Seminararbeit im Studiengang Angewandte Informatik (BSc)

Self-Adaptive Architecture in Stream Processing

Version 1.0 vom 25. Dezember 2019
(Vor Abgabe entfernen)

Leon Meister

285631

meister@uni-hildesheim.de

Betreuer:
MSc Cui Qin, SSE

Arbeitsgruppe Software Systems Engineering • Institut fi¿½r Informatik
Universität Hildesheim • Universitätsplatz 1 • D-31134 Hildesheim

Eigenständigkeitserklärung

Erklärung über das selbstständige Verfassen von "Self-Adaptive Architecture in Stream Processing"

Ich versichere hiermit, dass ich die vorstehende Arbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der obigen Arbeit, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen wurden, habe ich in jedem Fall durch die Angabe der Quelle bzw. der Herkunft, auch der benutzten Sekundärliteratur, als Entlehnung kenntlich gemacht. Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet und anderen elektronischen Text- und Datensammlungen und dergleichen. Die eingereichte Arbeit ist nicht anderweitig als Prüfungsleistung verwendet worden oder in deutscher oder einer anderen Sprache als Veröffentlichung erschienen. Mir ist bewusst, dass wahrheitswidrige Angaben als Täuschung behandelt werden.

Hildesheim, den 25. Dezember 2019

Leon Meister

Kurzfassung

Eine kurze Zusammenfassung der Arbeit, die Interesse beim Leser wecken soll.

Abstract

Gerne zusätzlich oder alternativ in Englisch.

Inhaltsverzeichnis

Abbildungsverzeichnis iii

Tabellenverzeichnis iii

Quellcode-Verzeichnis iv

Abkürzungsverzeichnis v

1 Introduction 1

2 Fundamental Concepts 3
- 2.1 Stream Processing . . . 3
  - 2.1.1 Stream Processing Systems . . . 3
  - 2.1.2 Data Stream Management Systems . . . 3
  - 2.1.3 Requirements for Stream Processing Systems . . . 3
- 2.2 MAPE-K Loop . . . 4
- 2.3 Self-Adaptive Systems . . . 4

3 Approaches for Self-Adaptive Architectures in Stream Processing 5
- 3.1 Dhalion . . . 5
  - 3.1.1 An Outline of Heron . . . 5
  - 3.1.2 Dhalion's Architecture . . . 5
  - 3.1.3 Discussion of Dhalion . . . 5
- 3.2 Hierarchical Control Architectures . . . 5
  - 3.2.1 Elastic and Distributed DSP Framework . . . 5
  - 3.2.2 Possible Solutions for Controlling the Adaptation of Data Stream Processing Operators . . . 5
  - 3.2.3 Discussion of EDF . . . 5
- 3.3 Title?? . . . 5

4 Summary And Conclusion 6
- 4.1 Summary . . . 6
- 4.2 Conclusion . . . 6

A Anhang 7
- A.1 Beispiele . . . 7

Glossary 9

Literaturverzeichnis 10

Abbildungsverzeichnis

1.1 The Growth of the Global Datasphere [RGR18, p.6] 1
1.2 The growth of real-time data as part of the Global Datasphere [RGR18, p.13] 1

2.1 Left: An example for an SPS displayed as a directed acyclic graph. Right: Same SPS with introduced parallelity in one operator, marked gray for visibility. Circles are input/outputs, squares are operators, arrows are streams. 3

A.1 Das Logo der SUH 7

Tabellenverzeichnis

A.1 Eine Tabelle 7

Quellcode-Verzeichnis

A.1 HelloWorld 8
A.2 Beispiel eines Log-Eintrags 8

Abkürzungsverzeichnis

DPS	Data Stream Processing
EDF	Elastic and Distributed DSP Framework
IDC	International Data Corporation
SPS	Stream Processing System

1 Introduction

Situation nowadays -> Lots of data (industry 4.0, other use cases, etc.)

