wegweisend digital
T-SYSTEMS MULTIMEDIA SOLUTIONS

MODERNES LOGGING
DATENSAMMELN OHNE REUE

Bert Radke

Marco Grunert

T-Systems Multimedia Solutions GmbH

```
System.out.println("Customer " + 637208 + " not found");
```

VIELEN DANK FÜR EURE AUFMERKSAMKEIT!

FRAGEN? ANMERKUNGEN?

```
System.out.println("Customer " + 637208 + " not found");
```

REICHT DAS WIRKLICH?

WARUM SAMMELN WIR DATEN?

- Zustand von Anwendungen überwachen
- Abläufe nachvollziehen
- Entwickler unterstützen
- Performance messen

FÜR WEN SAMMELN WIR DATEN?

- Entwickler
- Betrieb
- Anwender

WAS WOLLEN WIR EIGENTLICH?

- Trennung von Content und Präsentation
- Steuerung durch Konfiguration
  - Was wird geloggt?
  - Wohin wird geloggt?
  - Wie wird Ausgabe formatiert?

WAS VERWENDEN WIR HEUTZUTAGE

- JUL (java.util.logging)
- Apache Java Commons Logging
- Log4J (Version 1 und 2)
- LOGBack

GRUNDLEGENDE KONZEPTE

- Log Level
- Logger
- Appender
- Layout
- Encoder

LOG LEVEL

- ERROR
- WARNING
- INFO
- DEBUG
- TRACE

LOGGER

- Hierarchie

```
"org.apache.commons.io.FileUtils"
"org.apache.commons.io"
"org.apache.commons"
```

- immer genau einem Log level zugewiesen
- mindestens einem Appender zugewiesen

APPENDER, AUSGABEZIEL

- File
- Database
- Socket
- Console
- JMS
- Syslog
- Mail
- SNMP
- ...

LAYOUT

- PatternLayout
- JsonLayout
- HtmlLayout
- XmlLayout
- CSV, SyslogLayout, ...

ENCODER

- Flexibler als Layouts
- können an Layouts delegieren / wrappen

```java
package com.tsmms.jug.datensammeln;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class LogDemo {
    static final Logger logger =
                LoggerFactory.getLogger(LogDemo.class);

    public static void main(String[] args) {
        logger.info("application {} started", args[0]);
        Alice alice = new Alice(23, "Naive Project Manager");
        Bob bob = new Bob(42, "Grumpy IT Guy");
    }
}
```

```
21:22:02.458 - [Main] - INFO - c.t.j.d.LogDemo - application Demo started
```

