



# Vergleich von Streamingframeworks: STORM, KAFKA, FLUME, S4

vorgelegt von

Eduard Bergen

Matrikel-Nr.: 769248

dem Fachbereich VI – Informatik und Medien –  
der Beuth Hochschule für Technik Berlin vorgelegte Masterarbeit  
zur Erlangung des akademischen Grades

**Master of Science (M.Sc.)**

im Studiengang

**Medieninformatik-Online (Master)**

Tag der Abgabe 27. Oktober 2014

<b>1. Betreuer</b>	Herr Prof. Dr. Edlich	Beuth Hochschule für Technik
<b>Gutachter</b>	Herr Prof. Knabe	Beuth Hochschule für Technik

Entwurf

---

## Kurzfassung

Mit der enormen Zunahme von Nachrichten durch unterschiedliche Quellen wie Sensoren (RFID) oder Nachrichtenquellen (RFD newsfeeds) wird es schwieriger Informationen beständig abzufragen. Um die Frage zu klären, welcher Rechner am häufigsten über TCP frequentiert wird, werden unterstützende Systeme notwendig. An dieser Stelle helfen Methoden aus dem Bereich des Complex Event Processing (CEP). Im Spezialbereich Stream Processing von CEP wurden Streaming Frameworks entwickelt, um die Arbeit in der Datenflussverarbeitung zu unterstützen und damit komplexe Abfragen auf einer höheren Schicht zu vereinfachen.

## Abstract

Entwurf

Entwurf

---

# Inhaltsverzeichnis

<b>1</b>	<b>Einführung</b>	<b>3</b>
<b>2</b>	<b>Grundlagen</b>	<b>5</b>
2.1	Grundbegriffe . . . . .	5
2.2	Technologie . . . . .	5
2.3	Zusammenfassung . . . . .	5
<b>3</b>	<b>Analyse</b>	<b>7</b>
<b>4</b>	<b>Vor- und Gegenüberstellung Streaming Frameworks</b>	<b>9</b>
4.1	Apache Storm . . . . .	9
4.2	Apache Kafka . . . . .	9
4.3	Apache Flume . . . . .	9
4.4	Apache S4 . . . . .	9
4.5	Zusammenfassung . . . . .	9
<b>5</b>	<b>Anwendungsfall und Prototyp</b>	<b>11</b>
<b>6</b>	<b>Auswertung</b>	<b>13</b>
6.1	Benchmark Ergebnisse . . . . .	13
6.2	Erkenntnis . . . . .	13
<b>7</b>	<b>Schlussbetrachtung</b>	<b>15</b>
7.1	Zusammenfassung . . . . .	15
7.2	Einschränkungen . . . . .	15
7.3	Ausblick . . . . .	15
<b>A</b>	<b>Quelltext zum Prototyp</b>	<b>21</b>

---

Entwurf

---

# Abbildungsverzeichnis

Entwurf

Entwurf

---



# Kapitel 1

## Einführung

Social media streams, such as Twitter, have shown themselves to be useful sources of real-time information about what is happening in the world. Automatic detection and tracking of events identified in these streams have a variety of real-world applications, e.g. identifying and automatically reporting road accidents for emergency services. [MMO<sup>+</sup>13]

Im Internet steigt das Angebot zu unterschiedlichen Informationen rapide an. Gerade in Deutschland wächst das Datenaufkommen, wie die Studie der IDC [Dig14, S. 2-3] zeigt, exponentiell. Dabei nimmt ebenfalls das Interesse an wiederkehrenden Aussagen über die Anzahl bestimmter Produkte, die Beziehungen zu Personen und die persönlichen Stimmungen zueinander zu. So wird in [Dat14] eine interaktive Grafik zum Zeitpunkt der Ansprache zur Lage der Union des Präsidenten der USA angezeigt. Je Zeitpunkt und Themenschwerpunkt wird in der Ansprache zeitgleich die Metrik Engagement zu den einzelnen Bundesstaaten aus den verteilten Twitternachrichten berechnet ausgegeben.

In der Infografik [Jam14b] von Josh James, Firma Domo wird ein Datenwachstum von 2011 bis 2013 um 14,3% veranschaulicht. Es werden unterschiedliche Webseiten vorgestellt. Dabei werden unterschiedlichen Arten von Daten, die pro Minute im Internet erzeugt werden gezeigt. In der ersten Fassung [Jam14a] waren es noch 2 Millionen Suchabfragen auf der Google-Suchseite [Goo14]. Die zweite Fassung gibt über 4 Millionen Suchanfragen pro Minute an. In Facebook [Fac14] konnten in der Fassung mehr als 680 Tausend Inhalte getauscht werden. In der zweiten Fassung werden mehr als 2,4 Millionen Inhalte pro Minute getauscht.

Um die Sicherheit bei Verlust einer Kreditkarte zu erhöhen und gleichzeitig die höchste Flexibilität zu erhalten, gibt es im Falle eines Schadens bei der von unterschiedlichen Orten gleichzeitig eine unerwünschte Banküberweisung stattfindet, für die Bank die Möglichkeit, die Transaktion aufgrund der Positionserkennung zurückzuführen [SCZ05, S. 3, K. Integrate Stored and Streaming Data].