Motivation Goal -> Research question definieren und spezifizieren, discuss the different architectures Struktur erlÃ¤utern

The advancements in technology of the past decades has lead to enormous data creation. Technology has become ubiquitous, with the evolution of cell phones to smartphones and the digitilization of industrial processes, Industry 4.0, causing creation of information to grow exponentially. It is estimated that the global datasphere (**Todo: Explain in glossary**) will reach the size of 175 zettabytes by 2025, as shown in figure 1.1.

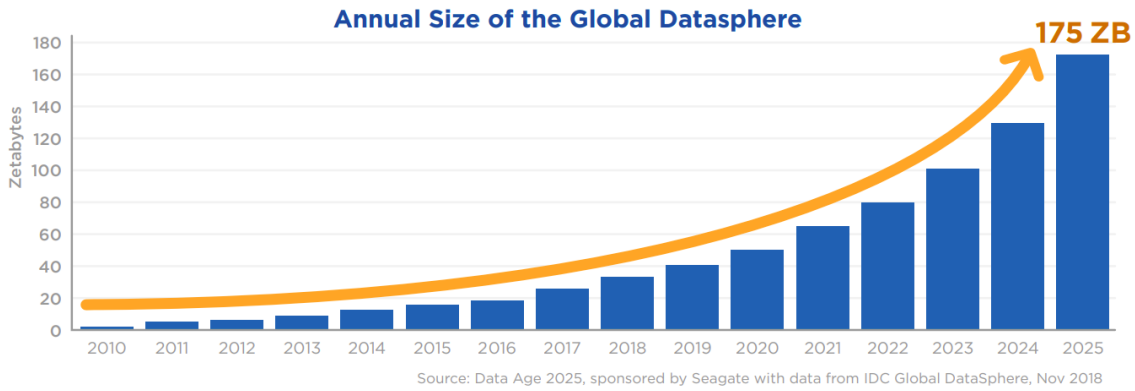

Abbildung 1.1: The Growth of the Global Datasphere [RGR18, p.6]

Data has become an important factor in decision making and optimization in virtually every industry, especially in finances. **TODO: Wieso?** The financial market is dominated by data driven decisions, with emphasis on data processing in a (near) real-time fashion. However, real-time data is becoming of importance in multiple sectors; the International Data Corporation estimates that real-time data will be responsible for a share of 30 percent of the total global datasphere by 2025, as shown in figure 1.2.

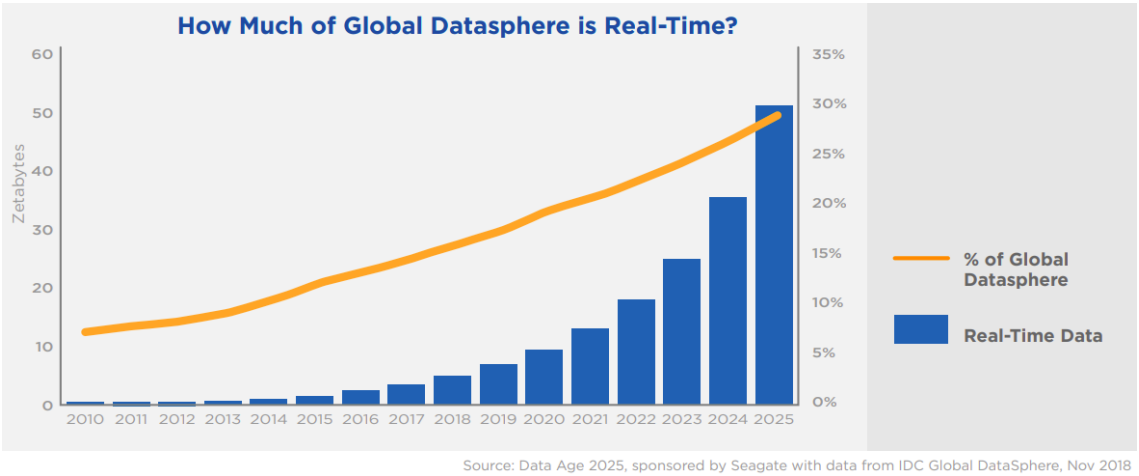

Abbildung 1.2: The growth of real-time data as part of the Global Datasphere [RGR18, p.13]