BEISPIEL KONFIGURATION

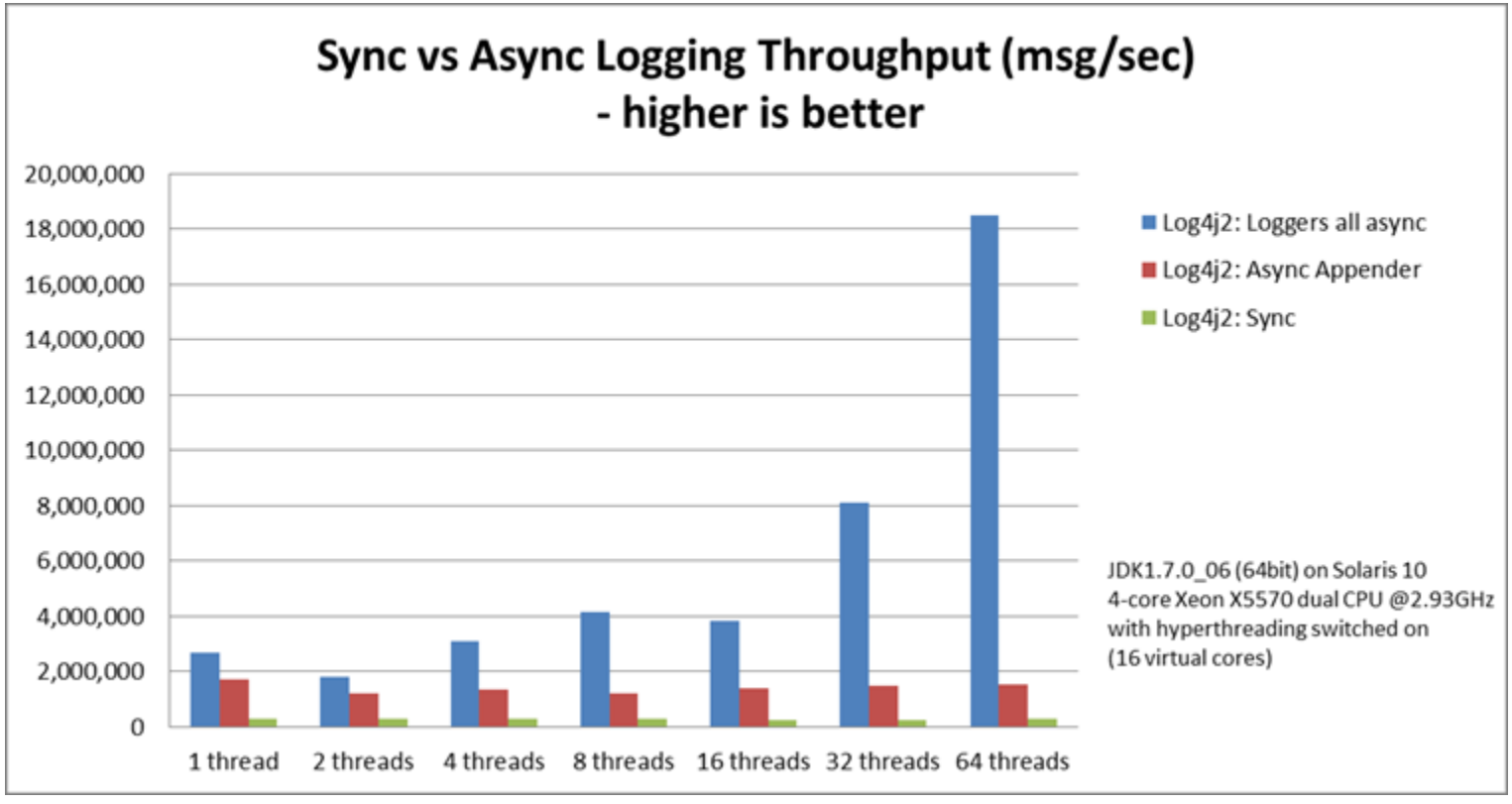
```
<configuration status="INFO" name="example-config">
    <properties>
        <property name="LOG_DIR">${sys:web.root}/logs</property>
        <property name="ARCHIVE">${LOG_DIR}/archive</property>
        <property name="PATTERN">%d{yyy-MM-dd HH:mm:ss.SSS} - [%t] - %-5level - %logger{36} - %msg%n</prop
        <property name="ARCHIVE-PATTERN">${ARCHIVE}/example.log.%d{yyyy-MM-dd}.gz</property>
    </properties>
    <appenders>
        <console name="STDOUT" target="SYSTEM_OUT">
            <patternlayout pattern="${PATTERN}">
        </patternlayout></console>
        <rollingfile name="fileWriter" filename="${LOG_DIR}/example.log" filepattern="${ARCHIVE-PATTERN}">
            <patternlayout pattern="${PATTERN}">
            <timebasedtriggeringpolicy>
        </timebasedtriggeringpolicy></patternlayout></rollingfile>
    </appenders>
    <loggers>
        <root level="ERROR">
            <appenderref ref="fileWriter">
        </appenderref></root>
        <logger name="com.tsmms.jug.some.sub" level="TRACE" additivity="false">
            <appenderref ref="STDOUT">
        </appenderref></logger>
        <logger name="com.tsmms.jug" level="INFO" additivity="true">
    </logger></loggers>
</configuration>
```

