbank.

- In – Database – MapReduce Funktionen

3.1.2.4 Graphendatenbank

„Graphendatenbanken spezialisieren sich auf vernetzte Informationen und eine möglichst einfache und effiziente Traversierung[18], die sich mit relationalen Datenbanken nur sehr mühselig abbilden lassen. Es existieren viele Anwendungen, die sich auf Graphen zurückführen lassen:

- Hyperlink-Struktur des WWW
- Bedeutung von Seiten für Suchmaschinen (Page Rank)
- "Wer kennt wen"-Beziehungen in sozialen Netzen (kürzeste Wege im Graph)
- Fahr-/Flugplanoptimierung (maximaler Fluss)
- Geoinformations- und Verkehrsleitsysteme (kürzeste Wege)
- …

„[19]

Graphendatenbanken speichern die Daten in einer Netzstruktur, in der die einzelnen Datenelemente durch Knoten mit bestimmten Eigenschaften repräsentiert werden. Diese Knoten sind über Beziehungen, welche ebenfalls Eigenschaften besitzen, miteinander verbunden.

Zu den verbreitersten Vertreter gehört Neo4j:

- Richtung einer Beziehung ist egal nur beim Traversieren
- Zugriff über Java- Schnittstelle
- JVM
- Master/Slave Replikation

3.1.3 NewSQL

NewSQL ist eine Kategorie von SQL-Datenbankprodukten, welche die Probleme in Performance und Skalierbarkeit von traditionellen Transaktionsverarbeitung (OLTP) von relationaler Datenbanksysteme (RDBMS) zu bewältigen versucht. Solche Systeme sollen die Skalierbarkeit der NoSQL-Systeme (scale out) erreichen und dabei immer noch die ACID-Eigenschaften von traditionellen relationalen Datenbanken

[18] Eine Route bestimmen, bei der jeder Knoten und jede Kante eines baumförmigen Graphen genau einmal besucht wird.

[19] Edlich, Friedland, Hampe, Brauer: NoSQL - Einstieg in die Welt der nichtrelationalen Web 2.0-Anwendungen, 2010

gewährleistet können. NewSQL Datenbanken sind vor allem für Unternehmen interessant, welche „High Profile“ Daten verarbeiten, Skalierbarkeit benötigen und höhere Konsistenzanforderungen haben, als diese NoSQL-Datenbanken bieten können. Obwohl die verschiedene NewSQL Datenbanken sich in ihrer internen Architekturen unterscheiden nutzen sie alle das relationale Datenmodell und SQL als Abfragesprache.

Bekannte Vertreter: VoltDB, Google F1 (wird für AdWords genutzt)

3.1.4 In- Memory- Datenbanken und Caching

Eine In-Memory-Datenbank (IMDB) ist ein Datenbankmanagementsystem, das den Arbeitsspeicher eines Computers als Datenspeicher nutzt, dies ermöglicht ein schnelles Speichern und Lesen der Daten. Folglich steigt die Verarbeitungsgeschwindigkeit von Anfragen und verkürzt somit die Antwortzeiten. Um von den Performance-Vorteilen durch In- Memory zu profitieren, müssen jedoch die Applikationen an die neue Technik angepasst werden. Es ist ebenfalls zu bedenken, dass bei vielen In-Memory- Datenbanken Datenverluste in Kauf genommen werden um eine möglichst hohe Performanz zu erreichen. Je nach Ansatz kann eine Priorisierung von Performanz oder auch Datenvollständigkeit im Vordergrundstehen. So würde bei dem „Write Through- Cache“- Prinzip sofort auf die Festplatte geschrieben werden, was jedoch die Performanz einschränken würde, zumindest bei Schreib- Zugriff.

Bekannte Vertreter: SAP HANA, ExaSol, Redis

Redis:

- Open Source
- Key/value Store im Hauptspeicher
- Schnelle Anfragen beantworten und Daten platzieren

In-Memory-Technik bildet einen guten Ansatz, um die Probleme im Bereich Velocity von „Big Data“ entgegen zu wirken.

3.1.5 Hadoop HDFS

Bei HDFS handelt es sich um das verteilte Datensystem des Hadoop-Frameworks. Es wurde von Googles GFS Dateisystem abgeleitet. HDFS verteilt blockweise (Standard sind 64 MB) und gleichmäßig die Daten auf den einzelnen Rechner- Knoten. Die Struktur ermöglicht es das System beliebig zu erweitern. Die redundanten Daten werden nicht auf dem gleichen Knoten gehalten sondern über die Knoten verteilt. Die Datenreplikation ist frei konfigurierbar. Dieses Vorgehen wird gewählt, da bei einem verteilten System die Wahrscheinlichkeit steigt, dass einzelne Knoten ausfallen. Hadoop setzt auf Standard- Hardware, daher ist die Erweiterung des Clusters relativ preisgünstig.

