Design and Implementation of a Tool to Collect Execution- and Service-Data of Big Data Analytics Applications

Bachelor's Thesis

for obtaining the academic degree
Bachlor of Science (B.Sc.)

at

Beuth Hochschule für Technik Berlin
Department Informatics and Media VI
Degree Program Mediainformatics

1. Examiner and Supervisor: Prof. Dr. Stefan Edlich
2. Examiner: Prof. Dr. Elmar Böhler

Submitted by: Markus Lamm
Matriculation number: s786694
Date of submission: 06.09.2016

Ackknowledgements

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Contents

1 Introduction **1**
1.1 Motivation 1
1.2 Objective 1
1.3 Structure of thesis 2

2 Theoretical Foundations **3**
2.1 Big Data 3
2.2 Big Data Analytics Applications 5
2.3 Stream Processing 7
2.3.1 Apache Flink 8
2.3.2 Apache Kafka 8
2.4 Summary 8

3 Data Analysis **9**
3.1 System data 9
3.2 Java Management Extensions (JMX) 9
3.3 Representational State Transfer (REST) 9
3.4 Data Quality 9
3.5 Summary 9

4 Requirements **10**
4.1 Collection 10
4.2 Transport 10
4.3 Persistence 10
4.4 Summary 10

5 Architecture **11**
5.1 Collected data as time-series based stream 11
5.2 Microservices and Service-Discovery 11
5.3 System components 12
5.3.1 CollectorClient 12
5.3.2 Service-Discovery 12
5.3.3 CollectorManager 12
5.3.4 Message-Broker 12
5.3.5 Indexer 12
5.3.6 Persistence 12
5.4 Summary 13

6 Implementation **14**
6.1 The "collect"-algorithm 14

7 Evaluation **15**
7.1 Local test envirenment 15
7.2 Docker environment 15
7.3 Observations 15
7.4 Discussion 15
7.5 Summary 15

8 Conclusion **16**
8.1 Summary 16
8.2 Outlook 16

List of Figures **A**

List of Tables **B**

List of Source Codes **C**

Bibliography **D**

Articles **E**

Online resources **F**

Image resources **G**

Anhang A **H**

A.1 Diagrams . . . . . H
A.1.1 Use Case diagram . . . . . H
A.1.2 Class diagrams I
A.1.3 Sequence diagrams . . . . . N
A.1.4 Component diagram . . . . . O
A.1.5 Deployment diagram . . . . . P
A.2 Tabelle . . . . . P
A.3 Screenshot . . . . . P
A.4 Graph . . . . . P

Eigenständigkeitserklärung **Q**

1 Introduction

1.1 Motivation

According to a survey in Germany, nine out of ten companies (89 percent) analyze large volumes of data from a variety of diffent sources for operational decision-making processes using modern Big Data Analytics Applications, where 48 percent of the respondents see the greatest potential of Big Data [Jör14a]. The analysis of continuous data streams is taking up a growing importance for companies and therefore constitutes an important factor for business success.

Collecting, storing and analyzing system and operational data of Big Data Applications is therefore an essential tool in order to ensure successful operation and to prevent failures. Even though logfiles are usefull for tracing problems in software systems, problems can be tracked and potential sources of error can be identified much earlier by collecting and storing execution and service data at runtime to describe the state of the system at a given point in time.

Due to the distributed character of Big Data Applications, where a system is composed of several interacting components, the examination of log data is not an adequate choice to gain insight into an entire system [Les14].

1.2 Objective

The main goal of the thesis is the design and implementation of a software system to ingest and store system and operational data of Big Data Analytics Applications on the example of the streaming frameworks Apache Flink and Apache Kafka. It will be examined which

data is available and can be collected at all, what data is relevant and how to collect from source systems. Furhermore, the collected data must be stored in a persistence system to become available for possible consumers like visualization applications, analytical processes or as a data source for applications from the context of Machine Learning for example.

In preparation for this thesis my supervisor Prof. Dr. Stefan Edlich once said *"Sie sammeln alles, was nicht bei drei auf dem Baum ist"*. According to this statement the collection and storage of as much data as possible is the main focus of this work and excludes processing, visualization or analysis of the collected data.

1.3 Structure of thesis

After a short introduction to the topics and the main goals of the present thesis in this chapter, the Chapter 2 covers basic concepts of Big Data Analytics Applications, discusses the concept of stream processing and introduces Apache Flink and Apache Kafka as representatives of widely used stream-processing frameworks.

Chapter 3 investigates which sources for collecting data exist for Apache Flink and Apache Kafka and which data should be collected and stored in a persistence system regarding to its relevance and data quality.

The requirements and the target definition of the software system will be introduced in Chapter 4, Chapter 5 describes the software solution by giving a detailed conceptional overview of the software components whilst Chapter 6 will explain implementation details for selected items.

In chapter 7 we'll see how to setup the technical environment for the usage of the prototype to verify the correct functionality related to the requirements defined in Chapter 4.

The last Chapter 8 covers a conclusion and summary of the present work.

2 Theoretical Foundations

After a short introduction to the terminology of Big Data, this chapter will discuss the main characteristics of Big Data Analytics Applications and introduces the concept of stream processing, which is one of the main characteristics of the popular streaming frameworks Apache Flink and Apache Kafka. The underlying concepts both of these systems and how they're used in context of Big Data Analytics will be explained at the end of this chapter.

2.1 Big Data

According to [Nat15] the term "Big Data" is a misleading name since it implies that pre-existing data is somehow small, which is not true, or that the only challenge is the sheer size of data, which is just one one them among others. In reality, the term Big Data applies to information that can't be processed or analyzed using traditional processes or tools.

In the past decade the amount of data being created is a subject of immense growth. More than 30,000 gigabytes of data are generated every second, and the rate of data creation is only accelerating.[Nat15]. People create content like blog posts, tweets, social network interactions, photos, servers continuously log messages, scientists create detailed measurements, permanently.

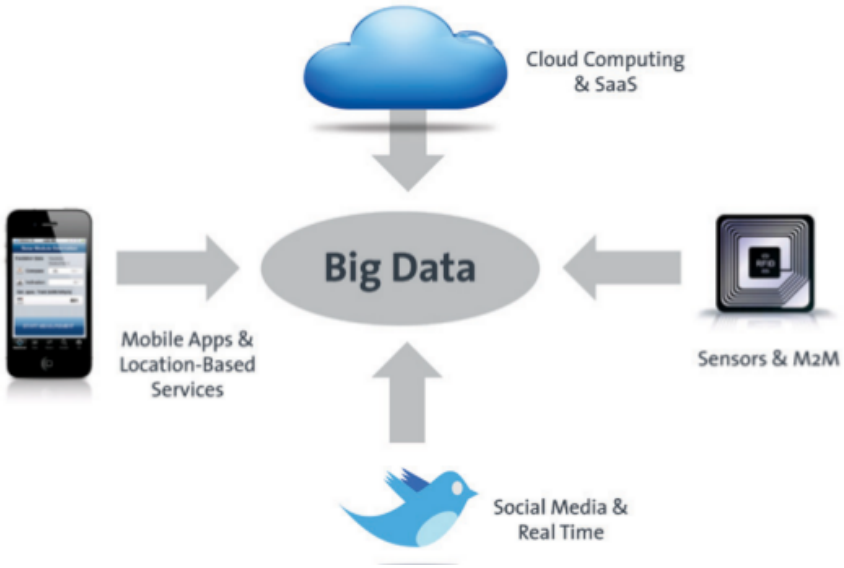

Figure 2.1: Sources of Big Data[Jör14b]