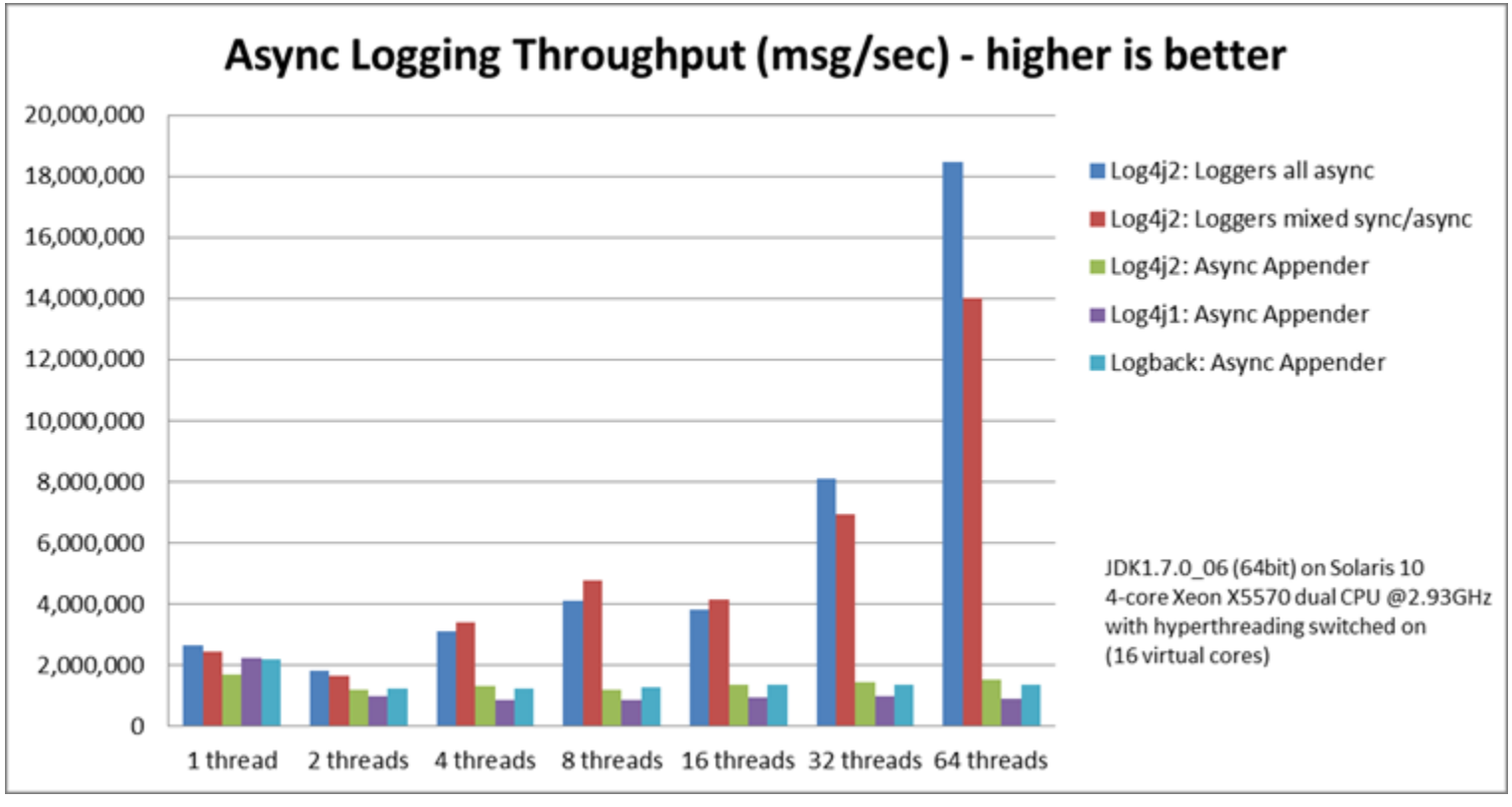
ADVANCED FEATURES

- Asynchrone Logger
- Mapped Diagnostic Context

ASYNCHRONE LOGGER / APPENDER

- Entkopplung Sender und Empfänger
- Log4j 1 und LOGBack: DQueue
- Log4j 2 LMAX Disruptors