A global study led by IBM in 2012 has shown that 71 percent of the firms in the financial market use information (including big data) in order to achieve an advantage over their competitors, compared to 36 percent, which IBM has found in an earlier study conducted in 2010. [TSS13, p.1]

As it is no longer feasible to save all the data before then analyzing it (in batches), due to computational cost and lack of storage capacity, a new approach was designed in order to handle data in a (near) real-time fashion, Stream Processing Systems (SPS). **TODO: SPS Traditional DBMS.. another reason and then dsms**

2 Fundamental Concepts

TODO: In this chapter we... This chapter lays out the fundamental concepts that are needed in order to discuss the topic.

2.1 Stream Processing

In this section I will split the concept of Stream Processing into three further components. In 2.1.1 I will then define Stream Processing Systems, explain how they work and give examplary fields of application. Afterwards in 2.1.2 I will then move onto the topic of Data Stream Management Systems and finally in 2.1.3 I will talk about some requirements that SPS should meet.

2.1.1 Stream Processing Systems

This subsection should explain what stream processing systems are, how they work, what kind of SPS are around **TODO: Wie sieht Data in SPS aus? discuss with DBMS approach, too** An SPS takes in one or multiple continuous streams of data, each element of the stream then gets processed by a number of operators and eventually, the SPS puts out a stream of processed data.

In order to increase efficiency, an SPS can, if (computational) resources are available, create replicas of operators to introduce parallelity. Conversely, if there is little input, it may also reduce the amount of replicas in order to save or free up additional resources, as shown in figure 2.1.

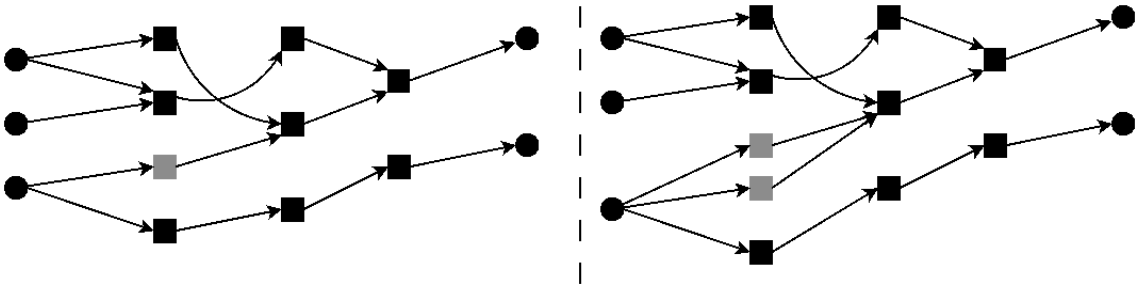

Abbildung 2.1: Left: An example for an SPS displayed as a directed acyclic graph. Right: Same SPS with introduced parallelity in one operator, marked gray for visibility. Circles are input/outputs, squares are operators, arrows are streams.

2.1.2 Data Stream Management Systems

what are dsms, why dsms as opposed to dbms

2.1.3 Requirements for Stream Processing Systems

What are the requirements, why do they matter to us (Elaborate on this)

Due to the nature of the fields in which SPS are used, there are important requirements that SPS should meet in order to be viable, which Stonebraker et al. point out in [ScZ05], of which the ones most important to us can be summarized as the following:

1. **Keep the Data Moving:** In order to minimize latency, data must not be stored, as these are costly operations.