Da es sich um ein Master/Slave- System handelt wird innerhalb des Dateisystems zwischen zwei Diensten unterschieden. So gibt es den NameNode, welcher alle Dateioperationen im Hadoop-Dateisystem kontrolliert und regelt, und den DataNode, welcher die einzelnen zugeteilten Dateiblöcke verwaltet.

NameNode:“

- Speicherung von Metadaten des Dateisystems im Hauptspeicher
- Koordinieren Verteilung der einzelnen Datenblöcke
- Überwachung der einzelnen Rechner-Knoten, um einen Ausfall schnell erkennen zu können„[20]

DataNode:“

- Verwaltung der einzelnen Dateisystem-Blöcke
- Dateitransfer für Replikation der einzelnen Dateisystem- Blöcke
- Zustandsinformationen für die NameNode bereitstellen„[21]

.

4 Das Hadoop- Framework

Bei Hadoop handelt es sich um ein Java- basiertes Open- Source- Framework für die skalierbare und verteilte Verarbeitung von großen Datenmengen. Im Abschnitt 3.1.5 wurde bereits auf das Dateisystem von Hadoop eingegangen, es stellt eines von drei Kernkomponenten von Hadoop dar. Bei den beiden anderen handelt es sich um das MapReduce- Framework und Hadoop Common. Auf Basis des Hadoop- Frameworks ist es möglich bestimmte Arten verteilter NoSQL-Datenbanken (wie HBase)

[20] Roman Wartala: Hadoop – Zuverlässige, verteilte und skalierbare Big- Data- Anwendungen, S. 24
[21] Roman Wartala: Hadoop – Zuverlässige, verteilte und skalierbare Big- Data- Anwendungen, S. 25

aufzubauen. Datensätze können mit einer geringen Reduzierung der Leistung über Tausende von Servern verteilt und verarbeitet werden. Hadoop ist für die Verarbeitung großer Datenmengen im Batch- Prozess ausgelegt und ist somit nicht für CEP (Complex Event Processing), also (fast) Echtzeitverarbeitung, geeignet. Somit ist es nicht das Allheilmittel für „Big Data“, für das es von vielen gehalten wird. Es gibt mehrere kommerzielle Distributionen des Hadoop- Frameworks auf dem Markt, darunter Cloudera, MapR[3] M3 und M5 und Greenplum HD von EMC.

4.1 Hadoop Common

Hadoop Common stellt die Grundfunktionen bereit, die alle anderen Komponenten benötigen. Dazu zählen eine implementierungsneutrale Filesystem-Schnittstelle, die Schnittstelle für die "Remote Procedure Call"- Kommunikation im Cluster und Bibliotheken für die Serialisierung von Daten.

4.2 Map Reduce

Mit MapReduce sind oft zwei Dinge gemeint. Zum einen versteht man es als Programmiermodell und zum anderen bezeichnet man damit MapReduce-Frameworks. Letztere arbeiten nach dem MapReduce- Modell, unterscheiden sich aber durch die Wahl der Programmiersprache und in den Implementierungsdetails. Der Ansatz des Programmiermodells geht ursprünglich auf eine Idee von Jeffrey Dean und Sanjay Ghemawat von Google zurück.[22] Diese Systemarchitektur nutzte Doug Cutting zur Realisierung von Hadoop. Aufbauend auf das verteilte Dateisystem wurde der Map Reduce Ansatz realisiert.

4.2.1 Funktionsweise

Im Kern besteht der Map Reduce Ansatz aus eigenständigen Funktion, die nacheinander, in einer sogenannten Pipeline, auf die Datensätze angewendet werden. Sowohl Map als auch Reduce stellen eigenständige benutzerdefinierte Funktionen dar. Dies sind jedoch nicht die einzigen Funktionen, so gibt es noch eine Split- und eine Combine- Funktion. Map Reduce setzt auf die Verteilung der Daten auf mehreren Knoten, so erklärt sich auch die Split- Funktionen. Der Verarbeitungsablauf ist Split,

[22] Jeffrey Dean and Sanjay Ghemawat: MapReduce: Simplied Data Processing on Large Clusters / http:// research.google.com/archive/mapreduce.html

Map, Combine und Reduce. Die Hadoop- Dienste JobTracker und TaskTracker übernehmen die Aufgabe, die einzelnen Funktionen abzuarbeiten. „Die TaskTracker laufen im Normalfall auf einem DataNode und führen Map- und Reduce- Tasks auf einem Rechner- Knoten aus.“[23] „Ein TaskTracker meldet sich periodisch bei einem JobTracker, welcher die Verteilung der einzelnen Verarbeitungs-Jobs im Hadoop-Cluster übernimmt. Der JobTracker fragt bei der NameNode an, auf welchen DataNodes sich die Daten für die Verarbeitung der einzelnen Jobs befinden. Dann sucht der JobTracker die den Daten nächsten TaskTracker- Instanz aus, die am wenigsten ausgelastet ist, um auf dieser die dem Job zugeordneten Map- und Reduce-Tasks auszuführen.“[24]