Sync vs Async Logging Throughput (msg/sec)
- higher is better
20,000,000
18,000,000
16,000,000
14,000,000
12,000,000
10,000,000
8,000,000
6,000,000
4,000,000
2,000,000
0
1 thread
2 threads
4 threads
8 threads
16 threads
32 threads
64 threads
Log4j2: Loggers all async
Log4j2: Async Appender
Log4j2: Sync
JDK1.7.0_06 (64bit) on Solaris 10
4-core Xeon X5570 dual CPU @2.93GHz
with hyperthreading switched on
(16 virtual cores)


Async Logging Throughput (msg/sec) - higher is better
20,000,000
18,000,000
16,000,000
14,000,000
12,000,000
10,000,000
8,000,000
6,000,000
4,000,000
2,000,000
0
1 thread
2 threads
4 threads
8 threads
16 threads
32 threads
64 threads
Log4j2: Loggers all async
Log4j2: Loggers mixed sync/async
Log4j2: Async Appender
Log4j1: Async Appender
Logback: Async Appender
JDK1.7.0_06 (64bit) on Solaris 10
4-core Xeon X5570 dual CPU @2.93GHz
with hyperthreading switched on
(16 virtual cores)

MAPPED DIAGNOSTIC CONTEXT

- Erweiterung um spezielle Informationen
- Zugriff über X-Attribut
- als eigenständiges Objekt in Ausgabe verwendbar

Logging

```
MDC.put("java.vm.name", System.getProperty("java.vm.name"));
```

Layout

```
log4j.appender.stdout.layout.ConversionPattern=
    ... %-5p %c{1}:%L - %m - JVM %X{java.vm.version}%n
```

Ausgabe

```
... INFO  LogDemo:18 - application started - JVM 25.65-b01
```

WELCHES FRAMEWORK?

- JUL und JCL nicht mehr aktuell und keine Weiterentwicklung.
- Vorteil JUL: keine zusätzliche Abhängigkeiten.
- Entscheidung zwischen SLF4J/Log4j 2 und SLF4J/LOGback.

LESSONS LEARNED/BEST PRACTICE

- Verwendung der SLF4J API im eigenen Code.
- Loggingframework wird zur Laufzeit durch Classpath bestimmt.
- Fremdbibliotheken haben häufig feste Abhängigkeit zu bestimmmten Framework.
- Auswahl per Build-Tool.
- Ersetzen von JUL oder JCL durch SLF4J - Stubbs.

```
<dependency>
    <groupid>org.springframework</groupid>
    <artifactid>spring-core</artifactid>
    <version>${spring.version}</version>
    <exclusions>
        <exclusion>
          <groupid>commons-logging</groupid>
          <artifactid>commons-logging</artifactid>
        </exclusion>
    </exclusions>
</dependency>
<dependency>
    <groupid>org.slf4j</groupid>
    <artifactid>jcl-over-slf4j</artifactid>
</dependency>
```

BEST PRACTICE

- Was loggen wir?
- Auf welchem Level?
- Abstimmung mit Betrieb, NFR?
- Systemgrenzen
- Aufrufzeiten
- Exceptions, nur einmalig
- Vorgaben festhalten
- einheitliches Format / Layout

PERFORMANCE

- Kosten für .debug(), .info(), ...
- String Konkatentation vermeiden -> varargs
- Level Prüfung im eigenen Code selten sinnvoll

```
if(logger.isDebug()) {
    logger.debug("found customer: {}" ,customer);
}
```

- PatternLayout vermeiden von %C , %F, %L und %M da aufwendig in Bestimmung

REICHT DAS?