Through advances in communications technology, people and things are becoming increasingly interconnected. Generally referred to as machine-to-machine (M2M), interconnectivity is responsible for double-digit year over year data growth rates. Finally, because small integrated components are now affordable, it becomes possible to add intelligence to almost everything. As an example, a simple railway car has hundreds of sensors for tracking the state of individual parts and GPS-based data for shipment tracking and logistics.[**Ziko12**]

Besides the extremely growing amount of data, an increase in data diversity goes hand in hand. It comes in its raw and unstructured, semistructured or structured form, which makes processing it in a traditional relational system impractical or impossible.[Jör14b] describes, that around 85 percent of the data comes in an unstructured form, but containing valuable information.

According to [Nat15] [**Ziko12**], Big Data is defined by three characteristics:

Volume The amount of data present is growing because of growing amount of producers, e.g. environmental data, financial data, medical data, surveillance data.

Variety Data varies in its form, it comes in different formats from different sources.

Velocity Data needs to be evaluated and analyzed quickly, which leads to new challenges like analysis of large data sets with answers in seconds range, data processing in realtime, data generation and transmission at highspeed.

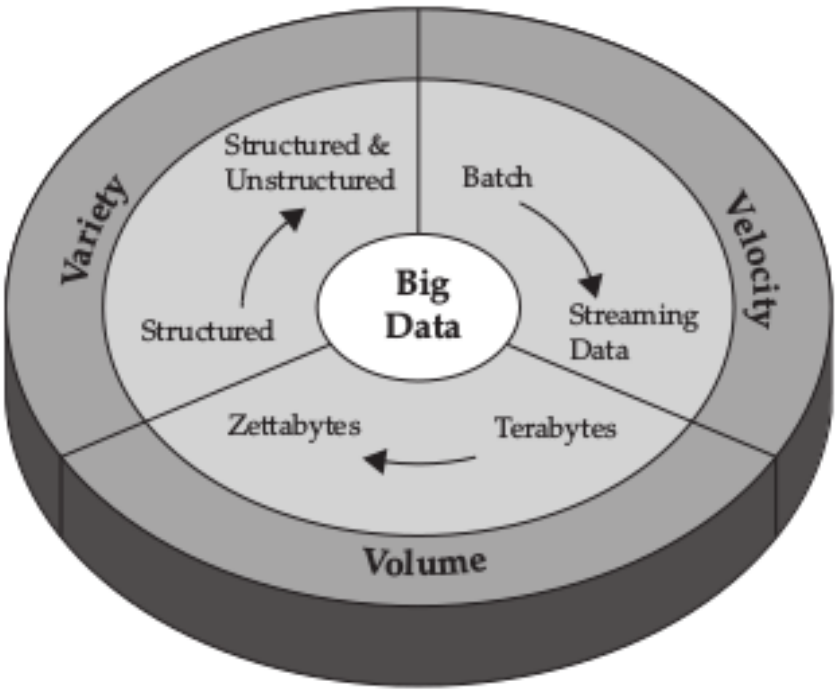

Figure 2.2: The three 'V's of Big Data[**Ziko12**]

A possible definition for Big Data could be derived as follows: *Big Data refers to the use of large amounts of data from multiple sources with a high processing speed for generating valuable information based on the underlying data.*

[Jör14b] proposes another characteristic as a fourth point called "Analytics", which will be explained in the next section.

2.2 Big Data Analytics Applications

Big Data Analytics describes the process of collecting, organizing and analyzing large volumes of data with the aim to discover patterns, relationships and other useful information extracted from incoming data streams [Nat15]. The process of analytics is typically

performed using specialized software tools and applications for predictive analytics, data mining, text mining, forecasting and data optimization.

The analytical methods raise data quality for unstructured data on a level that allows more quantitative and qualitative analysis. With this structure it becomes posssible to extract the data that is relevant by iteratively refined queries.

The areas of applications may be extremely diverse and ranges from analysis of financial flows or traffic data, processing sensor data or environmental monitoring as explained in the previous chapter.

The illustration below summarises the six-dimensional taxonomy [Jör14a; Gro14] of Big Data Analytics Applications.

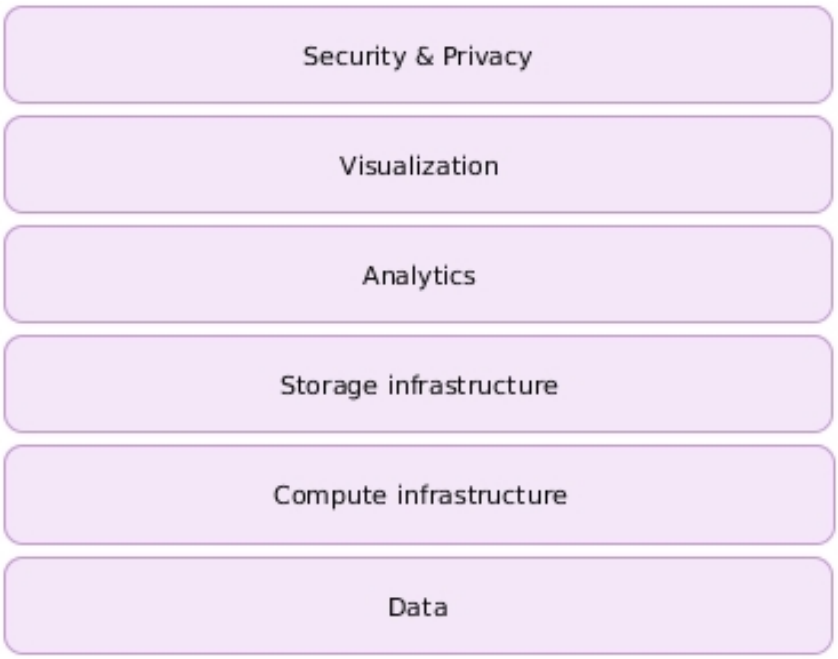

Figure 2.3: Taxonomy of Big Data Analytics Applications [Jör14a; Gro14]

The following section will discus the topic stream processing, which is part of the "Compute infrastructure" layer shown in the figure above,

2.3 Stream Processing

Computing paradigms on big data currently differ at the first level of abstraction on whether the processing will be done in batch mode, or in real-time/near real-time on streaming data. This section is focussed on processing continous data streams in real-time/near real-time and introduces Apache Flink and Apache Kafka as representants of streaming frameworks.