2. **Handle Stream Imperfections:** Expecting only perfect data is utopian, so one must prepare the system with built-in mechanisms for data that might be missing or out-of-order.
3. **Integrate Stored and Streaming Data:** For an SPS to be able to perform comparisons between "predecessor"data and current data, operators must keep an efficiently manageable state.
4. **Guarantee Data Safety and Availability:** Recovering from a failure is detrimental for real-time data processing, so a system must be in place to guarantee the highest availability possible.
5. **Process and Respond Instantaneously:** Systems must be highly optimized in order to provide (near) real-time responses.
6. **Partition and Scale Applications Automatically:** Systems must be able to be split across multiple machines and threads. The system must also be able to automatically scale and distribute the load across the machines.

2.2 MAPE-K Loop

The MAPE-K Loop was introduced by IBM [KC03] and refers to a proposed solution for self-adaptive or autonomic systems. This model has since become the basis or reference architectural pattern for many self adaptive systems, which I will show in the third chapter. The acronym MAPE-K refers to the components that make up the model:

1. **M**onitor: The *Monitor* component gathers data about the system and its environment, aggregates and filters it. As soon as a symptom is encountered that needs to be analyzed, the information is forwarded to the *Analyze* component.
2. **A**nalyze: The *Analyze* component analyzes the previously gathered data and determines whether or not an adaptation should be performed. The decision is made based on performance or cost gain and should include the adaptation cost as well. This component's analysis is influenced by the *Knowledge* base.
3. **P**lan: If the choice to adapt the system has been made, the *Plan* component then decides how to reconfigure the system. Once the decision has been made, the information is then forwarded to the *Execute* component.
4. **E**xecute: Given the *Plan* component's decision, the *Execute* component then executes said plan and the loop returns to the initial monitoring state.
5. **K**nowledge: Represents the knowledge base, which is shared between the other components. This base is created by the *Monitor* component and contains information in the form of metrics, policies, symptoms and logs.

2.3 Self-Adaptive Systems

This subchapter should explain what self-adaptive systems are and how they function. What kind of self-adaptive systems are around?

Cheng et al define self-adaptive systems as

> "[...] systems that are able to adjust their behaviour in response to their perception of the environment and the system itself [...]"[CLG+09, p.1].

3 Approaches for Self-Adaptive Architectures in Stream Processing

Explain that this chapter showcases a few select strategies, which are then elaborated on further in the subchapters Question: Even more approaches? e.g. Master-Slave pattern or Coordinated Control pattern (Both MAPE based)? **Add and explain a few more MAPE Based architectures**

3.1 Dhalion

Quick Introduction to Dhalion, this chapter will deal with the Dhalion paper.

3.1.1 An Outline of Heron

Small outline of Heron, as Dhalion is built on top of Twitter's Heron.

3.1.2 Dhalion's Architecture

Explanation of Dhalion's Architecture **KERNPUNKT DER SECTION DHALION**

3.1.3 Discussion of Dhalion

Discuss the approach and compare it to the reference architecture (Mape?) **TODO: Maybe discuss how they evaluate, look at metrics relevant to architecture**

3.2 Hierarchical Control Architectures

Quick Introduction to hierarchical control architectures, this chapter deals with the Cardellini paper (An example for such an architecture)

3.2.1 Elastic and Distributed DSP Framework

Explanation of the EDF Architecture, their approaches

3.2.2 Possible Solutions for Controlling the Adaptation of Data Stream Processing Operators

3.2.3 Discussion of EDF

3.3 Title??

TODO: Discuss among all of them, critical thinking..

TODO: If enough material compare the architecture relevant metrics of the approaches

4 Summary And Conclusion

4.1 Summary

Summarize the paper

4.2 Conclusion

Conclude the paper

A Anhang

A.1 Beispiele

Zitat ohne Seitenangabe: [**?**]

Zitat mit Seitenangabe: [**?**, S.1]

Referenz eines Glossareintrags: Computer

Eine normale Liste:

- Ein Punkt
- Ein anderer Punkt

Eine nummerierte Liste:

1. Erstens
2. Zweitens
3. ...

Eine Tabelle:

Tabelle A.1: Eine Tabelle

Spalte 1	Spalte 2	Spalte 3
1	2	3

Fetter Text

Kursiver Text

Eine Referenz: Siehe Tabelle A.1: Eine Tabelle auf Seite 7.