- Wie werden die Daten weiterverarbeitet?
- Wie gehen wir mit mehreren Datenquellen um?
- Wie finden wir notwendige Informationen?
- Haben wir die richtigen Informationen erfasst?

WIE WERDEN DATEN WEITERVERARBEITET?

- zentrale Erfassung in größeren Systemen
- Analyse der Daten durch unterschiedliche Personenkreise
- Einbindung in zentrale Nutzerberechtigung
- ...

WIE GEHEN WIR MIT MEHREREN DATENQUELLEN UM?

- Systeme sind üblicherweise verteilt
- eindeutige Identifier notwendig
- zentrale Zeitquelle
- ...

WIE FINDEN WIR NOTWENDIGE INFORMATIONEN?

- Suche in Datenbeständen notwendig.
- Identfier müssen über mehrere Datenquellen transportiert werden
- Arbeit in Textdateien mit regulären Ausdrücken
- Suche ist nicht performant
- ...

HABEN WIR DIE RICHTIGEN INFORMATIONEN ERFASST?

- Suche abhängig von Fragestellung
- Fehler nicht vorher bekannt
- Nutzerverhalten nicht vorher bekannt
- Auswirkungen auf Änderungen müssen nachvollzogen werden
- ...

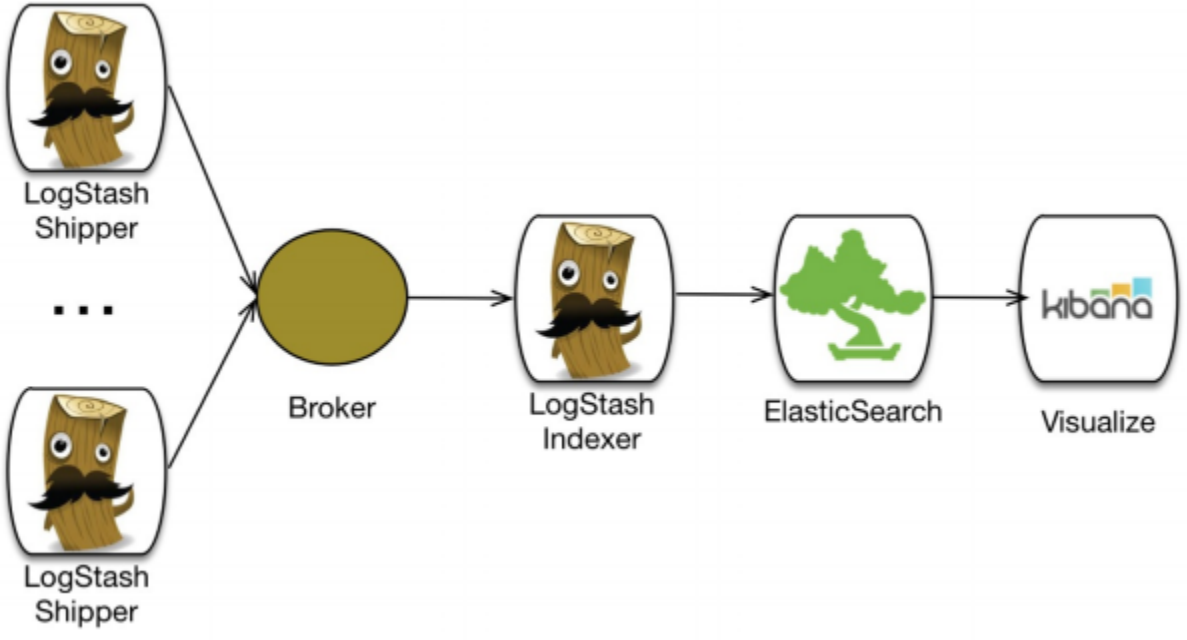
LOGGING WEITER GEDACHT

- Zentraler Datastore
- Strukturierung der Daten
- Aggregation von Daten
- Performanter Zugriff