Mit steigenden Anforderungen, wie in der Umfrage [Cap14, S. 8] durch schnellere Analyse, Erkennung möglicher Fehler und Kostenersparnis dargestellt, und damit einem massiven Datenaufkommen ausgesetzt, kann die herkömmliche Datenverarbeitung [CD97, S. 2, K. Architecture and End-to-End Process] durch das Zwischenlagern der Daten in einem Datenzentrum keine komplexen und stetigen Anfragen zeitnah beantworten [MMO<sup>+</sup>13, S. 2 K. Related Work: Big Data and Distributed Stream Processing]. Damit müssen Nachrichten, sobald ein Nachrichteneingang besteht, sofort verarbeitet werden können. Allen Goldberg stellt in [GP84, S. 1, K. Stream Processing Example] anhand eines einfachen Beispiels Stream processing zu deutsch

---

Verarbeitung eines Nachrichtenstroms ausgehend von loop fusion [GP84, S. 7, K. History] vor. Da Allen Goldbergs Beschreibung zu Stream processing in die Ursprünge geht, soll ein einfaches Modell eines Stream processing Systems für die weitere Betrachtung als Grundlage dienen.

So wird in [AAB<sup>+</sup>05, S. 2, K. 2.1: Architecture] die distributed stream processing engine Borealis vorgestellt und als große verteilte Warteschlangenverarbeitung beschrieben. Die Abbildung [AAB<sup>+</sup>05, S. 3, A. 1: Borealis Architecture] zeigt eine Borealis-Node mit Query processor in der Abfragen verarbeitet werden. Eine Borealis-Node entspricht einem Operator, in dem laufend Datentupel sequentiell verarbeitet werden. Mehrere Nodes sind in einem Netzwerk verbunden und lösen dadurch komplexe Abfragen. Damit die Komplexität, die Lastverteilung und somit die Steigerung der Kapazität für die Entwicklung von neuen Anwendungen vereinfacht werden, wurden Streaming frameworks entwickelt. Streaming frameworks stellen auf einer höheren Abstraktion Methoden zur Datenverarbeitung bereit.

Bisher werden einzelne Streaming frameworks separat in Büchern oder im Internet im Dokumentationsbereich der Produktwebseiten vorgestellt. Dabei werden vorwiegend Methoden des einzelnen Streaming frameworks erläutert und auf weiterführenden Seiten vertieft. Als Software Entwickler wird der Nutzen für die Streaming frameworks nicht sofort klar. Zum Teil sind die Dokumentationen veraltet, in einem Überführungsprozess einer neuen Version oder es fehlt ein schneller Einstieg mit einer kleinen Beispielanwendung.

In dieser Arbeit soll es eine Übersicht mit Einordnung und Spezifikation über die einzelnen Streaming frameworks Apache Storm [Mar13], Apache Kafka [KNR13], Apache Flume [PMS13] und Apache S4 [GJM<sup>+</sup>13] geben. Dabei werden außerdem die Streaming frameworks diskutiert und verglichen.

Da viele Begriffe aus dem englischen Sprachraum kommen, wird der englische Begriff kurz erläutert und im weiterführenden Text kursiv gekennzeichnet. Kapitel werden eingeleitet und ausgeleitet.

Die Arbeit ist in zwei Bereiche geteilt. Im ersten Bereich werden Grundlagen geschaffen, es wird analysiert und die Streaming frameworks werden vorgestellt. Im zweiten Bereich werden die Streaming frameworks diskutiert und verglichen. Dabei wird ein praxisnaher Anwendungsfall vorgestellt. Und im Schlussteil wird zusammengefasst, die Erkenntnis vorgestellt und ein Ausblick gegeben.

Das erste Kapitel befasst sich mit der Einführung und im zweiten Kapitel werden die Grundlagen geschaffen. Dabei werden die Grundbegriffe und die zum Einsatz notwendige Technologie vorgestellt, eingeordnet und zusammengefasst. Im dritten Kapitel findet eine Analyse in Verbindung der gewonnen Grundlagen statt. Kapitel Vier stellt einen großen Teil dar. Darin werden die einzelnen Streaming frameworks vorgestellt. Das Kapitel endet mit einer Zusammenfassung. In Kapitel Fünf wird ein Anwendungsfall vorgestellt, in dem die Streaming frameworks im Einsatz gezeigt werden. Das sechste Kapitel knüpft an das vorangegangene an und stellt die Diskussion und den Vergleich. Das letzte Kapitel Sieben enthält die Schlussbetrachtung mit einer Zusammenfassung, einer Erkenntnis und einem Ausblick. In Anhang A sind zusätzliche Inhalte und Quelltexte hinterlegt.