According to [Kle16], stream processing is the real-time processing of data continuously, concurrently, and in a record-by-record fashion in which data is treated not as static tables or files, but as a continuous infinite stream of data integrated from both live and historical sources. It is needed, if "immediate" response to each event as it occurs is demanded by the application. Various data streams could have own features. For example, a stream from the financial market describes the whole data. In the same time, a stream for sensors depends on sampling (e.g. get new data every 5 minutes).

The general approach is to have a small component that processes each of the events separately. In order to speed up the processing, the stream may be subdivided, and the computation distributed across clusters. Stream processing frameworks primarily addresses parallelization of the computational load; an additional storage layer is needed to store the results in order to be able to query them.

Benefits of stream processing:

- Accessibility: live data can be used while still in motion, before being stored.
- Completeness: historical data can be streamed and integrated with live data for more context.
- High throughput: high-velocity and high-volume data can be processed with minimal latency.

In a formal way, a data stream is described as an ordered pair (S, T) where:

- S is a sequence of tuples.
- T is a sequence of positive real time intervals.

It defines a data stream as a sequence of data objects, where the sequence in a data stream is potentially unbounded, which means that data streams may be continuously generated at any rate [Nam15] and leads to the following characteristics:

- the data arives continous
- the arrival of data is disorderen
- the size of the stream is potentially unbounded

After this short introduction to the basics of stream processing, the following sections cover the introduction of the streaming frameworks Apache Flink and Apache Kafka and explains the usage in the context of Big Data Analytics Applications.

2.3.1 Apache Flink

Apache Flink a framework for distributed stream and batch data processing. [Fli16]

2.3.2 Apache Kafka

Apache Kafka is publish-subscribe messaging rethought as a distributed commit log. [Kaf16]

2.4 Summary

3 Data Analysis

3.1 System data

3.2 Java Management Extensions (JMX)

3.3 Representational State Transfer (REST)

3.4 Data Quality

3.5 Summary

4 Requirements

TODO: Based on the topic of this, three main components. Describe general. see [Les14]

4.1 Collection

4.2 Transport

4.3 Persistence

4.4 Summary

realtime?

5 Architecture

Welche Teilprobleme leiten sich aus der Zielstellung weiter ab? Was sind die Rahmenbedingungen für die Probleme und wie können wir diese lösen.

Lösungsbeschreibung, warum so?

Bauen Sie auf der Methodik aus der Einführung auf. Ein System bauen und dann beobachten (machen Sie tens)

z.B. wenn Sie ein System bauen. Wie sieht Ihre Architektur aus? Was sind die Resultate des Entwurfs? Welche Alternativen gibt es zu Ihrem Entwurf? Warum haben Sie sich gerade für Ihre Lösung entschieden? Was sind deren Vor und Nachteile?

Welche Prozesse unterstützt die Architektur? Datenquellen? Transformationen? Datenverarbeitung?

5.1 Collected data as time-series based stream

TODO see [Kle16]

5.2 Microservices and Service-Discovery

TODO

5.3 System components

TODO maybe split Infrastructure / Software components

5.3.1 CollectorClient

The CollectorClient tier is our entry point for bringing data into the system...

5.3.2 Service-Discovery

Registraction for CollectorClients

5.3.3 CollectorManager

Gives overview, uses Consul as service-discovery

5.3.4 Message-Broker

Transport, "Event-Log", see [Kre13]

5.3.5 Indexer

Receive messages from Kafka, roote data, create ES index, why, describe context BDAA

5.3.6 Persistence

ES as search index for time-series based data, easy vizualization with Kibana, why?

5.4 Summary

Maybe Spring alternatives, Lagom, VertX, Play? Maybe collector as agent instead of microservice, alternatives REST, maybe (Web-)Sockets

6 Implementation

Introduce software stack

6.1 The "collect"-algorithm

Eigenschaften des Algorithmus, Komplexität, Wie gehen Sie vor?, Beschreiben Sie was wann passiert

Java8, CPs, non-blocking streams

7 Evaluation

TODO intro

TODO Aufbau der Messumgebung, see docker-deploy

Server/Betriebssystem etc.,Datensätze, Anfragen, Systeme/Ansätze gegen die Sie sich vergleichen, Wie messen Sie? Methodik und Maßeinheiten?

7.1 Local test envirenment

7.2 Docker environment

7.3 Observations

7.4 Discussion

Wurden Sie überrascht? Stimmten Ihre Hypothesen? Sind Sie besser, anders als das andere System? Wichtigster Erkenntnisgewinn 1 Wichtigster Erkenntnisgewinn 2 Wichtigster Erkenntnisgewinn N Anwendbarkeit? Szenario?

7.5 Summary

Beschreibung der Ergebnisse, Diagramme, Darstellen von Zusammenhängen

8 Conclusion

TODO

8.1 Summary

Was war die Zielstellung? Wie war unsere Vorgehensweise? Konnten wir das Problem/die Probleme lösen? Wichtigste Erkenntnisgewinne?

8.2 Outlook

Was würden Sie an dem Thema machen wenn Ihnen jetzt jemand die nächsten drei Jahre finanziert? Was würde Google / Oracle / IBM machen? Sollten wir eigentlich solche Dinge, die Sie in Ihrer Arbeit machen, auch wirklich erforschen oder bauen? Wer verliert dadurch, wer gewinnt?

List of Figures

2.1 Sources of Big Data[Jör14b] 4
2.2 The three 'V's of Big Data[**Ziko12**] 5
2.3 Taxonomy of Big Data Analytics Applications [Jör14a; Gro14] 6

A.1 Use Case Diagramm . . . . . H
A.2 Class diagram 'JvmCollector' I
A.3 Class diagram 'DStatCollector' . . . . . J
A.4 Class diagram 'FlinkRestCollector' . . . . . K
A.5 Class diagram 'FlinkJmxCollector' L
A.6 Class diagram 'KafkaBrokerJmxCollector' L
A.7 Class diagram 'CollectorClient' . . . . . M
A.8 Class diagram 'CollectorManager' . . . . . N
A.9 Sequence diagram 'Client discovery' . . . . . N
A.10 Sequence diagram 'Client scheduling' . . . . . O
A.11 Component diagram . . . . . O
A.12 Deployment diagram . . . . . P

List of Tables

List of Source Codes

Bibliography

[Kle16] Martin Kleppmann. *Making Sense of Stream Processing*. First edition. Sebastopol, CA 95472: O'Reilly Media, Inc., 2016. ISBN: 978-1-491-94010-5.

[Nat15] James Warren Nathan Marz. *Big Data - Principles and best practices of scalable real-time data systems*. Shelter Island, NY 11964: Manning Publications Co., 2015. ISBN: 978-1-617-29034-3.

Articles

[Les14] Tammo van Lessen. “Wissen, was läuft - Mit Laufzeitmetriken den Überblick behalten”. In: *Javamagazin* 10000.11 (2014), pp. 48–52.

[Nam15] Dmitry Namiot. “On Big Data Stream Processing”. In: *International Journal of Open Information Technologies* 1 (2015), pp. 48–51.

Online resources

[Fli16] Apache Flink. *Apache Flink Documentation*. 2016. URL: http://flink.apache.org (visited on 08/18/2016).