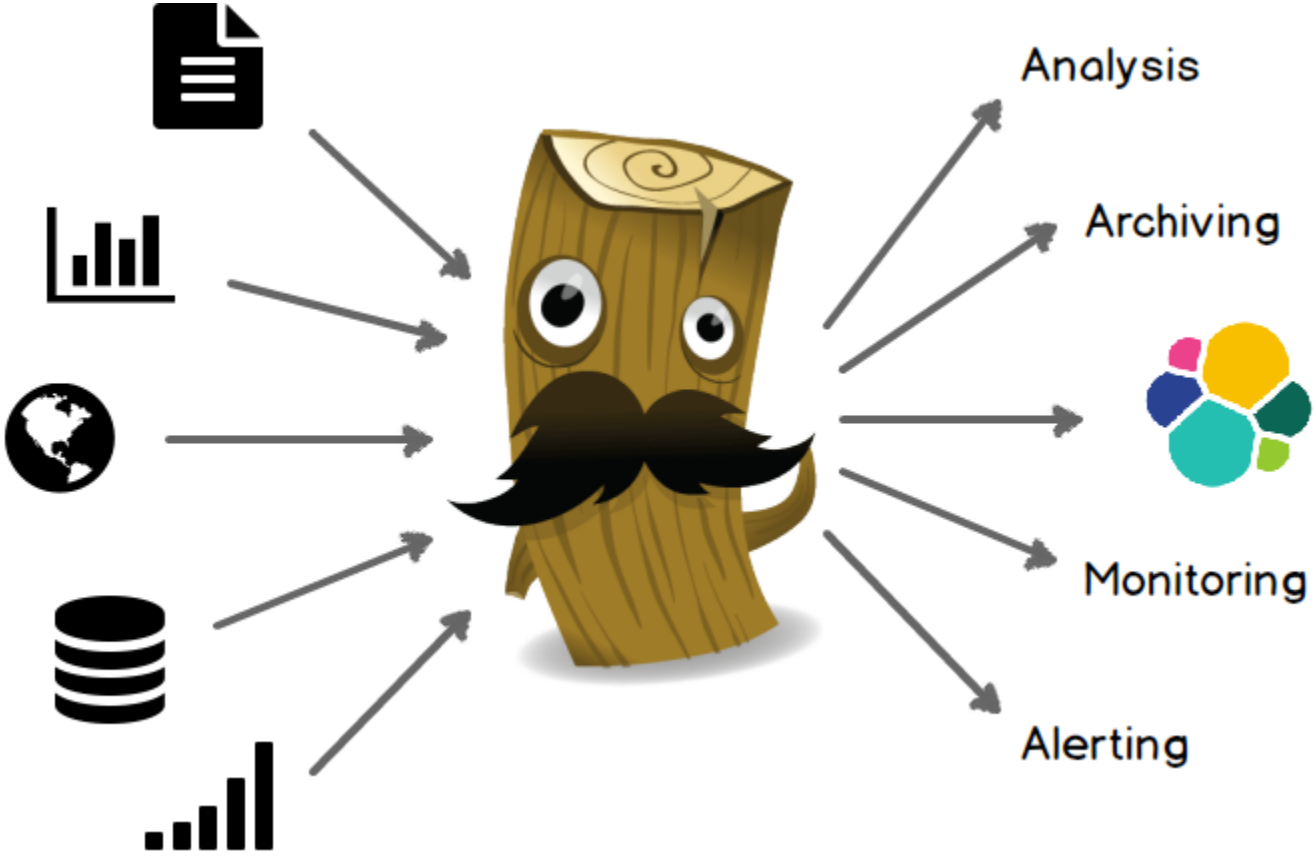
ZENTRALE LOGGING-SERVER

- passender Logging-Appender + Datenbank
- Eigenbau
- Splunk
- **E**lastic Search - **L**ogstack - **K**ibana
- ...