Die vier Phasen von Map Reduce :

- **Split:** Teilt eingehende Daten in Blöcke auf und verteilt diese auf verschiedene Knoten im Cluster. In Hadoop wird diese Funktion mit der Map-Funktion zusammengefasst, da diese Phase bereits zum Zeitpunkt des Speicherns durchgeführt wurde (siehe 3.1.5).
- **Map:** Macht aus einer Liste von Key/Value- Paaren eine andere Liste, die aus den zu verarbeitenden Key/Value- Paare besteht, indem sie auf jedes Element der Originalliste eine benutzerdefinierte Funktion anwendet. (Die Eingabedaten lassen sich durch das Hadoop-Framework als Key/Value-Paare aufbereiten)
- **Combine:** Reduzierung der Daten auf dem Mapper
- **Reduce:** Reduziert die Liste auf weniger Werte, andere Werte oder einen einzigen Wert.

Hier wurde nun mehrmals von benutzerdefinierten Funktionen gesprochen. Diese stellen das Gegenstück zu einer SQL- Abfragen dar, nur dass in der Basis- Version von Hadoop diese in Java formuliert werden müssen.

[23] Ramon Wartala: Hadoop – Zuverlässige, verteilte und skalierbare Big- Data- Anwendungen, S.29

[24] Ramon Wartala: Hadoop – Zuverlässige, verteilte und skalierbare Big- Data- Anwendungen, S.29/30

5 Fazit

Die IT- Branche ist dafür bekannt mit Akronymen und Modewörtern, wie SOA, Green IT, Cloud und ähnlichem um sich zu schmeißen. Auch gibt es immer wieder Trendthemen die genauso schnell verschwinden, wie sie aufgetaucht sind, daher verwundert es nicht, dass in der Diskussion um „Big Data“ immer mal wieder das Wort Hype fällt.

Ja, „Big Data“ ist ein Modewort, von Personen und Unternehmen forciert, die sich Profite durch Verkauf von Produkten, welche all die Probleme im Bereich Datamanagement lösen sollen. Und doch sind die Technologien und Ansätze, die sich hinter diesem Begriff verbergen bereits heute ein wesentlicher Erfolgsfaktor.

Nein, „Big Data“ ist kein Trendthema, denn Informationsgewinnung und Informationsanwendung sind keine neuen Themen. Die exorbitante Masse an Daten, die analysiert werden kann ist eine Herausforderung, die es zu lösen gilt, und genau dafür steht „Big Data“. Es ist anzunehmen, dass die Problemstellungen in diesem Bereich nicht abnehmen, sondern an Komplexität zunehmen werden. Die innovativen Ansätze – Nutzung von NoSQL Datenbanken und unteranderem Hadoop (ist bereits ein Quasistandard) - werden oder sind bereits zentral für ein Unternehmen um das volle Potential aus den Daten zu holen. Bei all diesen analytischen Möglichkeiten darf aber nie vergessen werden, dass Schlussendlich immer noch der Mensch entscheiden sollte. Der Markt von „Big Data“- Lösungen ist unübersichtlich und unstrukturiert, unteranderem deswegen weil es noch ein sehr junger Markt ist. Wenn sich mit „Big Data“ beschäftigt wird muss man daher auf technischer Ebene polyglott aufgestellt sein, weshalb auch die Nachfrage nach versierten Personal steigt. So bietet „Big Data“ nicht nur für Unternehmen ein wirtschaftlicher Mehrwert sondern auch für den einzelnen eine Chance.

Literaturverzeichnis

Baron, Pavlo. *Big Data für IT- Entscheider - Riesige Datenmengen und moderne Technologien gewinnbringend nutzen*. Carl Hanser Verlag, 2013.

„Big Data."

Verfügbar unter: http://www.gi.de/nc/service/informatiklexikon/detailansicht/article/big-data.html

Stand: 29.05.2014

„Big- Data- Technologien - Wissen für Entscheider." Leitfaden. 2014.

Verfügbar unter: http://www.bitkom.org/files/documents/BITKOM_Leitfaden_Big-Data-Technologien-Wissen_fuer_Entscheider_Febr_2014.pdf

„Big data: The next frontier for innovation, competition, and productivity." 2011.

Verfügbar:
http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation

Edlich, Friedland, Hampe, Brauer. „NoSQL - Einstieg in die Welt der nichtrelationalen Web 2.0- Anwendungen." 2010.

Laney, Doug. „Application Delivery Strategies." 2001.

Verfügbar unter: http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf

Wartala, Ramon. *Hadoop - Zuverlässige, verteilte und skalierbare Big- Data-Anwendungen*. Open Source Press, 2012.

Abbildungsverzeichnis