[Gro14] Cloud Security Alliance - Big Data Working Group. *Big Data Taxonomy*. 2014. URL: https://downloads.cloudsecurityalliance.org/initiatives/bdwg/Big_Data_Taxonomy.pdf (visited on 08/18/2016).

[Jör14a] u.a Jörg Bartel Axel Mester. *Big Data Technologien – Wissen für Entscheider*. 2014. URL: https://www.bitkom.org/Publikationen/2014/Leitfaden/Big-Data-Technologien-Wissen-fuer-Entscheider/140228-Big-Data-Technologien-Wissen-fuer-Entscheider.pdf (visited on 08/06/2016).

[Jör14b] u.a Jörg Bartel Dr. Bernd Pfitzinger. *Big Data im Praxiseinsatz – Szenarien, Beispiele, Effekte*. 2014. URL: https://www.bitkom.org/Publikationen/2012/Leitfaden/Leitfaden-Big-Data-im-Praxiseinsatz-Szenarien-Beispiele-Effekte/BITKOM-LF-big-data-2012-online1.pdf (visited on 08/06/2016).

[Kaf16] Apache Kafka. *Apache Kafka Documentation*. 2016. URL: http://kafka.apache.org (visited on 08/18/2016).

[Kre13] Jay Kreps. *The Log: What every software engineer should know about real-time data's unifying abstraction*. 2013. URL: https://engineering.linkedin.com/distributed-systems/log-what-every-software-engineer-should-know-about-real-time-datas-unifying (visited on 08/18/2016).

Image resources

[Nat15] James Warren Nathan Marz. *Big Data - Principles and best practices of scalable real-time data systems*. Shelter Island, NY 11964: Manning Publications Co., 2015. ISBN: 978-1-617-29034-3.

A

A.1 Diagrams

A.1.1 Use Case diagram

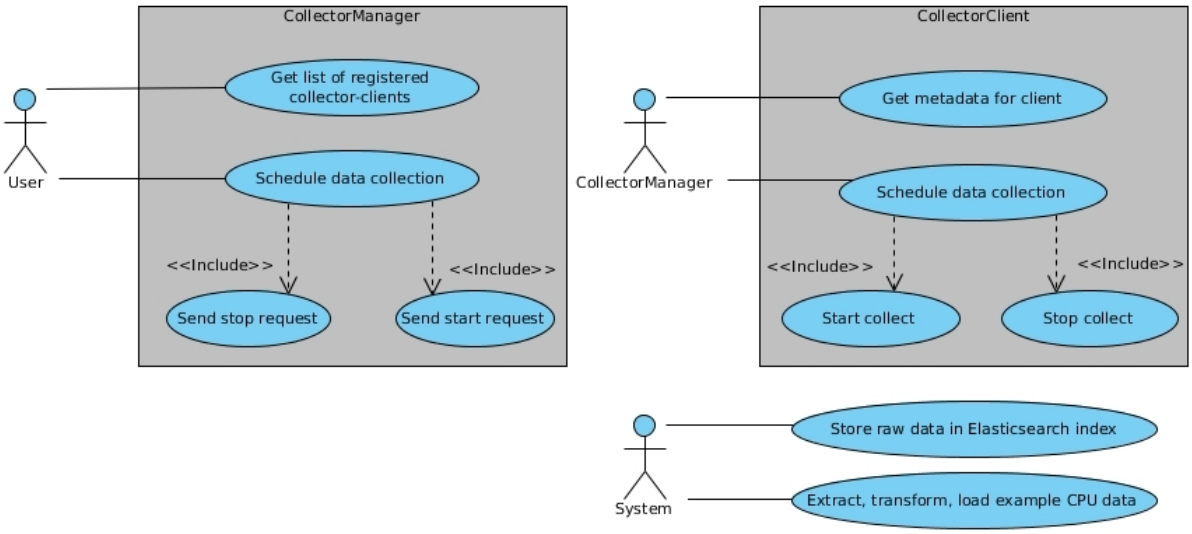

Figure A.1: Use Case Diagramm

A.1.2 Class diagrams

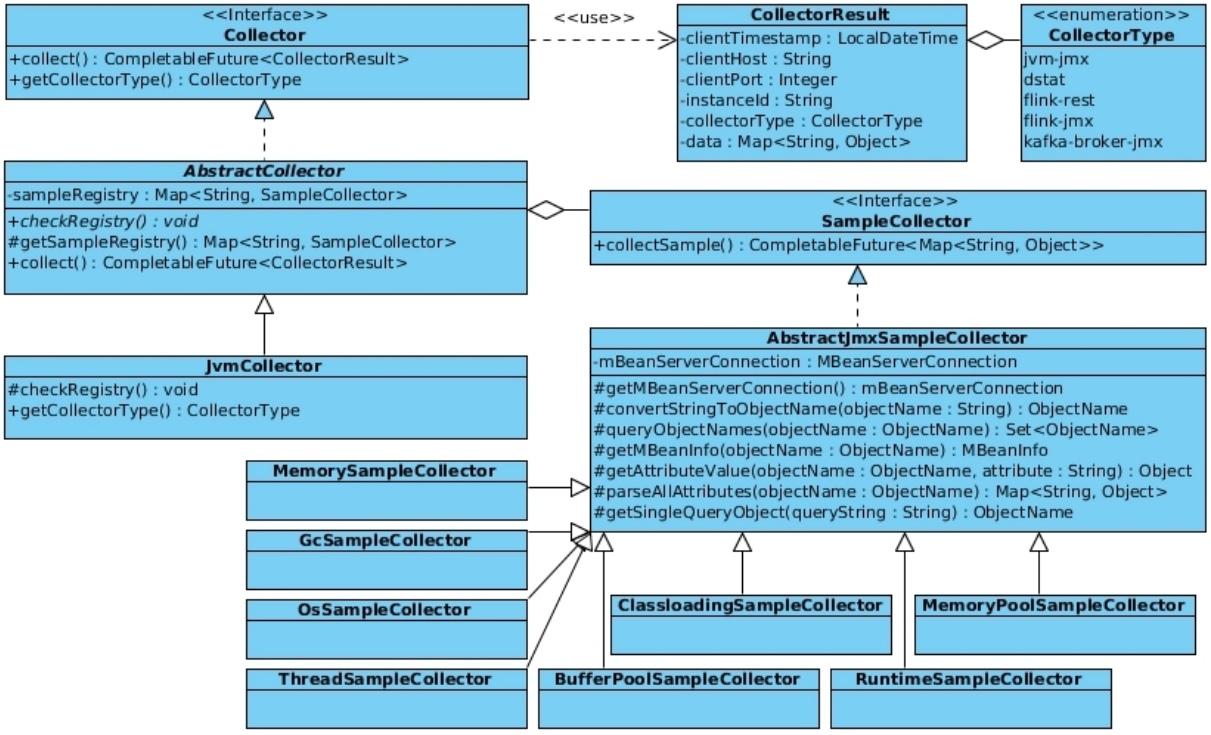

Figure A.2: Class diagram 'JvmCollector'