## **Kapitel 2**

# **Grundlagen**

**2.1 Grundbegriffe**

**2.2 Technologie**

**2.3 Zusammenfassung**

Entwurf

Entwurf

---

## **Kapitel 3**

## **Analyse**

Entwurf

Entwurf

---

## **Kapitel 4**

# **Vor- und Gegenüberstellung Streaming Frameworks**

**4.1 Apache Storm**

**4.2 Apache Kafka**

**4.3 Apache Flume**

**4.4 Apache S4**

**4.5 Zusammenfassung**

---

Entwurf

---



## **Kapitel 5**

# **Anwendungsfall und Prototyp**

Entwurf

Entwurf

---

## **Kapitel 6**

# **Auswertung**

### **6.1 Benchmark Ergebnisse**

### **6.2 Erkenntnis**

Entwurf

Entwurf

---

## **Kapitel 7**

# **Schlussbetrachtung**

**7.1 Zusammenfassung**

**7.2 Einschränkungen**

**7.3 Ausblick**

Entwurf

Entwurf

---

# Literaturverzeichnis

- [AAB<sup>+</sup>05] ABADI, DANIEL J, YANIF AHMAD, MAGDALENA BALAZINSKA, UGUR CETIN-TEMEL, MITCH CHERNIACK, JEONG-HYON HWANG, WOLFGANG LINDNER, ANURAG MASKEY, ALEX RASIN, ESTHER RYVKINA et al.: *The Design of the Borealis Stream Processing Engine*. In: *CIDR*, Band 5, Seiten 277–289, 2005.
- [Cap14] CAPITAL, KPMG: *Going beyond the data: Achieving actionable insight with data and analytics*, Januar 2014.
- [CD97] CHAUDHURI, SURAJIT und UMESHWAR DAYAL: *An overview of data warehousing and OLAP technology*. SIGMOD Rec., 26(1):65–74, März 1997.
- [Dig14] DIGITAL, EMC: *Digital universe around the world*, April 2014.
- [GP84] GOLDBERG, ALLEN und ROBERT PAIGE: *Stream processing*. In: *Proceedings of the 1984 ACM Symposium on LISP and functional programming*, LFP '84, Seiten 53–62, New York, NY, USA, 1984. ACM.
- [MMO<sup>+</sup>13] MCCREADIE, RICHARD, CRAIG MACDONALD, IADH OUNIS, MILES OSBORNE und SASA PETROVIC: *Scalable distributed event detection for Twitter*. In: *2013 IEEE International Conference on Big Data*, Seiten 543–549. IEEE, October 2013.
- [SÇZ05] STONEBRAKER, MICHAEL, UĞUR ÇETINTEMEL und STAN ZDONIK: *The 8 requirements of real-time stream processing*. ACM SIGMOD Record, 34(4):42–47, 2005.
-

Entwurf

---



# Internetquellen

- [Dat14] DATA, TWITTER: *State of The Union address 2014* URL: <http://twitter.github.io/interactive/sotu2014/#p1>, May 2014. Abgerufen am 12.05.2014.
- [Fac14] FACEBOOK: *Facebook* URL: <https://www.facebook.com/>, May 2014. Abgerufen am 12.05.2014.
- [GJM<sup>+</sup>13] GOPALAKRISHNA, KISHORE, FLAVIO JUNQUEIRA, MATTHIEU MOREL, LEO NEUMEYER, BRUCE ROBBINS und DANIEL GOMEZ FERRO: *S4 distributed stream computing platform*. URL: <http://incubator.apache.org/s4/>, June 2013. Abgerufen am 30.06.2013.
- [Goo14] GOOGLE: *Google Search* URL: <https://www.google.de/>, May 2014. Abgerufen am 12.05.2014.
- [Jam14a] JAMES, JOSH: *Data Never Sleeps 1.0* URL: <http://www.domo.com/blog/wp-content/uploads/2012/06/DatainOneMinute.jpg>, May 2014. Abgerufen am 12.05.2014.
- [Jam14b] JAMES, JOSH: *Data Never Sleeps 2.0* URL: [http://www.domo.com/blog/wp-content/uploads/2014/04/DataNeverSleeps\\_2.0\\_v2.jpg](http://www.domo.com/blog/wp-content/uploads/2014/04/DataNeverSleeps_2.0_v2.jpg), May 2014. Abgerufen am 05.05.2014.
- [KNR13] KREPS, JAY, NEHA NARKHEDE und JUN RAO: *Apache Kafka is publish-subscribe messaging rethought as a distributed commit log*. URL: <http://kafka.apache.org/>, June 2013. Abgerufen am 29.06.2013.
- [Mar13] MARZ, NATHAN: *Storm is a distributed realtime computation system*. URL: <https://github.com/nathanmarz/storm/wiki/Home/>, June 2013. Abgerufen am 29.06.2013.
- [PMS13] PRABHAKAR, ARVIND, PRASAD MUJUMDAR und ERIC SAMMER: *Apache Flume distributed, reliable, and available service for efficiently operating large amounts of log data*. URL: <http://flume.apache.org/>, June 2013. Abgerufen am 29.06.2013.
-

Entwurf

---

## **Anhang A**

### **Quelltext zum Prototyp**

Entwurf