Ein Bild:

Abbildung A.1: Das Logo der SUH

Eine abgesetzte Formel:

$$\sum_{n=0}^{3} n = 6 \tag{A.1}$$

Eine Formel im Fließtext: $2^2 = 4$

Ein Listing aus einer Datei:

```
1 public class HelloWorld {
2     public static void main(String[] args) {
3         System.out.println("Hello World!");
4     }
5 }
```

Listing A.1: HelloWorld

Ein 'on-the-fly' erstelltes Listing:

```
1 (Sun Sep 13 23:02:20 2009): ODBC Driver <system32>\wbemdr32.dll not present
2 (Sun Sep 13 23:02:20 2009): Successfully verified WBEM OBDC adapter (incompatible version
      removed if it was detected).
3 (Sun Sep 13 23:02:20 2009): Wbemupgd.dll Registration completed.
```

Listing A.2: Beispiel eines Log-Eintrags

Glossar

Computer ist ein elektronisches Gerï¿½t, das Daten verarbeitet.

Literaturverzeichnis

[CLG+09] Cheng, Betty H. ; Lemos, Rogério ; Giese, Holger ; Inverardi, Paola ; Magee, Jeff ; Andersson, Jesper ; Becker, Basil ; Bencomo, Nelly ; Brun, Yuriy ; Cukic, Bojan ; Marzo Serugendo, Giovanna ; Dustdar, Schahram ; Finkelstein, Anthony ; Gacek, Cristina ; Geihs, Kurt ; Grassi, Vincenzo ; Karsai, Gabor ; Kienle, Holger M. ; Kramer, Jeff ; Litoiu, Marin ; Malek, Sam ; Mirandola, Raffaela ; Müller, Hausi A. ; Park, Sooyong ; Shaw, Mary ; Tichy, Matthias ; Tivoli, Massimo ; Weyns, Danny ; Whittle, Jon: Software Engineering for Self-Adaptive Systems. Version: 2009. http://dx.doi.org/10.1007/978-3-642-02161-9_1. Berlin, Heidelberg : Springer-Verlag, 2009. – DOI 10.1007/978–3–642–02161–9$_1$. – – $ISBN 978 – – 3 – – 642 – – 02160 – – 2, Kapitel Software Engineering for Self – Adaptive Systems : A Research Roadmap, 1 – – 26$

[KC03] Kephart, Jeffrey O. ; Chess, David M.: The Vision of Autonomic Computing. In: *Computer* 36 (2003), Januar, Nr. 1, 41–50. http://dx.doi.org/10.1109/MC.2003.1160055. – DOI 10.1109/MC.2003.1160055. – ISSN 0018–9162

[RGR18] Reinsel, David ; Gantz, John ; Rydning, John: The Digitization of the World. Version: November 2018. https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf. International Data Corporation, November 2018. – Forschungsbericht

[ScZ05] Stonebraker, Michael ; Çetintemel, Uğur ; Zdonik, Stan: The 8 Requirements of Real-time Stream Processing. In: *SIGMOD Rec.* 34 (2005), Dezember, Nr. 4, 42–47. http://dx.doi.org/10.1145/1107499.1107504. – DOI 10.1145/1107499.1107504. – ISSN 0163–5808

[TSS13] Turner, David ; Schroeck, Michael ; Shockley, Rebecca: Analytics: The real-world use of big data in financial services. Version: Mai 2013. https://www.ibm.com/downloads/cas/E4BWZ1PY. IBM, Mai 2013. – Forschungsbericht