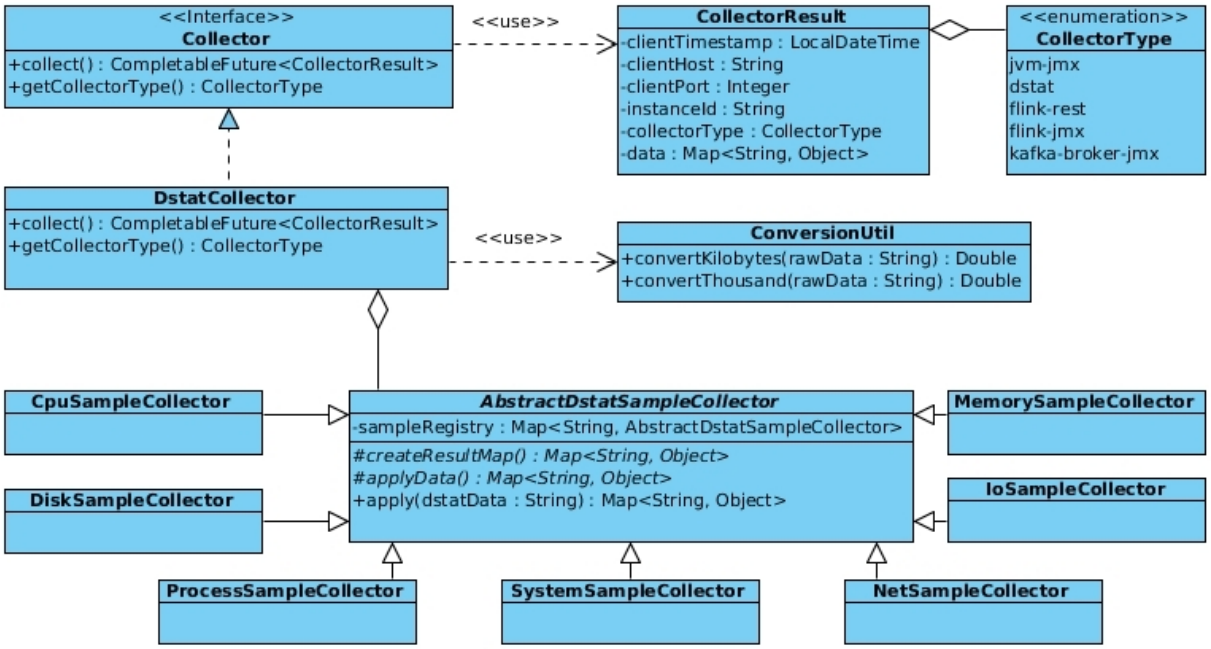

Figure A.3: Class diagram 'DStatCollector'

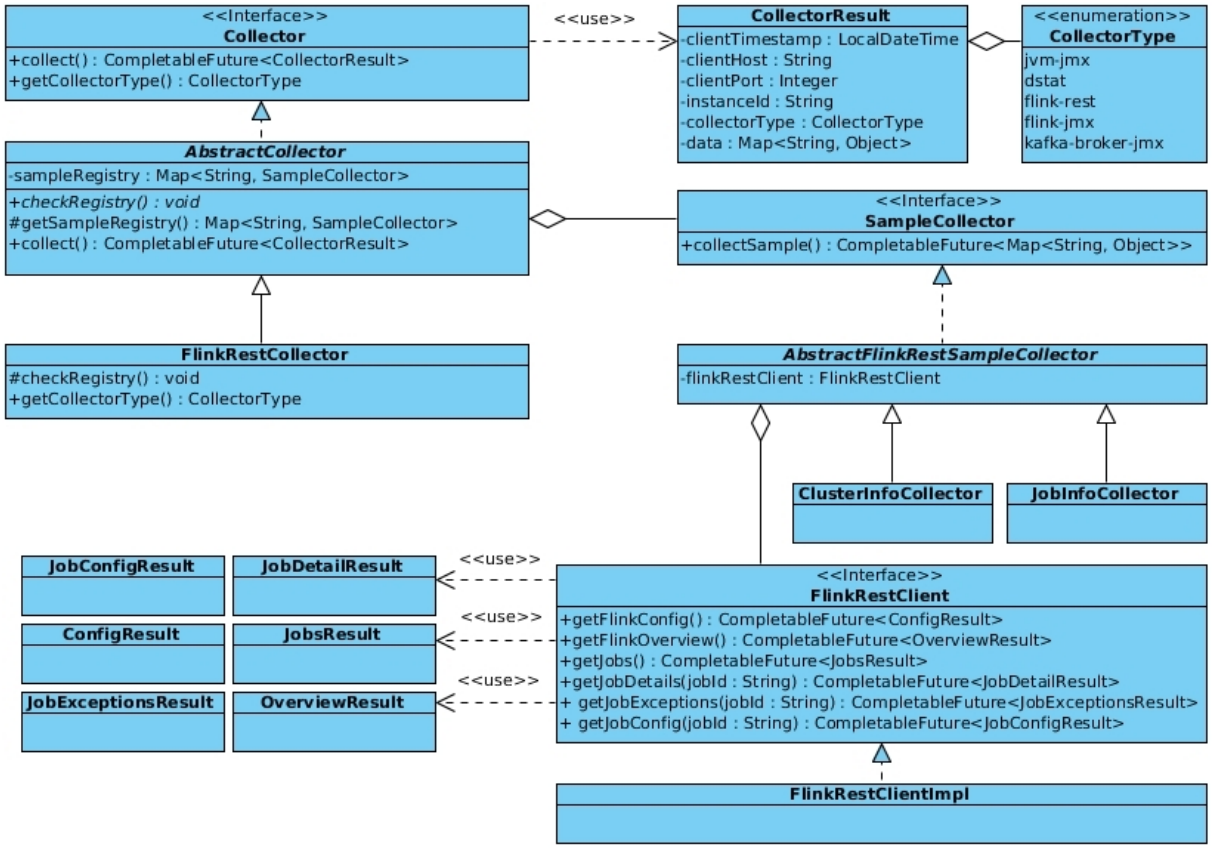

Figure A.4: Class diagram 'FlinkRestCollector'