ELK STACK


LogStash
Shipper
...
LogStash
Shipper
Broker
LogStash
Indexer
ElasticSearch
kibana
Visualize

LOGSTASH



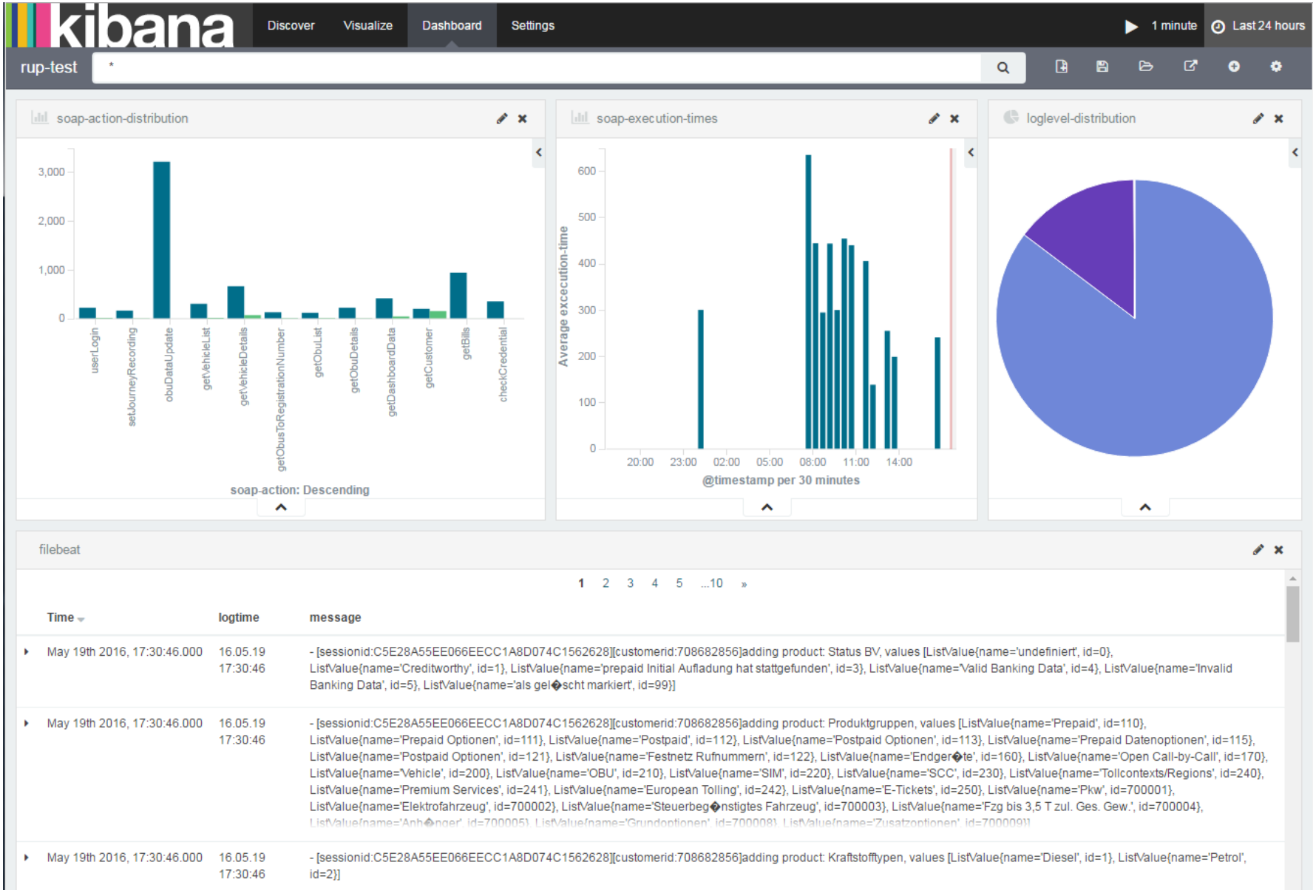
LOGSTASH

- Data Collection Engine
- Datenaggregation
- Verarbeitung verschiedener Quellen
- Normierung
- https://www.elastic.co/guide/en/logstash/current/introductio