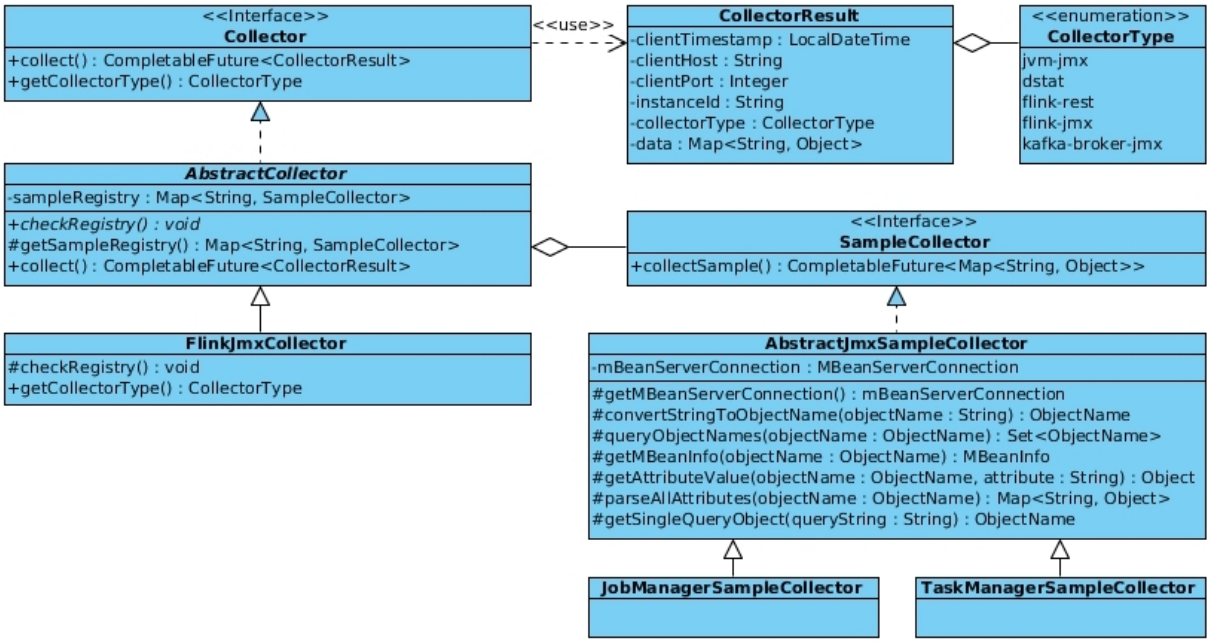

Figure A.5: Class diagram 'FlinkJmxCollector'

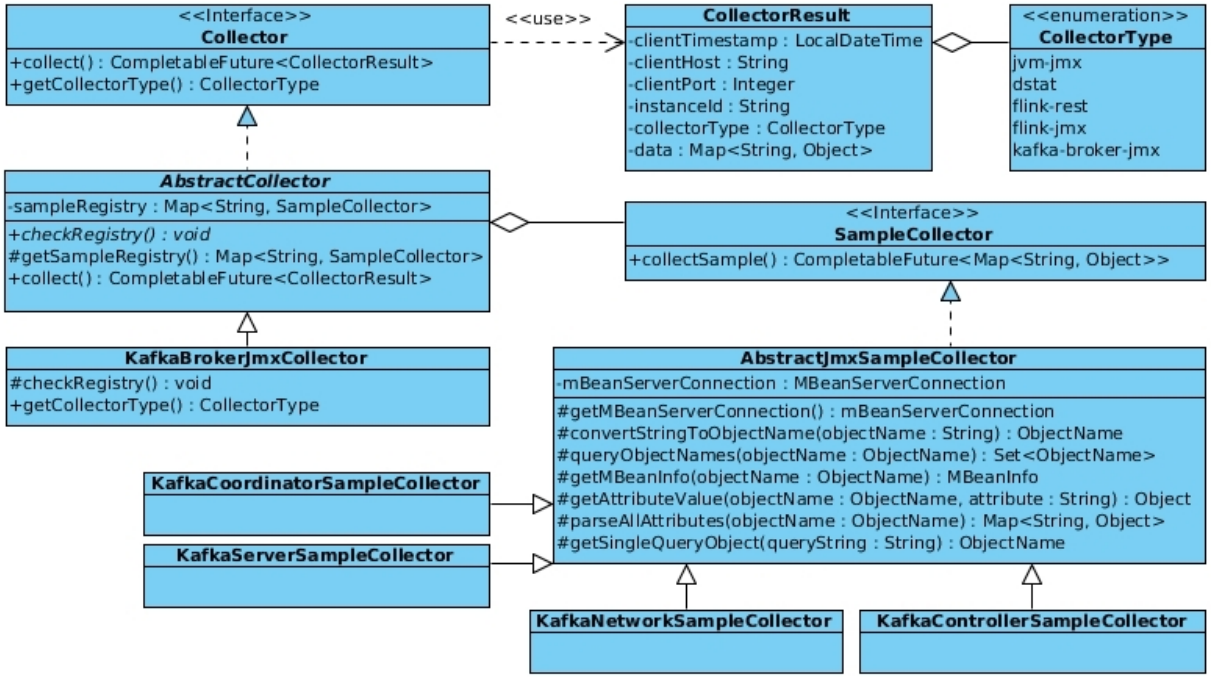

Figure A.6: Class diagram 'KafkaBrokerJmxCollector'

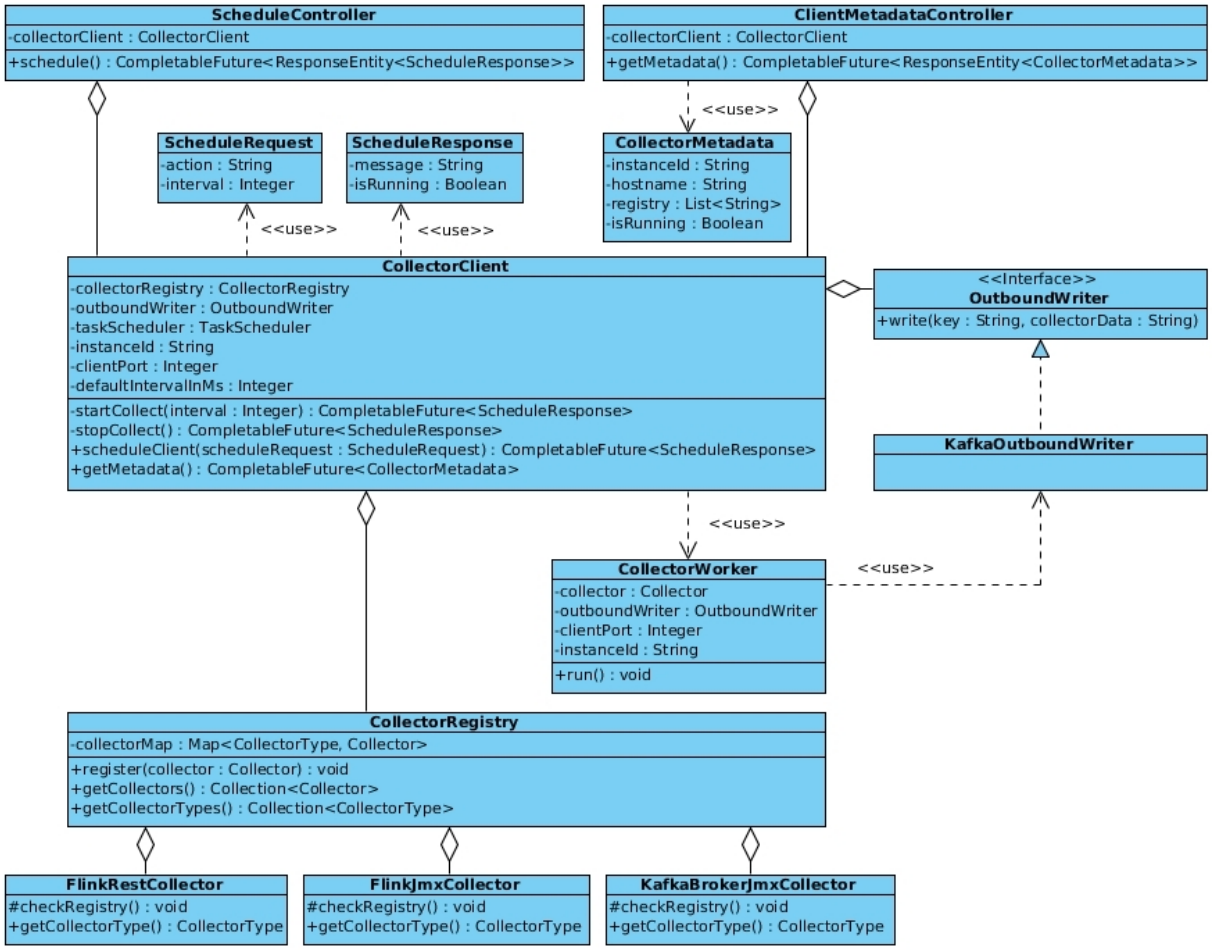

Figure A.7: Class diagram 'CollectorClient'

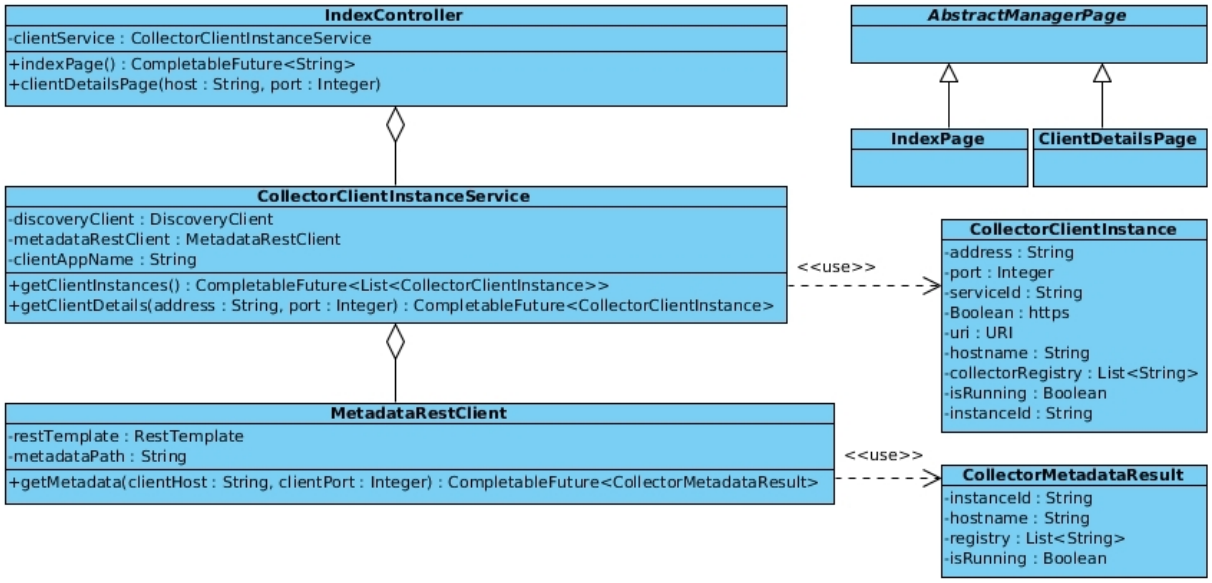

Figure A.8: Class diagram 'CollectorManager'

A.1.3 Sequence diagrams

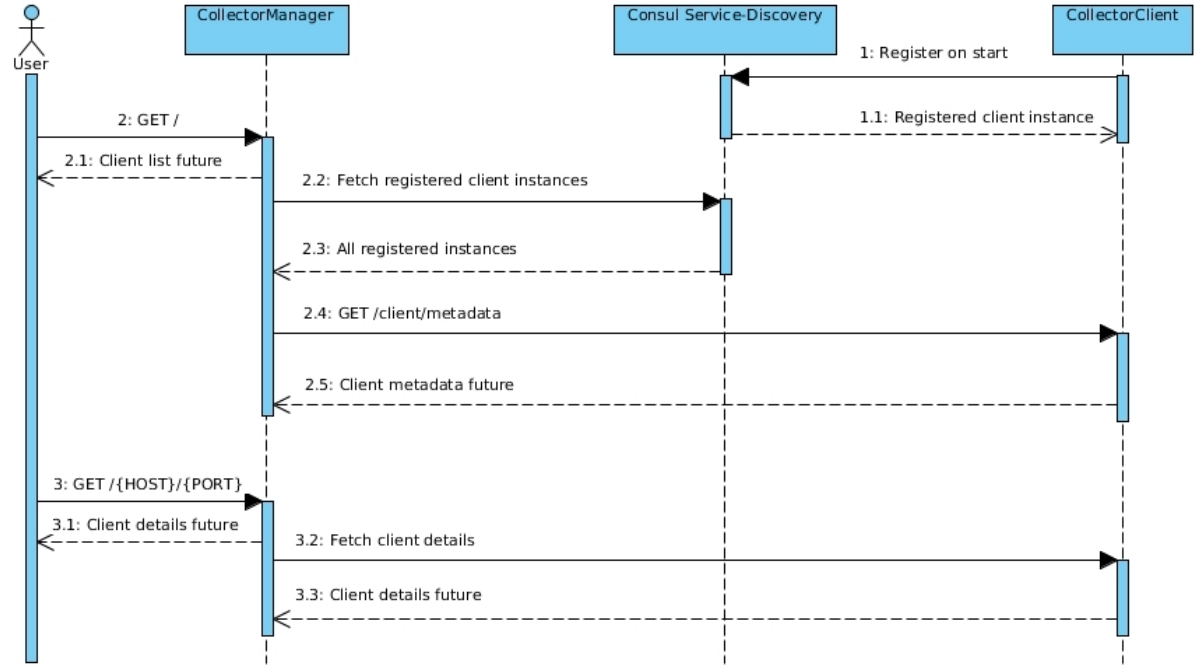

Figure A.9: Sequence diagram 'Client discovery'