ELASTICSEARCH

- Speichern
- Suchen
- Analysieren
- https://www.elastic.co/guide/en/elasticsearch/reference/curr
  started.html

KIBANA


kibana
Discover
Visualize
Dashboard
Settings
1 minute
Last 24 hours
rup-test
*
soap-action-distribution
3,000
2,000
1,000
0
userLogin
setJourneyRecording
obuDataUpdate
getVehicleList
getVehicleDetails
getObusToRegistrationNumber
getObuList
getObuDetails
getDashboardData
getCustomer
getBills
checkCredential
soap-action: Descending
soap-execution-times
600
500
400
300
200
100
0
Average excecution-time
20:00
23:00
02:00
05:00
08:00
11:00
14:00
@timestamp per 30 minutes
loglevel-distribution
filebeat
1 2 3 4 5 ...10 »
Time
logtime
message
May 19th 2016, 17:30:46.000
16.05.19 17:30:46
- [sessionid:C5E28A55EE066EECC1A8D074C1562628][customerid:708682856]adding product: Status BV, values [ListValue{name='undefiniert', id=0}, ListValue{name='Creditworthy', id=1}, ListValue{name='prepaid Initial Aufladung hat stattgefunden', id=3}, ListValue{name='Valid Banking Data', id=4}, ListValue{name='Invalid Banking Data', id=5}, ListValue{name='als gel�scht markiert', id=99}]
May 19th 2016, 17:30:46.000
16.05.19 17:30:46
- [sessionid:C5E28A55EE066EECC1A8D074C1562628][customerid:708682856]adding product: Produktgruppen, values [ListValue{name='Prepaid', id=110}, ListValue{name='Prepaid Optionen', id=111}, ListValue{name='Postpaid', id=112}, ListValue{name='Postpaid Optionen', id=113}, ListValue{name='Prepaid Datenoptionen', id=115}, ListValue{name='Postpaid Optionen', id=121}, ListValue{name='Festnetz Rufnummern', id=122}, ListValue{name='Endger�te', id=160}, ListValue{name='Open Call-by-Call', id=170}, ListValue{name='Vehicle', id=200}, ListValue{name='OBU', id=210}, ListValue{name='SIM', id=220}, ListValue{name='SCC', id=230}, ListValue{name='Tollcontexts/Regions', id=240}, ListValue{name='Premium Services', id=241}, ListValue{name='European Tolling', id=242}, ListValue{name='E-Tickets', id=250}, ListValue{name='Pkw', id=700001}, ListValue{name='Elektrofahrzeug', id=700002}, ListValue{name='Steuerbeg�nstigtes Fahrzeug', id=700003}, ListValue{name='Fzg bis 3,5 T zul. Ges. Gew.', id=700004},
May 19th 2016, 17:30:46.000
16.05.19 17:30:46
- [sessionid:C5E28A55EE066EECC1A8D074C1562628][customerid:708682856]adding product: Kraftstofftypen, values [ListValue{name='Diesel', id=1}, ListValue{name='Petrol', id=2}]

KIBANA

- Auswertung
- Visualisierung
- Interaktion mit Elasticsearch
- https://www.elastic.co/guide/en/kibana/current/introduction

UND NUN IHR!

BERT RADKE

(Google Plus) Bert Radke / bert.radke@t-systems.com

MARCO GRUNERT

(Twitter) @magomi / marco.grunert@t-systems.com

VIELEN DANK FÜR EURE AUFMERKSAMKEIT!

JETZT ABER WIRKLICH.