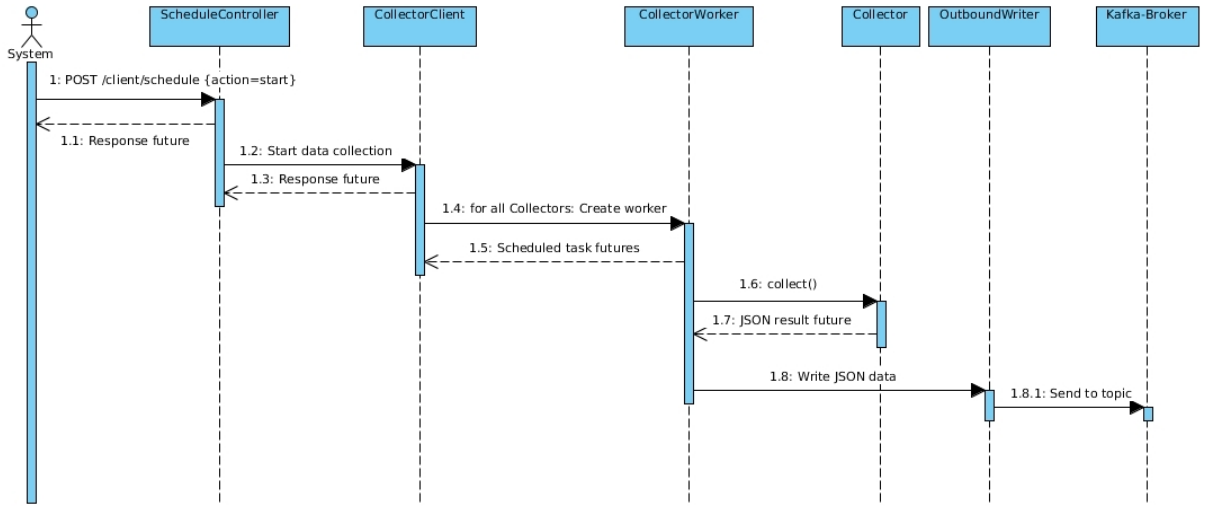

Figure A.10: Sequence diagram 'Client scheduling'

A.1.4 Component diagram

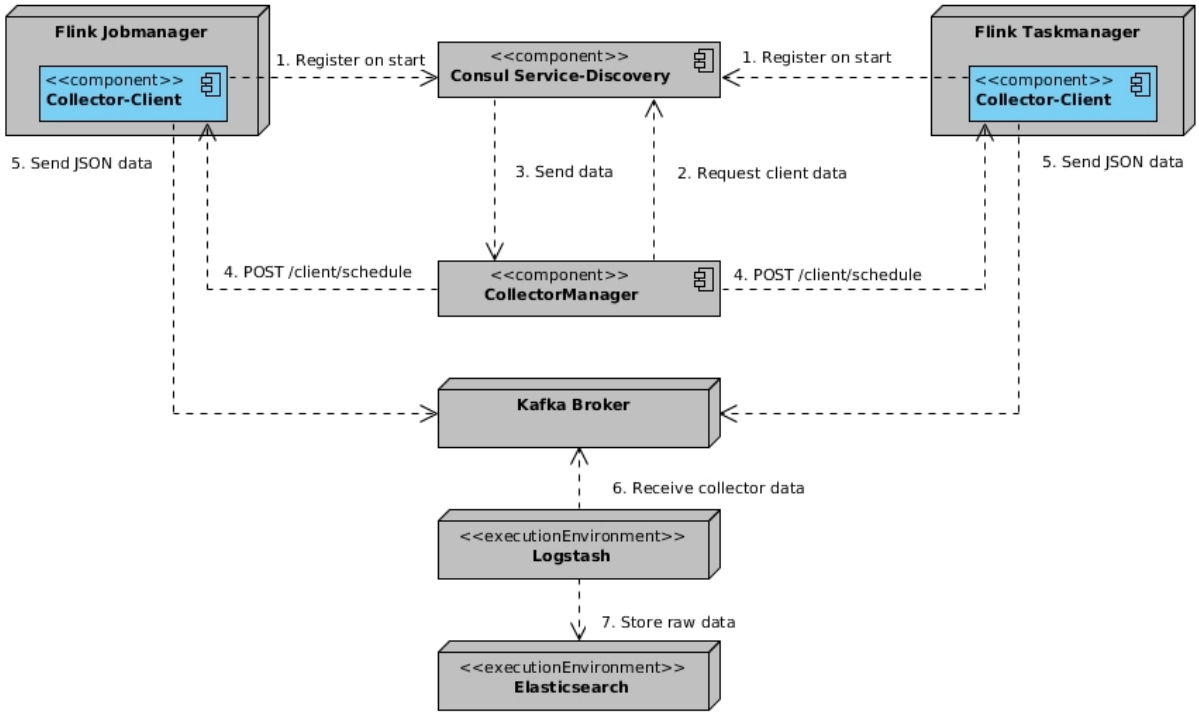

Figure A.11: Component diagram

A.1.5 Deployment diagram

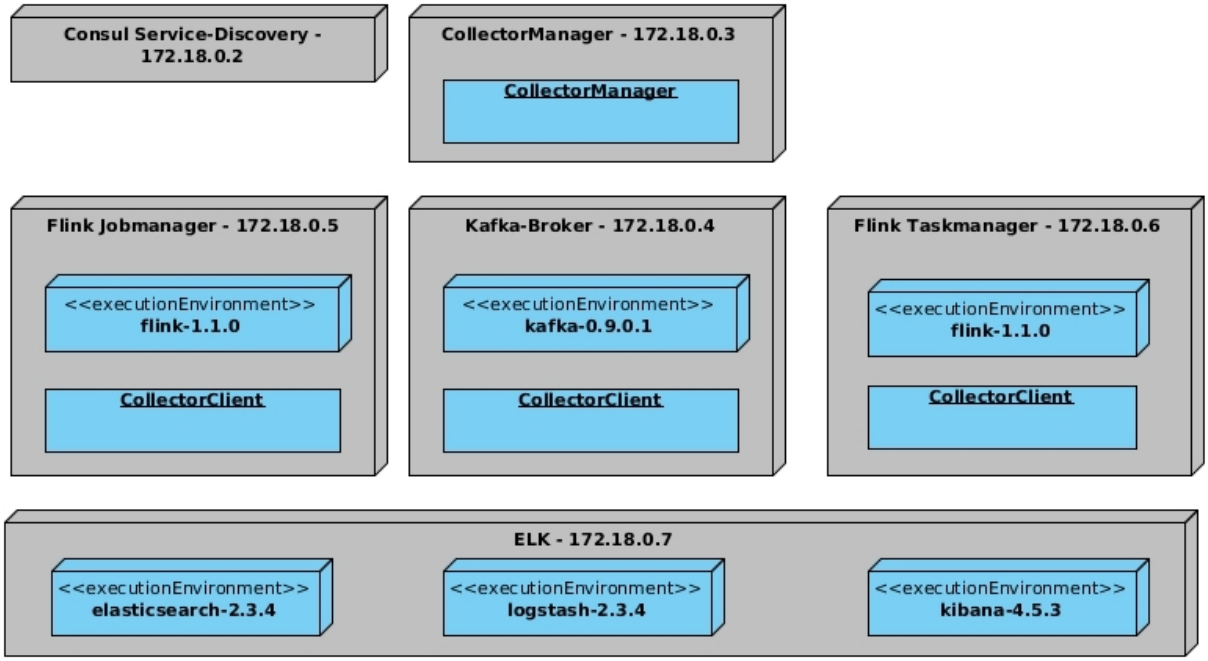

Figure A.12: Deployment diagram

A.2 Tabelle

A.3 Screenshot

A.4 Graph

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Masterarbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel verfasst habe. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

Stadt, den xx.xx.xxxx

Max Mustermann